On a Wildlife Tracking and Telemetry System: A Wireless Network Approach

Andrew Markham
BSc(Eng) in Electrical Engineering, 2004

A thesis submitted to the Department of Electrical Engineering, University of Cape Town, in fulfilment of the requirements for the degree of Doctor of Philosophy.

August 2008

Declaration

I declare that this dissertation is my own, unaided work. It is being submitted for the degree of Doctor of Philosophy at the University of Cape Town. It has not been submitted before for any degree or examination in any other university.

Signature of Author ....

Department of Electrical Engineering

University of Cape Town

August 2008

Abstract

Existing approaches to monitoring wildlife with wireless networks have not taken into account the vast heterogeneity inherent in the Animal Kingdom, especially with respect to bodyweight (and hence tag carrying capacity). This has resulted in a single design of tag which is only suitable for placement on larger animals. Thus, with existing technology, small animals cannot be monitored using the wireless network system.

Motivated by the diversity of animals, a hybrid wildlife tracking system, EcoLocate, is proposed, with lightweight VHF-like tags and high performance GPS enabled tags, bound by a common wireless network design. Tags transfer information amongst one another in a multi-hop store-and-forward fashion, and can also monitor the presence of one another, enabling social behaviour studies to be conducted. Information can be gathered from any sensor variable of interest (such as temperature, water level, activity and so on) and forwarded through the network, thus leading to more effective game reserve monitoring. Six classes of tracking tags are presented, varying in weight and functionality, but derived from a common set of code, which facilitates modular tag design and deployment. The link between the tags means that tags can dynamically choose their class based on their remaining energy, prolonging lifetime in the network at the cost of a reduction in function. Lightweight, low functionality tags (that can be placed on small animals) use the capabilities of heavier, high functionality devices (placed on larger animals) to transfer their information. EcoLocate is a modular approach to animal tracking and sensing and it is shown how the same common technology can be used for diverse studies, from simple VHF-like activity research to full social and behavioural research using wireless networks to relay data to the end user. The network is not restricted to only tracking animals – environmental variables, people and vehicles can all be monitored, allowing for rich wildlife tracking studies.

To transfer the obtained data effectively through resource diverse nodes, a network protocol, termed the Adaptive Social Hierarchy (ASH) was designed that ranks nodes according to their resources, such as energy or connectivity. ASH provides a scalable and adaptable method for nodes to discover the role within the network, inspired by the way animals form linear dominance hierarchies through dyadic (pairwise) interactions. Three different methods of forming the social hierarchy are presented. In the first method, pairwise ASH, pairs of nodes exchange their attributes and their estimates of rank in a two-way exchange. Although this is a simple method of forming the hierarchy, it does not take advantage of the broadcast nature of the radio channel. In light of this, a one-way method of updating ranks is proposed and shown to be able to estimate the node ranks faster than pairwise ASH, due to multiple nodes receiving the same beacon. However, both methods are unable to form an accurate social hierarchy in a stationary network, due to a limited visibility horizon. It is shown how to extend the horizon by creating pseudo-connections between unconnected nodes, using an agent based approach. Simulation results are presented that demonstrate how the ASH concept can be used as a network underlay to enhance existing protocols or how it can form a cross-layer protocol in its own right. ASH is simple, scalable and has a negligible load on the network as ASH data piggybacks on top of existing network discovery packets.

The focus is shifted from the network design to considering how to better schedule location fixes for power hungry GPS receivers. Existing wildlife tracking collars acquire fixes at constant time intervals. This leads to undersampling of high speed motion, and multiple redundant fixes when the animal is stationary. Uniform distance sampling of GPS locations is thus proposed. A low power, neck mounted, accelerometer is used to capture brief acceleration snapshots. An adaptive model, relating the snapshots to host speed is trained when the GPS unit is active. When the GPS receiver is powered down, the model is used to predict the speed of the host, and thus schedule GPS fixes at uniform distance intervals. The proposed scheme is implemented on low power 8 bit microcontrollers, and demonstrates that it is able to automatically learn the habits of the host animal or person. This technique can reduce collar power consumption by as much as 50%, whilst generating more accurate traces than uniform time sampling, as well as creating a detailed speed-time profile as a byproduct of the speed estimation process.

This work has considered the problem of tracking and monitoring wild animals and their interaction and dependency on their environment. A scalable, modular and adaptable solution has been proposed, that allows small and large animals to be monitored using the same system and which automatically sends data through the network to the end user. Thus, this work has the potential to greatly enhance the understanding of animal behaviour, by providing large amounts of inter-related sensor data with minimal human input.

Acknowledgements

I am indebted to a number of people for their guidance, help and advice during the course of my PhD.

First and foremost, I would like to thank my supervisor, Dr Andrew Wilkinson, for his support and friendship over these years, from when I first knocked on his door and asked him to supervise my undergraduate thesis. Dr Wilkinson has the ability to cut past the fluff and go to the core of the problem, and this has benefited me immensely. He has also given me academic freedom to pursue whatever path I thought best, with nudges now and then when I was getting seriously off course.

Dr Ken Stratford of Ongava Research Centre has been a wonderful sounding board, allowing me to bounce ideas off someone who is actually working in the field. Without his input, I doubt whether my work would be physically realisable and would be some blue sky abstraction. He also generously sponsored the purchase of GPS receivers and microcontrollers to build and test more toys.

I would also like to acknowledge a debt of gratitude to Prof. Braae for donating his 'paperclip' fund together with Andrew so I could attend a conference in Italy and for reading through some of my papers. Prof. Braae also put me in touch with Ken, which helped immensely.

Sam Ginsberg has been a great help too – it is helpful having the guru of microcontrollers just round the corner. He is always ready to shed some light on peculiar problems (such as MOSFETs misbehaving on remote islands). Sam also showed me how to use the milling machine, so I could make toys to my heart's content.

My lab mates, Kush, Kent and Rachel have been a great help in keeping disasters at bay and I am going to miss the deliberation over where to go for lunch every day. My friends have been very patient with me through this all, and I thank them for helping keeping me sane. In particular, thanks to Shannon for reading through the draft and giving me useful comments from the other side of the fence.

I would like to thank my family, both old and new for their help during the course of my research. Phil showed me the black art of aluminium welding and he welded up the aluminium frames used for the penguin logger – for this I am very grateful. The Cunninghams have welcomed me into their home and fed and watered me and I would like to thank them for their generosity. I would like to thank Tony for drilling home modularity many years ago and for his constant stream of jokes into my inbox. Thanks to my little sister for reading through my dissertation – it always helps having feedback from MIT. David has kept me going with his flippant comments about the state of the dissertation. I would like to thank my parents from the bottom of my heart for their support and love and for giving me the greatest gift of all – a passion for learning and discovery.

Finally, this would not have been possible without the light of my life by my side. Kathy has helped so much in getting this all done, from emergency missions to Dassen to cups of tea to keep me going. She has listened to me rattle on about hierarchies, ranks and nodes with infinite patience. Thank you for your unwavering love and support through this all.

Contents

List of Figures

List of Tables

Nomenclature

Definitions of terms and abbreviations used in this thesis:

ADC — Analog to Digital Converter

ATDA — Adaptive Threshold Distance Activation

ASH — Adaptive Social Hierarchy

DIP — Dual Inline Package

GPS – Global Positioning System

GLS — Global Location Service

GPRS — General Packet Radio Service – GSM protocol

GSM — Cellular network standard

LMS — Least Mean Squares

MAC — Medium Access and Control

PDR – Pedestrian Dead Reckoning

QFP — Quad Flatpack IC package

QFN — Quad Flatpack No-leads IC package

RISC — Reduced Instruction Set Computer

RF — Radio Frequency

RFID — Radio Frequency Identification

RLS — Recursive Least Squares

RS-232 — A simple asynchronous serial interface protocol

RTC — Real Time Clock

SD — Secure Digital (a type of memory card)

SPI — Serial Peripheral Interface

SMS — Short Message Service - GSM protocol

SOIC — Small Outline IC package

USART — Universal Synchronous/Asynchronous Receiver/Transmitter

VHF — Very High Frequency (Radio band, in the region of 30-300MHz)

List of symbols

Definitions of symbols used in this thesis:

Symbols used in Chapter 3

$\bar{M}$ — Average current consumption of a Marker class node

$\bar{S}$ — Average current consumption of a Spotter class node

I — Length of beacon interval

L — Average time between spotting windows

i_T — current consumed whilst transmitting a beacon

i_R — current consumed whilst receiving beacons

i_S — current consumed in sleep mode

t_T — time taken to transmit a beacon

D_r — receiver duty cycle

N — number of nodes

$\bar{N_R}$ — number of nodes correctly heard in a beacon window

Symbols used in Chapter 4

A — A node's attribute

E — Estimated rank (on [0;1] domain)

$\hat{R}$ — Correct rank (normalized to [0;1])

L — Level of node in the hierarchy

N — Number of nodes in the network

N_s — Number of states in ranking graph

τ — Kendall correlation coefficient

ϵ — Error threshold

e_{max} — Maximum absolute rank error

γ_e — Euler-Mascheroni constant = 0.57721

T_{max} — Maximum number of meetings required for system to converge

D — Vector of dwell times in each state

S_N — Reduced state matrix

$T_{\pm\epsilon}$ — Number of meetings required to achieve a maximum error tolerance of ϵ

$t_{\pm\epsilon}$ — Time taken to achieve a maximum error tolerance of ϵ

α — Reinforcement parameter (pairwise ASH)

W — Win total (one-way ASH)

M — Total number of observed nodes (one-way ASH)

B — Limit on the maximum number of nodes (one-way ASH)

G — Number of transmitted agents (agent ASH)

C — Size of cache (agent ASH)

δ — Update rate (agent ASH)

γ_d — Reinforcement parameter for domination (agent ASH)

γ_s — Reinforcement parameter for submission (agent ASH)

γ_0 — Minimum value of reinforcement parameter (agent ASH)

γ_{max} — Maximum value of reinforcement parameter (agent ASH)

ζ — Reinforcement parameter update rate (agent ASH)

U — Radio range radius

D_k — Traffic density at level k of hierarchy

λ — Message generation rate

p_L — Link probability

β — Reinforcement rate for connectivity updates

η — Age rate in connectivity updates

Symbols used in Chapter 5

$y[n]$ — Predicted speed

$d[n]$ — Desired (actual) speed

$e[n]$ — Speed error

$x[n]$ — Parametric inputs to model

c[n] — Vector of model coefficients

β — Discard factor for GPS filtering window

M — Number of samples in GPS filtering window

ρ — RLS forgetting factor

Φ^{-1} — Inverse autocorrelation matrix

D_k — Estimated cumulative distance travelled

δ — Innovation parameter for cumulative distance

R — Desired fix rate

1 Introduction

This dissertation describes a new approach to wildlife tracking using heterogeneous wireless networks. In particular, the diversity of the Animal Kingdom is exploited in order that both small and large animals can be tracked and monitored using the same system. This research presents the design of a system that can acquire information from a wide range of sensors (not necessarily attached to animals) and deliver it to an end user. The thread that runs through this thesis is that of adaption to provide resilience in an uncertain and dynamic environment.

This chapter provides an overview of the thesis. Firstly, the need for this research is discussed in Section 1.1. Next the scope of the problem and its design space is considered in Section 1.2. Following this, the main contributions are outlined in Section 1.3. Lastly an overview of the structure of the dissertation is provided in Section 1.4.

1.1 Background

Tracking Modalities

To determine the behaviour and location of wild animals, they need to be monitored in some way. Typically, to be able to detect animals remotely, an electronic tracking device (or tag) is attached to the animal.

The first such method that was used was VHF tracking, first used in 1963 [1]. A tracking device placed on the animal periodically emits a radio signal. The position of the animal can be determined using triangulation. Triangulation involves taking bearings (heading angle relative to North) to the tracking device from multiple locations – the position of the animal can be deduced to be at the intersection of two or more bearings. VHF tags are light and inexpensive (ca. U$100 each [2, 3, 4, 5, 6, 7]), but the tracking process is labour intensive which is a significant contributor to the cost of the study. As the researcher needs to be in the field to obtain tracking data, there is the possibility of bias introduced through habitat disturbance [8]. More recent systems alter the pulse rate of the VHF signal in relation to some measured parameter, such as temperature [3, 5]. Thus, not only can an animal's location be determined using VHF tracking, but they can also relay telemetric (sensor) data. VHF tracking is suitable for monitoring a wide range of species, from small to large, as the tags can be made very small (the lightest tag currently available is 0.2 g [9]).

Space based tracking, in particular using the ARGOS system (launched in 1978 [10]), allows researchers to automatically determine the position of animals anywhere in the world [11]. This system is similar to VHF tracking, where the animal is equipped with a transmitting device. However, unlike VHF tracking, position is inferred from the Doppler shift of the carrier wave received at the satellite [10]. From this data, approximate position can be determined. As the signal has to be strong enough to be received in space, satellite tracking devices are consequently larger than VHF tags (the smallest tag available is 15 g and solar powered [12]). In addition, ARGOS tags are expensive (ca. U$3000 each [12]) and annual data processing fees have to be paid (ca. U$3000–U$5000 [10]). Location accuracy can also be quite poor, often with errors in the region of a few hundred metres [10]. However, satellite tracking allows animals to be located globally and the position to be known within a few hours of reception at the satellite, making them especially useful for migratory species. Small amounts of telemetry data can also be uploaded to the satellite.

The third method, which has revolutionized the field of wildlife tracking by providing excellent spatial accuracy (typically < 5 m) with a high sampling rate (up to 4 Hz) is GPS tracking, first used for wildlife tracking in 1994 [5]. Unlike the two prior methods, the animal is equipped with a receiver which is used to determine the animal's position by determining the time-of-flight from multiple satellites. However, the power consumption of a GPS receiver results in a large tag for long deployments (approximately 500 g for a tracking device that can take 24 fixes a day for a year [5]). This precludes their use on smaller animals. In addition, as the locations are determined by the tracking device itself, they must somehow be transferred to the end user. One method is to use an archival tag, where data is stored on board for retrieval at the end of the study. However, loss or destruction of the tracking device will result in complete loss of the dataset. Thus, GPS tracking devices have been equipped with wireless communications links (such as UHF modem, GSM or satellite upload [13, 2, 3, 5, 7]). All these options, whilst providing remote downloading, add size and expense to the tracking device. In particular, the UHF modem option requires each animal to be located to download the data – this can be time consuming and difficult [5].

Wireless networks for wildlife tracking

ZebraNet (deployed in 2004) was proposed in order to minimize the effort involved in obtaining animal tracking data [14, 15]. Each tracking device was equipped with a radio transceiver, which allowed tags not only to transmit but also to receive data. Collars shared tracking data with one another, forming a wireless network. Thus a researcher can obtain data from many animal tracking collars, by downloading information from only a few. ZebraNet is an excellent proof of concept – that wireless networks can be used 'in the wild' to transfer information. It also pioneered in the actual deployment of an animal based tracking collar, leading to a spate of related work in addressing compression [16], middleware design [17] and hardware issues [18]. ZebraNet is specifically designed for attachment on zebras [18]. Whilst the collars can easily be adapted to fit on other large mammals, the size and weight (138 g for the latest devices [18]) of the collars restricts their usage on smaller animals. This is as a result of the 5% guideline which recommends that collar weight should not exceed 5% of the bodyweight of the host [19]. Another recent wireless network for wildlife tracking is TurtleNet (2006), which is designed for monitoring the positions of turtles [20].

1.2 Research Objectives and Solutions

Existing research into wireless networks for wildlife tracking has resulted in homogeneous solutions. This is the 'one size fits all' approach, where a single type of tracking device has been designed. This has segmented the solution space into animals which can be tracked using wireless networks and those that cannot, due to weight restrictions placed on the tracking collar.

The objective of my research is to design a single wireless network based system that can be used to track and monitor both small and large animals. I argue that the vast diversity in the Animal Kingdom, especially with respect to bodyweight, should not be viewed as a hindrance, but rather something to be exploited. My philosophy is that devices with low functionality (due to weight or cost restrictions) should use the capabilities of more complex devices in order to result in a powerful network solution.

The research objectives of this dissertation are:

- To design a system that is able to monitor a wide variety of animals, in a typical gamepark environment (areas of tens to hundreds of square kilometres).
- To make the system scalable, so it works well both on small and large numbers of nodes.

- To make the system adaptable to device insertions and removals.
- To make the system modular and flexible.
- To make the system energy conscious to maximise lifespan.
- To validate the system design with real world tests.

To address these objectives, research was undertaken in three main areas, namely in the system design of a wildlife tracking network, in the formulation of a network management protocol that can deal with large amounts of heterogeneity and in the proposal of a uniform distance sampling based approach to GPS tracking.

1.2.1 Ecolocate: A heterogeneous wireless network system for wildlife tracking

The first avenue that was considered is how to design a wireless network system that is able to monitor a wide variety of animals. The operations that are required to be undertaken in a network of this nature are first examined. Data relevant to the animal's behaviour (which is not limited to location, and can also include biophysiological parameters) needs to be **sensed**. As the network exists over a vast geographical area, node density will be typically very sparse, resulting in a (complete) lack of end-to-end connectivity. Thus, a store-and-forward approach has to be adopted. Hence, the second operation that must be undertaken in the network is that of **storage**. The data acquired from one node must then be relayed through the network, possibly through multiple hops, before it reaches the end destination. The network (termed 'EcoLocate') is designed for data gathering (as opposed to peer-to-peer transfer[1]) and thus the end destination is a gateway or basestation (of which there may be more than one). Hence the third operation is that of **sending** data. With these required operations in mind, a wireless network was designed that is able to sense a wide variety of information, store it onboard and send it to the end destination.

To achieve these aims, a modular solution is proposed. It will be shown how multiple classes of nodes can be implemented, where simpler classes have a subset of the functionality of complex devices. Thus, only one code base needs be maintained. The device class can be specified at design time, or dynamically adjusted at run time depending on energy constraints and availability. Extra functionality, such as sensing devices or memory options is implemented as plug in modules. Any variable which can be digitized

[1]In a peer-to-peer network, data is shared or flooded amongst all nodes in the network, it thus has a flat structure. In a data gathering network, a super-node or base-station collects all the information that is generated within the network and thus has a hierarchical structure.

can in theory be transferred into the network (subject to memory and bandwidth constraints). Thus, GPS locations are treated simply as another type of data, rather than requiring each node to be GPS capable. Simple specification of tags and their function, highlighting their modularity, is accomplished using a graphical representation of the system.

By equipping nodes with transceivers, they can not only transmit beacons (similar to normal VHF tracking) but also receive beacons from other nodes or tags in the network, which can be used to create contact logs. This data can be used in a number of ways. It can be used to analyse social relationships and dependencies between individuals (which need not be of the same species). The contact logs can also be used to determine visit frequency to focal points of attraction, such as waterholes, which can be used to monitor resource usage. Another use of this information is to infer the coarse position of non-GPS equipped nodes, when in range of a device with a known position. Techniques that have been used for many years in VHF tracking, such as triangulation and homing [21] can also be carried forward into this system. Thus, by using a wireless network approach, it is shown how conventional VHF tracking can be enhanced using many automatic transmitting and receiving devices. The other purpose of beacons is to update network parameters and discover nodes within range, this is elaborated on further in the network section. Based on the modular representation of the tracking devices, a graphical, block based approach to designing tracking devices is proposed, in order that the users of the technology can specify their own systems in a simple manner.

1.2.2 The Adaptive Social Hierarchy (ASH): A Network Ranking System

Due to the wide heterogeneity supported by the EcoLocate system design, a method of providing scalability of resources (such as energy or connectivity), regardless of their distribution, was required. As a motivating example, consider the two networks in Fig. 1.1, both containing three animals. For both networks, assume that the animals carry tracking collars which have an amount of battery energy that is proportional to the bodyweight of the host. In the first network (Fig. 1.1 (a)), the lioness collar has the least energy in comparison to the larger elephant collars. Thus, in this network scenario, the lioness collar should use the capabilities of the elephant collars to transfer information to the end user. It should not be active in routing tasks and should conserve its energy. Conversely, in the second network scenario (Fig. 1.1 (b)), the lioness collar has the most energy with regards to its peers. Thus, it must assume the most active role in the network and be responsible for routing information from the honey badgers to the end user [2].

This simple example demonstrates that the role of a device is not related to absolute resources or attributes, but rather their relationship relative to their peers. Essentially, what is required is a means for nodes to determine their ranking (e.g. 'best', 'good', 'poor', 'worst') within the network, based on some attribute. This cannot be specified *a priori* as wireless networks operate under dynamic conditions and need to be resilient to node insertions and removals. Central control is one option, where one node informs other nodes of their rank within the network. However, this results in a single point of failure and leads to poor scalability with increasing numbers of nodes. Instead, a method is required where each node discovers its own role based on information from obtained from its local neighbourhood.

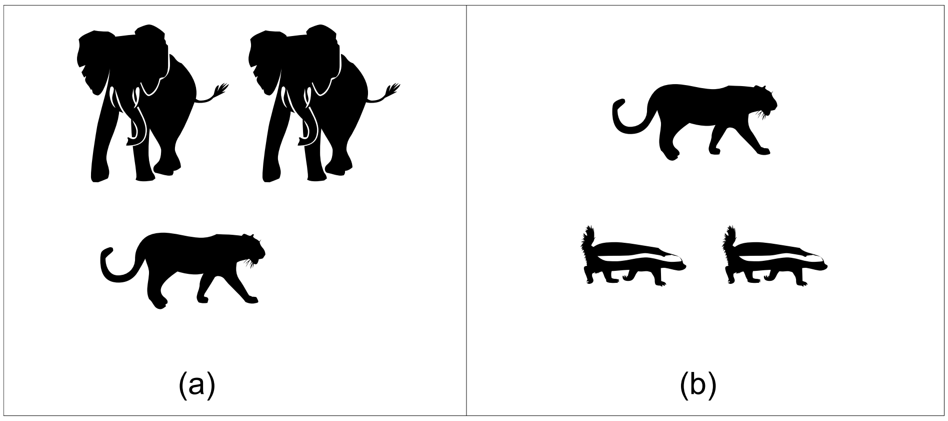


Figure 1.1: Example scenario demonstrating the need for a ranking system. Assume that animals carry tracking collars with battery reserves that are in direct proportion to their weight. In network (a), the collar on the lioness has the lowest energy in the network, whereas in network (b), it has the greatest amount of energy in the network. Thus, in (a), the lioness collar does not need to be active in network tasks like routing, whereas in (b), it will need to adopt an active profile so that the network functions well. Hence, the role of a device in a network is not dependent on the absolute amount of energy but its energy relative to its peers.

To achieve these aims, the Animal Kingdom itself is turned to for inspiration. Animals form societal groupings – these are such a common feature that collective nouns exist for different species. A pervasive structure in animal societies is a linear dominance hierarchy [22, 23, 24, 25, 26, 27]. In these structures, the fittest individual (as measured by some attribute such as bodyweight) is the super dominant or alpha member and dominates all other animals in the group. This means that the alpha individual will have preferential access to resources, such as food or the right to mate. The next individual,

[2]It must be noted that energy is not necessarily a good indicator of usefulness in routing information towards a basestation. As will be discussed in Chapter 4, any attribute (such as hop-count or remaining lifetime) can be ranked to determine a node's role within the network.

the beta member, will dominate all other individuals but be submissive to the alpha and so on. The last member of the group, the omega individual, will be dominated by all other members of the group. The relative ordering of individuals in social dominance hierarchies in natural groupings are by no means static, and alter as a result of variations such as injury or the introduction of a new member. Social hierarchies are not limited to the Animal Kingdom, and many human organizations resemble a dominance hierarchy (military ranks, academic ranks and so forth).

The idea of a linear dominance hierarchy is taken and applied to a wireless network, in order that individual nodes can determine their rank. This has been termed an *adaptive social hierarchy* (ASH) to highlight the way it adapts to changes. Energy is a prime example of an attribute to rank, but it is shown that any network variable that can be measured on a per-node basis (such as connectivity and so forth) can be cast into a ranking.

A number of methods of forming the ranking are presented, starting from a simple pairwise exchange. Assumptions about the behaviour of the nodes and restrictions on medium access are removed, resulting in an agent based approach to forming the ASH ranking. Through simulations, it is shown how nodes can determine their ranks, and adapt to changes in the rank order, using information obtained from their peers. The methods presented are lightweight, and the focus is on formulating simple rules to allow nodes to discover their own rank within the network.

It is shown how the ASH methods can be used as an underlay for existing routing protocols, helping to provide resource scalability and reduce reliance on application dependent tuning factors. It is also shown how ASH can be used to form a simple cross-layer protocol.

1.2.3 Uniform distance GPS sampling

GPS receivers, whilst providing unsurpassed location accuracy anywhere in the world, have a relatively high power consumption (in the order of 90mW [28]) and take some time to acquire a fix (best case is a few seconds, but in reality more often in the range of 10–30s [29, 18]). They are thus responsible for a large component of a tracking device's energy budget. Existing GPS enabled wildlife tracking devices acquire fixes at constant time intervals (for example, every 15 minutes [5, 3]) in order to reduce the power consumption of the tracking device and prolong its life. However, this method does not take into account the variable motion patterns of the host animal. Animals are stationary for a large proportion of time (some animals, such as lions, are only active 40% of the time [30]). Using constant time sampling, multiple fixes will be taken at the same location, expending valuable energy whilst not gaining any additional information.

Conversely, when the animal undertakes a high speed foray, such as a predation related event, constant time sampling will undersample the detail, leading to a possible loss of information.

A shift from constant time to uniform distance sampling is thus proposed. To estimate the distance travelled, the motion of the neck of the animal is sensed using a low power accelerometer. As animals have different gait patterns, a model of the relationship between the summary statistics of the acceleration snapshot and the speed of the host is constructed. The model is dynamically adapted when the GPS receiver is active, such that the collar, over time, learns the relationship between the speed of the host and the acceleration of its neck. Results from tests on humans and animals are presented, showing that the learning algorithm is able to adapt to the gait patterns of the host, even though they might be markedly different. An adaptive distance triggering threshold is proposed, for the case where the average distance travelled by an animal over a period of time is not known prior to deployment. Simulations of energy conserved by using uniform distance sampling are also presented.

1.3 Contributions of this work

Specifically, the original contributions of this research are as follows:

- The design of a wireless network based tracking solution that can be used on a wide variety of animals, where small animal collars (with limited functionality and small energy reserves) use the capabilities of large ones (with high functionality and large energy reserves) to send information through the network to the end user. The network is not restricted to animal deployment, and can be include environmental sensing and vehicle and human monitoring.
- The proposal of six different classes of tracking devices with varying degrees of complexity that are based on a unified hardware and software design. Simpler devices are implemented by limiting the functionality of more complex devices. This can be done dynamically in the field, allowing devices to alter their class based on their remaining energy reserve, leading to a longer lived network. Devices were fabricated and tested by the author in order to demonstrate the flexibility of the modular approach.
- Developing a system that allows social relationships (contacts between animals), location and proximity (presence or absence of the animal within a certain area) to be determined, without requiring that all devices are GPS enabled.

- Developing a lightweight, adaptive network management protocol that is able to handle large degrees of heterogeneity in resources, allowing a device to determine its ranking within the network (which can be fixed or mobile). This provides resource scalability and can be used as an underlay to enhance the performance of existing routing protocols.

- Proposing a novel way of acquiring GPS samples, based on the accelerometer dependent estimate of distance that the host has travelled, rather than at uniform time intervals, leading to an increase in device lifetime without loss of location information. A method that learns and adapts to the way the host moves is also developed. As a byproduct of distance estimation, a detailed speed-time profile is generated, which can be used to characterize animal behaviour.

1.4 Guide to the thesis

The remainder of this thesis is organized in the following chapters.

Chapter 2 reviews currently available tracking technologies for wildlife tracking, their uses and constraints imposed upon them. This reviews the state of the art and places this research in context of the existing body of work.

Chapter 3 presents EcoLocate, a modular approach to wildlife tracking and monitoring using wireless networks. A number of classes of tracking devices with varying degrees of functionality and energy consumption are introduced. The modular design is presented, showing how different tracking studies can be built up from simple blocks.

Chapter 4 introduces a novel network management protocol, that provides resource scalability. This is inspired by the way animals form social dominance hierarchies, such that each node is able to determine its rank within the network according to a certain attribute or attributes. A number of different methods of forming these hierarchies is discussed, with particular emphasis on scalability and rate of convergence. The use of the ranking systems to enhance the operation of existing routing protocols is shown, as well as how they can be used to form a simple cross-layer network protocol.

Chapter 5 considers how to implement uniform distance GPS sampling. Existing methods acquire GPS fixes at constant time intervals which leads to undersampling of high speed data and oversampling whilst the host is stationary. An adaptive model is used to learn the relationship between the gait of the host (as measured by a low power, neck mounted accelerometer) and its ground speed. Using this information, the GPS receiver is triggered to acquire fixes at uniform distance intervals.

Chapter 6 discusses the design of the hardware and firmware that is used to validate the approaches of the prior chapters. The focus is on modularity, both in terms of the firmware and hardware design, and it is shown how simple devices are formed by restricting the extent of the state universe.

Chapter 7 details real world testing and deployment of some of the various classes of nodes. It is shown how the same basic firmware can be adapted to perform multiple studies, with very different aims.

Chapter 8 is the concluding chapter, where the main findings of the work are summarized. Future research avenues are also proposed.

2
Taxonomy of Wildlife Tracking and Telemetry Technologies

2.1 Introduction

Wildlife tracking involves acquiring information about the behaviour of animals in their natural habitat. This information is used both for scientific and conservation purposes. The primary form of information that needs to be obtained is the location of the animal at certain points in time and this is generally referred to as **tracking** or radio–tracking [31]. Other forms of information such as physiological parameters (for example: heart-rate [32, 33]; body temperature [34]; vaginal temperature [35]) or activity (for example: head tilt [5] or defecation [36]) can be acquired and this is referred to as **telemetry** [31]. However, due to the similarities in obtaining the information, the terms are frequently used interchangeably. Essentially, tracking involves determining where an animal is, and telemetry refers to recording data that can be used to infer its activity, at certain points in time.

For the most part, due to the difficulties in locating a biological entity over large areas, the creature is augmented with a tracking device (generally electronic). There are remote methods that can be used to track and identify animals visually [37, 38, 39] and through acoustic signals [40, 41]. The drawback of these methods is that they are only suitable for some species and their detection range is limited. However, as they do not involve disturbance to the animal, they are an attractive option in the situations where they can be used.

In this review, the focus is mainly on terrestrial tracking and telemetry techniques, but reference is also made to the large body of work on marine and avian species. In the next section, constraints common to all affixed devices are examined, imposed both by the animal and the technology itself. In Section 2.3 various methods of determining the location of an animal are presented, discussing their relative merits and drawbacks. Following that, in Section 2.4, an overview of various sensing techniques and options are examined. Lastly, in Section 2.5, a summary of the chapter is provided.

2.2 Real world constraints

The decision to attach a tracking device to an animal is one which must be carefully weighed in terms of the benefits (generally to the researcher) and the drawbacks (generally to the animal in question). In light of the impact of humans on animal populations, habitat and behaviour, there is a pressing need to understand more about animal biology so that informed conservation decisions can be made [42]. Considering how long animal tracking has been undertaken (since the 1960's [1]), there are still no globally defined regulations, or even guidelines, on the restrictions on tag attachment [42]. In part, this is because it is difficult or even impossible to quantify what corresponds to baseline performance, as the very act of attaching a tracking device affects the behaviour of the host, as 'measurement affects performance' [43]. In this section, some of the factors that need to be borne in mind when deciding what (if any) type of technology suitable for tracking a certain animal species are examined. Some constraints are imposed by the animal (such as the shape of the tracking device), and others are imposed by the tracking device itself (such as the length and orientation of the antenna). Clearly, these need to be balanced, such that the device chosen provides the best possible amount of data with the least impact to the animal[1].

2.2.1 Size, weight and shape of device

A tracking device comprises the following components: the electronics, board and antenna associated with the tracking device, a source of power and a method of attachment. Together, these result in an overall device weight. A 'rule of thumb' has been proposed stating that the weight of the tracking device should not exceed 5% of the host's bodyweight in order to avoid detrimental effects to behaviour [19, 44]. Other work has suggested that this be revised downwards to 4% [45] or as low as 2% [46]. Still other research has demonstrated no adverse effects on fish when swimming when tags were as high as 12% of bodyweight [47]. Wilson and McMahon claim that a bodyweight guideline is naive, as it does not take into account the behaviour of the animal [42]. They propose that the impact of the device on the energetics of the host be considered [42]. As an example, a bird which spends more time in the air flying is more likely to be detrimentally affected by a large tag than one which flies less frequently.

[1]Discussions on the ethical implications of animal tracking can be found in [8].

Using the 5% bodyweight guideline, the allowances for the real weight of the tag result in orders of magnitude differences. For example, at one extreme, an adult, bull African Elephant (*Loxodonta africana*) can weigh 6000 kg [48], resulting in an allowable tag weight of 300 kg. On the other hand, for a small mammal such a Deer Mouse (*Peromyscus maniculatus*) which weighs 22 g [49], the tag can be a maximum of 1.1 g to satisfy the 5% bodyweight rule. This has the implication that tag weight is a critical factor in dictating the type of technology that can be deployed on small animals.

For flying and aquatic creatures, not only is the tag weight important, but also the tag shape, in particular the front cross-section profile[2] [50, 51, 52]. This is because the drag of the device, which is often attached on the animal's back, can dramatically increase the amount of energy required to move, resulting in increased foraging times and other deleterious effects [53, 54, 55]. For example, it has been shown that an antenna which weighs 0.16% of a penguin's mass resulted in an 80% increase in drag whilst swimming [56, 57]. This demonstrates that it is not only factors such as the weight of the tag that must be considered, but also the impact of the shape of the device. Studies on implanting tags into eiders [58] and penguins [59] showed that there was little or no effect of the device on animal behaviour and mortality. It was hypothesized that implants have less of an effect as they do not alter the hydrodynamic profile of the animal [58].

Protruding artefacts from the tracking device can also present issues to animal behaviour and mortality. Birds have become tangled in vegetation as a result of the antenna and harness [60, 61, 62]. In one reported case, electrocution occured as a result of a wire whip antenna coming into contact with power lines [63].

2.2.2 Methods of attaching tags

According to Mech *et al.* [64], there are five criteria that should be satisfied in order for a collar to be regarded as 'ideal', namely:

- Minimum weight.
- Minimum effect on animal.
- Maximum protection for the transmitter.
- Permanence of attachment.
- Maximum protection of transmitter from mortality related incidents.

[2]For aquatic creatures, it is also important that the tag has neutral buoyancy so as to not detrimentally affect the way they dive and swim [57].

In addition to these recommendations, White and Garrott also advise that tags should not alter the cryptic coloration of animals and that tags be tested beforehand on captive individuals [8].

These criteria can be used to help decide what method of tag attachment to use. The type of method selected must be considered very carefully, as an unsuitable method can result in death or injury to the host animal or destruction or poor performance of the tracking tag. For mammals, the tag is generally attached to a collar which is fastened around the neck of the animal [65, 21]. Collars can be made from a variety of materials such as nylon, leather or brass depending on the host species [2]. In some animals, particularly social carnivores, the collar needs to be reinforced or armoured to prevent destruction [21]. An example of a collar which was damaged by a captive lion is shown in Fig. 2.1. There are also various methods for fastening the collar, ranging from screw attachments to cable ties [2]. To retrieve collars at the end of the study (essential for archival data-loggers and desirable for expensive satellite tracking devices), collars can be fitted with drop-off or remote-release mechanisms [5, 66]. To protect the tracking device itself, it can be encased or potted in a rigid, waterproof substance such as epoxy resin [2, 6].

However, some animals, such as hedgehogs, cannot be equipped with a collar as they lack a distinct neck. Tracking devices have been successfully attached to hedgehogs by clipping an area of spines and gluing the tracking device to the animal [67]. Velcro can also be used for easy removal of the tag - one piece is glued to the animal and another strip to the tracking device [67]. Another method of attaching a tracking device to an animal is to use a backpack harness [52, 68]. Specialized structures on animals can also be used to attach the tracking device. As an example, tracking devices have been placed within the horn of a rhinoceros [69]. In marine animals, tracking devices can be attached using suction cups [70].

Where external attachment is impossible or not suitable, tags can be implanted within the animal [71, 72]. Davis *et al.* implanted radio-transmitters in the peritoneal cavity in ten beavers, as previous studies failed within a few weeks due to the destruction of tail mounted transmitters [73]. A follow up necropsy found that one animal had died as a result of adhesion between the tag and the intestinal organs, whilst there were no adverse effects for the other animals [74]. An issue with implants is that the radio range can be significantly reduced, sometimes to less than 50% of the original range [75]. In addition, there can be issues associated with the implant, such as post-operative infection and pathological complications [72]. The method of attaching the tracking device within the incision also is important, with free-floating devices showing less negative effects than those using retaining sutures in armadillos [72]. An alternative to surgical implantation is an encapsulated tag which is placed in the animal's stomach [76]. The

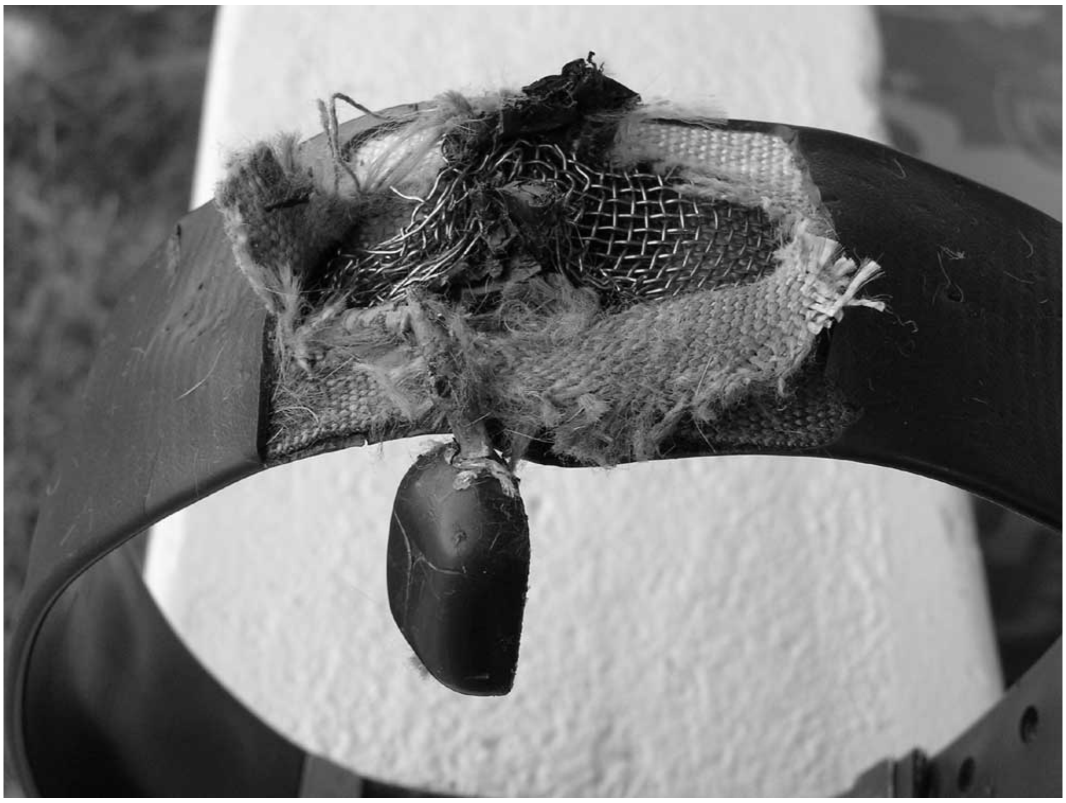


Figure 2.1: A damaged heavy duty tracking collar, showing internal construction and reinforcement with steel mesh. [Photo courtesy K. Stratford]

size of the implant must be carefully chosen so as to avoid discomfort to the animal [42].

The animal's behaviour can be detrimentally impacted by the method of attachment. If a radio-tracking collar is improperly attached, it can lead to irritation and fur loss [77, 78]. The weight of the tracking collar is also important, as elaborated on in Section 2.2.1, as heavy collars can result in increased foraging time and altered energy budgets [78].

2.2.3 Power sources

Most tracking devices are powered by primary[3] batteries, as they provide the highest energy to weight ratio [79], which is of critical importance in reducing the weight of animal tracking devices. In particular, lithium batteries are frequently used, as they operate over a wide temperature range, have a good energy density and a long shelf life [79, 7, 5]. In general, there is a proportional relationship between the weight of the collar and its lifetime, due to the contribution of the battery to total device weight (often

[3]A primary battery refers to one that is not rechargeable. Secondary batteries are those that are rechargeable, but generally have a lower energy density, resulting in a heavier battery for the same energy capacity[79].

measured in the number of fixes as opposed to a lifetime measured in days or months [5]).

For some animal species, solar powered tags are an attractive option. With a solar powered tag, a small photovoltaic module is used to charge a secondary-type battery, allowing operation even when the tag is not insolated [7, 5, 80]. This theoretically allows indefinite operation of the tracking device but, in reality, is checked by the limited number of charge/discharge cycles of secondary batteries [79]. Whilst solar modules add bulk, they can reduce the overall weight of the tracking package, by allowing a smaller battery to be used [80]. However, solar powered tracking devices are only suitable for species which are regularly in bright sunlight and are unlikely to damage or obscure the panel. Solar tracking devices are particularly well suited to tracking birds [81, 82, 83], as the panel can be oriented towards the sun by placing the tracking device on a backpack harness [80]. Conversely, it was found in a test on turtles, that the amount of energy received from the solar panel was significantly reduced when the animal was submerged [84]. Furthermore, tests showed that turtles spend 98% of the time underwater, leading to a very much reduced energy budget in practice [84]. Solar powered tracking devices also have reduced performance in polar regions, especially during winter when there are few hours of daylight [12].

In human-powered systems, there has been progress in thermoelectric generators which are powered by the thermal gradient between the wearer and the ambient environment [85, 86]. In addition, another area of research is in mechanical or vibration powered systems [85, 87, 88, 89]. Although the energy density of these systems is quite poor relative to solar powered modules, they are a possible option for powering tracking devices. One possibility, although it leads to a bulkier device, is to use a hybrid solar/thermal renewable power source [90].

2.2.4 Animal capture and sedation

A major factor in tracking animals is in the initial deployment. The animal has to be located, which can be problematic for elusive or cryptic species and then immobilized for long enough to attach the tracking device. Wild animals generally react to capture by humans the same way they react to predators - as a ‘matter of life or death’ [42]. Capturing techniques vary according to the species - some animals such as penguins can be instrumented without the use of sedatives [91], whilst large herbivores such as elephants [92] and buffalo [93] require helicopters and ground teams. The use of helicopters to dart animals makes the capture and handling component of the study a significant contributor to the cost [93]. Animals panic when distressed and will flee from the source of disturbance until they are exhausted and can die of resultant heart failure, termed capture myopathy [94]. For this reason, the stress imposed on wild animals during capture and handling should be kept to a minimum and loud noises and movement should be avoided [94]. During handling, stress hormones (in particular, cortisol) are produced at an elevated level [95]. There is also a danger to humans during the capture process as:

> ‘the injection or consumption of only a drop of M-99 is sufficient to kill an adult man within a few minutes if the correct antidote treatment is not administered immediately’ [94].

M-99 (Etorphine Hydrochloride) is a powerful immobilizer used to sedate large herbivores [94]. Access to M-99 is highly restricted as it is opiate based and can only be administered by a trained veterinarian, further increasing the cost of affixing a tracking device to a wild animal [94]. In addition, handlers can be gored if an animal is not completely restrained or incorrectly tranquilized [94]. As a result of the dangers both to the animal and to people during the capture process, it is important that the tracking device generates sufficient data to warrant the danger. Recapture collars have thus been designed to minimize stress if an animal needs to be recaptured. The collars are fitted with sedative darts which are injected into the animal upon reception of a remote signal [96].

Another approach to reducing capture trauma is automatic tracking device attachment. Such a device for attaching a coloured (non-electronic) tag was described in 1962 for deer [97]. A snare or noose containing the tracking collar is placed in the field. When the animal walks through the snare, it tightens the collar around the subject’s neck. Another study attached color-coded cable tie collars to squirrels, using both a mechanical and an electrical fastening method [98]. However, a recent attempt to deploy self-attaching VHF tracking collars onto deer showed a poor success rate, with less than 17% of tags successfully attaching to the host [99].

2.3 Tracking Technology

To remotely determine the position of the tracking device, there needs to be some sort of signal (in general, a radio frequency (RF) signal) either transmitted or received by the device. This can be undertaken in a vast number of ways, each with relative advantages and disadvantages in terms of cost, size and suitability. In this section, the three main technologies in use today for tracking wild animals are examined, namely VHF (Section 2.3.1), Satellite (Section 2.3.2) and GPS tracking (Section 2.3.3). Lastly, in Section 2.3.4, a brief overview of other tracking technologies is presented.

2.3.1 VHF Tracking

Very High Frequency (VHF)[4] technology was the earliest modality used for tracking and identifying individual wild animals electronically. The first successfully tested system was demonstrated in 1963 [1]. A VHF tracking system consists of two components – the transmitter (mounted on the animal) and the remote receiver.

Transmitters

A VHF tracking device consists of a power supply, a transmitting device and a radiating element or antenna. The device periodically emits a short RF burst (typically 15 to 50 ms, depending on the transmitter bandwidth and technology [2]) on a dedicated narrow band frequency. The transmitted signal pulse rate is generally in the range of 30 – 120 pulses per minute [2]. The reason that the transmitter is pulsed and not operated continuously is for the purpose of battery conservation – a lower duty cycle results in a lower average current consumption and consequently a greater tag lifetime. Tags in a study are tuned to different frequency bands (typically 5 - 10kHz apart [2]) so that individual animals can be identified and tracked by a receiver.

The first transmitter units used were simple one stage units (essentially an RF oscillator), which are light but have a low output power [65]. Two stage units comprise an RF oscillator and an amplifier to boost the output power at the cost of increased power consumption [65, 2]. Modern units have the transmitter frequency precisely controlled with a quartz crystal to reduce drift due to temperature variations [2]. Microprocessor controlled transmitters have greatly increased the flexibility of VHF transmitters [6]. Controlled activation (for example, 8 hours on, 16 hours off) can be used to dramatically increase the lifetime of the tracking device and align it with typical tracking times [6]. These times can be programmed in by the user, along with the digitally controlled transmitter frequency [6].

Depending on which country they are used in, VHF transmitter tags use different frequency ranges. Common bands are 148 – 152 MHz, 163 – 165 MHz, and 216 – 220 MHz, although frequencies can range from 27 MHz to 401 MHz [21]. Lower frequencies require long antennas (for example, a quarter wave whip at 148 MHz needs to be 50 cm long) which are often impossible to place on smaller animals. Higher frequencies, although requiring a shorter antenna, suffer more from ‘signal bounce’ due to more pronounced multipath effects from the environment, affecting their useful detection distance [100]. Transmitter power varies according to the requirements of the study, but is generally between -10 dBm and +10 dBm [2, 6].

[4]A historical hangup - VHF refers to signals in the region of 30 – 300MHz, whereas with current technology, a signal in this range would be considered a relatively low frequency.

The signal emitted by a VHF tag can be coded in order to communicate more information. This is commonly effected by altering the interval between successive pulses in relation to a measured variable. For example, a common feature of VHF tracking collars is a 'mortality indicator' which alters the pulse rate so that a researcher knows that the collar is stationary [2]. Another use of a variable pulse rate is to communicate the temperature of the transmitter [101]. Thus, both location and sensor data can be inferred from the output signal of the tracking device.

Receivers

In order to remotely determine the position of the animal, the tracking signal must be received and detected. A typical receiver consists of an antenna, an amplifier and a detector. The detector discriminates between the presence or absence of the RF signal, and in some cases also gauges the strength. The detector can be a researcher listening to the demodulated radio signal (this is called manual tracking) or a data-logger (this is termed automatic tracking).

The various methods of estimating the position of an animal are discussed as follows.

Proximity Detection This is the simplest method of determining the coarse location of animals, and records the presence or absence of tracking devices within the receiver's radio range. This gives the animal's proximity to a certain location (which could be a place of interest such as a den or a waterhole). This method is well suited to automatic detection, as a long term monitoring station is relatively simple to deploy. An automatic proximity receiver consists of a scanning receiver (which scans through the individual frequency bands associated with the attached tags) and a logger which records the presence or absence of the signal at a particular time [102]. Antennas used in proximity detection are generally omni-directional in order to determine the proximity in an approximately circular area around the detector.

Triangulation Triangulation is a method used to determine the position of an animal by estimating its bearing (angle relative to a fixed reference, such as North) from two or more locations [103]. The position of the animal is determined by the intersection of these bearings [104]. This provides a much greater accuracy than simple proximity detection, at the cost of more complex receiving equipment. To determine the bearing, a directional antenna on the receiver needs to be used such as the Adcock (H) or loop antenna [105], but the most commonly used one is the Yagi antenna [21]. The number of elements used in the Yagi controls the sensitivity and directionality, but many elements

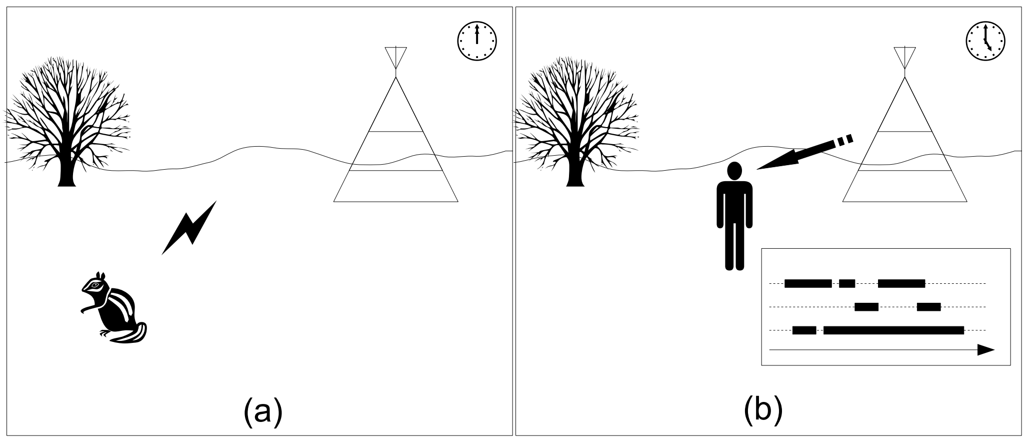


Figure 2.2: Automatic VHF proximity detection. (a) A fixed station records the frequencies (corresponding to unique tags) of all tracking devices within radio range. (b) The user downloads the data and can produce contact logs which indicate the times at which tags were within radio proximity of the receiving station.

result in a heavy and bulky antenna which is not suitable for manual triangulation at lower frequencies [106].

For manual triangulation, a researcher holding a directional antenna rotates in the direction of maximal signal strength from a tag [21, 105]. This bearing is recorded, and the researcher moves to another location to take a second bearing [105]. From the intersection of the bearings, the position of the animal can be determined. Clearly, if the animal moves between successive bearings, the location will be inaccurate [21]. To avoid this problem, two or more researchers can take simultaneous bearings to the tag in question [21]. Manual triangulation can also be used in aerial tracking studies [107].

Automatic triangulation systems use either rotating or non-rotating antennas. The advantage of mechanically based rotating systems is that only one receiving element needs to be used per receiver tower [103]. However, mechanically rotated antenna systems require frequent maintenance for continual operation [108]. Non-rotating direction finding involves using multiple antennas and estimating the angle either through phase or signal strength based measurements [109, 108].

A particularly successful automatic triangulation system is the ARTS (Automated Radio Tracking System) which uses seven 45 m high towers each equipped with multiple fixed log-periodic antennas, installed on Barro Colarado Island, Panama [101]. All the towers are linked together with a wireless infrastructure to a central database which records the positions of all detected tags [101]. Location accuracy is approximately 70 m, depending on the number of bearings and signal strength [101]. This installation allows for the monitoring of multiple animals and their physiological parameters in real-time

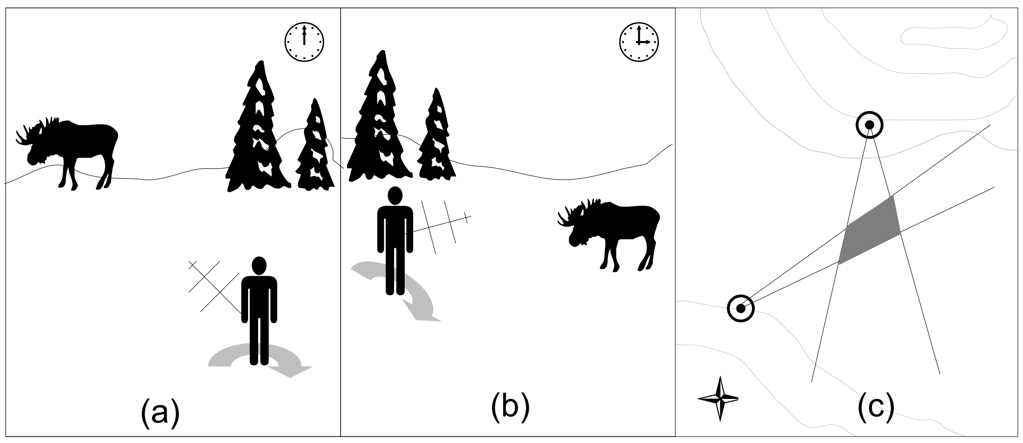


Figure 2.3: VHF Triangulation. (a) The user obtains a bearing to the transmitter, using a directional receiver. (b) The user moves to a different location and obtains another bearing to the transmitter. (c) Knowing the locations where the fixes were taken, the user then marks in the bearings on a map and estimates the position of the transmitter.

[101]. Of particular interest is the ability to detect predation events, inferred by the convergence of location traces from predator and prey [110].

Homing Homing is the process of moving in the direction of the signal until the animal is physically sighted or detected. Homing has the possibility of introducing statistical bias into the obtained locations, as the animal may be disturbed, resulting in altered behaviour [8]. Homing can also be used in aerial tracking – in this scenario, two directional antennas are placed on either side of the aeroplane or helicopter and the pilot flies in the direction of maximum signal strength, generally in a spiralling circular pattern [107].

Advantages and Disadvantages of VHF tracking

VHF tracking is a simple, relatively low cost method of tracking animals. A typical transmitter tag (depending on options) costs in the range of U$100 to U$200 [2, 3, 4, 5, 6, 7]. Manual receivers depending on the complexity vary between U$500 and U$2500 [2, 3, 4, 5, 6, 7]. Because of the low transmitter power and duty cycle of the VHF tags, battery capacities do not have to be large. The smallest VHF transmitters available weigh approximately 0.2 g, but these have a limited output power and a very short lifetime (two – three weeks) [9]. For longer studies, a tracking collar that weighs approximately 500 g can transmit an RF signal for over 4 years [5]. VHF tracking is applicable to a wide range of animal species, from very small animals (and even insects [9]) to large mammals [5].

The main disadvantage of VHF tracking is the labour (and costs associated with labour) involved in triangulation, leading to a paucity of fixes [103]. Although the cost of the equipment is the lowest of the three tracking methods (being VHF, satellite and GPS tracking), the cost per fix is very high – one study found that the cost per fix for VHF was U$65 in comparison to U$8 for GPS [111]. Homing also has the possibility of distorting locations through disturbance to the animal's habitat [8]. Automatic tracking systems provide a vast amount of information with very little labour input but the initial deployment costs are high [101]. Interpretation of location data is also complicated by the non-uniform error distribution – the further an animal is from the receiving antenna, the greater the size of the error polygon[5] [112].

2.3.2 Satellite Tracking: ARGOS

The drawback of VHF tracking is limited coverage - a researcher needs to be within radio range of an animal in order to determine its location. The maximum extent of practical tracking areas vary between 50 and 300 km^2 [65]. This precludes the use of conventional VHF tracking on wide ranging species such as wild dogs [113]. Satellite tracking provides truly global location estimation by using space based receivers (as contrasted to GPS tracking (see Section 2.3.3) which uses space based transmitters).

The system that is in widespread use at present for tracking wildlife is ARGOS, a joint French Space Agency and US National Oceanics and Aeronautics Administration (NOAA) satellite constellation [11]. These satellites localize and identify the signals sent by transmitters (called Platform Transmitter Terminals – PTT) [12].

Two NOAA satellites are currently deployed, with plans for a third to be added [10]. These orbit at an altitude of 850 km in sun-synchronous orbits [10]. Each satellite has a look down field of view of approximately 5000 km in diameter and the orbital period is approximately 102 minutes [10]. ARGOS provides global coverage, however it does not provide continuous access at all locations due to its orbital trajectory [10]. Furthermore, as the satellites are in a polar orbit, location estimates and coverage towards the equator are poorer than those nearer the poles [10]. Depending on the terminal's location, a satellite will be overhead for between 2 and 12 minutes per orbit [10]. The number of times in a day that the PTT can be localized depends on the latitude and varies between 7 (equator) and 28 (polar) [10, 12].

The tags all transmit on the same frequency, 401.650 MHz, and for this reason there is a limit on how often tags are allowed to transmit in order to avoid interference between PTTs [10]. The satellite records the Doppler shift of the incoming signal, for instance,

[5]The error polygon defines the region of uncertainty in the estimation of the animal's position.

Location Class	Estimated Accuracy
3	$<$ 150 m
2	150 m to 350 m
1	350 m to 1000 m
0	$>$ 1000 m
A	No estimate of accuracy [Only 3 messages received]
B	No estimate of accuracy [Only 2 messages received]
Z	Invalid location

Table 2.1: Typical accuracy of ARGOS location fixes [10]

if the satellite is directly overhead, there will be no Doppler shift [10]. However if the path of the satellite is advancing towards or receding from the PTT, the frequency will be shifted upwards or downwards respectively [10]. The best position estimate is obtained if four or more transmissions are obtained from the PTT [10]. From the known position of the satellite and the Doppler shifted signals, a location can be determined [10]. Depending on the signal strength and the location of the PTT and the satellite, the accuracy varies as shown in Table 2.1. In practice however, researchers have found (by comparing the ARGOS fixes to those simultaneously acquired from GPS receivers) that the accuracy was worse, with a mean error of 5.0 km for Class 1 fixes, which is a factor of 5 greater than specified by ARGOS [114]. A filtering algorithm has been proposed which discards erroneous satellite fixes from the data-set – approximately 30% of data fixes for gray seals were removed [115]. A more recent approach uses a correlated random walk mobility model to filter and reject outliers for satellite tracking data from turtles [116]. In addition, the authors found that ARGOS accuracy degraded with changes in sea-surface temperature, which is possibly related to thermal effects on the stability of the transmitter carrier frequency [116]. Small amounts (32 bytes per transmission) of telemetric data can be uploaded by the PTT to the ARGOS system, allowing for the real-time upload of sensor data [10].

The transmitting power of PTT tracking devices varies from 0.5W to 2W, which is significantly higher than for VHF tracking devices with a typical power output of 10mW [10, 12, 6]. Thus, PTT transmitters are larger and heavier than VHF tracking devices due to the increased battery requirements [80]. Although PTT's are designed to be tracked from space, it is also possible to detect them using ground-based receivers, which can be used to locate an animal or to recover a transmitter with low output power [117].

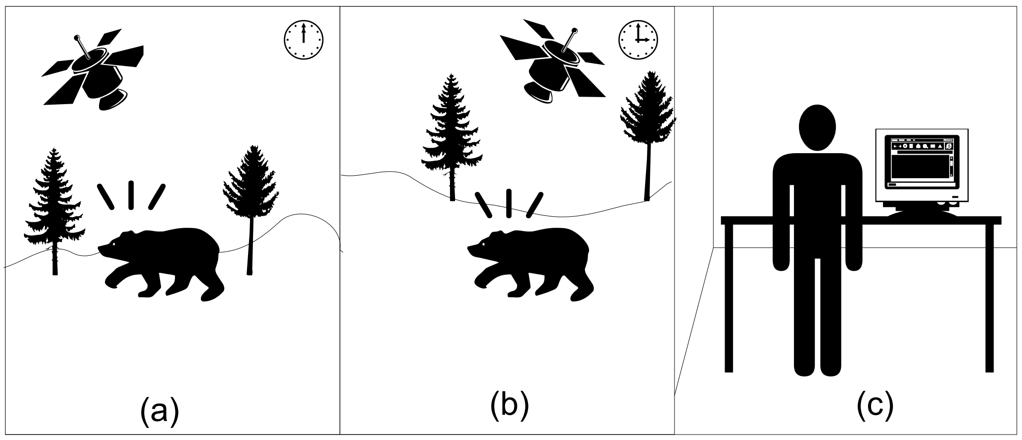


Figure 2.4: Satellite Tracking. (a) The animal carried PTT emits a signal, whose Doppler shift is calculated by the satellite. (b) The satellite receives more signals from the PTT. (c) The position of the animal is estimated from the known position of the satellite and sent electronically to the end-user.

Advantages and Disadvantages of Satellite tracking

ARGOS provides truly global tracking coverage, with near real-time updates of position available. Once deployed, locations acquired from the PTT tag are relayed directly to the user, meaning that no labour is entailed in tracking the animal. This has the added benefit that there is no habitat disturbance either.

The drawback of the satellite tracking system is the cost of obtaining data – the tags themselves cost U$3000 [5, 6, 80] and the user has to pay service fees to the data processing centre in the order of U$3000 to U$5000 per year [10]. The location accuracy of ARGOS tracking tags is poor compared to VHF and GPS tracking [10]. In addition, the percentage of successful location fixes is often low (one study found that percentage fix probability to vary between 11 % at the bottom of a valley and 56 % on the top of a mountain [118]). Furthermore, the number of fixes that can be taken per day is small compared to other methods such as GPS tracking, due to the limited time window where the satellites are overhead [10]. The main drawback of satellite tracking devices is the weight of the PTT unit, due to the high transmitter output power, necessitating large batteries for extended lifetime [5, 6, 80]. PTT tags generally weigh in the region of 200 to 400 g, but solar powered units as light as 15 g are available [12, 80].

In the quest for a global tracking system that is able to monitor smaller animal species, Wikelski *et al.* have formed the Icarus initiative that aims to deploy sophisticated technology on Near Earth Orbit satellites that can locate sub-gram tags [119]. However, there is yet to be a launch of a dedicated animal tracking satellite.

2.3.3 GPS Tracking

The completion of the Global Positioning System (GPS) in 1993 heralded a new era in wildlife tracking [120]. GPS enables precise (within 5 m) location of a receiver anywhere in the world, 24 hours a day [121].

Principles of GPS

GPS uses a constellation of 24 satellites vehicles (SV) orbiting the earth in precise 12 hour orbits which broadcast signals on the L1 (1575.42MHz) and L2 (1227.60MHz) high frequency bands[6]. There are plans to introduce three more signals (L3, L4 and L5) to improve accuracy for both civilian and military purposes. The L1 signal is a CA (Coarse Acquisition) code modulated to carry information that the GPS receiver uses to calculate its position, namely a pseudorandom code, ephemeris data and almanac data. The pseudorandom code uniquely identifies each satellite. Ephemeris data is transmitted by each satellite which indicates the precise position of all the satellites, and is valid for 4-6 hours. Almanac data is status information for each satellite in the constellation (such as healthy or unhealthy and coarse orbital information), as well as current date and time.

A GPS receiver needs to acquire signals from at least four satellites in order to determine its three dimensional position. The receiver determines the time of flight from each of the satellites by performing cross-correlation in the time domain. From the time of arrivals, and knowing the exact position of the satellites from the ephemeris data, the receiver can be located precisely in three dimensions. The 'extra' variable that the receiver solves for is time, as all the satellites are precisely time aligned and contain on-board atomic clocks to maintain accuracy. Thus, precise time synchronisation occurs as a byproduct of locating a receiver, which is useful for timestamping acquired data.

A GPS receiver takes time to obtain a location fix however, and this depends both on the technology used in the receiver and when last the receiver obtained a good fix. For instance, when a u-blox NEO-4S GPS receiver [28] first starts, or it has lost synchronization (that is, no ephemeris, time or almanac data), this is referred to as a 'cold-start' and takes an average of 34 s [122]. A 'warm-start' occurs when the receiver has valid ephemeris and almanac data and a previous position estimate, but the receiver has been powered off for some time (up to a few hours) and thus has lost precise time synchronization and typically takes 33 s on average [122]. A 'hot-start' is when the receiver has valid position, ephemeris, almanac and time data, and has been powered down for

[6]Unless otherwise specified, the details of operation of GPS are taken from [121]

a short amount of time (a few minutes), and this takes 3 s [122]. These figures are from the manufacturer's datasheet and are:

'measured with good visibility and -125 dBm signal strength' [122].

In reality, many factors affect the accuracy and availability of GPS estimates - these include foliage cover, multipath effects from strong reflectors, Ionospheric delays, the number of visible SV's and the accuracy of the oscillator in the receiver. Whilst tracking wild wolves, Coelho *et al.* found that GPS fixes were more likely to be successful at night when the wolves hunted in the open, as opposed to day when they slept in dense vegetation [123]. They also found the fix probability to be in the region of 90% [123]. An earlier study only obtained 5% of attempted fixes on tags attached to Pacific Walruses [124]. The effect of environmentally induced biases (such as thick vegetation) on the probability of obtaining a successful fix can act as a distortion in habitat selection studies, and the effect of this must be taken into account so as to not draw invalid conclusions [125].

GPS receivers have a relatively high power consumption (in the region of 100mW when active [122]) due to the high-gain radio front-end and also the multi-channel cross-correlation searching process. In addition, the receiver needs to be active until a fix is acquired, which can take a few minutes in some cases [13], resulting in a high power cost per fix. As a result of the high power consumption of GPS receivers, it is not practical to operate them continuously for long term studies and they are thus duty-cycled by only powering them up at uniform time-intervals (typically between 5 minutes and 4 hours [5, 6]). However, short term studies (less than 24 hours) have been undertaken using always-on GPS receivers to examine the fine details of foraging behaviour in African Penguins and demonstrated that reduced sampling rates (greater than 10 minutes between samples) had the effect of significantly underestimating the actual length of foraging trips [126]. In addition, they found that the re-acquisition performance of the tracking devices was poor - with a 10 s sampling rate, the lag between samples had a median time of 44 s [126]. In light of the long acquisition times of GPS receivers, in particular when attached to diving marine animals which spend a short period of time on the surface, a recently introduced (2006 prototype) acquisition strategy (with the market name FastLoc) has been designed [127]. Greatly reduced acquisition times (< 100 ms) are possible, at the cost of reduced accuracy and an increased message size [127]. The GPS receiver does not calculate the position of the receiver, but measures the time of flight from the visible satellites for further post-processing to generate position fixes [127]. Accuracy is also degraded and depends on the number of visible satellites [127].

Contrary to satellite tracking (See Section 2.3.2) where the ground based unit is a transmitter, a GPS unit is a receiver. Thus, the locations calculated by the receiver have to

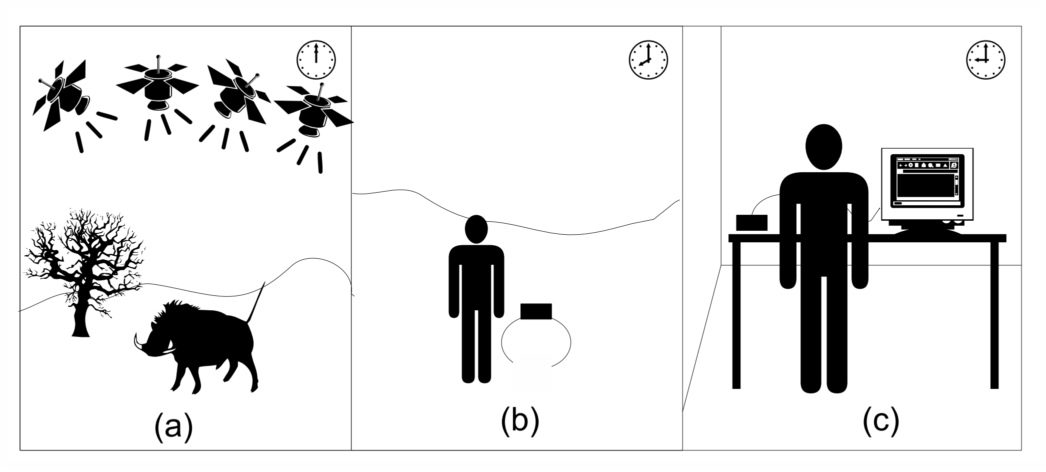


Figure 2.5: GPS Store on board device. (a) The ground based receiver determines its position by calculating the time of flight from the signals transmitted by the satellites. (b) At the end of the study, the unit is retrieved from the field. This may involve resedation of the animal. (c) The data is downloaded to a computer via a communications link for analysis.

somehow be transferred from the GPS receiver to the end user. There are a number of different methods which are discussed as follows:

Store on board This is the simplest option, but has the greatest possibility of complete data loss [5, 6]. Store-on-board collars log the acquired position fixes to non-volatile memory. At the end of the study, the collar is retrieved and the locations are downloaded. Whilst this is a simple strategy, it means that the collar *has* to be retrieved to obtain *any* data. For this reason, these collars are equipped with a VHF transmitter (generally with its own, separate power source) so that the collar can be located [5, 6]. If the collar is still attached to the animal, then the animal must be re-sedated/captured so that the logger can be retrieved. Some collars are equipped with a 'drop-off' mechanism that physically breaks the collar so it slips off the animal's neck [5, 6]. The steps involved in acquiring locations from a GPS datalogger are shown in Fig. 2.5.

The main drawback of the system is the threat of data loss through collar failure, loss or destruction. However, due to their simplicity and power efficiency, they are an attractive solution for some species. A lesser issue of the logger approach is that there is a long time lag between the deployment of the data and the retrieval of the collar. In light of these problems, various wireless upload options are available.

Wireless Upload Options To solve some of the problems posed by the GPS logger, some units are equipped with VHF modems to upload data to researchers in the field [4]. The tags still function as loggers, but at fixed times (real time synchronization provided

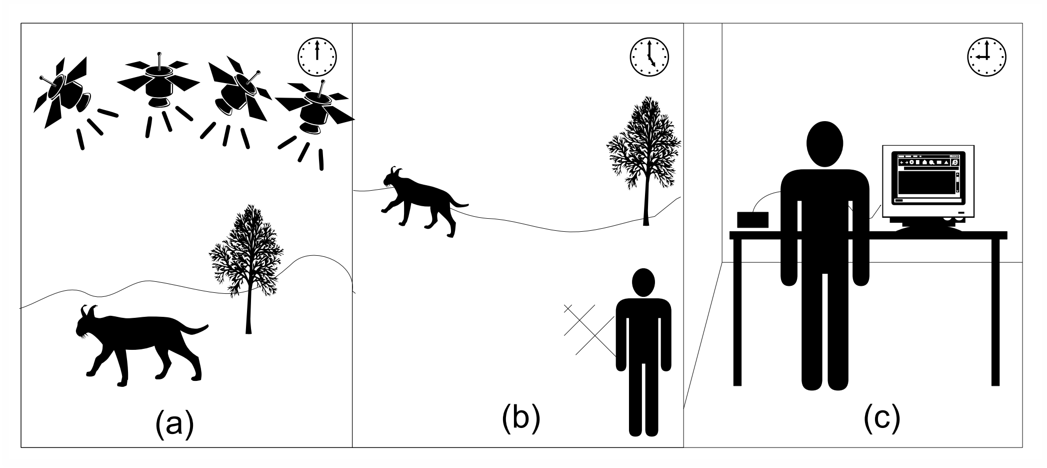


Figure 2.6: GPS with UHF upload. (a) The ground based GPS receiver estimates its position based on the time of flight of signals from the overhead satellites. (b) At periodic time intervals, a user locates the tag (via VHF/UHF homing) and with a field based-receiver, downloads data from the tag. Some collars allow configuration parameters to be uploaded as well. (c) The field receiver is later connected to a computer and the data downloaded.

by GPS), the tag will upload a subset of the last acquired GPS locations. Researchers can receive this data, and obtain recent positions of the animal. These tags also have VHF or UHF beacons so the animal can be tracked and found in order to download its data [13]. More complex tags have two way modems that enable the tag to be remotely interrogated and reprogrammed [5]. At the end of the study, the tag is retrieved in the same way as an ordinary logger, and all the acquired data is downloaded. In this way, by providing a VHF uplink, the time lag between acquiring the data and retrieving it can be reduced, and the impact of logger failure is reduced. However, these devices require a larger energy source than a simple logger due to the power requirements of the VHF modem, leading to a heavier and bulkier collar [5]. The steps involved in obtaining locations from a UHF/VHF collar are shown in Fig. 2.6.

Tags can also be equipped with a GSM module that allows data upload to a cellular network [3, 13, 5]. The tag acquires GPS locations, and buffers them up until it is in range of a GSM tower, at which point it sends the data, either by SMS or GPRS [3, 13, 5]. GSM modems have a high power consumption (1 W peak transmission power), and need to register on the network before they can send data, which can take up to 30 s [128]. Many areas, especially remote areas where wildlife needs to be monitored, do not have cellular coverage, which precludes their use. As the GSM interface is two-way, data can be queried from the collar (such as the current location), and alerts (such as straying outside a particular area) can be delivered to a user automatically [13]. GPS-GSM collars are heavier than GPS dataloggers, due to the GSM modem and antenna and also to increased power requirements, necessitating a larger battery [13].

In order to provide real-time access to data with global coverage, GPS tags are available that upload position data directly to satellites. Depending on location, satellite constellations such as the Inmarsat [13] are used, but the most commonly used is the ARGOS system as it provides global access [5, 12, 10]. In this way, highly accurate position estimates can be determined anywhere in the world and uploaded from anywhere in the world. One advantage of the GPS unit is that it provides time-synchronization which enables the satellite transmitter to power up only when the receiving satellite is overhead (in the case for ARGOS, Inmarsat is geostationary), reducing power consumption [12]. These are the most expensive (in the region of U$3000 to U$5000 [12, 80], excluding data costs) and bulkiest tags, but provide excellent accuracy with global coverage.

Advantages and Disadvantages of GPS tracking

GPS tracking has had a significant impact on wildlife research by enabling the acquisition of detailed location information anywhere in the world with excellent accuracy. A GPS receiver is able to acquire locations on demand, something which is impossible with satellite tracking (see Section 2.3.2), due to the sparsity of satellite coverage. The locations obtained by GPS tracking are much more accurate than Satellite tracking (typical accuracy for GPS is 5 m, whereas the best satellite tracking accuracy is in the order of 100 m).

A great advantage of GPS tracking in comparison to VHF tracking (see Section 2.3.1) is that minimal time is needed in the field. As the GPS unit calculates its position itself, the labour costs associated with GPS tracking are substantially lower [111]. In addition, the possibility of bias introduced through disturbing the habitat of the subject being studied is greatly reduced [8]. VHF tracking also has very limited coverage – it is not suitable for wide ranging animals, as the maximum range that an animal can be detected is 8 – 10km [8] (or 15 – 30 km from air [107]). GPS on the other hand is a global tracking system.

The main drawback of GPS tracking in comparison to VHF tracking is the power consumption of the GPS receiver. For this reason, GPS receivers are duty-cycled such that they spend the majority of their time in low power sleep mode [13, 5, 6]. As a (simple and optimistic) example, a GPS receiver acquiring locations every 10 minutes, with an average time-to-first-fix (TTFF) of 20 s, consumes an average power of 3.3 mW, if the power consumption when the receiver is active is 100 mW. In comparison, a VHF beacon transmitting at a power of 10 mW every second for 30 ms at an efficiency of 40% will consume an average power of 0.75mW. Thus, for the same lifetime, the GPS unit must have more than four times as much battery capacity as for the VHF unit. As the majority of the weight of the unit is due to the energy source, this has the implication that the GPS unit will weigh approximately four times more than the VHF unit.

Due to their complexity, GPS collars are more expensive than VHF beacons. A typical VHF transmitter costs in the region of U$100 – U$200 [2], whilst a GPS collar will cost in the region of U$2500–U$4500, depending on which options (such as satellite upload) are installed [80, 6].

2.3.4 Miscellaneous Tracking Technologies

This section overviews other technologies which can be used to track wild animals.

One such technology is the use of heading based sensors in order to reconstruct the path taken by the animal in two or three dimensions [129, 130, 131, 132]. A sensor measures the heading (bearing angle relative to North) of the tag and another sensor measures the distance the tag has travelled [133, 129]. This approach is referred to as dead-reckoning or path integration [129]. Dead-reckoning systems provide unparalleled short-term (in the time frame of seconds to hours depending on sensors, errors and animal movements [134]) position accuracy, but suffer from offsets (and drifts due to sea currents for the case of marine creatures [129]), leading to large long term errors [135]. As the acceleration signal is integrated twice to determine displacement, any errors/noise will propagate rapidly. Dead-reckoning is a relative location determining system, as the position of the host is measured relative to the starting position [132]. In order to avoid some of the errors associated with the rapid error accumulation, dead reckoning can be combined with absolute location techniques such as GPS [136]. The GPS receiver is periodically enabled and the acquired locations are used to 'calibrate' the dead-reckoning module [136].

Passive (Radio Frequency Identification) RFID tags, also known as PIT (Passive Integrated Transponder) devices, are uniquely coded transponders that emit their ID in response to an interrogation signal [137]. PIT's are very lightweight (0.6 g) and small as they do not have a power source of their own - they are powered from a reader [137]. As such, their range is poor (between 20 cm and 2 m depending on the tag size and reader technology) compared to other tracking technologies, and are more useful for presence/absence detection in a certain area, as opposed to fine-grained location [137]. PIT's however are inexpensive (U$3 each in quantity), providing one of the most cost effective methods to tag large numbers of animals [138, 137]. The readers are relatively expensive (U$500 - U$1000) each and have heavy power requirements, in the region of 10 – 20 W for a continual scanned system, limiting the extent of their deployment[138]. PIT's are also standardized, by ISO11784/11785 which dictates their numbering scheme and operation, leading to interoperability between manufacturers, something which is rare in animal tracking applications [139, 139]. As there is no power source to be depleted, PIT's can be regarded as having an 'infinite' lifetime. In addition, the frequencies used for animal tagging (134 kHz) are not affected by water or flesh, making implantation and underwater operation a possibility [138]. Active RFID's contain a power source, and so have a greater read range and the capability to sense and relay environmental variables such as temperature and pressure [140]. Due to their power source, their lifetime is limited and they are bulky compared to their passive counterparts. Active RFIDs can be regarded as intelligent VHF/UHF tags.

Harmonic radar can be used to track the position of animals [141]. A high power radar illuminates a tag, which backscatters radiation at a higher harmonic [141]. A detector records the presence of the backscatter which can be used to determine the device's position and the radar dish can be rotated in order to sweep the area and obtain the position of the tag in three dimensions [142]. The devices placed on the animal can be made very lightweight as consist of a diode, which through its non-linear switching action generates high frequency harmonics of the carrier, and an antenna which receives the radio signal [141]. Harmonic radar has been used to track the flight paths of insects, as these tags are small enough to place on larger insects such as carabid beetles [141] and honeybees [142]. However, the tag has a trailing antenna which adds aerodynamic drag and limits the suitability of this approach [143]. To detect groups of insects, as opposed to individual insects, traditional radar techniques can be used without having to attach any devices [144]. In some regards, harmonic radar and PIT's can be regarded to be similar, as they are both powered by incident radiation. However PIT's carry a unique code and respond in the same frequency band.

Another method of determining approximate position is to use a global location service (GLS) system [145, 146]. This device measures and records the incident luminosity on a light sensor, and contains an accurate real time clock (RTC) [145]. The length of the day (from dusk to dawn) is used to determine latitude, and longitude is calculated from the local time of midday or midnight [135]. A filter can be placed over the light sensor to restrict the incident wavelengths to the deep blue spectrum which is affected less by cloud cover [135]. Although the accuracy of this system is poor (one study found it to be within 31 km of the true position) and the number of location fixes a day restricted to two, GLS devices are an inexpensive and lightweight means of determining global position for wide ranging animals [135]. In research on albatross, the accuracy of the GLS was found to be in the order of 180 km [147]. As a GLS device is a datalogger, it needs to be retrieved at the end of the study in order to derive the positions of the host. Light sensor orientation is also a critical parameter, but can be compensated for using tilt sensors [135]. Another method of improving GLS accuracy is to compensate the position using measurements of Sea-Surface-Temperature (SST) which is simultaneously sampled by satellite [148]. GLS sensors which operate for two years are as light as 20 grams [135].

Many animal species have a distinct coat marking which can be used to identify individuals. Automated visual recognition has successfully been used to identify African Penguins through their chest spot patterns [37, 38]. The problem with visual recognition is its limited detection range and its susceptability to factors such as dirt, orientation and ambient lighting [38]. However, it is a non-invasive method of monitoring large populations of animals, as tracking devices do not need to be attached or implanted. Remote camera traps can be used to record the presence of certain animals at various locations - these are triggered to take a picture by motion or sound in the vicinity (for an extensive review, refer to [149]). Although remote camera traps are simple and relatively inexpensive due to the proliferation of low cost digital camera technology, they require a large amount of manual interpretation to classify the recorded images [150], although machine classification techniques can also be used on these still images. Like passive RFID, visual recognition systems can be regarded as proximity rather than true position detectors due to their limited range of operation.

Some animals (in particular birds [151], frogs/toads [41] and cetaceans [152]) emit loud calls to communicate with their peers. An array of microphones can be used to triangulate the position of these animals and recognize the calling species [153]. In the case of sufficiently distinct calls, it is also possible to identify individuals [152]. The drawback of this approach is that animals can only be detected when they are vocalizing, however, unlike visual recognition they need not be out in the open and visible.

There are other methods which can be used to identify individuals, such as banding/ringing [154] and chemiluminescence [155] but these can be regarded as aids to manual identification rather than automatic tracking technology.

2.4 BioSensing Technology

In this section, what environmental variables can be measured and what they indicate about the behaviour and state of the host animal are examined. This is not an exhaustive review of all sensor modalities and technologies, but rather an overview of some which are commonly used.

With all sensing technology, there are a number of factors which affect their usefulness and suitability to accurately represent the variable they measure. Sensing systems are characterized by their accuracy and precision. Closely related to the precision of the sensing is the resolution of the analog to digital converter (ADC). An 8 bit converter quantizes the measurand into 256 discrete bins, whereas a 16 bit converter has 65 536 discrete steps. The obvious drawback to a higher converter resolution is increased storage requirements, but with modern solid state memory, this is less of an issue. The bandwidth or speed response of the sensing system may also be important if it is to correctly capture transient activities. For example, measurements of dive profiles in sea birds was hampered by the slow response times of datalogger temperature monitors, the best having a time constant (time taken to reach 63% of the final value) of 16 s [135]. The orientation or position of the sensor is also important, depending on the variable being measured, which can have an impact on the design and suitability of the sensing device.

Many data acquiring tags are equipped with temperature sensing capabilities [13, 3, 5, 6, 127]. However, in many cases, the temperature measured is not the temperature of the host, nor the temperature of the environment, but rather the temperature of the tag which is a combination of both [135]. Measuring the temperature of the tag is useful to correct temperature induced bias in other sensors, in particular pressure sensors [135]. Measurements of the ambient temperature can be used to infer the animal's habitat and behaviour - in particular, it can be used on diving animals to measure when they are underwater so the length of foraging dives can be determined [156]. Internal temperature measurements help researchers understand how animal behaviour influences their core temperature, especially for those species which live in extreme environments [157]. Sensing internal temperature can also be used to characterize feeding behaviour, as rapid decreases in stomach temperature are linked to the ingestion of prey in endothermic marine species [158, 159]. However, Ponganis et al., questioned the accuracy of the stomach temperature pill as a determinant of ingestion by using simultaneous video recording of foraging Emperor penguins [160]. Internal temperatures can indicate altered physiology such as oestrus or rutting or hibernation. Related to direct temperature measurement is heat flux or the flow of thermal energy to or from an animal's surroundings [161]. This type of sensor can be used to determine how animals thermoregulate and their energetic output with various activities [162, 163].

Dive profiles can be recorded using time-depth recorders (TDR) which determine depth from the increase in water pressure with position in vertical column [164, 165].

Acceleration sensors are a relatively recent introduction to the field of wildlife telemetry, made possible through the minituarization of accelerometers [166]. These can be used to measure position through inertial navigation (see Section 2.3.4) but can also be used to measure animal behaviour, in particular gait and locomotor activity [167, 168]. Angle of tilt can also be determined, which can be used to compensate for angle induced effects on other sensors' performance, in particular incident light level [135]. Acceleration sensors have been used as a practical alternative to traditional respiratory gas measurements of exertion and energetics as they can be used on free ranging animals [169]. A very recent study (March 2008) used tri-axial accelerometers to identify movement patterns on 12 different animal species and demonstrated that this technique shows promise for fine grained characterization and analysis of animal behaviour [170].

With the advent of miniaturised video cameras, it is now possible to visually record an animal's behaviour and its surrounding environment [171, 172]. A miniature video camera is connected to a recording device (which can be tape, flash memory or hard-drive [31]). Video loggers, as exemplified by the 'CritterCam' used in National Geographic programs [172], provide a vast amount of data about animal behaviour, much of which is difficult or impossible to capture using other types of sensors. For example, it has been shown that emperor penguins engage in co-operative foraging behaviour [173]. Video systems are also useful for placing other sensor data (such as depth or vocalization) into the context of animal specific behaviour [31, 174].

Animals can also be equipped with sensors that measure emitted vocalizations and ambient noise [175]. This information can be used to determine the behaviour of animals and also what class of food they are eating [176]. Nelson *et al.* found that sounds were recorded most accurately when the microphone was attached directly to the cranium, rather than being collar mounted [177].

Sensors can also be used to measure physiological parameters of the host directly. These include heart-rate [178, 179, 180], blood pressure [181], respiration [180] and measurements of muscle activity [182]. Heart-rate monitors (electrocardiograms (ECG)) sense the beating of the heart by measuring the potential across the cardiac tissue. They can either record the beats per minute (which can suffer from invalid detection of the T wave instead of the R wave) or a detailed ECG (which has issues arising from the volume of data generated by the act of sampling at 50 – 100 Hz) [183]. Respiration can be measured indirectly from ECG data through spectral analysis [184], but can also be measured from mandible angle [185, 186]. Using a hall-effect sensor on one half of the beak and a magnet on the other, the inter-mandible angle can be determined, which is a useful indicator of both respiration and feeding [186].

This brief overview of sensing options demonstrates that there are a vast number of variables that can be measured in addition to animal location.

2.5 Chapter Summary

There are three major location finding techniques currently in use, namely VHF tracking, satellite tracking and GPS tracking. VHF tracking is inexpensive, and due to the modest power requirements of the transmitting units, can have lightweight packages for long term operation. Thus, it can be used to track a wide range of animal species. However, VHF tracking is labour intensive and accuracy dependent on the tracking method used. Detection range is typically in the order of a few kilometres, but aerial tracking can increase the range up to as much as 30 km. Conversely, satellite tracking allows for real-time location of an animal anywhere in the world. Although the tags and data costs are expensive, there are no continual labour costs associated with tracking, as locations are determined automatically. Tag sizes, unless the possibility exists for solar power, are large due to the high power consumption of the transmitting unit. Accuracy is quite poor, with the best case fixes being in the region of a few hundred metres of true position. In addition, a limited number of fixes can be taken per day, leading to possible undersampling of detailed motion paths. However, satellite tracking is ideally suited for wide ranging or migratory animals. GPS tracking provides excellent accuracy (< 5 m), anywhere in the world, and on demand. GPS receivers have fairly high power consumption, precluding continual operation for long term studies. Due to their power requirements, they are only suited for monitoring larger animals. The overbearing issue with GPS tracking devices is that the animal carried receiver determines the location itself, resulting in the need to somehow download the data from the tracking device. Store-on-board devices, whilst the simplest option, have the possibility of total data loss if the tag is destroyed or cannot be found at the end of the study. The wireless upload options reduce the criticality of tag retrieval, but further increase the bulk and weight of the tracking collar. GPS tracking devices are also significantly more expensive than VHF tracking devices, but can generate large volumes of precise location data with minimal labour input.

With regards to sensing technology, there are a wide variety of parameters that can be measured, ranging from temperature to detailed acceleration profiles. The sampling frequency has to satisfy the Nyquist criterion on sensor bandwidth, but memory and energy constraints must also be taken into account. To reduce memory requirements, sensor data can be pre-processed by the tracking device itself, only recording summary statistics (such as maximum and minimum values over an interval).

From this review, it is apparent that there are a variety of tracking technologies available, with different ranges of operation, resolutions and costs. Animal tracking is a cross-discipline field, that requires careful analysis of the available technology and the constraints imposed on it by the target species, to arrive at the best solution, with the bias being toward minimizing risk to the host animal. The constraints imposed both from the biological and technological sides have the result that there is no single technology available that can simultaneously satisfy the following requirements:

- Small size.

- Low weight.
- Excellent spatial accuracy.
- High sampling frequency.
- Global coverage.
- Long lifetime.
- Real time access to data anywhere in the world.
- High resolution data from multiple sensors.
- Low cost.

Tracking devices are thus specified such that some parameter (such as lifetime) is maximized, at the cost of a decrease in some other desirable feature (such as sampling frequency). Thus, a 'one-size-fits-all' tracking device is not possible at this present point in time, and is likely to remain this way for some years to come.

All animals are equal, but some animals are more equal than others

Animal Farm, George Orwell (1903–1950)

3 EcoLocate: A heterogeneous wireless network system for wildlife tracking

3.1 Introduction

As the literature review demonstrated, there exist a plethora of tracking and telemetry systems[1]. As a consequence of both technological and biological constraints, there is currently no animal tracking device available that is suitable for deployment on all species. This has resulted in a piece-meal approach to wildlife tracking, symbolized by a characteristic lack of standardization. In order to monitor inter-species interactions, researchers often have to use two or more different technologies, as the more complex and heavier tags cannot be placed on smaller animals. The simpler technologies such as VHF and RFID have some degree of interoperability between manufacturers, primarily as a result of national communications regulators that assign frequencies of operation. However, the more complex devices such as GPS receivers with UHF modems for uplink show little promise of standardization – a receiver manufactured by one company is unlikely to be able to interface to a GPS collar from another manufacturer, leading to technological lock-in.

In addition, due to the vast diversity in the Animal Kingdom, severe constraints are imposed on the size, weight and form-factor of tracking devices. In general, the principal contributor to tag weight is the weight of the power source. This has the result that tracking devices with a large power drain are unsuitable for attachment to small animals. There are two approaches to designing tracking devices subject to this constraint: design one type of tag which satisfies a minimum weight requirement (the homogeneous solution) or to design multiple variants, which are different weights, allowing a greater variety of animal species to be tagged (the heterogeneous solution). Although the homogeneous solution leads to a simpler design procedure, as only one type of tag has to

[1]Portions of the work in this chapter have been published in [187].

be fabricated, a wide variety of animals cannot be tagged, resulting in an unfair bias towards monitoring larger animals. The heterogeneous solution appears to be the best approach from the point of view of being able to monitor more components of the ecosystem, but has the technological drawback of resulting in a wide range of variations on tracking devices. This represents the current state of affairs, with a vast array of tracking technologies available to researchers, but no unification between them.

A third way is thus advocated: a homogeneous technical design that results in a heterogeneous solution domain. There are three primary functions that tracking devices need to perform: they need to sense data, store and process the information and send the data to the end-user. In existing tracking and telemetry solutions, each tag is treated as an individual and distinct entity. This however places severe constraints on the choice of communications device, as the high-end systems such as GSM modems and satellite links are bulky, power-hungry and expensive. Rather, observe that if tracking devices are equipped with low power radio-transceivers, a wireless network of tracking devices can be formed. Data is relayed from collar to collar until it reaches an end-point (which can be equipped with a high-end communications device) whereupon it is relayed to the end-user, providing a common solution to the issue of transferring data.

A modular approach is thus proposed, whereby a complex tracking device is designed which is able to perform all requisite functions, such as routing, sensing, replicating data and interfacing to the end user. By disabling certain functions, such as routing, a simpler and consequently lighter tag can be constructed. Note that this results in a single technical design, but a proliferation of possible implementations, varying in capability, functionality and sensing options.

According to a paper on the state of the art and applicability of wireless sensor networks for ecological monitoring,

> ‘there are many technical problems to be overcome in using sensor networks (for ecological monitoring), such as sensor development, network scalability and power demands to address the grand-challenge problems’ [188].

In this chapter, these issues are addressed and a system design for a generic and modular wildlife tracking and telemetry system, using a wireless network as a unifying factor, is presented. As an example application, we take a typical African bushland, characterized by some very large animals (such as elephants) and many smaller animal species. The large animal species can generally be equipped with rugged, robust collars that can last for many years, whereas the lifetime of the attachment on smaller animals is less of an issue. Section 3.2, examines similar work, in particular existing wireless network approaches for wildlife tracking. In Section 3.3, the use of a wireless network as a communications medium is discussed, followed by the requirements of a tracking system in Section 3.4. The various classes of components are presented in Section 3.5, along with the beacon protocol which is common to all classes and allows them to communicate with one another. Based on the observation that a simple tag is a complex tag with certain functions disabled, a dynamic method of disabling tag functions to result in a longer lifetime is outlined in Section 3.6. The multiple uses of transmitted beacons is presented in Section 3.7, followed by the modular approach to the design of the tracking system in Section 3.8. Example deployments, from simple VHF studies to full network designs, are presented in Section 3.9. Section 3.10 takes the modular design of the system, and shows how it naturally leads to a graphical system design, where zoologists can specify their own tag design and alter tag parameters. Lastly, conclusions are drawn in Section 3.11.

3.2 Related work

3.2.1 Wildlife tracking using Wireless Networks

To the best of our knowledge, the first presentation of a wireless network attached to animals was the ZebraNet project, reported in the literature in 2002 [14]. The motivation for the study was the difficulty in retrieving data from existing GPS collars, as each animal had to been contacted individually in order to obtain the data. This is a laborious affair when animals range over a wide geographical area. To address this issue, zebras were equipped with solar powered collars that contain radio transceivers (different versions of the collars had either one or two radio links with different ranges, power consumption and bandwidth) [18]. The collars share tracking information (temperature and location) with one another in a flooding manner. Thus, by coming into contact with a few animals, data from the remainder of the group can be obtained. The basestation in ZebraNet is mobile, carried in a vehicle (either ground or aerial). The obvious drawback with the flooding approach is that data requirements grow as $O(N^2)$, where N is the number of nodes in the network [14]. To try to tame the data load, a simple history based protocol was used, where nodes which recently were in contact with the basestation were more likely to be receivers of information. In a further attempt to reduce bandwidth and data requirements, two techniques (namely data compression [16] and erasure coding [189]) were also proposed and demonstrated to reduce the power consumption of the tracking devices.

A ‘middleware’ layer, named Impala was incorporated into the ZebraNet collars – this provides an abstraction between the low level hardware and the high level application, providing features such as in-the-field reprogramming [190, 17]. The data traces from the ZebraNet project have also been made available in the CRAWDAD online database for other researchers to use real mobility data in their research [29].

ZebraNet made significant contributions to both the technical aspects of collar design and also the recovery of biological data. However, it also was a successful demonstration of the effectiveness of wireless network technology in the collection of animal tracking data. The project also provided valuable lessons about the real world factors that need to be considered when deploying a wireless network, both in terms of hardware and software.

A shortcoming of ZebraNet however, is the lack of scalability, as transmission slots are scheduled *a priori* [14]. Synchronization amongst multiple collars is provided by the GPS receiver [18]. This restriction also prevents the incorporation of non-GPS enabled devices into the network. In addition, ZebraNet is a completely homogeneous solution and has not considered the vast diversity in the Animal Kingdom. ZebraNet also has not considered interacting with stationary environmental sensors.

Another animal tracking project is the Electronic Shepherd [191]. This is a wireless network designed for monitoring the composition and presence of sheep and reindeer herds in remote areas. This is simpler than the ZebraNet project in that the leader of the flock carries a GPS/GPRS capable tag that detects the presence of transmit only tags from other members of the flock. Thus, this is not a true multi-hop wireless network, but rather a one-hop network.

Similar to the Electronic Shepherd is the NEAT (network for endangered animal tracking) system. In this network, animal collars (which do not share data amongst each other) send information to stationary devices. The stationary devices store the data on board (i.e. they themselves do not form a wireless network) and upload it to a basestation carried by a field researcher. The contribution of this work is that it allows data to be obtained from the wild animal, without having to be in range of it. This is essentially a two-hop network, from collar to stationary node and then to the basestation, with no routing or relaying amongst nodes.

A recent (2007) project is TurtleNet which is as the name suggests, is a network for tracking turtles [20]. A commercially available sensor platform, the Crossbow MICA2Dot [192], is used as the controller for the solar equipped GPS tracking device. As the device is solar powered, the functionality of the device has to be adaptively controlled so as to prolong its life using the eFlux energy aware compiler [193, 84]. Like the other animal tracking projects, the deployment scope is homogeneous.

Another approach has been in the design of a network protocol for the acquisition of data from whales [194]. This was based on the standard model used for the spread of epidemics through a population, treating a packet of data as an 'infection'. The different states are susceptible, infected and re-infected. This was an analytical approach to controlling epidemic routing, using the idea of a whale tracking network.

Work has also been undertaken in the area of domestic animal tracking using wireless network technology, in particular those which are agriculturally important. A system for the large scale monitoring of cattle has been presented which uses sensors (such as pedometers) to monitor cow activity and determine physiological states such as oestrus [195, 196]. As a test of their approach, cows were fitted with collars which carried mobile phones and GPS receivers [197].

A more ambitious project is one which seeks not only to monitor animal locations, but also to influence their behaviour through the creation of 'virtual fences' [198]. To influence animal motion, aversive stimuli or cues (such as auditory signals or electrical shocks) are emitted by the animal carried collar [199]. In early experiments, the tracking devices consisted of COTS (commercial, off-the-shelf) GPS devices linked to a wireless network card, but demonstrated that cows could be made to herd as a group through the application of various stimuli [199]. Custom sensor nodes, Flecks, have been designed which can be augmented with various devices such as GPS and external FLASH memory [200]. The applications of the system range from preventing bulls from fighting [201] to the prevention of over-grazing by monitoring soil condition using stationary sensors [202]. The Fleck sensors have also been used in other projects such as water quality monitoring [203].

3.2.2 Environmental Monitoring

A number of stationary environmental monitoring systems have been deployed, with varying purposes and number of nodes.

A nest monitoring system was installed on Great Duck Island [204, 205]. This acquired data on nest occupancy, as well as microclimatic indicators such as temperature and humidity. Over 150 nodes were deployed to obtain data at a high spatial resolution [205]. This real world installation provided a vast amount of information not only related to the biological application, but also the difficulties inherent in actual deployment. The authors found that transmitting periodic health information (such as node battery voltage) aided in preventative maintenance and detecting node failures [204]. A related project involved monitoring the microclimatic conditions of a Redwood tree, by placing wireless nodes in a vertical column up the trunk of the tree [206]. Their results showed the variation in humidity and radiation levels with time and height above the ground. The authors used multi-dimensional analysis to extract information from the nearly one million data points collected.

To monitor a large volume of space, such as a forest, a huge number of sensor devices would be required. In an effort to reduce the number of devices, a hybrid system has been proposed [207]. Stationary devices are deployed as before, but augmented with autonomously mobile devices. The mobile devices are suspended from a cable strung between two trees, and can vary their height and their position along the cable. Thus, these nodes can sample data over a two-dimensional plane. An adaptive sampling algorithm was proposed for accurate spatial reproduction of a sensed variable [207].

Wireless sensor networks have also been used to monitor volcanic eruptions [208]. 16 nodes relayed siesmic data through a multihop network to a central server. A time synchronisation protocol was used to timestamp the acquired data, which the authors found to be a major postprocessing task [208]. Due to bandwidth and energy restrictions, nodes do not send all their data via the wireless network, but only data of interest, corresponding to possible siesmic events. A software error caused a three day loss of communication between the network and the basestation. This network operated at a high sampling rate (100Hz) to capture tremors and eruptions, which contrasts with microclimate monitoring systems which have sampling intervals in the order of minutes or even hours [204]. Thus, node lifetime is greatly reduced. Other environmental deployments include fire detection [209, 210] and glacier monitoring [211].

Wireless sensor networks can also be used to record and localize creatures through their vocalizations. An automatic recognition system for monitoring different species of frogs and invasive cane toads is described in [41]. Low functionality sensor nodes relay the acoustic data to a high end device which performs frequency analysis and uses a machine learning algorithm to deduce the species that made the call. In this work, the authors did not discuss the possibility of localization of the frogs.

A woodpecker monitoring and localization system has been proposed [212]. Calls are monitored on wireless nodes, each equipped with four microphones in a square. The direction of arrival (DOA) can be determined from the relative phase shift of each microphone. Using multiple nodes, the position of the woodpecker can be deduced.

3.2.3 Proximity Detection

There has been recent work in the development of tags that are able to record when two individuals are within range of one another [213, 214]. The earlier work by Ji *et al.* on the 'MateID' system involved one group of animals wearing pulse-coded transmitters and another group of animals equipped with VHF recorders that logged contact ID's. A commercially available logger that acts as a transceiver, allowing more detailed contact logs to be constructed between individuals has been developed and used to study social interactions of animal groups [214]. The range of the proximity loggers can be made to vary from 0.5 m to 100 m by adjusting the output power of the transmitter [214]. These have been used to analyze contact between raccoons [214], calf-cow interactions [215] and possums [216, 213]. The main issue with these devices is that they are archival data loggers, necessitating their recovery in order to download data [214]. In addition, currently only a limited number of ID codes is supported (less than 255 [214]) and there is no provision for data from other sensors (such as GPS or temperature) to place the contact logs into context. The Ecolocate system allows tags to spot transmit-only tags as well as other Spotter class tags. In addition, in Ecolocate, the spotting data is treated as just another source of data that is carried by the wireless network. Thus devices need not be retrieved in order to download their data.

3.3 Why a Wireless Network?

Obtaining data from tracking tags deployed in the field can be a time consuming and laborious affair. In particular, archival loggers need to be recovered as:

> 'the use of bio-loggers on free-ranging animals necessitates ingenuity in the recovery of the recorder in order to download data. This had led researchers to take advantage of natural behaviours (i.e foraging trips to sea and/or return to nesting or colony sites). A limitation of such experimental protocols is the restriction of data collection to a particular season, age group or sex of animal (i.e. maternal foraging trips of female fur seals).' [183]

Moreover, for some of these systems (in particular manual VHF tracking), the presence of the researchers in the field tracking the animals causes disturbance to their natural behaviour. This disturbance is likely to introduce bias into the obtained data [8].

Some recent advances in technology have seen the deployment of a wireless communication module into the tracking tag itself. One such modality of communication is the use a UHF modem [13]. At prescheduled times, the module wakes up and listens for a command from a hand carried controller. If the command is not received, the unit goes back to sleep. However, if a download command is received, recently acquired data is transmitted wirelessly to the receiver unit. Another method of sending data involves using the GSM cellular network [13]. Animal collars periodically attempt to contact a cellular tower, whereupon they register on the network and transmit their data via SMS or GPRS. With both of these methods of wireless communication, the collars can be reprogrammed in the field, as the communication channel is bidirectional. This means that parameters such as GPS sampling rate can be adjusted mid-study if an event of interest (or non-interest, such as hibernation) occurs.

The incorporation of a two way wireless communication system enables data to be downloaded remotely[2]. However, with the UHF modem approach, a researcher needs to 'contact' each and every tracking collar in order to download the data. Although the GSM modem approach is automatic in that it downloads without intervention, it suffers from the problem that by their very virtue of being unpopulated (with humans at least), game reserves typically have very poor or no GSM reception. In addition GSM modems have a high power consumption and thus result in a heavy tracking device compared with store-on-board devices. The UHF modem method can in theory be turned into an automatic system by removing the need for a human user to interact with the collar and placing a few downloading stations at strategic locations such as waterholes.

However, this can be taken a step further from one hop system (from collar to download station) to a multihop system (from collar through multiple other collars/devices to download station), forming a wireless network of animal tracking collars. This requires more intelligence to be placed in the tracking devices, but makes acquiring data from the field trivial, as information is sent through the communications network without any human intervention. In addition, data can be collected from collars which are never in range of a downloading station, but indirectly by forwarding the acquired data through one or more intermediate devices. What is immediately apparent is that data can be acquired from any variable that can be sensed (such as temperature or water level) and not only from animal collars. Thus, the wireless network approach results in a shift from an animal tracking solution to a park wide management system. An overview of the proposed scope of the system is shown in Fig. 3.1. This demonstrates an example scenario, comprising of animal carried collars, stationary environmental sensors and vehicle based tracking devices. Data is gathered from low capability devices by higher functionality devices and relayed to the end user, who can then collect and act on the received information.

3.4 System Design Factors

In this section, the requirements for design of a wireless network for wildlife tracking and monitoring are examined.

[2]In theory and according to manufacturer's datasheets, this distance should be large enough to be practical in the field. Typical quoted distances range from 300 m up to a few km. However, in practice, field researchers have reported download ranges of less than 100 m with collars out of the box. Some even need to be within 30 m of the animal [217]!

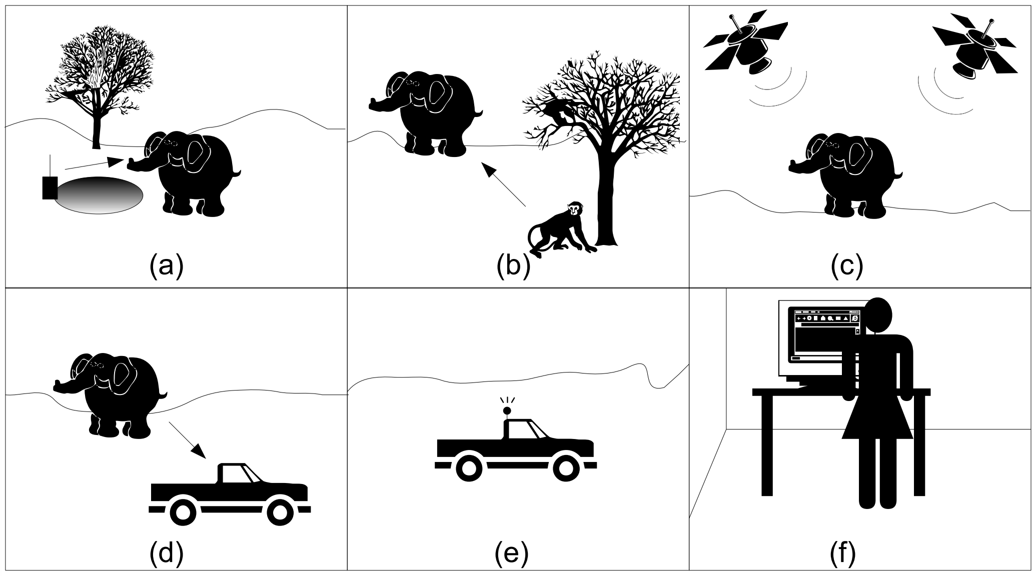


Figure 3.1: Overview of a wireless network for wildlife tracking. (a) The data from a water-hole sensor is automatically uploaded into an elephant carried collar whilst the animal is drinking. (b) The elephant walks through the reserve and passes by a monkey. The collar on the monkey contacts the elephant collar and uploads details of the animal's activity over the past three days. (c) The elephant tracking collar takes a GPS fix, determining its location. The approximate position of the monkey can be inferred. (d) A vehicle on a routine game-drive retrieves the data off the elephant collar, buffering it into its memory. (e) When within range of the park-wide network, the data collected from the reserve is upload on a long-range UHF link. (f) The park manager is able to view near-real time data on the monitor, and notices that the water-hole is running out of water. In addition, the system alerts the user that preventative maintenance should be undertaken on the vehicle.

The need for a widely heterogeneous solution

Due to the inherent diversity in the Animal Kingdom, especially with respect to body weight, a widely heterogeneous system design is proposed. This is primarily as a result of the 5% bodyweight guideline. Exceeding this guideline can result in adverse effects, ranging from discomfort and irritation to severe impacts on survivability and lifetime due to the increased effort of carrying the extra weight. The principle contribution to tag weight is the power source, which is typically a primary battery. Thus, there is an almost direct correspondence between tag power consumption and tag weight, under the assumption that tags are expected to last the same length of time.

Table 3.1 displays bodyweight, and corresponding maximum tag weight for a variety of animals. Note that whilst the African Elephant can carry over 100kg (equivalent of five car batteries), a small mammal such as a monkey can only carry a tag of weight 90g, including collar material and electronics.

Table 3.1: Typical bodyweights and corresponding tag weights according to 5% guideline (Animal weights from [218])

Species	**Body Weight [kg]**	**Tag Weight [g]**
African Elephant (*Loxodonta africana*)	4 000	200 000
Lion (*Panthera leo*)	150	7 500
African Wild Dog (*Lycaon pictus*)	20	1 000
Vervet Monkey (*Cercopithecus aerthiops*)	1.8	90

Such orders of magnitude in allowable tag weight should be exploited so that the tracking devices on smaller animals use the functionality of the devices placed on larger animals, with a correspondingly larger source of energy. This is an area which has not been investigated – to date, wireless networks for animal tracking have arrived at homogeneous system designs. The result of a homogeneous solution is that only a small section of possible animal species can be monitored using animal carried wireless tracking devices. This needs to be addressed, such that a single system is designed that is able to monitor a wide range of animal species, from small to large.

Tag longevity needs to be maximized

Tagging wild animals is a dangerous procedure, both to the animal in question and to humans. The drugs used to tranquilise wild animals (in particular, M-99) can be fatal to humans if accidental contact occurs and the antidote is not rapidly adminstered [94]. Tagging animals is also a costly procedure, especially for those species that require the use of both air and ground crews [93].

Fitting tracking devices to wild animals is not a simple procedure, and hence any device attached to an animal should last as long as possible. Currently, tracking devices operate with the same functionality over their lifetime. However, tracking devices can extend their lifetime, at a cost in reduced function, by disabling certain functions (such as power hungry GPS units).

Guaranteed eventual delivery of information is desirable

In the wildlife tracking application, latency is less of a design consideration as compared to human operated systems which need rapid response times[3]. What is important however, is eventual delivery of information. As highlighted in Section 3.4, attaching collars to animals is complicated and dangerous. Thus, all information captured by the tags should be relayed through to the end user, even if it arrives days or even weeks later.

[3]Delay times of days or months in obtaining data from animal tracking devices are not a major issue. On the other hand, waiting an hour for a web page to load would be unacceptable

This is because wildlife monitoring and management is a long-term strategy. Thus, the overall goal of the system is eventual rather than timeous delivery. This means that wireless nodes will need to have very large buffers in order to store packets without having to discard old data due to buffer overruns. In addition, it needs to be noted that this is a data-gathering network, not a peer-to-peer network. This has the implication that data can be delivered to any basestation (if there is more than one in the network). This differs from the ZebraNet project which involved peer-to-peer flooding to disseminate information to all devices in the network.

Able to handle mixed mobility

A network of this nature is not purely mobile nor purely stationary. Some nodes will be stationary all the time (such as water-hole monitors), but other nodes will have periods of motion followed by times of rest. For example, an animal such as a lion will be active during the early evening and night, with periods of rest during the day. This means that the network protocol must be able to effectively handle information transfer between stationary and (intermittently) mobile nodes in order to transfer the data to the end user.

GPS should be used where possible

The use of GPS receivers as a means of location should be used wherever possible. However, due to the high power consumption of present GPS receivers, they can only realistically be used on medium sized ($>$ 15 kg) animals and upwards for long term deployments. GPS receivers cannot be used on smaller animals as the weight of the collar required to operate for a useful length of time would violate the 5% bodyweight guideline. However, this does not mean that no technology at all should be used.

Able to operate equally well in sparse and more dense network areas

This network is for the most part sparse, due to the large geographical extent of typical game reserves, but there are areas where the network density is much higher than average. These areas will typically be around water-holes and other focal points of interest such as salt-licks. Around these areas, animals gather, often at the same time of day, leading to an increase in network density. The network must be able to operate well in both scenarios.

Network composition should be dictated by the application, not by the tracking system

The network needs to be flexible in terms of the introduction of new nodes with different features into the network and also robust to node failures through expiry or destruction. Thus, whether there are ten or a thousand tracking devices, the network should still function well i.e. it should be scalable. In addition, the network should adapt to node insertions and removals with zero configuration required from the user – the network should be a transparent method of information delivery, not a system which requires technical expertise to operate and configure.

3.4.1 Summary of requirements

In summary, these are the design constraints that are imposed on the network:

- The network structure needs to be widely heterogeneous.
- Tag longevity is an important consideration.
- Data needs to be delivered, but latency is not an overriding issue.
- The network is characterised by mixed mobility.
- GPS can only be used on larger animals.
- For the most part the network is sparse, but node density increases around focal points.
- Network composition must be flexible and scalable.

3.5 System Components

With the requirements from Section 3.4 in mind, a generic wireless network based wildlife tracking and telemetry system is designed. The highly diverse Animal Kingdom demands a widely heterogeneous solution. To this end, the wireless network is comprised of a number of classes of tags, with differing levels of functionality. The higher level nodes have the same functionality as the lower level nodes, in addition to increased features. This leads to a unified design approach, as they all run the same basic firmware. Simpler nodes run a subset of the entire code. The development cycle is consequently simplified as only one set of source code has to be maintained - the class of the tag is chosen either at design time (through conditional compiler switches) or dynamically at run time in the field (discussed in Section 3.6). There are six different classes of nodes, which in ascending order of functionality have been termed Archival Loggers, Markers, Active Loggers, Spotters, Pack and Base Tags. Although this may appear to be a more complex approach to wildlife tracking, this actually results in a modular design process, as the same basic hardware and software can be used for archival tags right through to wireless network capable devices.

The block diagram of a generic data-gathering tag is shown in Fig. 3.2. At the heart of the tag is the processing unit, whose complexity can vary according to the requirements of the tag class. Attached to the tag are sensors which record environmental data (such as temperature and GPS location) and network parameters (such as connectivity). The tag is able to communicate data through an information transfer block, the choice of which depends on the application at hand, ranging from wired connections to wireless network transceivers. As the availability of a communications interface is not guaranteed, the tag needs to be able to buffer acquired data in its memory before transferring it opportunistically to another node or end-user. Lastly, the tag requires some sort of power source so that it can operate, and depending on the application this can range from a primary battery to a solar module. Not all components need be present in any tag implementation, but this can be regarded as the 'blue-print' for a generic data-sensing tag.

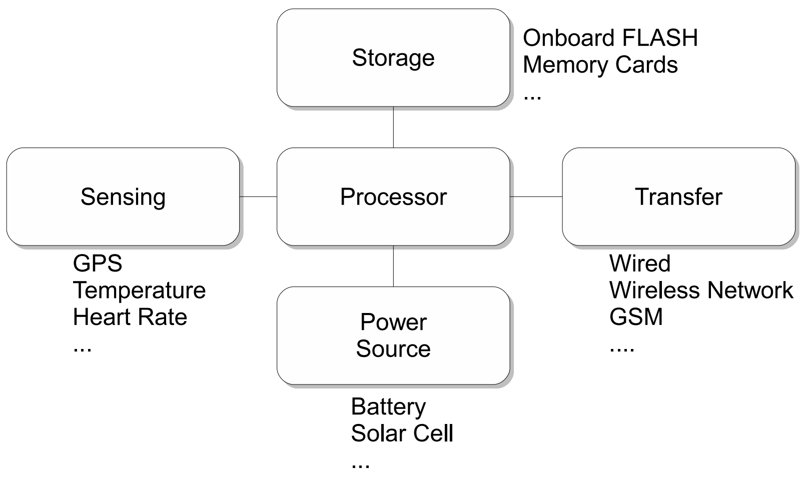


Figure 3.2: System blocks of a generic data-gathering tag. Sensed data is processed and stored in on-board memory for later transfer to other nodes or the end user.

To validate the design approach, prototypes have been constructed using PIC series microcontrollers, Nordic NRF905 radio transceivers and u-blox GPS receivers. More details on the tags and their operation are given in Chapter 7. In this Chapter, node functionality is expressed in generic terms without reference to a particular hardware or software implementation. Before the various classes of tags are discussed in detail, the manner in which the network itself discovers neighbouring nodes and transfers information is first presented.

3.5.1 Beacon protocol

Central to this work is the beacon protocol, which controls how nodes access the wireless medium and how they can discover ('spot') each other when within radio range. All nodes have access to a common beacon channel[4], which nodes access randomly. A beacon is a short packet of data[5] which contains information such as the node ID, its rank/attribute (see Chapter 4) and any other network-centric data such as sensor variables. The action of transmitting a beacon is called 'marking', as it is in some ways similar to an animal marking its territory by advertising its presence through chemical or visual signals. Beacon signals have two purposes – they function both as VHF-like beacons (meaning that existing technology such as triangulation and homing can be used) and also as network discovery or 'Hello' packets.

Beacon transmissions from various nodes are shown in Fig. 3.3. Note that nodes access the same channel randomly, which can lead to collisions (as in the figure, when nodes B and C transmit concurrently), but obviates the need for network-wide synchronization[6]. An analysis of collisions and the choice of inter-beacon periods is presented later in Section 3.7.4.

The other action that can be undertaken by nodes is to listen to beacons on the channel. This is termed 'spotting'. As beacons carry the transmitting node's ID number, when a node listens to transmissions on the channel, it can record the overheard ID's, building up a contact log, which indicates the variation in node proximity over time. In this way, nodes can each act as proximity (or logical connectivity) detectors. The data obtained from the proximity detection process can be used for various studies that are elaborated

[4]This work assumes the existence of a commonly available channel which can be used for beacon signals, specified prior to deployment. However, it is also possible to dynamically choose a beacon channel by common consensus to mitigate effects of channel fading and loss. One possible way of doing this would be to use a frequency-hopping protocol [219].

[5]Refer to Section 6.3.5 for details on packet structure.

[6]Collisions can be reduced through the use of a Carrier Sensing protocol. However, for the most part, the network is expected to be sparse (only a few neighbouring nodes at any one time), and thus collisions will be rare. Thus, using Carrier Sense in this case will detrimentally impact on node lifetime whilst providing little benefit. For example, using a Nordic NRF905 radio transceiver, scanning the channel before sending a beacon will increase energy consumption by a factor of 1.5, assuming that the channel is scanned for the length of a single beacon packet [219]. Lastly, Marker class nodes are explicitly made to be transmit-only: they do not necessarily have the ability to perform Carrier Sense.

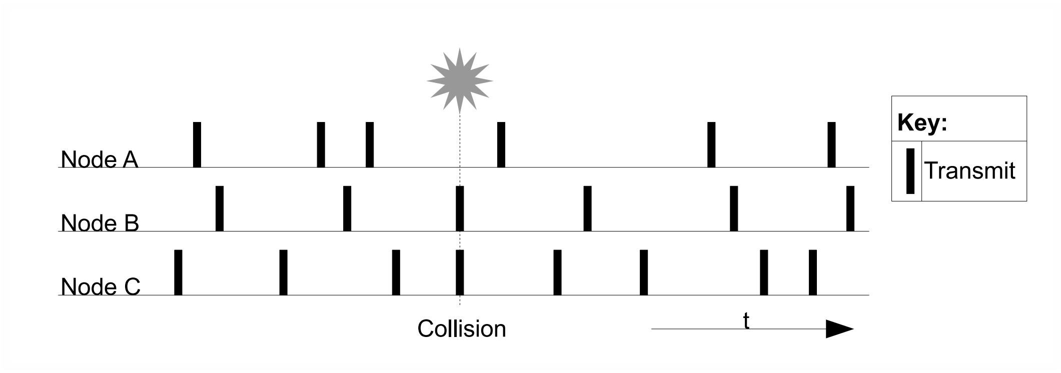


Figure 3.3: Nodes beaconing in the shared beacon channel. Note the collision between Nodes B and C when they attempt to transmit at the same time. Neither node is aware of the collision.

upon in Sections 3.7.1 to 3.7.3. The spotting process shown in Fig. 3.4 demonstrates how a node listens for other beacons and constructs a contact log. Nodes A through C transmit beacons as before, but at a random point in time Node C switches to spotting mode, where it listens to the beacon channel and records any overheard beacon ID's. At the end of the spotting interval, it compiles and stores a list of other nodes within radio proximity. Note that not all nodes have to enter spotting mode, and can act solely as Markers as before. Thus, it can be seen that nodes that only mark are a reduced function version of nodes that can mark and spot.

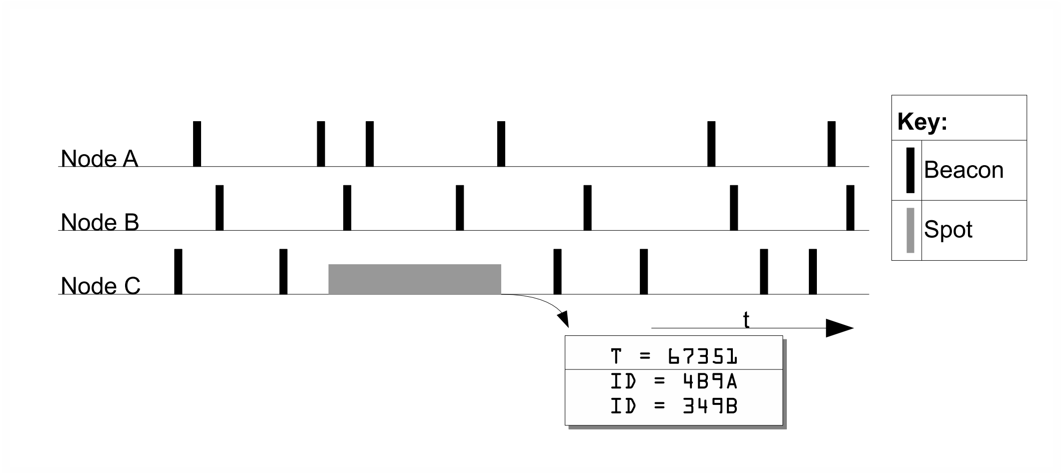


Figure 3.4: Node C spotting other nodes within its radio proximity. During its spotting (listening) procedure, it spots both Nodes A and B marking the channel with their beacons and records this within its contact log for future upload to the end user.

The other function of beacons are to act as network discovery packets to arrange data exchange between nodes. Nodes transfer data in transmission channels (termed 'Data Channels') separate from the main beacon channel. Although this may appear to be a drawback, necessitating a multi-channel capable transceiver, many modern transceivers (e.g. from Nordic Semiconductor [219] and Zigbee compatible devices [220]) are able to digitally select a different channel. The advantage of this approach is that beaconing and data transfer are entirely distinct from one another. Sending data is likely to take much longer than the transmission of a single beacon, and thus the channel is not 'locked' for the duration of the exchange. Multiple data channels can be used to further reduce the probability of collision amongst nodes transferring data.

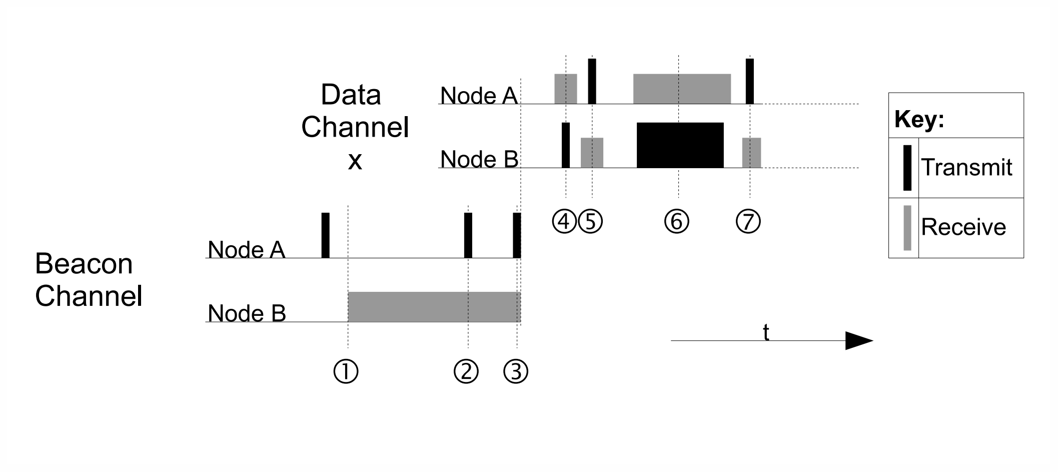


Figure 3.5: Sequential diagram showing steps in setting up a data-exchange. Refer to the text for an explanation of the various stages.

The steps in setting up a data-exchange are as follows (refer to Fig. 3.5 for a graphical representation of the procedure). The stateflow diagram of the processs is also shown in Fig. 3.6.

1. The node with data to transmit (in this case, Node B) enters spotting mode. It scans the beacon channel, detecting beacons from nodes within its neighbourhood. These beacons contain metrics which indicate the usefulness or health of the node as ranked by the ASH protocol (refer to Chapter 4 for details on ASH). Thus, Node B will scan the channel for a length of time and determine which is the best node (in terms of data delivery and reliability) to send their data to. This is why they will not send data to the first node that is heard, but rather decide which node is the best for data transfer. Nodes also use the received beacons to update their ASH parameters.

2. Node A transmits its beacon, advertising its availability for data transfer. At the end of the spotting window, Node B decides to send data to Node A. (Node B will also update its contact log at this point, indicating that Node A was spotted).

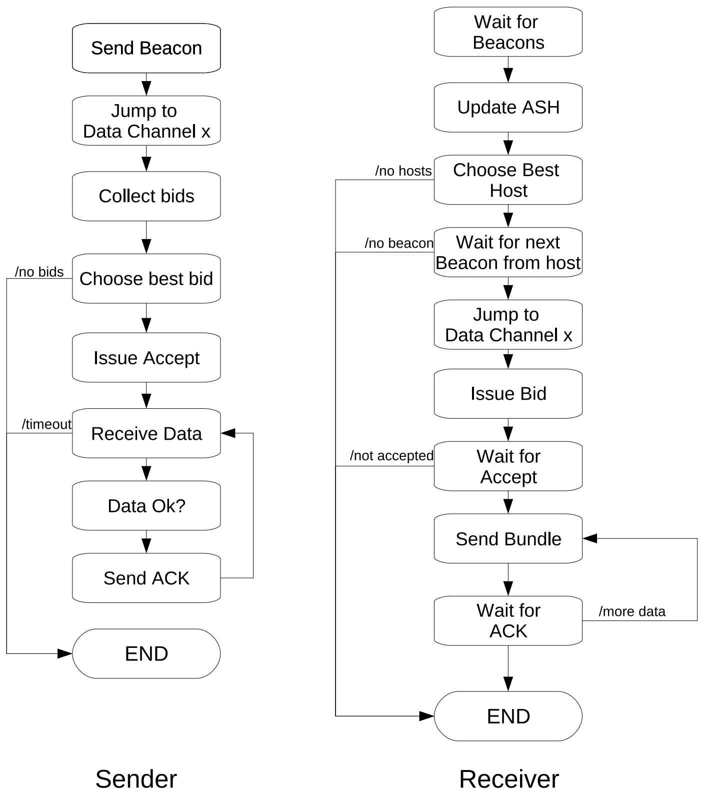


Figure 3.6: Stateflow diagram showing steps taken in sending data using the Beacon Protocol.

3. Node B remains in spotting mode until it hears the next beacon from Node A, which instructs any receivers to jump to Data Channel x to commence the data exchange[7]. Thus, the node which is accepting data transfer (in this case Node A), controls which Data Channel to use. This allows the receiving nodes to pick a data channel which is not being used by other nodes within radio range.

4. Nodes A and B jump to Data Channel x. Note that Node A will always jump to the Data Channel and wait for any other nodes to contact it to arrange a data transaction. If no nodes attempt a contact, it will leave the Data Channel and go back to sleep mode. Node B sends a 'bid' packet, which is an expression of interest in uploading data to Node A[8].

5. Node A sends an 'accept' packet to Node B, instructing it to commence data transfer

[7]In the event of Node B hearing multiple beacons from network capable nodes, Node B will choose one of the nodes as the destination. How this is done depends on the network protocol itself, but one option could be to choose the 'best' node in the interval, as reflected by its ranking (refer to Chapter 4). Another option is to choose a random destination, with a bias towards better nodes.

[8]To minimize collisions in the event of multiple nodes simultaneously bidding, each node sends its bid with a random delay.

[9].

6. Node B sends a bundle of data.

7. Node A issues an acknowledgment, indicating successful reception of the bundle. A new bundle may now be sent if more data is buffered or Node B may return to normal mode. Node A will stay on the Data Channel and wait for more data from Node B. It leaves this mode when it receives an 'end-of-bundle' message or if no data is received within a certain time-period.

The advantage of this protocol is that there is no network data at all in the beacon channel, leaving it clutter-free. In addition, the choice of a random data channel also reduces the probability of collision with existing data transfers. Nodes that are unable to receive data will leave the channel advertisement in the Marker packet blank – a receiving node will thus be able to determine that the transmitting node is not a candidate for a possible data transaction. Once again, this demonstrates that nodes that only perform spotting and marking are a subset of fully network capable nodes, highlighting the modular design approach.

This overview of the beacon protocol shows that there are three main functions that nodes undertake:

1. Marking or transmitting a beacon

2. Spotting or detecting beacons within range

3. Transferring data to other nodes

The manner in which data is transferred or routed between nodes is beyond the scope of this Chapter on system design and is presented in Chapter 4.

The various classes of nodes arise from the functions that they are able to perform. They are now introduced with reference to their allotted functionality. The four main classes are Markers, Spotters, Packs and Base tags, but there are also two types of loggers, Archival Loggers and Active Loggers, which have a greatly reduced role in the network, but share features common to the other classes.

[9]Similarly, if multiple bids are received by Node A, it will only accept one. Unlike the bidding process, it would be sensible to accept the bid from the worst node, so as to conserve its resources.

3.5.2 Marker Tags

These tags just act as beacons, periodically emitting ('marking') their ID and other salient information, such as temperature and movement parameters in a packet[10]. Marker tags never listen for other nodes' IDs and consequently consume a miniscule amount of power as they spend the majority of their time in low power sleep mode. Thus, they can be regarded as analoguous to standard VHF tags, albeit digital. The system components of a Marker tag are shown in Fig. 3.7. Note that although the Marker tag is equipped with a transceiver, it is only operated in transmit mode and never switches to receive mode. Marker nodes can be powered from small batteries, resulting in a lightweight and small package size. Marker tags are inexpensive due to their simplicity and suitable for tagging a wide variety of animals, from small to large.

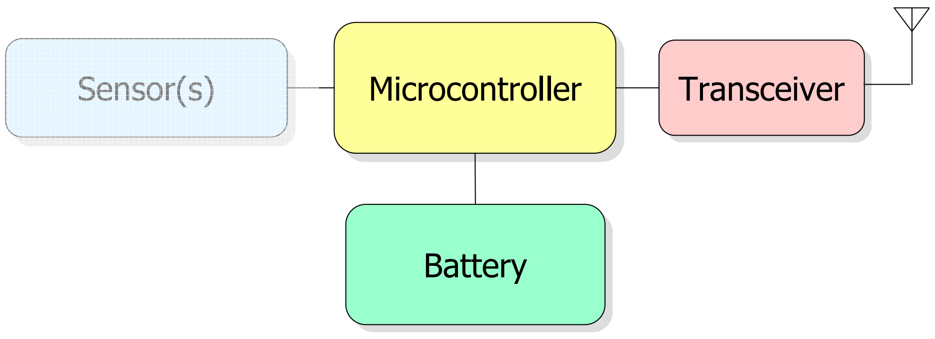


Figure 3.7: System overview of a Marker tag. The transceiver operates only as a transmitter.

3.5.3 Spotter Tags

These tags provide all the functionality of the Marker nodes but periodically 'spot' or listen to other nodes within their radio range. They store the overheard node IDs and any other transmitted data in memory, along with a timestamp. When in range of a Pack or Base station tag they transfer the stored information through the wireless network. However, they only act as leaf or end nodes in the network and do not route other nodes' packets. Their power consumption is higher than the simpler Marker type tags, as they have to remain awake in active receive mode when spotting. The length of time that they

[10]Data messages that are too large to fit in a single packet can be segmented and repeatedly transmitted at random for later assembly. Alternatively, digital fountain codes can be used which allow the message to be recreated if a sufficient number of encoded packets are received [221].

are required to be listening for markers is twice as long as the average time between beacon transmissions. In this way, they are guaranteed of spotting all nodes within radio range, excepting those whose markers collide. The block diagram of a spotter node is shown in Fig. 3.8. The memory requirements of Spotter class nodes vary depending on their sensing capabilities and the number of nodes in the network as well as the typical interval between successful uploads.

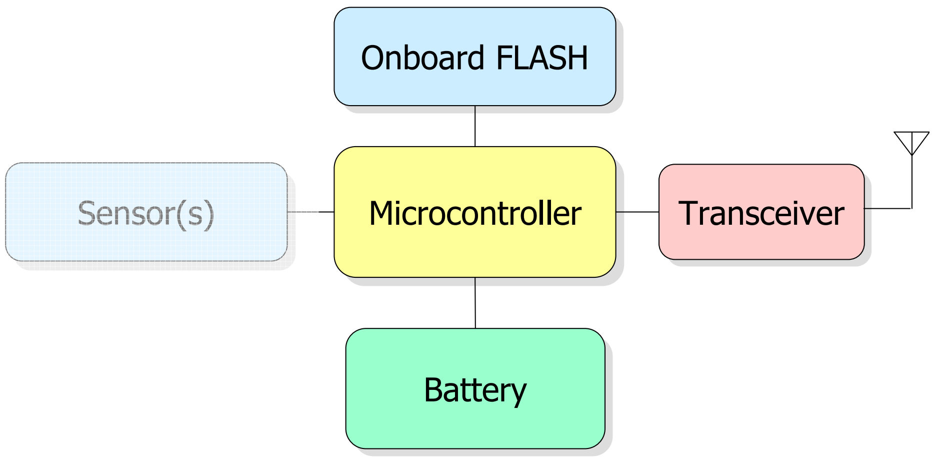


Figure 3.8: System overview of a Spotter Tag. Observed packets are stored in the micro-controller's on-board FLASH memory, for later injection into the network. In a Spotter node, the transceiver acts as both a transmitter and receiver.

3.5.4 Pack Tags

Pack tags form the multi-hop network itself. They perform all the tasks of Spotter nodes and also route information through the wireless network in a store-and-forward fashion. As the radio radius is small compared to the total area, the network is very sparse and thus information is transferred opportunistically upon contact with another Pack tag or Base station. The block diagram of a Pack class tag is shown in 3.9. In this case this tag has been equipped with a triaxial accelerometer and a GPS receiver. Pack tags have large memory requirements, as they have to store not only their own data but also data from other nodes within the network, without loss due to buffer over-runs. Pack tags have high energy requirements, as they not only perform marking and spotting tasks, but also relay and route large volumes of data. Pack tags are most suitable to place on medium to large animals.

Pack tags transmit a special beacon packet which contains their ID and information which other tags use to update their ranking within the network. Thus, the beacon packet acts both as a proximity locator for the purpose of animal tracking and also as

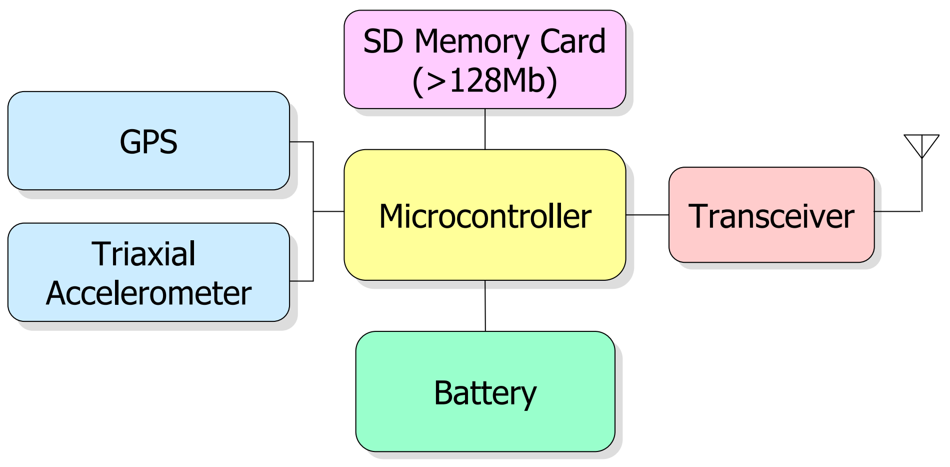


Figure 3.9: System overview of a Pack Tag. These are full functioned network tags and store all data in SD memory cards. This Pack tag has been equipped with a GPS receiver and a Tri-axial accelerometer.

a network control or discovery packet. Pack tags also transmit a random data transfer channel that pack tags and other nodes can use to initiate a data transfer. To decide how to route packets, nodes assess their ranking (the ranking is similar to reputation schemes in wireless security) in terms of the global distribution of resources (such as remaining battery energy or connectivity) using local information. This is dynamically assessed, and thus if new nodes (with large amounts of battery energy) are inserted into the network, these new nodes will assume a higher rank and participate more fully in the processes of routing, removing the load of routing from nearly exhausted nodes. The details of determining a node's rank within the network are discussed in full detail in Chapter 4.

3.5.5 Base Tags

These nodes act as sink nodes in the network and provide an interface between the wireless data gathering network and the end users. They are essentially Pack tags, except that all received data is forwarded out of the network via some other communications interface. This interface can either be wired or wireless, depending on the application. Base stations can be placed at convenient sites (such as at the top of a hill) or at points of attraction for animals (such as waterholes or salt licks). Base tags can be mobile, and these can be carried by people (game-rangers or tourists), attached to vehicles or affixed to large animals, such as elephants. The choice of the communications interface depends on the local infrastructure, but can be cellular GSM modem, UHF modem or even satellite upload. As the communications interface is long range, its power consumption is typically large. Thus, Base tags should be equipped with a solar panel or attached to a source of mains power if possible.

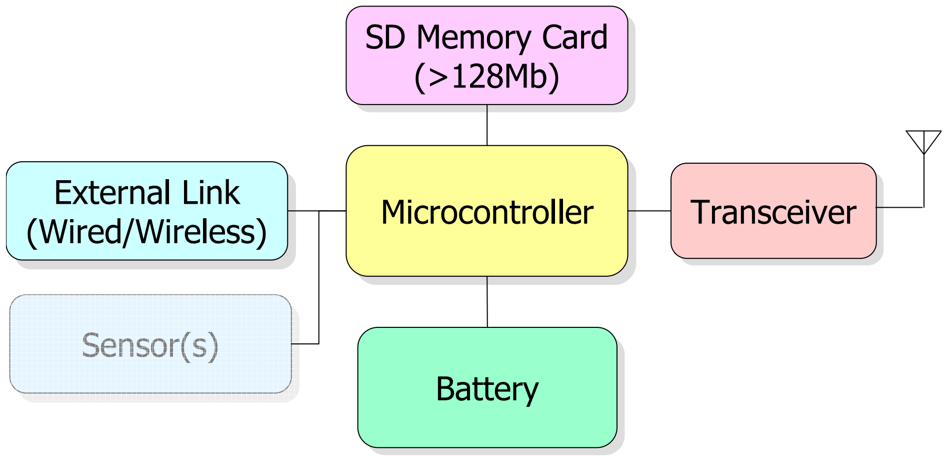


Figure 3.10: System overview of a Base Tag. These tags have some link to the end-user, either wired or wireless. Base tags are basically Pack tags with an external interface.

Base nodes can also be equipped with directional antennas. This will increase the accuracy of location estimates, as the bearing to a beaconing node can be determined. In addition, fixed Base tags can be equipped with more sensitive radio receivers and larger antenna, which will increase the range at which beacons can be detected. Thus, many of the well established techniques of conventional VHF tracking (such as automated triangulation) can be carried over to this system.

3.5.6 Active Loggers

Active Loggers are tags which neither mark nor spot, but inject packets into the network when possible. They acquire and store sensor data, and contact Pack or Base tags to transfer the data out of the tag. As loggers can generate vast amounts of data, depending on the wireless interface, the device may not be able to download the entire dataset due to power constraints or link longevity. Summary and compression techniques can be used to limit the download volume, with indicator data being downloaded through the wireless network and the bulk of the data downloaded upon tag recovery. Thus Active Loggers reduce the possibility of complete data loss by transferring salient data to the end-user. The power consumption of the Active Loggers depends on the type of sensors attached, but is greatly reduced as no periodic marking or spotting operations are undertaken. In Section 8.3.1 a method is sketched which schedules upload attempts based on prior success and correlations with sensor data. A block diagram of an Active Logger tag is shown in Fig. 3.11. An example scenario of where an Active Logger could be used is in locomotion monitoring of an animal. The detailed acceleration data is stored on-board for later retrieval, but summary statistics (for example of the maximum acceleration deviation over a time period) are uploaded through the rest of the network.

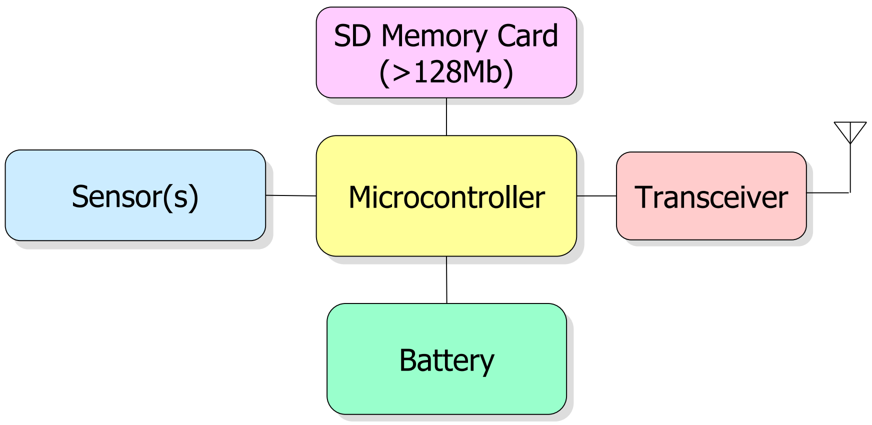


Figure 3.11: Block diagram of Active Logger Tag. Although very similar to spotter nodes in terms of capabilities, these tags have large memories and do not spot other nodes.

3.5.7 Archival Loggers

Although not strictly part of the wireless network, Archival Loggers are simple data-storage devices that share a great deal of common functionality with the other classes of tags. Archival loggers need to be retrieved for the saved data to be downloaded through a communications interface which can be wired or wireless. This leads to a higher possibility of complete data loss through tag loss, failure or destruction. However, as Archival Loggers do not need a wireless communications interface, their power consumption, and consequently their size and weight, is much smaller than the other classes of tags. As the Archival Loggers are not limited by the speed of the communications interface with respect to the amount of data that can be transferred out of the tag, they can be used to store vasts amounts of detailed data, such as constant logging of tri-axial acceleration or still video images. A block diagram of a typical Archival Logger is shown in Fig. 3.12. This unit has been equipped with a tri-axial accelerometer and a large non-volatile memory. Data is downloaded from the device via the communications interface.

3.5.8 Overview of system classes

Although having multiple classes of tags may appear to introduce unnecessary complexity to the design of the system, there is a common sequence of operations that need to

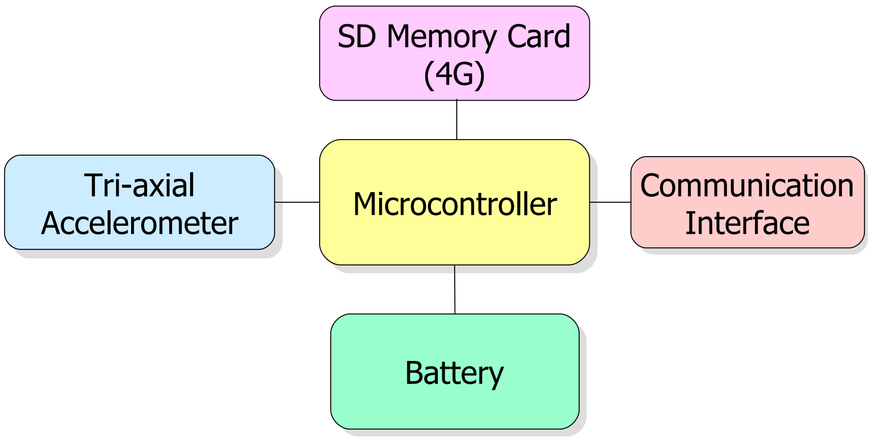


Figure 3.12: Block diagram of Archival Logger. Note that there is not necessarily a wireless transceiver.

be performed, namely: sense, store, send. An overview of the various functions implemented for each tracking class are shown in Table 3.2. Also shown is the typical power consumption and application domain for each type of tracking device. Power consumption is strongly dependent on the sensing options chosen, as well as network functions. Archival Loggers are data-storage devices with no network capability that would typically be used in obtaining vast amounts of data about the activities of an animal. They have no network function at all, and consequently their expected power consumption is low. Active Loggers are Archival Loggers with the ability to send ('inject') data into the network. Active Loggers thus are leaf or end-nodes within the network. Markers are transmit-only tags that periodically emit beacons that other tags can detect. Marker tags are the analog of VHF tags. Spotter tags are tags which transmit beacons, but also detect tags within radio range, storing this time-stamped information for later injection into the network. Spotter tags do not route or relay data. Pack tags are proper network tags *per se* as they can inject packets into the network, but also route and relay data from other nodes. As such, their expected power consumption is high relative to the lower classes of nodes. Lastly, Base tags are gateways to the real-world. Any packets which are sent to them are relayed to the end-user. Depending on the type of transmission link to the end-user, power consumption can range from high to very high.

3.6 Dynamic Role Adaption

The power consumption of the various classes of nodes varies according to the network load, the type of transceiver used and the sensors that each tag is equipped with. How-

Table 3.2: Functionality of the various classes of tracking devices within the network, demonstrating implemented functions and typical power consumption and applications.

	Functionality					Typical	Typical Applications
Tag Class	**Marking**	**Spotting**	**Data Injection**	**Data Routing**	**External Interface**	**Power consumption**	
Archival Logger					✓	Very Low	Long term/high volume data-logging
Active Logger			✓			Low	Long term data-logging with periodic upload of summary data
Marker	✓					Low	Small, inexpensive tags for marking presence
Spotter	✓	✓	✓			Medium	Proximity studies, leaf nodes
Pack	✓	✓	✓	✓		High	Full functioned sensing, routing and replication
Base	✓	✓	✓	✓	✓	Very High	Gateway between network and the real-world

ever, there is a strong relationship between the tag complexity and its corresponding power consumption. For example, a Pack type node with GPS would consume dramatically (typically an order of magnitude) more power than a Marker node. This suggests that as a high level node ages, it should dynamically disable certain functions in order to prolong its lifetime at the cost of a reduction in functionality. Thus a node may start off as a Pack node, and when it has consumed 80% of its initial supply of energy, it can decrease in functionality and become a Spotter node[11]. As it consumes more of its remaining energy, more functions (such as routing or sensing) are disabled until it eventually acts as a simple Marker class node[12]. Although the functionality of a Marker node is signifcantly less in comparison to a Pack type node, it allows the tag to participate for longer in the wireless network, satisfying the design requirements. To disable various functions is trivial, as the Pack type tags have all the functions of simpler tags. In this implementation, a Finite State Machine (FSM) approach to the firmware operation has been used, and disabling functions is as simple as restricting the possible state universe, dynamically according to the remaining tag energy. More details on this are given in Chapter 6.

As justification of this approach, consider if a Pack type node uses 10 units of energy per day, a Spotter type node uses 2 units and a Marker uses 1 unit of energy per day[13]. Assume that a Pack type node has 1000 units of energy to start with. Now, without dynamic role switching, a Pack type node would be expected to last 100 days before expiring. If dynamic role switching is used, such that once the node has used 80% of its energy it drops to being a Spotter node and once it has used a further 10% of its total energy it drops to acting as a Marker node, this node would be expected to be active in

[11]Functionality can be disabled based on the absolute amount of remaining energy, or, using the ranking method of Chapter 4, in relation to the amount of energy in the collar relative to its peers.

[12]One option is to explicitly disable functions. Still another is to reduce the duty cycle of operation, especially for power hungry sensors.

[13]As justification of these ratios, consider the values shown in Table 6.4, which show current consumption of 0.11 mA for a Marker node, 0.43 mA for a Spotter node and 1.77 mA for a Pack tag. The ratio between these current drains is 1:4:16 which compares well to the example ratios chosen.

the network for

$$\frac{0.8 \times 1000}{10} + \frac{0.10 \times 1000}{2} + \frac{0.10 \times 1000}{1} = 230 \text{ days} \tag{3.1}$$

which is more than a two-fold increase in the effective lifetime in comparison to a normal Pack tag. This is demonstrated graphically in Fig. 3.13. The graph clearly demonstrates that dynamic role assignment results in greater tag longevity. Why this approach is justified in the wildlife tracking scenario is a result of the cost of deployment, both in terms of labour and logistics. In addition, the possibility of harm to the animal also has to be taken into account. Although the data that can be obtained from the collar is much poorer, it is better than no data at all. A further reason why this approach is valid for this application domain is that an animal tracking network would typically be deployed incrementally, due to the difficulties inherent in tagging animals. Thus, as new nodes are inserted into the network they take a greater responsibility in terms of packet routing and delivery than older nodes.

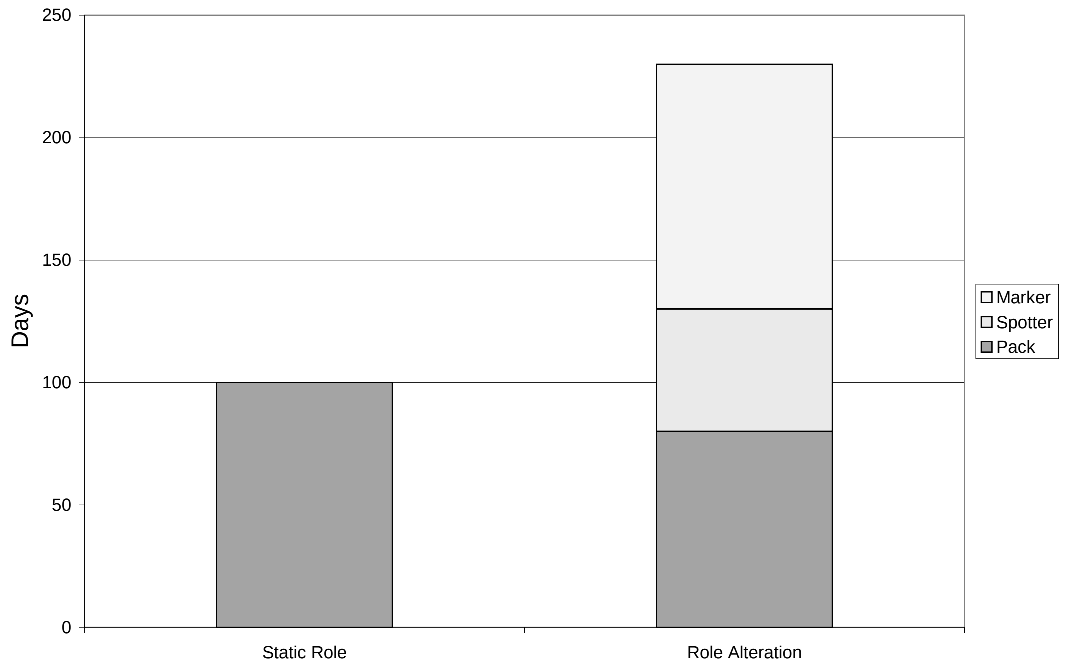


Figure 3.13: Time spent in the different roles. Although Static Role assignment results in a longer time in the higher function Pack role, note that dynamic role alteration results in an overall greater lifetime albeit at lower capacity.

Base tags, which are treated in conventional wireless networks as 'super-nodes' with infinite energy can also be subject to dynamic role alteration. This is because in general the link to the end-user is power hungry (such as a WiFi or GSM connection) and thus a Base tag will consume energy at a greater rate than a node with a lower role. Thus, Base tags can also drop their role as their energy is consumed. However, this must be undertaken with caution, as a case may occur when no tags capable of acting as Base tags actually enable their link to the external world. As Base capable nodes can be perceived as having a direct link to the central server, the server can perform the task of managing the Base tags. Before a Base tag switches to a lower role, it queries the server and determines if it is allowed to drop its role. If there are not enough active Base tags in the network, the server will deny the request. To prevent a situation of network isolation from the server due to Base tag failure or exhaustion, Base capable tags that have dropped to a lower role periodically requery the server in order to determine whether they are still allowed to be inactive in forwarding packets out of the network. As the server knows how many nodes are active in Base station capacity, it can instruct nodes to either remain in their lower role or adopt their prior functionality as a Base station. The server can also inform a human user if it appears as though there will be a shortage of Base stations in the near future, based on projected node lifetime. The user can then take steps to introduce more Base stations into the network to maintain performance.

It should be noted that role switching does not always have to be in the direction of decreasing functionality – for example a solar powered node may drop its role from a Pack node to a Spotter node as a result of being covered in mud. When the mud is washed away, the batteries will recharge and the node can adopt its previous role in the network. Thus, the role switching can be performed dynamically in accordance with changing conditions that the node is subject to.

3.7 The uses of beacons in the wireless network

As discussed earlier in Section 3.5, all nodes (with the exception of the two logger variants) transmit ‘Markers’ or digital beacons that serve to both advertise presence and also are used for network discovery and control. Nodes that are able to spot (namely Spotters, Packs and Base tags) randomly wake up and scan the beacon channel, recording the ID’s of any nodes within range. These contact logs are time-stamped, so the proximity of nodes at various times can be analyzed. Essentially, any node equipped with spotting capability acts not only as a transmitter but also an intelligent receiver, opening up many interesting research avenues. In this section, how these marker beacons can be used in the context of the wildlife tracking application is discussed. Section 3.7.4 examines how the inter-beacon interval can be determined, to trade off energy drain between Marker and Spotter class nodes.

3.7.1 Determining social behaviour and relationships through proximity detection

The social behaviour of animals, how they organize and form groups is an area which is increasingly being studied. According to Krause *et al.*

> 'Social network theory[14] has made major contributions to our understanding of human social organisation but has found relatively little application in the field of animal behaviour.' [223]

For the most part, this is possibly due to the difficulty in obtaining detailed network information. Existing methods of obtaining data on proximity of individuals are either time consuming (e.g. visual identification of individuals daily [224] or repeated trapping of badgers [225]) and/or limited in their resolution (e.g. hourly sampling rate of GPS collars [226]). A proximity type datalogger has been designed and used to characterize the contacts between individuals of a population but suffers from the same problem as all archival dataloggers in that it needs to be recovered at the end of the study [213, 214]. Social structure, in particular the way fission-fusion societies coalesce and split, are thought to have a strong impact on the rate of spread of infectious diseases through wild animal populations and thus there is a need for accurate contact data between individuals so as to adequately control epidemics [227].

Spotter class nodes are able to determine connectivity[15] and relay the obtained information through the wireless network. In addition, other sensor data such as activity or GPS obtained position can be used to add context to the connectivity map. Social behaviour logging is not restricted to a single animal species and can be used to monitor inter-species interactions, such as between predator and prey. Spotters can also be used to log human-animal interaction, whether it be locals living on a park boundary or tourists within the park. The simplest Spotters are low-cost, lightweight devices as they are nothing more than a microcontroller and a UHF transceiver, allowing them to be fitted on a wide range of species. Spotter class nodes can also detect the presence of Marker class tags, allowing social studies to be undertaken on animals that are too small to carry a Spotter tag.

[14]Social network theory is a branch of sociology that studies the way human societies form and organize. A famous example of the use of social network theory was the experiments of Milgram in mapping the number of intermediaries between two individuals, leading to the term 'six degrees of separation' [222].

[15]Depending on the type of transceiver used, it may also be possible to infer the approximate distance between nodes based on the received signal strength or time-of-flight.

3.7.2 Visit frequency to focal points of attraction

The presence or absence of animals within various areas is also of importance for animal behaviour studies. For example, a Spotter class node placed at a waterhole could monitor how often different species of animals visit, and with the incorporation of a temperature sensor, could also determine the variation in visit frequency with average ambient temperature. Such data could be used for management decisions about whether or not an artificial waterhole should be created, and when it should be supplied with water [228]. Another example would be to monitor the proximity of parents to their young (in a lair or nest), so as to determine parenting roles and how they are shared. Human settlements can also be regarded as focal points of attraction (or repulsion in some cases). The frequency of visits of various animals to these locations can be recorded and used to assess the level of human-animal interaction.

3.7.3 Proximity detection: Coarse Location Estimation

Information from Spotters, coupled with location data can be used to determine the coarse location of nodes that transmit beacons. Location data can be obtained from GPS enabled nodes (such as Pack tags) or from fixed tag locations (which can be any class of tag that transmits a Marker). This can be integrated with the contact logs to provide coarse location capability to tags which do not have GPS receivers. This is shown in Fig. 3.14, where the position of a low functionality node (such as a Marker or Spotter) can be determined by knowing the locations of one or more other nodes, and the typical radio range.

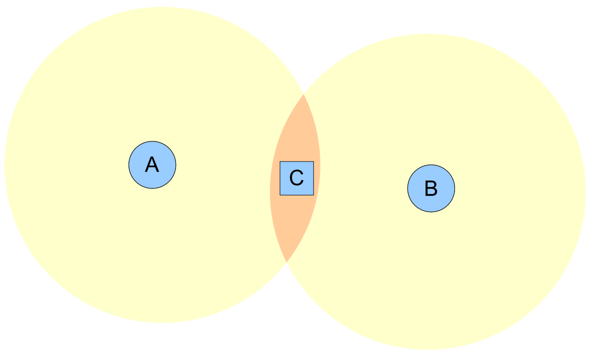


Figure 3.14: Inferring the approximate location of a Marker or Spotter class device (device C), by knowing the positions of one or more other nodes (devices A and B). The intersection of the circles determines the region of possible locations where the tag could be.

Spotter type nodes can also be equipped with directional antenna, either fixed or rotating. The contact logs from these nodes will incorporate angular information, increasing the resolution of the detection process. Thus, conventional VHF techniques such as automatic triangulation can be carried over into this system. Co-operative localization techniques can also be used to refine the accuracy of location estimation if the network is sufficiently dense [229, 230]

3.7.4 Choosing an inter-beacon time

In this section, the choice of an inter-beacon time is discussed, that satisfies the requirements of the study and results in a lightweight package. Clearly, the larger the interval between transmissions, the lower the power consumption of the transmitting node, and the longer its lifetime. In addition, the longer the inter-beacon interval, the lower the probability of packet collisions. This suggests that a large inter-beacon interval would be desirable. However, to scan the neighbourhood and record the ID's of all the nodes within radio proximity will take a correspondingly longer time for Spotters, consequently increasing their power consumption. Thus, there are conflicting constraints on the choice of the inter-beacon interval.

To choose the average inter-beacon interval, there are some factors that need to be specified by the user. We define the beacon frame length to be I seconds and assume that nodes transmit a beacon at random within each interval, with a uniform distribution. The first factor is how often Spotter tags should switch to spotting mode and detect nodes within range[16]. This is application dependent and can range from seconds to hours according to the expected time varying change in the local neighbourhood. Spotter nodes enter spotting mode at random, with a uniform probability distribution and an average time between spotting slots of L seconds. What also needs to be specified is the expected maximum number of nodes in the neighbourhood that spotter nodes should be able to correctly record. This places a lower bound on I, which will be derived in the following section.

For a Spotter node to be guaranteed of detecting a node marking its presence on the beacon channel at least once during a spotting interval (assuming there are no collisions), the length of the spotting interval needs to be $2I$[17]. Hardware specific factors include the

[16]Spotter nodes can be made to scan the channel continuously, at the cost of a high power consumption. As these nodes will never transmit, they can be regarded as passive observers as they do not alter the way the network operates in any way. Such nodes can be placed around focal points of interest in order to log the ID's of any nodes within range at all times.

[17]This is because the maximum possible interval between two successive beacons from a particular node is $2I$. This will occur if during the first beacon-frame, the node transmits at the beginning of the frame and in the next frame transmits right at the end. Thus the total interval between these successive transmissions is $2I$, as the length of the beacon-frame is I. For a Spotter node to guarantee the detection of this extreme case, the length of the spotting interval must be made equal to $2I$. This assumes that the length of the beacon packet is negligible in comparison to the length of the window, which is generally the case.

current[18] consumed whilst transmitting i_T, drawn whilst receiving i_R and current drain whilst in sleep mode i_S. We shall derive expressions for the average current consumption of Markers and Spotters using these parameters. In addition to the current drain, the choice of hardware also dictates the time that the transmitter is 'on air' and actively transmitting a packet. This is denoted as t_T, and in this formulation we discount startup and settling times. The timing and definition of symbols used are shown in Fig. 3.15

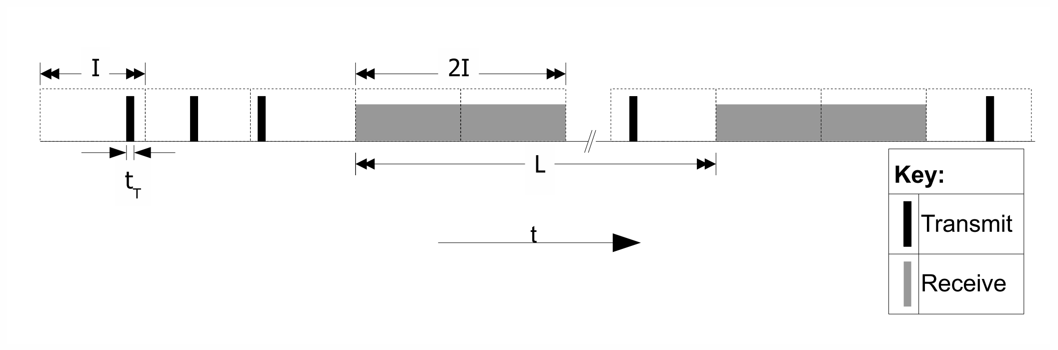


Figure 3.15: Timing diagrams for the actions of marking and spotting. Once every beacon-frame, which has a length I s, a node will mark its presence, staying on the air for t_T s . When it switches to spotting mode, it will receive packets for a duration of $2I$, so as to ensure correct recovery. Spotting mode is entered on average every L s

The average current consumption of a marker node will be

$$\bar{M} = \frac{t_T}{I} \cdot i_T + \frac{I - T_t}{I} \cdot i_S.$$

Note that t_T, i_S and i_T are parameters inherent to the particular hardware deployment, the length of the packet and the transmission speed. Thus, the only parameter that can be altered for Marker nodes is I, and the larger I is, the lower the overall current consumption for the Marker node. However, in the limit for large I, the current drain will tend towards i_S. Thus there is a diminishing return in increasing I.

The mean current consumption of a Spotter node can be expressed as

$$\bar{S} = (1 - D_r)\bar{M} + D_r \cdot i_R,$$

where $D_r = \frac{2I}{L+I}$ is the duty cycle of the spotting process. For small I, D_r will be small and consequently the current draw of the Spotter node will actually be dominated by $\bar{M}$, the marking process. As I is increased, $\bar{M}$ will fall, decreasing the current consumption of the Spotter node. However, as I is increased, the contribution from the spotting

[18]Note that it is also possible to use the following analysis to calculate the average power consumed (as opposed to average current drain) by substituting the power consumed whilst transmitting, receiving and sleeping into i_T, i_R and i_S respectively.

process will rise increasing the current consumption of the Spotter node, in an approximately linear manner.

As stated previously, a lower bound is placed on the choice of I according to the maximum number of tags that can be expected to be detected in the neighbourhood, accounting for collisions. To analyze the probability of collision with increasing number of nodes N, we define the transmitter duty cycle as $D_t = \frac{t_T}{I}$. For a given D_T, the number of collisions increases with N up to a point when few beacons can reliably be discerned from the channel due to frequent packet loss.

If we assume that traffic is uniformly distributed, it follows that the probability of collision will take on a Poisson distribution [231]. The channel occupancy is $D_T N$. Because access to the medium is unsynchronised, nodes can start transmitting at any time. This situation is essentially the same as ALOHA [231], giving the average number of nodes successfully heard in the neighbourhood as

$$\bar{N_R} = Ne^{-2\frac{t_T}{I}N}.$$

In order to calculate the minimum length of the beacon interval, we need to choose the percentage of nodes in the neighbourhood that need to be successfully heard on average in order to build a reliable connectivity map. For the wildlife tracking scenario, characterised by high link failure rates due to tag orientation and topography, a success rate in the region of 75% would probably be acceptable.

Thus, if we wish to hear a fraction $k < 1$ of the transmitted beacons, we solve the following equation

$$kN = Ne^N e^{-2\frac{t_T}{I}N}$$

for I, giving us the requisite minimum interval as

$$I = \frac{2Nt_T}{-\ln(k)}.$$

As an example calculation, it is given that there are a maximum of 50 nodes that can be expected to be within radio range, and it is required in this case that Spotter nodes be able to detect 75% of these nodes. In addition, Spotter nodes should spot their neighbourhood every 5 minutes. These and other simulation parameters are shown in Table 3.3.

The results from the simulation are shown in Fig. 3.16 demonstrating how the current consumption varies with increasing I for both Spotter and Marker class nodes. It can be

Table 3.3: Simulation Parameters for determination of inter-beacon interval

Parameter	**Symbol**	**Value**
Transmission Time	t_T	6 ms
Spotting Interval	L	300 s
Maximum neighbourhood	N	50
Detection Accuracy	k	0.75
Transmit current	i_T	30 mA
Receive current	i_R	10 mA
Sleep current	i_S	60 μA

seen that for small I, the current consumption of both Spotter and Marker class nodes are virtually identical, as the current drain is dominated by the act of transmission. As I is made larger, the current consumption of Marker class nodes decreases downwards toward to the current consumed in sleep mode. Conversely, as I is made larger, the current consumption for Spotter class nodes starts to rise due to the contribution of the spotting process. Also shown on the graph is a shaded region where the detection accuracy of 75% is not satisfied.

The choice of I value is dictated by the application, but should strike an acceptable compromise between the current consumption of the Marker and Spotter class tags. For example, with $I = 2.5$ s, $\bar{M}$ = 134 μA and $\bar{S}$ = 295 μA, making the power consumption of Spotter class tags approximately 2.2 times as large as Marker class tags. However, if I is increased to 5 s, then the current consumption of Marker class tags falls to $\bar{M}$ = 95 μA at the cost of increasing the Spotter class node's current consumption to $\bar{S}$ = 420 μA, resulting in a factor of 4.4 difference between their consumption. Depending on the allowed sizes of Marker and Spotter class tags, a compromise value of I needs to be determined which results in acceptable lifetimes for both classes of nodes.

Fig. 3.17 shows the relationship between the current consumed and the beacon detection probability. It can be seen that requiring a high detection rate dramatically increases current consumption for the Spotter node as the length of the spotting interval becomes large.

3.7.5 Synchronizing Timestamps in the Network

A large body of research in wireless networks concerns the synchronization of timestamps to a global timebase [235, 236, 237]. In the EcoLocate system, the level of time-synchronization required depends greatly on the the application requirements. As an example, if camera-traps are used to validate system operation, relatively precise (within a few seconds or less if acceleration monitoring is undertaken) synchronization will be required to correlate observed behaviour with monitored information. However, if the

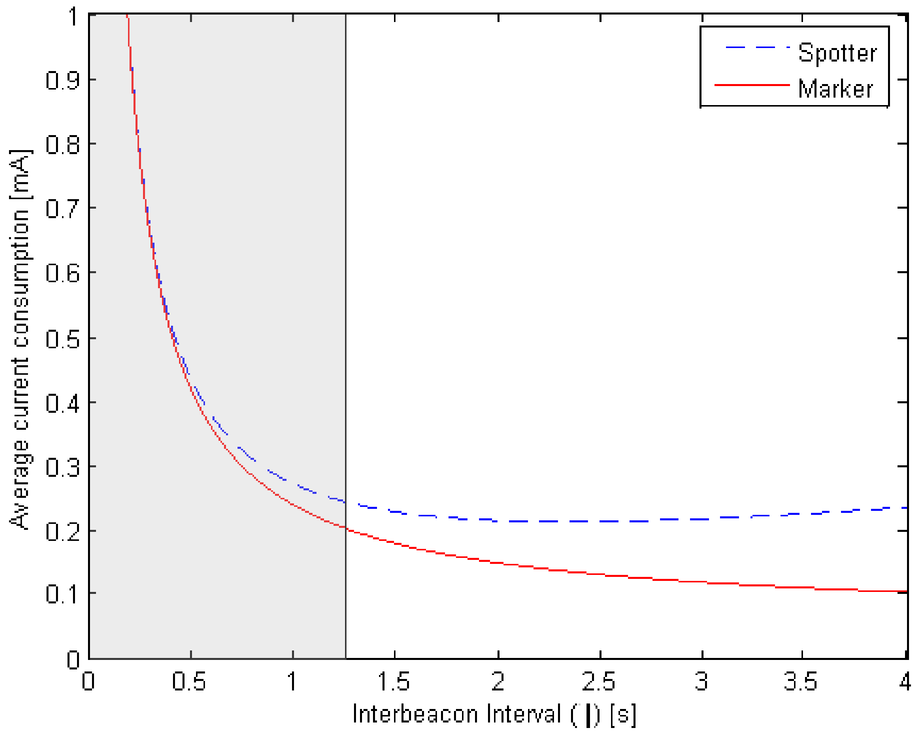


Figure 3.16: Graph showing variation in current consumption for Spotter and Marker nodes with interbeacon interval. The shaded region represents the range of I values that result in less than 75% of nodes being detected when the neighbourhood contains 50 nodes. The simulation parameters are specified in Table 3.3.

system is used merely to observe contacts between individuals, an accuracy of minutes may be acceptable.

If the EcoLocate system contains GPS enabled nodes, these can be used for precise (within 100 ns) time synchronization. All GPS nodes can thus be regarded as being perfectly synchronized and can be used to time-stamp packets from other nodes in the network. If however there are no GPS nodes in the network, the base-station can be used as a timing authority. Alternatively, nodes in the network can be equipped with a Real Time Clock chip to provide an absolute timebase.

Nodes have their own free running clocks and time synchronization is done post-hoc once data is received at a base-station. When a node enters spotting mode, it records its timestamp, as well as the timestamps of any overheard beacons. If the spotting node happens to be time-synchronized, it will also record the real time. Thus, when a base-station or GPS enabled node detects beacons within range, their free-running timestamps are locked to a global timebase. Nodes that never are within range of a synchronized node are synchronized post-hoc through multi-hop time synchronization.

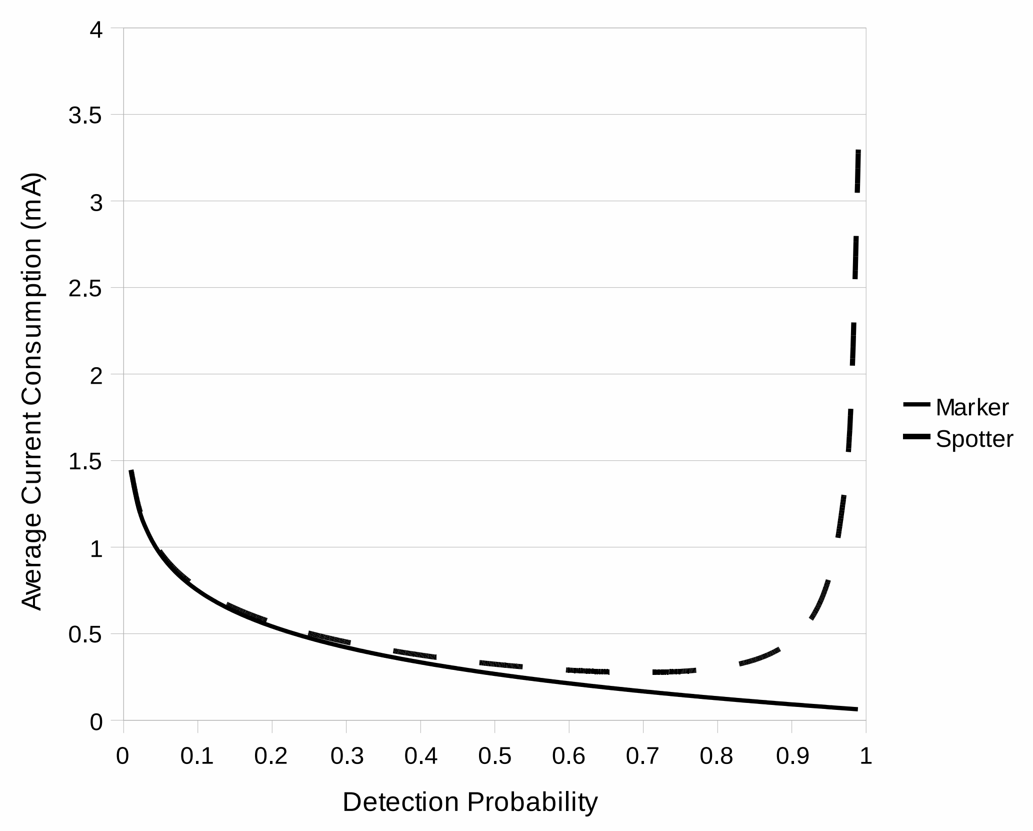


Figure 3.17: Current consumption for Marker and Spotter nodes in relation to required probability of detection.

As nodes are constantly emitting beacons and spotting beacons from other nodes, this means that the beacon protocol serves an additional function of providing time synchronization amongst the various free running clocks.

3.8 Modular System Design

Now that the different classes of tags have been presented, the approach to the system design in a modular fashion is discussed. This is especially useful to the users of the system who can express their requirements for the tracking system in a simple, graphical manner (which is presented in Section 3.10). There are six main groups of add-on blocks, namely Sensors; Memory; Firmware capabilities; Power sources; External Interfaces and Direction Finding, in addition to the basic substrate module. The benefit of a modular approach is that end-users do not have to be technologically conversant with the underlying technology in order to design their systems.

3.8.1 Basic Modules

These are the substrate modules which the other modules attach to[19]. The modules provide the functionality required for each class of tag, from Marker up to Base tags. The main substrate functions that can be programmed into a tag are:

- Marking
- Spotting
- Data injection (Transfer of stored data into network)
- Data forwarding and routing (Network ability)

The different functions implemented dictate the class of the tag – refer to Section 3.5.8 for an overview of class assignment.

3.8.2 Sensor Blocks

Sensor blocks provide an interface between real world variables (such as temperature or humidity) and the network. Any variable which can be digitised can in theory be input into the information delivery system, subject to bandwidth and power constraints. Some typical sensor blocks are shown in Fig. 3.18. For the wildlife tracking application, the variables that are of most interest are position (generally acquired from a GPS receiver) and energetics/activity (acceleration and temperature of host). Other variables of interest are related to microclimatic indicators such as insolation, temperature, humidity and pressure. If animals are equipped with passive RFID tags, then the readers can be interfaced to the network, delivering the time and ID of the detected tag. Wildlife behaviour is greatly influenced by the availability of natural resources, in particular water [228]. Water holes can be equipped with water level sensors (for example, an ultrasonic range finder) which provide an indication of the amount of water available and its rate of change (due to environmental effects such as evaporation and also consumption). Ambient sound is another variable which can be recorded - collars can store vocalizations of animals and noises from their surroundings. This can be used to correlate specific behaviours with certain sounds, helping researchers to understand why and when animals make noises. Still or video footage can also be taken. Due to the high-bandwidth requirements of the video feed, video would be best stored on an archival type data logger so as to not load the rest of the wireless network. Nodes can also sense information

[19]Nodes are given a unique ID number which is used to identify them. In the current prototype version (see Chapter 6), a 16 bit number is used as the ID code giving 65536 unique numbers in the network. This can easily be extended to 24 or 32 bits.

about their current status or health (such as battery voltage, memory usage and network load), which can be used to decide when to replace nodes in the network.

Temperature	Temperature measurement device. Range and accuracy depend on application
Acceleration (x)	Acceleration sensor. Number of sensing axes is denoted by x.
Heading	Heading sensor. Digital compass that indicates direction of travel relative to magnetic north.
GPS	GPS receiver. Determines location in three dimensions anywhere in the world. Also provides precise time synchronization.
Sound	Audio recorder. Records animal vocalization and ambient noise.
RFID Reader	Passive RFID reader. Detects the ID codes of PITs within range.

Figure 3.18: Different sensor options for incorporation into the basic module.

3.8.3 Memory

Nodes can be equipped with different types and sizes of memory, depending on the requirements of the application. Small on-board memories are used in typical Spotter class nodes - they provide memory sizes between 10 kbytes and 100 kbytes, depending on the type of processor used. Other nodes can generate and route vast amounts of information. From the point of view of resilience and error recovery, it makes sense to store all the information acquired permanently in a large memory. SD memory cards provide a simple and inexpensive way of providing non-volatile storage of vast amounts of data (the largest size currently available is 16 Gb). SD memory cards are widely available and can be controlled via an SPI interface, which makes interconnection to standard microcontrollers trivial. Nodes can store all the information they generate, but only send to the end-user a small subset of summary data. If the node is recovered at the end of the study, all of the buffered data can be retrieved, resulting in a richer dataset.

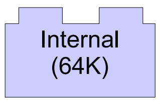
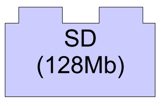
	Internal memory. Capacity is shown in brackets.
	SD Memory card. Capacity is shown in brackets.

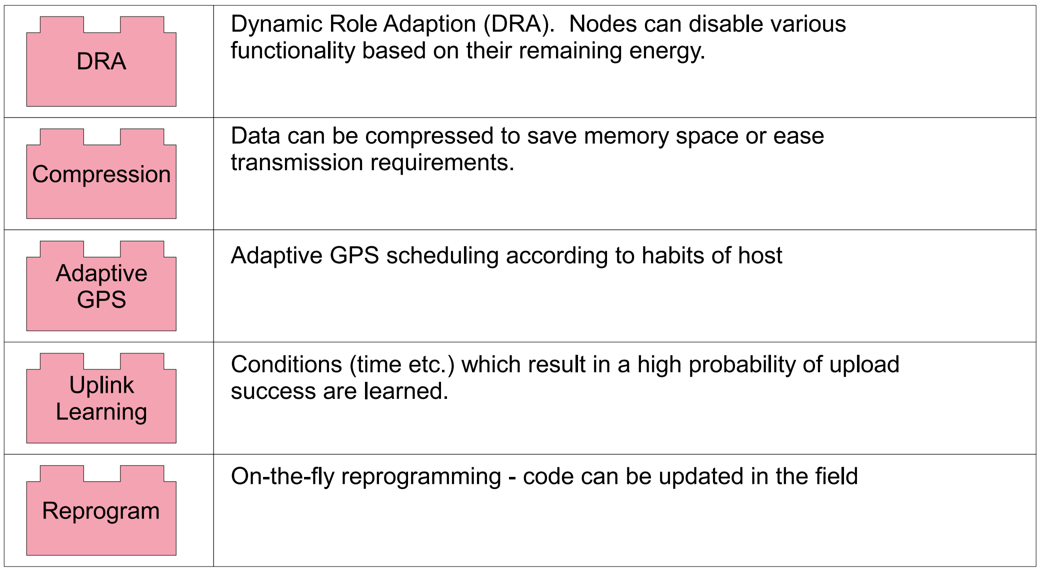
Figure 3.19: Different memory blocks for incorporation into the basic module.

3.8.4 Firmware Capabilities

Blocks do not have to be physical, and can actually be features of the firmware, such as signal processing functions to handle the acquired data. Some possible firmware blocks are shown in Fig. 3.20. Dynamic Role Altering is the functionality that allows a node to vary its responsibilities in the network in response to changes in energy, as discussed in Section 3.6. Wildlife tracking data typically has a great deal of structure and repetition (consider for example that successive location fixes will differ from one another very slightly) and thus can be compressed to reduce the energy required to send the information [16]. Another possible block that can be incorporated is Uniform GPS Sampling, a technique that adaptively schedules GPS fixes based on host movement. This is presented in Chapter 5. Specifically for Active Loggers is the ability to learn when to schedule uploads of data based on the correlation between sensor data and the presence of a host. This is discussed in Section 8.3.1. Lastly, it is possible to issue code updates in the field to devices, enabling further functionality as they can be reprogrammed. The blocks presented here are not an exhaustive list of functionality, but merely an overview of some possible ones.

3.8.5 Power sources

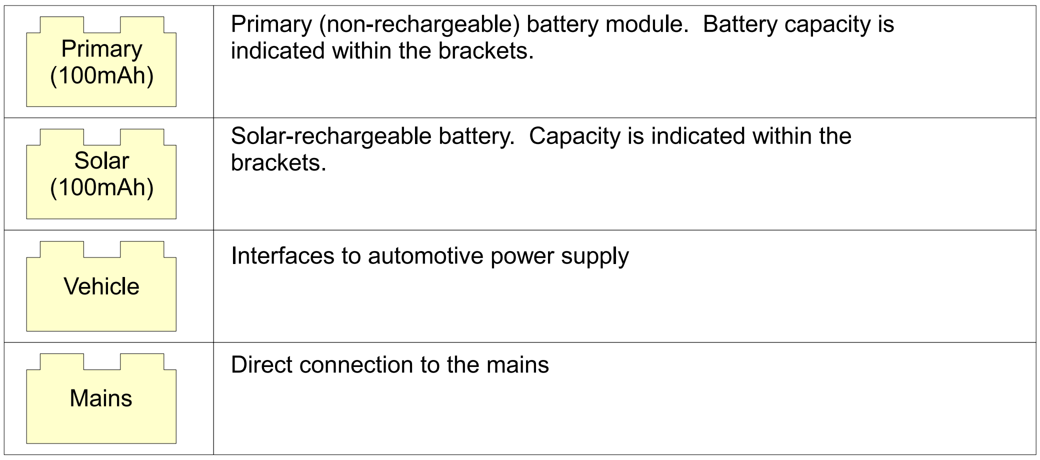
Due to the diversity inherent in the animal tracking application, nodes are likely to be able to be powered from many different sources of energy, ranging from primary batteries to renewable sources of energy. The allowable size of these power sources will also vary depending on the bodyweight of the host animal. Different supply options are shown in Fig. 3.21. Primary type batteries of various sizes can be used to power tags – these provide good energy density but cannot be recharged. Secondary batteries can be recharged using a solar panel to provide operation over extended periods of time. The size of the rechargeable battery must be chosen so as to provide continuity of operation



DRA	Dynamic Role Adaption (DRA). Nodes can disable various functionality based on their remaining energy.
Compression	Data can be compressed to save memory space or ease transmission requirements.
Adaptive GPS	Adaptive GPS scheduling according to habits of host
Uplink Learning	Conditions (time etc.) which result in a high probability of upload success are learned.
Reprogram	On-the-fly reprogramming - code can be updated in the field

Figure 3.20: Different firmware options for incorporation into the basic module.

with worst case solar radiation[20]. Nodes can also be powered from a vehicle or connected to a reliable source of power such as the mains, where possible.



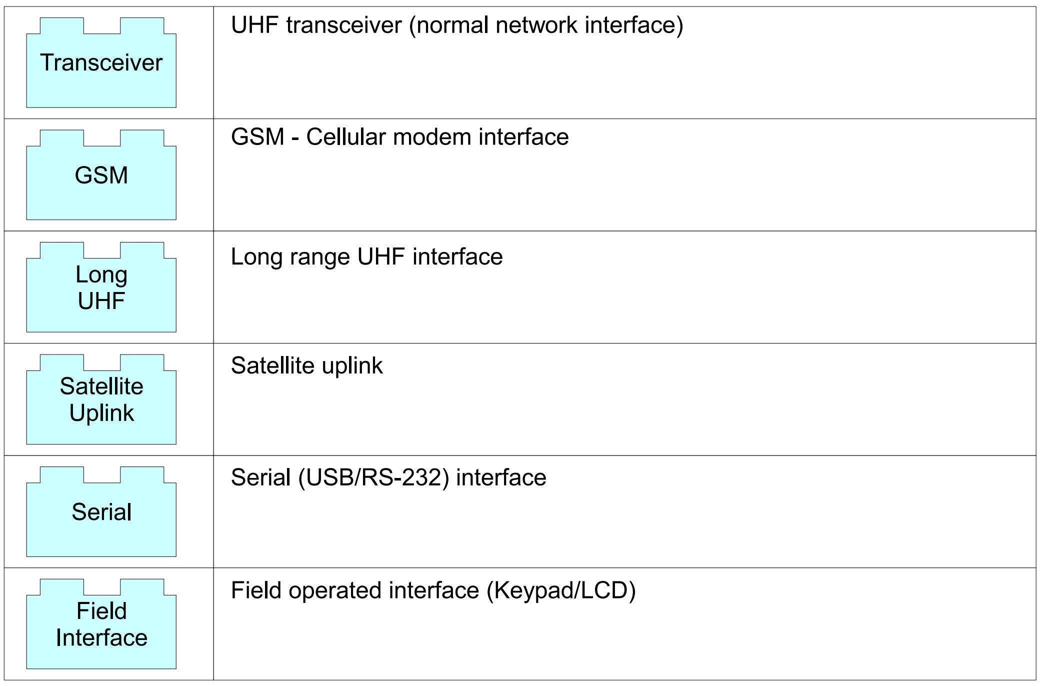
Primary (100mAh)	Primary (non-rechargeable) battery module. Battery capacity is indicated within the brackets.
Solar (100mAh)	Solar-rechargeable battery. Capacity is indicated within the brackets.
Vehicle	Interfaces to automotive power supply
Mains	Direct connection to the mains

Figure 3.21: Different supply blocks for incorporation into the basic module.

[20]The size of the solar panel used depends on the battery technology, the charging control circuit (whether or not maximum power point tracking is used) and other factors such as typical hours of daylight and panel orientation.

3.8.6 External Interfaces

External interfaces provide a link between the network and the end-user. These interfaces can be wired, wireless or even incorporated into the network device to be field operable by a human user. All tracking devices (with the possible exception of Archival Logger nodes) are equipped with a network transceiver[21] that is used to beacon to other nodes and to transfer information through the network. Base class tags can also have interfaces that link directly to the end-user. These can include GSM modems, UHF modems and satellite uplink. For tags that are recovered, a wired serial interface can be used to download the data off the tracking device. Base type tags can also be equipped with a field type interface that allows researchers to monitor and interact with the network in the field. This could consist of an LCD and a keypad to select what operations to undertake.



Transceiver	UHF transceiver (normal network interface)
GSM	GSM - Cellular modem interface
Long UHF	Long range UHF interface
Satellite Uplink	Satellite uplink
Serial	Serial (USB/RS-232) interface
Field Interface	Field operated interface (Keypad/LCD)

Figure 3.22: Different interfaces for incorporation into the basic module.

[21]The particular type of network transceiver chosen depends on the requirements of the application in terms of bandwidth, communication range, power consumption and physical size. The technology used can range from long range VHF transceivers to 2.4 GHz short range devices, without affecting the EcoLocate system design.

3.8.7 Direction Finding

As most nodes in the network periodically emit Marker beacons, techniques that have been used for many years in VHF tracking can be applied to the EcoLocate system. Some direction finding blocks are shown in Fig. 3.23. For field based-receivers or stationary systems, directional antennas can be used to restrict the angle of arrival of Marker beacons, allowing the bearing to the transmitter to be determined. Directional antennas typically also have a higher gain than an omni-directional antenna, resulting in an increase in the detection distance. Rotating directional antenna can be installed at fixed sites, enabling bearings to transmitter tags to be detected for location estimation through triangulation. For improved range, high gain antenna (which can be active, that is, containing an internal amplifier to boost signal strength or passive, that is, a standard antenna with no active filtering or gain) can be used to obtain data from tags which are out of range of standard antenna.

Directional Antenna	Directional Antenna (Fixed)
Rotating Antenna	Rotating Directional Antenna
High Gain	High gain antenna

Figure 3.23: Direction Finding blocks

3.9 Scalable Wildlife Tracking

In this section, example deployments and configurations are shown, demonstrating how the Ecolocate system can behave like existing tracking technology and also how the network functionality can greatly improve the scope and possibilities for wildlife tracking.

3.9.1 VHF equivalent tracking

VHF-like studies can be performed using the basic configuration shown in Fig. 3.24. Marker nodes act like VHF beacons, periodically ‘Marking’ or transmitting their unique ID codes and any other sensor data. Two marker nodes are shown in the graphic, and it can be seen that they basically consist of a controller module with Marker functionality, a UHF transceiver and a source of energy. These are simple, low power tracking tags. At the receiving end, a module has been configured to act as a Spotter class node. It is equipped with a Directional Antenna, which can be used to determine the bearing to a tag for triangulation or homing studies. In this system, the tracking device is supplied from a vehicle power source. To provide an interface to the user, such that the ID code of the beacon can be determined, a field interface consisting of an LCD and keypad is provided.

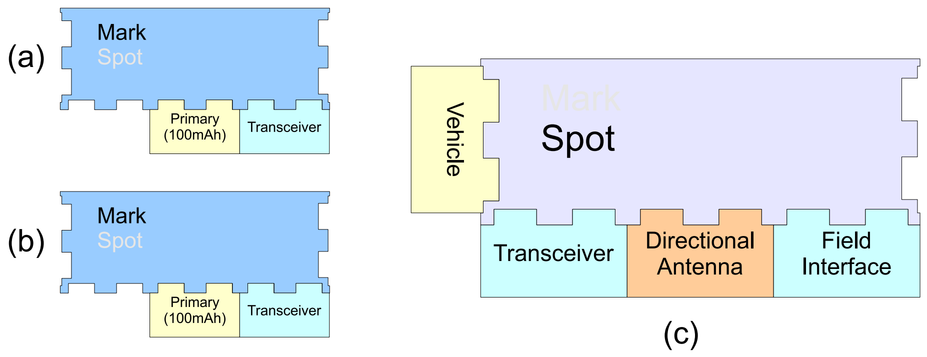


Figure 3.24: VHF equivalent tracking. Animal mounted tracking devices (a) and (b) periodically mark their unique ID's. A field receiver (c) with a directional antenna is used to determine the bearing to the tags. The receiver is powered from a vehicle.

3.9.2 Visit frequency to focal points

A similar configuration can be used to log the times and durations of visits to focal points such as water-holes. Marker type tags are used as in the prior example, but the detector is equipped with a large memory and a solar panel so as to give long operation whilst remotely deployed. Note that the receiver unit has no provision for wireless upload and thus will need to be recovered so that data can be uploaded. For this reason, a serial interface is provided. The system diagram of this configuration is shown in Fig. 3.25.

3.9.3 Active Logger with position and acceleration sensors

In order to monitor and log the behaviour of individuals, they can be equipped with a GPS receiver and acceleration sensors. The obtained data is logged to internal memory. However, to prevent complete data loss in the event of tag loss or destruction, a subset of the data can be uploaded when possible to a receiving station. This can include the GPS

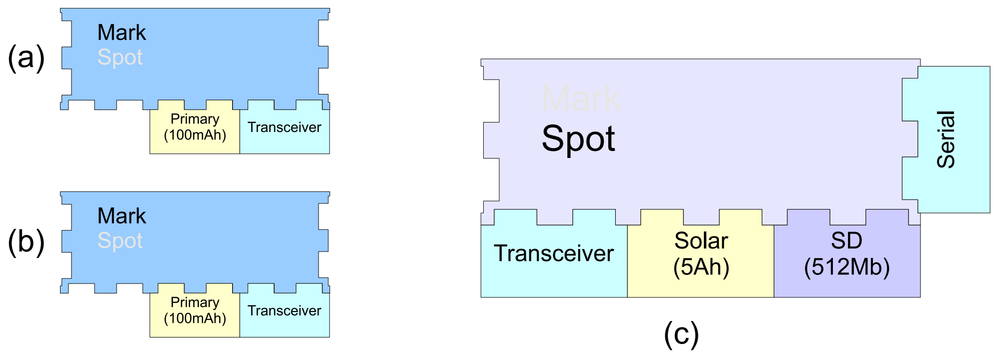


Figure 3.25: Detection of visit frequency to fixed locations. Animal mounted tracking devices (a) and (b) periodically mark their unique ID's. A receiver (c) is placed at a fixed location and is kept powered from a solar module with battery backup. Spotted Marker ID's are recorded in the large (512Mb) memory for later download via the serial interface.

data and summary statistics of acceleration profiles. This is shown in Fig. 3.26. The Active Logger also incorporates the 'Uplink Learning' module, so uploads are scheduled based on possible correlations between the sensor variables and being within range of a receiving station.

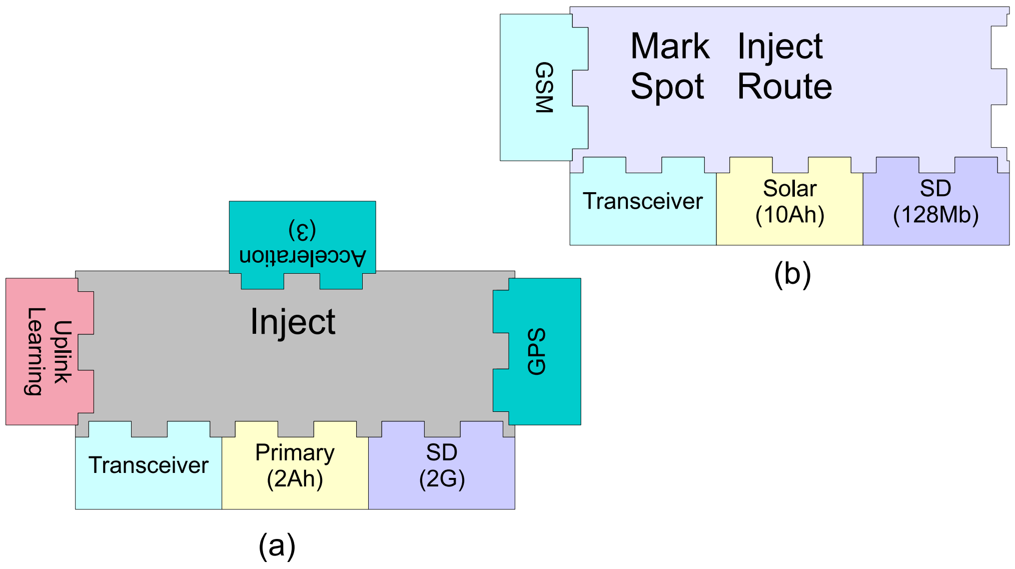


Figure 3.26: Active Logger. Tag (a) is the Active Logger tag and it logs GPS and acceleration data into its large memory. When possible, it uploads data to the network logger (b) which then sends the received data to the end-user via a GSM modem.

3.9.4 Social proximity studies

In order to determine social behaviour, individuals can be equipped with Spotter class tags. These tags detect when they are within range of another device, and record its ID and time-stamp the data. From this a contact log can be constructed. A sample configuration is shown in Fig. 3.27. All tags have Spotter and Marker functionality, and thus they can detect the presence of each other. Tags (a) and (b) record the detected ID's in small on-board memories which they upload (inject) opportunistically to the Base station tag (c) which sends the data via a long-range UHF link to the end-user. Tag (a) is powered from a primary supply, whereas tag (b) has the ability to recharge from a solar panel. Tag (c) is installed on a vehicle and obtains data from the Spotter class tags.

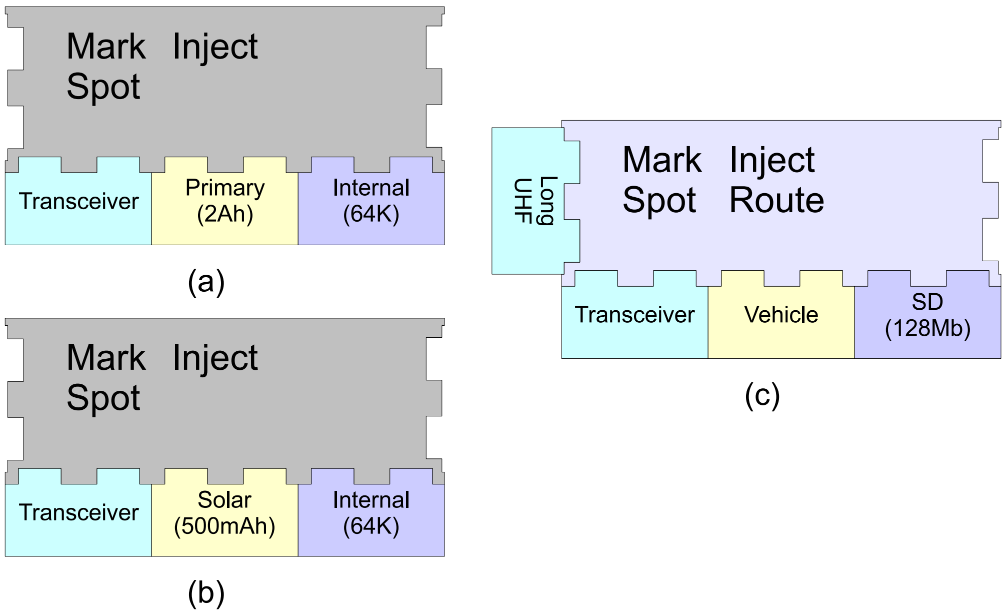


Figure 3.27: Social contact and proximity studies.

3.9.5 Tracking the location and behaviour of a group of animals

The social behaviour and approximate position of groups of animals can be determined by fitting a few individuals with full-function GPS equipped Pack tags and the rest of the members with low cost Spotter class tags. The tags in this system are shown in Fig. 3.28. All tracking devices in this system have both marking and spotting capability. Tags in (a) are purely Spotter nodes, thus they detect the presence of other tags and store the contact logs for future upload to network capable nodes, like (b) and (c). The tags in (b) are Pack class tags that are GPS capable. Thus, the approximate position of the Spotter class tags can be determined if they are within range of a GPS enabled device. One or two animals in the group can also be fitted with Base class tags, (c), which have an external link to the user in the form of a GSM modem. Due to the higher power consumption of these devices, they are equipped with a solar panel and rechargeable battery rather than a primary type cell.

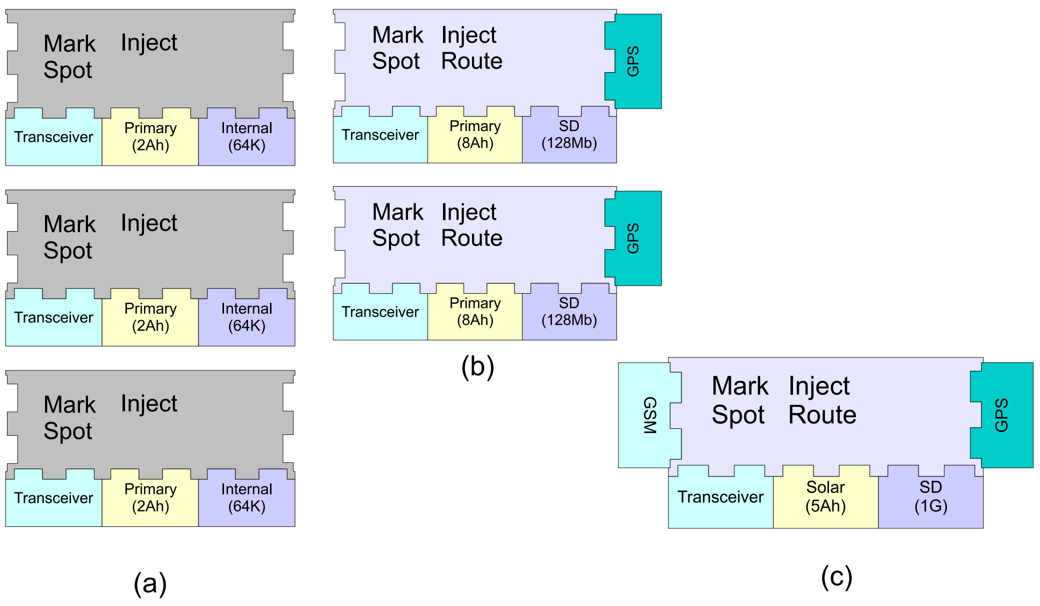


Figure 3.28: Group behaviour studies. (a) Spotter class tags. (b) Pack class tags. (c) Base station tag with GSM upload.

3.9.6 Full Network Studies

The prior functions discussed can be incorporated into a larger network with more entities and components. A hypothetical example is provided, demonstrating how the components can be pieced together, to result in a powerful and useful network.

As an example scenario, the user wishes to simultaneously monitor the behaviour and possible interactions of African wildcats, lions and elephants. The activity patterns (specifically acceleration) of vervet monkeys are also to be measured. In addition, waterholes should be equipped with water level sensors and microclimatic information (temperature and humidity) around a baobab tree needs to be measured. Ground hornbills are nesting close to the baobab tree, and their brooding techniques need to be studied. Also of interest, is the impact of the game rangers (on foot) and the tourist trips in vehicles on the behaviour of the animals. Some of the elephants can be equipped with GSM modems and one of the waterholes can be fitted with a UHF link to the central server. Animal carried tags should not exceed 3% of the host's bodyweight.

Based on these requirements, tracking devices can be designed for each of the targets in turn. The system design is shown in Fig. 3.29

Wildcats Wildcats are small felids (3-6 kg). They cannot carry a GPS enabled tracking device (due to weight restrictions), but will be able to carry a Spotter class tag. These tracking devices will enable the detection of other animals (not necessarily only other wildcats that are within range, but also the elephants and lions). In addition, when observed by a GPS capable device, the approximate position of the wildcat can be determined.

Lions Lions are large carnivores which are able to carry GPS enabled devices. Thus, lions can be fitted with Pack class tracking devices. Lions are typically crepuscular, being inactive during the day. Thus, the uniform distance sampling scheme presented in Chapter 5 is used to schedule when to take GPS fixes. To perform this, a low power accelerometer is used to monitor the gait patterns of the host.

Elephants Elephants are large herbivores and they are equipped with Pack tags, but some of them are also fitted with GSM enabled Base class tags. In terms of sensing, their location is monitored using GPS receivers and their vocalization measured using a microphone. Studies can thus be undertaken on the way and range over which elephants can communicate.

Vervet Monkey Detailed accelerometer patterns of Vervets are required to be obtained on three orthogonal axes, so fine grained locomotion can be determined. Due to the volume of the data that will be generated, uploading the entire dataset into the wireless network will not be feasible. Thus, these creatures are fitted with Active Loggers. Summary statistics of the general locomotion of the animal is relayed throughout the network, with the bulk of the data being stored on-board for later retrieval. The devices are equipped with large memories that allow the high resolution data to be stored.

Waterhole sensor Waterholes are equipped with water level sensors (ultrasonic level meters). Due to their siting, they can be solar powered. As these devices have an 'unlimited' amount of energy, they can scan the beacon channel almost constantly, obtaining high resolution data on visit frequency of animals to the waterhole. These devices are configured as Pack class tags, except for one which is a Base tag with long range UHF link.

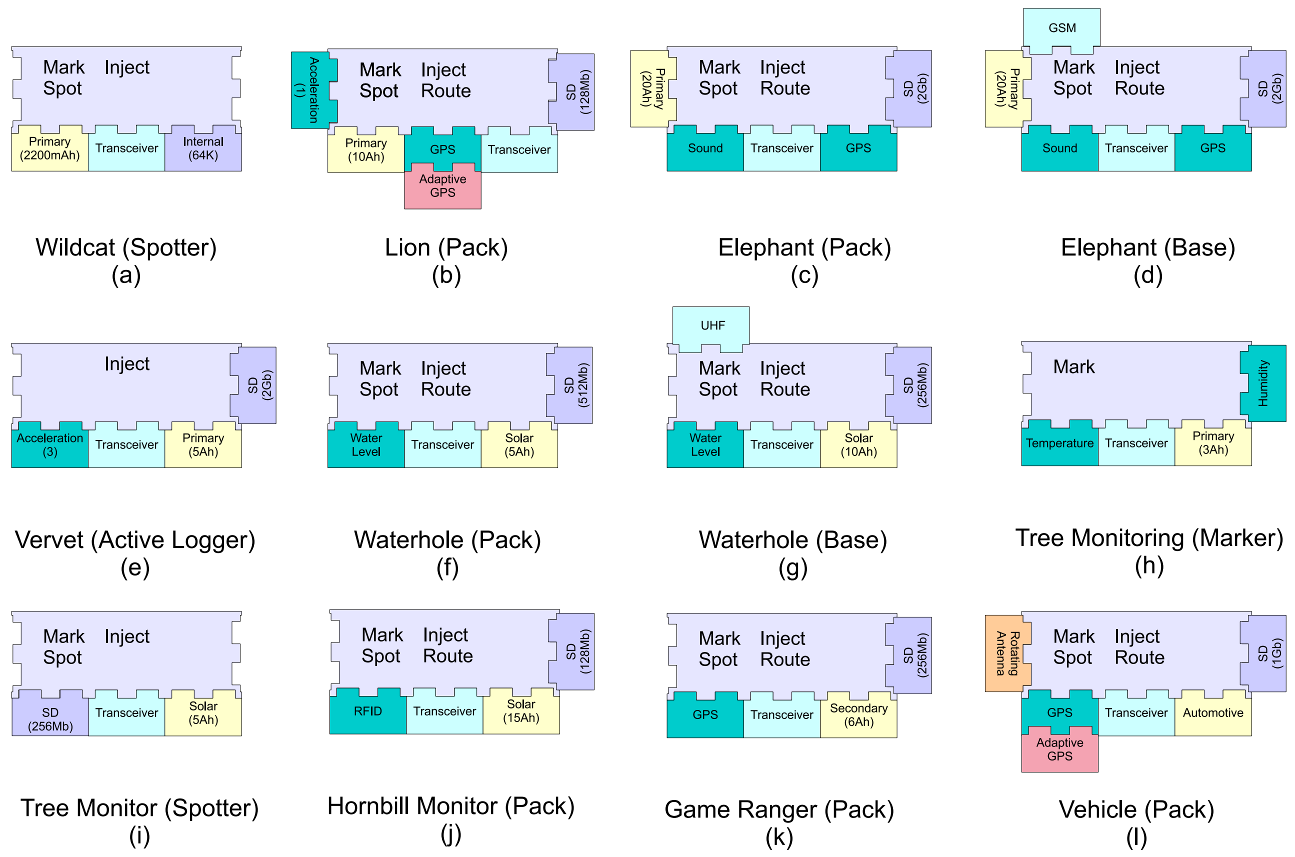


Figure 3.29: Full network study. Refer to the text for an explanation of each block.

Measuring microclimatic information As a multihop network is not required to cover the extent of the tree, a simple solution is to use a number of Marker class tags which transmit their temperature and humidity to one or more Spotter class nodes. Collisions will occur, as the Markers access the medium randomly, with no knowledge of existing traffic. However, if the length of the beacon window is made large, collisions will be less frequent. Solar powered Spotter class tags store the received temperature and humidity information for upload to a Pack tag when within range.

Brooding behaviour of hornbills To determine how parenting is shared (if at all) amongst ground hornbills, the antenna for an RFID reader is placed inside the burrow. The parents are trapped and fitted with leg mounted RFID transponders. Due to the power requirements of the RFID reader, the tag is solar powered. This device is chosen to be a Pack class tag, and thus, the Spotter tag from the Baobab tree would periodically send the acquired microclimate data to the Pack tag for further relaying through the network.

Game rangers The game rangers spend a week at a time in the reserve checking fences and preventing poaching, on foot. As they return to the base camp regularly, the tracking devices they carry can be equipped with rechargeable batteries. These devices are equipped with GPS receivers and frequently spot the presence of animal tracking tags within range. As their power requirements are relaxed, they are made to be Pack class tags.

Tourist Vehicles The position of the vehicles within the reserve can be monitored at a high resolution with GPS enabled devices, parasitically powered from the automotive supply. The uniform distance sampling approach is used, but with different parameters to the lion tracking devices. The device is configured to take location fixes once a second when the vehicle is moving (as measured by its acceleration) and fixes once every hour when it is stationary. A rotating directional antenna is used, both to increase the range of the communications link and also to determine the bearing angle to any tags that Mark. As the game drive vehicles regularly return to the base camp, they are configured as Pack tags, with large memories as they are likely to interact with many tags in a journey. Thus, the vehicles act as high-end data gathering devices from the network.

This example demonstrated how a single system can be designed to obtain data from animals, humans, vehicles and the environment. The heterogeneity of the various carriers is exploited to the benefit of the network as a whole.

3.10 Graphical System Design

The example system designs presented in Section 3.8 show that the modular system design lends itself well to a user-friendly graphical specification of tags and their functions. Whilst the diagrams presented clearly show the function of each tag, it is also possible for the process to be made interactive on a computer. In this way, wildlife researchers can design their own tags for a particular study, leading to application specific tracking devices. Such an interface could be either web-based or run as a free-standing program. The user would create their own tags and then send the designs to the tag manufacturers who would then assemble them according to the user's specifications.

The power consumption and size of tag can be calculated as the user assembles the tracking device, leading to a better understanding from the user of how the various sensing options impact on the power consumption and the subsequent effect on tag lifetime for a given battery size. Each block could itself be configured. For example, a user could click on a GPS modular block and change the desired fix rate. The expected power consumption would be updated and thus the user would be able to observe the link between certain modular options and the energy drain. The form factor of the tracking device can also be specified, and constraints imposed on the system design such that it does not exceed a certain weight or size. Based on the modules chosen, it would also be possible to calculate the cost of the tracking device.

The use of the graphical interface is not limited solely to the design process, as the same interface can be used to reconfigure tracking devices. The parameters of the tracking device can be downloaded from the field, and the expected power consumption compared to the actual power consumption. Based on this and other factors, the user can alter the behaviour of certain blocks so as to either prolong the tag lifetime or increase the volume of data generated. In this way, tracking devices can be dynamically reconfigured in the field to perform a substantially different function to what they were initially designed for (subject to hardware restrictions).

3.11 Discussion and Contributions

This chapter has presented a modular approach towards an automatic data gathering system for wildlife tracking. EcoLocate differs from prior wireless network systems for wildlife tracking in that the heterogeneity of the Animal Kingdom is exploited, so that small animals can still be part of the network, albeit with reduced function devices. Both ZebraNet and TurtleNet are homogeneous in their design and deployment scope, whereas EcoLocate has a number of classes of device. In addition, this system has not restricted device deployment to be solely animal based, and has considered including environmental sensors and human and vehicle carried devices in order to be able to monitor all facets of a typical game reserve. The example scenarios presented here show how the same system can be used for very basic studies, right up to entire network deployments. As the network is self configuring, new tracking devices can be added at any time, allowing researchers to expand the scope of their study. Thus, the extent of the network can be incrementally increased as and when required.

With the reality of data-gathering comes the possibility of acting on the acquired data, either manually or automatically. This has not been elaborated on in this chapter, as the focus has been on the design of a system for data gathering, but if data can be transferred out of the network, it is also possible to issue commands into the network. This can be done using the same network protocol or it could use another layer of networking, such as connecting devices to be controlled to the GSM network so as to not load the scarce energy reserves of the animal tracking collars. Such functionality would enable the automatic and remote control of devices in the field, which could range from water-pumps to pan-tilt-zoom video-cameras. In addition, information about the positions and habitats of various individual animals could be relayed on demand to users in the field. Thus, a zoologist could run a query to find the last known location of an animal in a study. Alternatively, the system itself could trigger alerts if it detects anamolous data, such as a sick animal. For example, if an animal altered its behaviour significantly (such as long periods of rest or a dramatic variation in gait pattern), an alert could be generated and sent to a vet who could observe the animal and decide what (if any) action to take.

Briefly, the contributions of this chapter are as follows:

- The design of a highly modular system for wildlife tracking that is able to monitor a wide diversity of animals, from small to large, using the same basic firmware and hardware.
- The unification of the previously disparate technologies of VHF tracking, proximity detection, data logging and GPS monitoring into an all encompassing wireless network, bound by a common design process.
- Exploiting, rather than being restricted, by the 5% bodyweight rule, such that large animals in the network carry a higher data load than small animals.
- The introduction of a simple beacon protocol, that allows social behaviour and coarse location to be determined, opening up new research areas.
- The ability to dynamically alter the class of the tracking node, so as to prolong the useful device lifetime, at the cost of functionality.

- The proposal of a user-friendly visual approach to structuring tracking studies and designing tracking devices.

Ultimately, the possibilities for an integrated wildlife and eco-system monitoring system are immense. Indeed, according to one paper:

> We foresee an evolutionary process in the design and use of biological sensor networks, wherein the enabling sensor network infrastructure stimulates new ideas and concepts of what is possible and in turn, stimulates new demands for that infrastructure.[188]

The presented EcoLocate system is a step towards unifying tracking technology, such that multiple studies at different resolutions can be undertaken using the same basic devices. A single *system* has been presented here that can be used to track and monitor multiple animals and their environment, in a scalable and modular fashion.

In the land of the blind, the one eyed man is king

Desiderius Erasmus (1466 - 1536)

4 Adaptive Social Hierarchies: A Network Ranking System

4.1 Introduction

A group of network nodes can be regarded as a society which is performing a task (such as sensing)[1]. How well the group performs this task affects information delivery and network longevity. Nodes in a network are differentiated by explicit attributes (such as size of energy reserve, amount of buffer space) and implicit attributes (such as connectivity). The network must efficiently transfer information from source to sink, generally through intermediate nodes in the absence of direct connectivity between source and sink. Choosing which intermediates to transfer through is the subject of routing protocols, of which there exist a plethora (e.g. refer to [235, 236, 237]).

Instead of introducing another routing protocol, this chapter considers how a network could manage itself, given that nodes 'knew' their role within the group. For example, a node with low remaining energy level is likely to be exhausted soon and should thus adopt a less active role in the network than a node with a large amount of battery energy. What is meant by 'a less active role' is application dependent, but could range from avoiding routing other node's packets to disabling energy hungry sensors or altering their duty cycle. In a network where all the nodes are initialized with equally sized energy reserves, determining whether a node should have a low or high role is simply related to its remaining proportion of energy. Metrics other than energy can be ranked, and some suitable attributes to rank (such as connectivity) are discussed further in Section 4.7. However, this simple example considers tag energy.

Consider now the case where nodes are not introduced with the same initial amount of energy. Such would be the case in a widely heterogeneous network. In Chapter 3,

[1]Portions of the work in this chapter have been published in [232, 233] and [234]

the example scenario of an animal tracking network was introduced, where the size of the host animal dictates the maximum weight of the tracking collar [187]. The multiple allowable battery weights results in a large degree of diversity in initial energy reserves. Assessing when a node is a low energy node now becomes dependent on the composition of the group, and no longer related to the absolute amount of energy. For example, in a network with nodes with energy levels of 100J, 200J and 300J, a 100J node has the lowest energy and thus should adopt a low role. Conversely, in a network composed of nodes with energy levels 20J, 50J and 100J, the 100J node is the ‘best’ in terms of energy and thus should be the most active. Although the energy of the node is the same, its behaviour is dictated by the energy reserves of its peers.

This simple example demonstrates that some sort of network wide discovery system is required in order to determine the role of nodes in the network. To undertake this, nodes could send their attributes to a central server which would gather network information and from this instruct each node to adopt a certain behaviour. Whilst simple, it presents a single point of failure and does not scale well to large numbers of nodes in the network. A simple and lightweight localized discovery scheme is thus required so that each node can decide on its own level, based on information acquired from its peers.

Animals are able to determine their role within a society without any centralized control, by forming a hierarchical structure. Animals form collective groups, with an order of precedence, where some individuals are superior to others in their preferential access to resources (such as mates and food). A similar method of social organization is applied to the problem of discovering a node’s rank in a wireless network, so that nodes can determine their role and access scarce resources (such as network bandwidth and energy).

The application and construction of social hierarchies to wireless networks are discussed in Section 4.2. Next, social hierarchies in nature are examined in Section 4.3, to understand how and why they form and how they adapt to changes. A method of creating a stable social hierarchy through pairwise exchange is presented in Section 4.4, and refined through introducing the mechanism of reinforcement. Based on the observation that pairwise exchanges place a high burden on the medium access and control (MAC) layer, a broadcast version of social hierarchy formation is elaborated upon in Section 4.5. To remove the restriction that the network be mobile in order for the hierarchy to correctly form, pseudo-mobility is introduced through the action of random network agents in Section 4.6. This section also examines how mobility models impact on the rate of convergence and adaption. The focus of the chapter then turns to suitable network attributes to rank in Section 4.7, followed by a presentation of some example scenarios showing the use of the social hierarchy approach in Section 4.9. Lastly, the work is contrasted with related work in Section 4.10, before conclusions and future direction are posed in Section 4.11. The application of hierarchies in wireless networks is now discussed.

4.2 Hierarchies in wireless networks

Wireless networks are by their very nature amenable to the imposition of a social hierarchy. A hierarchy is a relative ranking system, based on measurable differences between individual nodes. Essentially, a hierarchy provides an abstraction of the real world resources onto a framework which indicates relative performance (such as poor, good, best). The purpose of a network is to transfer information from a source to a sink, through multiple nodes. To conserve resources (such as bandwidth and energy), the information is sent along the shortest path (as measured by some metric such as hop-count or latency) between the sink and the source, through intermediate nodes. The aim of a routing protocol is in essence to determine the 'best' path for the information to take. This means that the set of all possible paths can be cast into a hierarchy, ordered by the desired metric, and the routing algorithm decides which is the best path. Routing protocols which are based on minimizing some metric are implicitly forming a hierarchy amongst the nodes in the network and basing routing decisions upon differences between individuals. However, explicitly constructing and adapting a hierarchy is an area which yet to be investigated fully. The purpose of this work is to allow nodes to determine their relative importance or performance, so they can decide their usefulness to other nodes in the network.

4.2.1 Existing use of hierarchies in wireless networks

Hierarchies are a common feature of network protocols, but many are imposed at design time, based on physical or functional differences between nodes. For example, in the Data Mule system, there are three classes of nodes – low energy stationary nodes, high capability mobile nodes and base-stations [238]. Another example of a static hierarchy is the two-tier network used by Wang et al. to process and locate bird calls [239]. In this network, simple nodes relay acoustic data to high-end signal processing nodes [239]. These nodes have statically assigned roles and are unable to alter their roles in response to changes in network conditions.

Some network hierarchies are able to react to dynamic changes in network structure and composition. For example in the Low Energy Adaptive Clustering Hierarchy (LEACH), a node is selected from a small group to act as a 'cluster-head' which is responsible for relaying the combined information of the local group [240]. As nodes age, the role of cluster-head is rotated amongst the group. This two-level hierarchy is an example of a despotic or totalitarian structure, in that a single node is chosen to 'lead' the cluster and the remaining nodes have equivalent, subservient roles. Other cluster based schemes have improved on the performance of LEACH, yet still retain the two-tier structure [241, 242].

4.2.2 The need for a dynamic ranking system

A network should be able to learn the differences between nodes, rather than have the differences specified prior to deployment. In addition, nodes should be able to determine their usefulness to the network, in a richer fashion than the despotic hierarchies as used in LEACH and similar protocols. Nodes should also be able to discover their usefulness to the entire network, not merely their one-hop local neighbourhood.

In this way, the network will be able to dynamically adapt to insertions and removals and provide resource scalability. The term *adaptive social hierarchy* (ASH) is used in this work to reflect the fact that the hierarchy adapts to changes in attributes and is not static.

A hierarchy can be thought of as a mapping from an absolute metric (such as 1 joule of energy, 3 hops from sink) to a relative metric (e.g. best node in network). An example of this mapping is shown in Fig. 4.1. Although the real world values of the three nodes in network B are ten times as large as the metrics in network A, they both map to the same system when viewed from a relative ranking perspective. In the rest of the chapter, the rank is scaled to lie within [0;1], where 0 corresponds to the lowest ranked node and 1 corresponds to the highest ranked node[2].

Node management and routing is based on the relative parameters, rather than the absolute. This means that network decisions are based on factors which are independent of the real values, leading to scalability in the resource domain. In a relative domain, network decisions do not need to take into account real values, with the implication that a network protocol can be used in vastly different application scenarios, without having to alter network tuning parameters. In addition, any parameter (or composite function of parameters) that can be measured can be placed into a social hierarchy. Thus, the design of network protocols can be separated from the real world scenario. For example, in a static network, hop count is an important metric, whereas in a delay tolerant mobile network other metrics such as utility (transitive connectivity or degree of connectivity) are used. Traditionally, each of these would require a different routing protocol, whereas if a social hierarchy is used, a single network protocol can be used for both problems.

[2]Any sensible scaling can be used – another method would be to scale nodes to reflect their percentage usefulness. In terms of implementation, it might be simplest to scale ranks to lie on [0;255] for a single byte representation of rank.

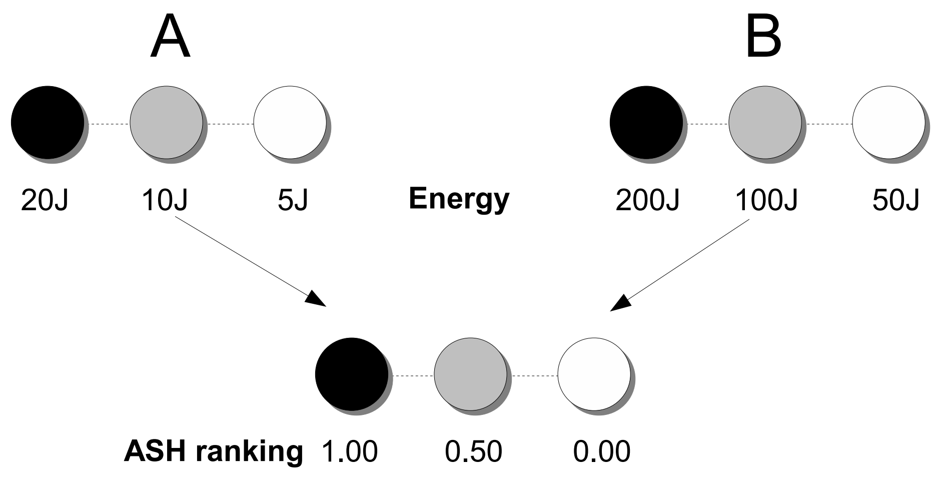


Figure 4.1: Illustration of ASH ranking procedure. Although the networks in A and B have different energy levels, they rank to the same network when viewed in the ASH sense. This resource abstraction makes network design independent of real world values.

This is illustrated in Fig. 4.2 for a stationary and a mobile system, showing how they both map into the same relative network, simplifying network design. In this way, the same routing rules can be used if a social hierarchy is constructed. This is a powerful new concept, as it allows the specification of routing protocols to be separated from real world values and instead be based on a fixed [0;1] domain.

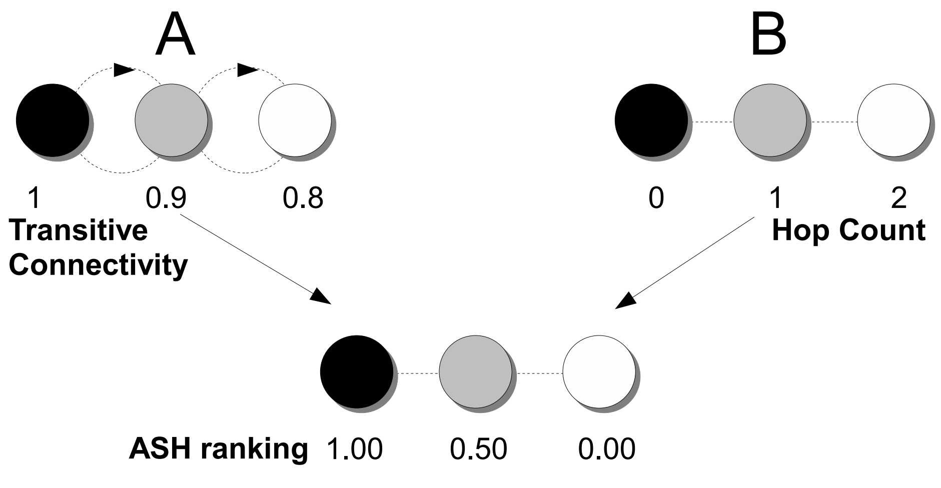


Figure 4.2: ASH working on two different connectivity metrics. In mobile network A, connectivity is measured by the transitive connectivity, whereas in the stationary network B, connectivity is measured using the traditional hop count. Note that both networks map to the same network when ranked. The circles in network A indicate the motion of the nodes.

Thus, there are now two different areas in network protocol design: constructing the social hierarchy and the formulation of protocol rules. This work concentrates on formulating methods that can be used to construct the social hierarchy amongst a group of nodes, but outlines of possible protocol rules are proposed, so as to illustrate the ASH approach. A brief foray into the Animal Kingdom is presented in the next section, examining social hierarchies in nature. The formation and adaption of these social hierarchies is used as an inspiration for the introduced methods, which draw and expand on the way in which animals form hierarchies.

4.3 Social Hierarchies in Nature

Many animals live in societies or groups of individuals. The size and composition of these groups vary, but most have a common feature of the imposition of a social hierarchy. A social hierarchy can be regarded as the organisation of individuals of a collective into roles as either leaders or followers with respect to each other. In biological terms, leading and following are referred to as dominance and submission respectively [25]. A individual dominant over another typically has preferential access to resources such as food, water or mating rights [25]. Note that although one animal may be dominant over another, it itself may be dominated by yet another member of the group. Social hierarchies are such a pervasive natural formation that words describing their behaviour have taken on colloquial usage, such as ‘pecking order’, ‘leader of the pack’ and ‘alpha male’ [25].

4.3.1 Formation and maintenance of hierarchies

Different species form hierarchies in different ways, but there is a common feature to all: *communication* of relative dominance or submission. Based on the received communication, animals subsequently update their perception of position within the hierarchy. Communication can be implicit, such as another animal observing the physical size of another or it can be explicit in the form of an interaction, such as an aggressive fight [25]. Communication can take on many forms, depending on the capabilities of the particular species and can be tactile, olfactory, acoustic/vocal or visual. Whatever its form, the communication can be regarded as a stimulus emitted by the sender which alters the receiver’s behaviour. The communication exchange followed by the hierarchy update is referred to as a dyadic (pairwise) *tournament* [243]. Tournaments generally result in a winner and a loser, with the winner increasing its role in the hierarchy upwards and the loser downwards. An animal which frequently wins encounters with other animals is likely to have a high role in the hierarchy [243]. Some attributes which are used to construct the social hierarchy and how they are communicated are now considered.

In a hive of bees, there is only one queen. This individual is responsible for the laying of all the eggs within the hive. The queen is tended to by her sister bees, and the queen emits a hormone which suppresses the formation of ovaries in her peers [244]. This is an example of a totalitarian or despotic hierarchy [244]. If the queen bee leaves to form a new nest or dies, in the absence of the hormone, the sister bees start to regrow their ovaries. One bee will dominate the rest, becoming the new queen. A similar structure also occurs in other eusocial organisms such as wasps [245], ants [27] and termites [246]. This is similar to the static hierarchies used in existing wireless network systems, such as the Data Mule concept [238].

The formation of dominance structures in chickens is a well studied area [247], [243]. Hens form a linear dominance structure, with a dominant individual meting out pecks to subordinates, hence the term ‘pecking order’ [25]. When a group of hens are assembled, there are frequent fights and changes in rank initially. Over time, as the structure becomes known, there are fewer and fewer fights as the hens are aware of their role or position within the hierarchy [244]. If a new hen is introduced into the flock, the other hens will attack it as a group [24]. In a linear dominance hierarchy, an alpha or super-dominant individual dominates all others. The next individual in the hierarchy, the beta individual dominates all bar the alpha and so on. Linear dominance hierarchies are a common feature in many species, such as crayfish [26], anemones [23], goats [248] and ibex [249].

Hierarchies are also a common system in primates, such as baboons and rhesus monkeys [250]. Baboons have a complex culture of social interaction which cements the troop together. Within human societies, hierarchies are also a pervasive theme. Political and business structures are organized along the lines of dominance hierarchies, with presidents, vice presidents and so forth. Even the academic world is an example of a social hierarchy, with professors, lecturers and students being its constituents.

There are many attributes which are hypothesized to be important in the formation of social hierarchies, and these depend on the individual species. In crayfish, there is a strong correlation between claw length and social status [26]. In anemones the length of the feelers results in a difference in ranking in individuals [23]. In hens, there was a strong correlation between rank and comb size [24].

The exact mechanism which is behind the formation of a social hierarchy in natural systems is sometimes unclear [251]. In repeated experiments with the same group of fish, researchers demonstrated that the rank order formed in successive trials was not identical, as would be the case under the assumption that rank order was solely based on observed differences between individuals [252]. There is a strong correspondence between the rank of the attribute and the resulting social hierarchy ranking, but it does not appear to be the only process at work [22]. It has been hypothesized that there is a form of positive reinforcement, where an individual which has recently won a tournament is more likely to win a subsequent tournament [252]. The reasons for this are unknown, but it is assumed to be as a result of increased hormone production, leading to the individual becoming more aggressive.

It must be made clear that social hierarchies are formed along the lines of comparative or relative differences between individuals, rather than absolute attributes. For example, in one group of animals, an individual may be ranked at the top, but in a fitter group of animals, it may only be ranked in the middle. Thus, an individual's role within the hierarchy is not only dependent on its own attributes, but also the composition of the group itself. Social hierarchies are not static structures and dynamically react to changes as the result of insertion or removal of other creatures. They also alter if the attributes of a creature change, such as a result of injury or age.

4.3.2 Constructing an Adaptive Social Hierarchy

As previously stated, a hierarchy can be viewed as a mapping from an absolute parameter space to a relative parameter space. This mapping can be undertaken in many different ways. Ranking needs to be determined in a distributed fashion amongst all the nodes, as opposed to a centralized system which has a single point of failure. As nodes are typically resource constrained, methods to determine ranking should not result in a high level of overhead in the form of ASH ranking and updating. In addition, the ranking needs to be scalable not only with the number of nodes in the network, but also with the range of parameter values. Before various ranking systems are discussed, some terminology is introduced.

Let S be the set of a particular attribute A of the N nodes[3] in the network

$$S = \{A_1, A_2, \ldots, A_N\}\,. \tag{4.1}$$

When the order of the elements in this set is taken into account, an ordered set is formed. For example, consider the four element set of attributes $S = \{10, 4, 8, 6\}$. The ordered set, $\phi = [4, 6, 8, 10]$ is the original set, sorted into order of size. The rank order of the sorted set is given by $R = [4, 1, 3, 2]$ where the rank refers to the index of the original element. In

[3]The number of nodes in the network, N, is not known by the network. The formulation here is a definition which is used to compare the performance of the protocols to their correct relative value, which could be calculated if global information was known.

our case, however, we are interested in the scaled or normalized rank, which is expressed as

$$\widehat{R} = \frac{R-1}{N-1}. \tag{4.2}$$

Thus, for the given example with rank order $R = [4, 1, 3, 2]$, the normalized rank is found to be $\widehat{R} = [1, 0, \frac{2}{3}, \frac{1}{3}]$. By normalizing the rank, the rank is made independent of the number of nodes in the network, leading to scalability. This is because it is known that the maximum rank is 1 and the minimum rank is 0. Thus a node with an attribute that has a rank of 1 can be regarded as the 'best' node in the network, in terms of that attribute. The rank as estimated by a method of ranking is denoted as E.

The goal of this work is to present various methods of evaluating the ranking in a distributed fashion, much like animals form linear social hierarchies based on measurable or perceived differences. This work concentrates on forming linear dominance hierarchies, which essentially are scaled ordinal mappings[4]. In these mappings, the value of the attribute is not important, only its ordered position. Thus, regardless of the value or spread of the attributes, the linear dominance hierarchy will uniformly space the ranks of nodes. In order to characterize and compare ranking methods, various measures of performance are now discussed.

4.3.3 Measures of performance

There are a number of factors which are relevant to the formation and maintenance of the social hierarchies. Probably the most important factor is the rate of convergence or settling time of the social hierarchy, from initialization to a point when it is deemed to be settled. Another factor is the stability of the hierarchy - whether nodes remain in the correct order, or whether there are time varying errors in the ranking. Lastly, the adaptibility of the social hierarchy – how long it takes the system to react to disturbances, such as node insertion or removal or a change in a node's attribute, is also important. With all of these performance metrics, it is necessary to determine whether they are scalable in terms of increasing N, the number of nodes in the network.

To determine when the hierarchy has converged, a metric is required that reflects the degree of order of the rankings of the nodes in the system, and whether they are in concordance with the order of the absolute parameters. The simplest metric is one which indicates when the estimated ranks are perfectly ordered. In this case, one can say the system is correctly ordered when $E = \widehat{R}$.

[4]Another approach would be to form a scaled representation of the attributes in the network, essentially a linear mapping from the attributes to the unit domain. This is suggested as a possible area for further research.

In a network of this nature, subject to frequent changes and rank reversals, it is not necessary for the hierarchy to be perfectly sorted in order for the nodes to know their approximate role in the network. One possible measure of error is the mean squared error (MSE). This reflects how close the system is to its desired steady state values, but does not weight incorrect orderings highly. Another metric, formulated especially for determining ordinality of the ranking, is the Kendall correlation coefficient, denoted by τ [257]. This indicates the normalized number of pairwise exchanges or switches required for the set of estimated ranks to be perfectly sorted. A Kendall τ of -1 corresponds to perfectly reversed ranking (for example [4;3;2;1]), whereas a τ of +1 reflects that the system is perfectly ordered.

However, the worst case performance of a node is probably of most interest, and this would be expected from the node with the greatest error in ranking. For example if a node has a small amount of energy, and it is erroneously ranked highly relative to its peers, then depending on the application, it could expire rapidly as a result of increased use of its remaining energy. Thus, the convergence time is defined as the expected time taken for the maximum rank error in the network to drop below a certain threshold. The maximum (absolute) rank error is defined as

$$e_{max} = max \left| E - \widehat{R} \right| , \tag{4.3}$$

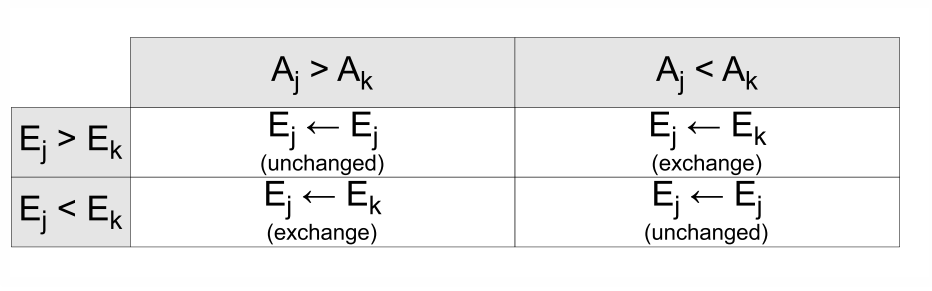
and the system is converged when $e_{max} < \epsilon$, where ϵ is the acceptable percentage error tolerance. The time taken for this to occur is denoted as $T_{\pm\epsilon}$. Note that T_0 corresponds to the expected time to be perfectly settled. $T_{\pm 20\%}$, $T_{\pm 10\%}$, $T_{\pm 5\%}$ and T_0 (corresponding to a maximum error of 20%, 10%, 5% and 0 respectively) are used to measure the rate of convergence (ROC). Different applications will require different error thresholds depending on their requirements.

Closely related to the initial ROC is the performance of the system when nodes are subject to changes in their attributes or nodes are inserted or removed. In control theory terms, the rate of adaption (ROA) is a measure of disturbance rejection. A system which takes a long time to recover from perturbations will not be suitable for use in a wireless network, which is characterised by frequent variations in resources and connectivity. To measure the ROA, 10% of the nodes' attributes are varied to new, random values once the network has converged to the specified error threshold. The ROA is defined as the time taken for the system to re-achieve $T_{\pm\epsilon}$ after the pertubation.

Now that metrics have been formulated for the expression of performance of ranking schemes, the first method is introduced in the next section.

4.4 Pairwise ASH

Much like the way animal dominance hierarchies form through dyadic interactions, the first method presented consists of a simple pairwise switch or exchange method of sorting the ranks of the nodes, in concordance to their attributes. When a pair of nodes meet, they trade their attributes and ranks (i.e. node A will send its ranks and attributes to node B and vice versa). Each node now runs a tournament to assess whether it should alter its ranks. If the order of their ranks are in contradiction to the order of their attributes, their ranks are switched. If there is no contradiction, their ranks are left unchanged. The rules for this method of ranking are shown in 4.3. There are four possible combinations - two correspond to ranks and attributes with the same order and the other two deal with contradictions. For example, assume nodes j and k meet, and the attribute of node j is greater than that of node k (i.e. $A_j > A_k$). This means that the rank of node j should be greater than that of node k, if the ranks are correctly ordered. However, assume that this is not the case, and in fact the rank of node j is less than that of node k (i.e. $E_j < E_k$). Node j will thus adopt the rank of node k, and, at the same time, node k will adopt the rank of node j. Thus, the nodes switch their ranks with one another. Conversely, if there is no rank contradiction, the ranks are left unchanged.



	$A_j > A_k$	$A_j < A_k$
$E_j > E_k$	$E_j \leftarrow E_j$ (unchanged)	$E_j \leftarrow E_k$ (exchange)
$E_j < E_k$	$E_j \leftarrow E_k$ (exchange)	$E_j \leftarrow E_j$ (unchanged)

Figure 4.3: Rank update rules for pairwise ASH. Both nodes perform the same rules at the same time, which will lead to a simultaneous exchange of ranks in the event of a contradiction

A demonstration of the sorting action is shown in Fig. 4.4 for a 10 node network, where in each simulation iteration, two nodes are picked at random and meet. Initially, the nodes were assigned to have ranks in perfect reverse order, with the node with the highest attribute being given a rank of 0, and the node with the lowest attribute a rank of 1. Attributes were randomly assigned such that each node's attribute is unique from $1...N$. After 89 meetings, the network is perfectly ordered. Note the frequency of switches with respect to the number of meetings – initially, most meetings result in rank alteration. However, as the system becomes more ordered, fewer node meetings result in a rank exchange. This is similar to what is observed in many animal social hierarchies which are characterized by a high initial competition rate which decreases as the hierarchy becomes ordered. The corresponding maximum rank error plot for the trajectory of Fig. 4.4 is shown in Fig. 4.5. It can be seen that the error decreases rapidly, with the maximum error of any node being less than 30% after 30 meetings. However, to reach perfect order, it takes a further 60 meetings.

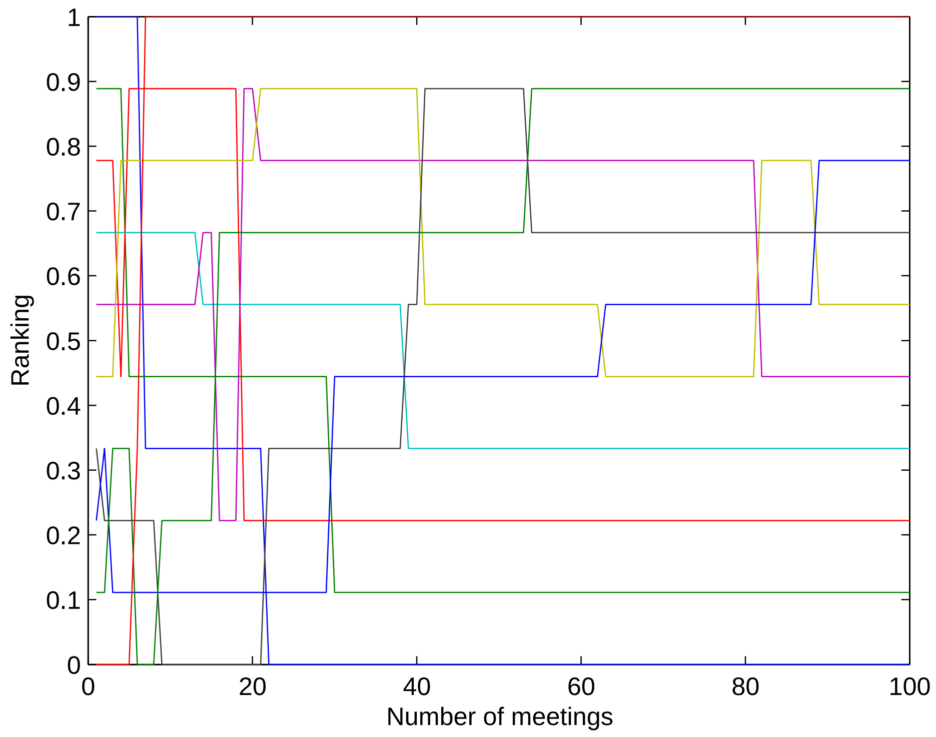


Figure 4.4: Rank trajectories for a 10 node network using pairwise ASH, starting from perfect reverse order.

This simple method of sorting is used as a baseline to compare other methods against. The possible transitions between all the possible orderings, as well as the probability of the transition are shown in Fig. 4.6 for a three node system. Note that the transition graph is a directed acyclic graph (DAG), as it can be topologically sorted. In addition, the Kendall correlation coefficients are strictly increasing with the transitions. This has the implication that any meeting that results in a transition will result in a more ordered network. It is impossible for a meeting to result in a network which is less ordered.

An upper bound can be placed on the expected number of meetings by examining how long it will take if the system traverses the maximum number of switches from perfectly disordered to perfectly ordered. The maximum number of switches is given by $N_s = \frac{N(N-1)}{2}$, which is the same as the number of possible combinations in which two unique

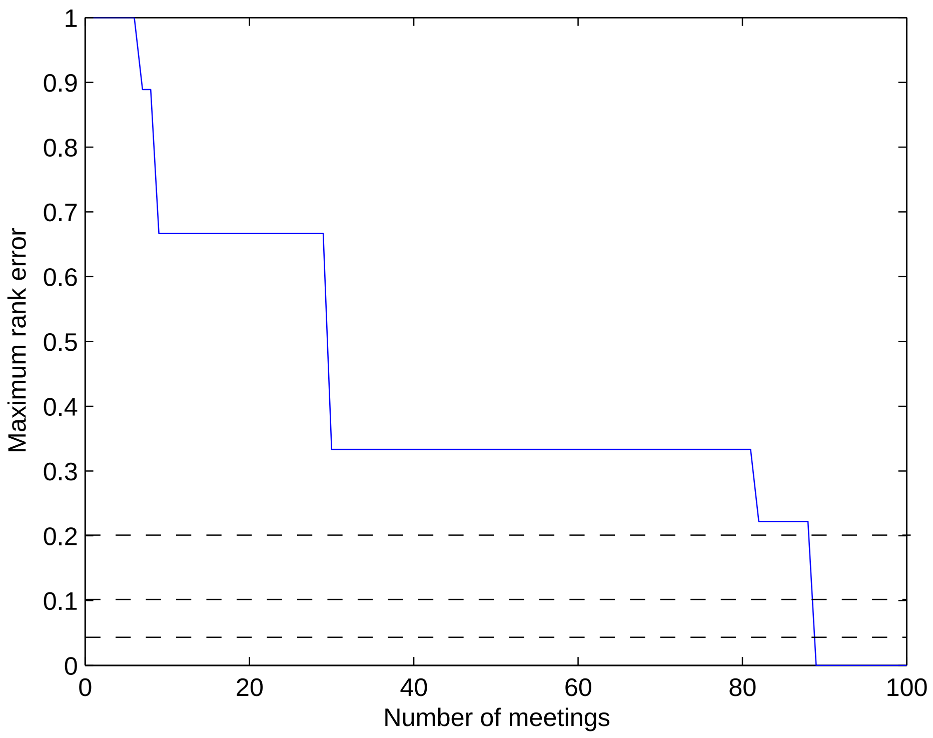


Figure 4.5: Maximum rank error, e_{max}, for the trajectories shown in Fig. 4.4. Note that the network is perfectly ordered after 89 meetings. The error thresholds corresponding to the determination of $T_{\pm 20\%}$, $T_{\pm 10\%}$ and $T_{\pm 5\%}$ are shown on the diagram as dotted lines.

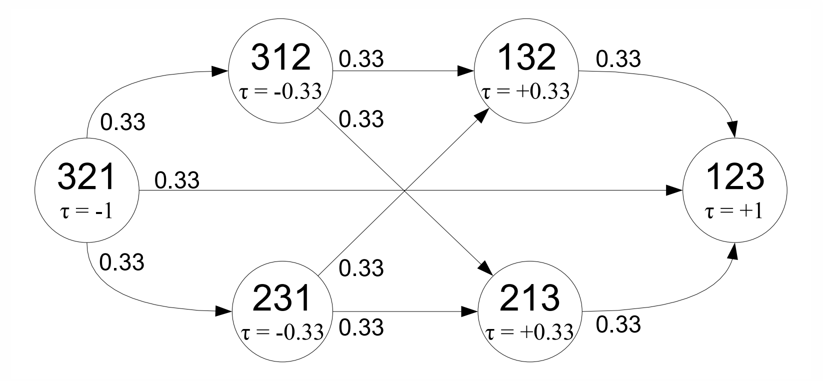


Figure 4.6: Transition diagram for the possible sequences of a three node network. Probabilities next to each arrow are the exit probabilities from each state. The τ value shown is the Kendall rank correlation coefficient. It should be observed that τ is monotonically increasing from left to right, and that the transition graph results in strictly increasing values of τ.

nodes can be picked at random. For a three node system, this results in the maximum number of switches being 3. For a four node system, the maximum number of switches will be 6 and a five node system will have a maximum number of switches as 20. Thus, the maximum bound grows in the order $O(N^2)$.

It will now be shown how to calculate the upper bound on the expected number of meetings, given that the system starts from a perfectly reversed rank order, that is, with sorted rank order [3;2;1]. From this ordering, every possible meeting results in a transition to a new sequence. There are three possible permutations of a pair of nodes meeting, and there are three possible switches that can be undertaken. Thus, the time that the system remains in this sequence before leaving is three meetings out of three possible switches, giving a dwell time of one meeting. Consider if the next ordering that is arrived at is [3;1;2]. From this point, two out of a possible three meetings will result in a transition to a new ordering. Hence, we can expect the system to remain in this ordering for $3/2$ meetings. Lastly, consider if the rank ordering arrived at is [1;3;2]. Only one meeting (one between nodes 2 and 3) will result in a transition to the correct order of [1;2;3]. The system will be expected to take $3/1$ meetings on average to arrive at the correct ordering. Once in the final state, the system will never leave. Thus, the total expected time to transition from completely disordered to perfectly ordered is the sum of the average number of meetings required to reach this state, which is $3/3 + 3/2 + 3/1 = 5.5$ meetings. In general the upper bound on the expected number of meetings can be written as

$$T_{max} = \sum_{i=1}^{N_s} \frac{N_s}{i} = N_s \sum_{i=1}^{N_s} \frac{1}{i}, \tag{4.4}$$

where $N_s = \frac{N(N-1)}{2}$.

For large N, this can be expressed as

$$T_{max} = N_s(\ln(N_s) + \gamma_e), \tag{4.5}$$

where $\gamma_e = 0.57721...$ is the Euler-Mascheroni constant [258].

Thus, in the limit, this shows that the upper bound on the settling time varies as $O(N_s \ln(N_s))$, or in terms of N as $O(N^2 \log(N))$. This has the implication that the required number of meetings for the system to converge perfectly appears to be prohibitively large as N increases[5]. Later, in Section 4.6.3, it is shown that in the more realistic scenario where more than one node can meet per time interval, the effect of increasing node density counteracts the polynomial convergence time, resulting in a decrease in convergence time with increasing N.

[5]The pairwise swap is similar to the standard Bubble Sort, except executed in a distributed manner. Parallel sorting algorithms have been studied by the computer science community, but their relevance to this analysis is slight, as communication amongst parallel machines can be regarded as being largely deterministic, whereas in this case, communication is randomized.

However, whilst the result of Eq. 4.5 is useful to place an upper bound on the performance of the switching scheme in terms of the expected number of meetings, in reality the meetings are random and thus the system is unlikely to pass through every possible combination to reach the ordered state. In addition, initial node rankings are likely to be random, not in perfect disorder.

To calculate the expected settling time, this problem can be formulated as a Markov Chain [259]. The terminal state, corresponding to perfectly ordered, is called an absorbing state, and the ROC is equivalent to the expected absorption time. The possible state transitions are expressed as a transition matrix. This indicates the probability of moving from one state to another. The transition matrix for a system with N nodes is denoted by P_N. Thus, the transition matrix for the case of N = 3 is given by

$$P_3 = \begin{array}{c|cccccc} & 321 & 312 & 231 & 132 & 213 & 123 \\ \hline 321 & 0 & \frac{1}{3} & \frac{1}{3} & 0 & 0 & \frac{1}{3} \\ 312 & 0 & \frac{1}{3} & 0 & \frac{1}{3} & \frac{1}{3} & 0 \\ 231 & 0 & 0 & \frac{1}{3} & \frac{1}{3} & \frac{1}{3} & 0 \\ 132 & 0 & 0 & 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ 213 & 0 & 0 & 0 & 0 & \frac{2}{3} & \frac{1}{3} \\ 123 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \tag{4.6}$$

There are some things to notice about the P matrix, which will be later used to simplify the calculations. Firstly, it is of size $N! \times N!$, which has the implication that large N results in state space explosion. It is strictly upper triangular, as it is impossible for two nodes to meet and result in the system becoming more disordered. The first element on the diagonal is 0, as any meeting will result in an exit from this state. The last element on the diagonal is 1, as once the system is in this state it can never exit, corresponding to the absorbing state.

To determine the expected time to absorption, we express P in canonical form as

$$P = \left[\begin{array}{c|c} Q & R \\ \hline 0 & I \end{array}\right]. \tag{4.7}$$

Using a standard result [259] from the theory of Markov Chains, the expected time from each state to absorption is given by

$$D = (I - Q)^{-1} = I + Q + Q^2 + Q^3 + \ldots. \tag{4.8}$$

D is a vector of dwell times, and the mean time to absorption from any state (assuming there is an equal chance of starting in any state, including starting off perfectly ordered) can be calculated as [259]

$$t_0 = \frac{\sum_{i=1}^{N!} D}{N!}. \tag{4.9}$$

For P_3, the mean time to absorption is 3.167, which means that on average, just over three pairwise meetings are required to perfectly sort the system.

To prevent state explosion, we attempt to reduce the size of the state universe. We note, that although the number of possible states varies as $N!$, the number of possible exchanges only grows as $\frac{N(N-1)}{2}$. We thus redefine our states as the expected number of exchanges to reach perfect order. This smaller matrix will be numerically easier to invert in order to find the expected time to absorption.

We define the reduced state transition matrix as S_N. The reduced transition matrix for a three node network is given by

$$S_3 = \begin{array}{c|cccc} & 3 & 2 & 1 & 0 \\ \hline 3 & 0 & \frac{2}{3} & 0 & \frac{1}{3} \\ 2 & 0 & \frac{1}{3} & \frac{2}{3} & 0 \\ 1 & 0 & 0 & \frac{2}{3} & \frac{1}{3} \\ 0 & 0 & 0 & 0 & 1 \end{array} \tag{4.10}$$

The expected time to absorption still evaluates as 3.167, indicating that this approach of reducing the state universe is valid. However, even constructing the smaller matrix and inverting it is tedious. Table 4.1 shows the expected mean time to absorption as N varies for the various methods. Also shown is the convergence time as evaluated by the Monte Carlo simulation of the ranking process. Note that in the simulation, nodes meet at random, with a pair of randomly chosen nodes meeting at each point in the simulation time. In reality, it is unlikely that meeting patterns will be entirely random, especially if the nodes are attached to wild animals. In this situation, one would expect some animals to meet more frequently than others. For example, it would be expected that animals are more likely to have intra-species interactions as opposed to inter-species interactions. These issues are discussed more fully in Section 4.6.3, and for the sake of analytical tractability, random mobility models are used as they are well studied.

To this end, the following expression is hypothesized to be an approximation to the expected convergence time for large N. This was deduced through curve fitting for different values of N, and the prior result from Eq. 4.5.

$$T_0 = \frac{N_s}{2} + N_s(ln(N) + \gamma) \tag{4.11}$$

Table 4.1: Convergence times as predicted by the various methods for different N. Also shown is the upper bound on the expected settling time, T_{max}.

N	T_0 (Full Markov)	T_0 (Reduced Markov)	T_0 (Simulated)	T_{max}
3	3.17	3.17	3.18	4.50
4	9.13	9.14	9.13	11.20
5	18.90	18.93	18.89	21.67
6	32.79	32.82	32.81	36.24
7	51.02	51.04	51.15	55.13

A diagram showing the Monte Carlo simulated convergence time compared against the approximation given by Eq. 4.11 is shown in Fig. 4.7. This shows that the approximation achieves a good correspondence to the Monte Carlo simulated convergence times. For comparison, the upper bound of expected convergence time, T_{max} is also shown.

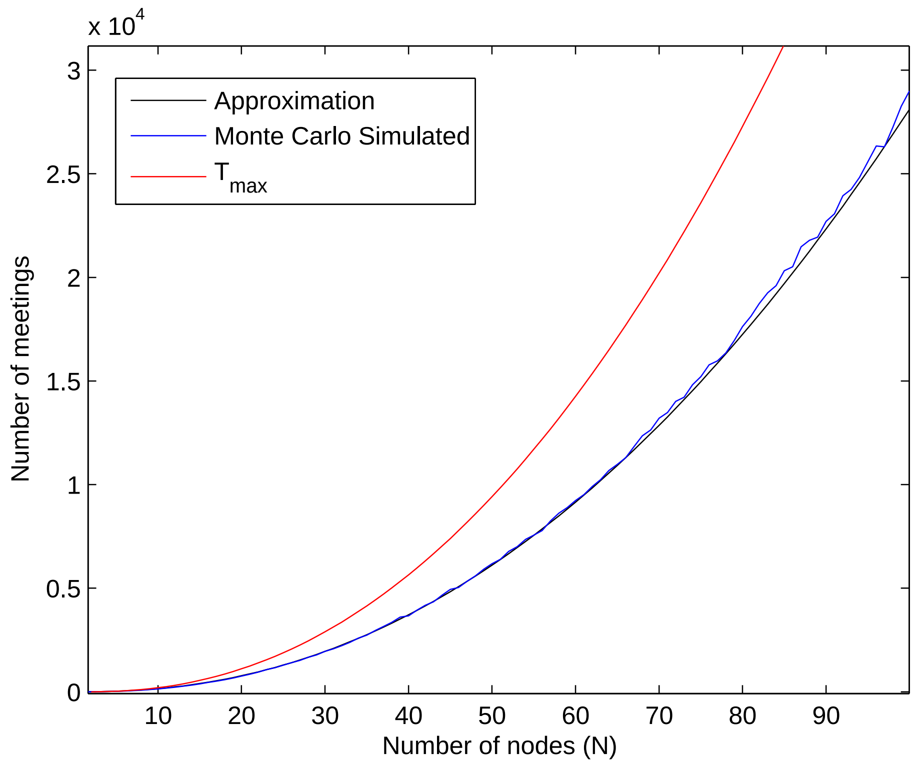


Figure 4.7: Variation in Rate of Convergence against N for different error thresholds. The diagram shows that the approximation of 4.11 accurately predicts the convergence time as estimated by the Monte Carlo simulation. The upper bound on the convergence time is also shown.

Until this point, the time taken for the system to be perfectly converged has been considered as a guideline. However, the question that must be posed is 'how close to correctly ordered, must the system be to behave well?' The acceptable error tolerance of the rank estimator depends on the application that the ranking is used for. As such, we introduce different error thresholds which can be used to characterize how quickly the ranking method converges to an acceptable tolerance. Consider that once a node has determined its approximate rank in the network, the decisions made by the node are generally not fine-grained. For example, consider the case of a node ranking its energy within the network, and using the ranking to determine its sleep-wake duty-cycle. The node behaviour will be the same whether its rank is 0.161 or 0.162, as it is able to determine that it has a low amount of energy relative to its peers.

Error plots for the rank estimator are shown in Fig. 4.8 for two different error thresholds ($\epsilon = 0.5$ and $\epsilon = 0.05$) for a 100 node network. Fig. 4.8 (a) shows that a maximum rank error of 0.5 leads to a wide spread of the ranks around the correct value. Using the example above, this could lead to a low energy node erroneously assuming a very high duty-cycle. Fig. 4.8 (b) demonstrates that for a maximum rank error of 0.05, the correspondence between actual rank and estimated rank is very close to linear.

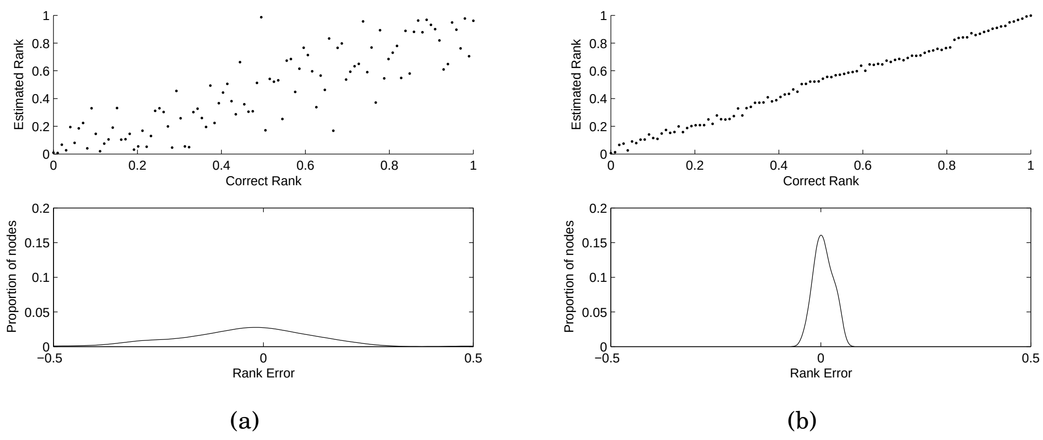


Figure 4.8: Ranks and the distribution of errors for two different error thresholds. Note how the distribution of the error becomes more peaked as the ϵ value is decreased, and how the distribution is approximately normal. The top plots show the scatter between estimated and correct rank and the bottom plots show the distribution of the error. (a) Error diagrams for $\epsilon = 0.5$ (b) Error diagrams for $\epsilon = 0.05$

Fig. 4.9 shows how $T_{\pm 20\%}$; $T_{\pm 10\%}$ and $T_{\pm 5\%}$ vary with increasing N[6]. For comparison, T_0 is also shown. It can be seen that allowing for a looser definition of converged results in more rapid settling times. In addition, the growth of the convergence times with increasing N slows with increasing error tolerance. This can also be seen in the plot in Fig. 4.5 which shows the rapid drop in the initial error, followed by a long time

[6]Note the 'sawtooth' pattern in the plots. This is due to the effect of the quantization of ranks into error bins. The sawtooth pattern is most dramatic for $T_{\pm 5\%}$, as for this scenario the ranks must be within 5% of their final value for the system to be deemed settled. The sawtooth pattern has a 'frequency' of $100\%/5\%$ as this is the length of the quantization bin.

to converge to perfectly settled. Based on simulations and the previously determined results, it was found that the rate of convergence with N tended towards a log-linear relationship, where the coefficients were empirically determined by curve fitting:

$$T_{\pm 20\%} = 3.32N(\ln(N)) \tag{4.12}$$

$$T_{\pm 10\%} = 7.08N(\ln(N)) \tag{4.13}$$

$$T_{\pm 5\%} = 14.2N(\ln(N)) \tag{4.14}$$

These equations show that the number of meetings required for the system to converge increases in sub-polynomial time, illustrating that this simple protocol will scale acceptably to large N. It should be noted that this is for the case when there is one pairwise meeting per unit time. In a realistic network scenario (which is encountered in Section 4.6.3), the number of meetings per unit time depends on the underlying mobility model. The mobility model is deliberately decoupled from this analysis in order to provide general results that can be applied to many different application scenarios.

4.4.1 Pairwise ASH with reinforcement

In Section 4.4, it was assumed that the initial ranks were equally spaced over the extent of the ranking interval. The restriction of equally spaced ranks makes initialization complex. Worse still, adaption to node insertions and removals is impossible, as the former will result in duplicate values and the latter in gaps in the ranking order. This makes scaling to dynamically varying numbers of nodes in the network impossible, precluding its use in most scenarios. In this section, a method is presented which does not require the number of nodes in the network to be known prior to deployment – the method is able to dynamically adapt to node insertions and removals. By using the reinforcing technique, ranks will automatically spread out uniformly, without requiring the interval to be known in advance.

To deal with this problem, nodes on startup choose a random normalized rank value between 0 and 1, with no input from their peers[7]. Based on their meetings, nodes switch their rankings as in the prior section. However, in order that the ranks converge to be

[7]In the case where some rank information is known *a priori*, initial ranks can be chosen to reflect their order. Alternatively, all ranks can be initialised to have the same value, but this will result in slow convergence, as convergence will be reliant on reinforcement, rather than the rapid switching action. Thus, although node ranks are initially randomly assigned, there is no restriction on the 'quality' of the random number generator that is required.

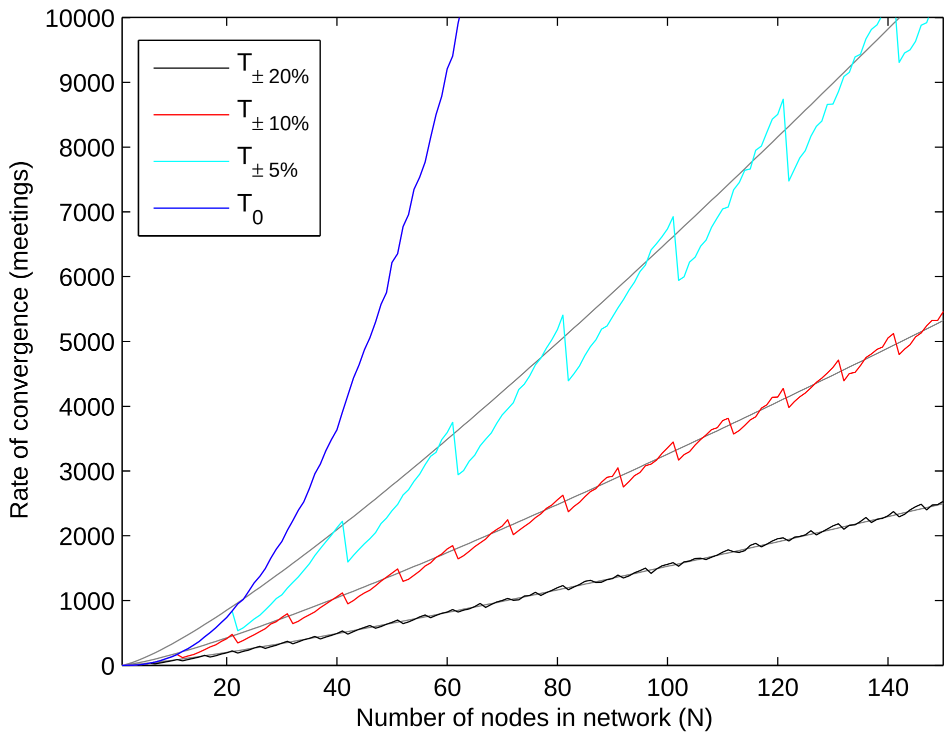


Figure 4.9: Variation in Rate of Convergence against N for different error thresholds. The approximations to the ROC for the differing thresholds are shown in grey. Note the slow growth of $T_{\pm 20\%}$ compared with T_0.

equally spaced over the interval[8] [0;1], regardless of the number of nodes in the network, an additional update rule is introduced. If two nodes meet and the order of their ranks are in agreement with the order of their attributes, each node reinforces its rank. The node with the lower attribute reinforces its rank towards 0 and the node with the higher attribute reinforces its rank towards 1. A simple method for reinforcement upwards is

$$E = E(1-\alpha) + \alpha \tag{4.15}$$

and reinforcement downwards as

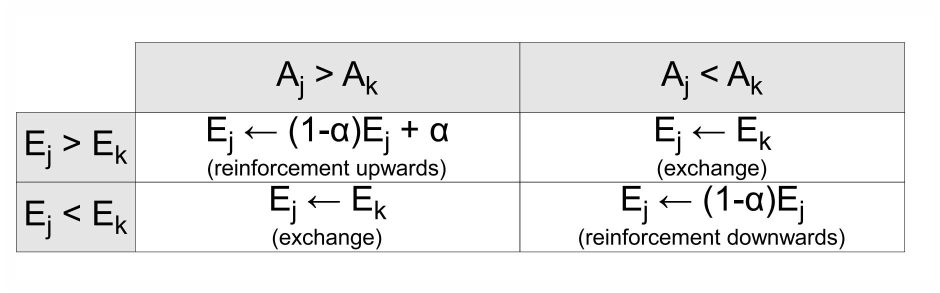
$$E = E(1-\alpha), \tag{4.16}$$

both under the condition that $0 < \alpha < 1$, where α is the reinforcement parameter. The ranking update rules are shown graphically in Fig. 4.10. For example, assume nodes j and k meet, and the attribute of node j is greater than that of node k (i.e. $A_j > A_k$).

[8]Recall that ASH implements a linear dominance hierarchy, and thus nodes are ranked in accordance to their ordinality.

This means that the rank of node j should be greater than that of node k, if the ranks are correctly ordered. However, assume that this is not the case, and in fact the rank of node j is less than that of node k (i.e. $E_j < E_k$). Node j will thus adopt the rank of node k, and, at the same time, node k will adopt the rank of node j. Thus, the nodes switch their ranks with one another. Conversely, if there is no rank contradiction, each node reinforces its rank, with one node reinforcing toward 1 and the other toward 0[9]

A large α results in rapid rank updates, but leads to rank instability, whereas a very small α leads to little success in spreading the ranks equally. Note that for the case when $\alpha = 0$, this method devolves to that presented in Section 4.4. Thus, this method can be regarded as a more generalized version of pairwise ASH.



	$A_j > A_k$	$A_j < A_k$
$E_j > E_k$	$E_j \leftarrow (1-\alpha)E_j + \alpha$ (reinforcement upwards)	$E_j \leftarrow E_k$ (exchange)
$E_j < E_k$	$E_j \leftarrow E_k$ (exchange)	$E_j \leftarrow (1-\alpha)E_j$ (reinforcement downwards)

Figure 4.10: Rank update rules for pairwise ASH with reinforcement. This allows for dynamic insertion and removal of nodes, resulting in equal spacing of ranks over time. The reinforcement parameter, α controls the rate of adaption.

Rank trajectories highlighting how this method can handle node insertion are shown in Fig. 4.11, and the corresponding maximum rank error plot in Fig. 4.12. At the start of the simulation, 10 nodes with random ranks are placed into the network. The reinforcement parameter was set to be α = 0.01. After 500 pairwise meetings, an additional 10 nodes with random ranks were injected into the network. This causes a spike in the maximum error, which is rapidly corrected through the switching (contradiction) action of the pairwise ASH protocol. The reinforcement action gradually causes the ranks to vary towards their correct values. The effect of random meetings can be seen as 'noise' in the ranks of the nodes.

The value of the reinforcement parameter affects how rapidly the system is able to adapt to changes in network composition. However, if the reinforcement parameter is too large, the ranks will 'chatter' and never converge to their correct value. Consider the extreme case if $\alpha = 1$. In this case, reinforcement will result in a node either having a rank of 0 or 1, depending on the direction of reinforcement. Thus, there will be no other values

[9]In the case where nodes have equal rank, their ranks are left unchanged. Through random meetings with other nodes in the network, these nodes will converge to having the correct rank values, as rank evolutions are noisy in any case.

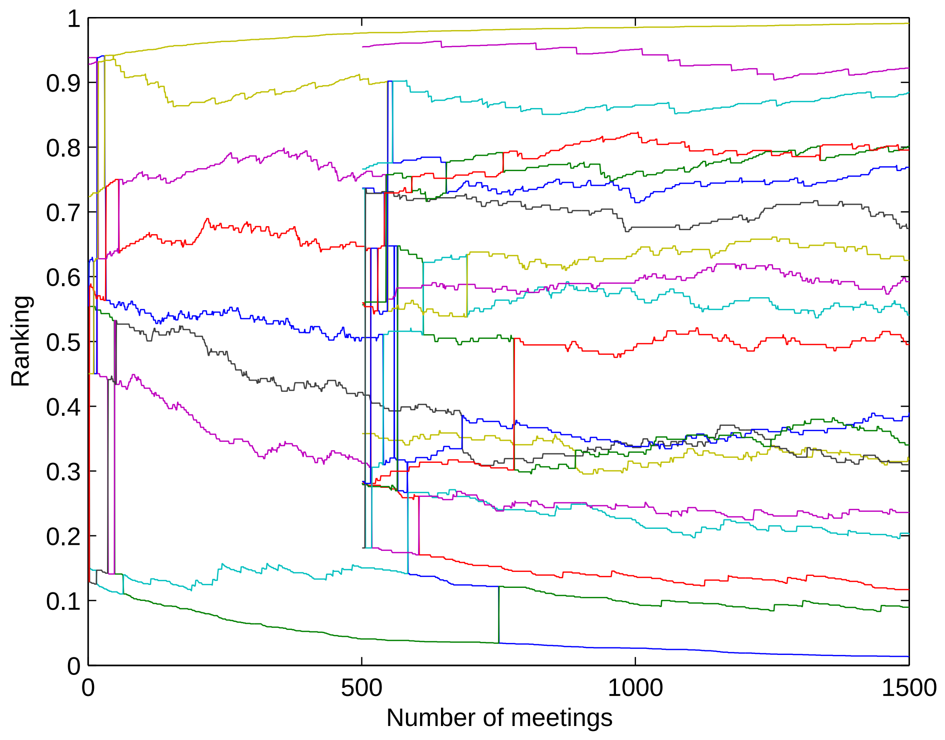


Figure 4.11: Rank trajectories for pairwise ASH with reinforcement demonstrating node insertion. The reinforcement parameter was set to $\alpha = 0.01$. Initially ten nodes with randomly assigned ranks were present in the network. After 500 meetings, another ten nodes (also with randomly assigned ranks) were introduced into the network. Note how the trajectories spread out evenly.

for the ranks in the network, leading to rapid switchings from maximum to minimum rank. To determine suitable values of α the average rate of convergence (for different error thresholds) against α for a 50 node network has been plotted in Fig. 4.13.

4.5 One Way ASH (1-ASH)

The algorithms presented so far deal with the case when a pair of nodes meet and trade their attributes and ranks. In a more typical network scenario the broadcast nature of the radio medium should be used – from one transmitter to many receivers.

For many routing protocols, nodes emit 'beacons' as network discovery packets (commonly termed 'Hello' packets [260]). ASH parameters can piggyback on top of these 'Hello' packets or the beacons presented in Chapter 3 such that they do not lead to a detrimental increase in network overhead. When nodes are in receive mode, discovering active nodes within their neighbourhood, they can update their ASH ranking accord-

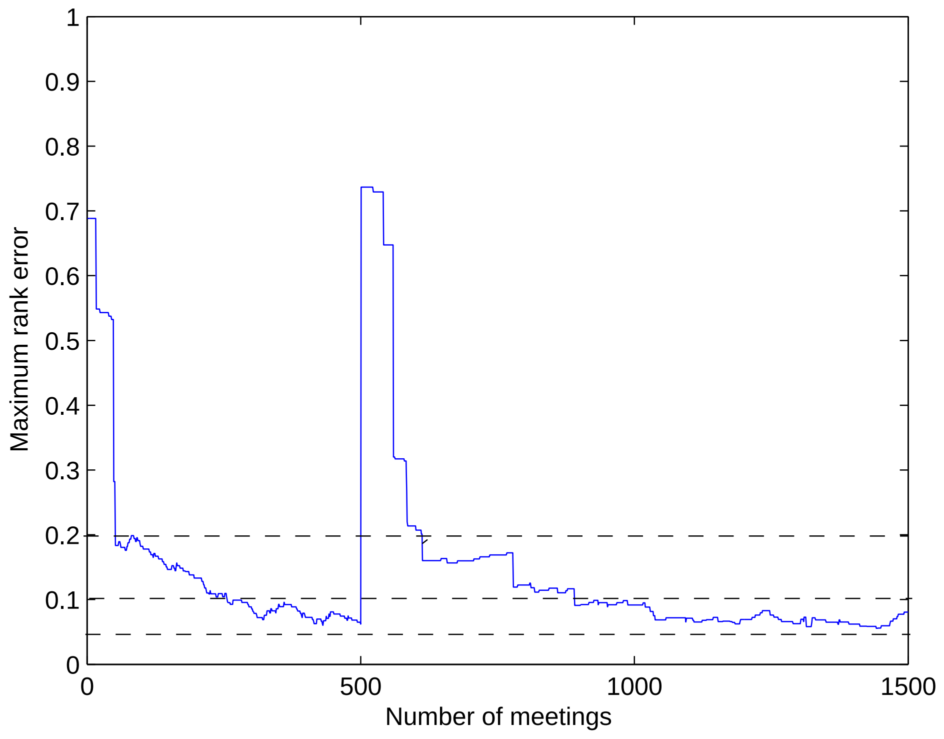


Figure 4.12: Maximum rank error for the trajectories shown in Fig. 4.11. The error thresholds corresponding to the determination of $T_{\pm 20\%}$, $T_{\pm 10\%}$ and $T_{\pm 5\%}$ are shown on the diagram as dotted lines.

ing to the transmitted ASH attribute/rank pairs. The issue with this approach, is that although the receiver nodes can update their rankings according to the newly acquired information, the transmitting node is unable to update its ranking until it later switches to receive mode. This is essentially an asynchronous method of updating the ranks, and it can lead to slower convergence and more churn.

4.5.1 Domination ASH

A hypothesized method of forming linear dominance hierarchies in nature is thought to be the win/loss ratio [22]. In this method, each node tracks how many nodes it dominates (i.e. the number of nodes which it exceeds in the value of its attribute) relative to the total number of nodes it meets. The rank update rules are shown in 4.14. Every time a node is met, the total number of observed nodes, M, is increased by 1 (this is shown in the rank update rules as $M++$, which is the same as $M \leftarrow M+1$). If the attribute of the receiving node dominates that of the transmitter, the receiver's win counter, W, is also increased by 1. Note that the rank information received from the transmitting node

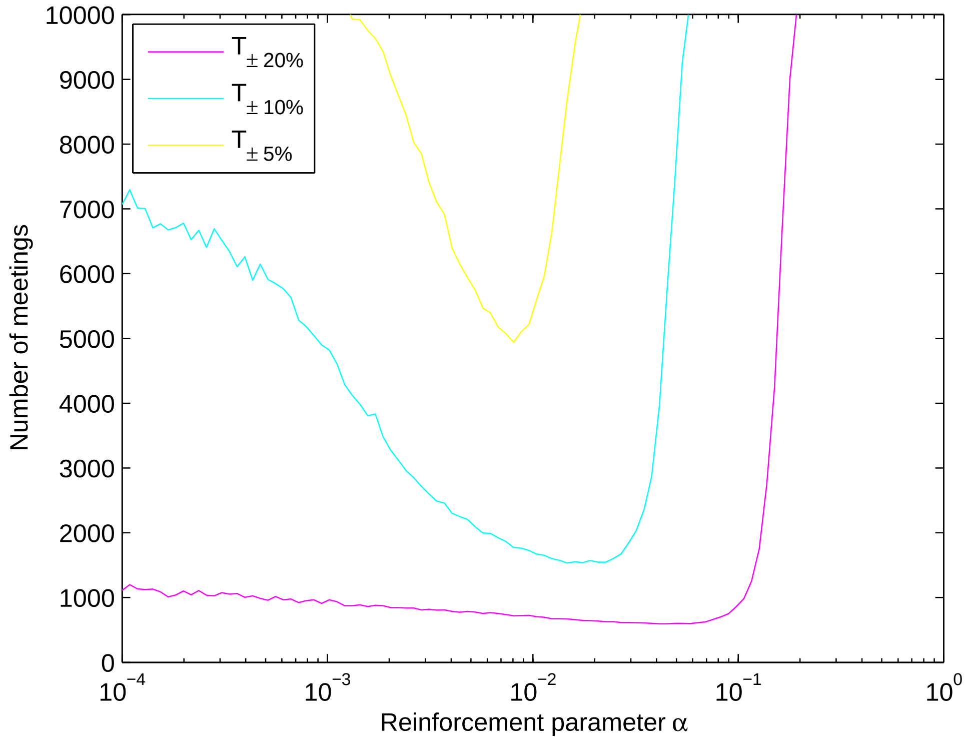


Figure 4.13: Rate of convergence of a 50 node network with α for the various error thresholds, using pairwise ASH with reinforcement. Note that large α results in excessively high (or infinite) convergence times.

plays no part in updating the node's rank. Initially, W and M are both set to zero.

	$A_j > A_k$	$A_j < A_k$
$E_j > E_k$	$E_j \leftarrow$ (W++)/(M++) (domination)	$E_j \leftarrow$ W/(M++) (submission)
$E_j < E_k$	$E_j \leftarrow$ (W++)/(M++) (domination)	$E_j \leftarrow$ W/(M++) (submission)

Figure 4.14: Rank update rules for domination ASH. As this is a one way process, only the receivers update their ranks in relation to the transmitted attributes. W is a node variable which records the number of nodes dominated and M is the total number of nodes met. $M++$ means 'increment M by one'.

As this is a ratiometric measure, it is not dependent on the absolute number of nodes in the network, leading to good scalability. It is simple to compute, but adapts slowly to changes and converges slowly. On a memory constrained microcontroller, the rank can be efficiently calculated as a fractional fixed point number. As it is a fixed point number, precision depends on the number of bits used in the representation. Again, the precision required depends on the application requirements, but in many instances a single byte would provide sufficient precision, with the least-significant bit corresponding to $1/256$. Another approach is to use floating point arithmetic, although computationally expensive in comparison, results in less roundoff error. However, each node's rank converges to the correct asymptotic value with increasing M. Essentially, the domination ratio can be thought of as the probability of dominating another node chosen at random[10]. This is shown in Fig. 4.15 which demonstrates the long settling time coupled with the diminishing rank variation as the number of meetings increase for a 20 node network. After 1000 meetings, two nodes (corresponding to 10% of the nodes) attribute values are randomly changed[11]. Fig. 4.16 shows the change in error with respect to time for the simulation conducted in Fig. 4.15. The error plot shows that recovery from this disturbance is slow as M is large. In the next section, a means of improving the convergence speed by bounding M is presented.

4.5.2 Domination ratio with switching

There are two main problems with the approach of Section 4.5.1. The first drawback is that it reacts very slowly to node insertions and removals, especially for large M. The second issue is that it only uses the comparison between the attributes to update its rank. This is clear from Fig. 4.14, where it can be seen that the rank comparison has no role in updating the ranks of the nodes. To address these two issues, the rank update rules are modified slightly, incorporating the idea of switching from Section 4.4, and limiting the maximum value of M (and hence W as it is a ratiometric measure).

These new rules are shown in Fig. 4.17.

Before the update rules are executed by the receiving node, the total number of nodes that have been observed is compared against a limit B. If M exceeds B, both M and W are multiplied by a factor $(B-1)/B$. This parameter controls the 'memory' of the rank update. A large value of B leads to slow convergence but stable ranks, whereas a value of B which is too small leads to excessive rank oscillation. Rank trajectories for a 20 node network with $B = 100$ are shown in 4.18. After 1000 meetings two nodes' attributes are randomly changed to new values[12]. The corresponding error plot is shown

[10]The domination ratio can also be viewed in frequentist terms of assessing the fairness of a coin based on repeated trials.

[11]Note that for consistency between simulations, the random seed was chosen to be the same for all simulations so that fair comparison can be made.

[12]Note that for consistency between simulations, the random seed was chosen to be the same for all simulations so that fair comparison can be made.

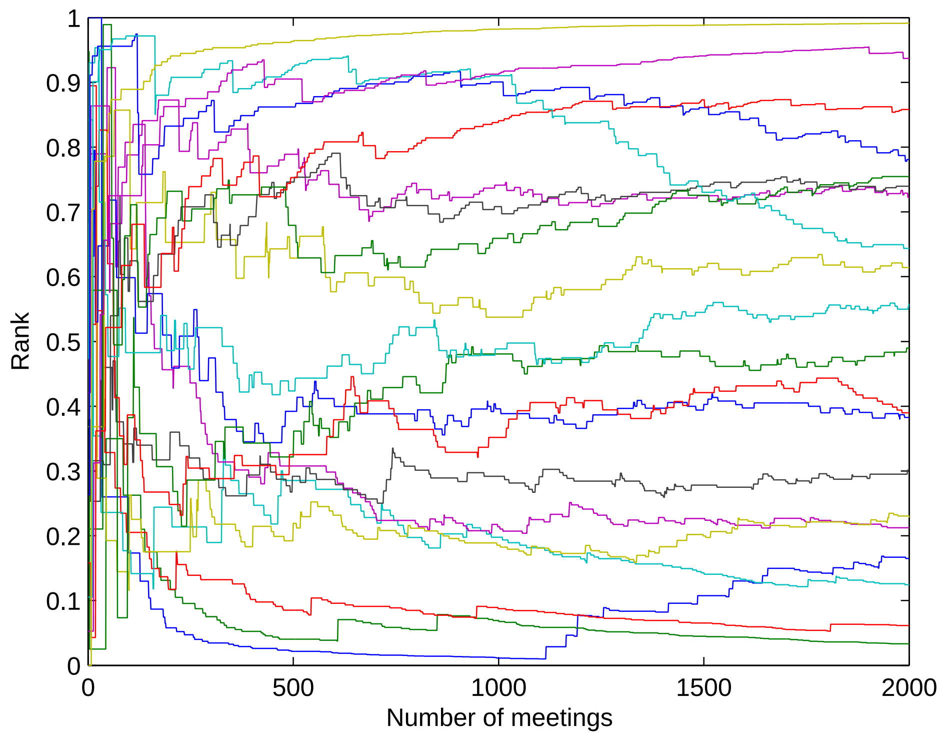


Figure 4.15: Rank trajectories for a 20 node network using one-way ASH. After 1000 meetings, two nodes' attributes are randomly changed.

in 4.19. Note how the system recovers much more rapidly from the perturbation than the previous domination algorithm with no switching (refer to Fig. 4.16).

The simulation plots shown so far deal with the case when only one node is listening to the ASH broadcast. How the ROC varies when multiple nodes listen to the same transmitter in each time interval is now considered. Fig. 4.20 demonstrates how increasing the number of receivers in a time window leads to a much more rapid rate of convergence, using one-way ASH with switching. The bound, B was set to 100. Increasing the average number of receivers results in an increase in the dissemination of information across the network.

It should be noted that if the number of receiver nodes is increased from 1 to 2, the expected rate of convergence halves, regardless of the value of N. This is an important result, as it demonstrates that it is not the proportion of nodes receiving rank information that is critical, but rather the number of nodes. Thus, it would be expected that in a network with increasing N, the important factor is the average node degree, not the edge density. Lastly, in Fig. 4.21, it is shown how one-way ASH with switching can converge more rapidly than pairwise ASH (without reinforcement), for the situation where there

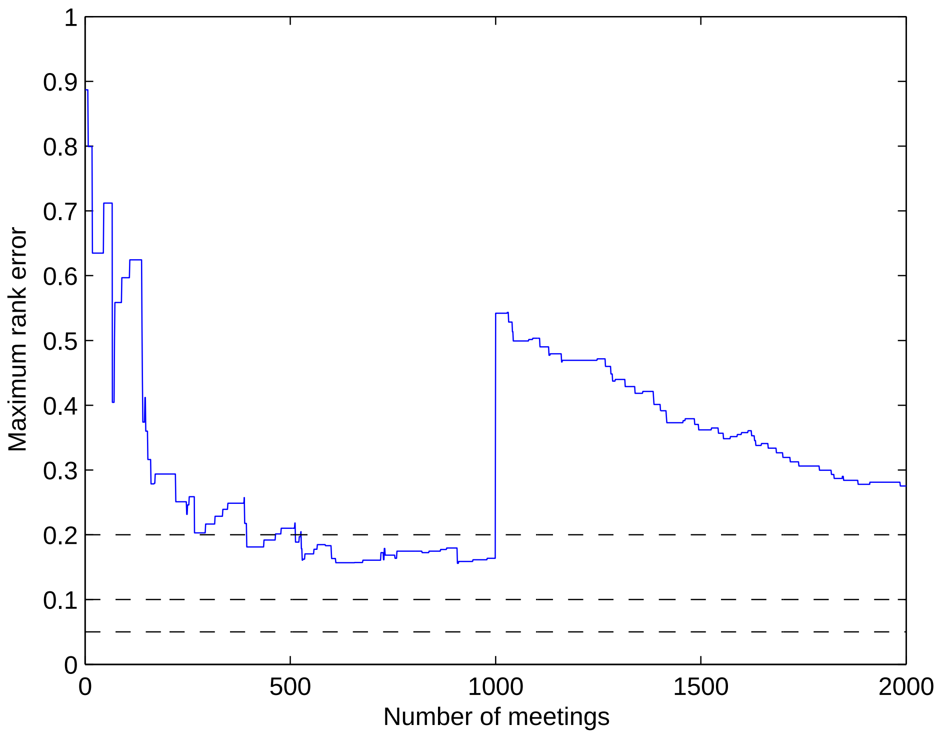


Figure 4.16: Maximum rank error for the trajectories shown in Fig. 4.15. Note how the error 'spikes' at 1000 meetings when two nodes randomly change their attributes and that the recovery from this error is very slow. The error thresholds corresponding to the determination of $T_{\pm 20\%}$, $T_{\pm 10\%}$ and $T_{\pm 5\%}$ are shown on the plot as dotted lines.

are multiple receivers (in this case 3) to each node broadcast. For the case of N = 150, a node degree of 3 corresponds to a network density of 2%. This shows that one way ASH can converge quickly even in sparsely connected networks.

Fig. 4.21 demonstrates the performance of one-way ASH compared to pairwise ASH for varying N. The ROC for $T_{\pm 10\%}$ for both approaches is shown, when each transmitter in one-way ASH transmits to three receivers. This shows that although one-way ASH suffers from asynchronous updates, it can exceed the performance of pairwise ASH whilst being more suited to the broadcast nature of the wireless medium.

4.5.3 Discussion of presented methods

Four different methods have been presented so far in this section. Pairwise ASH was first introduced, along with a Markov Chain analysis which demonstrated that convergence time could be given by $T_0 = \frac{N_s}{2} + N_s(ln(N) + \gamma)$. In Pairwise ASH, nodes exchange or 'switch' their ranks if they contradict the ordinality of their attributes. The problem

```
if (M > B):
    M = M(B-1)/B
    W = W(B-1)/B
endif
```

	$A_j > A_k$	$A_j < A_k$
$E_j > E_k$	$E_j \leftarrow (W{+}{+})/(M{+}{+})$ (domination)	$W = M \times E_k$ $E_j \leftarrow (W)/(M{+}{+})$ (exchange)
$E_j < E_k$	$W = M \times E_k$ $E_j \leftarrow (W{+}{+})/(M{+}{+})$ (exchange)	$E_j \leftarrow W/(M{+}{+})$ (submission)

Figure 4.17: Rank update rules for one-way ASH with switching. Before nodes update their rank, they first check the limit on M. If M exceeds the limit B, both W and M are proportionally reduced. The nodes then run the update rules in the table. In the event of a contradiction, the receiver will adopt the transmitter's rank.

with Pairwise ASH is that it assumes that there is knowledge about N, the number of nodes in the network, which is used to determine the interval between adjacent ranks. This is a major shortcoming, as it means that the method cannot adapt to node insertions and removals, which are characteristic of a wireless network. To address this, Pairwise ASH was enhanced through the use of reinforcement. The same switching method as the parent method is used, which is shown to lead to rapid convergence in the event of rank mismatch. Reinforcement involves driving the ranks towards the limits of 0 and 1, using an exponentially weighted moving average. In this way, nodes evenly space themselves over the ranking domain, regardless of the number of nodes in the network. This is an important result, as it means the system is scalable to large N and is able to adapt to node insertions and removals.

The pairwise methods both place a heavy burden on the MAC layer, by requiring pairwise exchange of rank-attribute information. Wireless networks are broadcast (one-to-many) in nature, and for this reason, two methods were introduced that exploited this behaviour. For example, in the EcoLocate system, beacons are used which carry ASH information. In the protocol that is used, there is no pairwise exchange of ASH information. First, inspiration was taken from win/loss ratio dominance methods from the Animal Kingdom to form Dominance Based One-Way ASH. This however was shown to have very slow convergence, and was unable to converge to $T_{\pm 10\%}$ within 1000 meetings of injection of new nodes. In Pairwise ASH, it was seen that the switching action leads to rapid convergence, and for this reason, the method of switching was incorporated, to

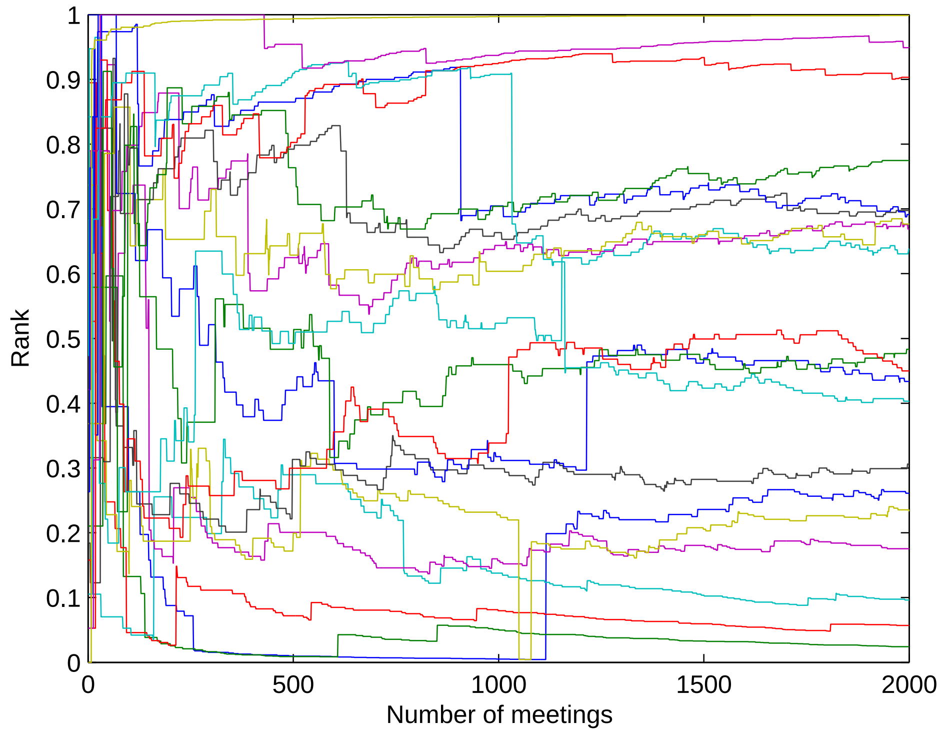


Figure 4.18: Rank trajectories for a 20 node network using one-way ASH with switching with a limit B = 100. After 1000 meetings, two nodes' attributes are randomly changed. Note the rapid switching action at the start of the simulation due to the rank exchange.

result in One-Way ASH with switching. This demonstrated that the incorporation of switching would result in convergence to $T_{\pm 10\%}$ in 300 meetings after the injection of new nodes into the network. Lastly, One-Way ASH (with switching) was compared to Pairwise ASH (with switching). Pairwise ASH would be expected to have better performance, as simultaneous switching will correctly reorder ranks, whereas a single switch does not correctly change the rank order. However, it was demonstrated that if three nodes listened to each ASH broadcast, convergence of One-Way ASH would always be quicker than convergence of Pairwise ASH.

These methods discussed in this chapter have assumed random mobility. In the next section, we consider what happens in the event of mixed mobility.

4.6 Dealing with mixed mobility: An agent based approach

The prior approaches to ranking nodes discussed in Sections 4.4 and 4.5 rely on the assumption that nodes meet at random (with a uniform probability) in order to percolate

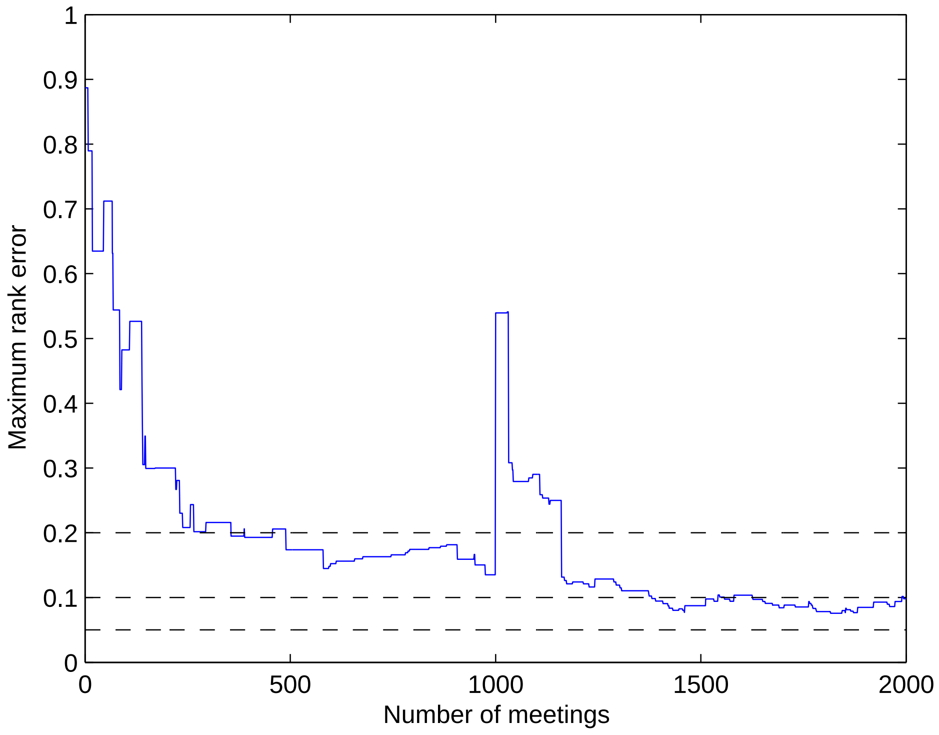


Figure 4.19: Maximum rank error for the trajectories shown in Fig. 4.18. Note how the error 'spikes' at 1000 meetings when two nodes randomly change their attributes. The error thresholds corresponding to the determination of $T_{\pm 20\%}$, $T_{\pm 10\%}$ and $T_{\pm 5\%}$ are shown on the plot as dotted lines.

the attribute/rank information throughout the whole network. If nodes are stationary, the previously presented techniques can fail as a result of the restricted neighbour horizon, leading to limited node discovery.

Pseudo-mobility can be introduced by recreating the effect of randomized meetings. Nodes listen to transmissions from nodes within their immediate radio range. If they repeat these transmissions to their neighbours, two nodes which do not have a direct connection can 'observe' each other and update their ASH ranking. This has the effect of increasing the probability of connection (even if they are not physically within range of one another) between any two nodes, increasing the network connectivity. Thus, the goal of disseminating rank/attribute pairs to a large number of nodes in the network can be achieved, leading to a more accurate representation of the social hierarchy. These re-broadcasted rank/attribute pairs are referred to as *agents*, as they can be viewed as independent carriers of information. A possible problem with the agent based approach is that agents can be carrying outdated information. Hence the churn (or rank variation with time) is expected to be higher in this scheme than in the other methods.

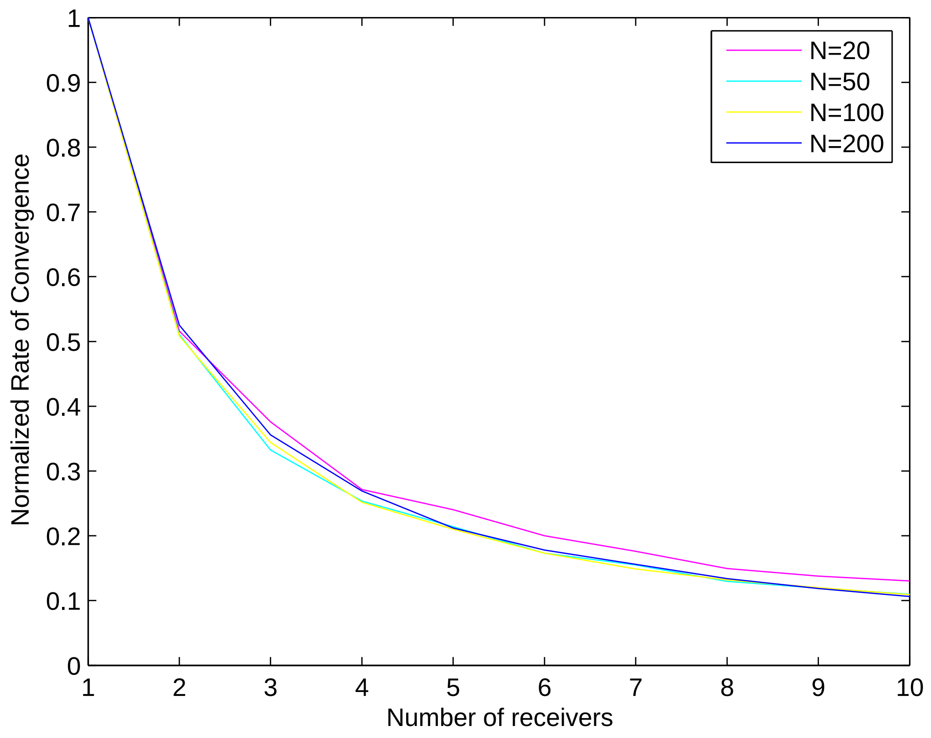


Figure 4.20: Normalized rate of convergence ($T_{\pm 10\%}$) against number of receivers for varying number of nodes in the network. The time to converge for one receiver is taken as the base figure of 100%. Note how increasing the number of receivers leads to a rapid decrease in the normalized ROC.

To prevent flooding and unacceptably high overhead, nodes only rebroadcast other node's data as part of the 'Hello' packet, as in the previous sections. Nodes select at random which rank/attributes to rebroadcast from a small local buffer, and upon overhearing new data, pick a rank/attribute pair in the buffer to replace. This way, ranks are randomly rebroadcast, without detrimentally loading nodes in the network. One-way ASH with switching can be used as a rank update mechanism, however, the domination method can suffer from poor convergence speeds in a stationary network. An alternative method, based on interpolating a node's rank, is thus presented. Before introducing the new ranking method, the way agents are created and relayed is first discussed.

4.6.1 Agent Rules

Each node has a cache of length C entries. Each entry can store one rank/attribute pair[13]. When nodes broadcast their 'Hello' packets, they send their own rank/attribute

[13]The attribute is a scalar. A vector is made of multiple attribute/rank pairs.

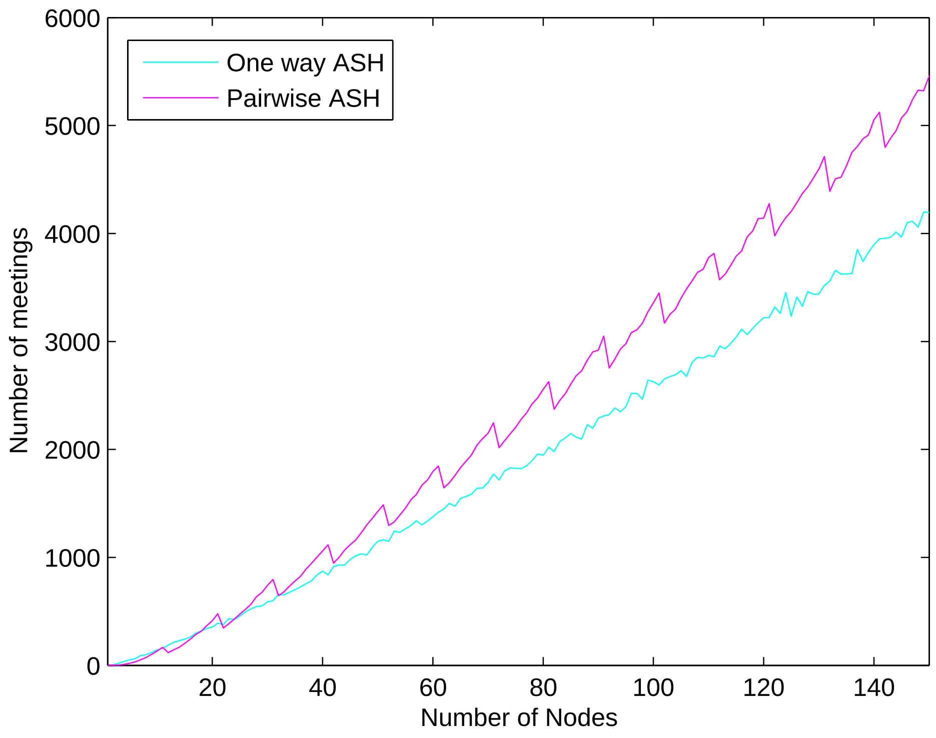


Figure 4.21: Rate of convergence ($T_{\pm10\%}$) against number of nodes for pairwise ASH and one-way ASH with switching. For one-way ASH,B = 100 and the number of receivers for each transmission was set to 3.

pairs as before. However, they now append $G <= C$ 'agents' (rebroadcast rank/attribute pairs) from their local cache. For the case where $G < C$, not all entries in the cache are re-broadcast. Each entry is thus picked without replacement at random with a uniform probability of G/C. Obviously, when $G = C$, the probability of an entry being picked from the cache is 1.

When a node overhears a broadcast 'Hello' packet, it can use this new information to update its cache. It randomly replaces entries in its local cache from the rank/attribute pairs sent in the message. There are $G+1$ entries in the message, as each node sends its own rank/attribute pair followed by G agent entries. To populate its local cache with this new information, the receiving node places each new piece of information into a random slot in the cache. Thus, the local cache is refreshed with new information as it arrives. To prevent the cache from containing duplicate information from the same node, agents also carry their ID. Cache entries are refreshed if the incoming agent has the same ID as an entry in the cache[14].

[14]To enhance the behaviour of the protocol, agents can be time-stamped by the generating node. In this way, there is no possibility of an agent carrying older information replacing one carrying more recent

The receiving node updates its rank using only the agent information carried in the message. It does not use the neighbouring node's rank/attribute information, as this will lead to bias for frequently encountered peers, distorting the ranking process. This is not to say that a node will never use its neighbour's data. It is possible that it will observe this information indirectly through the reception of another node's agent data.

This is a very simplistic approach to using agents to spread information through the network, using no state information carried in the agents to control their spread and route. However, these random agents improve the connectivity of the underlying graph by creating 'pseudo-edges', without increasing network load significantly. This effect of the agent approach is shown graphically in Fig. 4.22 which demonstrates the increase in network edge density for a 20 node stationary network.

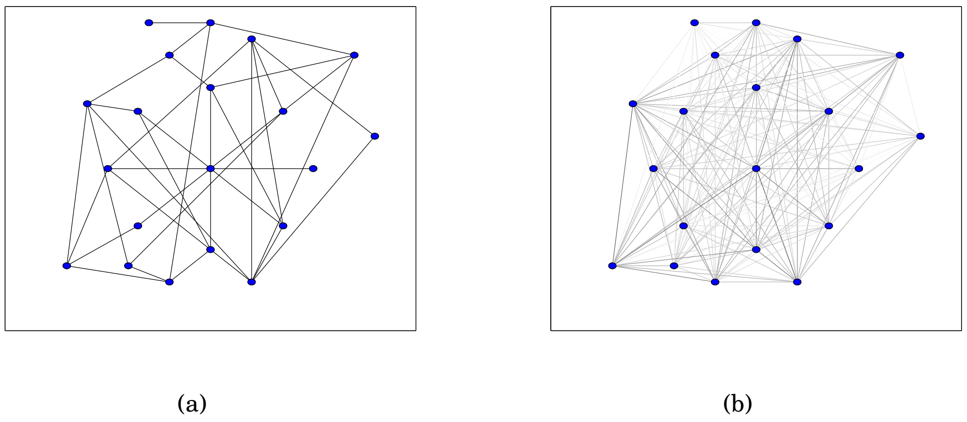


Figure 4.22: Graphs showing connectivity between nodes for a 20 node network. The graph on the right demonstrates the action of the agents which enable nodes which are not connected to each other to overhear their rank/attribute broadcasts. (a) Direct connections between nodes (indicated by solid lines) (b) Density of connectivity between nodes (line intensity denotes proportion of time edge is present)

4.6.2 Ranking Rules for Agent Based ASH

The rules used to construct the hierarchy are now presented. The domination rules from one-way ASH can be used to form the hierarchy, but an alternative approach is instead presented which demonstrates rapid convergence and good stability.

information. In this work, the aim is to keep the protocol as lightweight as possible and thus time of generation information has not been used, as performance appears to be acceptable without it. However, it would be an interesting area for future study, to determine if convergence speed and stability could be increased by removing outdated information from the network. Time-To-Live (TTL) parameters could possibly be used to determine when information is no longer useful to relay.

Consider if all nodes in the network, with the exception of one, have had their ranks correctly assigned. The node with an unknown rank can find its rank within the hierarchy by knowing only two pieces of information – the smallest rank of all the nodes whose attributes exceed the unknown node's attribute, and the greatest rank of all the nodes whose attributes are less than the unknown node's attribute. Essentially, what is needed is to find the ranks of its nearest neighbours in the ordering of attributes. The unknown node's rank will be halfway between its neighbours (in the ordering sense), as the hierarchy is linear. Hence, this method can be regarded as interpolating a node's rank based on the ranks of its peers.

To assess a node's rank, a search could be performed through all the other nodes, recording the largest rank which it dominates and the smallest rank which it is submissive to. It can be seen that this search would be simple to undertake, consist of two simple rules and only require the storage of the closest dominating and submissive rank. There are however two special nodes in the network – the node with the largest rank in the network will only have one neighbour, as would the node with the smallest rank. These two exceptions can be handled by clamping the allowable ranks to the $[0;1]$ domain, and initializing the rank which the node dominates (`RankDominate`) to -1 and the rank which the node is submissive to (`RankSubmit`), to +2. Thus, regardless of the other ranks that exist in the network, the best and worst nodes would be guaranteed to have ranks of 1 and 0 respectively. This adds an extra rule to the rank update procedure, requiring the ranks to be bounded.

However, the ranks of nodes must somehow be assessed where there is no existing rank order information to base the interpolation process upon. Each node is required to run the same rules, in an attempt to determine its rank. It updates its estimation of the smallest rank that it is submissive to and the largest rank that it dominates, based on the outcomes of tournaments undertaken everytime a new agent is received. From these estimates, it updates its rank. This information is then used by other nodes to update their ranks and so on. Based on the initialization of `RankDominate` and `RankSubmit`, all nodes have an initial rank of 0.5. This contrasts to pairwise ASH, where nodes were given random ranks on startup. The reason that nodes are given the same initial rank in this method is because there is no switching action to percolate the random initial ranks through the network. Nodes update their estimation of closest submissive and dominant ranks by incorporating information from received agents. If the attribute of the node is greater than that of the incoming agent (that is, the node dominates the agent) and if the rank of the agent is less than the current dominating rank, then the dominating rank is updated by including this new information using

$$\texttt{RankDominate} = (1-\delta)\texttt{RankDominate} + \delta\texttt{AgentRank},$$

where `AgentRank` is the rank of the agent and $0 < \delta \leq 1$ is the update or innovation parameter. If δ is set to be equal to 1, the rank that the node dominates is made to be the same as the agent's rank. Although this results in rapid update of dominant rank information, outdated information carried by agents will result in network instability. This suggests the gradual incorporation of agent rank, setting δ to a small value such as 0.1. A similar process is undertaken to update the `RankSubmit` estimation.

In the preceding scenario, it was assumed that the order of the attributes was time invariant and the number of nodes in the network static. In reality, these are both dynamic factors. Consider a node which, through the loss of the prior alpha dominant node, is now the node with the greatest attribute in the network. It is no longer submissive to any other nodes, and thus its submissive rank should be increased towards +2. The same approach should be undertaken for the lowest ranked node in the network – its dominant rank should be decreased towards -1. Thus, the ranks need to be driven towards the limits, so there will be a node with a rank of +1 and a node with a rank of 0 in the network. This suggests a mechanism that handles reinforcement.

In pairwise ASH, where reinforcement was discussed, it was shown that the choice of the reinforcement parameter resulted in a trade-off between rank stability and rate of adaption/convergence. Consider instead if the reinforcement parameter is not set to a constant value but varies dynamically according to the prior tournament results. For example, a node which is constantly winning its tournaments should increase its reinforcement parameter, so as to rapidly drive its rank towards +1. Thus, nodes with a long string of wins will rapidly ascend the hierarchy. However, if the node is itself dominated, it should reset its reinforcement parameter to the initial value of the reinforcement parameter, γ_0. The initial value should be made quite small, so that rank oscillation is minimized (values in the region of 10^{-4} to 10^{-10} appear to be useful).

Each node thus has two reinforcement parameters, corresponding to the reinforcement of the dominance rank γ_d, and the reinforcement of the submissive rank γ_s. In essence, it can be seen that we are 'reinforcing the reinforcement parameter', to reward those nodes with consecutive strings of wins or losses so they rapidly alter their ranks. Every time an agent is received, the reinforcement parameters are increased by multiplying them by a factor $\zeta > 1$. A bound is placed on the maximum value of the reinforcement parameters to prevent oscillations, by clamping them to a value γ_{max}.

Putting these concepts together, the rank update rules can be specified, which are executed every time an agent is received. These are shown in Algorithm 1.

Algorithm 1: Agent update rules

```
RankMax = 2;
RankMin = -1;
/* Reinforce the dominating and submissive ranks */
RankDominate = $(1-\gamma_d)$RankDominate + $\gamma_d$RankMin;
RankSubmit = $(1-\gamma_s)$RankSubmit + $\gamma_s$RankMax;
/* Reinforce the reinforcement parameters */
$\gamma_d = \zeta\gamma_d$;
$\gamma_s = \zeta\gamma_s$;
/* Check Limits on reinforcement Parameters */
if $\gamma_d > \gamma_{max}$ then
    $\gamma_d = \gamma_{max}$;
endif
if $\gamma_s > \gamma_{max}$ then
    $\gamma_s = \gamma_{max}$;
endif
/* Do we dominate this node? */
if Attribute > AgentAttribute then
    if RankDominate < AgentRank then
        RankDominate = $(1-\delta)$RankDominate + $\delta$AgentRank;
        /* Reset $\gamma_d$ */
        $\gamma_d = \gamma_0$;
    endif
endif
/* Are we submissive to this node? */
if Attribute < AgentAttribute then
    if RankSubmit > AgentRank then
        RankSubmit = $(1-\delta)$RankSubmit + $\delta$AgentRank;
        /* Reset $\gamma_s$ */
        $\gamma_s = \gamma_0$;
    endif
endif
/* Update the rank */
Rank = (RankSubmit + RankDominate)/2;
/* Bounds checking */
if Rank > 1 then
    Rank = 1;
endif
if Rank < 0 then
    Rank = 0;
endif
```

4.6.3 Simulation Results

Stationary Networks

Agent ASH is first compared with one-way ASH, showing that it is able to form the social hierarchy whereas one-way ASH is unable to discover all the nodes in the network, due to a limited horizon. This is shown in Fig. 4.23, where it can be seen that the agent approach is able to result in a lower error and faster convergence for a 100 node network with stationary nodes. This network was generated randomly to be fully connected with an edge density of 5%[15]. One node was randomly chosen to be the transmitter for each iteration. The cache size, C, was set to 6 entries, and the number of agents transmitted per message (G) was set to 3. This illustrates that Agent ASH can correctly discover the network ranking with a modest increase in node resources, even in sparse stationary networks. Agent ASH is not restricted to stationary networks however – Section 4.6.4 shows its performance under mobility.

Ranking trajectories for a stationary 20 node connected network with an edge density of 10% are shown in Fig. 4.24. In this example, the cache size was set to 6 and the number of agents transmitted in each message to 3. The value of γ_0 was set to 10^{-5} and the reinforcement rate $\zeta = 1.03$. The bound on the maximum reinforcement parameter, γ_{max} was made equal to 0.1. The error plot for this simulation is shown in Fig. 4.25 which demonstrates the rapid recovery from the pertubation imposed after 1000 meetings. In this simulation, for fair comparison to the other protocols, one node was chosen at random to be the transmitter at each interval in the simulation. This simulation example shows that nodes using agent ASH are able to accurately determine their rank, even in a stationary network.

4.6.4 Realistic meetings

In the prior sections, it was assumed that only one node is active and transmitting at any one point. In reality, the way nodes meet is dependent on the underlying mobility model. In a realistic network scenario, it is possible that at any point in time, that no nodes meet or more than two meet (possibly in geographically distinct locations). To account for this, the effect of the mobility model is incorporated into the way nodes meet.

The performance of Agent ASH when subject to two commonly discussed mobility models – the random walk model and the random waypoint model is investigated. In the random

[15]The mean connectivity can be calculated as the number of outgoing edges from each node. The number of possible edges for a 100 node network is $(100)(99) = 9900$. 5% of these edges have links on them, giving a total of 495 links. On average, each node will be connected to $495/100/2 = 2.48$ other nodes.

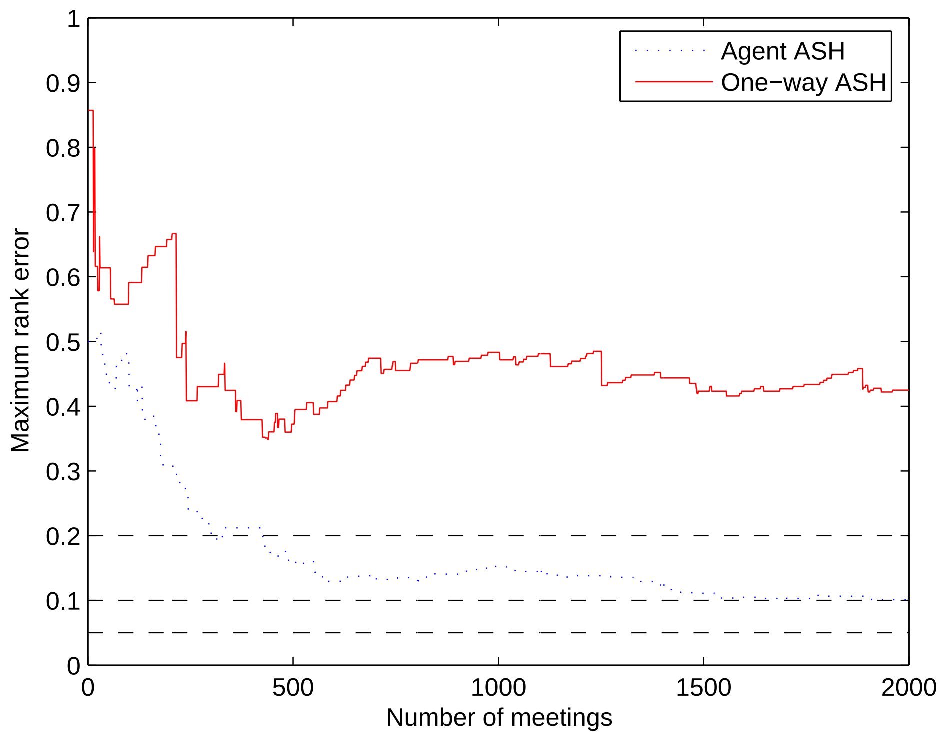


Figure 4.23: Rank errors for one-way ASH and Agent ASH applied to the same 100 node stationary network. Observe how the error for one-way ASH never drops lower than 0.35. This is as a result of the limited discovery horizon. The error thresholds corresponding to the determination of $T_{\pm 20\%}$, $T_{\pm 10\%}$ and $T_{\pm 5\%}$ are shown on the plot as dotted lines. For one-way ASH, simulation parameter B = 500. For agent ASH, G = 3, C = 6, $\gamma_0 = 10^{-4}$, $\gamma_{max} = 0.1$, $\zeta = 1.03$.

walk model, at each point in time, a node chooses to move to another location that is one-step away. In the random waypoint model, nodes travel in a straight line at a randomly chosen speed until they reach their destination. Once the destination is reached, nodes choose a new destination and velocity and travel towards that. The random walk model exhibits a very slow mixing time[16], in the order $O(K^2)$ where K is the length of one side of the simulation area, whereas the random waypoint has mixing times of the order $O(K)$ [261]. Thus, it would be expected that the performance of ASH will be worse when subject to the random walk model in comparison to the random waypoint mobility model.

For the random walk model, N nodes are placed on a $K \times K$ toroidal simulation area, where $K = 100$ units of distance. The radio radius, U, is varied from 5 to 15 units of distance. The transmission range (along with the number of nodes) controls the connectivity of the network. The rate of convergence is examined for N = 50; 100 and 200. The

[16]The mixing time is the convergence rate of the random walk [261].

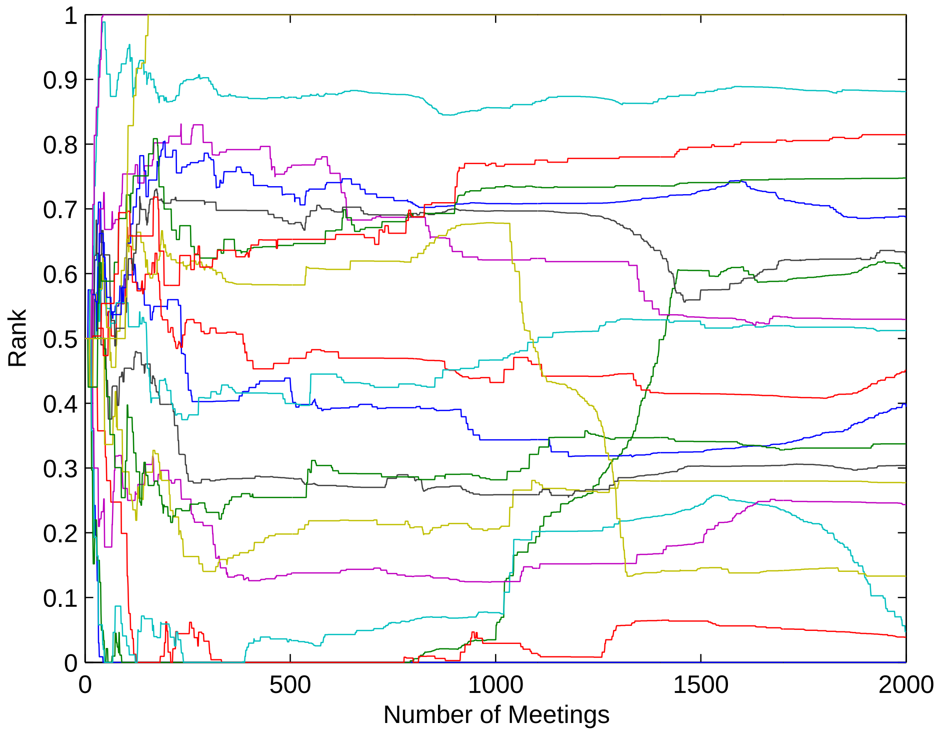


Figure 4.24: Rank trajectories for a 20 node network with C = 6 and G = 3, using agent ASH. After 1000 meetings, two nodes' attributes were randomly changed.

time taken for the system to converge to $\pm 20\%$ of its final value is examined, based on the mean time from five simulations, and denoted as $t_{\pm 20\%}$, where the lower case t indicates results where more than one node can be actively transmitting at any one time, as opposed to the number of meetings. To introduce some realism into the MAC layer, it is assumed that nodes transmit 'Hello' packets 10% of the time. For the remaining time, the nodes are in receive mode. Unless otherwise stated, the parameters for the Agent ASH model were chosen to be G = 3; C = 6 and L = 500.

The simulation results for the Random Walk model are shown in Fig. 4.26. The graph shows that the radio range has a large effect on the Rate of Convergence. Increasing the number of nodes has the effect of reducing the convergence time. Contrast this to the previous results from Section 4.4 which showed that an increase in N resulted in an increase in the number of meetings for the system to converge. This demonstrates the scalability of the ranking system - an increase in N actually results in a faster convergence time, as it results in a more dense network.

In the simulation of the Random Waypoint Model, all common parameters are kept the same as the Random Walk. Nodes have a random speed uniformly chosen between 1

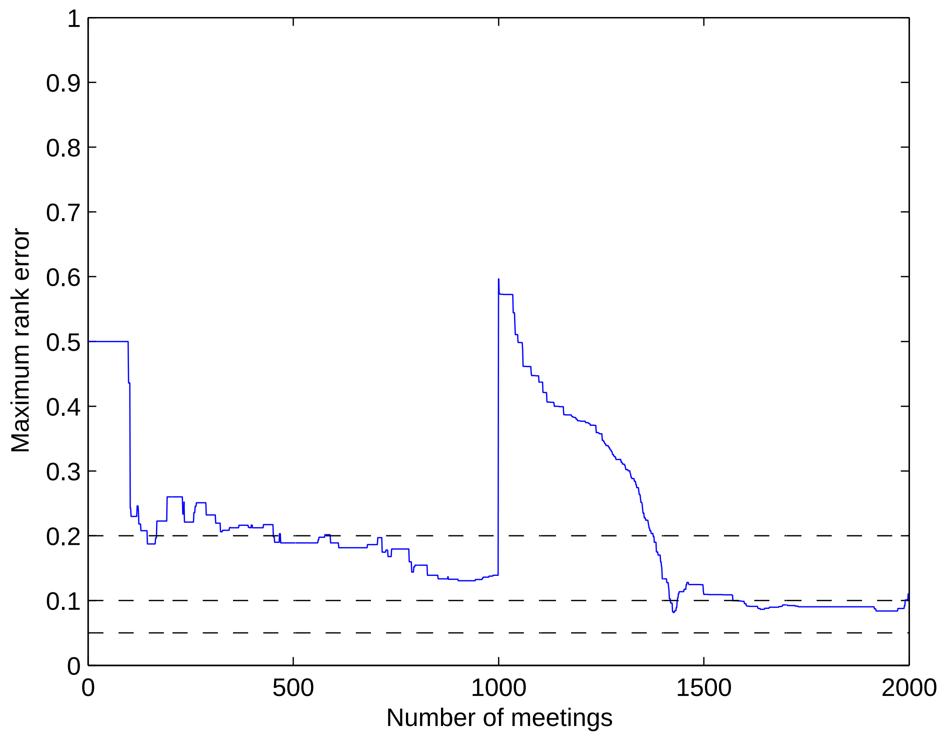


Figure 4.25: Maximum rank error for the trajectories shown in Fig. 4.24. When the attributes are randomly changed after 1000 meetings, the error sharply peaks, followed by a rapid switching action to correct the erroneous ranks. The error thresholds corresponding to the determination of $T_{\pm 20\%}$, $T_{\pm 10\%}$ and $T_{\pm 5\%}$ are shown on the plot as dotted lines.

and 2 units/s. The results from this simulation are shown in Fig. 4.27. Comparing these results to those of the random walk simulation, it can be seen how the random waypoint model leads to faster convergence as it has a faster expected mixing time. Like the random walk results, it can be seen that an increase in N results in a decrease in the ROC.

The ROC is thus related to the network density, rather than the actual number of nodes in the network. This shows, that for a network in a fixed size geographical area, increasing the number of nodes will actually allow the network to settle more rapidly. This result highlights the scalability of ASH.

The Agent ASH algorithm demonstrates that it is possible to sort or order a network according to its attributes, even if the network is purely stationary or subject to mixed degrees of mobility. In addition, only small amounts of local information are used to infer global attribute distribution. ASH piggybacks on top of existing network discovery packets, so does not present a large overhead burden. Now that various methods of forming a social hierarchy have been described, how to use this information for network

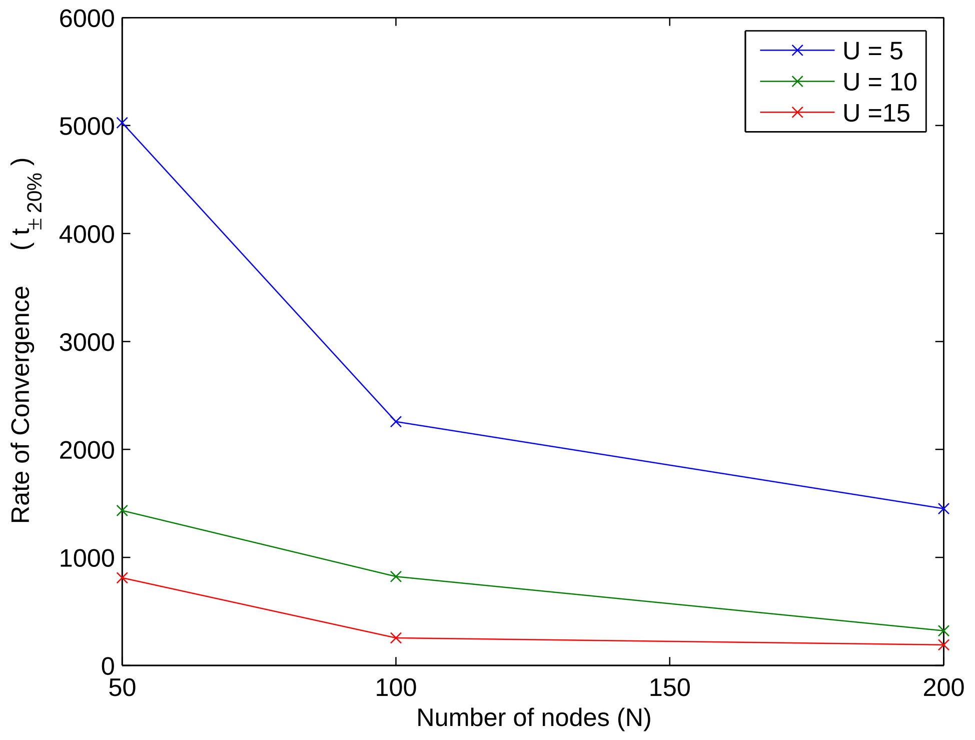


Figure 4.26: Rate of convergence to $t_{\pm}20\%$ for the Random Walk Mobility model for different numbers of nodes. Note how the radio radius (U) has a strong impact on the ROC, as it alters the connectivity of the graph.

control, access, management and routing is now considered. First though, some suitable network attributes to rank are discussed.

4.7 Suitable attributes to rank

Any network parameter which can be measured on a per-node basis can be ranked in the global sense using one of the ASH ranking methods. Suitable attributes are obviously ones which have a direct and useful impact on network performance and control. A few useful attributes are presented, which are later used in Section 4.9 to demonstrate the power and flexibility of the ASH philosophy.

4.7.1 Energy/Lifetime

Of prime importance in deeply embedded and remote networks is energy, and coupled to that, rate of use of energy. This is reflected in the vast number of energy aware network routing protocols (refer to [262] and [235] for more information). Nodes, depending on

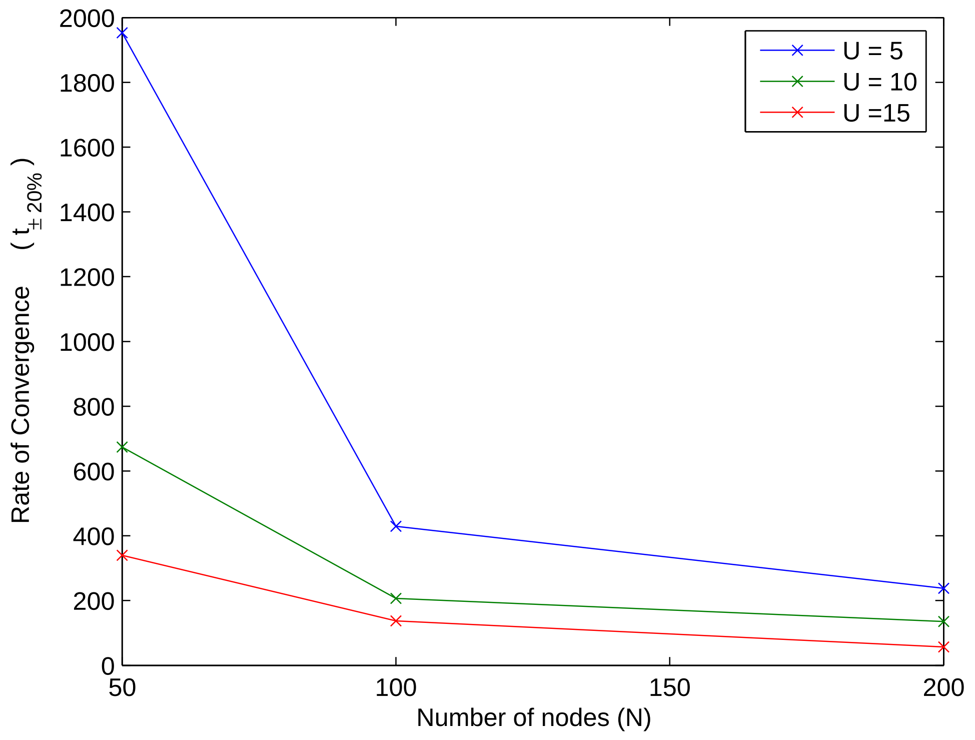


Figure 4.27: Rate of convergence to $t_{\pm}20\%$ for the Random Waypoint Mobility model for different numbers of nodes.

their hardware capabilities can measure their energy reserves and usage, in an absolute sense (such as 100J remaining). However, it is also possible to estimate energy usage in software (for example using the Contiki Operating System [263]), and knowing the size of the battery that the unit is equipped with, determine the remaining amount of energy. A node with a large amount of energy in relation to its peers can be expected to survive longer in the network. However, this assumption of survival is made under the pretext that all nodes consume energy at the same rate. Thus, a more useful indicator is estimated node lifetime, based on a long term average of prior energy usage and current reserves.

4.7.2 Connectivity

The purpose of a wireless network is to transfer information from source to sink, through intermediate nodes. The purpose of a routing protocol is to deliver the data along the 'best' possible path, where the desirability of each path depends on some underlying cost function of suitability (such as delay, hop count or redundancy). In a network for information delivery, nodes need to send information to base-station(s). At each hop, data should be sent to a node which is 'closer' to a base-station. Lindgren et al. present a method of evaluating transitive connectivity to a base-station [264], which is based on the observation that if A and B frequently meet, and B and C frequently meet, then A and C can be regarded as being transitively connected through B. In static networks, the conventional hop-count metric can be used a measure of connectivity.

Another metric which impacts the delivery of information is the local traffic density. In regions of high traffic density, it would be expected that collisions will be more frequent. Nodes can assess the local error rate, possibly by monitoring how many of their transmissions fail. This information can be used to build a ranking of the expected congestion in the network. Data can be sent along paths where the expected congestion is low, thus balancing traffic across the network.

4.7.3 Buffer space

Depending on their physical memory size and their role within the network, nodes will have varying amounts of buffer space available for messages from other nodes. This can be ranked, although the ranking would be expected to be very dynamic as buffers are cleared upon successful delivery. A long term average of available buffer space would possibly be a better metric of delivery rate.

4.7.4 Functions of attributes

Ranked attributes do not need to be based on a single measure of network performance, but can be a composite of multiple attributes, weighted in various ways. There are many ways of constructing a combined attribute. Context aware routing (CAR) combines attributes to form a single weighting [265]. Of particular interest with the CAR approach is that the availability and predictability of the various attributes is incorporated into the weighting procedure [266]. The ranking process can be performed on the combined attributes, or ranked attributes can be combined (and further ranked if necessary).

4.8 Levels and loops

In a practical network scenario, a node needs to decide whether to send a message to another node based on the difference in their ranks. If a node always sends its data to another node with a greater rank, this can lead to an explosion in traffic (the number of messages being sent per unit time in the network). In addition, there is the possibility of data forming a loop if the rank of a node varies over time. This section introduces levels, which are essentially thresholds that only allow messages to be sent when a significant gain can be made in rank ordering. In addition, we discuss how to avoid the formation of loops by recording the rank of the last-hop of a message and not allowing it to descend the rank hierarchy, even if the ranking of the current holder of the message drops. In essence this means that nodes are allowed to change rank and levels, but a message can only be sent to the next level up.

To address these issues, the rank is quantized into L discrete levels. As all the ASH methods presented in this chapter generate linear hierarchies, it can be seen that each level (or bin) will contain approximately N/L nodes once the rankings have converged and are stable. This is because the ranks are evenly spaced over [0;1], and thus each bin will have the same number of nodes. Assume that traffic is generated at a rate of λ messages per unit time, and nodes only send messages to nodes which have a greater level. The traffic density of the lowest level nodes is $D_1 = \lambda$ as they will only forward the packets they generate, and not route any other node's traffic. A level 1 node will send its messages to any node with a higher level. There are $L - 1$ levels that are higher, and they will thus each expect to receive (and subsequently be responsible for forwarding) $\lambda/(L-1)$ messages. Thus, a level 2 node will send a total amount of traffic:

$$D_2 = \lambda + \frac{\lambda}{L-1} = \lambda(1 + \frac{1}{N-1}) \tag{4.17}$$

Thus, in general, we can express the traffic density at each level in the hierarchy using the recursive equation

$$D_k = D_{k-1}(1 + \frac{1}{L-k+1}), \tag{4.18}$$

where $1 \leq k \leq L$ [232].

This can be simplified to obtain an expression for the traffic volume at level k as:

$$D_k = \frac{\lambda L}{L-k+1}. \tag{4.19}$$

This important result shows that average traffic density is *independent* of the number of nodes in the network, and only related to the number of levels in the quantized hierarchy. This is because ASH evenly spreads ranks out over the unit domain, regardless of the number of nodes.

To prevent packets from looping through the hierarchy, due to time-varying ranks, packets can be tagged with the rank of the node which sent the data to the current node. As packets can only ascend the hierarchy this means that loops cannot form. Another method is to place a hysteresis on level changes, such that only large changes in rank result in a transition to a new level.

4.9 Example scenarios of ASH used as a network protocol

It is now shown how ASH can be used as an underlay to enhance existing protocols and also to form a cross-layer protocol in its own right.

4.9.1 Enhancing Utility Based Schemes

In mobile or intermittently stationary networks, routing is a non-trivial task. This is because table based approaches cannot be realistically used, due to the overhead required in maintaining accurate information in the table. This is even more so in sparse networks, where the concept of a routing table becomes meaningless, as connectivity is the exception rather than the norm.

Instead, utility based routing protocols have been proposed [267, 264, 268, 265]. These involve each node maintaining a local value representing its usefulness (hence utility) to the network. Utility based protocols can be regarded as gradient climbing methods. There are many different metrics of utility, such as energy or connectivity related measures. However, existing utility based protocols suffer from a threshold problem. If the threshold is made too large, very little data is forwarded in the network leading to long latencies. If the threshold is too small, the network floods, leading to excessive energy consumption. There are currently no methods of dynamically determining what threshold to use in the network. Current methods pick a 'convenient' threshold value, regardless of the underlying mobility patterns. This means that network performance is uncertain and critically dependent on the chosen threshold value. In this section, we show how ASH can be used to eliminate the threshold problem, providing scalability regardless of the utility metric chosen or the underlying mobility pattern.

PROPHET

One of the first works to investigate utility based routing in mobile networks was PROPHET, which was designed for peer-to-peer data transfer [264]. In this protocol, the concept of transitive connectivity (as first introduced by Chen and Murphy [267]) was refined. Each node maintains a table of all the other known nodes in the network, along with their delivery predictabilities. When two nodes encounter one another, they update their delivery predictability, increasing it to reflect that they are likely to be good carriers for delivery of messages to each other. At each time step, the delivery predictabilities are aged, or reduced in value to reflect that they are less likely to encounter one another in the future. The last method is that of transitivity, or if node A often encounters node B, and node B often encounters node C, then node C is a good candidate to forward messages to C.

Spray and Focus

Spray and Focus [268] is a multi-copy controlled replication routing protocol. Routing is divided into two phases. The first phase, Spraying, creates multiple copies of a message around a source. There are different ways of undertaking the spraying, but the authors show that binary spraying is optimal [268]. In this situation, given that L distinct copies are to be created, the source will hand over L/2 copies to the first node it meets, keeping L/2 for itself. At each point in time, a node can hand over half the remaining copies of the message, until there are L nodes in the network, each carrying a single copy. Nodes then enter a Focus phase, where data is forwarded along a utility gradient towards the sink. A timer indicating the time of last contact with the destination is used as the utility parameter. To capture transitive connectivity, the timer value is updated if a node meets an intermediate node which was recently in contact with the destination. The use of the focus phase dramatically reduces delivery latency, whilst maintaining the benefits of controlled replication.

sCAR

Musolesi and Mascolo presented Context Aware Routing (CAR), which is a utility based routing protocol for intermittently connected networks [265]. In CAR, nodes evaluate their 'utility' which is a metric indicating the usefulness of a node in terms of network specific parameters such as connectivity or energy level. These different parameters are combined to give a single utility value for each node in the network. One of the main contributions of this work was the idea of predicting future values of the utility, using Kalman filters (or the reduced form of an exponentially weighted moving average) [265]. However, in the work it is not clear how far nodes predict the future evolution of the utility metrics into the future.

A more recent work considered the application of CAR to sensor networks with mixed mobility (so called SCAR [269]). In this scenario, sources of information (which can be fixed or mobile) deliver information to sinks (which can also be fixed or mobile). As connectivity is intermittent, information is buffered at intermediate nodes which are more likely to be able to forward the data to a sink. Data can be delivered to any sink, making this a data gathering as opposed to peer-to-peer forwarding. To reduce the delivery delay, messages are replicated to multiple neighbouring nodes before forwarding along the utility gradient towards the sinks, in a similar manner to Spray and Focus [268]. However, one copy is marked as a Master copy in an attempt to prevent over-writing.

Three measures of context information are used to decide on routing. The first, is the change rate of connectivity (CRC) which essentially is a metric of the change in the local connectivity graph, a measure of relative mobility through the network. The degree of colocation with the sink is proposed as the second measure of utility. Lastly, the remaining battery level. The battery level is the proportion of remaining energy, relative to the initial battery level.

Issues with utility protocols

Common to all these utility based protocols is the problem of deciding when to forward a copy of the message and when to keep it for later transfer. One way is simply to send the message to any node with a higher utility value. This method results in thrashing (local loops) and uncontrolled message forwarding, resulting in poor scalability. This method is used in PROPHET, and they note that it is problematic.

Another way is to use a threshold value, as used in Spray and Focus and sCAR. Nodes will only forward data when the encountered node's utility exceeds the host's utility by a certain threshold. This helps to control the number of transmissions in the network, but the choice of threshold is critical. If the threshold is too small, traffic will rise rapidly, leading to energy depletion. Conversely, if the threshold is chosen too large, forwarding will rarely occur resulting in long latencies and non-delivery of information. The threshold value is network dependent, but both in Spray and Focus and in sCAR is specified by the designer *a priori*. This static threshold value prevents the network from dynamically adapting to network conditions.

A simple example will be presented that demonstrates how ASH can address this issue, reducing the reliance on precisely chosen 'tuning' factors. A six node stationary, linear network is used as an example layout. Nodes each evaluate a connectivity metric, u, that reflects their transitive connectivity to the base-station. The base-station has a connectivity of 1 at all times, as it is the end-destination. If a node j encounters another node k with a better connectivity, it updates its connectivity,

$$u_j = (1 - \beta)u_j + \beta u_k, \tag{4.20}$$

where $0 < \beta < 1$ is the adoption rate of transitive connectivity. Conversely, if a node does not meet another node with better connectivity, or no node at all, it ages its metric using the following,

$$u_j = (1 - \eta)u_j, \tag{4.21}$$

where $0 < \lambda < 1$ is the ageing rate.

Network links are not guaranteed, and thus, we define the probability of the links being active as p_L. Agent ASH is used, as the network is stationary. Cache length C is chosen to be two entries, and the number of agents transmitted per message $G = 1$. The reinforcement update parameter γ is set to 1×10^{-5}, with $\gamma_{max} = 0.1$. Three simulations will be conducted, one base-line to compare against and two others where network parameters, namely link probability and adoption rate, are varied. The time limit for the simulation is set to T = 1000.

The first simulation used the following parameters: adoption rate $\beta = 1 \times 10^{-3}$, age rate $\eta = 1 \times 10^{-4}$ and link probability $p_L = 0.99$. The simulation results are shown in Fig. 4.28. The basestation was set to be the top lefthand node. Thus, the bottom node, which is furthest from the base-station has a very low connectivity metric. Note how the ranks are evenly spread out from 0 to 1, with the basestation having a rank of 1.

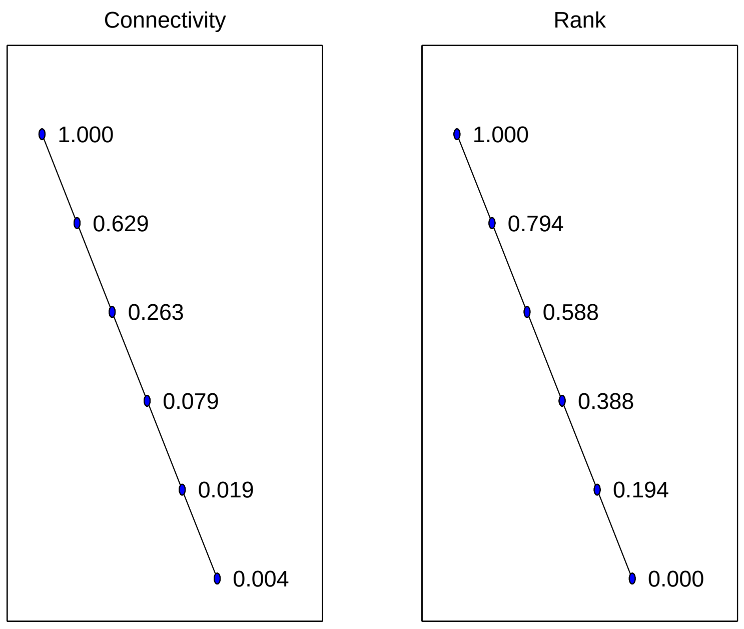


Figure 4.28: Connectivity and ranking for a six node linear network, with $\beta = 1 \times 10^{-3}$, $\eta = 1 \times 10^{-4}$ and $p_L = 0.99$. (a) shows each node's connectivity, with the top left node being designated as the basestation and hence has a connectivity of 1. (b) shows each node's rank, as calculated by agent ASH.

Effect of altering link probability Consider now the effect of altering the link probability on the connectivity, leaving all other parameters the same as before. The link probability for this simulation is set to be equal to 0.5 (i.e. only half the transmissions undertaken are successfully received). The results from the simulation (conducted with the same random seed for consistency) are shown in Fig. 4.29.

The simulation shows that altering the link probability has a large impact on the spread of the connectivity values, with the node closest to the basestation dropping its connec-

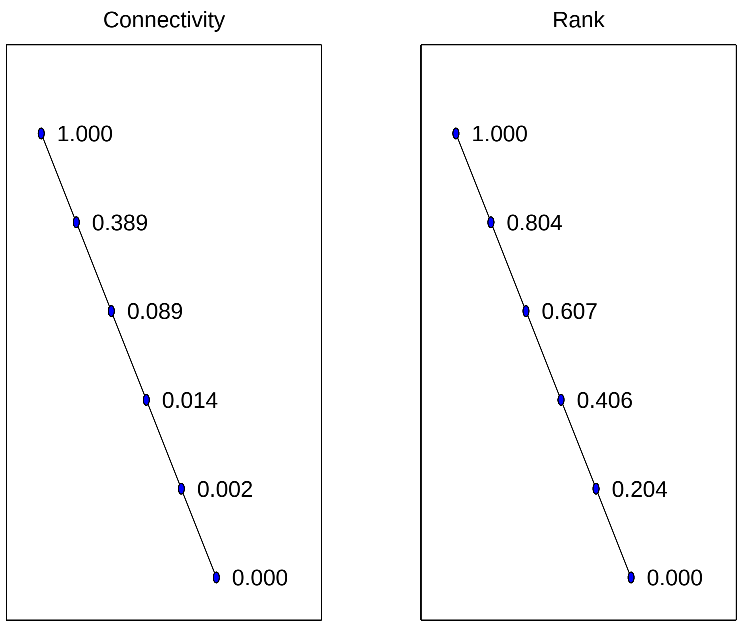


Figure 4.29: Effect of altering link probability on connectivity and rank, with $\beta = 1 \times 10^{-3}$, $\eta = 1 \times 10^{-4}$ and $p_L = 0.50$. (a) shows each node's connectivity. (b) shows each node's rank, as calculated by agent ASH.

tivity metric from 0.629 to 0.389. However, the topology of the network has not changed, and this is reflected in the ASH rankings, which demonstrate that the nodes have a similar (but not the same) ranking as previously. The reason the rankings are not identical is because fewer transmissions were received to update the ASH ranking model. However, they are still evenly spread over the unit domain.

Effect of altering connectivity parameters Consider now the effect of altering a parameter that determines connectivity, leaving all other parameters the same as the first simulation. The adoption rate is increased from $\beta = 1 \times 10^{-3}$ to $\beta = 6 \times 10^{-3}$. The results from the simulation (conducted with the same random seed for consistency) are shown in Fig. 4.30.

Increasing the tuning factor β has resulted in markedly different connectivity values, with the smallest connectivity metric (0.715) being larger than the best non-base station connectivity in the first simulation which had a connectivity metric of 0.629. However, the ranks of the nodes are exactly the same as in the first simulation, highlighting the invariance of ASH to the absolute value or spread of the attribute.

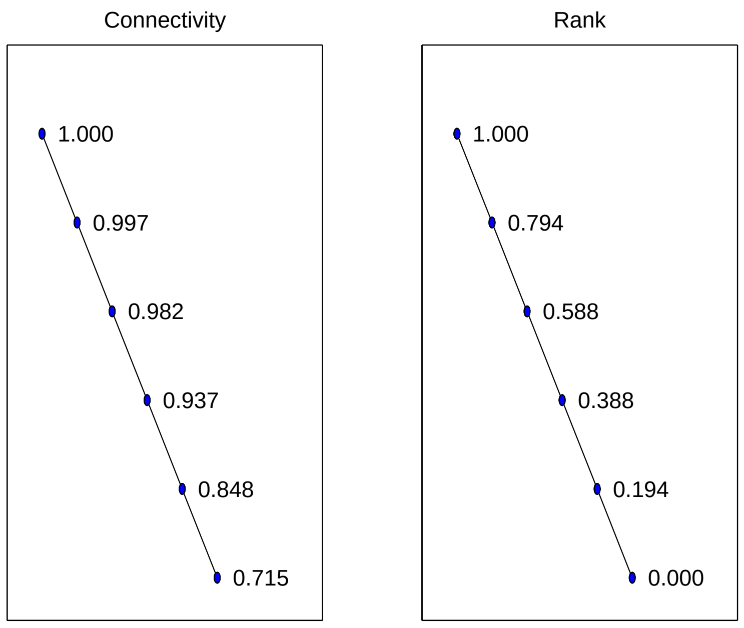


Figure 4.30: Effect of altering connectivity parameters on connectivity and rank, with $\beta = 6 \times 10^{-3}$, $\eta = 1 \times 10^{-4}$ and $p_L = 0.99$. (a) shows each node's connectivity. (b) shows each node's rank, as calculated by agent ASH.

Discussion

These simple simulations demonstrate the sensitivity of the connectivity metric to both network specific parameters and tuning parameters. ASH was able to correctly rank the nodes in all the systems, reducing the dependence on precisely chosen parameters. This makes network protocol design simpler, as deployment specific characteristics do not have to be built into the network protocol. With reference to the utility based routing schemes, it can be seen that the choice of a suitable threshold is application dependent and specified prior to deployment. However, if ASH is used as an underlay to scale the resource to the unit domain, the choice of the threshold becomes much simpler, as the spread of the resource has no effect on the spread of the rank. In addition, if the spread of connectivity alters due to changing network conditions, ASH dynamically adjusts the ranking to maintain it on the unit domain.

ASH can be used to rank metrics other than connectivity. For example, in sCAR, remaining battery power is expressed as a proportion of the initial value, with 1 corresponding to a full battery and 0 corresponding to an empty battery. It can be seen that sCAR is unable to handle heterogeneity in the initial battery capacity, as the node's energy is simply expressed as the proportion of remaining battery energy. ASH however can be used to rank the absolute amount of energy in each node's battery, resulting in the node with the largest battery reserve having a rank of 1 and the one with the smallest energy reserve having a rank of 0, regardless of the size of the batteries.

Thus, it can be seen that ASH can be used to enhance the performance of existing routing protocols, by considering the relative ordering of attributes, rather than their absolute value. ASH rankings are invariant to the spread of attributes, which is beneficial to networks where the spread is not known *a priori* or dynamically varies. ASH is lightweight and thus can easily be incorporated into existing routing protocols as a network underlay.

4.9.2 A simple cross layer protocol

In this example, it is demonstrated how network information can be used at all levels of the traditional network stack, collapsing the strict segmentation, leading to a simpler implementation. Both energy and connectivity are ranked separately and the ranked data is used throughout all the levels of the network stack.

Medium Access

Nodes with low ranks both in connectivity and energy are not active in network tasks such as routing and replication. They essentially act as leaf nodes, only injecting packets into the network. As a leaf node is not required to route other node's packets, there is no cause for it to ever attempt to listen for other node's data transmissions (it does still need to observe 'Hello' packets in order to maintain its correct rank). In addition, due to its scarce resources, it should not have to compete equally with higher ranked nodes for access to the medium. Furthermore, as packets can only ascend the hierarchy, there is no point in a low ranked node listening to a high ranked node's transmission. A simple slotted MAC scheme is presented to demonstrate how ranking can result in a sensible preferential access to the shared medium. This is shown in Fig. 4.31 which shows the behaviour of each node in its assigned slot. This is only one possible arrangement of slots. A more realistic approach might be to have wider or more slots for higher ranked nodes as they spend more time on the medium as they will have a greater amount of data to send. The scheme does not have to be slotted (relaxing the requirement of synchronization), but can also be made into a random access protocol, where the probability of listening to the medium is based on the node's level.

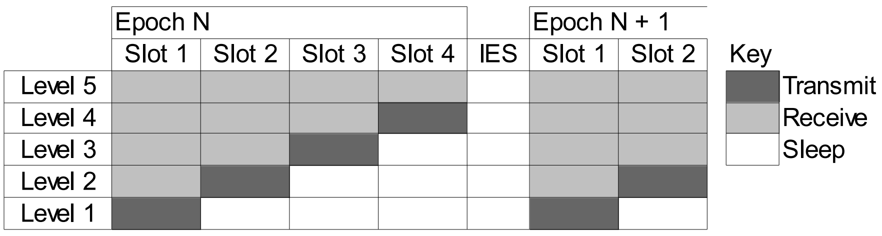


Figure 4.31: Slotted MAC organisation example for a 5 level network. Based on their level, nodes choose what action to take in each slot. Note that the lowest level nodes spend the majority of their time in low power sleep mode.

Routing and Replication

One possible method to control data delivery in a network ranked by ASH is presented. Low ranked nodes perform direct delivery to higher ranked nodes. High ranked nodes share information amongst themselves, epidemic style [268]. Thus, replication and delivery is controlled according to rank. Clearly, duplicating a message across low level nodes does not achieve any useful redundancy, as these nodes are not active in disseminating information and are likely to be severely resource constrained compared to their higher ranked peers. Thus, messages are replicated with a probability that increases with node level. Hence, delivery amongst low level nodes will resemble direct routing (with low traffic overhead, but high latency) and delivery between high level nodes will resemble epidemic routing (with high traffic overhead and low latency) [233].

To accomplish the objectives of rank based routing, let the probability of a level k node sending a message to a level j node can be given by

$$p_{send} = \begin{array}{c} 1 : j > k \\ p_t : j = k \\ 0 : j < k \end{array} \tag{4.22}$$

where p_t is the horizontal (that is, across the same levels in the hierarchy) transmission probability. Once the message has been sent to a higher level node, the node can either keep the message or delete it. Let the probability of a level k node keeping a sent message for future replication be given as

$$p_{replicate} = r^{L-k+1} \tag{4.23}$$

where r is a parameter that controls the degree of the replication. This equation results in a probability of keeping a message for future replication that increases with level in the hierarchy. The value of r affects the expected number of copies present in the network. Thus, low ranked nodes will not keep their packets for replication, whereas replication becomes more common as the hierarchy is ascended. Thus, low capability nodes are not unfairly burdened with large amounts of traffic.

To prevent thrashing due to rank promotion and demotion as a result of attribute changes, packets are not allowed to descend in rankings, even if the host's ranking has dropped. This prevents issues with packets forming loops in the hierarchy, leading to rapid network exhaustion.

Application

Information about each node's local variables, such as energy and connectivity can also be used at the application layer. A node with a low ranking in energy is likely to expire sooner than once with a high energy ranking. Thus, based on the energy ranking, the application running in the node can shut down or throttle back energy consuming tasks, as described in dynamic role altering (refer to Section 3.6). For example in a GPS based system, the GPS receiver consumes a large amount of power. To conserve energy, the sampling rate of the system can be reduced, leading to a greater node lifetime at the cost of location resolution. The application can also use lossy compression algorithms to reduce the volume of data that needs to be sent over the radio medium. The amount of compression can be controlled by the rank in the hierarchy, so that low level nodes use high compression factors, thus minimising the amount of data that they need to send.

4.10 Related Work

The primary contribution of this chapter is the presentation of methods that can be used to rank heterogeneous attributes in a network into a relative framework. As such it is complementary to many routing protocols as it can act as a network underlay, enhancing their performance. The idea of using hierarchies in networks is not a new one, but to the best of our knowledge this is the first work that has considered the general situation of how to dynamically cast any measurable resource into a network wide ranking system, by using a linear dominance hierarchy.

The work that is most closely related to that presented in this chapter is the role assignment algorithms of Römer et al [270], [271]. In this scheme, nodes decide on their role within the network based on information acquired from their local neighbourhood, by populating a cache of node properties and running node assignment rules based on their cache. Their algorithms were explicitly designed for stationary networks, as a change in the neighbourhood would result in a cache update. They introduced high level compiler directives that are used to decide on the most suitable role for a node (such as cluster-head or ordinary node), whilst satisfying requirements such as coverage. Their algorithms require the specification of time-out factors and explicit update methods - ASH is lightweight in comparison to the role assignment algorithms as it piggybacks onto existing network control packets providing a transparent evaluation of local role. To control flooding, the role assignment algorithms use a limited hop neighbourhood, whereas ASH discovers network wide information in order to assess rank.

The ranking of nodes can be viewed as sorting them into order according to their attributes. A scaled domain is used though to represent the maximum value of the attribute as 1 and the minimum as 0, so this method is not entirely a simple sorting procedure. Some recent work has been performed on the theory of random sorting networks [272]. Random sorting networks are networks for sorting information that are created at random. This is similar to our pairwise ASH scheme presented in Section 4.4 which can also be viewed as a randomized version of a Bubble Sort.

4.11 Conclusions and future work

4.11.1 Future directions

This chapter has discussed methods of forming a social hierarchy amongst nodes in various situations. However, there is scope for further exploration into some of the areas which have yet to be addressed. One such avenue is in the use of an optimal estimator of node rank, for example by using Bayesian methods. Essentially, the problem is that of assessing the most likely rank for a node, given a prior set of information. The methods used in this work are computationally simple, and it remains to be seen whether more optimal methods of determining rank would be implementable on low cost devices.

All the methods presented in this chapter form a linear dominance hierarchy. In this type of hierarchy, only ordinality is relevant, not the degree of difference between nodes. In a heavily resource partitioned network, there will be nodes with a large attribute value and nodes with low attribute values, but nothing in between. Such would be the case with small battery powered sensors connected into a mains network, with no intermediate nodes in terms of resource size. In this case, two nodes which are close to each other in rank value could have large difference in resource value. This could place an unfair load on a falsely ranked node. This suggests that a non-linear hierarchy should be formed in this instance. This could be done by nodes exchanging a histogram reflecting the distribution of resource values across the network.

In Section 4.6 it was discussed how to use simple random agents to disseminate rank/attribute information across the network, in order to recreate the effect of randomized meetings.

The agents used are extremely simple and essentially stateless. There is a large body of work on agent based (also known as ant inspired) algorithms for exploring networks and routing data. A very simple agent based approach has been used in this chapter. Far more sophisticated ant based routing and discovery methods have been presented in the literature which use stateful agents and pheremone trails [273], [274], [275]. However, the intention of this work was to keep ASH formation and maintenance as simple and lightweight as possible.

Another area which needs to be explored is how to factor in the rate of change of rank, both into the rank determination process and also the routing protocol. Thus nodes can be ranked according to their rank reliability or stability. Nodes would thus avoid using intermediates which display large rank variance. Some of the ideas presented in CAR [265], in particular the prediction process, could be applied to the ranking methods to result in a more stable and useful system.

The mobility patterns used in this simulation (with the exception of the stationary scenario) were random based. An area of future research is to analyze the performance of the various methods when subjected to different mobility models, such as group or cluster based. In addition it would be interesting to evaluate the performance of the system on real or synthetic animal trace data.

4.11.2 Conclusion

A novel biologically inspired method of forming a hierarchy based on differences between individual nodes has been presented. This hierarchy adapts to changes in node resources and provides the ability for nodes to determine their role in the network relative to their peers. Three different approaches to forming the hierarchy have been presented. The first method assumes that nodes engage in pairwise meetings and from this sort or exchange their ranks according to the relative order of their attributes. It was shown that this leads to poor adaption to node insertion and removal, and thus the protocol was refined by introducing a reinforcement factor which spreads the ranks out evenly across the ranking space.

In the next method, the assumption that nodes undertake pairwise exchange of information was removed, as this imposes constraints on the MAC layer. Instead, it was investigated how a single transmitter can broadcast its rank/attribute information to a number of receiving peers. Based on this newly acquired information, they update their ranks using a win/loss ratio method, similar to that used by researchers in the biological literature to measure dominance. This was found to have slow convergence and adaption to changes. To remedy this, the switching idea from pairwise ASH was incorporated to result in rapid adaption to resource variation.

Both of the methods are designed with mobile networks in mind. In stationary networks, due to a limited discovery horizon, the ranks do not converge to their correct values. This led to the introduction of a third method, agent ASH, which replicates the effect of randomized meetings by spawning random agents which carry rank/attribute information to non-neighbouring nodes, resulting in correct ASH convergence. In agent ASH, rank is estimated by estimating the rank of the node's nearest neighbours in the ordered list.

The effect of real world mobility was shown to result in more rapid convergence. Agent ASH works well for both purely stationary and mixed mobility networks, whereas the other two methods work best in mobile networks. To avoid the formation of loops, levels were introduced, along with hysteresis to ensure that messages only permeate upwards through the hierarchy. It was shown that levels lead to scalable and predictable traffic density, with the density at a level being independent of the number of nodes in the network.

Lastly some possible applications for the ASH approach were examined, showing how it can be used as a network underlay to enhance the performance of existing protocols by providing resource abstraction and also as a powerful cross layer management and routing protocol.

In summary, ASH provides a framework for nodes to discover their role within diverse networks, by allowing resource abstraction. This leads to simpler routing and management protocols, that are removed from the imposition of absolute values. The methods presented here allow nodes to discover their rank relative to their peers, when provided with zero initial information about the spread of resources in the network.

5 Uniform distance sampling strategy

5.1 Introduction

The primary purpose of a wireless network is to relay data from sensors to the end-user. This has led to a large amount of work performed on optimizing the process of transferring the data through the network, with special focus on energy conservation [237]. However, it also possible to save energy or generate more useful information by altering the way sampling is undertaken.

GPS receivers are power hungry, consuming in the order of 90mW upwards when acquiring a location fix [28, 276]. In addition, they take time to acquire a fix. The time taken to obtain a location estimate depends on a variety of factors such as when the last fix was acquired and the sky view. However, it typically takes between 10 and 30 seconds, but can take minutes or even fail [29]. Clearly, on an energy constrained device, the power consumption of the GPS receiver is significant and thus an attempt should be made to maximize the amount of useful information obtained from it.

To extend the lifetime of the tracking device, existing devices attempt to acquire GPS position fixes at regular intervals in time [5, 7, 6]. Whilst the GPS receiver is not actively obtaining a fix, it is placed in a low power sleep mode. When it is triggered, it wakes up and attempts to obtain a fix. Thus, by reducing the proportion of time that the receiver is in high power active mode relative to sleep mode, the overall power consumption can be substantially reduced, prolonging the longevity of the device. The duty cycle is chosen such that sufficient location detail is captured whilst satisfying lifetime requirements. Typical values for wildlife tracking collars are from one fix every 15 minutes to one fix every few hours, although higher sampling rates are possible for brief studies [5, 7, 6].

The conventional method of scheduling fixes at constant time intervals has two serious shortcomings. The first problem associated with this method is that by reducing the duty cycle, the path taken by the host creature (which can be human or animal) is often undersampled. Thus, for high speed forays, fine grained detail can be lost, leading to under-estimation of the distance travelled [277]. High speed information is of particular interest to wildlife researchers, as this behaviour is often correlated with predation activity [278]. As previously stated, the sampling frequency cannot be increased without detrimentally impacting the lifetime of the device.

The second problem is that unnecessary fixes are taken whilst the host creature is stationary. Animals (especially carnivores) spend a large proportion of time asleep [30]. Humans also have periods of activity and inactivity. Thus, it is wasteful of valuable energy to take redundant fixes when the tag carrier has not moved since the last fix, as no additional spatial information is provided.

A switch from uniform time based sampling to uniform distance based sampling (sometimes called Lebesgue, multi-rate or event based sampling [279]) is motivated, in order to address these two issues. Thus, the sampling frequency alters in relation to the speed of the host[1], simultaneously solving both the previously stated problems associated with uniform time sampling.

In this chapter, a system is presented that:

- uses a combination of GPS and accelerometer sensors to estimate the distance travelled by the host.
- is able to learn and adapt to the locomotion patterns of the host.
- attempts GPS fixes at approximately uniform distance intervals.
- is implementable on low cost, memory constrained microcontrollers using computationally simple algorithms.
- is able to reduce the power consumption of the tracking device, whilst still generating accurate motion tracks.
- generates detailed speed-time profiles using analysis of the acceleration signal acquired by the low power accelerometer, without requiring the high power GPS receiver to be powered up.

A conceptual flow diagram of the components of the system is shown in Fig. 5.1. The low power acceleration sensor is used to generate an estimate of the speed of the host through gait analysis. Acceleration waveforms are statistically analysed to generate a

[1]If the motion of the host creature was predictable and repetitive, then uniform distance sampling could be achieved by learning how far it travels at various times of day and scheduling subsequent fixes based on the prior activity patterns. Whilst simple, this approach would miss unusual events, which are often those of interest.

'signature' of the animal's behaviour. This signature is used to relate the acceleration waveforms to the ground speed of the animal, by using an adaptive model. It should be noted that the energy consumed by the process of estimating the speed from the accelerometer snapshot is a fraction of that consumed by the GPS unit. If the cumulative estimated distance travelled exceeds a threshold, the GPS unit is triggered to take a location fix. Whilst the GPS receiver is active, if the model's estimate of the host's speed is in error with respect to the GPS measured speed, the model is updated (trained) to incorporate this new information. The GPS receiver is then put back into low power sleep mode, awaiting a further trigger.

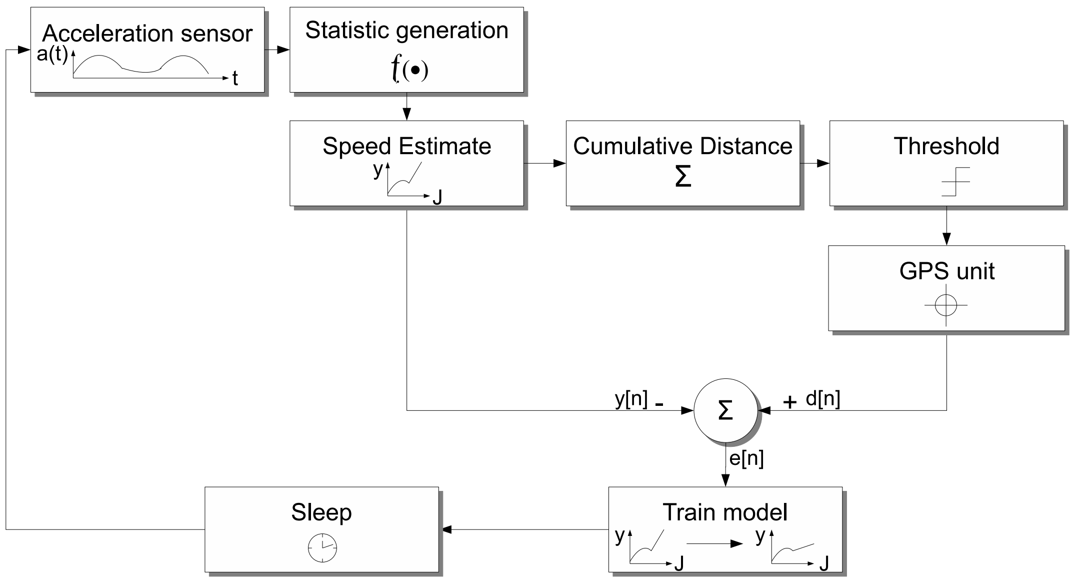


Figure 5.1: Conceptual system overview of the uniform distance sampling system.

In this chapter it is shown how a uniform distance sampling strategy can be achieved. First, how people and animals move (generating acceleration profiles) is discussed, and the relationship between statistical measures of the host's acceleration and the speed of the host is examined in Section 5.2. Next, in Section 5.3, a method to train a model which can be used to predict the speed of the host, based on the acquired acceleration signatures is presented. Results from real world tests are shown in Section 5.4. Section 5.5 examines a number of methods that can be used to intelligently schedule GPS fixes based on the estimated distance travelled. The increase in device lifetime that can be obtained by using this method is shown in Section 5.6, followed by examples of path reconstruction in Section 5.7. Existing work is contrasted in Section 5.8. Lastly, in Section 5.9 conclusions are drawn and the contributions of the work stated.

5.2 Statistics from gait

To trigger the GPS unit at uniform distance intervals, a means of estimating how far the host creature has travelled (without using the high power GPS unit) is required. A simple method is to affix a pedometer to one of the host creature's limbs and count the number of steps taken. Simple pedometers use a weighted switch to determine foot-fall, but more advanced devices often contain a low power MEMS (micro-electro-mechanical systems) accelerometer which measures acceleration directly and with the aid of a microprocessor estimates the length of each step [280]. A large body of work exists on pedestrian dead reckoning systems (PDR), which are used to provide an estimate of a human's position without using a GPS receiver or when there is no GPS reception (such as in a building or dense forest) [280, 281, 282, 283]. These systems can be regarded as simplified inertial navigation systems, as they estimate distance taken in each stride using an accelerometer and the heading (generally measured with a digital compass) to reconstruct the path of the user [282]. Shoe/foot attachment is a common method of device placement in PDR [280, 281], but this is an unsuitable location for attachment to wild animals due to robustness concerns.

Legged animals (either bipedal or quadrupedal) effect overall body locomotion by moving their limbs in a certain sequence. Depending on the desired speed of motion, different sequences are used – these are referred to as gaits [53]. Bipedal animals typically have two gaits, walking or running. Some bipedal mammals such as macropodid marsupials also hop [284]. However, the majority of animals are quadrupedal. Quadrupedal animals have a much richer choice of gaits, due to the greater number of possible limb-sequence combinations. Typical gaits are walking, trotting and galloping [53].

It has been hypothesized and experimentally validated that animals change from one gait to another gait at certain speeds in order to optimally utilise energy [285]. It has also been shown that stride[2] frequency changes linearly with speed within a particular gait [286], [287]. For animals larger than 1 kg, maximum stride frequency is typically less than 6 Hz [288] and for large animals (such as a horse), maximum stride frequency is less than 3 Hz [286, 287]. However, for gaits resulting in a greater speed (such as galloping) the slope of the stride frequency with speed becomes smaller, due to the animal increasing its stride length [287]. This is shown graphically in Fig. 5.2 which demonstrates the variation in stride frequency with speed for different animals. It can be seen that there is a non-linear relationship between stride frequency and speed, and that different animals have vastly different speed–frequency relationships.

These observations have been made on foot motion, but a tracking collar is typically

[2]A stride refers to measurements made on one foot/limb only

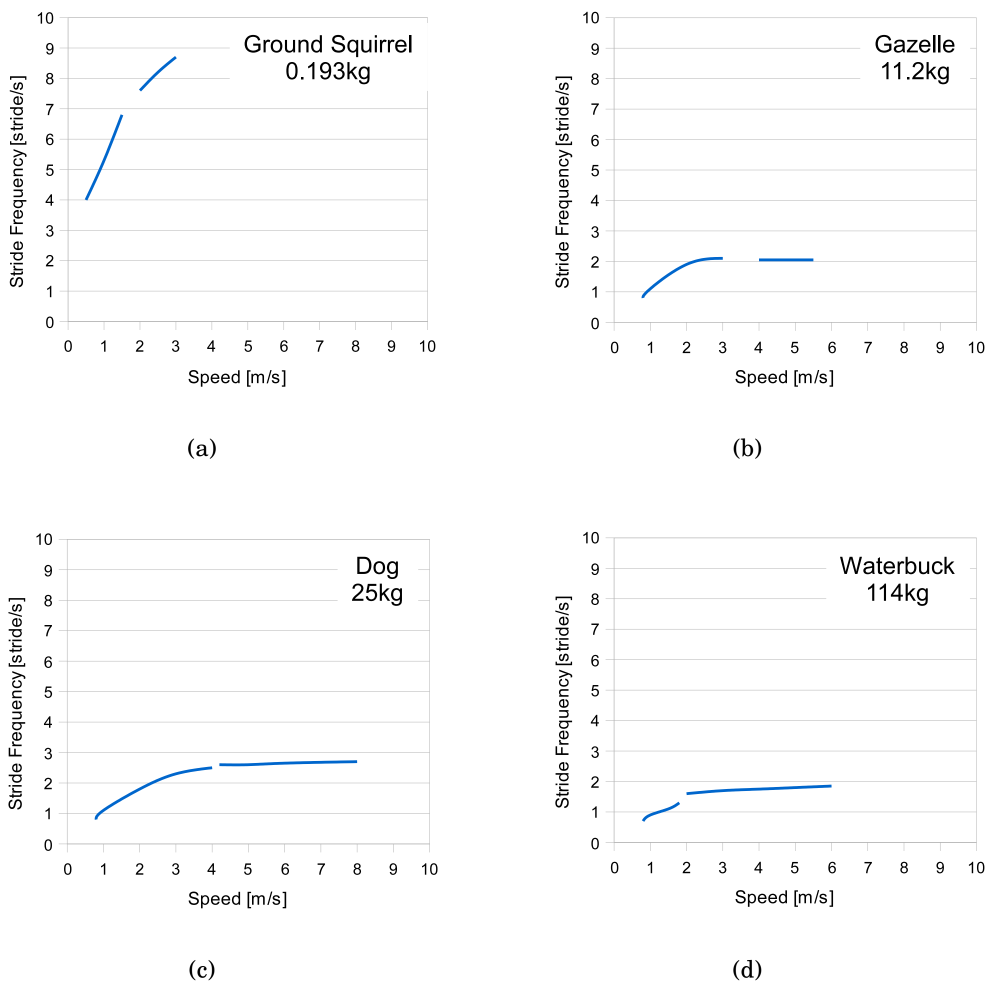


Figure 5.2: Variation in stride frequency with speed for different animals. There are two lines on each graph, as these correspond to the two distinct gaits of walking (lower speed) and galloping (higher speeds). These graphs are adapted from [287]

placed on the neck of the animal. However, the neck of the animal does not act in isolation from the limbs and thus there will be some coupling between the activity of the feet and the motion of the neck. From the above review, it would be expected that the frequency of acceleration of the neck would increase with speed. Furthermore, from basic physics, it would also be expected that the magnitude of the acceleration would also increase with speed due to an increase in the vertical ground reaction force. These observations guide the choice of suitable statistics – both frequency and amplitude related measures are likely to be useful in determining overall host speed.

To examine whether a neck mounted (on an animal) accelerometer could be used to estimate host speed, an acceleration logger was designed and constructed by the author.

This consisted of an u-blox LEA-4P GPS module [289] used to measure the animal's speed, an ADXL103 uniaxial accelerometer [290] and a PIC18LF4620 [291] microcontroller, which controls the operation of the system. The prototype logger mounted on a standard dog collar is shown in Fig. 5.3.

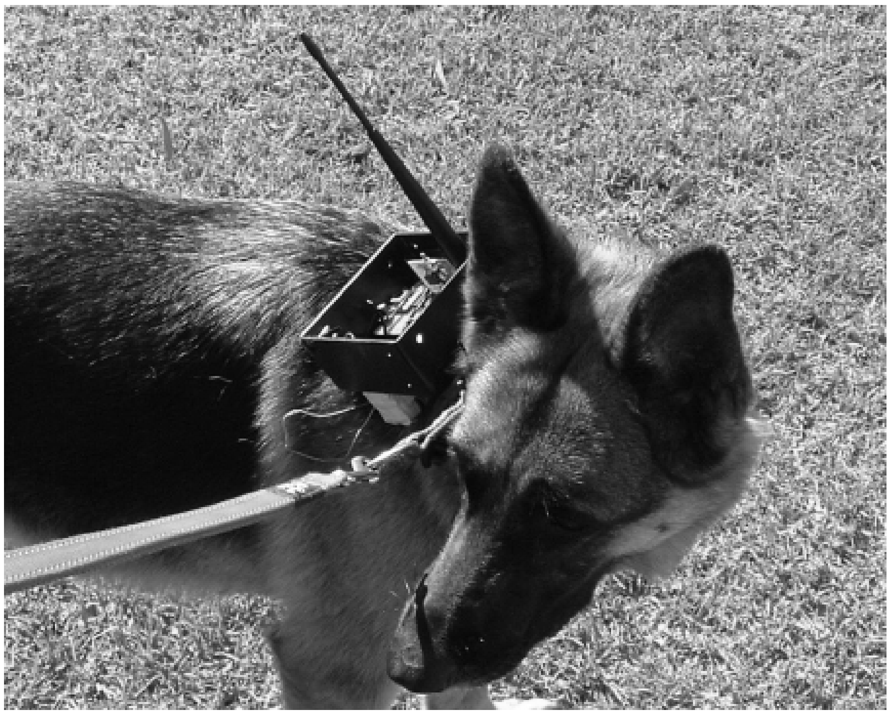


Figure 5.3: Accelerometer logger (early prototype unit) attached to the collar of a dog. Also visible is the antenna used to form the wireless network.

Acceleration snapshots were recorded from the dog shown in Fig. 5.3 for a variety of gaits, whilst being led on a lead. The vertical (perpendicular to the animal's neck) component of the acceleration was captured. Acceleration snapshots were also obtained from the same device when it was carried in the left hand of the author.

For this system, what is required is a simple energy efficient method of periodically estimating the speed of the host in order to intelligently schedule the times at which GPS fixes are attempted that can be implemented on a low cost microcontroller. Thus, rather than continual monitoring of the accelerometer as in pedestrian dead reckoning (PDR), brief snapshots of the collar acceleration are taken. It was found that snapshots between one and two seconds in length provided a good compromise between capturing sufficient detail and expending excessive amounts of energy in their acquisition. Hence, the length of the snapshot was chosen to be 1.2 s. Typical acceleration profiles are shown in Fig. 5.4, obtained with a 32 Hz sampling rate whilst the subjects were running at 3.5 m/s (as measured by the GPS unit). For comparison, acceleration snapshots for the subjects walking at 1.5 m/s under the same conditions are shown in Fig. 5.5.

It should be noted that the accelerometer does not need to be calibrated and the units shown are the results of the 10 bit ADC conversion on the microcontroller. However, for interest, it was found that the peak–to–peak excursion of the acceleration for a dog whilst running was in the range of $2.2 - 2.6\,g$ and for a human running was in the range $1.8 - 2.1\,g$.

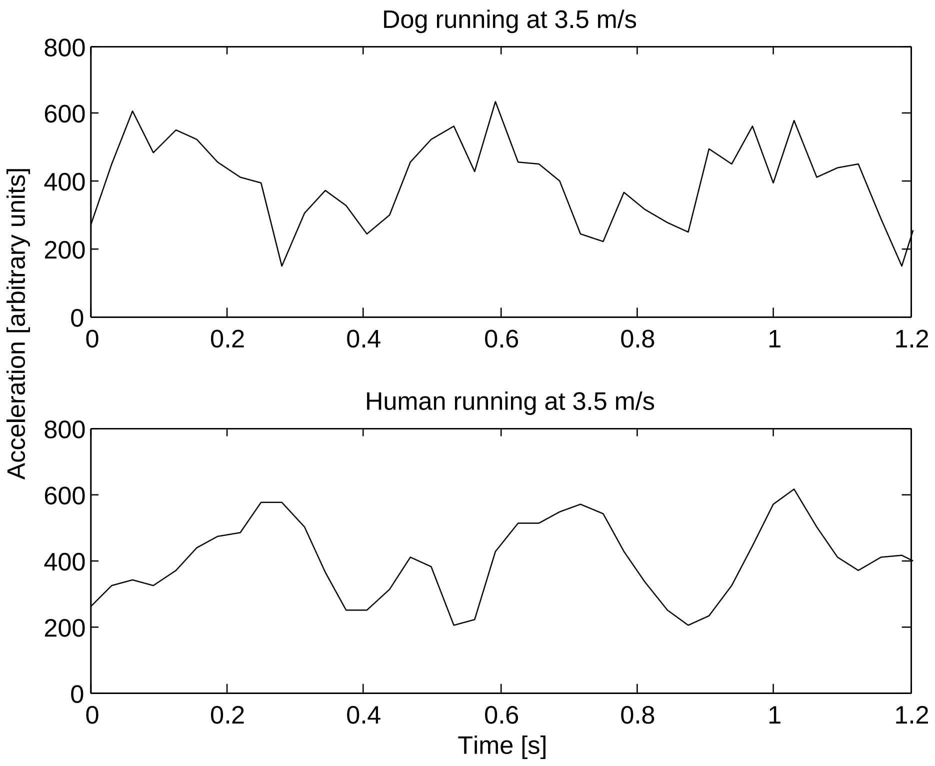


Figure 5.4: Accelerometer snapshots. Top figure: Dog running at 3.5 [m/s]. Bottom figure: Human running at 3.5 [m/s].

These snapshots show that there are large differences (as would be expected) between a bipedal and quadrupedal animal in the shape of the acceleration waveform. However, from inspection of multiple datasets, it was found that there exists an underlying common relationship where the peak excursion of the acceleration increases with speed. This is due to a greater ground reaction force [292]. In addition, due to the bio-mechanics of animal locomotion, there is a correlation between the frequency of these peaks and the speed, as a higher speed is generally associated with a more rapid movement of the limbs [288, 285, 286, 293]. Before the calculation of statistical measures representative of the motion of the host are discussed, an issue that is present in the lag of the speed output of GPS receivers is highlighted.

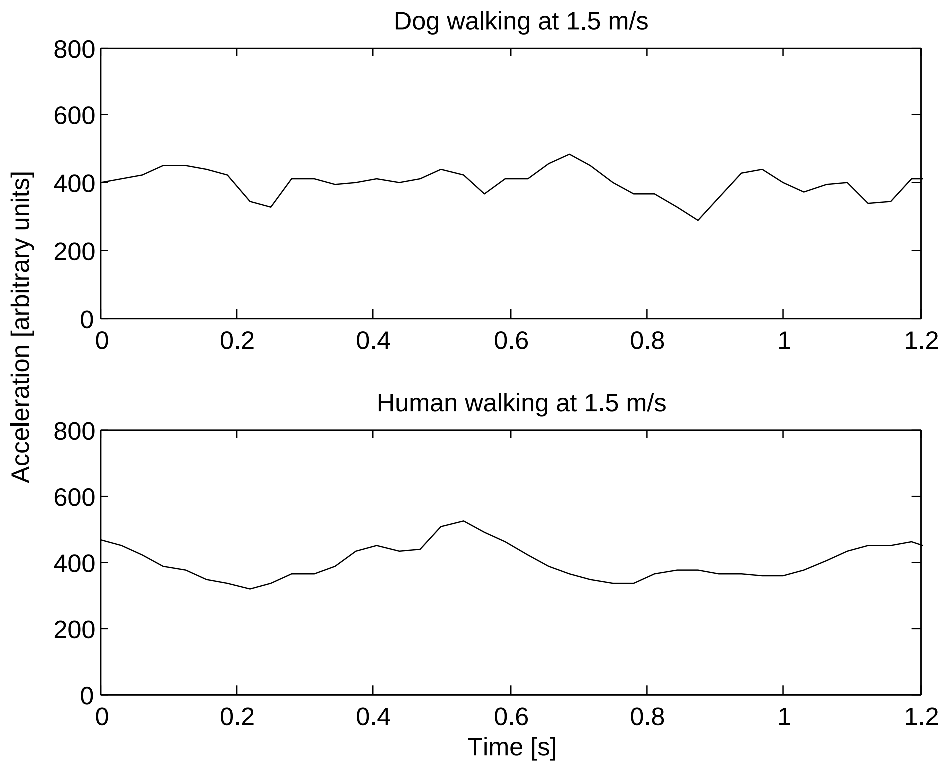


Figure 5.5: Accelerometer snapshots. Top figure: Dog walking at 1.5 [m/s]. Bottom figure: Human walking at 1.5 [m/s].

5.2.1 GPS Outlier Filter

GPS receivers provide both a location and a speed estimate [122]. By using the speed estimate from the GPS receiver, one does not have to compute the speed by taking the time-derivative of position. However, it was found in practice that the reported GPS speed sometimes was inaccurate with respect to the observed speed, especially with rapid changes in speed, such as when a creature starts running from a standstill. This is due to the effect of a Kalman filter present in the GPS receiver which smooths the GPS speed to reduce the effect of noise, but results in a lag between the actual speed and the reported speed [122]. Depending on the dynamic model selected, the lag will have different characteristics. For example, if the GPS receiver is set to pedestrian mode, the lag is larger than if it is set to aircraft mode, which rapidly responds to changes in motion [122]. However, in aircraft mode, the speed reported by the GPS receiver was found to be virtually unusable due to the large amount of noise. Thus, for this system, it was found that setting the GPS receiver to use the pedestrian dynamic model resulted in the best reported speed data, but had a relatively large associated lag.

This associates an incorrect speed measurement with an acceleration snapshot which corrupts the dataset. Thus, an outlier filter was used which discarded acceleration–

speed snapshot pairs when the speed as reported by the GPS unit dramatically altered, effectively culling the outliers in the dataset. The filter was used on a sequence of M GPS speed values. From these values, the standard deviation was calculated (even though the standard deviation itself would be distorted by the presence of outliers, this method was found to be acceptable) and the speed-acceleration snapshot pairs were discarded from further analysis if the speed was more than a factor of β from the mean. In practice, M was chosen to be 5 samples (corresponding to a short window whilst the GPS unit is active) and β was 1.3 (such that data more than 1.3 standard deviations from the mean is discarded). These values typically result in between 15% and 20% of the data being discarded, but this depends on how the host varies its speed.

As a test of the GPS filter, the prototype was fitted with a buzzer that sounded at random intervals. A human subject changed state as rapidly as possible from running to stationary and vice-versa when the buzzer was heard. The results from the test are shown in Fig. 5.6. The top plot is the state of the human host[3]. The middle plot shows the speed of the host as reported by the GPS receiver. Note the lag between the two plots, which will have the effect of associating an incorrect speed with a particular acceleration, corrupting the learning process. The bottom plot demonstrates the action of the cleaning filter, showing how rapid variations in speed are discarded from the dataset. The discarded points are marked with an ‘x’.

5.2.2 Statistical Measures

In order to determine the relationship between tag motion and host speed, summary statistics from the acceleration snapshots were calculated. There are a variety of possible statistical measures that could be used, but measures of low computational complexity were required, to allow easy implementation on the low cost microcontroller. Two statistical measures were found to provide a useful amount of information whilst being simple to implement.

The first statistic used was the *range*, which is simply the peak-to-peak excursion of the acceleration signal, within the 1.2 s sampling window. This is defined as

$$\texttt{Range} = max(a(t)) - min(a(t))$$

where a(t) is the acceleration signal acquired during the sampling window[4]. This metric is simple to compute and can be done on a sample-by-sample basis without the need to

[3]As a validation that the user was indeed in the correct state, the acceleration data was also examined and showed that the user could rapidly change state from stationary to running and vice-versa, and that the lag was due to the GPS receiver.

[4]To reduce the effect of knocks and bumps to the device, the acceleration signal is first filtered before the range is calculated, as shown in Algorithm 2

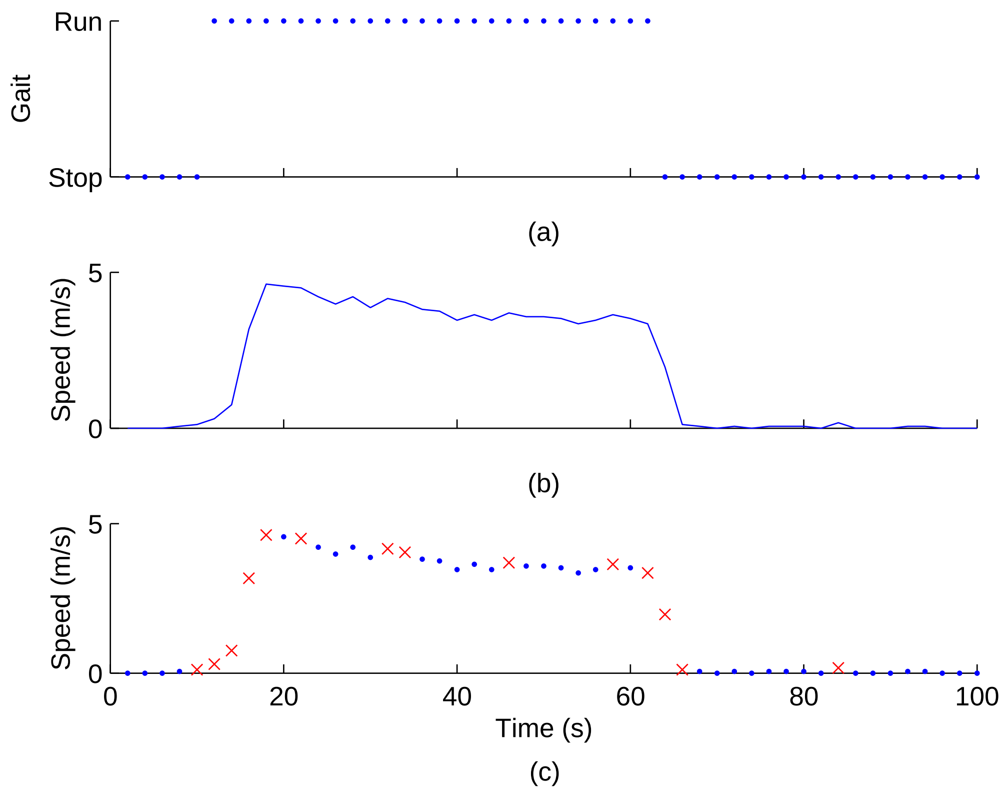


Figure 5.6: Test of the GPS outlier filter, demonstrating how it removes invalid data that corrupt the data-set. (a) The gait of the human host. (b) The speed as reported by the GPS receiver. Note the lag of the second plot relative to the first. (c) Cleaned data. Data that is discarded is represented by a cross, and data to be retained indicated by a dot. Observe how the outlier filter discards data-points with rapid speed variations.

store the entire dataset. This metric relates to the observation made previously that the vertical ground reaction force varies with speed. A scatter diagram of the range for both dog and human datasets (after being passed through the GPS outlier filter) is shown in Fig. 5.7. This scatter diagram demonstrates that there is a strong relationship between speed and range, as expected.

Statistics that measured frequency were also investigated. The Fast Fourier Transform and the Fast Walsh Transform [294] were unable to sufficiently capture the difference between related gaits (such as between running and walking). This is because the 3dB lobe width of the peak frequency component is approximately equal to $0.89/T$ where T is the length of the window in seconds. For a 1.2 s snapshot, the frequency resolution is 0.75 Hz. Longer snapshots (4 s in length) were recorded at the same sampling frequency, and it was found that for a human walking, the peak frequency was 1.6 Hz on average, whilst for running it was 2.3 Hz on average. The frequency resolution of the shorter

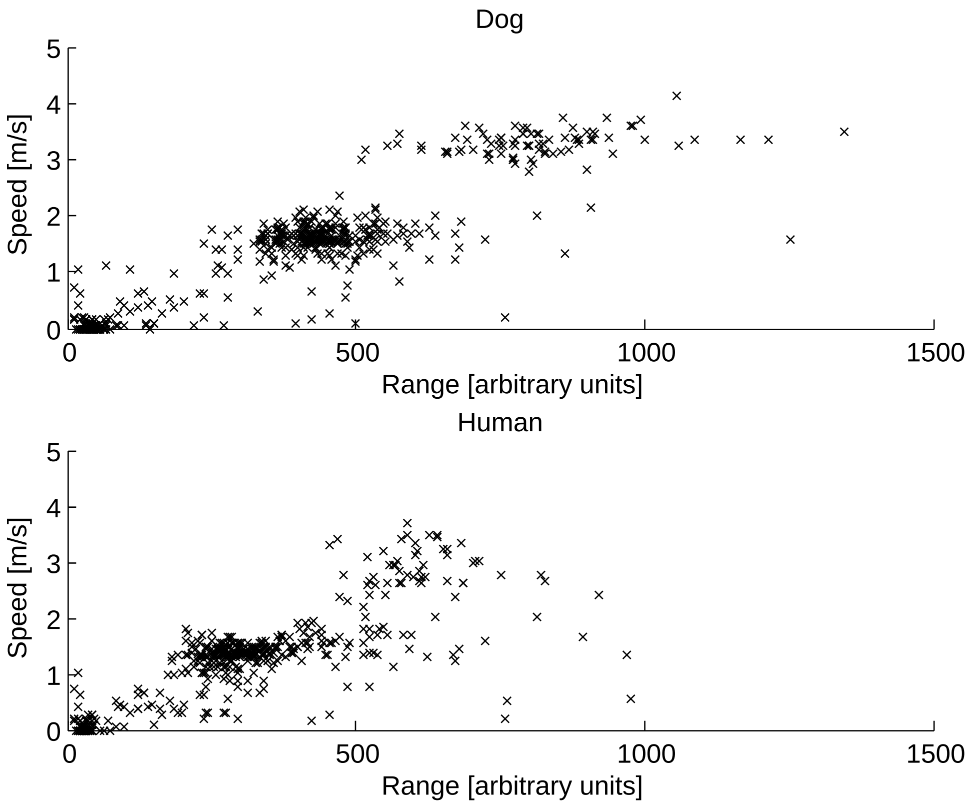


Figure 5.7: Scatter plot showing relationship between range of captured acceleration and speed of host as measured by the GPS unit

snapshot is thus insufficient to reliably discriminate between these two gaits. It was found that these methods could not provide sufficient frequency discrimination on the short snapshots with an acceptable computational burden.

A technique used in the PDR literature is to estimate stride frequency from the time domain acceleration signal, rather than using a mathematical transform [295]. Typically, the acceleration signal (in some cases the magnitude from three orthogonal axes) is filtered, reducing the effect of noise. The time between zero-crossings is used to determine the gait frequency and hence can be used to estimate the stride frequency. However, as the graphs in Fig. 5.2 demonstrate, for large animals the variation in stride frequency with speed is quite small (in the region of 0.1 Hz for a speed change of 1 m/s for a horse when galloping). Reliably determining the speed from the stride frequency is likely to be inaccurate due to these small variations, precluding the use of the direct stride frequency estimation techniques.

The second statistic used was the *variance of the jerk*, where the jerk is the time derivative of the acceleration signal. This method actually embeds the amplitude of the acceleration as well as the frequency into the metric, as will now be shown. Consider a sinusoidal[5] acceleration signal $a(t)$ with amplitude A and frequency ω [rad/s]

$$a(t) = Asin(\omega t) + a_0,$$

where a_0 is the static offset, due to gravitational effects.

The derivative of the acceleration is the jerk, $J(t)$, thus

$$J(t) = \frac{da(t)}{dt} = A\omega cos(\omega t).$$

Note that taking the derivative removes the static offset[6]. Taking the variance is the same as calculating the mean square value, where the mean of a sinusoid is zero. The jerk variance is

$$\overline{(J(t) - \bar{J}(t))^2} = \overline{J(t)^2} = A^2\omega^2 \overline{\cos^2(\omega t)} = 0.5A^2\omega^2. \tag{5.1}$$

Hence, this operation has the effect of amplifying the frequency through the action of taking the derivative of the acceleration. In addition, this calculation combines both frequency and amplitude into a single measure. Note that this method of calculating the variance of the jerk is computationally simple and can be computed on a sample-by-sample basis without needing to find the mean value. Graphs of the relationship between speed and jerk variance are shown in Fig. 5.8.

Figs. 5.7 and 5.8 demonstrate that there is a correspondence between the two chosen statistical measures and the speed of the host. Note that the data is clustered around certain speeds - this is because each speed is associated with a certain ambulatory gait such as walking or running. The data is relatively noisy, even with the action of the GPS outlier filter (which only acts on GPS data, and does not remove acceleration profiles on the basis of their 'noise'). This is possibly due to motion effects not correlated with gait, such as bumping the device whilst stationary. The scatter diagrams also show that the dog and the human have a different spread of summary statistics. In particular, the dog has a much larger jerk variance, due to the presence of higher frequency information in its acceleration profile, presumably as a result of having four legs. The complete algorithm for calculating the jerk variance and acceleration range is shown in Algorithm 2. The for-loop runs `Ns` times, where `Ns` is the number of samples taken during a snapshot. During each loop, the acceleration is passed through a low pass filter with update parameter α. The filter reduces the effect of noise and non-locomotion related artefacts such as bumps and knocks. Upon completion, the range is found in the variable `Range` and the

[5]A sinusoid is considered here as gait waveforms are predominantly sinusoidal in shape.

[6]This is beneficial, as the offset is due to non-motion related effects, such as gravity and accelerometer calibration. Thus, the action of taking the derivative eliminates the bias, without the need to calculate the mean and subtract it from the acceleration signal.

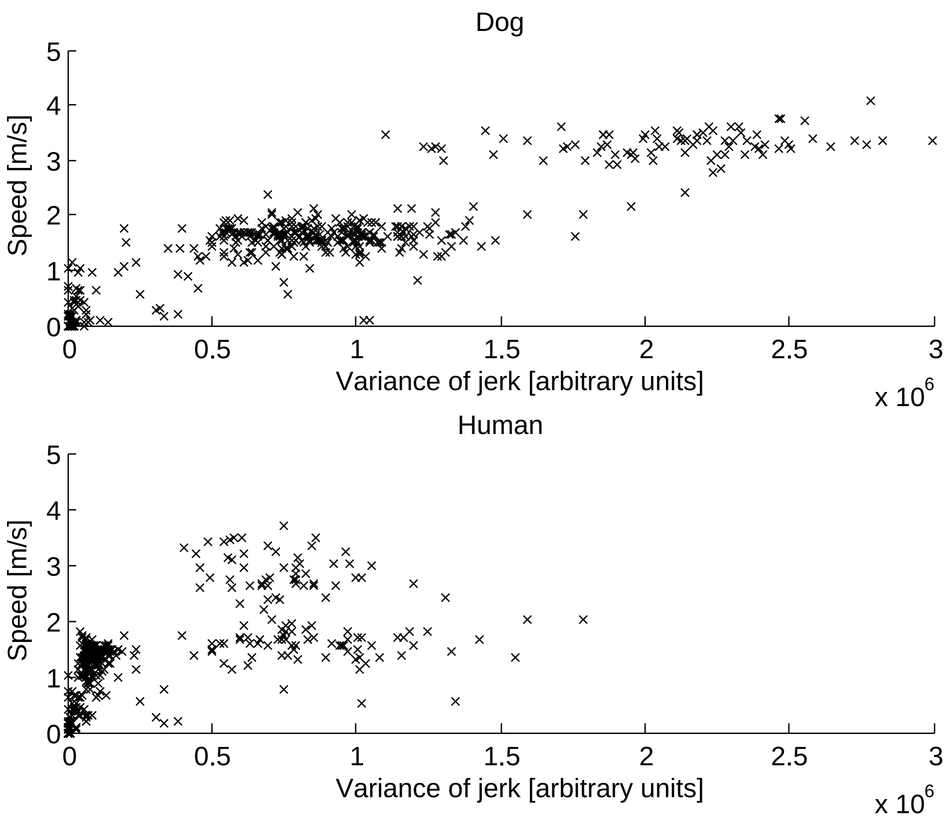


Figure 5.8: Scatter plot showing relationship between variance of jerk of captured acceleration and speed of host as measured by the GPS unit

jerk variance in the variable `Jsum`[7]. It can be seen that the algorithm is lightweight and computationally simple.

The following section discusses how to construct a mathematical model which adaptively learns how to predict the speed, given the summary statistics.

5.3 Model Training and Prediction

The analysis presented in Section 5.2 demonstrated that there was a relationship between the two chosen statistics and the speed of the host. However, this relationship needs to be determined, so that given a set of statistics, it is possible to estimate the speed of the animal[8].

[7]To further simplify the calculation, the variable `Jsum` is not divided by `Ns`, as would be required to calculate the variance. Thus the output in Jsum is a scaled version of the variance.

[8]Note however that this model assumes a terrestrially based animal. In the case of a climbing animal, for example a monkey, a much richer model would be required to correctly separate vertical from horizontal movement. This is suggested as a topic for further investigation.

Algorithm 2: Calculation of range and jerk variance

```
Jsum = 0;
a = GetAcceleration();
amax = amin = a;
for k = 1 : Ns do
    aold = a;
    a = GetAcceleration();
    a = $(1 - \alpha)$a + $\alpha$aold;
    if a > amax then
        amax = a;
    endif
    if a < amin then
        amin = a;
    endif
    Jsum = Jsum + (a − aold) $^2$;
endfor
Range = amax − amin;
```

One possible method would be to form a model off-line based on the captured data which would then be programmed into the collar. While this makes the implementation on the collar trivial (a simple lookup table would suffice), it would require that a model be built for each animal or species of animal depending on individual variability and required accuracy. This would be time-consuming and in some cases (such as a rarely encountered wild animal), logistically impossible. Furthermore, if the collar rotates around the neck of the animal to a different location, the acceleration signal (as captured by a uniaxial accelerometer) will alter, possibly leading to incorrect prediction of speed[9]. To some extent, collar rotation is prevented by the placement of the heavy battery pack below the animal's neck in wildlife tracking collars which acts as a counterweight [5, 3].

It would be desirable for the collar to build the model itself, and update it to changes in operating parameters. Thus, it needs to learn the relationship between the various statistics and the speed. There are many different learning algorithms and models that can be trained, varying in computational complexity, convergence speed and robustness. For this application, it is required that the algorithm be as computationally simple as possible, robust to noise and adaptable to new data. The least mean squares (LMS) algorithm [296] will be first discussed, with particular application to training a neural network to form a model. This will then be contrasted with the recursive least squares (RLS) algorithm [296], and differences between the two with regards to performance and suitability examined. First, however, an overview of learning algorithms in general is provided.

[9]This can be compensated for by using a triaxial accelerometer and calculating the magnitude of the acceleration from the three orthogonal axes [282].

Overview of learning algorithms

An adaptive model is a function which operates on its input(s) $x[n]$ to create an output signal $y[n]$ [296]. However, an adaptive model is initialised with little or no *a priori* knowledge of the relationship between the input data and output. Thus, the model must learn this relationship. To do so, it is given a reference signal, $d[n]$, representing the desired output given an input signal $x[n]$. The error $e[n]$ represents the difference between the model's predicted output and the desired output. A *learning algorithm* acts on this error to adjust the model in some way so as to minimise a measure of the error (such as the mean square of the error). This is referred to as the *training* process. Then, when the desired output $d[n]$ is unavailable, then the adaptive model can *predict* the actual output. This process is shown in Fig. 5.9, comprising the learning phase and the prediction stage. Note that the learning process is essentially a negative feedback control system, as it acts to reduce the error between the predicted and actual output values by altering parameters of the model.

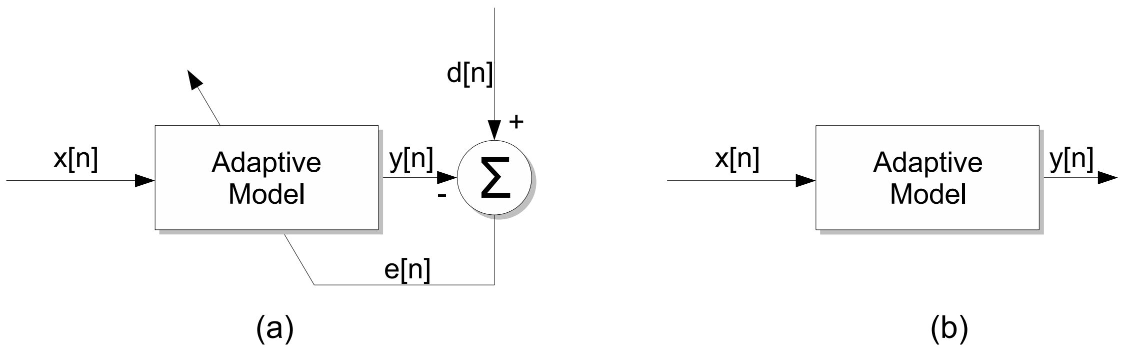


Figure 5.9: Learning algorithm structures: (a) The training phase to incorporate new data into the model; (b) The prediction or feedforward stage

Clearly two main choices need to be made: the form of the adaptive model and the type of learning algorithm. However, these two choices are not isolated, as the structure of the adaptive model will result in a certain subset of learning algorithms being applicable or more suitable. Essentially the adjustable parameters of the model form a parameter space which is searched in order to find the best solution to the given data subject to various constraints. As such, a wide variety of algorithms exist ranging from random or evolutionary driven searches [297], [298] to gradient based methods [299].

5.3.1 Types of model

A model is simply a function that maps the input vector to an output y:

$$y = f(\mathbf{x}), \tag{5.2}$$

where x is the input vector and $f()$ is the functional mapping.

The choice of function is dictated by the complexity of the data to be fitted and desired accuracy. One of the simplest models is a linear relationship,

$$y = c_1x_1 + c_2x_2 + \ldots + c_Mx_M + c_0 = \sum_{i=1}^{M} c_ix_i + c_0 = \mathbf{c}^T\mathbf{x} + c_0 \tag{5.3}$$

where the coefficients c_i represent the weights associated with each input x_i and c_0 is the bias or offset weight and **c** is the vector of coefficients of size $1 \times M$.

Whilst simple, it cannot accurately approximate non-linear functions, limiting its usefulness in this application as the graphs in Section 5.2 show that a linear model will not be sufficiently accurate for speed prediction. A more powerful model includes higher powers of the inputs and is referred to as a polynomial model. This can either include or ignore cross terms (products of separate inputs, such as x_1x_3). Ignoring cross terms has the result that the number of parameters of the model grows linearly both in terms of the number of inputs and the order of the polynomial. The nth order polynomial model (with no cross terms) is:

$$y = c_{11}x_1 + c_{12}x_1^2 + \ldots c_{1n}nx_1^n + \ldots c_{m1}x_m + c_{m2}x_m^2 + \ldots c_{mn}x_m^n + c_0 \tag{5.4}$$

This model is more useful in approximating complex functions compared with the simple linear model, but if it is fitted to discontinuous data it sometimes does not generalise well to data not in the training set. This model can be regarded as a transversal filter of length $m \times n$, where the inputs are pre-calculated to form the polynomial terms [296].

A very powerful, yet complex, model is the three layer (one input layer, one hidden layer and one output layer) feedforward artificial neural network with M inputs and N 'hidden' neurons [296]. The output, y, is given by

$$y = \tanh(b_0 + \sum_{i=1}^{N} b_i \tanh(\sum_{j=1}^{M} a_{ij}x_j)) \tag{5.5}$$

where b_i represents the weight from the hidden layer neuron i to the output layer and a_{ij} is the weight from input neuron j to hidden neuron i. The bias term is given by b_0.

In this case the $\tanh$ activation function is used as opposed to the more common logistic or sigmoid function as it is commonly available as a built-in function in the float library for microcontrollers. In addition, computation of the derivative is simpler, yielding a faster execution time for the training algorithm. An artificial neural network can in general approximate any non-linear function with sufficient numbers of hidden neurons [300]. They are computationally intensive however, as the transcendental hyperbolic tangent function is approximately an order of magnitude more complex to evaluate compared to a multiplication operation. In addition, adjusting the weights in a neural network is more complex than for the polynomial model, as the error adjustment needs to be made to two separate layers of weights. Also note that the neural network model has $N + NM + 1$ coefficients, with the implication that increasing the number of hidden neurons will result in a rapid growth in the number of model parameters.

Now that some models have been presented, algorithms that can be used to train them are discussed.

5.3.2 The Least Mean Square (LMS) Learning Algorithm

The Least Mean Square (LMS) algorithm, commonly known as the 'delta-rule', was introduced by Widrow and Huff and is a stochastic gradient descent algorithm [301]. It is computationally simple and is not memory intensive. However, it does suffer from slow convergence speeds. The LMS algorithm is an approximation of the steepest descent algorithm, as it utilises an instantaneous estimate of the gradient of the error function. It is stochastic in that it acts on samples in isolation which can be corrupted with random noise, but follow a general trend. The steps involved in updating the coefficients **c** of the adaptive model are as follows, where n is the iteration or sample number [296]:

Feedforward

$$y[n] = \mathbf{c}^T[n]\mathbf{x}[n]$$

Error calculation

$$e[n] = d[n] - y[n]$$

Weight update

$$\mu[n] = \frac{\kappa}{||\mathbf{x}[n]||^2}$$

$$\mathbf{c}[n+1] = \mathbf{c}[n] + \mu[n]\mathbf{x}[n]e[n]$$

Parameter κ controls the learning rate of the LMS algorithm. Large values of κ results in a large step size and rapid adaption to new input, but at the cost of 'forgetting' prior data. In addition large κ can result in oscillation around the function minimum [296]. Conversely, a small value of κ results in slow adaption to new information. The coefficient vector **c** is initialised to sensible values if prior information is known about the relationship between input and output, but if no information is available can be set to a zero vector [296].

It can be seen that there are three steps to updating the model. In the first step, the estimate of the function is calculated from the inputs. This is then compared to the desired signal, $d[n]$ in order to determine the error between the actual and predicted value. This error signal is then used to alter the coefficients in the direction that reduces the error.

For the neural network model, an extra step has to be taken to correctly update the weights through the hidden layer back to the inputs. This is commonly called the back-propagation algorithm, as the error alters the weights in a feedback direction [302]. The process is shown in Algorithms 3 and 4. Parameters μ_0 and μ_1 control the learning rates in the hidden and output layer weights respectively, in a similar manner to κ for ordinary LMS.

Algorithm 3: Feedforward

```
for j=1:N do
    H[j] = 0;
    for k=1:M do
        H[j] = H[j] + x[k] × a[j][k];
    endfor
    H[j] = tanh(H[j]);
endfor
y[n] = 0;
for j=1:N do
    y[n] = y[n] + H[j] × b[j];
endfor
y[n] = tanh(y[n]);
```

One major issue with the neural network is that the input and output data needs to be scaled (for the tanh model to lie on the domain [-1:1]). Clearly this is a problem when the maximum and minimum values are not known beforehand – thus they need to be set to extreme values to prevent saturation. This can create problems if the variation of the input/output signal is very small compared to the predefined limits. In addition the step-sizes μ_0 and μ_1 need to be carefully tuned to adapt the model whilst still 'remembering' prior data.

Algorithm 4: Weight Update (Backpropagation)

```
/* Error calculation                                          */
e[n] = d[n] - y[n];
/* Backpropagation                                            */
Delta = (1 - y[n] × y[n]) × e[n];
for j=1:N do
    HD[j] = (1-H[j] × H[j]) × b[j] × e[n] ;
endfor
for j = 1:N do
    b[j] = b[j] + ($\mu_1$× Delta × H[j]);
endfor
for k = 1:M do
    for j=1:N do
        a[j][k] = a[j][k] + ($\mu_0$× HD[j] × x[k])
    endfor
endfor
```

5.3.3 The Recursive Least Square (RLS) Learning Algorithm

Whilst the LMS algorithm is very simple, it does suffer from slow convergence speeds. Furthermore, the setting of the step update factor is a matter of trial and error, depending on the distribution of the input vector. The RLS Algorithm uses information from all past input samples to estimate the autocorrelation of the input vector, rather than an instantaneous estimate [296]. However, the autocorrelation matrix is calculated recursively, meaning that past input samples do not need to be stored [296]. It is thus more complex and memory intensive than the LMS algorithm, but converges more rapidly. The RLS algorithms is a specialised version of the Kalman filter [302]. The steps involved in each iteration of the RLS algorithm are as follows [296]:

Feedforward

$$y[n] = \mathbf{c}^T[x]\mathbf{x}[n]$$

Error calculation

$$e[n] = d[n] - y[n]$$

Weight update

$$\mathbf{k}[n] = \frac{(\rho^{-1}\mathbf{\Phi}^{-1}[n-1]\mathbf{x}[n])^T}{1+\rho^{-1}\mathbf{x}^T[n]\mathbf{\Phi}^{-1}[n-1]\mathbf{x}[n]}$$

$$\mathbf{\Phi}^{-1}[n] = \rho^{-1}\mathbf{\Phi}^{-1}[n-1] + \rho^{-1}\mathbf{k}[n]\mathbf{x}^T[n]\mathbf{\Phi}^{-1}[n-1]$$

$$\mathbf{c}[n] = \mathbf{c}[n-1] + \mathbf{k}[n](d[n] - \mathbf{x}^T[n]\mathbf{c}[n-1])$$

The prediction and error calculation steps are the same as for the LMS model, but at each iteration, the inverse autocorrelation matrix $\mathbf{\Phi}^{-1}$ is updated, by first computing an intermediate error vector $\mathbf{k}[n]$. This error vector is also used to alter the coefficients of the model, shown in the last line of the formulation.

Variable $0 < \rho < 1$ is the 'forgetting factor' which controls how the algorithm reduces the weight of old samples in calculating the least squares fit. If ρ is set to 1, the data is never discarded from the recursive memory. The other parameter which affects the convergence speed of the RLS algorithm is the initial setting of the inverse autocorrelation matrix, $\mathbf{\Phi}^{-1}$. This needs to be set to $\delta^{-1}\mathbf{I}$, where $\mathbf{I}$ is the identity matrix and δ is a small positive constant. Although the algorithm is more computationally intensive and requires storage of the inverse autocorrelation matrix[10], it converges much more rapidly than the LMS algorithm, as will be shown in the following section.

5.3.4 Performance comparison of LMS and RLS

Computational Complexity

The various algorithms differ greatly in their computational complexity, which impacts on their execution time. Furthermore, their memory requirements (both for temporary and permanent variables) are also different. Table 5.3.4 shows the number of operations required for the processes of each algorithm. In the table, N refers to the number of weights or co-efficients that need to be trained. For the backpropagation algorithm, there are an additional M weights in the hidden layer that also need to be trained. It can be seen that the LMS algorithm scales linearly i.e. requires time $O(N)$ to execute. The backpropagation algorithm scales as $O(NM)$ and the RLS procedure as $O(N^2)$. As low cost microcontrollers do not generally possess a floating point unit, floating point arithmetic is handled by multiple bytewise operations. Higher end DSP-based processors could be used to increase computation speed, but their power consumption is much higher than their simpler counterparts and thus they are not suitable for this application [303]. For the processor in question (PIC18F series), an addition typically takes 100 cycles, a multiplication 250 cycles and the $\tanh$ function takes 3 500 cycles.

A graph showing the variation in training time of the various algorithms is shown in Fig. 5.10[11]. All these algorithms were implemented on the PIC microcontroller by the author

[10]The autocorrelation matrix is also sometimes termed the covariance matrix. It is possible to reduce the storage requirements further by using a square-root decomposition, which makes the autocorrelation matrix symmetric, meaning that only half as much storage space is required.

[11]The processor used was clocked at 8 MHz, resulting in an execution speed of 2 million instructions per second (MIPs).

Table 5.1: Comparison of computational complexity for the various algorithms. Memory requirements are expressed in bytes

Algorithm	Process	Additions	Multiplications	Tanh	Memory
LMS	Feedforward	$N-1$	N	0	
	Error	1	0	0	
	Adjustment	$2N-1$	$3N+2$	0	$4N$
Backpropagation	Feedforward	$MN+N$	$MN+N$	$N+1$	
	Error	1	0	0	
	Backpropagation	$2MN+2N+1$	$2MN+5N+2$	0	$4MN$
RLS	Feedforward	$N-1$	N	0	
	Error	1	0	0	
	Adjustment	$2N^2+3N+1$	$7N^2+N+1$	0	$4N^2+4N$

for testing of their suitability and operation. For the neural network, it was assumed that the number of hidden neurons was equivalent to the number of inputs (i.e. $M = N$). It can be seen that as the number of weights increases, the training time of the RLS algorithm increases rapidly, whereas the training time for the LMS algorithm increases linearly. The RLS algorithm is more efficient than the backpropagation algorithm if the number of weights is less than five.

The memory requirements of the RLS algorithm are much greater than the LMS and neural network algorithms, as the $N \times N$ matrix of the autocorrelation of the input vector has to be stored between iterations. A floating point number (32 bit) requires 4 bytes of RAM to hold. Thus, the RLS algorithm requires a static buffer of $4N^2$ bytes. As an example, if the RLS algorithm has to train a model with 10 weights, 400 bytes of dedicated RAM are required, which is a large proportion of a low cost microcontroller's available RAM (in this case, 3986 bytes [303]). In addition, the RLS algorithm requires an additional $4N^2$ bytes for temporary storage during the training phase. The LMS algorithm does not require that any data (other than the coefficients of the model) be stored between iterations, relaxing its memory footprint.

Convergence speeds

As the RLS algorithm keeps a recursive memory of all past samples (without having to store them all), it converges to an accurate solution much more rapidly than the LMS algorithm. For this application, the model must incorporate new data rapidly, whilst still retaining old data, so that an accurate mapping between acceleration profile and speed can be rapidly constructed. The difference in convergence speeds is shown for the normalised LMS and RLS algorithms in Fig. 5.11 for a simple linear function corrupted with additive noise (see caption for simulation parameters).

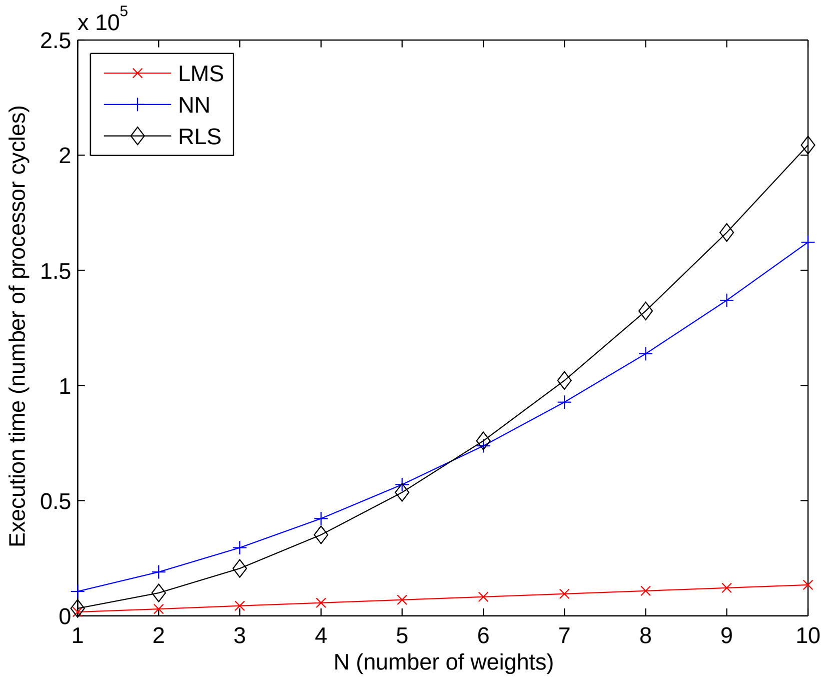


Figure 5.10: Number of cycles for one training iteration of the various algorithms for varying number of coefficients.

The plot shows that the RLS algorithm converges rapidly in comparison to the LMS algorithm. The simulation time-step is 1 second, and training is performed online at each iteration. Noise is gaussian with a variance of 0.1. After approximately 15 iterations, the RLS algorithm has reached a stable value, whereas after 100 iterations, the error in the LMS algorithm is still decreasing. This is as a result of the LMS algorithm not storing the autocorrelation matrix but estimating it from the instantaneous value of the input vector.

5.3.5 Learning performance of the algorithms with various models

The performance of the various learning methods were tested on real data acquired from a human walking. The inputs applied to each model were the acceleration range and jerk variance, as computed using the method described in Section 5.2. The sampling rate of the GPS receiver was 0.5 Hz, and the 1.2 s acceleration snapshot was taken in between GPS samples. The speed (as reported by the GPS unit) was used to train the model and to determine prediction performance.

To fairly compare the linear/polynomial models to the neural network, the inputs and outputs were scaled to be in the range $[-1:1]$[12]. A simple linear model (with a bias term)

[12]This is because the $\tanh$ function used in the neural network requires inputs to be on this range – refer

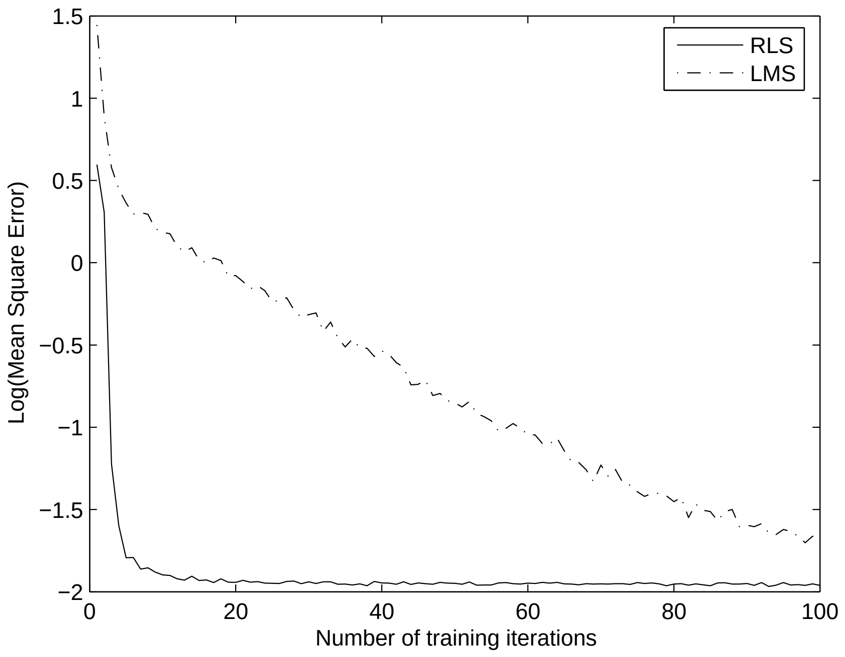


Figure 5.11: Convergence speeds of LMS and RLS to linear function $y(t) = 9t + 7 + n(t)$, where $n(t)$ is random noise with variance = 0.1. LMS parameter κ = 0.6 and RLS forgetting factor ρ = 0.9. A simple linear model was used for both RLS and LMS training methods.

was compared against the neural network model. The linear model used was

$$y[n] = c_1x_1[n] + c_2x_2[n] + c_0,$$

where y is the predicted speed, x_1 is the acceleration range and x_2 the jerk variance. Coefficients c_1 and c_2 are the coefficients for the range and jerk variance respectively and c_0 is the bias coefficient.

The neural network model used was

$$y = \tanh(b_0 + \sum_{i=1}^{3} b_i \tanh(\sum_{j=1}^{2} a_{ij}x_j))$$

where y is the speed as predicted by the model, b_i represents the weight from the hidden layer neuron i to the output layer and a_{ij} is the weight from input neuron j to hidden neuron i. The bias term is given by b_0.

Both the LMS algorithm and the RLS algorithm were used to train the linear model, and the backpropagation algorithm was used to train the artificial neural network. The

to Section 5.3.2.

neural network was configured to have two input neurons and three hidden layer neurons with a bias term, resulting in nine weights, whereas the linear model has three weights. The learning rates for the neural network were chosen to be μ_0 = 0.6 and μ_1 = 0.5, as these were found to result in the best performance. The step size factor for the LMS algorithm was set as κ = 1.0. For the RLS algorithm, δ = 0.01, but this value was not found to be critical, as long as it was made small. The forgetting factor, ρ, for the RLS algorithm was set to 0.99. The algorithms were trained for 250 samples of the input data (sampled at 2 s intervals and thus the training phase took 500 s to complete). It should be noted that this does not mean that 250 samples were applied to the learning algorithm, as the GPS filter removes samples that could be in error. After this time, the model were used to predict the speed of the host (that is, no further training occurred). The results of this test are shown in Fig. 5.12.

The test results show that the LMS algorithm on a simple linear model tracked the speed of the human subject very well during the training phase, but the error during the prediction phase was large (Mean Square Error (MSE) = 0.041). The neural network performed badly, but showed a smaller prediction error than the LMS model (MSE = 0.036) as its mean value is similar to the mean value of the actual speed. The RLS algorithm demonstrated excellent tracking and prediction performance (MSE = 0.010), as it converges rapidly to an accurate estimate of the process. In addition, it does not require careful setting of parameters, unlike the LMS algorithm where the choice of step size is critical. Lastly, it does not require *a priori* knowledge of the maximum and minimum ranges of signals as does the neural network. Thus, in spite of its complexity and increased memory requirements, the RLS algorithm is more suitable for this application and is thus chosen as the learning algorithm.

The test shown in Fig. 5.12 was undertaken using a simple linear model, which cannot fit non-linear functions. Higher order polynomial models were considered, both with and without cross-terms. It was found that a second order model with no cross terms provided acceptable performance without being excessively computationally intensive:

$$y = c_0 + c_1 x_1 + c_2 x_1^2 + c_3 x_2 + c_4 x_2^2 \tag{5.6}$$

where y is an estimate of the speed, x_1 is the range, x_2 is the jerk variance and parameters c_i are the co-efficients associated with each term in the equation. The choice of a second order model was also guided by shape of the graphs shown in Fig. 5.2, which show a non-linear relationship between stride frequency and host speed. For the rest of the chapter, the predicted speed is unscaled, i.e. will predict the true value.

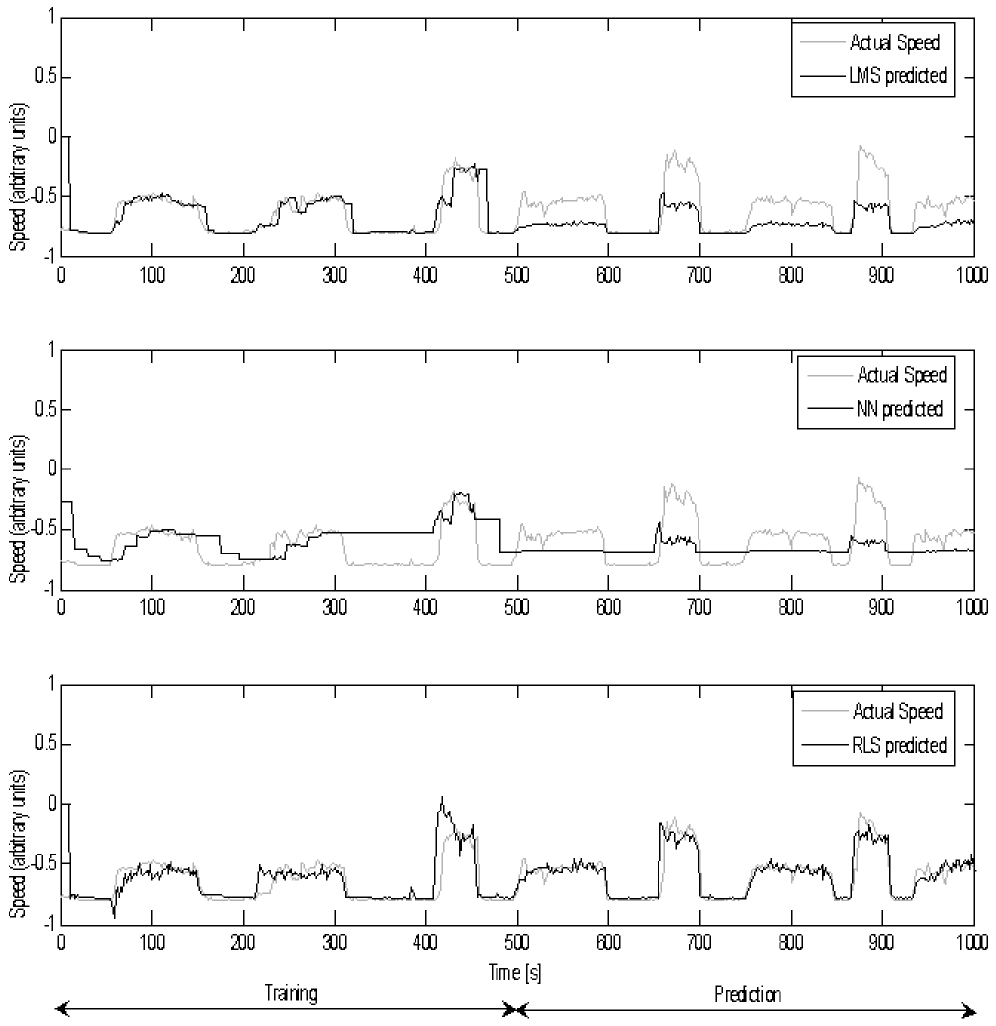


Figure 5.12: Comparison between the different models and learning algorithms. Top plot: LMS used to train the linear model. Middle plot: Neural network. Bottom plot: RLS used to train the linear model. The first half of the experiment was the training phase, where the model was updated. In the second half of the experiment, the speed was estimated. The actual speed is that reported by the GPS receiver.

5.3.6 When to train the model

It is not desirable that model updating happens every time the GPS unit is triggered (it typically takes 10–20 s to acquire the first fix, and to obtain 10 speed–acceleration training pairs will take a further 20 s, assuming a 0.5 Hz GPS sampling rate). Instead, the model is only updated if the prediction is no longer accurate[13]. Training is undertaken

[13]This would be the case if the host animal undertook a previously unss gait pattern or if the acceleration parameters altered, for example as a result of tracking collar rotation.

if:

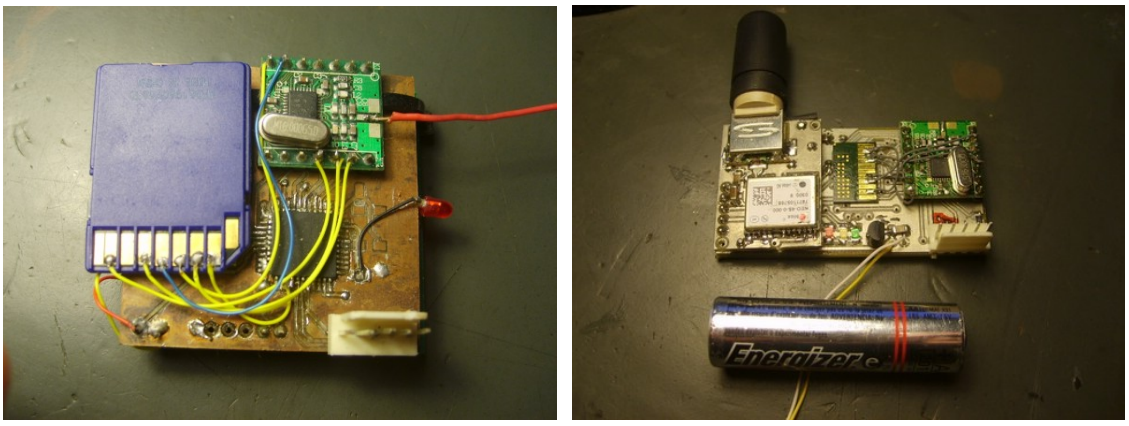
$$abs(d[n] - y[n]) > ad[n] + b, \tag{5.7}$$

where $d[n]$ is the speed as measured by the GPS receiver, $y[n]$ is the predicted speed, a is the proportional error to be tolerated and b is the bias. Values that were found to result in good performance for this application were $a = 0.1$ and $b = 0.2$ m/s.

5.4 Real World Testing and Performance of Training Algorithm

To test the suitability of the model and the training method, datasets were captured from two different dogs (using neck mounted collar) and from a human carrying the device on different locations of the body. Two variations of the devices were designed and fabricated by the author in order to test the training algorithm. These are shown in Fig. 5.13. These devices are greatly reduced in size compared to the first prototype shown in Fig. 5.3. Both units have a triaxial LIS3L02AS4 acceleration sensor[304] and use 256 Mb SD memory cards for mass storage [305]. However, only one axis of acceleration data was used in the generation of acceleration range and jerk variance. The only real difference between the two versions is that the newer unit has a more sensitive GPS receiver (the NEO-4S as opposed to the LEA-4P module[122]) and a more powerful microcontroller (the PIC18F67J11 as opposed to the PIC18LF4620 for the older unit[303]). The use of the SD memory card allows for inexpensive storage of vast amounts of data – an issue with the prototype data logger in Fig. 5.3 was that it could only store 40 kb of data, leading to rapid memory filling when acquiring and storing acceleration signatures for further analysis. The system design for the tracking device is shown in Fig. 5.14. Code was written in C, and the recursive least squares learning algorithm was implemented on the microcontrollers. The devices automatically uploaded their acquired data to a Base tag over the wireless network. Further details about the hardware and the software operation can be found in Chapter 6.

The subjects walked and ran around an urban neighbourhood, with intermittent stops at road junctions. To show the effectiveness of the learning algorithm in adapting the parameters of the model to the behaviour of the host, training was explicitly disabled after a time t = 300 s. The datasets for the two dogs are shown in Fig. 5.15. Dog A was a 35 kg labrador and Dog B was a 22 kg mixed breed. To test whether device location was critical for a human carried device, the device was carried in the hand and also attached to a belt. For the hand-carried test, the accelerometer is actually picking up arm-swings. A uni-axial accelerometer was used, oriented such that it captured acceleration in the vertical (z) axis. The collar on the dog was prevented from rotating by placing the batteries underneath, which acted as a counterweight. These two locations provided very



(a) (b)

Figure 5.13: The different devices used for testing the uniform distance sampling approach. (a) was equipped with a triaxial acceleration sensor and used an SD memory card for storage of the raw acceleration profiles. (b) is the latest version, also with a triaxial accelerometer and an SD memory card, but uses a high sensitivity GPS receiver with a helical stub antenna.

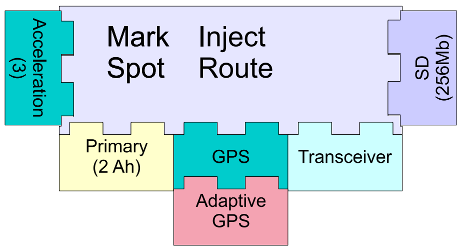


Figure 5.14: System diagram of the uniform distance sampling tracking device.

different summary statistics (acceleration range and jerk variance), yet the learning algorithm was able to train the model in both of these cases. The two human datasets are shown in Fig. 5.16. These speed-time datasets are those produced by the microcontroller. These demonstrate that the microcontroller is able to train the model and estimate speed online[14]. No post-processing has been undertaken on this data.

Figs. 5.15 and 5.16 demonstrate that the model is able to learn and predict the speed of different hosts, even with a very short training set. Quantifying error performance is problematic because the speed measured by the GPS unit is itself corrupted by noise. However, to provide a metric of prediction accuracy, the cumulative distance measured by the GPS unit was compared to the cumulative distance estimated by the trained model, for the time period from 300 s to 1000 s. Model training was explicitly disabled

[14]For the microcontroller used, speed prediction takes less than 1 ms and training approximately 60 ms. The microcontroller was clocked at 8 MHz, resulting in an execution speed of 2 MIPs.

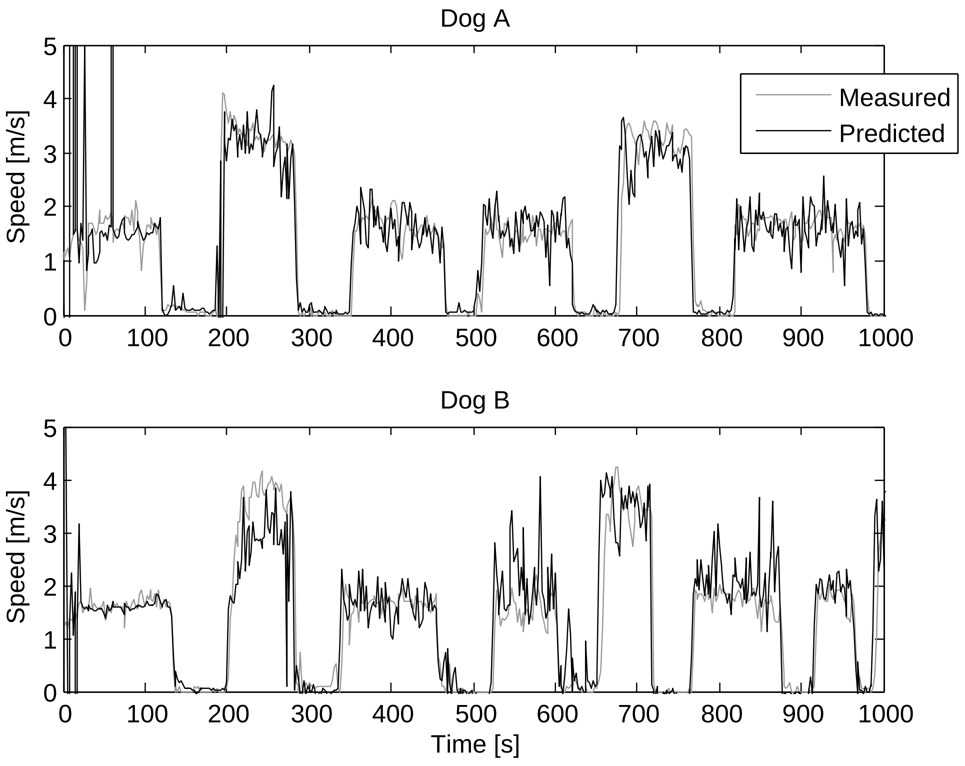


Figure 5.15: Speed-time plots illustrating the predicted and measured speeds for two different dogs. Dog A was a 35 kg labrador and Dog B was a 22 kg mixed breed. At time t = 300 s, further training was disabled. Note the large training errors at the start of the dataset.

for the 700 s interval. The results are shown in Table 5.2.

Table 5.2: Difference between distance measured by GPS unit and predicted by model after 1000 s

Test	Measured [m]	Predicted [m]	Percent Difference
Dog A	915	914	0.1%
Dog B	837	967	15.5%
Human (Hand)	674	624	-7.4 %
Human (Belt)	861	882	2.4 %

The results demonstrate that this system is capable of adequately estimating the speed for different animals from a short snapshot, without being provided with prior knowledge about the way the host moves. The percentage differences compare favourably with PDR systems with quoted errors in the region of 2% [282] to 10% [283], especially considering the simplicity of the chosen statistics[15].

[15]The reason why the error for Dog B is so much larger than the other tests is unclear. It is thought that this could be due to the fact that the collar did not fit this animal as snugly as the collar on Dog A, leading to the accelerometer being subject to regular knocks as the device bumped against the dog's shoulder blades.

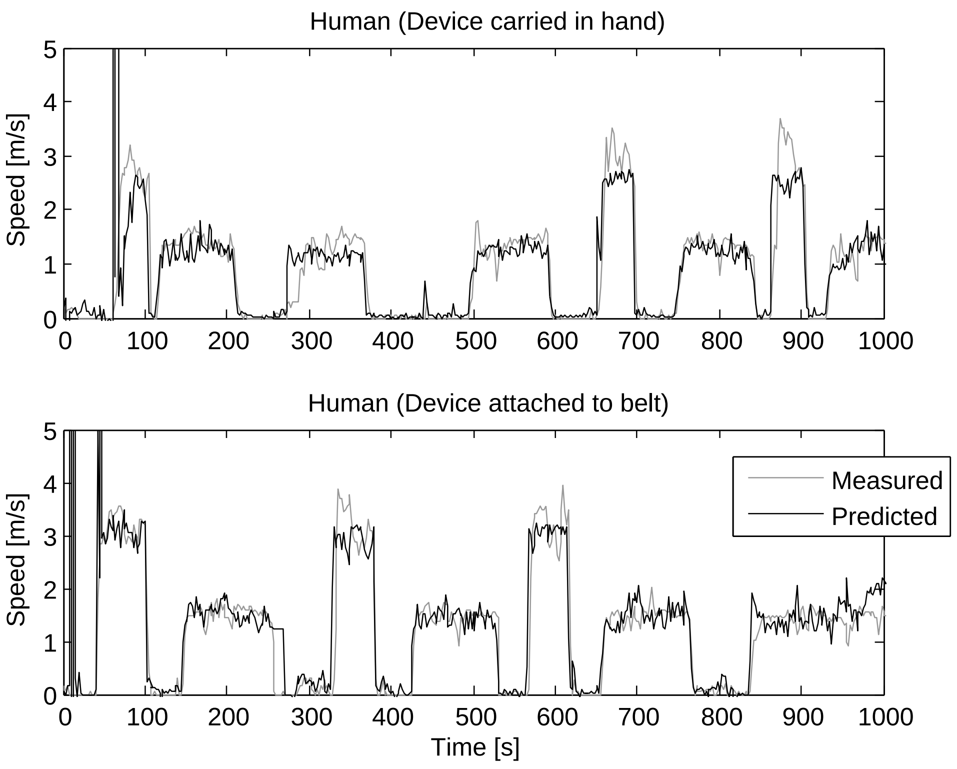


Figure 5.16: Speed-time plots illustrating the predicted and measured speeds for human carried device. In the top plot, the device was carried in the left hand of the author, in the bottom plot, worn on a belt. At time t = 300 s, further training was disabled. Note the large training errors at the start of the dataset.

Fig. 5.17 show the fitted polynomial surfaces in terms of the two variables (acceleration range and jerk variance) for the various subjects. Any negative speed predictions have been clamped to zero in the plots. These surfaces show the large variation in the predictors between the different tests. In particular, note the different shape of the surface for the different attachment methods to the human. These surfaces, coupled with the speed-time plots shown in Figs. 5.15 and 5.16 demonstrate that the algorithm is able to accurately learn the relationship between the summary parameters and the speed of the host.

The next section discusses how to schedule GPS fixes based on the estimated distance travelled.

5.5 GPS Triggering

To achieve the aims of distance based sampling, the GPS unit is triggered (that is, woken up and instructed to acquire a location fix) when the estimated distance travelled since

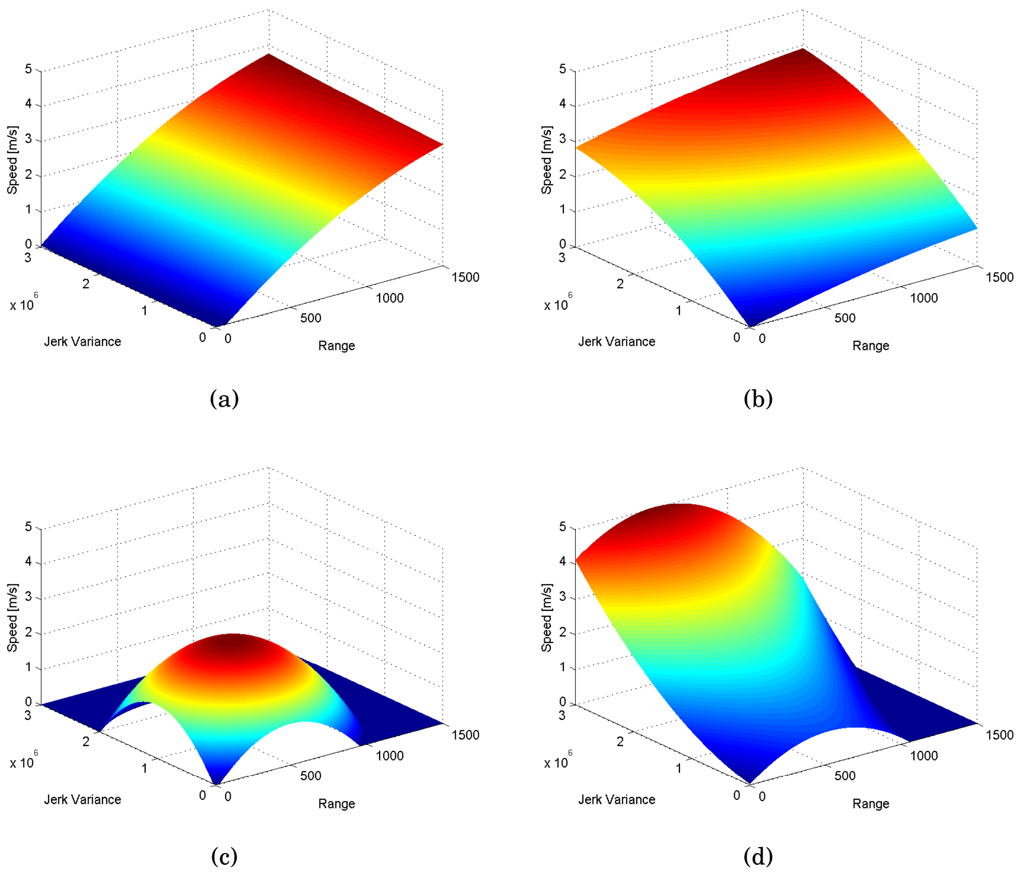
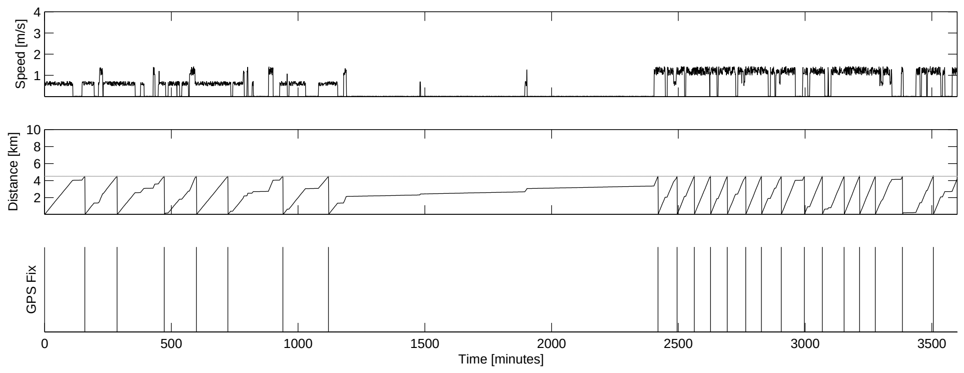


Figure 5.17: Surface plots of the generated model surfaces for the various tests. (a) and (b) are for dog A and B respectively. (c) is for the hand carried human device and (d) for the belt worn device.

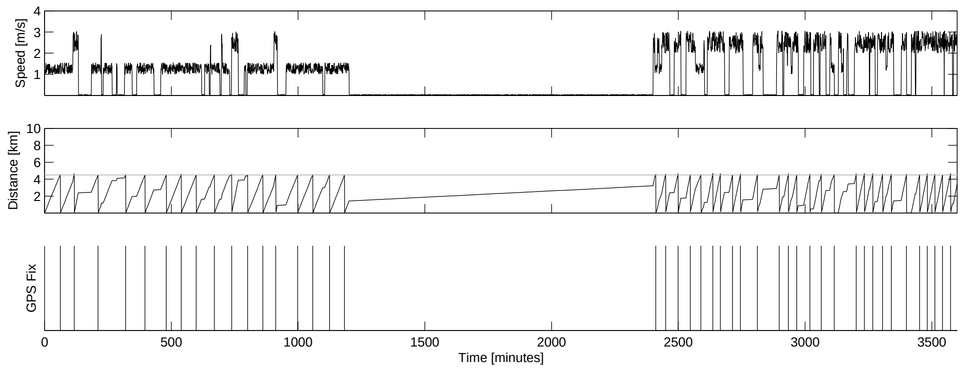
the last fix exceeds a certain threshold distance. Two different methods of determining what threshold distance to use are compared, one based on a preset threshold distance and another that alters the threshold distance in relation to the habits of the host. Both perform uniform distance sampling.

5.5.1 Constant Threshold Distance Activation

The threshold distance can be specified prior to attaching the collar on the animal (for example, one fix to be taken every 500 m), resulting in the number of attempted fixes being proportional to distance travelled. However, the choice of this threshold distance is critical, being dependent on the way the animal moves. If the threshold distance is set too high, the GPS unit will rarely be triggered, resulting in a sparsity of data. Conversely, if the threshold distance is made too low the GPS unit will fire often which will rapidly exhaust the batteries of the tracking device. In wildlife research, the motion of the animal is often unknown (hence the necessity of equipping it with a tracking collar) and so choosing a fixed distance threshold is somewhat a matter of trial and error. The lifetime of the device is also unpredictable, as it depends on how far the animal has travelled. The firing patterns of constant distance activation for two different *simulated* animals are shown in Fig. 5.18, over a 60 hour period. Both hypothetical animals are inactive for a portion of the simulation. The animal in the bottom graph moves faster than that in the top, resulting in a higher firing rate (43 fixes over the time period, as opposed to 22 for the bottom animal), which leads to a shorter tracking device lifetime.



(a)



(b)

Figure 5.18: Comparison of firing rates for two different simulated animal patterns using constant threshold distance activation. The horizontal grey line is the threshold. Note that a constant threshold results in an animal dependent firing rate, with the implication that collar lifetime is somewhat unpredictable. For each simulated animal, the top graph is the speed of the animal, the middle graph is the cumulative distance travelled since the last GPS fix and the bottom graph indicates when GPS fixes are taken.

5.5.2 Adaptive Threshold Distance Activation (ATDA)

Due to the difficulties of choosing a threshold distance, an adaptive threshold is proposed. Instead of choosing a constant threshold distance, the user specifies a desired *average* fix rate (number of fixes per unit time) over a *period* of time. As an example, assume that the desired rate is 48 GPS fixes over a 24 hour time period. At the start of each day, the collar determines a threshold distance to be used for the rest of the day. At the end of the time period, the total distance travelled in the prior 24 hour period is used to update the average distance travelled per day. Based on the updated average, a new threshold distance is calculated for the next day. Thus, the threshold distance is dynamically adjusted over time in order to achieve the desired daily number of fixes, whilst still performing uniform distance sampling within each day[16]. In addition, as the average fix rate is pre-specified, ensuring that the collar achieves a certain lifetime is made easier, and the volumes of data generated by the collar are more predictable.

To adjust the threshold or trigger distance an exponentially weighted moving average filter is used to 'learn' the cumulative distance travelled over the time period. The long term estimated cumulative distance D_k is updated with the cumulative distance from the last time period

$$D_{k+1} = \delta \sum_{i=1}^{N} y_i + (1-\delta)D_k \tag{5.8}$$

where $0 < \delta < 1$ is the adaption rate, N is the number of speed estimates taken in the time period and y_i is the estimated distance travelled (discussed in Section 5.3). The estimated distance is summed over an interval which is chosen to be long enough to capture the typical behaviour of an animal. The adaption rate controls how rapidly the long term estimated distance is updated with the new cumulative distance. Large values of δ result in a high rate of adaption, but correspondingly short memory of prior data. It was found in practice that a value of $\delta = 0.05$ provides a good long term estimate with sufficient adaption to variations in the host creature's behaviour.

The threshold distance D_{thresh} is adjusted such that the desired average fix rate R is achieved

$$D_{thresh} = \frac{D_k}{R}. \tag{5.9}$$

In this way, the GPS fixes are uniformly spaced in distance over a time period so as to achieve a preset fix budget.

There are however some practical concerns with uniform distance sampling with regard to the operation of the device. GPS units typically need to be powered up at least once every four hours or the ephemeris data becomes invalid, resulting in a much longer ac-

[16]Note that the time period does not have to be 24 hours, and can be days or weeks long.

quisition time (cold start) [289]. Thus, this implies that even if the animal is stationary, there should still be a minimum fix rate.

Another issue is that if the animal's behaviour suddenly changes (such as altering from a sedentary state to long distance migration pattern), the GPS unit will be triggered frequently until the threshold is updated. This will have a detrimental effect on the lifetime of the device, but will capture interesting information. If device lifetime is the primary concern, an upper bound on the fix rate can be specified such that the minimum lifetime of the device can be predicted. For animal tracking, a suitable upper bound is to take fixes no more often than once every five minutes. This figure could be very different for a mobile device carried by a person. It should be noted that by setting the maximum and minimum fix rates to be the same, it is possible to make the system act as a uniform time sampling device if so desired.

Firing patterns for the same two simulated animals in Section 5.5.1 are shown in Fig. 5.19, using the same simulation parameters and random seeds. The time period over which the threshold is adjusted is made to be three hours long, in order to highlight the way the threshold changes. The fix rate, R, was chosen to be two (that is, one fix every 90 minutes). The maximum fix rate was unlimited and the minimum fix rate was set so that the GPS unit would be triggered at least once every four hours.

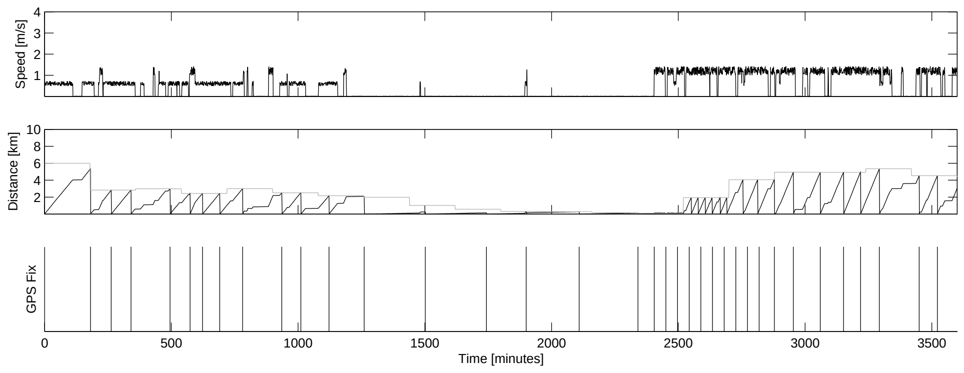
The threshold distance (shown on the graph as a light grey line) dynamically alters according to the distance travelled in each time window, resulting in similar firing pattern for both animals. The animal in the top graph acquired 35 GPS fixes, whereas that in the bottom graph acquired 38 GPS fixes. Also note that when the animal is stationary, fixes are still taken to satisfy the minimum firing rate, even though the threshold distance has not been exceeded. This avoids having to perform a cold-start on the GPS unit.

These simulation results demonstrate why adaptively altering the threshold distance is a valid approach to undertaking uniform distance sampling when the average distance travelled by the host animal is not known prior to deployment.

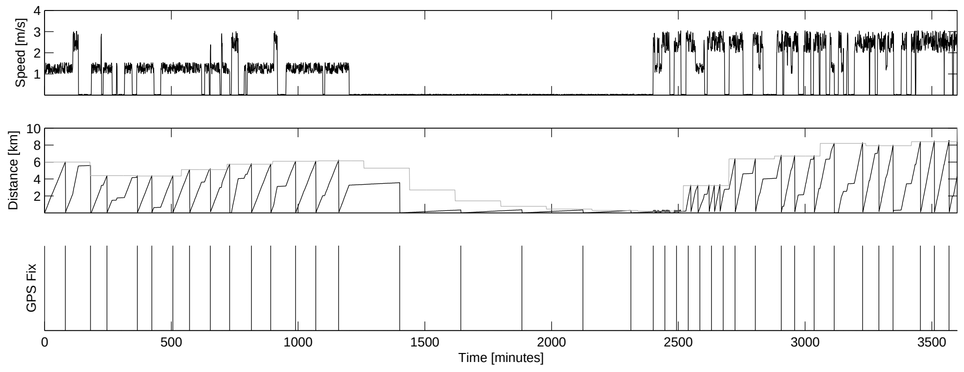
5.5.3 Constant vs Adaptive Thresholds

The two methods of scheduling fixes have different merits and drawbacks. A summary of these are shown in Table 5.3. Which method to choose depends on the wildlife study in question and what the requirements for the data are.

In most cases, adaptive threshold distance activation (ATDA) would be the most suitable as the behaviour of the animal does not need to be known in advance and the collar adapts to each individual in order to satisfy the pre-specified average number of fixes per unit time. As a consequence, the lifetime of the collars will be more predictable.



(a)



(b)

Figure 5.19: Comparison of firing rates for two different simulated animal patterns using adaptive threshold distance activation (ATDA). The grey line is the threshold. Note that with ATDA, the threshold is updated according to the behaviour and habits of the host. Even though these animals have different activity profiles, it can be seen that the firing profile is similar, with the implication that collar lifetime is more predictable. For each simulated animal, the top graph is the speed of the animal, the middle graph is the cumulative distance since the last GPS fix and the bottom graph indicates when GPS fixes are taken.

However, as the threshold distance changes over the course of the study, not all data points will be equally spaced in distance. For animals with regular activity patterns, once the threshold distance is established, data points will be spaced at approximately equal distance intervals[17].

Constant threshold distance activation would be useful in research on well-studied animals where a threshold distance can be intelligently specified. The main advantage of this method is that GPS data points over the entire study will be equally spaced in distance.

[17]A third approach could be undertaken, which is a hybrid of both techniques. When the collar is first placed on the animal, ATDA is used to learn a suitable threshold distance. Once the threshold distance is stable, the collar switches to constant threshold distance activation, using the threshold learned by ATDA.

Table 5.3: Comparison between the two methods of choosing the threshold

	Advantages	**Disadvantages**
Constant Distance Activation	GPS location data will be equi-distant	Choice of threshold distance is critical and depends on the individual animal
	Simple to implement	Collar lifetime is animal dependent
Adaptive Distance Activation	Collar lifetime is predictable	Distance interval between GPS samples will vary, possibly complicating statistical analysis
	No prior information required about animal habits	More complex to implement

5.6 Energy Comparison of Sampling Strategies

To demonstrate the usefulness of acquiring GPS fixes based on uniform distance sampling (using either constant or adaptive threshold activation) over uniform time sampling, the energy usage of the two methods are compared for a hypothetical animal.

The animal is assumed to be diurnal, spending approximately ten hours sleeping and fourteen hours grazing/running. Average energy usage is calculated for constant time sampling and ATDA (Section 5.5.2) for a total of 100 simulated days. The parameters used in calculating the energy expenditure of the two methods are shown in Table 5.6. The time window is how often the threshold is updated, and the learning parameter is the innovation rate, δ. The training frequency is the percentage of GPS fix attempts that result in updating the model via the learning algorithm. The training GPS on-time is how long the GPS unit has to be active to acquire enough fixes and acceleration snapshots to train the model. The prediction duty cycle is the proportion of time that the unit is predicting the speed – for the rest of the time it will be in a low power sleep mode. This duty cycle corresponds to acquiring an acceleration snapshot (which takes 2 s including calculation and storage of the estimated speed) every 30 s.

The results of the simulation for a maximum fix rate of 60 minutes for ATDA are shown in Fig. 5.20. The diagram (see Fig. 5.20 (a), middle plot) shows how the uniform time sampling method takes needless fixes when the animal is stationary. This is made evident in the cumulative density function of the distance between successive fixes (see Fig. 5.20 (b), top plot). This demonstrates that for approximately 50% of the fixes, the distance travelled between successive fixes does not exceed 100 m. Over the 100 day simulation time, the uniform time method acquired 2400 GPS fixes, whereas the ATDA method only took 1411 fixes. However, they both capture the same information, as 50% of the uniform time GPS fixes are in fact redundant and unnecessary as they correspond to repeated fixes in the same location. In addition, it must be noted that detailed speed-time profiles are also generated by the ATDA method. Over the 100 day simulation time, the average current consumption of constant time activation is 0.24 mA and that

Table 5.4: Parameters used in the simulation to compare energy usage of constant time and Adaptive Distance Activation

Constant Time Activation		**Adaptive Distance Activation**	
Sampling Period	60 minutes	Time Window	24 hours
Average GPS acquisition time	25 seconds	Learning Parameter	0.05
Active GPS current	35 mA	Minimum Sampling Interval	4 hours
		Desired Sample Rate	60 minutes
		Average GPS acquisition time	25 seconds
		Active GPS current	35 mA
		Training GPS on-time	45 seconds
		Training frequency	10%
		Accelerometer current	0.6 mA
		Prediction Duty Cycle	6.70%

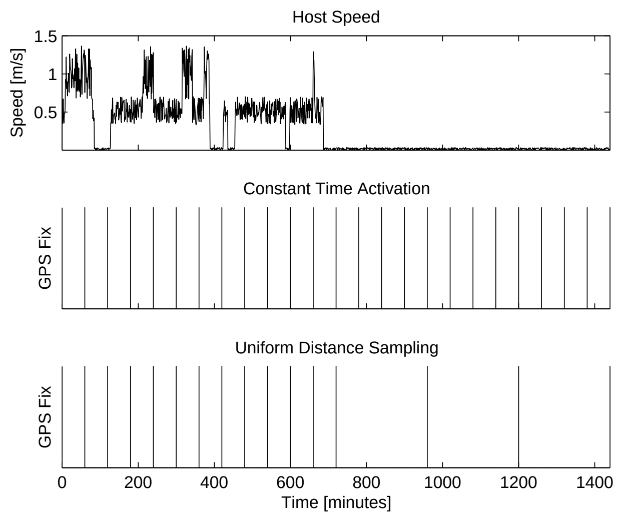
of ATDA is 0.19 mA. This implies that for the same capacity battery, the lifetime using the adaptive method will be approximately 25% greater than that of constant time activation whilst delivering more information (equivalent GPS locations and also the speed–time profile).

The sampling pattern of ATDA sampling with no limit on the maximum sampling rate is shown in Fig. 5.21. This shows that over 85% of GPS fixes are taken at a uniform distance interval. The number of fixes taken by the adaptive method slightly exceeds the desired rate – this is due to the effect of the minimum sampling interval which is used to prevent cold-starts. The energy usage of unbounded ATDA is 1.4 times as large as that of uniform time sampling, but the information acquired by the module is far richer. The maximum distance between fixes does not exceed 1 km for adaptive activation, but for constant time activation, some fixes are taken which are 3.8 km apart. This means that the accuracy of path reconstruction for ATDA will be greater than that of the standard method of acquiring GPS fixes at uniform time intervals.

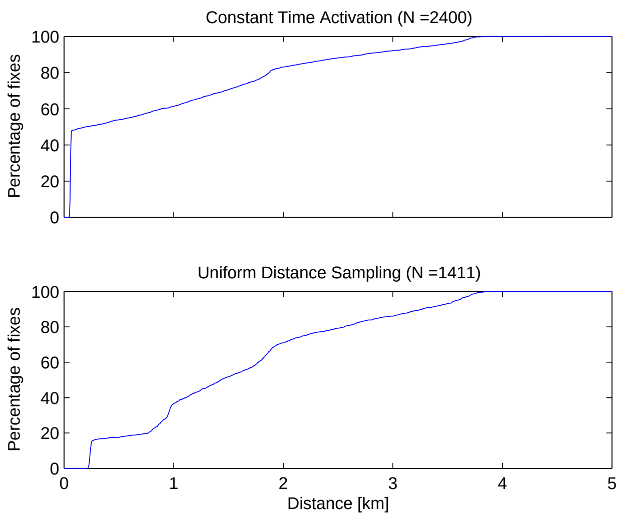
5.6.1 Effect of animal activity patterns on collar lifetime

The example from the prior section is now generalized to examine the energy savings for different animal activity patterns. Animals are often stationary for a large proportion of time. This proportion varies with different species of animal, and also with environmentally related effects (such as winter hibernation or migration). The behaviour of lions for example, is characterised by large periods of inactivity followed by brief periods of intense activity [30]. Lions are active for only approximately 40% of the time [30].

During these periods of inactivity, a conventional uniform time sampling GPS unit will acquire redundant fixes, which contribute no new information yet consume large



(a)

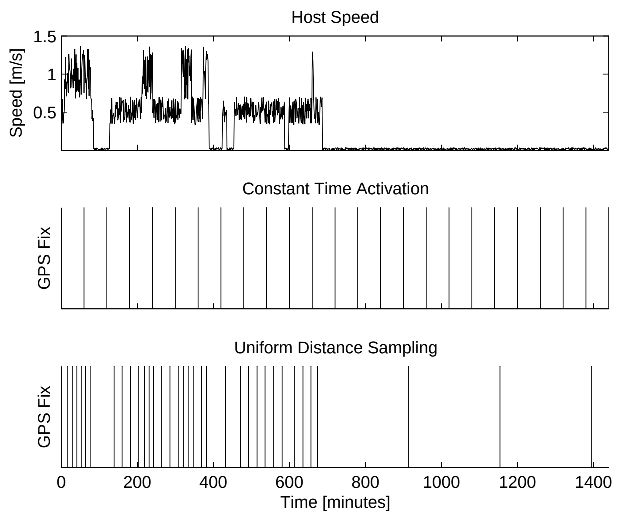


(b)

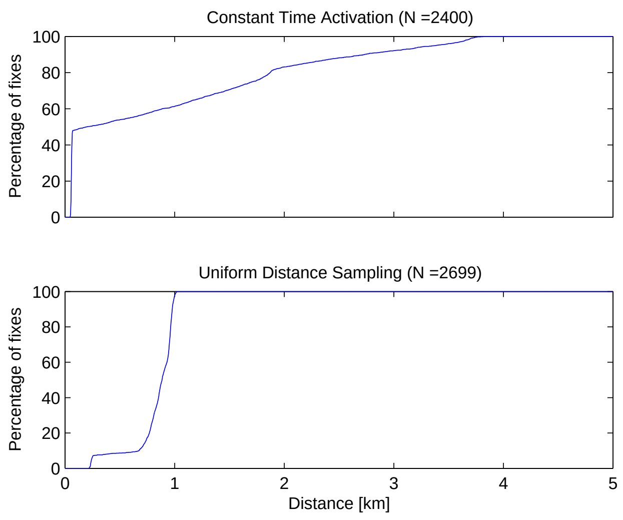
Figure 5.20: Simulation comparison with a maximum fix interval of 60 minutes for ATDA. (a) shows the speed of the animal with time and the corresponding GPS fix times for the two methods. (b) shows histograms of the distance between successive fixes. Average current consumption of constant time activation is 0.24mA and that of ATDA is 0.19mA.

amounts of energy. The uniform distance sampling strategy (whether constant or adaptive threshold distance activation is used) will keep the GPS unit in a low power sleep mode, prolonging the overall device lifetime with no loss of information.

In order to quantify the expected improvement in device lifetime, a simulation was performed with the following assumed parameters, using ATDA. The uniform time sampling system is set to sample locations once every 15 minutes. ATDA is configured to



(a)



(b)

Figure 5.21: Simulation comparison with an unlimited maximum sampling rate. (a) shows the speed of the animal with time and the corresponding GPS fix times for the two methods. (b) shows histograms of the distance between successive fixes. Average energy usage of constant time activation is 0.24mAh and that of ATDA is 0.34mAh.

sample locations with a maximum interval of 4 hours and a minimum interval of 15 minutes. It is assumed that when the animal is moving the minimum sampling interval will be used, and hence a desired fix rate does not need to be specified. The lifetime is calculated for a device that is equipped with a 3 V, 1000 mAh source of energy.

Fig. 5.22 shows device lifetime for the two different sampling strategies, plotted against the percentage of time for which the host animal is moving. The device lifetime of ATDA is substantially greater than the uniform time sampling strategy when the proportion of time that the host is active is low. As the percentage activity increases, this difference reduces until a point when the animal is 90% active, when the uniform time sampling strategy is actually slightly superior in terms of lifetime. This difference is due to the energy consumed by ATDA in training the model and acquiring snapshots in order to schedule fixes. Thus, the presented sampling strategy can extend the lifetime of the tracking device by a factor of 2.4 compared to conventional tracking for an animal which is only moving around for 30% of the time, whilst reproducing the path taken with equal fidelity. This is of great importance in particular to wildlife researchers due to the costs and logistics involved in capturing, sedating and collaring an animal.

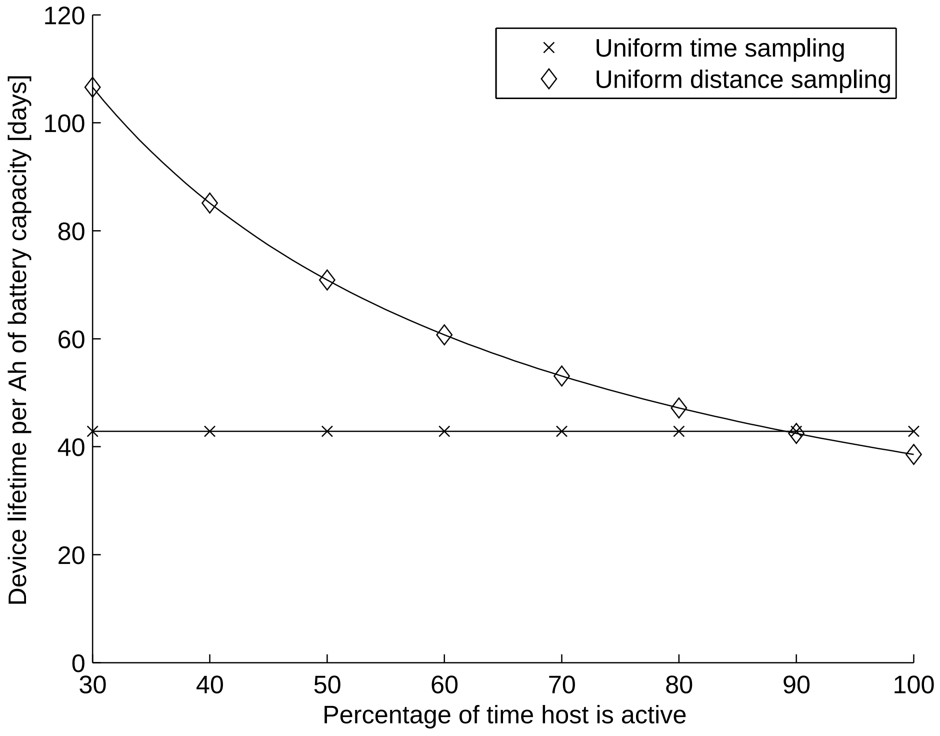


Figure 5.22: Comparison of device lifetime for the two different sampling strategies for different fractions of time that the host is moving around.

5.7 Path Reconstruction

Figs. 5.23 and 5.24 highlight the difference between conventional uniform time sampling and the proposed method of ATDA with an upper and lower bounded fix rate (in one spatial dimension only, so that time dependency can be shown). Fig. 5.23 is the reconstructed path of a hypothetical animal equipped with a uniform time sampling collar (shown by the dark line). The 'actual' path taken by the animal is shown by the lighter line. It can be seen that although the unit often takes GPS fixes (once every 15 minutes) it is unable to capture the high speed foray of this creature, effectively undersampling the actual distance travelled. Over this time interval, a total of 13 fixes have been taken.

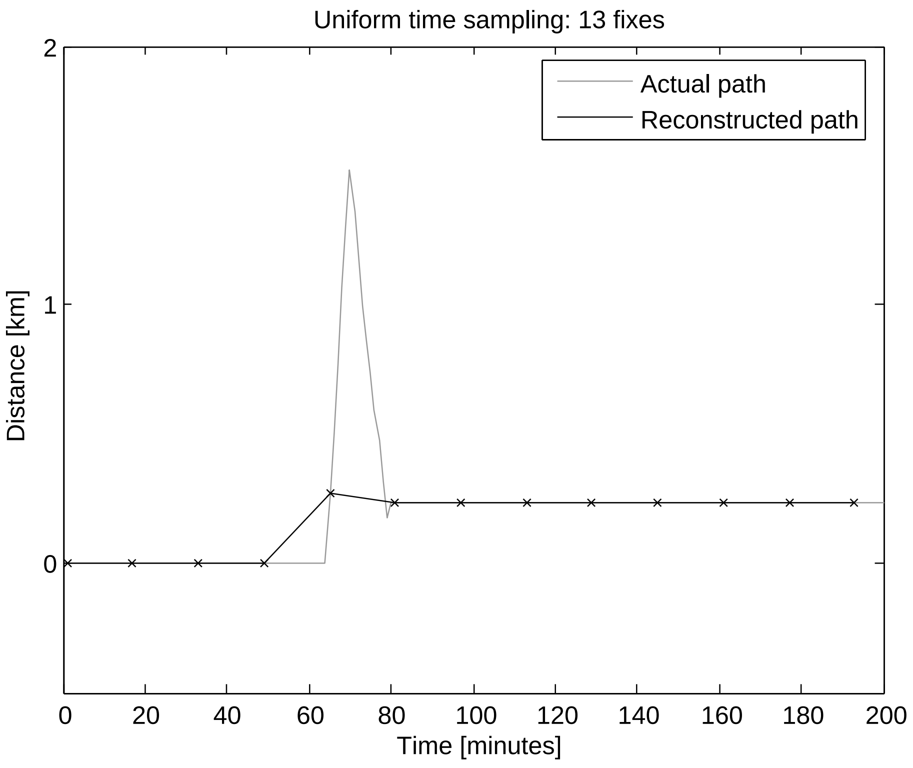


Figure 5.23: Reconstruction of path taken by a simulated host using uniform time sampling, with a sampling interval of 15 minutes. A total of 13 fixes were taken in this path reconstruction. The actual path taken is shown in grey, the reconstructed path overlaid in black.

Fig 5.24 shows a path reconstructed by ATDA, with a maximum time between samples of one hour. Whereas conventional sampling misses the high speed trip of the animal (which would be likely to be of great interest to a researcher) ATDA is able to accurately capture the path taken by the animal. Furthermore, only 6 GPS fixes were taken as compared to the 13 of the uniform time sampling method, resulting in a significant power saving.

5.8 Related work

The focus of the work in this chapter is different to that of pedestrian dead reckoning (PDR) systems. The purpose of PDR is to generate detailed motion paths when a GPS

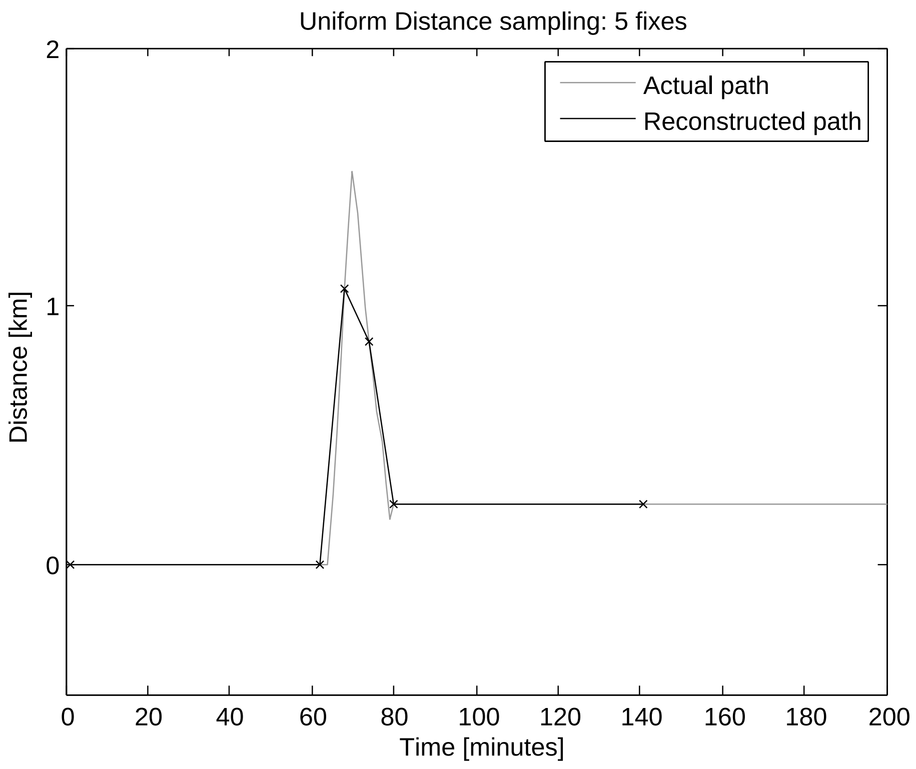


Figure 5.24: Reconstruction of path taken by a simulated host using ATDA, with a maximum time between fixes of 1 hour. A total of 6 fixes were taken in this path reconstruction. The actual path taken is shown in grey, the reconstructed path overlaid in black.

signal is not available (indoor environments in particular), whereas the aim of the uniform distance sampling approach is to intelligently schedule when to take GPS fixes. PDR systems operate continuously, as they need to constantly detect strides of the host [280]. In the presented methods for uniform distance sampling, the accelerometer is only powered up for a brief period of time in order to acquire an acceleration snapshot before returning to sleep mode. In addition, PDR typically uses two sensors (one for heading and another for pedometry), whereas a single uniaxial accelerometer is required for the method presented here, although accuracy could possibly be improved by using a triaxial accelerometer to compensate for collar tilt and orientation [280, 282]. Thus it can be seen that this work addresses a different goal to PDR. In fact, existing PDR systems can utilise the uniform sampling approach proposed in order to intelligently schedule times to acquire a GPS fix and calibrate the PDR model.

Acceleration sensors are used in wildlife research to determine animal activity and kinematics [306, 307]. Many tracking collars are already equipped with multi-axis acceleration sensors which measure neck tilt angles to determine if an animal is feeding or not [4, 5, 7, 6]. However, to date, no research has considered using this information to estimate the speed of the host in order to acquire GPS fixes at uniform distance intervals. Thus, to implement this method on traditional tracking collars will in most cases merely be a matter of upgrading the firmware, rather than involve a radical hardware change.

The work closest to ours is the recent (2007) classifier system presented by Schwager *et al.*[308]. In this, a magnetometer is used to assess the head tilt of a cow. A K-means classifier is trained using location data from the GPS receiver to determine whether the animal is active (i.e. moving) or inactive. The authors demonstrate that the classifier is robust and is able to correctly cluster head-tilt into the two classes of either active or inactive. The classification was performed off-line, although they mention that, in theory, it would be possible to perform on-line training. The work in this chapter demonstrates on-line training of data and was implemented on low cost microcontrollers. The authors mention that dual rate sampling would be desirable, with low rates when the animal is inactive, and high sampling rates when the animal is active, but do not provide details on practical implementation. The dual rate sampling approach assumes that the animal either moves at a constant speed or is stationary. In reality, an animal's speed can take on a wide range of values, and thus dual rate sampling, whilst better than uniform time sampling, still will not adequately reproduce high speed forays as it assigns the same sampling rate to both high and low speeds. In our work, gait analysis, rather than tilt, is used to estimate speed. This is more reliable, as there is not necessarily a correlation between tilt and speed, whereas locomotion inherently causes acceleration of the host. As a case in point, consider a human tracking device mounted on a belt. Whether the user is standing still or moving, the tracking device will still be oriented vertically and register the same angle of tilt. However, when the person is moving, the tracking device will experience a displacement and hence have a time varying acceleration signal. Their proposal is not a uniform distance sampling scheme, but rather a dual-rate method.

5.9 Discussion and Summary

In this chapter, it was demonstrated that the standard method of obtaining location fixes at periodic intervals is simplistic as it does not take into account the non-uniform motion of the animal/human that the device is attached to. It was shown how this results in redundant fixes whilst the animal is stationary and the possibility of under-sampling high speed information. A shift from the traditional fixed rate (uniform time) sampling to uniform distance sampling is advocated, using an adaptive dual sensor approach. This approach has the added benefit of generating a detailed speed-time profile which can be used to assess animal activity and behaviour on a finer scale. The adaptive scheduling system, ATDA, is simple to implement and adapts to the habits and behaviour of the host, with no calibration or pre-training required. Simulation results demonstrated that the tracking device was able to accurately recreate the path of the host, whilst using less energy. This strategy has the potential to vastly improve on the lifetime and the amount of information delivered by animal tracking collars and for devices worn or carried by people. The algorithms described were also programmed and implemented by the author on PIC microcontrollers, demonstrating their real world suitability and application.

It should be noted that the uniform distance sampling system has the added benefit of being able to store a fine-grained speed-time profile as a byproduct of predicting the animal's speed. The speed-time profile is of interest to wildlife researchers as it can be used to determine the animal's activity budget [309]. One method of acquiring animal activity profiles in use at present is to triangulate the positions of radio-tracked animals and record when their positions vary [30]. This is a highly labour intensive process and only captures large scale movements, thus limiting its usefulness and accuracy. The self-calibrating method presented in this chapter is able to automatically generate these profiles with no labour input.

To demonstrate the validity of the uniform distance sampling approach, gait analysis had to be performed, based on acceleration snapshots acquired from the animal's neck. Simple statistics (namely the acceleration range and jerk variance) were used to train the model and demonstrated good prediction performance. Some PDR work uses very sophisticated stride detection algorithms (for example the zero velocity update (ZUPT) method of Foxlin [280]), and it would be interesting to see if these techniques can be applied to the animal tracking scenario. This is proposed as a future direction of study.

Additionally, whilst this chapter concentrated on a using a uni-axial accelerometer, investigating the use of a tri-axial accelerometer is also proposed as a further area of research. By using a tri-axial accelerometer, acceleration signals can be made rotationally invariant, which will obviate issues introduced by collar rotation. In addition, by using three axes of acceleration data, it might also be possible to refine the speed estimates by further inference of animal motion. This could help boost accuracy especially for climbing and jumping animals. Another advantage of using tri-axial accelerometers is that other information such as tilt and roll of the tracking device could be logged.

In summary, the contributions of this chapter are:

- The introduction of the concept of uniform distance sampling for GPS tracking.
- The ability to create more detailed motion paths, whilst avoiding redundant fixes.
- The proposal of an adaptive threshold to be used on species where choosing a fixed distance threshold is difficult.
- The implementation and successful testing of the methods on a low cost 8 bit microcontroller.

- That this system retrofitted to existing tracking collars equipped with an accelerometer as a software upgrade.
- The generation of a detailed speed-time profile as a byproduct of the scheduling process.
- That the system can be used on people and animals with no restriction on placement.
- That uniform distance sampling can extend collar lifetime whilst generating more detailed motion paths.
- That by setting the maximum and minimum fix rates to be the same value, the collar can be configured to act as a uniform time sampling device.

To the best of our knowledge, this is the first presentation of a method that enables GPS fixes to be acquired at a rate proportional to the distance travelled, without requiring prior calibration or offline training.

6 Hardware and Firmware Design

6.1 Introduction

In order to test the concepts presented in the earlier chapters, custom wireless sensing devices were designed and constructed. This chapter discusses the structure of the hardware and software implementation to achieve the aims specified earlier, with a particular emphasis on modular design. Chapter 7 details some example deployments that were fabricated and deployed, as well as results that were obtained.

6.2 Hardware Variants

The details of the hardware implementation are discussed in this section. It should be noted that the prototypes were constructed merely to test the concepts of the prior chapters. Thus, devices from other manufacturers could be used, and where a particular device has been chosen, it is usually on the basis of local availability and cost rather than performance.

As the tracking devices were assembled by hand, packages had to be chosen that were suitable for manual soldering. This ruled out incorporating devices that were only available in leadless packages, such as QFN (Quad Flatpack, No leads) or BGA (Ball Grid Array). In addition, circuit boards were fabricated on a 2.5 axis milling machine which limited the allowable pitch and track spacing to 10 mil.

In spite of these restrictions, powerful and functional prototypes were designed and fabricated. In general, the devices selected were chosen to run on a 3.3 V supply. The details of the various devices are now elaborated upon.

6.2.1 Processors

The 8 bit PIC 18F family of microcontrollers was used [303]. These are RISC based processors that support self-programming of their internal FLASH memory and many low power operational modes. Built in peripherals include multi-channel 10 bit ADC, PWM, SPI, UART and timers. A table highlighting the features of the various microcontrollers used is shown in Table. 6.1.

Table 6.1: Features of the various PIC microcontrollers used in the implementation. Adapted from [303].

	Variant			
Feature	**PIC18LF452**	**PIC18LF2620**	**PIC18LF4620**	**PIC18F67J11**
Flash Memory (kB)	32	64	64	128
RAM (bytes)	1536	3986	3986	3930
Pins	44	28	44	64
Package	PDIP	SOIC	QFP	QFP
SPI	1	1	1	2
USART	1	1	1	2
Timers	3	4	4	5
A/D(10 bit)	8	10	13	11
Maximum Operating Frequency @ 3.3V	20MHz	20MHz	20MHz	48MHz
Supply current @ 32kHz (Run mode)	30	35	35	70

The PIC18LF452 variant was used in early prototypes to test transceiver operation and perform basic sensing tasks. The PIC18LF4620 and PIC18LF2620 microcontrollers are functionally equivalent and the only real difference is in the number of device pins, with the 2620 having 28 pins and the 4620 having 44 pins. It was found in practice that having only one USART was a drawback for the high end devices (in particular Base tags) that typically have a serially interfaced sensor (such as a GPS receiver or an RFID reader) and an external interface (for example, a PC connection). For this reason, the more functional PIC18F67J11 processor was also used. This unfortunately has a higher power consumption in standby mode due to the incorporation of an internal 1.8 V regulator. Other processors, such as the PIC18F6723 are pin and code compatible with the PIC18F67J11 and thus can be used as a drop-in replacement to reduce power consumption if required [303].

6.2.2 Memory Blocks

Two different memory options were implemented for non-volatile storage and buffering of sensor and network data, namely internal FLASH and external SD memory card. The PIC microcontrollers used can read and write their internal FLASH memory and thus data can be stored in their FLASH memories in addition to the program. The amount of memory available to the application depends on the size of the program memory and the device memory size[1]. For the microcontrollers used, the internal FLASH is accessed in 64 byte blocks, except for the PIC18F67J11 which accesses it in 512 byte blocks. The drawback of using internal FLASH memory is that a number of operations have to be taken to update the memory. The memory first needs to be read to a buffer, the particular block erased, the buffer modified with the new data and finally the buffer is written back to the FLASH. Thus, the PIC18F67J11 requires a 512 byte buffer to perform this operation, which is a large proportion of the total RAM available.

The other memory option implemented was mass storage using SD (Secure Digital) Memory cards, which are used in cameras and mobile phones for storing large amounts of data. As such, they are widely available and inexpensive (in the region of U$10 – U$30 depending on the size of the memory) [305]. They can be accessed using the proprietary MMC protocol, but they also support SPI access [310]. SPI access was chosen as the microcontroller has a built-in SPI controller. Data operations are performed in 512 byte blocks. However, unlike the internal FLASH memory, data can be written without first performing an erase, simplifying the operation. SD memory cards are available in sizes from 16 Mb to 32 Gb, although in this implementation, memory sizes of 2 Gb or less are handled[2]. Two different form factors were used, the standard size SD memory cards (32 mm $\times$ 24 mm) and the smaller Mini size (21 mm $\times$ 20 mm) [305]. Also available is the Micro form factor which is smaller yet (11 mm $\times$ 15 mm) [311]. These differ in size and shape but have an identical electrical interface, albeit with different pin connections.

6.2.3 Power Modules

Three different supply options were implemented. The first was direct connection to a 3V cell or battery. Whilst this is the simplest, it does suffer from the drawback that the voltage changes over the lifetime of the device, impacting on the transmission range of the transceiver and also complicating sensing of analog signals. This option was used for simple tags. Another method was linear regulation of a higher voltage supply. This was used for devices supplied from a large and stable voltage source, such as those used in a vehicle or powered from a computer. However, linear regulators are inefficient in comparison to switching regulators which can efficiently generate a stable output voltage from a wide range of input voltages. A switching regulator, (L6920DB [312]) was used to generate a stable supply when supplied from one or two 1.5 V cells. Switching

[1]Care must be taken to ensure that the data memory does not overlap program memory.

[2]This is because of issues regarding the block size of 512 bytes and the card ID register. The SD protocol only supports memory sizes up to 2 Gb, as it is essentially a FAT16 compatible system [305]. The SD-HC standard is a recent introduction which boosts capacity up to 32 Gb, but is not backwards compatible with the older SD memory standard[311].

regulators allow the device to be powered at a stable voltage even when the battery is nearly exhausted. The drawback of switching regulators is that they produce high frequency transients which can affect the operation of other devices. It was found in practice that the switching regulator increased the acquisition time of the GPS receiver, due to its high ripple. Extra filter capacitors had to be added to reduce the noise on the supply rail.

6.2.4 Network Transceiver

A short range transceiver was used as the network interface. The particular device selected was the multi-band NRF905 (Nordic Semiconductor [219]). This device comes in a 32 pin QFN package which makes hand assembly difficult. A commercially available daughterboard from Polygon Technologies packages all the required components on a PCB the size of a 14 pin DIP device [313]. This greatly simplified design as specialized design of matched impedance circuits could be avoided. The characteristics of the device are shown in Table 6.2.

Table 6.2: Features of the Nordic NRF905 radio transceiver. Adapted from [219].

Parameter	Value
Supply range	1.9V-3.6V
Transmitter power	-10dBm to +10dBm
Receiver sensitivity	-100dBm
Data rate	50kbps
Receive mode current	12.5mA
Transmit mode current @ +10dBm	30mA
Standby current	2.5uA

The NRF905 is a single chip communications interface that handles functions such as radio channel sampling, addressing and error checking. The microcontroller interfaces with the transceiver via the SPI interface and additional control lines that control modes of operation (transmit/receive and sleep). The main drawback of the NRF905 transceiver is that it does not allow received signal strength measurement (RSSI), precluding its use for signal strength estimation of distance [219]. The NRF905 can be used in the 433, 868 and 915 MHz frequency bands. For this implementation, 868 MHz was chosen as it resulted in acceptable length antennas. Quarter wave whip antennas, consisting of an 8.6 cm length of 0.5 mm diameter wire were used.

6.2.5 External Interface

Two external interfaces were implemented.

The first interface was an RS-232 serial interface to a PC. As level translation had to be performed between the 3 V microcontroller and the ±9 V required by the serial link, a level shifting chip was used. In early devices, the MAX233A was used and in later prototypes, the SP3232 which is 3 V compatible [314].

The other interface that was investigated was a long range UHF link, using AC4868 transceivers from AeroComm [315]. These have a transmission power of 250 mW, with a theoretical range of a few kilometres, but in practice did not perform well with the maximum distance reliably obtained being a few hundred metres. It is not clear why their range was substantially less than quoted in the datasheets.

Devices were also equipped with different colored LED's which were used to indicate operation in the field.

6.2.6 Sensing Modules

Acceleration sensors In an early prototype, the ADXL103 accelerometer was used to measure acceleration [290]. However, this is a uniaxial accelerometer which required mounting the device orthogonal to the main board to capture vertical acceleration. In later devices, the LIS3L02AS4 triaxial accelerometer was instead used [304]. This has a selectable range of either $\pm$ 2 g or $\pm$ 6 g and a bandwidth of 100 Hz. When active, it typically draws a current of 0.85 mA. This can be reduced by powering the device down between samples, reducing its duty cycle. Both these sensors output an analog voltage which was converted using the microcontroller's internal ADC to obtain a digitized value.

GPS receivers Two different types of GPS receiver were used, the LEA-4P and the NEO-4S, both from u-blox [122]. These are 16 channel GPS receivers with a maximum update rate of 4 Hz. The NEO-4S has a smaller form factor (16.0 $\times$ 12.2 mm vs 17.0 $\times$ 22.4 mm) and features high sensitivity tracking (tracking down to -158 dBm compared to -150 dBm). These interface to the microcontroller via an asynchronous serial port. This makes programming more complex than if they had an SPI interface as there are real time constraints on interaction between the microcontroller and the receiver. Parameters for the GPS receivers are shown in Table 6.3. Also shown is the new generation of u-blox receivers, the LEA-5S, which is a 50 channel receiver that is Gallileo compatible [122]. This device is a drop-in replacement for the LEA-4P, and the NEO-5S a replacement for the NEO-4S. These will further improve the performance of GPS signal acquisition.

Table 6.3: Parameters of the various GPS modules. Adapted from [122]

	LEA-4P	NEO-4S	LEA-5S
Number of Channels	16	16	50
Maximum Update Rate	4Hz	4Hz	4Hz
Cold Start	36s	36s	29s
Warm Start	33s	33s	29s
Hot Start	3.5s	3.5s	1s
Acquisition Sensitivity	-140dBm	-148dBm	-160dBm
Tracking Sensitivity	-148dBm	-158dBm	-160dBm

Environmental Sensors Temperature was measured using an STLM20 temperature sensor, which has an accuracy of ±1.5 °C [316]. Ambient light level was monitored using a light dependent resistor. Both these analog signals were sampled and acquired using the microcontroller's on-board ADC.

RFID reader A long range (70 cm) RFID reader, capable of detecting ISO11785 transponders was used [137]. Due to supplier issues, two different variants had to be interfaced, one with an RS-232 link and the other with an RS-422 interface. The readers are controlled by the microcontroller's serial port and can be operated either in continuous or polled mode. The power consumption of the readers is very high (average current consumption with 10 scans/second is approximately 1.5 A when operated from a 12 V supply), requiring a large source of energy to provide continual operation. In order that direction of travel could also be determined, an antenna change-over switch was designed which was controlled via two PWM channels of the microcontroller.

6.3 Firmware Discussion

Software for the tests was written in Microchip C18, Student Edition, which is a free ANSI C compiler [303]. Code was documented using Doxygen [317]. Existing libraries were used where possible, but most functionality had to be introduced as and when required. Rather than using a dedicated operating system (OS), such as TinyOS, a simple hierarchical finite-state-machine formulation was used. Although there are obvious benefits to having an OS, especially in terms of code reuse with other projects, this was the simplest approach to test the concepts presented in the prior chapters. The state machine was implemented using function pointers, as this provided the ability for a central calling function to alter program flow if so desired.

The main components of the system are shown in Fig. 6.1. Depending on the configuration (whether the node is to be a Marker, Spotter and so forth), different nodes have different components present. For example, a simple Marker would not have an external interface handler or a sensor handler block, whereas a full Base device would have all the blocks present. These blocks can be regarded as 'super-states' in the state machine based system. Each block has an internal state machine that is used to control operation. Note that the handler blocks do not communicate with one another directly, but act in isolation from one another. Linking all the blocks together is the Filesystem which provides access to the non-volatile memory (whether internal FLASH or external SD memory card). The operation of the Filesystem is first discussed, followed by the functionality provided by each handler. For the sake of brevity, flow diagrams have been simplified so as to highlight the important features, rather than provide detailed operation of each state.

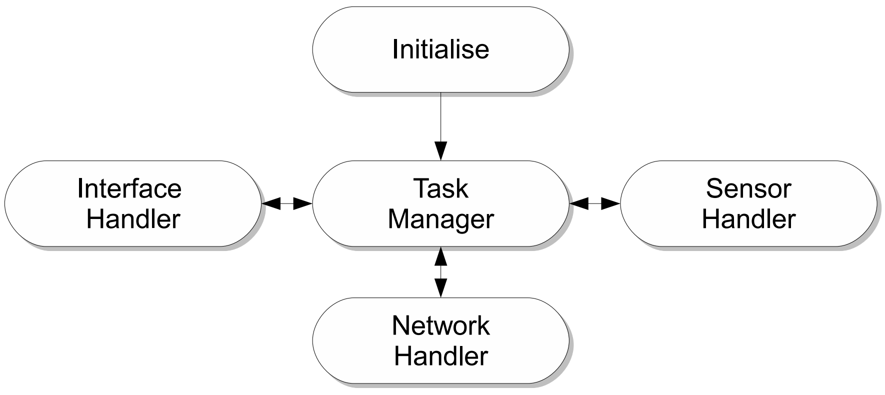


Figure 6.1: System flow diagram showing principal components

6.3.1 The Filesystem

The only shared space in the state machine is the Filesystem which the various handlers can write to and read from. The Filesystem is configured as a number of large blocks[3] or bundles of data. Each handler interacts with the Filesystem through an equally sized shadow buffer. When the shadow buffer is full, it is placed in the external memory. This lets the handlers fill up the shadow buffer on a byte-by-byte basis without the need for frequent reads and writes to the external memory and avoids issues with buffer locking. In addition, it means that any data present in the Filesystem is a full bundle in size.

The Filesystem is configured as a circular buffer that has a write pointer (indicating the next available location for storing a bundle) and a read pointer (indicating the next avail-

[3]In this implementation, 512 bytes in size, as this is the same page size as for the SD memory card

able location to read from). As an example of operation, the sensor handler generates data and 'stacks' it on top of the circular buffer (once it has filled its shadow buffer with data). The network handler pulls data from the bottom of the circular buffer and sends it, advancing the read pointer upwards towards the write pointer upon successful transfer. When the read and write pointers have the same value, the network handler would stop sending data. In a real situation, a network capable device would also stack data to be routed on top of the circular buffer. This is shown in Fig. 6.2 which demonstrates the handlers placing and retrieving data from the circular buffer. Note that the data, whether it is sensor data or network data from other nodes is interleaved on the buffer. In addition, it can be seen that the network and sensor handlers have no knowledge of the actions of one another, which simplifies the structure of the program.

It is possible to address individual blocks in a random access manner. However, the circular buffer is a simple programming concept with regards to asynchronous access to the external memory, as the handlers merely call push and pop functions which place and remove data from the next available location, without knowing the address of the data to be accessed. This leads to modularity, as the handlers do not need to know the size (or even the physical type) of the memory. It should be mentioned that the Filesystem provides a transparent interface to the memory chosen, whether it be FLASH or SD memory card. Upon power-up, the card and its size is detected. If no card is present, the device is configured to use its internal FLASH memory for data storage.

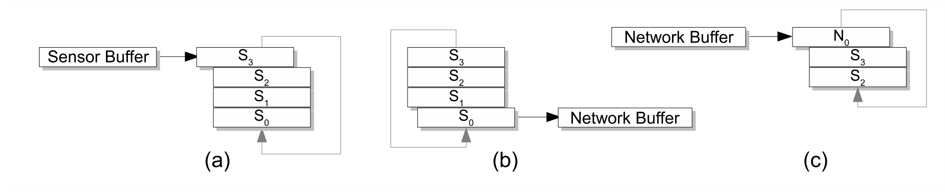


Figure 6.2: Pushing and pulling data off the circular buffer. In (a), the Sensor Handler places a completed sensor handler block onto the top of the buffer. In (b), the Network Handler uploads data to another device and sequentially pulls blocks from the bottom of the buffer. Lastly in (c), the Network Handler receives information from another node and places this at the top of the buffer. It can be seen that network and sensor data are interleaved in the circular buffer.

The functions undertaken by each block are now examined.

6.3.2 Initialise block

This module, common to all nodes, initializes the device and any attached peripherals. Depending on the processor variant, different functions and pin connections will be applicable. Some 'intelligent' peripherals such as GPS receivers and SD memory cards can be polled to see if they are connected and working correctly. If no reply is received, then functions associated with the devices are disabled. In this way, the tracking device can automatically configure itself according to the attached peripherals.

6.3.3 Task Manager

The Task Manager is the main system controller that decides what action to undertake, and also manages role adaption (refer to Section 3.6) in accordance with remaining device energy. The function handlers communicate through the Task Manager by passing variables indicating status and operation. The Task Manager also schedules sleep and wake-up times, so as to reduce the overall power consumption of the device. When a handler block passes control back to the Task Manager, they schedule a wake-up. There is obviously an issue of priority, as to which block should be called if there is a simultaneous wake-up request. Currently, this is set so the Sensor Handler has the highest priority, followed by the Network Handler and lastly the Interface Handler.

The Task Manager runs as a state machine, which cycles through the various handlers that need to be executed. Thus, it acts as a primitive operating system, controlling system flow. Each of the other handlers are re-entrant state machines. Thus, when control is passed to a handler by the task manager (i.e. the function associated with the particular state machine handler is executed) it will execute the next state in its state list. Depending on the response from the state, it can either return control immediately to the task manager such that another handler can be called, or it can block and immediately execute the next state in the same handler. This means that handlers can handle events in real-time if necessary. The handlers thus all execute as asynchronous re-entrant state machines.

If there is a need for precisely specified sampling intervals, this can be undertaken in one of two ways. If the number of samples to be taken is relatively small (such as acquiring a 1.2 s accelerometer snapshot every 30 s for the uniform distance sampling scheme in Chapter 5) the Sensor Handler can simply run the sampling routine as a blocking function, not yielding control back to the Task Manager until the sampling has completed. The other way is to use a timer-driven interrupt handler – the drawback of this is that this can interfere with other processes such as network transfer which have strict timing requirements.

The Task Manager is responsible for controlling the system behaviour. It does this through setting variables that are common to both the individual handler and the manager. For example, the Network Handler can share its energy level, as determined by the ASH ranking scheme (refer to Chapter 4), with the Task Manager. Based on the ranked energy level, the Task Manager can alter the behaviour of the Sensor Handler by reducing the rate of acquiring GPS location fixes, by decreasing the value of a variable shared by both the Task Manager and the Sensor Handler. In this way, handlers indirectly influence the behaviour of each other through the Task Manager. However, the Task Manager has ultimate control over the behaviour of the handlers.

6.3.4 Sensor Handler

The Sensor Handler determines what sensing action to perform and controls the filling of the sensor shadow buffer. An example of the states required to undertake the uniform distance sampling method of Chapter 5 is shown in Fig. 6.3. Also included is a temperature sensing block. The Sensor Handler block decides what action to undertake (whether to sense the temperature, obtain an acceleration snapshot or relinquish control to the Task Manager), based on the sensing requirements and the information shared with the Task Manager. Note that the Sensor Handler is always the terminal state for the sensing components. This enforces modularity, as any arbitrary sensing block commences and terminates on the Sensor Handler super-state.

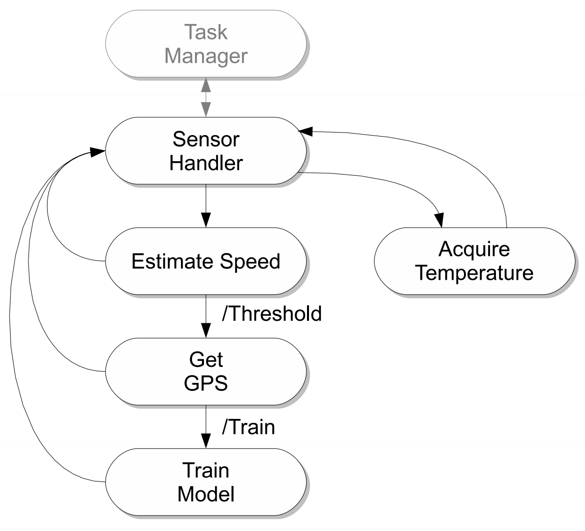


Figure 6.3: Sensor handler states: Uniform distance sampling GPS with additional temperature sensor

6.3.5 Network Handler

This handler is responsible for transferring data through the network, as well for sending and receiving beacons. It is in this handler that the main distinction between the classes of tracking devices discussed in Chapter 3 is effected, and that the modularity of the design approach is made most apparent. The Network Handler decides what action should be taken, based on the parameters shared with the Task Manager. For example, to reduce energy consumption, full network capability can be disabled by the Task Manager, reducing the size of the state universe. The state machines for the various functions are discussed in increasing order of complexity, commencing with simple marking, followed by spotting and lastly, full network operation.

First however, a brief overview of the packet format is provided in Fig. 6.4 which shows the various packets present in the system. The numbers on top of each field in the packet represent the number of bytes present in each field. Every type of packet in the network starts with an ID field. The Beacon packet is 8 bytes long, and is used for nodes to spot one another and arrange data transactions (Chapter 3) and also so that ASH parameters can be updated (Chapter 4). Thus, it contains a timestamp, used for beacon records, the data channel to jump to, as well as the ASH attributes and rank. Beacon packets are kept short and contain no data so as to not clutter the Beacon channel.

The Bid packet is sent in response to a Beacon packet and contains the sender's ID and its ASH attribute and rank so the receiver can update its ASH parameters. If a node is successful in its bid, an Accept message will be sent back, merely containing the ID of the winning node. Once this has been received, nodes can start sending their data in 512 byte blocks (which is the same size as a block in the SD card, thus packets can be shifted directly from memory into the transceiver without having to be repacked). As the maximum packet size of the Nordic transceiver is 32 bytes, 16 packets have to be sent. The data packet contains fields which indicate the type of data being sent, its sequence as well as a longer timestamp. Nodes can append extra information into the packet such as hop-path if desired. The packets are protected by a checksum which ensures that they are reassembled in the correct order. Once a Data packet has been succesfully received, an ACK (acknowledge) packet will be transmitted back to the sender.

In terms of data overhead, this network is designed for transferring relatively large amounts (a few kbytes at a time) of delay tolerant data, comprising a number of bundle blocks. Within each bundle, 14 bytes are used for flow identification and flow control, thus the overhead of this protocol is $14/512 = 2.7\%$.

Marking

A device that marks periodically transmits a beacon containing its ID and other salient parameters such as recent sensor data. As such, its state machine is very simple, consisting of a single state in addition to the parent handler. The Network Handler decides when to trigger this state, and the procedure consists simply of loading the data to be sent into the transceiver and waiting for it to be sent, before relinquishing control back to the Network Handler. This is shown in Fig. 6.5.

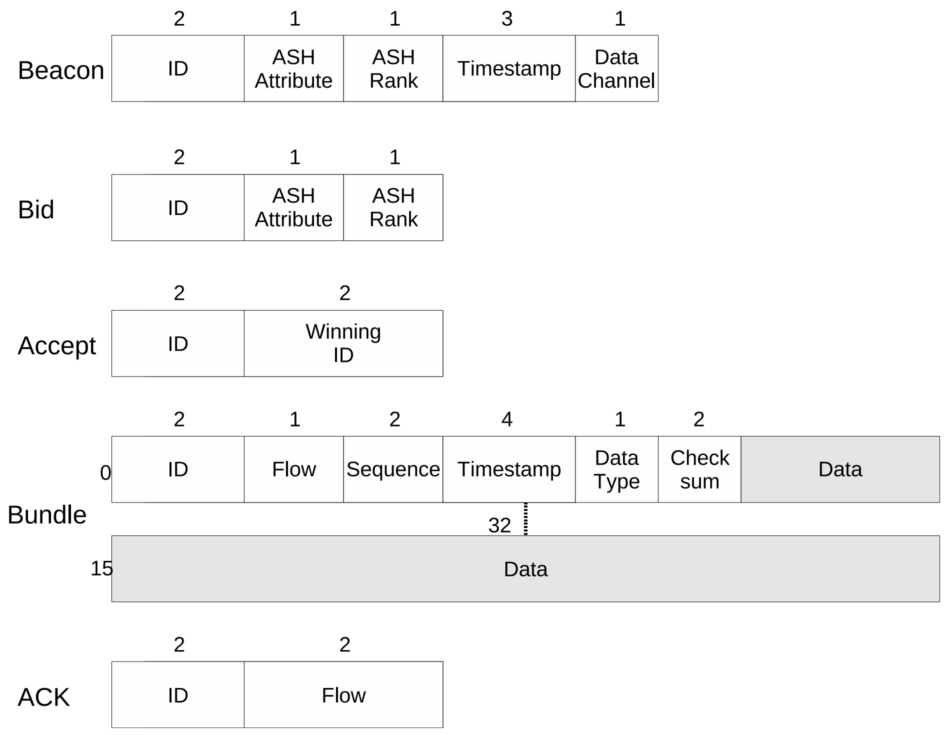


Figure 6.4: Packet alignment and structure for the various types of packets in the network.

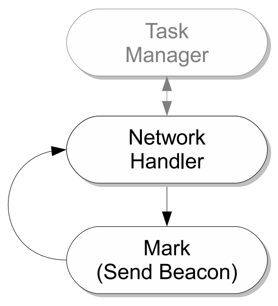


Figure 6.5: Network Handler for a Marker class tracking device.

6.3.6 Spotting

Spotting refers to the action of scanning the beacon channel for a length of time and recording the ID's of the observed nodes. Spotting is also used to discover suitable hosts for uploading data to and for updating ASH parameters. The possible states for a Spotter class node are shown in Fig. 6.6. Like a Marker class device, the Network Handler can transmit beacons as well. At random intervals, the device will enter spotting mode and scan the beacon channel. If a beacon is heard, the ID is recorded and then the parameters of the received beacon are used to update network parameters (such as connectivity) and ASH parameters (such as energy rank). The ASH parameters are shared with the Task Manager. If, at the end of the Spotting window there are no suitable devices in range to upload to, the system transitions to the Network Handler. Conversely, if a suitable host is found, the Spotter node attempts to upload its stored data. It does this by examining the host's beacon and determining the channel that will be used for data transfer. A short (4 byte) 'bid' packet is sent which indicates that the device has data to upload. If the host responds with an 'accept' packet, the two devices exchange data. Data is sent in 512 byte bundles, which corresponds to 16 32-byte packets sent by the Nordic transceiver. Data is protected at the link layer with a 16 bit CRC and at the network layer with a 16 bit checksum that ensures correct reassembly of the packets. At the end of this transfer, the host responds with an acknowledgment which indicates that the data was received without error. The transmitting device can now send more data if required. Note that a Spotter class tracking device never receives data and only injects data into the network.

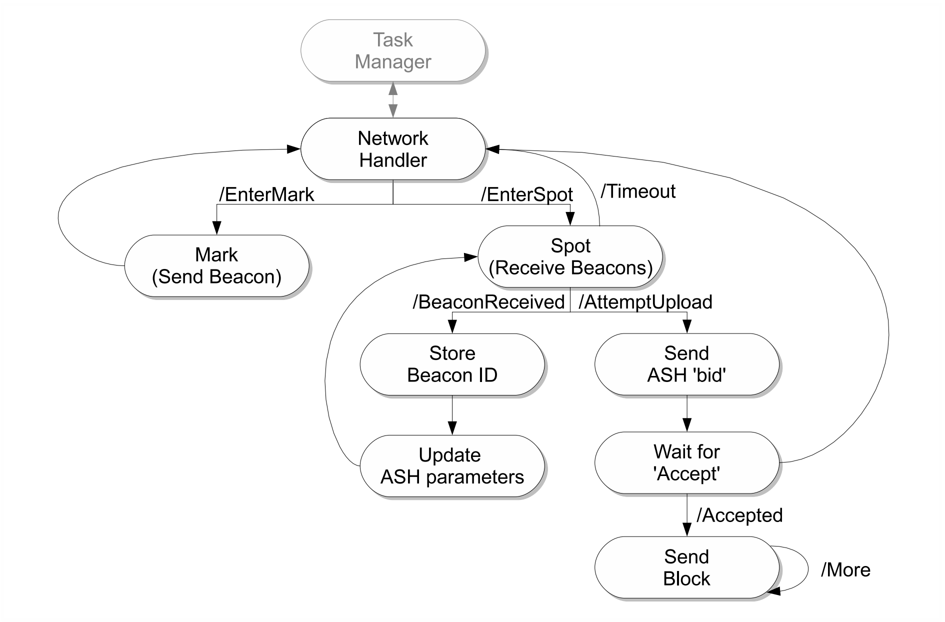


Figure 6.6: Simplified Network Handler for a Spotter class tracking device.

Full network functionality

A full network function device, such as a Pack Tag, adds an extra group of states which are responsible for receiving data in addition to all the other network functions. These devices set a bit in their Marker packets which indicates that they are capable of receiving data, as well as the channel that data transfer should take place on. Thus, after sending a Marker, they jump to the data channel and wait for a brief time to receive a ’bid’ packet. An ‘accept’ message is then sent back, which informs the other device that it may commence uploading. In the case of multiple ‘accept’ messages, bids are randomly accepted, with a bias towards accepting bids from weak nodes. The simplified state flow for a full network capable device is shown in Fig. 6.7.

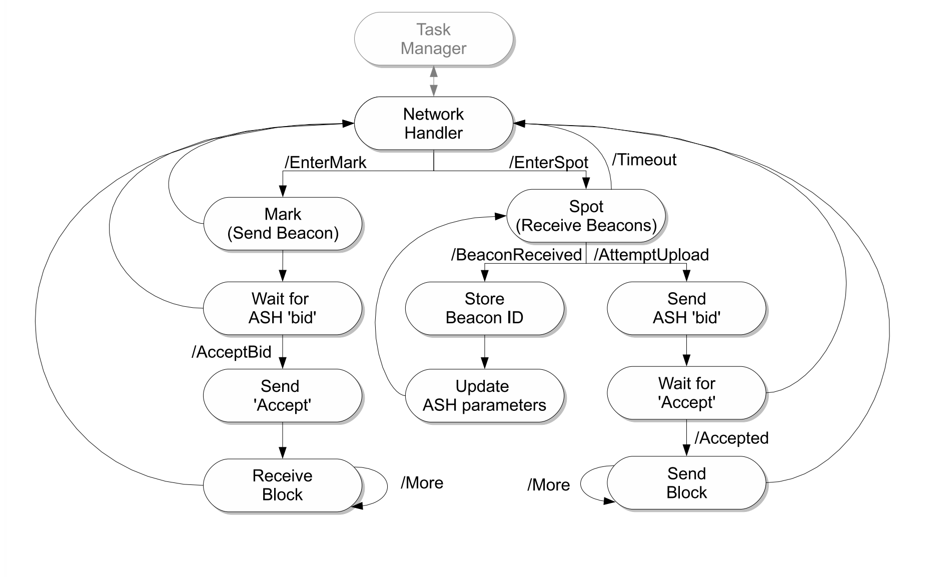


Figure 6.7: Simplified Network Handler for a full function network device (Pack)

Modularity

The state flow diagrams illustrate that simpler nodes have a subset of the functionality of more complex nodes. This however does not mean that multiple sets of code have to be written, with each class of device having a different network handler. Rather, one set of code is written for the most complex device, and simpler devices are implemented by restricting possible state transitions. Restriction can be done at the compiler level, through conditional compiler switches, which eliminate the unneeded code from compilation. For example, including the code for Spotting in a Marker class node would lead to unnecessary code bloat, as these states are unreachable. Thus, the code can be removed from compilation by setting a conditional compiler switch in the code.

State space restriction can also be implemented dynamically by the Task Manager. The Task Manager can select the class of the node according to information such as the available energy and the ASH level. This information is shared with the Network Handler which accordingly prevents certain states from being reached, so as to restrict functionality. Thus, the dynamic role altering method from Section 3.6 is accomplished simply by disabling certain states.

Hence it can be seen that modularity is enforced and easily implemented, with multiple classes of devices sharing the same code.

6.3.7 Interface Handler

This handler controls any external interfaces (such as a GSM modem or serial port) that are attached to the tracking device. Depending on the interface and the features supported, this can be a fairly complex component of the system design. Primarily the interface is used to upload data from the device, but it can also be used to alter the operation of the device, for example by altering the sampling rate of the GPS receiver.

An example of an Interface Handler is shown in Fig. 6.8. This handler is used to interact with the device using a PC, for example a laptop in the field. A serial interface is used as the communications interface. Two functions are supported, Upload Mode and Spot Mode. The user enters each mode by sending a single character (for example 'U' to start an upload and 'S' to scan the beacon channel), and can also abort the current operation by sending a break character (e.g. 'X'). Any other commands will be ignored. If the user enters Upload Mode, the contents of the circular buffer are sequentially sent to the PC, where they can be stored for further analysis. In Spot mode, the device enters a special mode where it continually listens to the beacon channel and echoes any received beacon packets to the PC. This is useful to determine what devices are within range at any point in time. Each mode can be exited or terminated by sending the break character, at which point control will be passed back to the Task Manager.

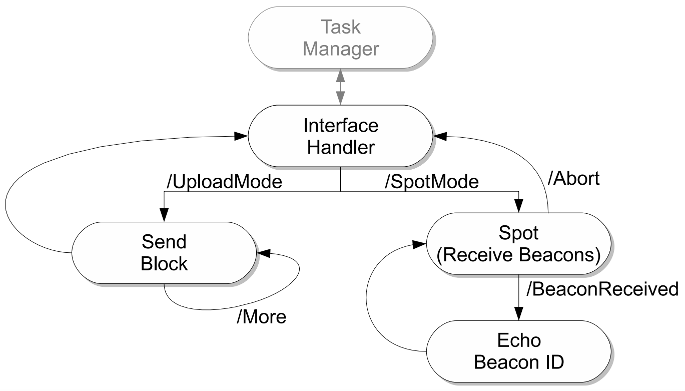


Figure 6.8: Simplified Interface Handler for a serial interface to a PC

6.4 Typical energy consumption of the various device classes

Table 6.4 shows the typical energy consumption for different classes of nodes. Also shown is the average battery weight required for a year of operation and the component cost of the device. In calculating the battery weight, it has been assumed that an amp-hour of battery capacity weighs 12g[4]. To calculate the component cost of a device, it has been assumed that a transceiver costs U$12, an SD memory card U$15, a microcontroller U$5, the GPS receiver and antenna U$80, an accelerometer U$8 and a GSM module U$150. In addition, battery cost is assumed to be U$2 per Ah. Board costs, packaging and collar material are excluded from this analysis as they are common to all classes.

For the loggers, it is assumed that they are capturing acceleration samples from a triaxial accelerometer at a rate of 32 Hz per axis and storing them on an SD memory card. The Active Logger also opportunistically uploads a summary of the acquired data. Note that uploading a small amount of data does not greatly impact the current consumption of the device. Thus, the expected lifetime for the Active Logger is 42 days as opposed to the Archival Logger with 45 days. This shows that periodic transfer of summary statistics during the course of the study provides a means of obtaining data even if the tag cannot be retrieved to download its entire dataset, without dramatically decreasing its lifetime.

The Marker node transmits a beacon (on air time is 6 ms) every 3 s. This results in an expected lifetime of over a year per Ah of battery capacity. For the Spotter node, it beacons like the Marker node, but also detects nodes within its radio range every 5 minutes. It also injects stored data records into the network when possible. The action of spotting dramatically increases current consumption, by a factor of 4, consequently reducing its lifetime in comparison to a Marker tag.

Pack tags also perform routing tasks in addition to Marking and Spotting. This increases their current consumption, resulting in a typical lifetime of 54 days per Ah of battery capacity. To demonstrate the effect of a GPS receiver on a tracking device's power consumption, the current drain of a GPS equipped Pack tag is shown. It is assumed that fixes are taken every 15 minutes, and average acquisition time is 20 s. The GPS receiver is responsible for over half of the device's energy budget, reducing its expected lifetime to 24 days per Ah. The current consumption of a GSM enabled Base tag is also shown. Although the GSM module is only active for 0.5% of the time, the high current consumption results in a lifetime of only 17 days per Ah of battery capacity.

In terms of device weight, it can be seen that there is a strong correlation between

[4]As justification for this: a CR123A Lithium battery weighs 17g and provides 1400mAh @ 3V [79]

functionality and weight. The simplest device, a Marker tag, only requires a 10 g battery for a year of operation whereas a GPS enabled Pack tag requires a 169 g battery – more than an order of magnitude difference. The device costs follow a similar pattern, with a GPS enabled Pack device costing U$140 compared to a Marker which only costs U$19.

6.5 Summary

A modular approach to designing the tracking devices has been followed through this chapter. Multiple options are available both in terms of hardware and software. The use of SD memory cards for mass storage provides an inexpensive means of incorporating vast amounts of memory storage. The main advantage of the SD memory card is that if the tracking device is retrieved at the end of the study, all the data that was generated and relayed through the device can be recovered. SD memory cards are small and power efficient if accessed in a buffered block fashion as opposed to bytewise.

The use of the hierarchical state machine simplifies programming practice, as procedures return control to the parent super-state upon completion. This makes altering the class and functionality provided by a device simple. However, guaranteeing real time performance using this approach can be difficult. Simple real time deadlines can easily be implemented using timer driven interrupts which fill buffers with data to be handled at a later stage by the requisite handler. However, more complex interactions may require a departure from the hierarchical state machine to an operating system specifically designed for real time performance. If this is done, the modularity of this approach should be carried forward into the new design.

It was shown how the current consumed by the devices varied based on the class and functionality. For example, attaching a GPS receiver to a Pack device doubles the device weight and triples the device cost. These figures demonstrate why a heterogeneous system design is necessary – increased functionality dramatically increases power consumption, device weight and cost. By including simple devices in the network, they can be made lightweight and use the functionality of more complex devices, allowing smaller animals to be monitored by the same system.

Table 6.4: Typical current consumption of the various classes of device. Also shown is the expected lifetime per Ah of battery capacity.

Tag Type	Operation	Duty cycle	Current (mA)	Mean Current (mA)	Lifetime per Ah (days)	Battery weight per year of operation (g)	Typical Device Cost
Archival Logger	Sample	99.8%	0.8	0.92	45	88	U$43
	Store	0.2%	60				
Active Logger	Sample	99.5%	0.8	0.99	42	95	U$56
	Store	0.2%	60				
	Injection	0.3%	26				
Marker	Sleep	99.8%	0.04	0.11	386	10	U$19
	Beacon Tx	0.2%	34				
Spotter	Sleep	97.7%	0.04	0.43	97	41	U$24
	Beacon Tx	0.2%	34				
	Beacon Rx	2.0%	15				
	Injection	0.1%	26				
Pack	Sleep	95.8%	0.04	0.93	45	88	U$47
	Beacon Tx	0.2%	34				
	Beacon Rx	2.0%	15				
	Route	2.0%	26				
GPS Enabled Pack	Sleep	93.6%	0.04	1.77	24	169	U$140
	Beacon Tx	0.2%	34				
	Beacon Rx	2.0%	15				
	Route	2.0%	26				
	GPS fix	2.2%	38				
GSM Enabled Base Tag	Sleep	95.3%	0.04	2.43	17	232	U$221
	Beacon Tx	0.2%	34				
	Beacon Rx	2.0%	15				
	Route	2.0%	26				
	GSM upload	0.5%	300				

7 Real World Testing and Results

7.1 Introduction

This chapter discusses real world fabrication and tests of the various classes of tracking devices[1]. Tests are roughly arranged in order of tracking device class.

7.2 Spotter Testing

Three Spotter class tracking devices were fabricated and sent to the research centre at Ongava Game Reserve, Namibia [318], for tests in the field of their suitability and robustness. A Base station was also fabricated which was used to download data from the tracking devices.

7.2.1 System Overview and Components

The system configuration of the two different classes of nodes is shown in Fig. 7.1. The Spotter class nodes were PIC18LF2620 microcontrollers and were powered by a CR123 2 Ah Lithium battery, which provided a 3 V supply. No regulation was used as it was found that the tags could reliably operate down to 2.2 V. The Base tag was a PIC18LF4620 microcontroller which was powered by a replaceable 9 V PP3 cell, regulated to 3.3 V using a linear regulator. A serial communications link provided an interface between the Base tag and the PC.

7.2.2 System Construction and Operation

To construct the Spotter nodes, the PIC18LF2620 microcontroller was attached to the transceiver board using superglue. Thin wires were used to connect the microcontroller

[1]All the devices shown in this chapter were designed, assembled (by hand) and tested by the author.

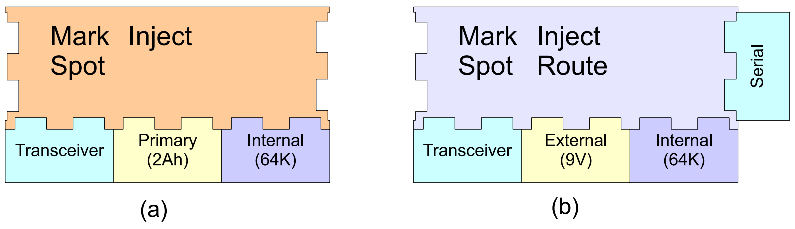


Figure 7.1: System diagram of the Spotter test. The three spotter nodes had the system diagram shown in (a), which consisted of a 2 Ah power source, a transceiver and used the internal memory to store spotter records. The Base station, which extracted data from the Spotter tags is shown in (b). This is virtually identical to (a), with the exception of a serial link which was used to communicate with the device.

pins to the corresponding pins on the transceiver board. Temporary wires were also attached which were used to program the devices and their unique ID's. A 9 cm length of 0.5 mm diameter steel wire was used as a quarter wave whip antenna. Once the devices had been programmed and tested, the CR123 battery was attached and the units sealed using rapid-setting epoxy resin. This provided a rigid and robust encapsulation for the devices, ensuring that they were weatherproof.

The Base tag was very similar to the Spotter nodes, except it used a PIC18LF4620 microcontroller and was powered via an LM317L linear regulator which provided a stable 3.3 V supply. The device interfaced to a serial ('COM') port on a PC via an RS-232 transceiver. The Spotter and Base devices are shown in Fig. 7.2.

The Base tag was controlled from the PC using a simple text driven command interface via a terminal program such as HyperTerminal. To obtain a list and explanation of the commands supported, the user can press '?' which will result in the following being displayed in the terminal program:

```
* Wireless Tag Upload Utility
* Version 1.00 build 2007/06/22
* (c) A.C.Markham 2007
* Press '?' to get a list of commands
* Command Set:
* 'U' : Upload Mode
* 'D' : Dump entire memory (Error Recovery)
* 'E' : Enter passive listen mode
* 'A' : Enter active transfer mode (normal mode)
* 'W' : Wake nodes up
* 'S' : Shut nodes down (low power mode)
```

To upload data from the tag, the Upload Mode is entered which dumps the contents of the circular buffer to the terminal program. In the event of an error, the entire memory can be uploaded which can be useful to recover stored data. In passive listen mode, the Base tag scans the channel and echoes any received beacons to the terminal program. In this mode, the device is a passive network observer and thus cannot obtain data from the Spotters. To return to normal mode, the 'A' key is pressed.

The Spotter units support a special low power sleep mode that is useful for long term storage. In this mode, the devices wake up every minute for 10 ms to scan the beacon channel. If a wakeup message is not received, the devices go back to sleep. With such a low duty-cycle, the power consumption is neglible (average current approximately 40 μA). If a wakeup message is received, then the devices enter full power mode. This avoids the use of magnets and reed switches that are used in commercial tracking devices to power devices down [5]. The Base tag can thus be used to place the devices in low power sleep mode and to wake them up as well.

Data is transferred in ASCII encoded hexadecimal, which is subsequently parsed to extract each node's Spotter records. The Spotter devices do not have a real time clock and thus each device has its own free running clock. Synchronization to real time is accomplished by the Base station, which reconstructs the Spotter records.

7.2.3 Results

One Spotter was strapped to a vehicle. The other two Spotters were colocated at a stationary location, close to where the vehicle would normally park. The Base tag was periodically powered up to download the contact logs from the Spotters. Data from one of the stationary Spotter devices is shown in Fig. 7.3

This test demonstrates that the Spotter tags are able to detect the proximity of one another. The colocated tag was correctly detected as being always within range, shown by the continuous line. The contact log of the vehicle based Spotter shows when the vehicle was within proximity. As the vehicle was normally parked close to the stationary Spotter nodes, the absence of a contact indicates times when the vehicle was being used. The Spotter also records the presence of the Base tag when it was powered up. Similar contact logs were retrieved from the other Spotter devices used in the test, and from the Base tag when it was powered.

7.2.4 Summary

This test demonstrated that Spotter class tags are able to detect the presence of one another, and also other classes of nodes (the Base tag). They are able to upload their

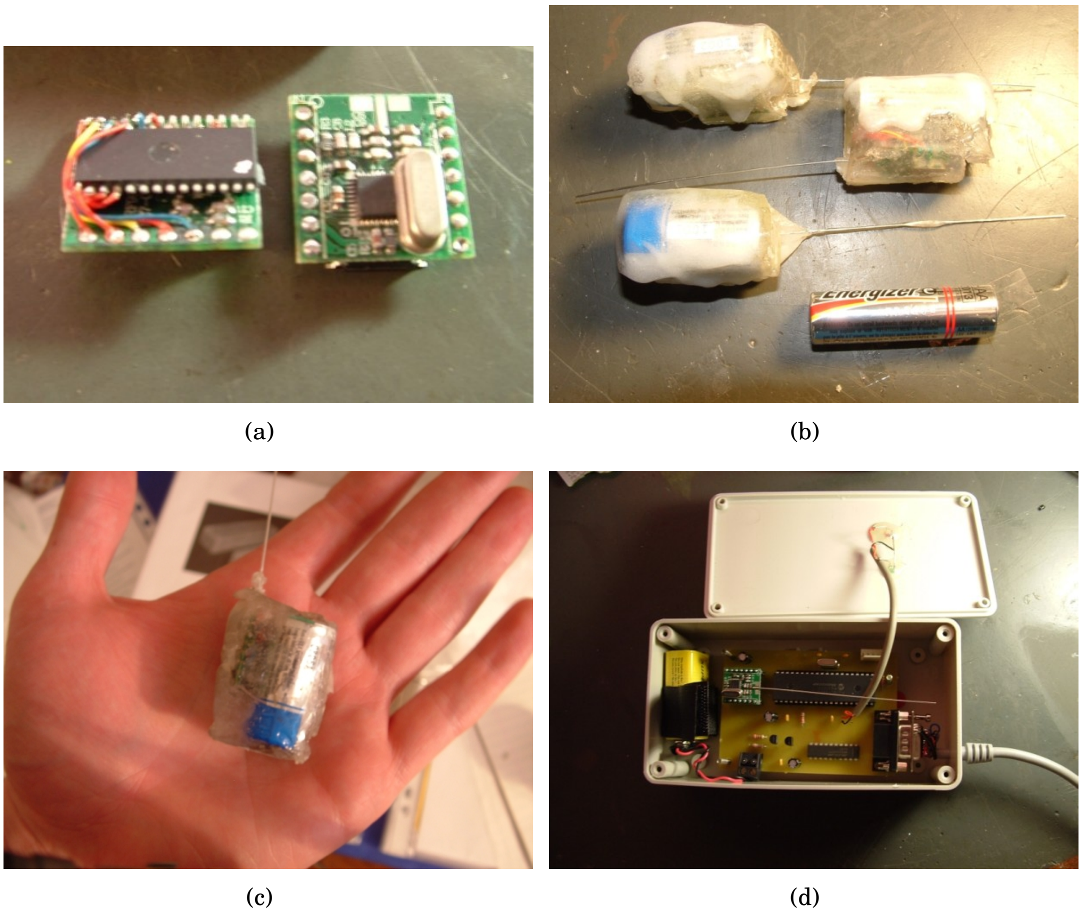


Figure 7.2: Construction of Spotter Class devices. (a) shows the construction of a Spotter by attaching thin wires between the transceiver board and the microcontroller. (b) shows three assembled Spotter tags encased in epoxy resin. (c) Illustrating the size of the Spotter tag. (d) The Base tag in a weatherproof box. The 9 V battery (left of image) is held in place with a velcro strip and the RS232 interface is on the right. LED's on the top of the box indicate device operation.

contact logs to higher class tags, in this case a Base tag. When instructed by the user, the contact log was uploaded to the PC, where it could be processed to obtain detailed proximity information from the nodes. The user interface, being terminal based, was not user friendly, and data analysis had to be done off-site, as a script had not been written to process the raw data.

7.3 Range Testing

In order to test the radio range of the devices in a realistic setting, an experiment was conducted in Rhodes Memorial, Table Mountain National Park, close to the University of Cape Town. This consisted of a stationary Marker class node periodically emitting its

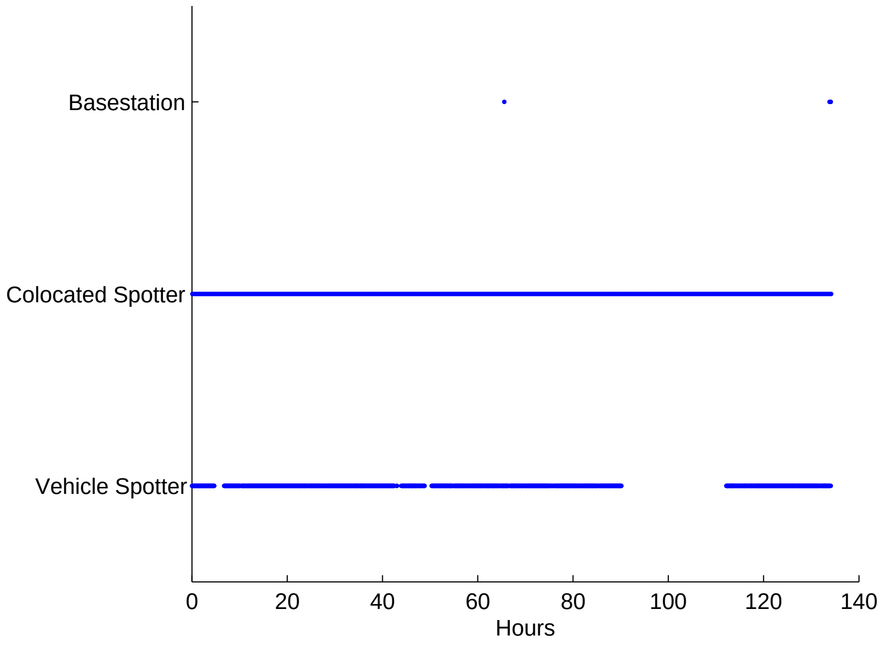


Figure 7.3: Contact logs from the Spotter test, downloaded from one of the stationary nodes. This shows that the Spotter correctly recorded the presence of the colocated node, shown by a continuous line. It also detected when the vehicle was parked close by. When the Base tag was powered up to retrieve the data from the Spotters, the Spotter also recorded the presence of the Base tag.

ID and a human carried Pack class tag which logged GPS locations and recorded when the Marker node was within range. At the end of the test, data was uploaded to a Base station for analysis on a PC.

7.3.1 System Overview and Components

The system overview is shown in Fig. 7.4. The Marker node consisted of a PIC18LF2620 microcontroller and a transceiver and was powered by a CR2450 3 V Lithium coin cell. The Pack tag used a PIC18LF4620 microcontroller, a 256 Mb SD card and a LEA-4P GPS module with patch antenna. This unit was powered by 2 AA batteries. The Base tag was a PIC18LF4620 microcontroller with a serial interface. The Base tag also had a 256 Mb SD memory card for data storage and was parasitically powered via a USB connector.

Photographs of the construction of the various devices are shown in Fig. 7.5. The Marker class tags consisted of a small PCB to which the PIC18LF2620 microcontroller was soldered. The GPS receiver used in the Pack class tags was a LEA-4P demonstration board

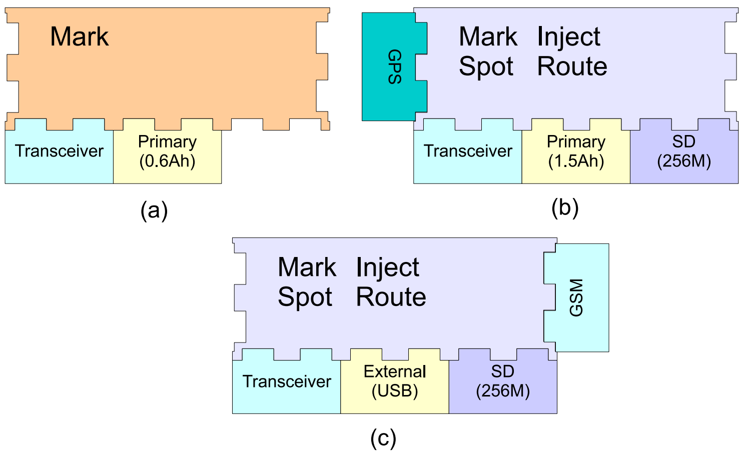


Figure 7.4: System diagram of the Radio Range test. The Marker class node is shown in (a). The system diagram of the GPS enabled Pack tag is shown in (b) and that of the Base class in (c).

and was mounted onto the main PCB. Short lengths of wire were used as antennas on the transceivers.

7.3.2 Results

The Marker tag was placed at a fixed location approximately 70 cm off the ground on a tree stump, with the whip antenna vertical. The Pack class tag was attached to a backpack, such that the patch antenna was facing skywards. The Marker tag and the Pack tag were powered up, and once the GPS receiver had successful satellite lock[2], a human subject walked on a hiking trail around the Marker device. The Pack tag executed a loop which first obtained a GPS fix and then entered spotting mode. In this way, the spotter record could be correlated with a particular location, allowing the expected range of the transceiver to be determined. The terrain was mountainous and there was tall grass but sparse tree cover. Once the test had been completed, the devices were taken back to the laboratory where the Pack tag automatically contacted the Base tag and downloaded all its data.

The data was parsed using a Python script to generate both a comma separated value (.csv) file and a keyhole markup language (.kml) file. KML is a file format which is compatible with Google Earth and allows for direct import of locations and traces [319] . Individual locations were colored according to whether the Marker was spotted at a

[2]To obtain a cold-start fix took three minutes on average, which is significantly longer than the quoted 36 s from the datasheet. This difference is presumably due a reduced signal strength in comparison to the -125 dBm level specified in the datasheet, which is a best-case scenario.

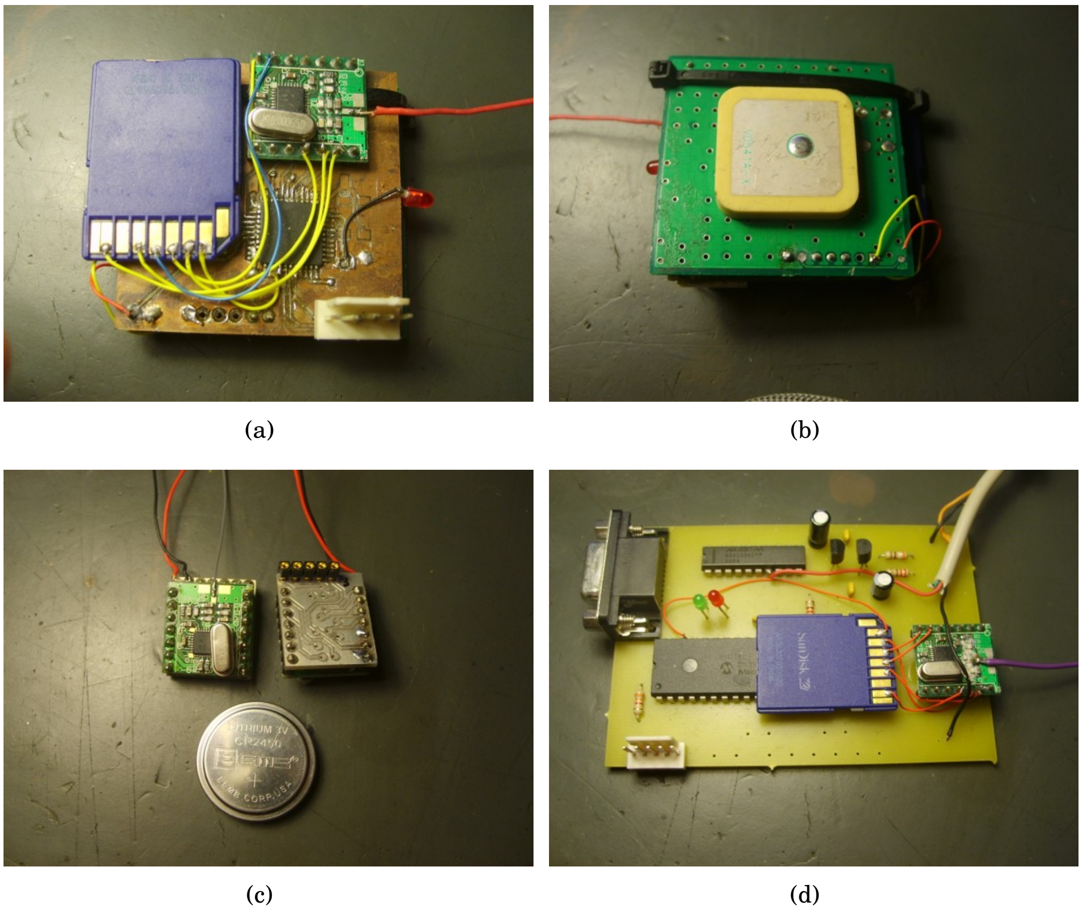


Figure 7.5: Construction of devices used in the range test. (a) shows the Pack class tag, with the SD memory card on the left, the transceiver on the top right and the microcontroller in the center. (b) shows the patch antenna on the top of the Pack tag. (c) is a picture of two Marker class tags next to a CR2450 coin cell. (d) shows the Base class tag used in this test with the serial interface connector on the left and transceiver on the right.

particular location or not.

The results from the test, after importing into Google Earth are shown in Fig. 7.6. The maximum radio range was found to be 354 m. Radio coverage however was not constant and varied with topography and vegetation.

7.3.3 Summary

This simple test demonstrated a number of key concepts. Firstly, that a Pack class tag could be used to record the beacons from a Marker class tag. Secondly, that the Pack tag could automatically upload its stored data to the Base tag when within range. Thirdly, that Google Earth provides a simple method of displaying location data and its context (in this, whether or not the Marker was detected). Lastly, that radio range using wire

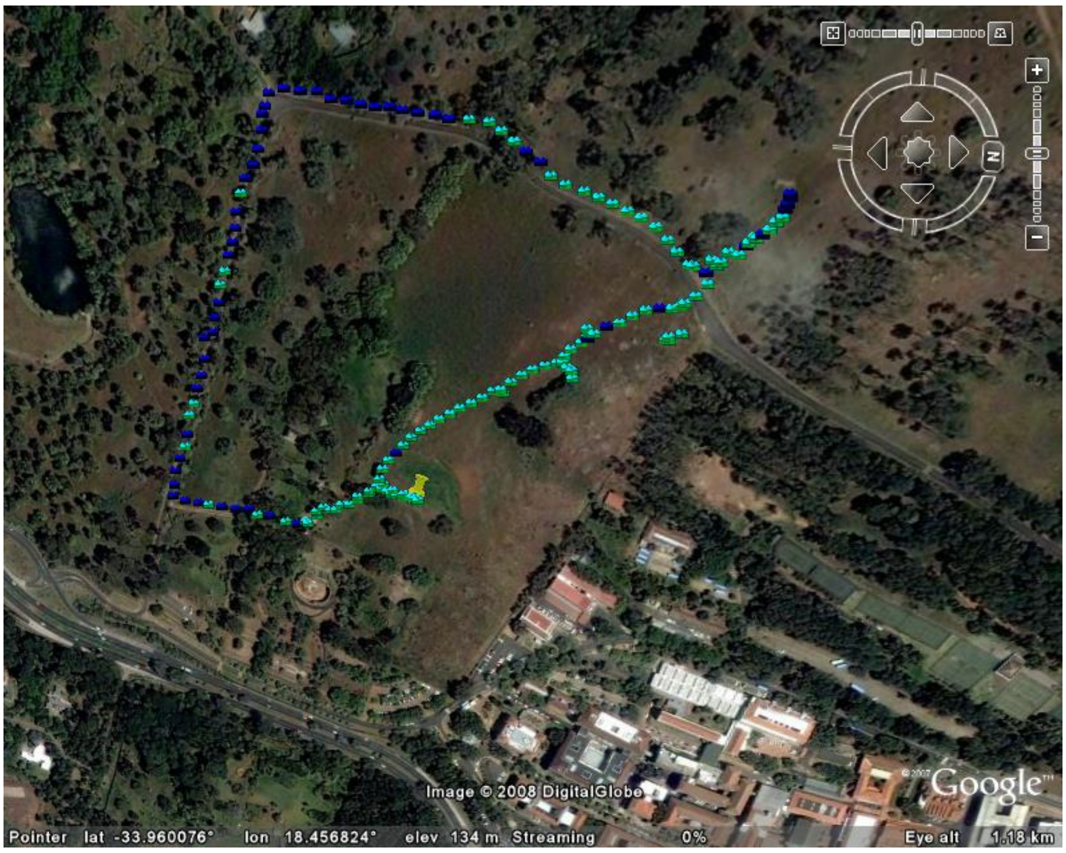


Image © 2008 DigitalGlobe
© 2007 Google
Pointer lat -33.960076° lon 18.456824° elev 134 m Streaming 0%
Eye alt 1.18 km

Figure 7.6: Results from the Range test as imported into Google Earth. Locations at which the Marker could be heard are denoted by light data pointers and points at which no signal was received by dark points. The position of the Marker is shown by a large icon. The university buildings can be seen in the bottom right hand corner.

whip antennas could be as high as 350 m, though was reliably approximately 100 m. Note that the position of the Marker tag (which is not GPS equipped) can be estimated from the contact log of the Pack tag, knowing the maximum radio range. Thus, this test also demonstrates that it is possible to determine the position of simple devices using GPS enabled devices.

7.4 GPS and acceleration testing

In order to obtain data for the uniform distance sampling approach presented in Chapter 5, a Pack tag was equipped with a GPS receiver and a triaxial accelerometer. Some of the results from the experiments are presented here. The tags that were used in the study were the same as those in Section 5.4 and thus details of their construction will not be repeated here.

A test was undertaken in order to investigate the effect of mountainous regions on the availability of GPS location fixes, as it would be expected that location fixes would fail if there was not a clear sky view. At the same time, the activity profile of the user was also to be determined by measuring the tag acceleration. The tracking tag was configured to take a 3.9 s single axis acceleration signature, obtained at a rate of 32 Hz, followed by a GPS location fix. The total time taken to acquire and store this data was seven seconds, giving a GPS sampling rate of 0.14 Hz. The tag was strapped to the left wrist of a human volunteer, and they went on a hike round the base of Table Mountain, Cape Town. When they returned, the data was automatically downloaded from the Pack tag through another Pack tag (to test the wireless network) and finally delivered to the Base tag, and analysis was performed on the acquired data. Similar to Section 7.3, the obtained data was parsed by a script to generate a keyhole markup file for import into Google Earth.

7.4.1 Results

The magnitude of the obtained acceleration data was used to add context to the map, by indicating activity at certain points of the trail. This was done by colorizing the icons according to the magnitude of the acceleration, with red indicating a high acceleration and blue a low acceleration. The obtained data is shown in Fig. 7.7. The gap in the data on the left hand side is presumably due to a loss of satellite lock, due to the effects of both the mountain and forest. A sample acceleration snapshot is shown in Fig. 7.8 corresponding to the user moving at a speed of approximately 1.5 m/s.

7.4.2 Summary

This test demonstrated that a Pack class tag could obtain GPS and accelerometer data and relay it through the network to the end user. Although this was only a two hop network, it still shows that it is feasible to upload information through intermediates to reach the Base tag. The map shows how context information (in this case acceleration magnitude) can be added to the location trail to provide additional information about the behaviour of the host at a point in time.

7.5 Archival Passive RFID logger

An archival type logger was constructed for monitoring the foraging behaviour of African Penguins (*Spheniscus demersus*) on Robben and Dassen Islands, South Africa. This project was commissioned by Marine and Coastal Management (MCM) to obtain long

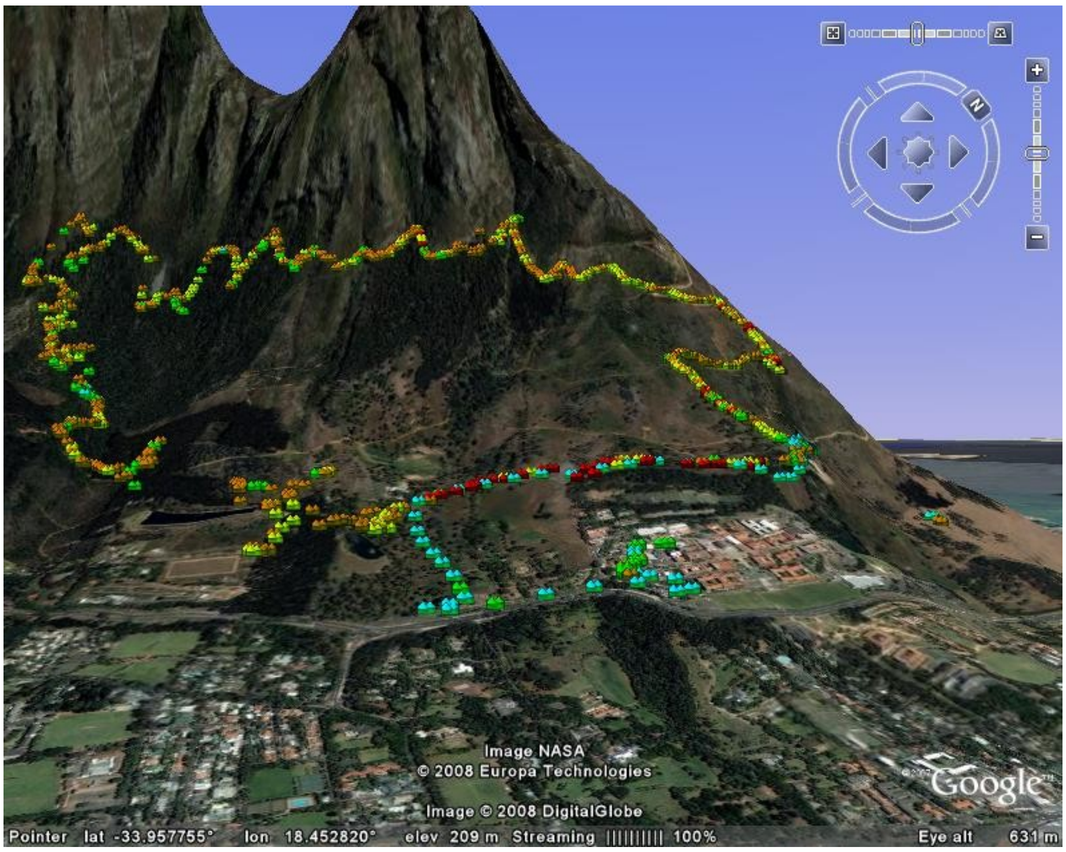


Image NASA
© 2008 Europa Technologies
Image © 2008 DigitalGlobe
Google
Pointer lat -33.957755° lon 18.452820° elev 209 m Streaming |||||||||| 100%
Eye alt 631 m

Figure 7.7: Results from the test of the availability of GPS fixes as imported into Google Earth. The path is colorized according to the magnitude of the tag acceleration, with red corresponding to a high acceleration and blue to a low acceleration.

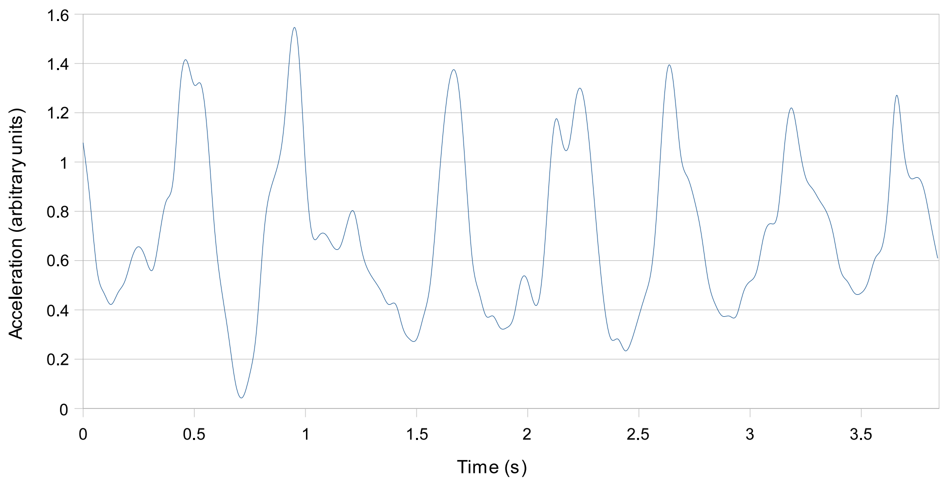


Figure 7.8: Acceleration snapshot from the mountain test.

term data about penguin survival. It is envisaged that 500 penguins will be transpondered over the duration of the study. Penguins walk along paths from their nests to the sea. By recording the time of exit and entry from the nesting colony, the duration of foraging trips can be determined. Penguins were injected with 32 mm glass encapsulated passive RFID transponders (by MCM personnel), which allow them to be uniquely identified. Passive RFID detectors have a very limited read range (typically 50–70cm) and thus can only be used for species which regularly pass a fixed point, but allow relatively inexpensive studies of large numbers of animals.

7.5.1 System Overview and Components

The overview of the system is shown in Fig.7.9. Note that the Archival Logger has no network functionality as it is not equipped with a transceiver. An Archival Logger is basically a Base Class tag without a transceiver. The microcontroller used in this project was the PIC18F67J11 as it has two serial ports (one for the serial link to the PC and the other to control the RFID reader).

A TI-RFID passive RFID reader was used to detect the transponders [137]. This only supports reading from one antenna, but to determine direction of travel two antennas had to be used. This necessitated the design and construction of an antenna multiplexor. This was controlled by a digital signal from the microcontroller which selected which antenna to read from. Due to supplier limitations, two slightly different readers were used – one with an RS-422 interface and the other with an RS-232 interface. Thus, the system had to support both standards. A wire link on the board was used to select the type of interface required.

The power consumption of the RFID reader required the use of a large battery reserve. When reading, the RFID reader unit consumes approximately 20 W. To reduce the power consumption, the reader is only triggered twice a second (corresponding to a read on each antenna every second). This reduced the total power consumption of the device to approximately 10 W. The unit is powered by a 102 Ah deep cycle battery, recharged by an 80W solar panel. The unit had to be solar powered as there is no source of mains power on Dassen Island and the deployment on Robben Island was far from a mains outlet.

Other sensors installed included a battery voltage monitor and a temperature sensor. The battery voltage monitor was used to prevent battery failure in the event of a long period of insufficient sun. It was configured to shut the system down at a battery voltage of 10.8 V and restart the system at 11.0 V. To accurately time-stamp the data, a battery backed real time clock was used, accessed via the microcontrollers's SPI interface. A large (256 Mb) memory card was used to store the acquired data.

Data is recovered off the device via an RS-232 link. As many modern laptops do not have RS-232 ports, a commercially available serial-USB converter was used to provide a bridge between the reader and the researcher's laptop. Operation of the logger is also indicated with three LEDs, which can be used for quick testing and validation of the reader's operation in the field.

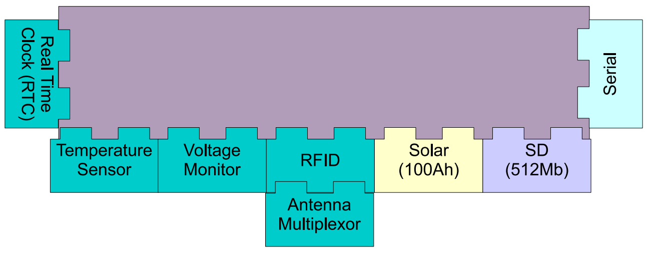


Figure 7.9: System diagram of the Archival RFID reader. The unit is powered by a 102Ah 12V battery which is recharged by an 80W solar panel.

7.5.2 System Construction and Operation

The electronic components for the RFID logger were installed inside a weatherproof box. Cables to the antenna, solar panel, battery and the serial interface were passed through waterproof glands to prevent water ingress. These were further covered with silicone sealant to reduce the possibility of a leak. This is especially important as the readers are placed within a few hundred metres of the ocean, leading to a high risk of corrosion. An aluminium frame was welded with marine grade filler rods. This frame provided the support for the large solar panel (dimensions: 1200 (width) $\times$ 650 (height) mm) and the battery and the box of electronics was attached to base of the frame. The solar panel was placed at an angle of 45°, as this corresponds to the elevation required for installation at a latitude of 34° South. The panel was oriented so it faced North. The battery was placed on the frame to provide a counterweight to wind induced moments that would result in the frame tipping over. This is especially important as winds on the island regularly reach a speed in excess of 65 km/h. To provide additional support, guy ropes were attached to the top of the frame and fastened with pegs on the ground. The antennas were constructed from CAT-5 network cable, as the multi-core cable reduces the impact of 'skin effects' at 134 kHz, which is the reader frequency of operation [137]. Ideally Litz cable (multiple stands of fine insulated wire) should be used, but CAT-5 provides an inexpensive substitute [137]. The antennas were buried under a path that the penguins frequented, and guides fabricated from plastic fencing were further installed to 'funnel' the birds along the route. Metal fencing could not be used as it would alter the resonant frequency of the antennas. To protect the CAT-5 cable, the antenna was wound inside 40 mm PVC pipe, made into a rectangle of dimensions 600 mm x 300 mm. Photographs of the various components are shown in Fig. 7.10.

A Graphical User Interface (GUI) was programmed using Python and wxPython (which is a wrapper for wxWidgets). This allowed the user to download data from the device, check its operation and alter settings. Data is downloaded from the device and parsed by the GUI into human readable data. This is saved in a comma separated value (.csv) file which is compatible with most spreadsheet and database programs. Data is acknowledged and checked with a 16 bit XOR checksum for errors before the next block is requested. In the event of an error, a block retransmission is attempted. Each block contains 32 16-byte packets, each of which can contain sensor data (temperature and voltage), tag data (transponder codes from the RFID reader) or system data (details of power failures and upload attempts).

Reading data from the logger is undertaken in a slightly different manner to the circular buffer presented earlier in Section 6.3.1, as data is read from the top of the buffer downwards. The transfer of data is terminated when the Python program encounters a record indicating a successful upload. Thus, if an upload fails for whatever reason, the upload will start again from the most recent record and carry on until a record is found which indicates that subsequent data has already been transferred. In addition, a special error recovery mode is supported where the user can request the download of a number of previous records.

The Archival Logger also outputs the timestamp, ID and antenna of transponders as and when they are read via the serial interface. This can be used so that a researcher can determine when a particular bird has passed through the device in case it needs to be recaptured. This can also be used to interface with other instruments, for example visual recognition systems [37], to validate their operation.

The unit was designed such that installation on the islands would be fairly simple and be able to completed in a few hours. The solar panel was only attached to the frame once on the island in order to reduce the chance of breakage. All that was required on site was to bury the antennas, connect the solar panel and battery and then set the system time via the GUI[3].

[3]In practice, deployment was complicated by the intermittent failure of one antenna due to a temperature related issue on the antenna changeover switch, necessitating three trips to Dassen Island, which were difficult to undertake due to adverse weather conditions.

7.5.3 Results

A day's data is shown in Fig. 7.11. The battery voltage starts off at 12.2 V and slowly drops until the solar panel starts charging the battery. By midday, the battery is fully charged and so the panel is disconnected from the battery to prevent overcharging. As the battery is loaded however, the battery voltage drops until the regulator applies charge again, rapidly restoring the battery to its fully charged level. This is responsible for the chatter seen in the plot. When the sun sets at approximately 17h30, the battery voltage reduces again to its overnight value. The temperature sensor inside the box indirectly measures the ambient temperature, which shows a peak at midday. The bottom plot shows the times of day when transponders were detected. As penguins are diurnal predators, this corresponds to an exit from the colony during the early hours of the day and a return after sunset.

7.5.4 Summary

The Archival Logger allows researchers to monitor the directions and transit times of penguins as they enter and leave the colony. The use of SD memory enables vast amounts of data to be stored (over 16 million time-stamped records) which reduces restrictions on how often researchers have to visit the islands to download the data. The solar panel, along with the large battery, allows the logger to run continuously, even through inclement weather conditions. Direct processing of the data downloaded through the GUI simplifies the procedure required to obtain time and direction of travel, as a separate parsing step is not required. Avoiding using a terminal program, which can be quite complex with regards to the correct setting of baud rates further simplifies the download process.

This test demonstrated that the same architecture used in Pack and Base tags could also be used for Archival Loggers. The unit is able to obtain data from a number of sensors and store it in its memory for later retrieval.

7.6 Future Deployment

At the time of writing, a number of tracking devices had been fabricated for field testing. These are currently being tested to ensure correct operation before they are encapsulated and shipped to Ongava Research Centre. Four GPS enabled Pack class tags, six Marker/Spotter tags and two Base class tags have been constructed.

The Pack tags consist of a PIC18F67J11 microcontroller, powered by a L6920DB switching regulator [312] which generates a stable 3.3 V supply from an input voltage range of

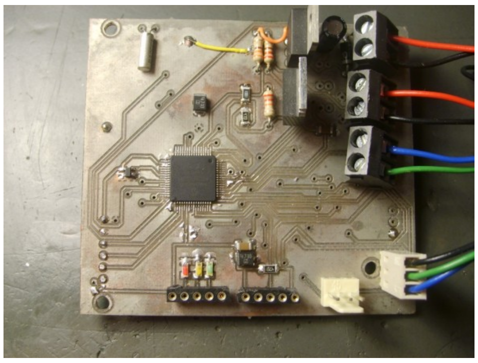
1.2-3 V. A 256 Mb SD memory card is used to provide mass storage – the plastic housing was removed from the card to reduce its size. This is glued directly onto the board and wires soldered to its terminals provide an interface to the microcontroller. The same transceiver board is used as for the other tests. Three LED's are installed for verification of operation in the field. An LIS3L02AS4 triaxial accelerometer [304] can optionally be installed for motion and acceleration based studies. A GPS receiver daughter-board can also be fitted to the main board if GPS functionality is required. The GPS receiver used in this implementation is the NEO-4S [28], which has increased tracking sensitivity. A Sarantel GeoHelix-SMP passive antenna is used to receive the GPS signal [320]. This antenna is less directional than a typical patch antenna, which makes device orientation less critical [320]. Space is also provided on the board for the installation of a serial transceiver, which allows for connection to a PC. Thus, this board can be configured to act as any of the classes presented in Chapter 3, ranging from an Archival Logger through to a Base tag. The Spotter tags constructed are the same as those that were used in Section 7.3. Photographs of the various tags and their construction are shown in Fig. 7.12.

7.7 Discussion and Summary

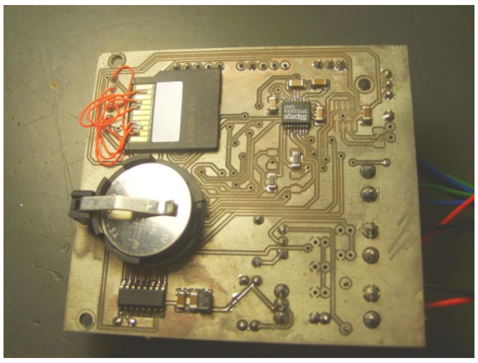
These real world deployments and tests show that the system design approach is indeed valid and useful. The same software structure was used in a number of different tests, with devices of varying complexity and functionality. The Spotter test showed that beacons can be used to detect the presence of other tags within radio range. The range test showed that the maximum radio range was 350 m, but the typical range was more likely to be in the region of 100-150 m. The RFID Archival Logger demonstrated that a long term data logger could be constructed using the EcoLocate system, and that it obtained useful data on the foraging trips of African Penguins.

The presentation of information is an important aspect of the data analysis. Mapping software, such as Google Earth, provided the ability to view not only the data, but also its context, in the form of colored data pointers. In addition, the user interface is an important aspect of the system, and it was found that a custom GUI was simpler to use than a terminal program.

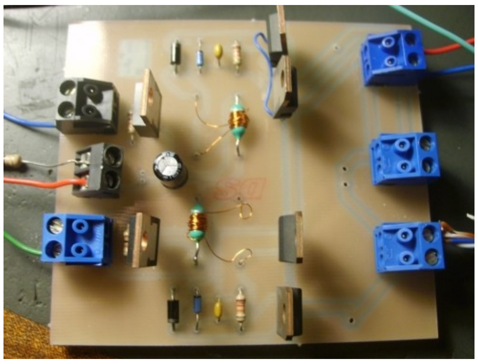
In summary, the results from these tests demonstrate the flexibility and modularity of the EcoLocate system.



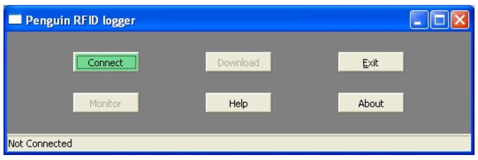
(a)



(b)

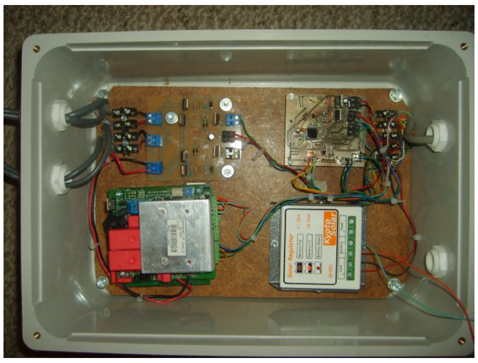


(c)

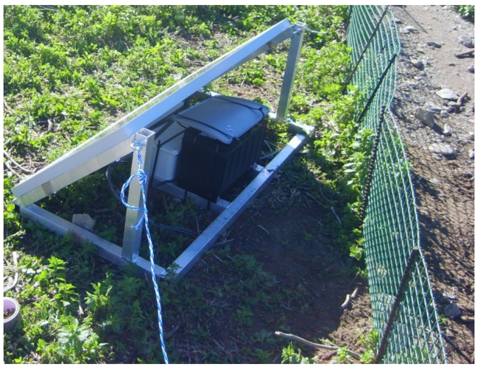


Not Connected

(d)



(e)



(f)

Figure 7.10: Penguin RFID Archival Logger. (a) shows the top side of the main controller board with the PIC18F67J11 microcontroller in the centre. The 32kHz crystal for time-keeping is on the top left and connections to the external devices are taken from the screw terminals on the right. (b) shows the underside of the main controller board, displaying the 256 Mb SD memory card and the battery-backed RTC. (c) The antenna multiplexor which is used to select which antenna to read from. (d) A screen-shot from the GUI. (e) The devices installed in the weatherproof box, ready for field installation. Clockwise from top right: logger board, solar regulator, RFID reader, antenna multiplexor. (f) Field deployment. The solar panel is mounted on the aluminium frame, and acts as a roof for the electronics box and battery. The antennas are buried underground within the fenced off path, on the right of the photograph.

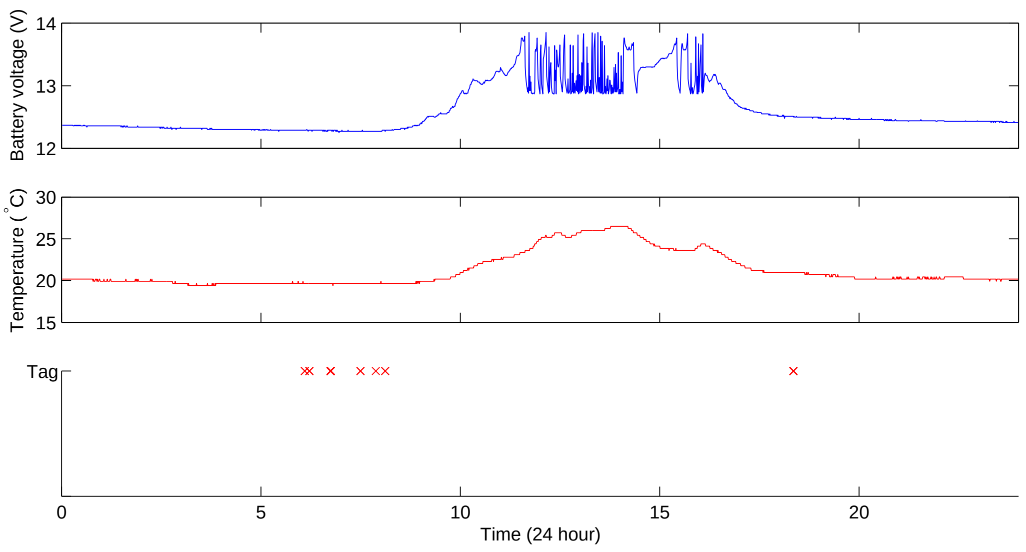
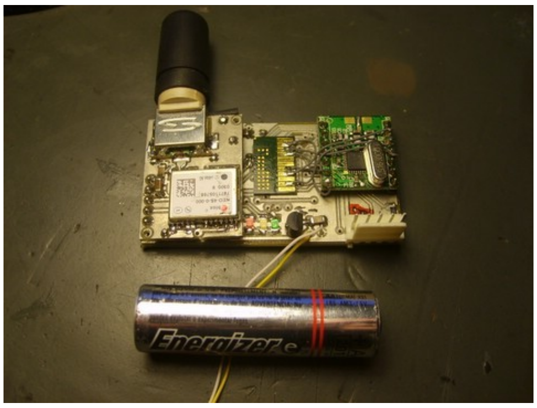
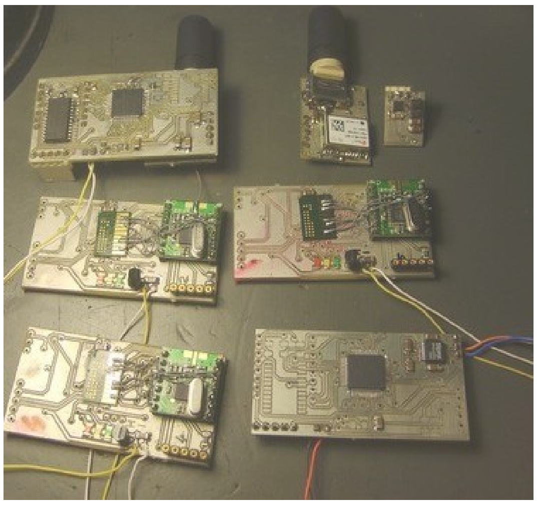
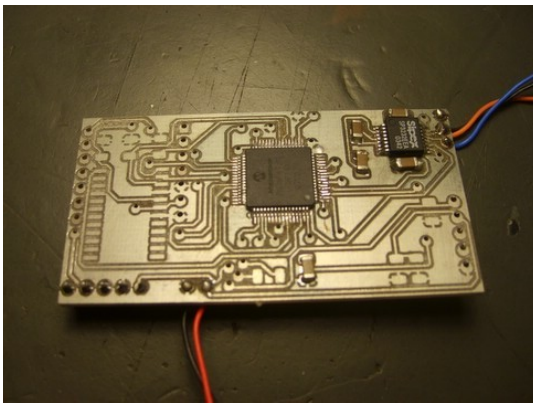


Figure 7.11: One day of data from the penguin RFID reader (23/06/2008). The top plot shows the battery voltage as measured by the unit. Chatter is due to the hysteresis action of the solar regulator. The middle plot is the temperature inside the Archival Logger box. The bottom plot indicates times at which transponders were detected. Note that the entry and exit times correspond to sunrise and sunset, as penguins are diurnal hunters.


(a)


(b)


(c)

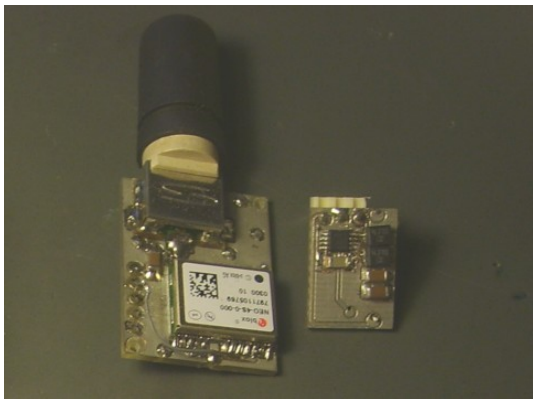

(d)

Figure 7.12: Fabrication of devices for field testing. (a) shows the top side of the Pack tag, next to an AA battery for comparison. The GPS module, with antenna, is on the left, the SD memory card in the middle and the transceiver can be seen on the right. (b) shows a selection of assembled devices. In (c), the underside of a tag is shown. The microcontroller is in the middle, and on the right is an RS-232 transceiver, which allows this device to be connected to a PC. The space on the left is for the optional installation of a triaxial acceleration sensor. (d) shows the GPS module daughterboard on the left and the switching regulator on the right.

8 Conclusions and Future Directions

8.1 Conclusion

Wildlife tracking using wireless networks has become recently possible due to the increasing minituarisation of electronic technology. This has allowed wildlife tracking collars to form multi-hop networks to transfer information from the field to the end-user. This is a new way of being able to track and monitor animals automatically. However, research to date has concentrated on instrumenting a single species of interest, resulting in a homogeneous (single tag design) solution. ZebraNet and TurtleNet placed GPS enabled wireless devices on zebras and turtles respectively. However, in both ZebraNet and TurtleNet, only one type of device was designed. However, their devices are only suitable for attachment to larger animals, thus limiting their deployment scope to a section of the Animal Kingdom.

In this dissertation, the problem of how to track and monitor a wide range of wild animals using wireless networks has been tackled. The literature review in Chapter 2 showed that there are many factors to be taken into account when designing a wildlife tracking system, some from the biological standpoint (such as tag shape and weight) and others from the technological standpoint (such as power requirements). As a consequence of the 5% bodyweight restriction, a homogeneous solution is not possible if small and large animals are to be monitored using the same tag design. This has resulted in many different solutions, ranging from VHF triangulation to GPS tracking.

The contribution of this work is the design of a wireless network thats bind together these disparate technologies and unifies them into a single data gathering system, that allows both small and large animals to be monitored. Thus, a heterogeneous solution (multiple classes of tags differentiated according to size, weight and functionality) was proposed, which led to interesting research both in the system design and with regards to network management and discovery. This research did not solely focus on the network and communications design, but also considered how to intelligently sense data as well. To validate the research approach, real world testing was undertaken using custom designed wireless sensor tags. Conclusions on the main areas of this work are now discussed.

EcoLocate: A heterogeneous wireless network system for wildlife tracking

EcoLocate, which is a modular and flexible approach to wildlife tracking, was presented in Chapter 3. It was shown how multiple classes of tracking devices could be created that varied in functionality. Thus, high functionality nodes with large amounts of energy can obtain information from lightweight nodes with reduced capabilities. Hence, the load of routing and networking is shifted to devices which are capable of doing so as a virtue of the larger size of their energy reserves. Typically, large animals (and vehicles and people) would carry full function nodes, acting as mobile data gatherers through the network, whereas smaller animals would carry simpler, lightweight devices.

Although multiple classes of nodes were introduced, it was shown how simpler nodes have a subset of the features of the more complex nodes. Thus, only one set of code needs to be maintained, simplifying design and maintenance. Determination of the node class can be performed at the compiler level or it can be dynamically altered in the field according to energy reserves and rate of consumption. It was shown how altering the class of the tracking device can result in a longer tag lifetime, at the cost of a reduction in functionality.

Central to the operation of the network is the beacon protocol, which implements a digital version of VHF tracking. Thus, many of the methods applicable to VHF tracking such as triangulation and homing can still be applied in this network. However, the network approach takes VHF tracking one step further, as nodes can act as receivers as well. In this way, proximity logs can be acquired by tracking devices, allowing social relationships to be determined. In addition coarse location can also be determined, allowing non-GPS enabled nodes to be located.

A modular hardware design was considered that allows for flexible tracking device specification and implementation. Based on the ‘block’ approach to the design, it was proposed that a graphical tool could be used to aid zoologists in designing their own tags and assessing the power consumption of the blocks. Example scenarios of the design were then presented, ranging from the very simple to fully featured networks. This showed how one system could be used at different levels of cost and complexity to undertake studies on wildlife tracking and monitoring.

The EcoLocate network design is based on modularity, both in terms of hardware and firmware. This allows for a single technical design which can be configured for multiple applications, from low power Marker tags to high functionality GPS enabled Pack tags. By allowing tags to alter their class dynamically at run time, tracking devices can survive in the network for longer. EcoLocate is not restricted to solely monitoring animals, and can be used to monitor vehicles, people and environmental data. Thus EcoLocate allows multiple aspects of the state of a game reserve to be monitored, allowing for in-depth research and informed management.

The Adaptive Social Hierarchy: A Network Ranking System

The next focus area was in the operation of the network itself in such a resource diverse environment. The adaptive social hierarchy (ASH) was presented in Chapter 4. This is a novel approach to network management and routing based on assessing a node's rank within the network, based on its attributes or resources. This method is inspired by the way animals form social hierarchies, where the fittest animal will typically be the leader. A number of methods were presented with the aim of accurately and rapidly forming a social hierarchy in a wireless network, only using locally available information.

Pairwise ASH was presented first, where nodes switch their ranks if they are in contradiction to the rank order of their attributes. Whilst relatively simple to analyze mathematically, this approach was shown to not be able to adapt to the insertion of new nodes. In order to equally spread the ranks, a reinforcement method was introduced. This 'rewarded' nodes if their ranks were correct in correspondence with their attributes, driving the rank of the best node towards 1 and that of the worst node towards 0. The same switching action was still incorporated, as this provides rapid alteration of incorrectly ordered ranks. It was demonstrated that this system was able to adapt to node insertions and removals, and spread ranks equally over the unit domain.

Pairwise ASH imposes an unfair load on the network, as nodes have to exchange data in pairs. Wireless networks are broadcast (one to many) in reality. The regularly transmitted beacons (used for the EcoLocate scheme to determine proximity) or 'Hello' packets of other network protocols to determine the ranks of nodes within the network can carry ASH information with minimal impact on bandwidth and network usage. The problem with the broadcast approach is that although the receivers can all update their rank in accordance to the newly obtained information, the transmitter will maintain the same rank, which could be incorrect. One-way ASH using domination was introduced, which is based on determining a node's win/loss ratio. Although this method was able to correctly determine nodes' ranks within the network, it suffers from slow convergence and does not incorporate the rank information into its update. The switching method from pairwise ASH was thus incorporated to improve performance, resulting in one-way ASH with switching. This showed good convergence speed and adaption to insertion and removal.

The prior approaches failed when they were applied to a stationary network, due to the effect of a limited horizon. To address this limitation, an agent based approach was presented, which creates pseudo-edges between non-adjacent nodes. Although network overhead is slightly increased, as ASH parameters and the parameters of the agent also have to be sent, this allows nodes to determine their ranks in both stationary and mobile networks. Each node keeps a buffer of agents and transmits these at random. Entries in the buffer are also replaced at random. Agents are transmitted in the same beacons or 'Hello' packets as in one-way ASH. An alternative method of interpolating a node's rank, based on estimating the ranks of its closest peers in the rank ordering was introduced. This showed good convergence and adaption to attribute variation. It was shown that agent ASH could be used both on mobile and static networks, and that scalability was good, with convergence time actually decreasing with increasing numbers of nodes in the same network area.

Suitable attributes to rank were then discussed, ranging from energy to connectivity. As ASH is a generic ranking system, any attribute which can be measured on a per-node basis can be used to form a hierarchy. It was then shown how ASH could act as a network underlay for existing routing protocols, by scaling the spread of the attributes to the unit domain. This reduces the reliance on tuning factors and allows the system to dynamically adapt to changes in network operating conditions. A simple cross-layer network management system was presented, which showed how ranking information could be used at multiple levels, from the MAC layer to the application itself.

The adaptive social hierarchy is a novel concept that orders nodes' attributes into a unit domain, irrespective of the spread of the attribute. This allows a network to dynamically adapt to varying attribute distributions, without requiring the respecification of network tuning parameters. This simplifies protocol design, as it does not need to be made application or scenario dependent. ASH is scalable, adaptable and can work in networks with mixed mobility.

ASH is based on assessing relative differences between nodes, using only locally overheard information. ASH does not load the network heavily, as rank information is sent as part of existing network discovery packets. The ASH concept is not limited to wildlife network monitoring, and is in theory applicable to any network, whether stationary or mobile. ASH can be used to improve the performance of existing routing protocols by allowing them to gracefully handle heterogeneity. In addition, ASH can be used at multiple layers of nodes, allowing for the simple implementation of cross-layer protocols.

Uniform Distance Sampling

In Chapter 5, the focus was shifted from the network itself to a power hungry sensor, the GPS receiver. In existing tracking systems, location fixes are attempted at uniform time intervals. When the animal is stationary (which accounts for a large proportion of some animal's behaviour), multiple fixes are taken which add no useful information. Conversely, when the animal is moving rapidly, there is the possibility of detail under-sampling. With these two drawbacks in mind, a novel method of scheduling GPS fixes at uniform distance intervals was proposed.

To estimate the distance travelled, a low power accelerometer was used to measure the gait of the host, as indirectly inferred from the motion of the neck mounted collar. Short acceleration snapshots are periodically acquired which are used to estimate the speed of the host. When the animal has exceeded a certain threshold distance, the GPS receiver is instructed to take a fix.

However, as there are many different animals, with different gaits and gait patterns, a self-learning algorithm was used. Summary statistics (acceleration range and jerk variance) are formed from the acceleration snapshot and used to predict the speed of the host, using a polynomial model. When the GPS receiver is active, it measures the speed of the host. This is then used to train the model using the RLS algorithm, such that it is able to learn the relationship between the speed of the host and the summary statistics. An adaptive distance threshold adjustment was proposed, so that the motion patterns of the host do not need to be specified in advance and the typical lifetime of the tracking collar can be estimated.

Results showed that the learning algorithm was able to accurately predict the speed of the host from acceleration snapshots, captured both on human and dog test subjects. It was shown how different placement on the human resulted in vastly different model parameters. This demonstrates that the learning approach is valid. Simulations were presented which showed that this method could be used to extend the lifetime of a tracking collar by a factor of 2.4 if the host was active only 30% of the time.

The uniform distance sampling approach has the added benefit of being able to generate detailed speed-time profiles which can be used for energetics studies in order to characterize the activity of the host at various times of day. Not only can the learning algorithm be used to alter the GPS fix rate, it can also be used to associate a particular gait waveform with a certain speed, thus allowing gait analysis to be performed in the field, using a neck mounted collar. In addition, as many wildlife tracking are already equipped with acceleration sensors, this sampling method can be easily implemented as a firmware upgrade. The learning algorithm was implemented on low cost 8 bit PIC microcontrollers and showed that it could realistically be used in an animal tracking collar.

This research has the potential to increase the amount of useful data obtained from a GPS collar, whilst using less power. This is a valuable contribution to the field of wildlife tracking.

Implementation and Results

In order to test whether the proposed methods were realistically implementable, several prototype classes of tags were designed and fabricated by the author. These used the PIC family of microcontrollers and were programmed in C. A number of different processor variants were used depending on the class of the device. In general, PIC18LF2620 devices were used for Marker and Spotter class devices, PIC18LF4620 for Pack class tags and PIC18F67J11 for Pack and Base class tags. SD memory cards were selected to provide inexpensive mass storage of large amounts of data.

Modularity was applied at the firmware level by restricting the size of the state machine universe. Thus simpler devices were made by preventing certain state transitions. This could be done at the compiler level, using directives, or dynamically by adjusting the allowable sequence of operations. This demonstrated that the approach of using multiple classes was easy to implement, as only one master code base (for the most complex tag) had to be written, and code for the simpler devices derived from that.

Results from some tests were presented in Chapter 7, demonstrating that the devices were usable in the field and that very different applications could be handled using the same basic design. In one test, Spotter class nodes were deployed to test their ability to log proximity to one another. The results showed that they were able to detect one another and upload their contact logs to a Base tag for analysis. To check the range of the transceivers, a simple test was performed using a Marker tag, a Pack tag and a Base tag. This demonstrated that the maximum range was 350 m, but typical distance was in the region of 150 m. The range of the devices depends greatly on the terrain and vegetation. Also presented were results from the deployment of an Archival Logger with RFID sensor, used to monitor the transit times of penguins on two islands off the coast of South Africa. The logger is essentially equivalent to a Base tag with no network interface, demonstrating that the same firmware can be used for a wide variety of studies.

Analysis of the power consumption of the various tracking devices demonstrated the need for a heterogeneous solution, as increased functionality resulted in an increase in device weight. Simple devices (such as Markers) can be made lightweight and are inexpensive, allowing them to be used in a wide range of studies and attached to many different animal species. High end devices, like GPS enabled Pack tags are heavier and more expensive, limiting their deployment scope to larger creatures.

8.1.1 Summary

In summary, a wireless network for wildlife tracking was designed that is able to track and monitor a wide range of animal species, from small to large. Particular emphasis was paid to modularity and scalability. A novel network protocol was conceived which is able to determine the rank of a node within the network using local information in a self-organizing manner. A new method of acquiring GPS samples, using a tracking collar that learns the gait patterns of the host creature was also developed. Devices were built and tested in order to validate the design concepts. This work is an initial step to the realistic monitoring of animals and their habitats using wireless networks, so that users from all around the world can analyse and observe data as it is collected by the system. Thus, one could envisage a researcher in America obtaining real-time updates on lion behaviour from a game reserve in Africa, without having to leave their desk. Ultimately, it is hoped that this work will help aid in conservation, by providing hard scientific data on the behaviour and habits of wild animals and their interaction with the environment.

8.2 Summary of Contributions

The principal contributions of this dissertation are as follows:

- The design of a wireless network based tracking solution that can be used on a wide variety of animals, where small animal collars (with limited functionality and small energy reserves) use the capabilities of large ones (with high functionality and large energy reserves) to send information through the network to the end user. The network is not restricted to animal deployment, and can be include environmental sensing and vehicle and human monitoring.

- The proposal of six different classes of tracking devices with varying degrees of complexity that are based on a unified hardware and software design. Simpler devices are implemented by limiting the functionality of more complex devices. This can be done dynamically in the field, allowing devices to alter their class based on their remaining energy reserve, leading to a longer lived network. Devices were fabricated and tested by the author in order to demonstrate the flexibility of the modular approach.

- Developing a system that allows social relationships (contacts between animals), location and proximity (presence or absence of the animal within a certain area) to be determined, without requiring that all devices are GPS enabled.
- Developing a lightweight, adaptive network management protocol that is able to handle large degrees of heterogeneity in resources, allowing a device to determine its ranking within the network (which can be fixed or mobile). This provides resource scalability and can be used as an underlay to enhance the performance of existing routing protocols.
- Proposing a novel way of acquiring GPS samples, based on the accelerometer dependent estimate of distance that the host has travelled, rather than at uniform time intervals, leading to an increase in device lifetime without loss of location information. A method that learns and adapts to the way the host moves is also developed. As a byproduct of distance estimation, a detailed speed-time profile is generated, which can be used to characterize animal behaviour.

8.3 Future Directions

Based on the presented work, a number of interesting research areas for future work are proposed.

8.3.1 Machine learning of behaviour patterns

In the uniform distance sampling approach (Chapter 5), it was showed how it is possible for a tracking collar to learn the gait patterns of the host and how these can be used to predict its speed. Here a sketch is presented on how to expand this idea further, such that a device is able to learn when to perform actions based on the input it receives from its sensors. A node adjusts its behaviour based on feedback received from the environment, so as to decide what action to undertake in the future when presented with a similar stimulus from its sensors. This is particularly relevant to the wildlife tracking scenario, as animals have habitual behaviour patterns, and these can be learned to automatically schedule certain actions.

As a motivating example, consider an Active Logger attached to a penguin. The Active Logger has a sea-water sensor and a real time clock (RTC), so the foraging behaviour of the animal can be determined. The device wakes up every few minutes and determines whether the device is immersed in water or not, timestamping this information and storing it in memory. Summary statistics of how long the animal is in the water are to be uploaded opportunistically to a Base tag placed along a penguin path, similar to that presented in Section 7.5. The transceiver in the penguin tag can only upload data to the Base tag when it is within range, which is generally for a few minutes in the early morning when it leaves the colony and again in the evening when it returns to its nest.

Due to the power consumption of the transceiver, it cannot be operated continually and thus must be duty cycled to reduce its power drain. The standard approach to this problem would be to attempt to contact the basestation every hour or at random points in time. However, a learning approach is motivated, where the tag learns when is the best time to contact the Base station. Similar to Chapter 5, a model is constructed that is dynamically adjusted when new data is encountered. If an upload is attempted by the Archival Logger and is successful, the model can be updated to reflect that there is a higher probability of contacting the Base tag at the current time. Over time, it would be expected that the model would reflect the diurnal nature of the animal, with high upload probabilities in the early morning and in the evening. The Archival Logger would preferentially attempt to contact the Base tag at these points in time, saving energy and reducing latency.

Note that not only is there a correlation between the time of day and possible contact times, but also that contact is impossible when the penguin is at sea. Thus, this shows that multiple sensor states should be input into the learning algorithm. The ultimate aim is for a device attached to an animal to automatically learn possible correlations between its sensor state (such as time, location, temperature, activity and so on) and the success of a particular action (such as attempting an upload or acquiring a GPS fix and so on).

8.3.2 Incorporation of aerial devices

Air based transceivers suffer less from terrain and vegetation induced range reduction, as compared with terrestrial based communication systems. Aerial tracking is often used for VHF tracking to boost the detection range [107]. To date, aerial tracking has been performed using human piloted aircraft. The wireless network design would be greatly improved with the incorporation of aerial mobile nodes, which could be autonomously controlled. These devices could be used to retrieve data from the network, and also fitted with cameras to detect and track wild animals. These devices could also be used to deploy stationary sensors, and possibly even attach sensors to wild animals. A major benefit of using airborne vehicles is that radio signals are less affected by local topography, boosting communications range.

8.3.3 Network Security

An area that has not been addressed in this work is how to make the system secure to attacks. For example in the ASH system, it would be simple for a node to misrepresent its rank and so ‘attract’ packets from nodes in the network. Data can also be overheard, using scanning receivers. The consequences of such a breach in security could have serious consequences. This is because wildlife tracking is often undertaken on rare or endangered species, some of which have obtained their status as a result of poaching. Additionally, animals are sometimes commercially valuable as a result of their rarity, implying that live capture of such animals is a possibility. A wildlife tracking system provides data on the location and behaviour of animals and thus this same information can be used by malicious parties to kill or steal animals. This suggests that strong encryption should be used on all data transfers to prevent interception and the use of frequency hopping or spread spectrum to prevent detection.

Another possible threat is Denial Of Service (DoS) attacks. One scenario is in the competition between the environment and the economy. As a hypothetical example, consider a company which is thought to be disturbing the behaviour of a particular animal or group of animals in the surrounding area, either through competition for resources or through pollution. Researchers want to show that threats to the animal population are directly correlated to the actions of the company. Thus they tag a number of creatures with tracking devices, in order to prove the impact of the commercial operations and subsequently halt them. The company is aware that without data, no correlation can be scientifically proven and thus use high power transmitters to jam the network, preventing data transfers from being undertaken. Whilst extreme, this scenario is not impossible. Thus, resistance to jamming and DoS attacks is another area that should be investigated.

8.3.4 Triggered GPS

The uniform distance sampling method presented in Chapter 5 demonstrated that it was possible to learn the gait patterns of the host and schedule GPS fixes at equally spaced intervals in distance. The same technique could be used for event triggered GPS, so that an event of interest (such as a high speed predator/prey pursuit) would trigger the GPS unit and leave it switched on for a few minutes, acquiring detailed data about these events. The tag could also enter spotting mode and scan the neighbourhood for beacons within range. In this way, it could be possible to detect and record predation events and identify the victim, if it were so equipped with a tracking device. In addition, for animals that travel in packs, the high speed pursuit event detected by one tracking tag could switch the other tracking tags to also enter continuous fix mode. In this way, pack predation behaviour could be detected without the requirement that the GPS unit has to be powered up continually. To detect automatically detect predation events, a model could be constructed of typical host behaviour, and when unusual events occur trigger detailed GPS sampling.

8.3.5 Data Analysis, Interpretation and Presentation

A problem that needs to be addressed is how to analyze the vast amounts of data that will be obtained from a wireless network for wildlife tracking like Ecolocate. Consider the volume of data generated by a GPS receiver sampling at a rate of 1Hz – in one day, 86400 locations will be acquired. With multiple animals and objects in the network, it is possible that millions of data points corresponding to location, temperature, acceleration, water-level and so on, will be obtained. The pressing question will be how to extract useful information from these massive data-sets.

Data mining and machine learning techniques can be used to find patterns in the data. There are two analyses that need to be undertaken – typical behaviour and anamolous behaviour. Characterizing the typical behaviour of an animal for example would be determining its typical home range and how this slowly varies over time. New data is compared against the typical behaviour, and if there is a significant departure, anamolous behaviour can be monitored. Based on the exceptionality of the data compared to the norm, different levels of alarms can be triggered. For example, if an animal does not move for 24 hours, then this could possibly generate a high level alarm, alerting the user that there is a potential problem. A less critical alert could be that the pump on a waterhole is working at an inefficient level. Detecting anamolous data can be undertaken automatically, for example by determining the likelihood of a particular event given its prior behaviour, or based on manual thresholds (which are based on likelihoods in any case) such as alerting the user if the level of water in a drinking hole drops below 10%. Either way, it is important that the system detects critical alerts without triggering too many alerts. This suggests a reinforcement approach, where alerts that should be displayed in future are rewarded and those which are not critical are penalized by a user. Thus, with feedback into the model, it can learn what events should be suppressed and what events warrant triggering an alarm.

A further issue with the huge datasets is how to effectively store and access the information. This suggests that compression algorithms will have to be designed that exploit the regularity in the environmental data.

8.3.6 The Case for Shared Data Repositories

Closely linked to data analysis is the need for wildlife tracking data to be made available to other researchers. Due to the high costs in obtaining the data, this does not have to be made free, but data should be shared. The disturbance and possible risk to a wild animal from the action of attaching a tracking device implies that any data obtained should be used for as many studies as possible. In fact, it is possible that there can be a split between the data acquisition component and the data analysis component. Currently, a single research team is involved in all areas of wildlife tracking, from sedating the animal and attaching the tracking collar, to downloading the data to performing statistical analysis upon it. In the future, it would make more sense for these tasks to be separated, such that one group tags the animals and sells (portions of) the data to other researchers for analysis. This is especially relevant when one considers the volumes of data generated and the number of research angles that can be undertaken. In this way, multiple research teams can essentially pool together to form a richer study than any one of them could undertake by themselves.

For example, one group could be interested in the behaviour of lions and another in the behaviour of zebra. They pool their resources and hire a specialized tracking company to simultaneously monitor lion and zebra in the same area, augmenting the data with other environmental variables. They can use the data for their own respective studies. Another group may be interested in the interactions between lion and zebra, and so obtains and analyzes the same data. Yet another team wants to determine the activity cycle of lions in this area compared to another area, and the correlation to water availability.

In summary, due to the impact of wildlife tracking on animal behaviour, it should be subjected to the largest scrutiny and analysis possible. Implementing such a model of data sharing and trading is likely to be a research avenue of the future.

8.3.7 Automatic deployment

Consumer electronics is likely to continue in driving the size and power requirements for GPS receivers, microcontrollers and sensors downwards. Wildlife tracking, which is a niche market, will benefit from these reductions, resulting in smaller and smaller tracking devices, allowing a broader spectrum of animals to be monitored. However, the fundamental difficulty in wildlife tracking is not in the tag design, but in attaching the tracking device to the animal. Whilst automatic methods of fastening collars to animals have been considered, success rates have not been very high, preventing their widespread use [97, 98, 99]. A more advanced tagging method could be used, possibly with a vision based system to only tag the 'correct' animals and prevent recollaring of the same individual. If the animal is to be tagged, it could be immobilized in a cage or net. Some automatic method could be used to fasten the collar around the animal's neck before it is released. As capture would be brief, and not involve humans, this may result in less stress to the animal and reduced deployment costs. Another method of remote deployment would be to somehow propel a self-affixing tag at an animal. This is an area which warrants further investigation, as a solution to this problem would greatly aid the field of wildlife tracking.

Bibliography

[1] W. W. Cochran and R. D. Lord, "A radio-tracking system for wild animals," *Journal of Wldlife Management*, vol. 27, pp. 9–24, 1963.

[2] BioTrack Telemetry. [Online]. Available: http://www.biotrack.co.uk/

[3] Bluesky Telemetry, UK. [Online]. Available: www.blueskytelemetry.co.uk

[4] Habit Research. [Online]. Available: www.habitresearch.com

[5] Lotek Fish and Wildlife Monitoring Systems. [Online]. Available: www.lotek.com

[6] Sirtrack Wildlife Tracking Solutions, New Zealand. [Online]. Available: http://www.sirtrack.com/

[7] Televilt Wildlife Tracking Collars. [Online]. Available: www.televilt.se

[8] G. C. White and R. A. Garrott, *Analysis of Wildlife Radio-Tracking Data*. Academic Press, Inc. San Diego, 1990.

[9] B. Naef-Daenzer, D. Fruh, M. Stalder, P. Wetli, and E. Weise, "Miniaturization (0.2 g) and evaluation of attachment techniques of telemetry transmitters," *Journal of Experimental Biology*, vol. 208, no. 21, pp. 4063–4068, 2005.

[10] ARGOS. Overview of the argos system. [Online]. Available: http://www.argos-system.org/

[11] W. S. Seegar, P. N. Cutchis, M. R. Fuller, J. J. Suter, V. Bhatnagar, and J. G. Wall, "Fifteen years of satellite tracking development and application to wildlife research and conservation," *Johns Hopkins APL Technical Digest*, vol. 17, p. 401411, 1996.

[12] Microwave telemetry. [Online]. Available: http://microwavetelemetry.com/

[13] Africa wildlife tracking. [Online]. Available: http://www.awt.co.za/

[14] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. S. Peh, and D. Rubenstein, "Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet," *ACM SIGPLAN Notices: Session on Emerging Systems*, vol. 37, no. 10, pp. 96–107, 2002.

[15] M. Martonosi, "Embedded systems in the wild: Zebranet software, hardware, and deployment experiences," *SIGPLAN Notices*, vol. 41, no. 7, pp. 1–1, 2006.

[16] C. M. Sadler and M. Martonosi, "Data compression algorithms for energy-constrained devices in delay tolerant networks," in *SenSys '06: Proceedings of the 4th international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2006, pp. 265–278.

[17] T. Liu, C. M. Sadler, P. Zhang, and M. Martonosi, "Implementing software on resource-constrained mobile sensors: experiences with Impala and ZebraNet," in *MobiSys '04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM, 2004, pp. 256–269.

[18] P. Zhang, C. M. Sadler, S. A. Lyon, and M. Martonosi, “Hardware design experiences in ZebraNet,” in *SenSys ’04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2004, pp. 227–238.

[19] W. Cochran, “Wildlife telemetry,” in *Wildlife management techniques manual*, S. Schemnitz, Ed. Random House, 1980.

[20] TurtleNet: UMass outdoor Mobile Environment. [Online]. Available: http://prisms.cs.umass.edu/dome/index.php?page=turtlenet

[21] L. D. Mech and S. M. Barber, “A Critique Of Wildlife Radio-Tracking And Its Use In National Parks A Report to The U.S. National Park Service,” p. 83, February 2002.

[22] E. Bonabeau, G. Theraulaz, and J. Deneubourg, “Dominance orders in animal societies: The self-organization hypothesis revisited,” *Bulletin of Mathematical Biology*, vol. 61, no. 4, 1999.

[23] R. C. Brace and J. Pavey, “Size-dependent dominance hierarchy in the anemone actinia equina,” *Nature*, vol. 273, pp. 752–753, 1978.

[24] B. Forkman and M. J. Haskell, “The maintenance of stable dominance hierarchies and the pattern of aggression: Support for the suppression hypothesis,” *Ethology*, vol. 110, no. 9, pp. 737–744, 2004.

[25] A. Guhl, “Social hierarchy and dominance,” in *Social behavior of the domestic fowl*, M. Schein, Ed. Dowden, Hutchinson, and Ross, 1975, pp. 156–201.

[26] F. A. Issa, D. J. Adamson, and D. HEdwards, “Dominance hierarchy formation in juvenile crayfish *procambarus clarkii*,” *Journal of Experimental Biology*, vol. 202, no. 24, pp. 3497–3506, 1999.

[27] T. Monnin and C. Peeters, “Dominance hierarchy and reproductive conflicts among subordinates in a monogynous queenless ant,” *Behavioral Ecology*, vol. 10, no. 3, pp. 323–332, 1999.

[28] u-blox AG, Switzerland, “NEO-4S ANTARIS 4 ROM Based GPS Module with SuperSense Data Sheet,” GPS.G4-MS4-06107, 2007. [Online]. Available: www.u-blox.com

[29] Y. Wang, P. Zhang, T. Liu, C. Sadler, and M. Martonosi, “CRAWDAD data set princeton/zebranet (v. 2007-02-14),” Downloaded from http://crawdad.cs.dartmouth.edu/princeton/zebranet, Feb. 2007.

[30] M. W. Hayward and G. J. Hayward, “Activity patterns of reintroduced lion *Panthera leo* and spotted hyaena *Crocuta crocuta* in the Addo Elephant National Park, South Africa,” *African Journal of Ecology*, vol. 45, no. 2, pp. 135–141, 2007.

[31] R. J. Moll, J. J. Millspaugh, J. Beringer, J. Sartwell, and Z. He, “A new ‘view’ of ecology and conservation through animal-borne video systems,” *Trends in Ecology & Evolution*, vol. 22, no. 12, pp. 660–668, December 2007.

[32] D. Chabot, “The use of heart rate telemetry in assessing the metabolic cost of disturbances,” in *Transactions of the North American Wildlife Natural Resources Conference*, no. 56, 1991, pp. 256–263.

[33] W. Stohr, “Long term heart rate telemetry in small mammals,” in *Biotelemetry X*, C. J. Amlaner, Ed. Univ. Arkansas Press, Fayetteville, 1989, pp. 353–375.

[34] V. A. Langman, “A radio-biotelemetry system for monitoring body temperature and activity levels in the Zebra Finch,” *Auk*, vol. 90, pp. 375–383, 1973.

[35] R. A. Garrott and R. M. Bartmann, “Evaluation of vaginal implants for mule deer,” *Journal of Wildlife Management*, vol. 48, no. 2, pp. 646–648, 1984.

[36] R. P. Wilson, “Reconstructing the past using futuristic developments: trends and perspectives in logger technology for penguins,” *Mem. Natl Inst. Polar Res., Spec. Issue*, vol. 58, pp. 34–49, 2004.

[37] T. Burghardt, P. J. Barham, N. Campbell, I. C. Cuthill, R. B. Sherley, and T. M. Leshoro, “A fully automated computer vision system for the biometric identification of african penguins (*Spheniscus demersus*) on robben island,” in *6th International Penguin Conference (IPC07), Hobart, Tasmania, Australia.*, E. J. Woehler, Ed., September 2007, p. 19.

[38] T. Burghardt and J. Calic, “Analysing animal behaviour in wildlife videos using face detection and tracking,” *IEE Proceedings in Vision, Image, and Signal Processing*, vol. 153, no. 3, pp. 305–312, June 2006.

[39] J. Rife, “Automated robotic tracking of gelatinous animals in the deep ocean,” Ph.D. dissertation, Stanford University, 2003.

[40] W. A. Watkins, M. A. Daher, J. E. George, and D. Rodriguez, “Twelve years of tracking 52-Hz whale calls from a unique source in the North Pacific,” *Deep Sea Research Part I: Oceanographic Research Papers*, vol. 51, no. 12, pp. 1889–1901, December 2004.

[41] W. Hu, V. N. Tran, N. Bulusu, C. T. Chou, S. Jha, and A. Taylor, “The design and evaluation of a hybrid sensor network for cane-toad monitoring,” in *IPSN ’05: Proceedings of the 4th international symposium on Information Processing in Sensor Networks*. Piscataway, NJ, USA: IEEE Press, 2005, pp. 71–77.

[42] R. P. Wilson and C. R. McMahon, “Measuring devices on wild animals: what constitutes acceptable practice?” *Frontiers in Ecology and the Environment*, vol. 4, no. 3, pp. 147–154, 2006.

[43] R. P. Wilson, W. S. Grant, and D. C. Duffy, “Recording devices on free-ranging marine animals: does measurement affect foraging performance?” *Ecology*, vol. 67, pp. 1091–1093, 1986.

[44] H. D. J. N. Aldridge and R. M. Brigham, “Load Carrying and Maneuverability in an Insectivorous Bat: A Test of The 5% ‘Rule’ of Radio-Telemetry,” *Journal of Mammalogy*, vol. 69, no. 2, pp. 379–382, 1988.

[45] P. Hooge, “The effects of radio weight and harnesses on time budgets and movements of acorn woodpeckers,” *Journal of Field Ornithology*, vol. 62, no. 2, pp. 230–238, 1991.

[46] J. D. Winter, “Underwater biotelemtry,” in *Fisheries techniques*, L. A. Nielsen and D. L. Johnson, Eds. American Fisheries Society, Bethesda, Maryland, 1983, pp. 371–395.

[47] R. S. Brown, S. J. Cooke, W. G. Anderson, and R. S. McKinley, “Evidence to challenge the ‘2% rule’ for biotelemetry,” *North American Journal of Fisheries Management*, vol. 19, pp. 867–871, 1999.

[48] V. Langman, T. Roberts, J. Black, G. Maloiy, N. Heglund, J. Weber, R. Kram, and C. Taylor, “Moving cheaply: energetics of walking in the African elephant,” *J Exp Biol*, vol. 198, no. 3, pp. 629–632, 1995.

[49] D. K. Koehler, T. D. Reynolds, and S. H. Anderson, “Radio-transmitter implants in 4 species of small mammals,” *The Journal of Wildlife Management*, vol. 51, no. 1, pp. 105–108, 1987.

[50] B. M. Culik and R. P. Wilson, “Swimming energetics and performance of instrumented Adelie penguins (*Pygoscelis adeliae*),” *Journal of Experimental Biology*, vol. 158, pp. 355–368, 1991.

[51] B. M. Culik, R. Bannasch, and R. P. Wilson, “External devices on penguins: how important is shape?” *Marine Biology*, vol. 118, pp. 353–357, 1994.

[52] K. Watson and R. Granger, “Hydrodynamic effect of a satellite transmitter on a juvenile green turtle (*Chelonia mydas*),” *Journal of Experimental Biology*, vol. 201, pp. 2497–2505, 1998.

[53] C. J. Pennycuick, “Mechanisms of flight,” in *Avian Biology, Vol 5*, D. S. Farner and J. R. Kings, Eds. Academic Press, NY, 1975, pp. 1–75.

[54] C. J. Pennycuick, M. R. Fuller, and L. McAllister, “Climbing performance of harris’ hawks (*Parabuteo unicinctus*) with added load: implications for muscle mechanisms and radiotracking,” *Journal Experimental Biology*, vol. 142, pp. 17–29, 1989.

[55] H. H. Obrecht, C. J. Pennycuick, and M. R. Fuller, “Wind tunnel experiments to assesss the effect of back-mounted radio transmitters on bird body drag,” *Journal Experimental Biology*, vol. 135, pp. 265–273, 1988.

[56] R. P. Wilson, J. A. Scolaro, and F. Quintana, “To the bottom of the heart: cloacal movement as an index of cardiac frequency, respiration and digestive evacuation in penguins,” *Marine Biology*, vol. 144, pp. 813–827, 2004.

[57] R. P. Wilson, J. M. Kreye, K. Lucke, and H. Urquhart, “Antennae on transmitters on penguins: balancing energy budgets on the high wire,” *Journal of Experimental Biology*, vol. 207, pp. 2649–2662, 2004.

[58] M. Guillemette, A. J. Woakes, A. Flagstad, and P. J. Butler, "Effects of data-loggers implanted for a full year in female common eiders," *The Condor*, vol. 104, no. 2, p. 448452, 2002.

[59] J. A. Green, J. L. Tanton, A. J. Woakes, I. L. Boyd, and P. J. Butler, "Effects of long-term implanted data loggers on macaroni penguins *Eudyptes chrysolophus*," *Journal of Avian Biology*, vol. 35, no. 4, pp. 370–376, 2004.

[60] A. Goth, A. Jones, and D. N. Jones, "Transmitter attachment and its effects on Australian brush-turkey hatchlings," *Wildlife Research*, vol. 28, pp. 73–78, 2001.

[61] J. A. Jackson, B. J. Schardien, and G. W. Robinson, "A problem associated with the use of radio transmitters on tree surface foraging birds," *International Bird Banding News*, vol. 49, pp. 50–53, 1977.

[62] S. J. Dougill, L. Johnson, P. C. Banko, D. M. Goltz, M. R. Wiley, and J. D. Semones, "Consequences of antenna design in telemetry studies of small passerines," *Journal of Field Ornithology*, vol. 71, no. 3, p. 385388, 2000.

[63] T. C. Dunstan, "Types and uses of radio packages for north american falconiform and strigiform birds," in *Proceedings of the First International Conference on Wildlife Biotelemetry, University of Wyoming, Laramie*, 1977, pp. 30–39.

[64] L. D. Mech, V. B. Keuchle, D. W. Warner, and J. R. Tester, "A collar for attaching radio transmitters on rabbits, hares and raccoons," *Journal of Wildlife Management*, vol. 29, pp. 898–902, 1965.

[65] R. Kenward, *Wildlife Radio Tagging: Equipment, Field Techniques and Data Analysis*. Academic Press, New York., 1987.

[66] S. B. Merrill, L. G. Adams, M. E. Nelson, and L. D. Mech, "Testing Releasable GPS Radiocollars on Wolves and White-Tailed Deer," *Wildlife Society Bulletin*, vol. 26, no. 4, pp. 830–835, 1998.

[67] D. B. Jackson and R. E. Green, "The importance of the introduced hedgehog (*Erinaceus europaeus*) as a predator of the eggs of waders (Charadrii) on machair in South Uist, Scotland," *Biological Conservation*, vol. 93, no. 3, pp. 333–348, May 2000.

[68] M. R. Fuller, W. S. Seegar, and L. S. Schueck, "Routes and travel rates of migrating Peregrine Falcons *Falco peregrinus* and Swainson's Hawks *Buteo swainsoni* in the Western Hemisphere," *Journal of Avian Biology*, vol. 29, no. 4, pp. 433–440, 1998.

[69] F. Anderson and P. Hitchings, "A radio-tracking system for the black rhinoceros," *South African Journal of Wildlife Management*, vol. 35, no. 4, pp. 636–643, 1971.

[70] J. A. Lerczak, K. E. W. Shelden, and R. C. Hobbs, "Application of Suction-cup-attached VHF Transmitters to the Study of Beluga, *Delphinapterus leucas*, Surfacing Behavior in Cook Inlet, Alaska," *Marine Fisheries Review*, vol. 62, no. 3, pp. 99–111, 2000.

[71] D. V. Vuren, “Effects of intraperitoneal transmitter implants on yellow-bellied marmots,” *The Journal of Wildlife Management*, vol. 53, no. 2, pp. 320–323, 1989.

[72] L. Herbst, “Pathological and reproductive effects of intraperitoneal telemetry devices on female armadillos,” *The Journal of Wildlife Management*, vol. 55, no. 4, pp. 628–631, 1991.

[73] J. R. Davis, A. F. Von Recum, D. D. Smith, and D. C. Guynn, “Implantable telemetry in beaver,” *Wildlife Society Bulletin*, vol. 12, no. 3, pp. 322–324, 1984.

[74] J. Guynn, David C., J. R. Davis, and A. F. V. Recum, “Pathological potential of intraperitoneal transmitter implants in beavers,” *The Journal of Wildlife Management*, vol. 51, no. 3, pp. 605–606, 1987.

[75] M. D. Samuel and M. R. Fuller, “Wildlife radiotelemetry,” in *Research and Management Techniques for Wildlife and Habitats*, T. A. Bookhout, Ed. Wildlife Society, Bethesda, Maryland, 1996, pp. 370–418.

[76] Y. Ropert-Coudert and R. P. Wilson, “Trends and perspectives in animal-attached remote sensing,” *Frontiers in Ecology and the Environment*, vol. 3, pp. 437–444, 2005.

[77] R. Ree and A. F. Bennett, “Home range of the squirrel glider (*Petaurus norfolcensis*) in a network of remnant linear habitats,” *Journal of Zoology*, vol. 259, no. 4, pp. 327–336, 2003.

[78] F. Blanc and A. Brelurut, “Short-term behavioural effects of equipping red deer hinds with a tracking collar,” in *First International Symposium on Physiology and Ethology of Wild and Zoo Animals*, 1996, pp. 18–26.

[79] Sanyo Battery Solutions. [Online]. Available: http://www.sanyo.co.jp/

[80] NorthStar Tracking. [Online]. Available: http://www.northstarst.com/

[81] A. J. Bamford, M. Diekmann, A. Monadjem, and J. Mendelsohn, “Ranging behaviour of cape vultures gyps coprotheres from an endangered population in namibia,” *Bird Conservation International*, vol. 17, pp. 331–339, 2007.

[82] B. U. Meyburg, M. Gallard, C. Meyburg, and E. Dimitrova, “Migrations and sojourn in Africa of Egyptian vultures (*Neophron percnopterus*) tracked by satellite,” *Journal of Ornithology*, vol. 145, pp. 273–280, 2004.

[83] S. Akesson and H. Weimerskirch, “Albatross long-distance navigation: Comparing adults and juveniles,” *Journal of Navigation*, vol. 58, pp. 365–373, 2005.

[84] J. Sorber, A. Kostadinov, M. Garber, M. Brennan, M. D. Corner, and E. D. Berger, “Eon: a language and runtime system for perpetual systems,” in *SenSys ’07: Proceedings of the 5th international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2007, pp. 161–174.

[85] J. A. Paradiso and T. Starner, “Energy scavenging for mobile and wireless electronics,” *IEEE Pervasive Computing*, vol. 4, no. 1, pp. 18–27, 2005.

[86] J. Weber, K. Potje-Kamloth, F. Haase, P. Detemple, F. Volklein, and T. Doll, "Coin-size coiled-up polymer foil thermoelectric power generator for wearable electronics," *Sensors and Actuators A: Physical*, vol. 132, pp. 325–330, 2006.

[87] L. Mateu and F. Moll, "Optimum Piezoelectric Bending Beam Structures for Energy Harvesting using Shoe Inserts," *Journal of Intelligent Material Systems and Structures*, vol. 16, no. 10, pp. 835–845, 2005.

[88] T. von Buren, P. Mitcheson, T. Green, E. Yeatman, A. Holmes, and G. Troster, "Optimization of inertial micropower generators for human walking motion," *IEEE Sensors Journal*, vol. 6, no. 1, pp. 28–38, Feb. 2006.

[89] E. M. Yeatman, "Rotating and gyroscopic mems energy scavenging," in *International Workshop on Wearable and Implantable Body Sensor Networks (BSN'06)*, 2006, pp. 42–45.

[90] H. Yu, Y. Li, Y. Shang, and B. Su, "Design and investigation of photovoltaic and thermoelectric hybrid power source for wireless sensor networks," in *3rd IEEE International Conference on Nano / Micro Engineered and Molecular Systems (NEMS 2008)*, 2008.

[91] J. Clarke and K. Kerry, "The effects of monitoring procedures on Adelie Penguins," *CCAMLR Science*, vol. 1, pp. 155–164, 1993.

[92] V. Galanti, G. Tosi, R. Rossi, and C. Foley, "The use of GPS radio-collars to track elephants (*Loxodonta Africana*) in the tarangire national park (tanzania)," *Hystrix*, vol. 11, no. 2, pp. 27–37, 2000.

[93] P. C. Cross and W. M. Getz, "Assessing vaccination as a control strategy in an ongoing epidemic: Bovine tuberculosis in african buffalo," *Ecological Modelling*, vol. 196, no. 3, pp. 494–504, 2006.

[94] H. Ebedes, J. V. Rooyen, and J. G. D. Toit, "Capturing wild animals," in *Game Ranch Management*, J. du P Bothma, Ed. Van Scheik, 2002, pp. 382–440.

[95] J. Hattingh, N. I. Pitts, and M. F. Ganhao, "Immediate response to repeated capture and handling of wild impala," *Journal of Experimental Zoology*, vol. 248, pp. 109–112, 1988.

[96] L. D. Mech and E. M. Gese, "Field testing the Wildlink Capture Collar on wolves," *Wildlife Society Bulletin*, vol. 20, pp. 249–256, 1992.

[97] L. J. Verme, "An automatic tagging device for deer," *The Journal of Wildlife Management*, vol. 26, no. 4, pp. 387–392, 1962.

[98] C. G. Mahan, R. H. Yahner, and L. R. Stover, "Development of remote-collaring techniques for red squirrels," *Wildlife Society Bulletin*, vol. 22, no. 2, pp. 270–273, 1994.

[99] M. Kirchoff and K. White, "Development of a passive-capture technique for radiotagging large animals," Alaska Department of Fish and Game. Federal Aid in Wildlife Restoration Final Research report. Proj. 15.10., 2002.

[100] G. Cederlund, T. Dreyfert, and P. A. Lemnell, *Radiotracking techniques and the reliability of systems used for larger birds and mammals*. Liber Tryck, Stockholm., 1979.

[101] Advanced Radio Telemetry System ARTS. [Online]. Available: http://www.princeton.edu/~wikelski/research/

[102] R. M. Smith and B. Trevor-Deutsch, "A practical, remotely-controlled, portable ratio telemetry receiving apparatus," in *A handbook on biotelemetry and radiotracking*, C. Amlaner and D. W. MacDonald, Eds. Permagon Press, Oxford, UK, 1980, pp. 269–273.

[103] W. W. Cochran, D. W. Warner, J. R. Tester, and V. B. Kuechle, "Automatic radio-tracking system for monitoring animal movements," *BioScience*, vol. 15, no. 2, pp. 98–100, 1965.

[104] K. L. Heezen and J. R. Tester, "Evaluation of radio-tracking by triangulation with special reference to deer movements," *The Journal of Wildlife Management*, vol. 31, no. 1, pp. 124–141, 1967.

[105] R. E. Kenward, *A manual for wildlife radio tagging*. Academic Press, San Diego, California, 2001.

[106] W. W. Cochran and L. L. Pater, "Direction finding at ultra high frequencies (UHF): improved accuracy," *Wildlife Society Bulletin*, vol. 29, no. 2, pp. 594–599, 2001.

[107] P. J. Seddon and R. F. Maloney, "Tracking wildlife radio-tag signals by light fixed wing aircraft," Department of Conservation Technical Series No. 30, New Zealand, 2004.

[108] R. P. Larkin, A. Raim, and R. H. Diehl, "Performance of a non-rotating direction-finder for automatic radio tracking," *Journal of Field Ornithology*, vol. 67, no. 1, pp. 59–71, 1996.

[109] A. Jun-Ichi, T. Jun-Ichi, D. Masatoshi, and A. Akira, "Real-time location estimation system for wild animals," *Papers of Technical Meeting on Instrumentation and Measurement, IEE Japan*, vol. IM-06, pp. 37–40, 2006.

[110] C. Carroll, "Following the stealth hunter," *National Geographic*, vol. 11, pp. 66–77, 2005.

[111] F. Lindzey, H. Sawyer, C. Anderson, and B. Banulis, "Performance of Store-on-Board GPS Collars on Elk, Mule Deer and Mountain Lions in Wyoming, USA," in *Tracking animals with GPS: An International conference held at the Macaulay land use research institute, Aberdeen, Scotland*, 2001.

[112] D. B. Siniff and J. R. Tester, "Computer analysis of animal-movement data obtained by telemetry," *BioScience*, vol. 15, no. 2, pp. 104–108, 1965.

[113] M. L. Gorman, M. G. L. Mills, and J. French, "Satellite tracking of the African wild dog *Lycaon pictus*," in *Wildlife Telemetry: Remote Monitoring and Tracking of*

Animals, I. G. Priede and S. M. Swift, Eds. Ellis Horwood, New York, NY, 1992, pp. 218–228.

[114] J. M. Burns and M. A. Castellini, "Dive data from satellite tags and time-depth recorders: A comparison in Weddell Seal pups," *Marine Mammal Science*, vol. 14, pp. 750–764, 1998.

[115] D. Austin, J. I. McMillan, and W. D. Bowen, "A Three-Stage Algorithm For Filtering Erroneous ARGOS Satellite Locations," *Marine Mammal Science*, vol. 19, no. 2, pp. 371–383, 2003.

[116] F. Royer and M. Lutcavage, "Filtering and interpreting location errors in satellite telemetry of marine animals," *Journal of Experimental Marine Biology and Ecology*, vol. 359, no. 1, pp. 1–10, 2008.

[117] L. Peske and M. McGrady, "A system for locating satellite-received transmitters (PTTs) in the field," *Wildlife Society Bulletin*, vol. 33, no. 1, pp. 307–312, 2005.

[118] K. A. Keating, W. G. Brewster, and C. H. Key, "Satellite telemetry: performance of animal-tracking systems," *Journal of Wildlife Management*, vol. 55, pp. 160–171, 1991.

[119] M. Wikelski, R. W. Kays, N. J. Kasdin, K. Thorup, J. A. Smith, and G. W. Swenson, "Going wild: what a global small-animal tracking system could do for experimental biologists," *Journal of Experimental Biology*, vol. 210, no. 2, pp. 181–186, 2007.

[120] S. Tomkiewicz, "GPS Application for Wildlife A Review," *Telonics Quarterly*, vol. 9, no. 1, 1996.

[121] E. Kaplan, Ed., *Understanding GPS : principles and applications*. Artech House, 1996.

[122] u-blox AG, Switzerland, "ANTARIS 4 GPS Modules System Integration Manual (SIM)," GPS.G4-MS4-05007-A1, 2007. [Online]. Available: www.u-blox.com

[123] C. Coelho, L. de Melo, M. Sabato, D. Rizel, and R. Young, "A note on the use of GPS collars to monitor wild maned wolves *Chrysocyon brachyurus*," *Applied Animal Behaviour Science*, vol. 105, no. 1–3, pp. 259–264, 2007.

[124] V. J. Chadwick and G. W. Garner, "Performance of a satellite-linked GPS on Pacific walruses *Odobenus rosmarus divergens*," *Polar Biology*, vol. 25, no. 3, pp. 235–237, 2004.

[125] R.G. D'Eon, "Effects of a Stationary GPS Fix-Rate Bias on Habitat-Selection Analyses," *The Journal of Wildlife Management*, vol. 67, no. 4, pp. 858–863, 2003.

[126] P. G. Ryan, S. L. Petersen, G. Peters, and D. Gremillet, "GPS tracking a marine predator: the effects of precision, resolution and sampling rate on foraging tracks of African Penguins," *Marine Biology*, vol. 145, pp. 215–223, 2004.

[127] Wildtrack telemetry systems limited. [Online]. Available: http://www.wildtracker.com/

[128] Telit Wireless Solutions, "GC864-QUAD-C2 GSM Modem," 2007.

[129] R. P. Wilson, N. Liebsch, F. Q. Ian M. Davies, H. Weimerskirch, S. Storch, K. Lucke, U. Siebert, S. Zankl, G. Muller, I. Zimmer, A. Scolaro, C. Campagna, J. Plotz, H. Bornemann, J. Teilmann, and C. R. McMahon, "All at sea with animal tracks; methodological and analytical solutions for the resolution of movement," *Deep Sea Research Part II: Topical Studies in Oceanography*, vol. 54, no. 3–4, pp. 193–210, 2007.

[130] R. P. Wilson, M. P. Wilson, R. Link, H. Mempel, and N. J. Adams, "Determination of movements of African penguins, *Spheniscus demersus*, using a compass system: dead reckoning may be an alternative to telemetry," *Journal of Experimental Biology*, vol. 157, pp. 557–564, 1991.

[131] R. P. Wilson and M. P. Wilson, "Dead reckoning: a new technique for determining penguin movements at sea," *Meeresforschung*, vol. 32, pp. 155–158, 1988.

[132] R. W. Davis, L. A. Fuiman, T. M. Williams, and B. J. L. Boeuf, "Three-dimensional movements and swimming activity of a northern elephant seal," *Comparative Biochemistry and Physiology - Part A: Molecular & Integrative Physiology*, vol. 129, no. 4, pp. 759–770, 2001.

[133] Y. Mitani, K. Sato, S. Ito, M. Cameron, D. Siniff, and Y. Naito, "A method for reconstructing three-dimensional dive profiles of marine mammals using geomagnetic intensity data: results from two lactating Weddell seals," *Polar Biology*, vol. 26, pp. 311–317, 2003.

[134] R. P. Wilson, "Beyond rings in birds for determination of movements: whither the archival tag?" *Ardea*, vol. 89, pp. 231–240, 2001.

[135] R. P. Wilson, D. Gremillet, J. Syder, M. A. Kierspek, S. Garthe, H. Weimerskirch, C. Schafer-Neth, J. A. Scolaro, C. Bost, J. Plotz, and D. Nel, "Remote-sensing systems and seabirds: their use, abuse and potential for measuring marine environmental variables," *Marine Ecology Progress Series*, vol. 228, pp. 241–261, 2002.

[136] W. Kao, "Integration of GPS and dead-reckoning navigation systems," in *Vehicle Navigation and Information Systems Conference*, 1991.

[137] RFID Systems, Texas Instruments. [Online]. Available: http://www.ti.com/rfid/

[138] Oregon RFID. [Online]. Available: http://www.oregonrfid.com/

[139] ISO 11784:1996, *Radio frequency identification of animals – Code structure*. ISO, Geneva, Switzerland.

[140] D. L. Swain, L. A. Wilson, and J. Dickinson, "Evaluation of an active transponder system to monitor spatial and temporal location of cattle within patches of a grazed sward," *Applied Animal Behaviour Science*, vol. 84, pp. 185–195, 2003.

[141] D. Mascanzoni and H. Wallin, “The harmonic radar: a new method of tracing insects in the field,” *Ecological Entomology*, vol. 11, no. 4, pp. 387–390, 1986.

[142] J. R. Riley and A. D. Smith, “Design considerations for an harmonic radar to investigate the flight of insects at low altitude,” *Computers and Electronics in Agriculture*, vol. 35, pp. 151–169, 2002.

[143] R. W. Piper and S. G. Compton, “A novel technique for relocating concealed insects,” *Ecological Entomology*, vol. 27, no. 2, pp. 251–253, 2002.

[144] V. Drake, “Radar observations of moths migrating in a nocturnal low-level jet,” *Ecological Entomology*, vol. 10, no. 3, pp. 259–265, 1985.

[145] R. P. Wilson, J. J. Ducamp, G. W. Rees, B. M. Culik, and K. Niekamp, “Estimation of locaction: global coverage using light intensity,” in *Wildlife Telemetry: remote monitoring and tracking of animals*, I. Priede and S. Swift, Eds. Ellis Horwood, London, 1992, pp. 131–134.

[146] R. Delong, B. Stewart, and R.D.Hill, “Documenting migrations of northern elephant seals using day length,” *Marine Mammal Science*, vol. 8, pp. 155–159, 1992.

[147] R. Phillips, J. Silk, J. Croxall, V. Afanasyev, and D. Briggs, “Accuracy of geolocation,” *Marine Ecology Progress Series*, vol. 266, pp. 265–272, 2004.

[148] S. Teo, A.Boustany, S.Blackwell, A.Walli, K. Weng, and B. Block, “Validation of geolocation estimates based on light level and sea surface temperature from electronic tags,” *Marine Ecology Progress Series*, vol. 283, pp. 81–98, 2004.

[149] T. L. Cutler and D. E. Swann, “Using remote photography in wildlife ecology: A review,” *Wildlife Society Bulletin*, vol. 27, no. 3, pp. 571–581, 1999.

[150] R. D. Heilbrun, N. J. Silvy, M. E. Tewes, and M. J. Peterson, “Using automatically triggered cameras to individually identify bobcats,” *Wildlife Society Bulletin*, vol. 31, no. 3, pp. 748–755, 2003.

[151] C.-E. Chen, A. M. Ali, and H. Wang, “Design and testing of robust acoustic arrays for localization and enhancement of several bird sources,” in *IPSN ’06: Proceedings of the fifth international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2006, pp. 268–275.

[152] S. Mazhar, T. Ura, and R. Bahl, “Vocalization based individual classification of humpback whales using support vector machine,” in *Oceans 2007*, 2007, pp. 1–9.

[153] C. Clark, J. Holland, J. Bower, J. Tavares, P. McGregor, and T. Dabelsteen, “Accuracy of a passive acoustic location system: empirical studies in terrestrial habitats,” *Ethology, Ecology and Evolution*, vol. 9, no. 3, 1997.

[154] L. J. Cole, “The tagging of wild birds as a means of studying their movements,” *Auk*, vol. 26, pp. 137–143, 1909.

[155] E. R. Buchler, “A chemiluminescent tag for tracking bats and other small nocturnal animals,” *Journal of Mammalogy*, vol. 57, no. 1, pp. 173–176, 1976.

[156] I. L. Boyd and T. Arnbom, "Diving behaviour in relation to water temperature in the southern elephant seal: foraging implications," *Polar Biology*, vol. 11, no. 4, pp. 259–266, 1991.

[157] B. M. Culik, K. Putz, R. P. Wilson, C. A. Bost, Y. L. Maho, and J. L. Verselin, "Core temperature variability in diving king penguins (*Aptenodytes patagonicus*): a preliminary analysis," *Polar Biology*, vol. 16, no. 5, pp. 371–378, 1996.

[158] R. P. Wilson, J. Ploetz, and J. Cooper, "Can we determine when marine endotherms feed? A case study with seabirds," *Journal of Experimental Biology*, vol. 167, pp. 267–275, 1992.

[159] A. Ancel, M. Horning, and G. Kooyman, "Prey ingestion revealed by oesophagus and stomach temperature recordings in cormorants," *Journal of Experimental Biology*, vol. 200, pp. 149–154, 1997.

[160] P. J. Ponganis, R. P. V. Dam, D. H. Levenson, T. Knower, K. V. Ponganis, and G. Marshall, "Regional heterothermy and conservation of core temperature in emperor penguins diving under sea ice," *Comparative Biochemistry and Physiology - Part A: Molecular & Integrative Physiology*, vol. 135, pp. 477–487, 2003.

[161] K. Willis and M. Horning, "A novel approach to measuring heat flux in swimming animals," *Journal of Experimental Marine Biology and Ecology*, vol. 315, no. 2, pp. 147–162, 2005.

[162] T. Williams, L. Fuiman, M. Horning, and R. Davis, "The cost of foraging by a marine predator, the weddell seal *Leptonychotes weddellii*: pricing by the stroke," *Journal of Experimental Biology*, vol. 207, pp. 973–982, 2004.

[163] D. Noren, T. Williams, P. Berry, and E. Butler, "Thermoregulation during swimming and diving in bottlenose dolphins, *Tursiops truncatus*," *Journal of Comparitive Physiology*, vol. 169, pp. 93–99, 1999.

[164] G. Kooyman, J. Croxall, and D. Costa, "Diving depths and energy requirements of king penguins," *Science*, vol. 217, pp. 726–727, 1982.

[165] R. P. Wilson and C. A. R. Bain, "An inexpensive depth gauge for penguins," *The Journal of Wildlife Management*, vol. 48, no. 4, pp. 1077–1084, 1984.

[166] K. Yoda, K. Sato, Y. Niizuma, M. Kurita, C. Bost, Y. Le Maho, and Y. Naito, "Precise monitoring of porpoising behaviour of Adelie penguins determined using acceleration data loggers," *Journal of Experimental Biology*, vol. 202, no. 22, pp. 3121–3126, 1999.

[167] S. Watanabe, M. Izawa, A. Kato, Y. Ropert-Coudert, and Y. Naito, "A new technique for monitoring the detailed behaviour of terrestrial animals: A case study with the domestic cat," *Applied Animal Behaviour Science*, vol. 94, pp. 117–131, 2005.

[168] R. P. Wilson, E. L. C. Shepard, and N. Liebsch, "Prying into the intimate details of animal lives: use of a daily diary on animals," *Endangered Species Research*, vol. 4, pp. 123–137, 2008.

[169] R. P. Wilson, C. R. White, F. Quintana, L. G. Halsey, N. Liebsch, G. R. Martin, and P. J. Butler, “Moving towards acceleration for estimates of activity-specific metabolic rate in free-living animals: the case of the cormorant,” *Journal of Animal Ecology*, vol. 75, no. 5, pp. 1081–1090, 2006.

[170] E. Shepard, R. Wilson, F. Quintana, D. Albareda, L. A. Gmez, L. Halsey, N. Liebsch, A. Gleiss, D. Morgan, A. Myers, C. Newman, and D. Macdonald, “Identification of animal movement patterns using tri-axial accelerometry (preprint),” *Endangered Species Research*, vol. 4, no. 3, pp. 1–14, 2008.

[171] G. Marshall, “A video-collar to study aquatic fauna: a view from the animal’s back,” in *Spirit of Enterprise: The 1990 Rolex Awards*, D. Reed, Ed. Buri International, 1990, pp. 57–59.

[172] G. Marshall, “Crittercam: an animal-borne imaging and data logging system,” *Marine Technology Society Journal*, vol. 32, pp. 11–17, 1998.

[173] A. Takahashi, K. Sato, Y. Naito, M. J. Dunn, P. N. Trathan, and J. P. Croxall, “Penguin-mounted cameras glimpse underwater group behaviour,” *Proceedings of the Royal Society B: Biological Sciences*, vol. 271, pp. 281–282, 2004.

[174] J. A. Seminoff, T. T. Jones, and G. J. Marshall, “Underwater behaviour of green turtles monitored with video-time-depth recorders: what’s missing from dive profiles?” *Marine Ecology Progress Series*, vol. 322, pp. 269–280, 2006.

[175] P. U. Alkon, Y. Cohen, and P. A. Jordan, “Towards an acoustic biotelemetry system for animal behavior studies,” *The Journal of Wildlife Management*, vol. 53, no. 3, pp. 658–662, 1989.

[176] L. DeVries and W. DeVries, “Acoustic measurement of intake and grazing behaviour of cattle,” *Grass and Forage Science*, vol. 55, pp. 97–104, 2000.

[177] D. N. P. U. Alkon and P. Krausman, “Using acoustic telemetry to monitor foraging by penned mule deer,” *Wildlife Society Bulletin*, vol. 33, no. 2, pp. 624–632, 2005.

[178] C. V. Grant, R. D. Thompson, and G. W. Corner, “Determining avian ECG and respiration with a single-channel radiotransmitter,” *Journal of Applied Physiology*, vol. 30, no. 2, pp. 302–303, 1971.

[179] P. J. Butler and A. J. Woakes, “Changes in heart rate and respiratory frequency during natural behaviour of ducks, with particular reference to diving,” *Journal of Experimental Biology*, vol. 79, no. 1, pp. 283–300, 1979.

[180] P. Butler and A. Woakes, “Heart rate and aerobic metabolism in Humboldt penguins, *Spheniscus humboldti*, during voluntary dives,” *Journal of Experimental Biology*, vol. 108, no. 1, pp. 419–428, 1984.

[181] D. Duarte, V. Silva, A. Jaguaribe, D. Gilmore, and C. Da Costa, “Circadian rhythms in blood pressure in free-ranging three-toed sloths (*Bradypus variegatus*),” *Brazilian Journal of Medical and Biological Research*, vol. 36, pp. 273 – 278, 2003.

[182] L. G. Ross, W. Watts, and A. H. Young, “An ultrasonic biotelemetry system for the continuous monitoring of tail-beat rate from free-swimming fish,” *Journal of Fish Biology*, vol. 18, no. 4, pp. 479–490, 1981.

[183] P. J. Ponganis, “Bio-logging of physiological parameters in higher marine vertebrates,” *Deep Sea Research Part II: Topical Studies in Oceanography*, vol. 54, no. 3, pp. 183–192, 2007.

[184] R. D. Andrews, D. P. Costa, B. J. L. Boeuf, and D. R. Jones, “Breathing frequencies of northern elephant seals at sea and on land revealed by heart rate spectral analysis,” *Respiration Physiology*, vol. 123, pp. 71–85, 2000.

[185] R. P. Wilson, A. Steinfurth, Y. Ropert-Coudert, A. Kato, and M. Kurita, “Lip-reading in remote subjects: an attempt to quantify and separate ingestion, breathing and vocalisation in free-living animals using penguins as a model,” *Marine Biology*, vol. 140, pp. 17–27, 2002.

[186] S. Fossette, P. Gaspar, Y. Handrich, Y. L. Maho, and J.-Y. Georges, “Dive and beak movement patterns in leatherback turtles *Dermochelys coriacea* during internesting intervals in French Guiana,” *Journal of Animal Ecology*, vol. 77, no. 2, pp. 236–246, 2008.

[187] A. Markham and A. Wilkinson, “Ecolocate: A heterogeneous wireless network system for wildlife tracking,” in *International Joint Conferences on Computer, Information and Systems Sciences and Engineering (CISSE’07)*, December 2007.

[188] J. Porter, P. Arzberger, H.-W. Braun, P. Bryant, S. Gage, T. Hansen, P. Hanson, C.-C. Lin, F.-P. Lin, T. Kratz, W. Michener, S. Shapiro, and T. Williams, “Wireless sensor networks for ecology,” *BioScience*, vol. 55, no. 7, pp. 561–572, Jul. 2005.

[189] Y. Wang, S. Jain, M. Martonosi, and K. Fall, “Erasure-coding based routing for opportunistic networks,” in *WDTN ’05: Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*. New York, NY, USA: ACM, 2005, pp. 229–236.

[190] T. Liu and M. Martonosi, “Impala: a middleware system for managing autonomic, parallel sensor systems,” in *PPoPP ’03: Proceedings of the ninth ACM SIGPLAN symposium on Principles and practice of parallel programming*. New York, NY, USA: ACM, 2003, pp. 107–118.

[191] B. Thorstensen, T. Syversen, T.-A. Bjornvold, and T. Walseth, “Electronic shepherd - a low-cost, low-bandwidth, wireless network system,” in *MobiSys ’04: Proceedings of the 2nd international conference on Mobile systems, applications, and services*. New York, NY, USA: ACM, 2004, pp. 245–255.

[192] Mica2dot. [Online]. Available: http://www.xbow.com/Products/productsdetails.aspx?sid=73

[193] J. Sorber, A. Kostadinov, M. Brennan, M. Corner, and E. Berger, “eFlux: Simple Automatic Adaptation for Environmentally Powered Devices,” in *Seventh IEEE Workshop on Mobile Computing Systems & Applications (WMCSA’06)*. Los Alamitos, CA, USA: IEEE Computer Society, 2006, p. 50.

[194] T. Small, Z. J. Haas, A. Purgue, and K. Fristrup, “The shared wireless infostation model: A new ad hoc networking paradigm (or where there is a whale, there is a way),” in *Fourth ACM International Symposium on Mobile AdHoc Networking & Computing, 2003*, 2003.

[195] M. Radenkovic and B. Wietrzyk, “Wireless mobile ad-hoc sensor networks for very large scale cattle monitoring,” in *Sixth International Workshop on Applications and Services in Wireless Networks (ASWN’06), Berlin, Germany*, 2006, pp. 47–58.

[196] M. Radenkovic and B. Wietrzyk, “Mobile ad hoc networking approach to detecting and querying events related to farm animals,” in *IEEE International Conference on Networking and Services (IEEE ICNS’06), Silicon Valley, USA*, 2006, pp. 109–115.

[197] B. Wietrzyk, M. Radenkovic, and I. Kostadinov, “Practical MANETs for pervasive cattle monitoring,” in *Seventh International Conference on Networking (ICN 2008)*, vol. 0. Los Alamitos, CA, USA: IEEE Computer Society, 2008, pp. 14–23.

[198] T. Wark, P. Corke, P. Sikka, L. Klingbeil, Y. Guo, C. Crossman, P. Valencia, D. Swain, and G. Bishop-Hurley, “Transforming agriculture through pervasive wireless sensor networks,” *IEEE Pervasive Computing*, vol. 6, no. 2, pp. 50–57, 2007.

[199] Z. Butler, P. Corke, R. Peterson, and D. Rus, “From robots to animals: Virtual fences for controlling cattle,” *International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 485–508, 2006.

[200] P. Sikka, P. Corke, L. Overs, P. Valencia, and T. Wark, “Fleck - a platform for real-world outdoor sensor networks,” in *3rd International Conference on Intelligent Sensors, Sensor Networks and Information (ISSNIP)*, 2007.

[201] T. Wark, C. Crossman, W. Hu, Y. Guo, P. Valencia, P. Sikka, P. Corke, C. Lee, J. Henshall, K. Prayaga, J. O’Grady, M. Reed, and A. Fisher, “The design and evaluation of a mobile sensor/actuator network for autonomous animal control,” in *IPSN ’07: Proceedings of the 6th international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2007, pp. 206–215.

[202] P. Sikka, P. Corke, P. Valencia, C. Crossman, D. Swain, and G. Bishop-Hurley, “Wireless adhoc sensor and actuator networks on the farm,” in *IPSN ’06: Proceedings of the fifth international conference on Information processing in sensor networks*. New York, NY, USA: ACM, 2006, pp. 492–499.

[203] T. L. Dinh, W. Hu, P. Sikka, P. Corke, L. Overs, and S. Brosnan, “Design and deployment of a remote robust sensor network: Experiences from an outdoor water quality monitoring network,” in *LCN ’07: Proceedings of the 32nd IEEE Conference on Local Computer Networks*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 799–806.

[204] A. Mainwaring, D. Culler, J. Polastre, R. Szewczyk, and J. Anderson, “Wireless sensor networks for habitat monitoring,” in *WSNA ’02: Proceedings of the 1st ACM*

international workshop on Wireless sensor networks and applications. New York, NY, USA: ACM, 2002, pp. 88–97.

[205] R. Szewczyk, E. Osterweil, J. Polastre, M. Hamilton, A. Mainwaring, and D. Estrin, "Habitat monitoring with sensor networks," *Communications of the ACM*, vol. 47, no. 6, pp. 34–40, 2004.

[206] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2005, pp. 51–63.

[207] M. A. Batalin, M. Rahimi, Y. Yu, D. Liu, A. Kansal, G. S. Sukhatme, W. J. Kaiser, M. Hansen, G. J. Pottie, M. Srivastava, and D. Estrin, "Call and response: experiments in sampling the environment," in *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2004, pp. 25–38.

[208] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh, "Monitoring volcanic eruptions with a wireless sensor network," in *Second European Workshop on Wireless Sensor Networks*, 2005.

[209] L. Yu, N. Wang, and X. Meng, "Real-time forest fire detection with wireless sensor networks," in *Wireless Communications, Networking and Mobile Computing*, 2005.

[210] C. Hartung, R. Han, C. Seielstad, and S. Holbrook, "FireWxNet: a multi-tiered portable wireless system for monitoring weather conditions in wildland fire environments," in *MobiSys '06: Proceedings of the 4th international conference on Mobile systems, applications and services*. New York, NY, USA: ACM, 2006, pp. 28–41.

[211] K. Martinez, J. K. Hart, and R. Ong, "Environmental sensor networks," *Computer*, vol. 37, no. 8, pp. 50–56, 2004.

[212] H. Wang, C. E. Chen, A. Ali, S. Asgari, R. E. Hudson, K. Yao, D. Estrin, and C. Taylor, "Acoustic sensor networks for woodpecker localization," in *Advanced Signal Processing Algorithms, Architectures, and Implementations*, J. C. Bajard, N. Meloni, and T. Plantard, Eds., vol. 5910, 2005, pp. 80–91.

[213] W. Ji, P. C. L. White, and M. N. Clout, "Contact rates between possums revealed by proximity data loggers," *Journal of Applied Ecology*, vol. 42, no. 3, pp. 595–604, 2005.

[214] S. Prange, T. Jordan, C. Hunter, and S. D. Gehrt, "New radiocollars for the detection of proximity among individuals," *Wildlife Society Bulletin*, vol. 34, pp. 1333–1344, 2006.

[215] D. L. Swain and G. J. Bishop-Hurley, "Using contact logging devices to explore animal affiliations: Quantifying cow-calf interactions," *Applied Animal Behaviour Science*, vol. 102, pp. 1–11, 2007.

[216] M. E. Douglas, W. Ji, and M. N. Clout, "MateId: design and testing of a novel device for recording contacts between free-ranging animals," *Wildlife Society Bulletin*, vol. 34, pp. 203–207, 2006.

[217] K. Stratford, "Pers. comm."

[218] R. M. Nowak, *Walker's Mammals of the World*. Baltimore: Johns Hopkins University Press, 1999.

[219] Nordic semiconductor. [Online]. Available: http://www.nordicsemi.com/

[220] Zigbee alliance. [Online]. Available: http://www.zigbee.org/

[221] D. J. C. MacKay, *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003.

[222] S. Milgram, "The small world problem," *Psychology Today*, vol. 2, pp. 60–67, 1967.

[223] J. Krause, D. P. Croft, and R. James, "Social network theory in the behavioural sciences: potential applications," *Behavioral Ecology and Sociobiology*, vol. 62, pp. 15–27, 2007.

[224] S. R. Sundaresan, I. R. Fischhoff, J. Dushoff, and D. I. Rubenstein, "Network metrics reveal differences in social organization between two fission - fusion species, Grevys zebra and onager," *Oecologia*, vol. 151, pp. 140–149, 2007.

[225] J. Vicente, R. J. Delahay, N. J. Walker, and C. L. Cheeseman, "Social organization and movement influence the incidence of bovine tuberculosis in an undisturbed high-density badger *Meles meles* population," *Journal of Animal Ecology*, vol. 76, no. 2, pp. 348–360, 2007.

[226] S. B. Merrill and L. D. Mech, "The usefulness of GPS Telemetry to study wolf circadian and social activity," *Wildlife Society Bulletin*, vol. 31, no. 4, pp. 947–960, 2003.

[227] P. C. Cross, J. O. Lloyd-Smith, J. A. Bowers, C. T. Hay, M. Hofmeyr, and W. M. Getz, "Integrating association data and disease dynamics in a social ungulate: bovine tuberculosis in african buffalo in the kruger national park," *Ann. Zool. Fennici*, vol. 41, pp. 879–892, 2004.

[228] H. F. Simon Chamaille-Jammes, Marion Valeix, "Managing heterogeneity in elephant distribution: interactions between elephant population density and surface-water availability," *Journal of Applied Ecology*, vol. 44, pp. 625–633, 2007.

[229] N. Patwari, J. Ash, S. Kyperountas, I. Hero, A.O., R. Moses, and N. Correal, "Locating the nodes: cooperative localization in wireless sensor networks," *IEEE Signal Processing Magazine*, vol. 22, no. 4, pp. 54–69, July 2005.

[230] P. Zhang and M. Martonosi, "Locale: Collaborative localization estimation for sparse mobile sensor networks," in *Information Processing in Sensor Networks IPSN'08*, 2008.

[231] L. G. Roberts, "ALOHA packet system with and without slots and capture," *SIGCOMM Comput. Commun. Rev.*, vol. 5, no. 2, pp. 28–42, 1975.

[232] A. Markham and A. Wilkinson, "The adaptive social hierarchy: A self organizing network based on naturally occurring structures," in *1st International Conference on Bio-Inspired mOdels of NEtwork, Information and Computing Systems (BIONETICS), Cavelese, Italy*, 11-13 December 2006.

[233] A. Markham and A. Wilkinson, "A biomimetic ranking system for energy constrained mobile wireless sensor networks," in *Southern African Telecommunications, Networks and Applications Conference (SATNAC 2007)*, September 2007.

[234] A. Markham, "Adaptive social hierarchies: From nature to networks," in *Bio-inspired Computing and Communication Networks*, Y. Xiao and F. Hu, Eds. To be published by Auerbach Publications, Taylor & Francis Group, December 2008, 2008.

[235] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, 2002.

[236] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: a survey," *IEEE Journal of Wireless Communications*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[237] K. Akkaya and M. Younis, "A survey on routing protocols for wireless sensor networks," *Ad Hoc Networks*, vol. 3, pp. 325–349, 2005.

[238] D. Jea, A. Somasundara, and M.Srivastava, "Distributed computing in sensor systems," in *Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks*. Springer Berlin / Heidelberg, 2005, pp. 244–257.

[239] H. Wang, D. Estrin, and L. Girod, "Preprocessing in a tiered sensor network for habitat monitoring," *Journal on Applied Signal Processing*, vol. 4, pp. 392–401, 2003.

[240] W. R. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive protocols for information dissemination in wireless sensor networks," in *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. New York, NY, USA: ACM, 1999, pp. 174–185.

[241] L. Ying and Y. Haibin, "Energy adaptive cluster-head selection for wireless sensor networks," in *Sixth International Conference on Parallel and Distributed Computing Applications and Technologies (PDCAT'05)*, 2005, pp. 634–638.

[242] I. Gupta, D. Riordan, and S. Sampalli, "Cluster-head election using fuzzy logic for wireless sensor networks," in *3rd Annual Communication Networks and Services Research Conference (CNSR'05)*, 2005, pp. 255–260.

[243] I. Chase, "Behavioral sequences during dominance hierarchy formation in chickens," *Science*, vol. 216, no. 4544, pp. 439–440, 1982.

[244] E. O. Wilson, *Sociobiology: The New Synthesis*. Harvard University Press, 1975.

[245] K. Tsuchida, T. Saigo, N. Nagata, S. Tsujita, K. Takeuchi, and S. Miyano, "Queen-worker conflicts over male production and sex allocation in a primitively eusocial wasp," *International Journal of Organic Evolution*, vol. 57, no. 10, pp. 2356–73, Oct 2003.

[246] B. Holldobler and E. Wilson, *The Ants*. Cambridge: Harvard University Press, 1990.

[247] T. Schjelderup-Ebbe, "Bietrage zur sozialpsychologie des haushuhns," *Zeitschrift fur Psychologie*, vol. 88, pp. 225–252, 1922.

[248] S. Cote, "Determining social rank in ungulates: a comparison of aggressive interactions recorded at a bait site and under natural conditions," *Ethology*, vol. 106, pp. 945–955, 2000.

[249] D. Greenberg-Cohen, P. Alkon, and Y. Yom-Tov, "A linear dominance hierarchy in female nubian ibex," *Ethology*, vol. 98, no. 3, pp. 210–220, 1994.

[250] S. Altmann, "A field study of the sociobiology of rhesus monkeys," *Annals of the New York Academy of Sciences*, vol. 102, pp. 338–435, 1962.

[251] W. M. Jackson, "Can individual differences in history of dominance explain the development of linear dominance hierarchies?" *Ethology*, vol. 79, pp. 71–77, 1988.

[252] I. D. Chase, C. Tovey, D. Spangler-Martin, and M. Manfredonia, "Individual differences versus social dynamics in the formation of animal dominance hierarchies," *Proceedings of the National Academy of Sciences*, vol. 99, no. 8, pp. 5744–5749, 2002.

[253] W. Hamilton, "Geometry for the selfish herd," *Journal of Theoretical Biology*, vol. 31, pp. 295–311, 1971.

[254] H. Kruuk, "Predators and anti-predator behaviour of the black-headed gull *Larus ridibundus*," *Behaviour Supplements 11*, vol. 11, pp. 1–129, 1964.

[255] C. Carbone, J. D. Toit, and I. Gordon, "Feeding success in african wild dogs: Does kleptoparasitism by spotted hyenas influence hunting group size?" *The Journal of Animal Ecology*, vol. 66, no. 3, pp. 318–326, 1997.

[256] S. Creel and N. M. Creel, "Communal hunting and pack size in african wild dogs, *Lycaon pictus*," *Animal Behaviour*, vol. 50, pp. 1325–1339, 1995.

[257] M. Kendall, *Rank Correlation Methods*. New York: Hafner Publishing Co, 1955.

[258] J. Havil, *Gamma: Exploring Euler's Constant*. Princeton University Press, 2003.

[259] W. R. Gilks, S. Richardson, and D. J. Spiegelhalter, Eds., *Markov Chain Monte Carlo in Practice*. Chapman & Hall, 1996.

[260] C. Perkins and E. Royer, "Ad-hoc on-demand distance vector routing," in *IEEE Mobile Computing Systems and Applications*, 1999, pp. 90–100.

[261] D. Aldous and J. Fill, *Reversible markov chains and random walks on graphs (monograph in preparation)*, 2006, http://stat-www.berkeley.edu/users/aldous/RWG/book.html.

[262] C. E. Jones, K. M. Sivalingam, P. Agrawal, and J. Chen, "A survey of energy efficient network protocols for wireless networks," *Wireless Networks*, vol. 7, no. 4, pp. 343–358, 2001.

[263] A. Dunkels, F. Osterlind, N. Tsiftes, and Z. He, "Software-based on-line energy estimation for sensor nodes," in *EmNets '07: Proceedings of the 4th workshop on Embedded networked sensors*. New York, NY, USA: ACM, 2007, pp. 28–32.

[264] A. Lindgren, A. Doria, and O. Schelen, "Probabilistic routing in intermittently connected networks," in *Service Assurance with Partial and Intermittent Resources*. Springer Berlin/Heidelberg, 2003.

[265] M. Musolesi, S. Hailes, and C. Mascolo, "Adaptive routing for intermittently connected mobile ad hoc networks," in *Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05)*, 2005, pp. 183–189.

[266] M. Musolesi and C. Mascolo, "Evaluating context information predictability for autonomic communication," in *WOWMOM '06: Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 495–499.

[267] X. Chen and A. L. Murphy, "Enabling disconnected transitive communication in mobile ad hoc networks," in *Workshop on Principles of Mobile Computing, colocated with PODC'01*, 2001, pp. 21–27.

[268] T. Spyropoulos, K. Psounis, and C. Raghavendra, "Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility," in *Fifth IEEE International Conference on Pervasive Computing and Communications Workshops*. Los Alamitos, CA, USA: IEEE Computer Society, 2007, pp. 79–85.

[269] C. Mascolo and M. Musolesi, "SCAR: context-aware adaptive routing in delay tolerant mobile sensor networks," in *IWCMC '06: Proceedings of the 2006 international conference on Wireless communications and mobile computing*. New York, NY, USA: ACM, 2006, pp. 533–538.

[270] C. Frank and K. Römer, "Algorithms for generic role assignment in wireless sensor networks," in *SenSys '05: Proceedings of the 3rd international conference on Embedded networked sensor systems*. New York, NY, USA: ACM, 2005, pp. 230–242.

[271] K. Römer, C. Frank, P. J. Marrón, and C. Becker, "Generic role assignment for wireless sensor networks," in *EW11: Proceedings of the 11th workshop on ACM SIGOPS European workshop*. New York, NY, USA: ACM, 2004.

[272] O. Angel, A. Holroyd, D. Romik, and B. Virag, "Random sorting networks," *Advances in Mathematics*, vol. 215, no. 10, pp. 839–868, 2007.

[273] P. B. Jeon and G. Kesidis, “Pheromone-aided robust multipath and multipriority routing in wireless manets,” in *PE-WASUN ’05: Proceedings of the 2nd ACM international workshop on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*. New York, NY, USA: ACM, 2005, pp. 106–113.

[274] F. Ducatelle, G. D. Caro, and L. M. Gambardella, “Ant agents for hybrid multipath routing in mobile ad hoc networks,” in *Second Annual Conference on Wireless On-demand Network Systems and Services (WONS’05)*. Los Alamitos, CA, USA: IEEE Computer Society, 2005, pp. 44–53.

[275] G. Di Caro, F. Ducatelle, and L.M.Gambardella, “Anthocnet: An ant-based hybrid routing algorithm for mobile ad hoc networks,” in *Parallel Problem Solving from Nature - PPSN VIII*. Springer Berlin/Heidelberg, 2004, pp. 461–470.

[276] Sirf Location Systems, “Sirf Star III GPS receiver module,” 2008. [Online]. Available: www.sirf.com

[277] K. J. Mills, B. R. Patterson, and D. L. Murray, “Effects of variable sampling frequencies on GPS transmitter efficiency and estimated wolf home range size and movement distance,” *Wildlife Society Bulletin*, vol. 34, no. 5, pp. 1463–1469, December 2005.

[278] G. Schaller and R. Keane, *The Serengeti Lion; A Study of Predator-Prey Relations*. Chicago: University of Chicago Press, 1972.

[279] K. J. Astrom and B. Bernhardsson, “Systems with lebesgue sampling,” in *Directions in Mathematical Systems Theory and Optimization*, A. Rantzer and C. I. Byrnes, Eds. Springer, 2003, pp. 1–13.

[280] E. Foxlin, “Pedestrian tracking with shoe-mounted inertial sensors,” *IEEE Computer Graphics and Applications*, vol. 25, no. 6, pp. 38–46, 2005.

[281] R. Jirawimut, P. Ptasinski, V. Garaj, F. Cecelja, and W. Balachandran, “A method for dead reckoning parameter correction in pedestrian navigation system,” *Instrumentation and Measurement, IEEE Transactions on*, vol. 52, no. 1, pp. 209–215, Feb 2003.

[282] S. Beauregard and H. Haas, “Pedestrian dead reckoning: A basis for personal positioning,” in *3rd Workshop on Positioning, Navigation and Communication (WPNC ’06)*, 2006.

[283] C. Randell, C. Djiallis, and H. Muller, “Personal position measurement using dead reckoning,” in *Proc. 7th IEEE International Symposium on Wearable Computers (ISWC)*, 2003, pp. 166–173.

[284] M.B.Bennett, “Fast locomotion of some kangaroos,” *J Zoo Lond*, vol. 212, pp. 457–464, 1987.

[285] R. M. Alexander and A. Jayes, “A dynamic similarity hypothesis for the gaits of quadrupedal animals,” *J Zoo Lond*, vol. 201, pp. 135–152, 1983.

[286] N. Heglund, C. Taylor, and T. McMahon, “Scaling stride frequency and gait to animal size: mice to horses,” *Science*, vol. 186, pp. 1112–1113, 1974.

[287] N. C. Heglund and C. R. Taylor, “Speed, stride frequency and energy cost per stride: how do they change with body size and gait?” *Journal of Experimental Biology*, vol. 138, pp. 301–318, 1988.

[288] R. M. Alexander, “Terrestrial locomotion,” in *Mechanics and Energetics of Animal Locomotion*, G. Goldspink, Ed. Chapman and Hall, London, 1977, pp. 168–203.

[289] u-blox, Switzerland, “LEA-4P data sheet.” [Online]. Available: http://www.u-blox.com/

[290] Analog Devices, “ADXL103 uniaxial accelerometer datasheet,” 2005.

[291] Microchip Technology, “PIC18LF4620 microcontroller data sheet.” [Online]. Available: www.microchip.com

[292] T. H. Witte, K. Knill, and A. M. Wilson, “Determination of peak vertical ground reaction force from duty factor in the horse (*Equus caballus*),” *The Journal of Experimental Biology*, vol. 207, pp. 3639–3648, 2004.

[293] T. Garland, “The relation between maximal running speed and body mass in terrestrial mammals,” *Journal of Zoology*, vol. 199, p. 157170, 1983.

[294] E. C. Ifeachor and B. W. Jervis, *Digital Signal Processing: A Practical Approach*. Prentice Hall, 2002.

[295] J. Kim, H. Jang, D. Hwang, and C. Park, “A step, stride and heading determination for the pedestrian navigation system,” *Journal of Global Positioning Systems*, vol. 3, pp. 273–279, 2004.

[296] S. Haykin, *Adaptive Filter Theory*, 2nd ed. Prentice Hall, New Jersey, 1991.

[297] J. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[298] V. Cerny, “A thermodynamical approach to the travelling salesman problem: an efficient simulation algorithm,” *Journal of Optimization Theory and Applications*, vol. 45, pp. 41–51, 1985.

[299] R. Fletcher and C. Reeves, “Function minimization by conjugate gradients,” *The Computational Journal*, vol. 7, pp. 149–154, 1964.

[300] C. M. Bishop, *Neural Networks for Pattern Recognition*. Oxford University Press, 1985.

[301] B. Widrow and M.E.Hoff, “Adaptive switching circuits,” *IRE WESCON*, vol. 4, pp. 96–104, 1960.

[302] P. Werbos, “Beyond regression: New tools for prediction and analysis in the behavioural sciences,” Ph.D. dissertation, Harvard University, Nov. 1974.

[303] Microchip technology, inc. [Online]. Available: www.microchip.com

[304] ST Microelectronics, "LIS3L02AS4 2g/6g triaxial accelerometer datasheet," 2004.

[305] S. Corporation, "SanDisk Secure Digital Card Product Manual, Version 1.9," December 2003.

[306] V. Gervasi, S. Brunberg, and J. E. Swenson, "An individual-based method to measure animal activity levels: A test on brown bears," *Wildlife Society Bulletin*, vol. 34, no. 5, pp. 1314 – 1319, 2006.

[307] W. I. Sellers and R. H. Crompton, "Automatic monitoring of primate locomotor behaviour using accelerometers," *Folia Primatol*, vol. 75, pp. 279–293, 2004.

[308] M. Schwager, D. M. Anderson, Z. Butler, and D. Rus, "Robust classification of animal tracking data," *Computers and Electronics in Agriculture*, vol. 56, pp. 46–59, 2007.

[309] M. L. Morrison, B. G. Marcot, and R. William Mannan, *Wildlife-Habitat Relationships Concepts and Applications*. Island Press, 2006.

[310] M. A. T. Committee, "The MultiMediaCard System Specification, Version 3.1," June 2001.

[311] SD Card Association, "SD Simplified Physical Layer Specification v2.00," 2006.

[312] ST MicroElectronics, "L6920DB Switching Regulator." [Online]. Available: www.stmicroelectronics.com

[313] Polygon Technologies (Cape Town, South Africa). [Online]. Available: www.pteq.net

[314] Sipex Corporation, "SP3232 3V RS232 transceiver," 2006. [Online]. Available: www.sipex.com

[315] AeroComm, "AC4868 868 MHz OEM radio transceiver datasheet," 2005.

[316] ST Microelectronics, "STLM20 precision analog temperature sensor datasheet," 2008.

[317] Doxygen: Automatic documentation generator. [Online]. Available: http://www.stack.nl/~dimitri/doxygen

[318] Ongava research centre. [Online]. Available: www.ongava.com/research

[319] Google earth. [Online]. Available: http://earth.google.com/

[320] Sarantel, "GeoHelix-SMP helical GPS antenna datasheet," 2007. [Online]. Available: www.sarantel.com