A Local Optimal User Position System for Indoor Wireless Devices

Author
Nicola Lenihan

Supervisor
Dr. Sean McGrath

Submitted for the Degree of
Masters of Engineering

University of Limerick
May 2004

Declaration

Author: Nicola Lenihan

Title: Local Optimal Positioning for Wireless Devices

I hereby declare that this thesis is entirely my own work and that it has not been submitted as an exercise to any other university.

Nicola Lenihan

To my parents, Anne and Patrick Lenihan

Abstract

The Local Optimal User Position (LOUP) system proposed in this project determines the location of the user in a Wireless Local Area Network (WLAN) and then ascertains the closest position that will improve the service to the user. The location of the user is identified using signal strength information. This information together with the propagation model of the locality allows the system to determine if the user is located in a low capacity area, for e.g. behind a cement pillar. In this situation, the system directs the user to move to another location, which will have a better signal strength value, in order to receive a better class of service.

The local optimal solution is achieved by discovering signal strength for a given area around the user and then selecting the highest signal strength location. The first step of the process is to create a radio map of the area. Once this is created, the location of the user can be determined using the signal strength received at the user device and the known fixed positions of the user. The signal strength information received by wireless devices is readily available by the WLAN software on the devices, e.g. the Client Manager from Orinoco [ORI]. After the location of the user is derived, the local optimal position can be obtained. Finally, the user is directed from their current position to the new optimal position by the shortest path. This is achieved by the A* shortest path search algorithm [HNR68].

Two novel algorithms are proposed as part of this thesis. The Multiple Matrix Correlation (MMC) algorithm determines the location of the user by comparing the radio map of the area with the signal strengths received at the user device from all transmitters in the area. The Local Optimal Position (LOOP) algorithm identifies the closest position to the current receiver position that is within a given threshold. This threshold can be set by the user at system start-up depending on the class of service desired by the user.

Acknowledgements

I would like to take this opportunity to thank my parents for all their help in the past few years. Only they know how much they have done for me in the past. And I look forward to having the opportunity to repay them in some way in the future. Also to thank my brother and sister, Gerry and Jackie, for always being there for me and to Vinny for his encouragement and motivation when I needed it most.

I especially would like to thank Dr. Sean McGrath for his continuous guidance, assistance and encouragement as my academic supervisor and also as a friend. I would like to thank all the Wireless Access Research group members for all the help and support especially Mike (for his helpful criticisms and great advice), Ronan and Edu (for the great soccer games), Maria (for the female company), Neville (for the maps) and Ian (for the pints of course).

TABLE OF CONTENTS

List of Figures

List of Tables

List of Abbreviations

Acronyms

1SM	One-Slope Model
AGPS	Assisted Global Positioning System
AoA	Angle of Arrival
AP	Access Point
AS	Anti-Spoofing
BS	Base Station
CDMA	Cell Division Multiple Access
CES	Coast Earth Station
CI	Cell Identity
COO	Cell of Origin
CRC	Cyclic Redundancy Check
DGPS	Differential Global Positioning System
DNS	Decca Navigation System
DoD	Department of Defense
DSSS	Direct Sequence Spread Spectrum
E911	Enhanced 911
EC	European Commission
EGNOS	European Geostationary Navigation Overlay Service
E-OTD	Enhanced Observed Time Difference
E-FLT	Enhanced Forward Link Triangulation
EPE	Ekahau Positioning Engine
ESA	European Space Agency
FAA	Federal Aviation Administration
FCC	Federal Communications Commission
FDMA	Frequency Division Multiple Access
FHSS	Frequency Hopping Spread Spectrum
GCC	Galileo Control Center
GDOP	Geometric Dilution of Precision
GEO	Geostationary Communication Satellite
GIC	GPS Integrity Channel
GLONASS	Global Orbiting Navigation Satellite System
GNSS	Global Navigation Satellite Systems
GPS	Global Positioning System
GSS	Galileo Sensor Station
GUS	Ground Uplink Station
HTAP	Hybrid TOA/AOA Positioning
IEEE	Institute of Electrical and Electronic Engineers
IESSG	Institute of Engineering Surveying and Space Geodesy
INMARSAT	International Maritime Satellite Organization
IODE	Issue of Data Ephemeris
INS	Inertial Navigation System
IPDL	Idle Period Down Link
IR	Infrared
ISI	Intersymbol Interference
ISM	Industrial, Scientific and Medical
ITU	International Telecommunication Union
ITU-R	International Telecommunication Union – Radio Communications Sector
LAAS	Local Area Augmentation System
LAM	Linear Attenuation Model
LCS	Location Services
LF	Location Fingerprinting
LGF	LAAS Ground Facility

LMU	Location Management Unit
LOCUS	Location of Cellular Users for Emergency Services
LOOP	LOcal Optimal Position
LOS	Line of Sight
LOUP	Local Optimal User Position
MEMS	Micro-Electrochemical Sensors
MEO	Medium Earth Orbital
MMC	Multiple Matrix Correlation
MS	Mobile Station
MST	Minimum Spanning Tree
MWM	Multi Wall Model
NAVSTAR	Navigation System and Ranging
NELS	Northwest European Loran-C System
NIC	Network Interface Card
NLOS	Non Line of Sight
NNSS	Navy Navigation Satellite System
OFDM	Orthogonal Frequency Division Multiplexing
OTDOA	Observed Time Difference of Arrival
PDA	Personal Digital Assistant
PDF	Probability Distribution Function
PL	Pathloss
PPS	Precise Positioning Service
PSAP	Public Safety Answering Point
RAIM	Receiver Autonomous Integrity Monitoring
RF	Radio Frequency
RSS	Received Signal Strength
RTCM SC-104	Radio Technical Commission for Maritime Services Special Committee 104
RTD	Real Time Difference
SA	Selective Availability
SAR	Search and Rescue
SES	Ship Earth Station
SPS	Standard Positioning Service
SNR	Signal to Noise Ratio
TA	Timing Advance
TDOA	Time Difference of Arrival
TM-UWB	Time Modulated Ultra Wideband
TOA	Time of Arrival
UDRE	User Differential Range Error
USCG	U.S. Coast Guard
UTC	Coordinated Universal Time
VHF	Very High Frequency
WAAS	Wide Area Augmentation System
WLAN	Wireless Local Area Network
WMS	Wide Area Master Station
WRS	Wide Area Reference Station

1 Introduction

This thesis proposes a novel method to combine user location and channel information in order to determine the best location and service delivery for a mobile wireless user in an indoor environment. Based on the indoor location of the wireless user and the propagation information for that location a service estimate for the user can be determined and corrective action taken if necessary.

This is achieved through the Local Optimal User Position (LOUP) system. The LOUP system proposed in this thesis performs the steps necessary to determine the best local position for the user and then informs the user of the shortest path to that position. Previous research in the area has focused on indoor location positioning. This thesis applies the location position information to improve the service to the wireless user.

Location positioning is the ability to find the geographical location of a mobile device, e.g. ship, airplane, mobile phone, etc. This has always been an important goal for mankind, back as far as the days when the stars were used as guides on ships. Position location for wireless telecommunications has become of great interest both technically and commercially in recent years. There are many factors driving this advancement, including the Federal Communications Commission (FCC) in the US, which mandates that all wireless communication carriers must be able to report the location of all 911 callers with an accuracy of 125m in 67% of all cases. This mandate is called E911 and phase 2 is due to be completed by 2005, [HAT02].

The use of Wireless Local Area Network (WLAN) technology to determine the position of a user in an indoor environment was first envisioned by Microsoft Research and is called RADAR [BP00]. RADAR determines the location of the user by combining empirical measurements with propagation modelling. Similar systems that use WLAN technology to infer location include the Halibut system by Stanford University [KMR01], the Nibble system by UCLA [CCK01], Amulet by the University of Rochester [HAR02], a system developed by the University of Linz [FWN01] and a system developed by Carnegie Mellon University [SSS01].

There are also two commercial products, which determine the location of a device from a wireless network, currently available to the general public produced by Ekahau and WhereNet. Ekahau offers the Ekahau Positioning Engine (EPE) that locates laptops or PDA devices within 1 meter (3.5 feet) in less than a second [EKA02]. EPE is a java based software-only solution that runs on Windows NT/2000/XP, WinCE 3.0 and Pocket PC 2002. An Ekahau Manager application records site data and tracks devices, with the ability to show a floor plan that displays the location of each wireless node.

WhereNet is another company that offers a line of positioning solutions [WHE02]. The WhereLan Access Point supplies an 802.11b wireless LAN that makes available a positioning infrastructure capable of supporting various types of location-based applications. WhereNet's Visibility Server Software is a software package that provides tools necessary to manage the assets and resources.

An alternative system based on Infrared (IR) is the Active Badge system [WHF92]. In this system a badge worn by a person emits an IR signal periodically, which is picked up by sensors placed at known locations around the building and relayed to the location management software. This software then uses triangulation to determine the location of the badge.

Another RF based system is the 3D-iD RF tag system built by PinPoint Corporation [WL98]. Antennas placed around the building emit a 2.4GHz RF signal. Tags, which are attached to people or equipment, respond to this signal with a 5.8GHz signal along with an identification code. Various antennas receive this response and send it back to a centralized controller, which triangulates the result to determine the location of the tag.

The tag and IR based systems suffer the major drawback of needing specialized hardware installation and maintenance, whereas the WLAN systems make use of an already available infrastructure. While much research has focused on locating a user within a wireless network, this has not been applied to providing a better service to the user. This thesis proposes such a system to improve the service to the user.

The Local Optimal User Position (LOUP) system proposed in this thesis determines the location of the user in a Wireless Local Area Network (WLAN) and then ascertains the closest position that will improve the service to the user. The location of the user is identified using signal strength information. This information together with the propagation model of the locality allows the system to determine if the user is located in a low capacity area, e.g. behind a cement pillar. In this situation, the system directs the user to move to another location, which will have a better signal strength value, in order to receive a better class of service.

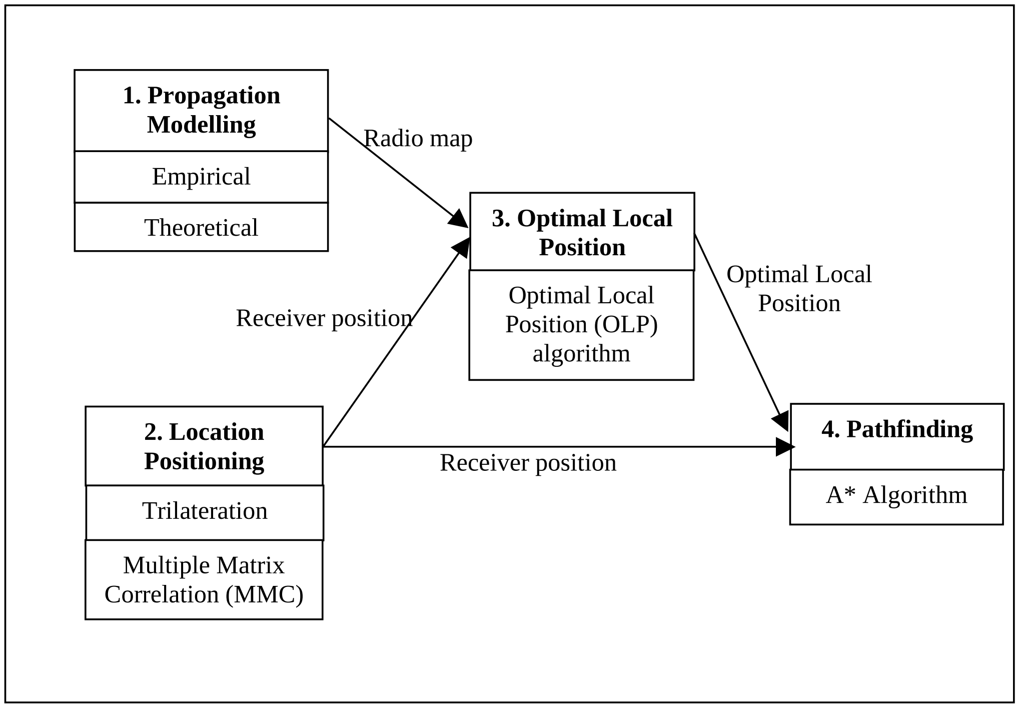
The LOUP system is broken into four different stages. These are:

1. The propagation modelling stage
2. The location position stage
3. The Optimal Local Position stage
4. The pathfinding stage

The system along with the four stages and their interaction together is shown in Figure 1.1. Each of the stages is introduced below.

Firstly, the system constructs the propagation models of the given indoor environment. A propagation model gives an estimate of the signal strength received at a distance from the transmitter. This is achieved in two ways, theoretically and empirically. The empirical method is based on a location fingerprinting method, which measures the signal strength values received from the transmitters in the area, at each location around the given area. The theoretical method uses indoor signal strength propagation model formulas to calculate the signal strength at each location in the area. The output of this stage is the radio map of the given area, which shows a map of the area and the given signal strength that is received at each location around the area.

Figure 1.1 – Local Optimal User Position (LOUP) System



After creating the radio map of the area the user's location can be determined. As well as using triangulation to solve for user location, this thesis introduces a novel algorithm, called Multiple Matrix Correlation (MMC) to improve the location estimation. The output of this stage is the current receiver location. This algorithm was developed and implemented for this thesis in Matlab.

Triangulation has been used predominately to solve position for this type of application. The signal strength received at the user device from each of the transmitters is known. Using this information the distance to each of the transmitters can be calculated by working backwards with the signal propagation formula. From these distances and the known fixed positions of the transmitters, triangulation gives the user location.

The MMC algorithm takes the signal strengths received at the user device from each of the transmitters and correlates them with the databases of the signal strengths from each of the transmitters in the area. These databases are already available from the propagation modelling stage, as previously discussed. The closest match is given as the user location.

The next step is to determine the local optimal location for the user. This thesis introduces an innovative algorithm called Optimal Local Position (OLP), to solve this problem. At this stage the user location is ascertained and the received signal strength from each of the transmitters given from the user device. OLP first determines if the signal strength at the current location is within a given acceptable threshold, for e.g. within 60% of the transmitted power lever. If the signal strength is within this threshold then the user does not have to move. Otherwise, the nearest location that is within this given threshold is computed. The algorithm searches the area immediately around the user position and finds the maximum signal strength in this area. This signal strength is then tested to determine if it is within the given threshold. If so, the algorithm finishes and returns the coordinates of the new position. If the maximum signal strength is not within the given threshold, the search area is widened. This will continue until a value within the threshold is found. The output of this stage is a Optimal Local Position.

Finally the system will inform the user how best to move from their current position to the new position, taking obstructions into account. This is achieved using the heuristic search algorithm, A*, which finds the shortest path between two points [HNR68]. The inputs to this stage are the receiver location and the Optimal Local Position. The output is the shortest path from the receiver location to the Optimal Local Position.

The threshold as used in the OLP algorithm, can be set by the user at the system start-up. The value input by the user would depend on the class of service desired by the user see Table 1.1. For example if the user requires a Class A service, the threshold would be set to be 75%, whereas for a user only requiring a Class D service the threshold might be set to 45%.

Table 1.1 – Classification of Service for IEEE802.11b

Class of Service	**Data Rate**
Class A	11Mbits/sec
Class B	5.5Mbits/sec and over
Class C	2Mbits/sec to 5.5Mbits/sec
Class D	1Mbits/sec to 2Mbits/sec
Class E	No service

1.1 Summary of Thesis

Chapter 2 focuses on the necessary background for radio propagation. This chapter introduces some of the main phenomena that affect the propagation of radio. It examines the indoor propagation environment before describing some of the better-known indoor propagation models. Following this, the chapter examines the effect of WLAN technology on radio propagation is examined.

Chapter 3 of this thesis presents a review of the current state of location positioning technologies. This chapter is broken into four classifications, satellite based systems, non-satellite based radio navigation systems, cellular based systems and indoor based location systems.

Pathfinding algorithms are introduced in Chapter 4. The basics of graph theory used in search algorithms in explained, before detailing several different pathfinding algorithms. The chapter finishes with an explanation of the A* algorithm, which is the path finding algorithm used in this thesis.

Chapter 5 gives the implementation details of the Local Optimal User Position (LOUP) system. This chapter gives specific details on creating the radio map of the area, on finding the position of the user, on determining the Optimal Local Position using the OLP algorithm and on moving from the current location position to the

Optimal Local Position. The Wireless Access Research Laboratory in the University of Limerick is the target environment.

The performance of the Local Optimal User Position (LOUP) system in the target environment is presented in Chapter 6, along with some comparison results of different techniques used throughout the system.

Chapter 7 presents the thesis conclusions and proposes some further areas for research.

2 Radio Propagation

2.1 Introduction

The ability to communicate with people, without the use of wires has evolved remarkably since Guglielmo Marconi first demonstrated its use in 1897 [BRI01]. Since then wireless communication has become ubiquitous and has been adopted by people throughout the world. In contrast to wireline communications, which relies on an infrastructure of cabling, radio can be used anywhere. Whereas the limiting factor with wireline networks is the network infrastructure, the capacity of radio networks is determined by the frequency spectrum available and the physical attributes of radio waves.

As a radio transmitter transmits information, the radio wave signal carrying the information travels over the air interface. This is called radio propagation. An understanding of the principles and issues associated with radio propagation is necessary for the design and maintenance of a wireless network. Radio propagation models are used to estimate the signal strength of a radio signal at a given distance from a transmitter. They work by taking into account the different factors that effect the radio propagation. Different propagation models are used for both indoor and outdoor environments. As this thesis is based on an indoor environment, this chapter will examine several different indoor propagation models. The study of outdoor propagation models is outside the scope of this thesis, for more information on the topic, see [RAP96].

Before examining the indoor radio propagation models, this chapter reviews the characteristics of propagation and the factors that cause degradation of the radio signal, including pathloss, multipath and shadowing. The chapter concludes by investigating some of the effects of radio propagation on one radio technology, namely IEEE 802.11b, which is the technology used in this thesis.

2.2 The Radio Spectrum

Electromagnetic waves exist with an extremely wide range of wavelengths and corresponding frequencies. The radio spectrum is generally considered to lie in the range of frequencies between 3kHz and 3000THz [ITU01]. Within the radio spectrum, the frequencies are divided into frequency bands, e.g., the VHF or Very High Frequency band is made up of the frequencies between 30MHz and 300MHz. The full list of frequency bands is given in Table 2.1. Most radio wave communication occurs in the range of zero to 275kHz, the range covered by frequency allocations in the

International Telecommunication Union, ITU Radio Regulations. The ITU is the standards body for the United Nations, with the ITU-R as the Radiocommunications Sector.

Table 2.1 Nomenclature for Frequency Bands

Band No.	**Symbol**	**Frequency Range**	**Wavelength**
	ELF	<300Hz	>1000 km
3	ULF	300 Hz – 3 kHz	1000 – 100 km
4	VLF	3 – 30 kHz	100 – 10 km
5	LF	30 – 300 kHz	10 – 1 km
6	MF	300 kHz – 3 MHz	1 km – 100 m
7	HF	3 – 30 MHz	100 – 10 m
8	VHF	30 – 300 MHz	10 – 1 m
9	UHF	300 MHz – 3 GHz	1 m – 100 mm
10	SHF	3 – 30 GHz	100 – 10 mm
11	EHF	30 – 300 GHz	10 – 1 mm
12		300 GHz – 3 THz	1 mm – 100 μm
13		3 – 30 THz	100 μm – 10 μm
14		30 – 300 THz	10 μm – 1 μm
		300 – 3000 THz	1 μm – .1 μm

2.3 Basic Principles

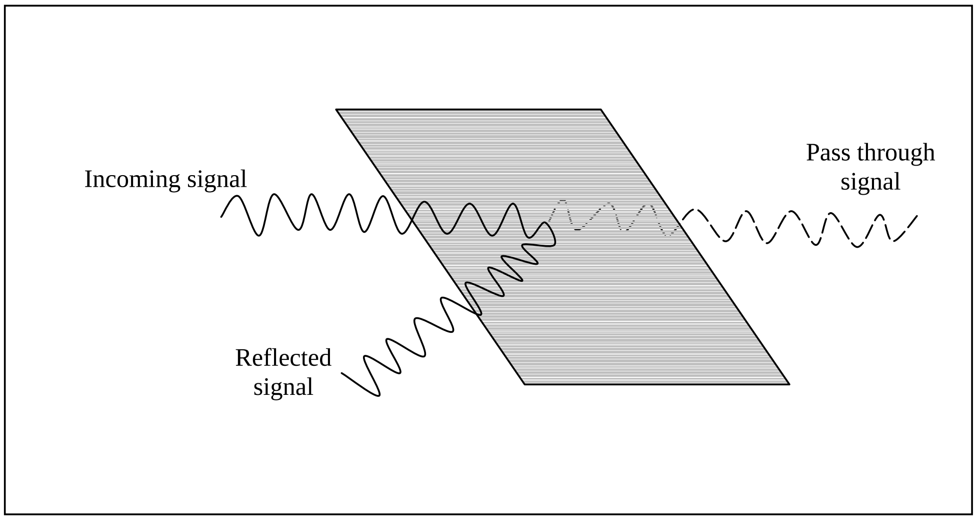
In radio communications, information is transmitted over air using electromagnetic waves. Radio propagation conditions limit the area, which can be covered by the transmitter and the maximum data rate of the system. An accurate understanding of the radio propagation channel is imperative for reliable propagation modelling of an area.

The three basic mechanisms of radio propagation are reflection, diffraction and scattering [RAP96]. All three of these phenomena cause radio signal distortion and give rise to signal fades as well as additional signal propagation loss. As a radio device moves throughout a coverage area, the three mechanisms have an impact on the instantaneous received signal in different ways. For example, in a line of sight (LOS) situation, reflection is likely to have a more consequential effect than scattering or diffraction.

2.3.1 Reflection

Reflection occurs when a propagating electromagnetic wave impinges upon an object, which has very large dimensions when compared to the wavelength of the propagating wave [WAL02]. The propagating wave strikes a surface and will either be absorbed or reflected. The reaction depends on the signal and on the physical properties of the surface. The signal properties are the arriving incident angle, orientation and wavelength. The physical properties of the surface include texture and material composition. Walls, floors, ceiling and furniture are major contributors to reflection.

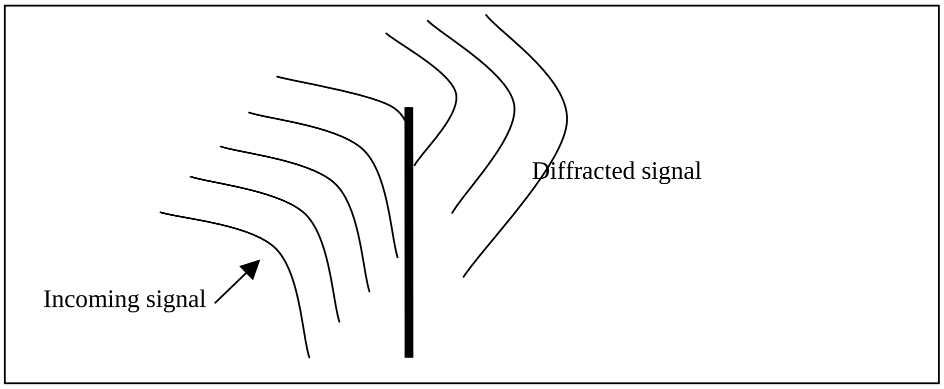
Figure 2.1 Reflection of a Signal



2.3.2 Diffraction

Diffraction occurs when the radio path between the transmitter and receiver is obstructed by a surface that has sharp irregularities (edges) [STEIN]. Diffraction occurs when the obstacles met by the waveform are impenetrable by the radio waves. The secondary waves resulting from the obstructing surface are present throughout the space and even behind the obstacle, giving rise to a bending of waves around the obstacle, even when a line-of-sight path does not exist between transmitter and receiver. At high frequencies, diffraction, like reflection, depends on the geometry of the object, as well as the amplitude, phase, and polarization of the incident wave at the point of diffraction.

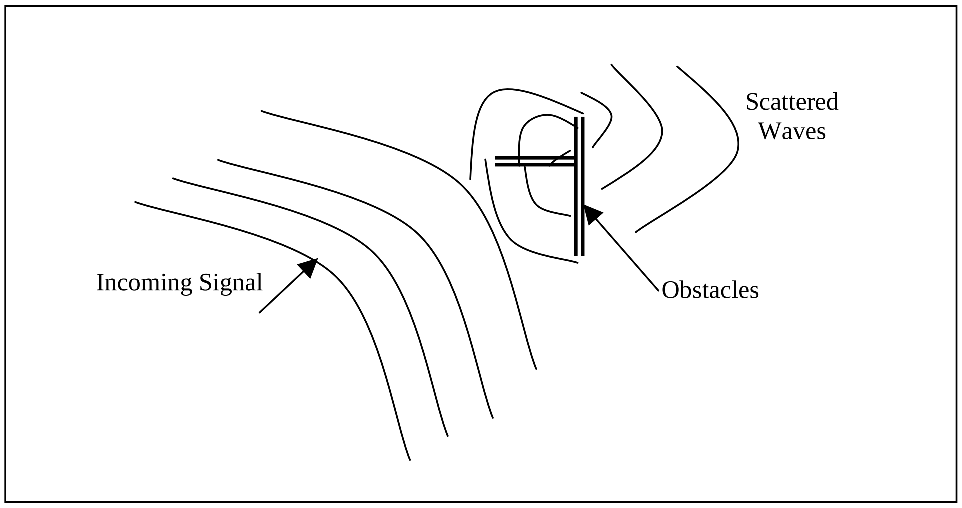
Figure 2.2 Diffraction of a Signal



2.3.3 Scattering

Scattering occurs when the medium through which the wave travels consists of objects with dimensions that are small compared to the wavelength, and where the number of obstacles per unit volume is large [RAP96]. The propagated wavefront breaks apart into many directions. The resultant signal scatters in all directions adding to the constructive and destructive interference of the signal. Scattered waves are produced by rough surfaces, small objects, or by other irregularities in the channel. In practice, foliage, street signs, and lampposts induce scattering in a mobile communications system.

Figure 2.3 Scattering of a Signal



2.3.4 Indoor versus Outdoor Propagation

With the proliferation of mobile phone, Personal Digital Assistant (PDA's), laptops and other personal communication systems, there is a great deal of interest in characterising the radio propagation inside buildings. The indoor channel differs from the outdoor channel in two main aspects; the distances covered inside a building are a lot smaller and the obstacles and variability of the environment are much greater. Propagation within buildings is influenced by such factors as the layout of the building, the construction materials, the building type and the number and type of obstacles such as office furniture. Indoor radio propagation is dominated by the same mechanisms as outdoor, i.e. reflection, refraction and scatter. However, there is many other factors to take into account such as floors, walls, furniture, open and closed doors. Even the number of people in a building will affect the propagation of the radio waves.

In an outdoor system the ultimate purpose is to ensure efficient coverage of the required area and to avoid interference. In an indoor system, the extent of coverage is well defined by the size of the building itself. The short range of coverage, along with high frequency bands means that small changes in the environment, such as opening a closed door, can have substantial effects on the radio propagation. For planning an indoor radio environment, it is necessary to estimate the number and location of transmitters necessary to provide the best coverage in the area and to estimate all potential interference to the system. Indoor propagation models that represent the propagation characteristics in the environment are used in order to achieve this [COS91].

2.4 Degradation of Radio Waves

As the signal propagates from the transmitter to the receiver it loses power or attenuates in strength. This causes errors in received messages, loss of information and poor signal quality. There are three main types of signal degradation caused by propagation.

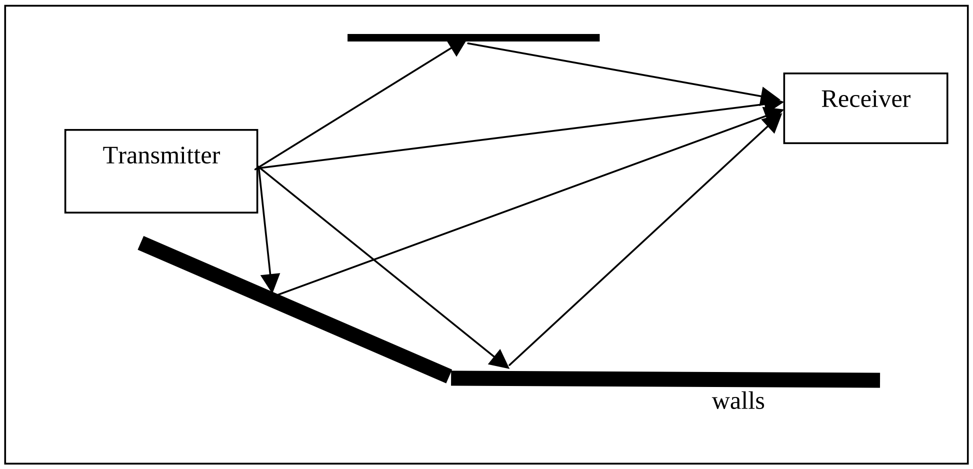
- Multipath
- Shadowing
- Pathloss

In the next section, shadowing and fading are examined before moving on to the more relevant reason for signal degradation, pathloss due to distance.

2.4.1 Multipath

In a realistic radio environment, waves reach a receiver from many different paths and directions [STEIN]. This is called multipath and is shown in Figure 2.4. When multiple signal propagation paths exist, both LOS and non Line of Sight (NLOS), the actual received signal is a sum of all the signals arriving at a given point in time. Some of these multipath signals will aid the direct path signal, while others will subtract from it.

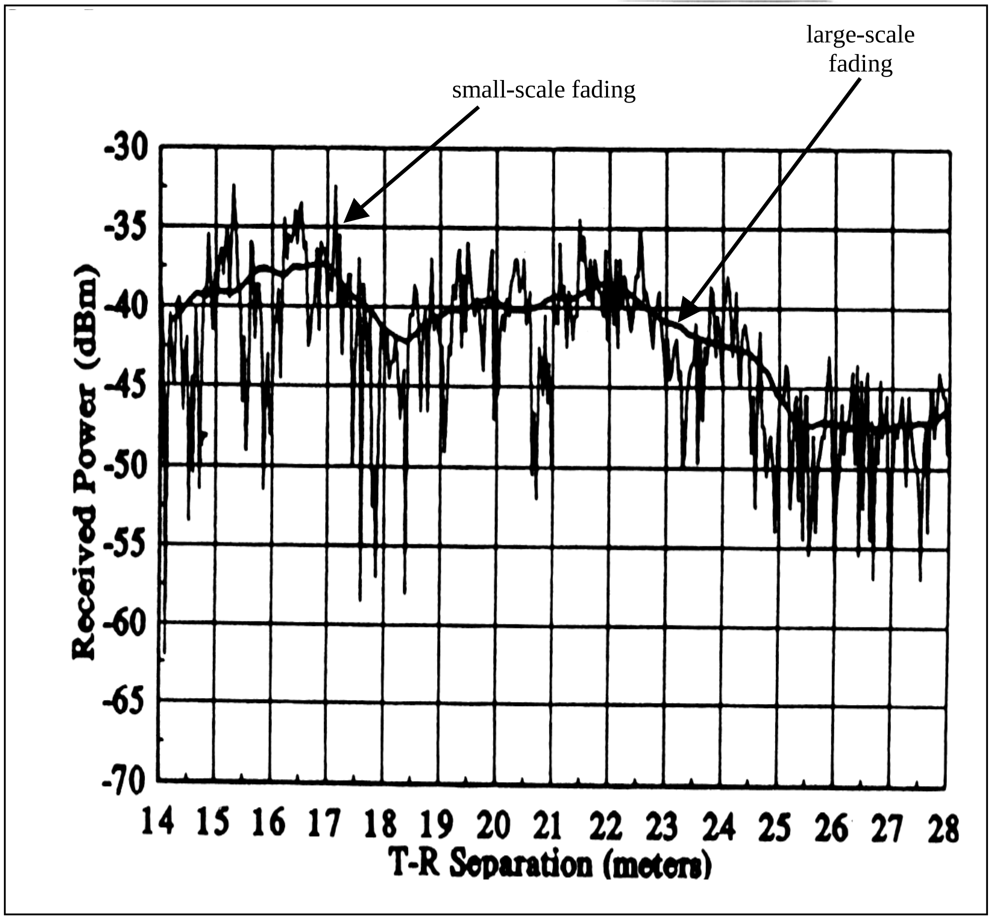
Figure 2.4 – Multipath Propagation



Multipath has two effects, one positive and one negative. The positive effect is that it allows communication in a NLOS situation. On a negative side, it causes many signal impairments, including, delay spread, interference between paths coming from the transmitter, and random frequency modulation due to the Doppler shifts on the different paths. Multipath effects also vary depending on the location of the antennas as well as the type of antennas used. Multipath propagation gives rise to fading of the signal, where the signal is increasing and decreasing in strength. This can be either small scale fading or large-scale fading. Figure 2.5 shows both small scale fading and large scale fading as a function of received power over distance from transmitter.

Large-scale propagation is defined as small variation over large distances. It predicts the average signal strength for a coverage area of a transmitter over a large transmitter receiver, TR, distance. Propagation models that characterize rapid fluctuations over small distances or short time durations are defined as small-scale fading, e.g. the metal blades of an oscillating fan cause small scale fading. These give the variability of the signal strength at close proximity to a particular location. Small scale fading gives rise to a spreading in frequency due to the Doppler effect as well as a spreading in time, due to the multipath signals arriving out of synchronisation.

Figure 2.5 Large Scale and Small Scale Fading, received power versus distance.

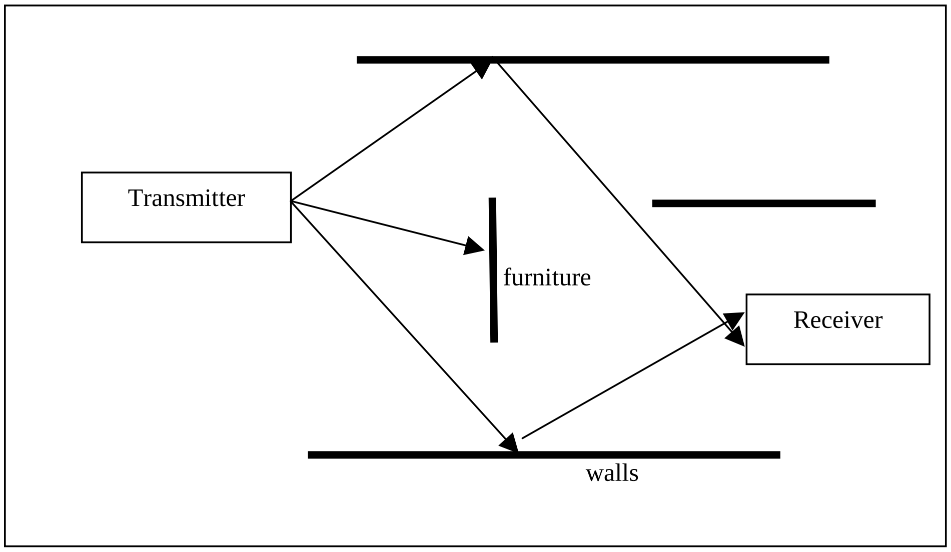


The rapid fluctuations of the signal and the small scale fading can be seen in Figure 2.5. As a receiver is moved further away from a transmitter, the average received signal strength at the receiver will decrease, as is seen in Figure 2.5 by the large scale fading line. It is this characteristic of the radio channel that large scale propagation models rely on.

2.4.2 Shadowing

Another degradation factor that affects radio signals is shadowing. This is the loss that is caused by obstacles, e.g. buildings or mountains in outdoor propagation and walls and furniture in indoor propagation. Shadowing hinders the Line Of Sight (LOS) path and facilitates the Non-Line Of Sight (NLOS) paths, giving rise to multipath, see Figure 2.6.

Figure 2.6 Shadowing caused by Obstacles in an Indoor Environment



As can be seen from Figure 2.6 the furniture obstructs the line of sight path. However, the signal can still reach the receiver through the non line of sight path by reflecting off the walls and other obstructions in the building. Some examples of loss due to obstacles are given in Table 2.2 [STEIN].

Table 2.2 – Signal attenuation due to Obstacles at 2.4GHz

Obstacle	**Loss in signal strength (dB)**
window brick wall	2
metal frame glass wall into building	6
office wall	6
metal door in office wall	6
cinder block wall	4
metal door in brick wall	12.4
brick wall next to metal door	3

2.4.3 Pathloss

As the distance between the transmitter and the receiver increases, the signal power decreases, even in a vacuum. This is known as Pathloss, PL. It is defined as an average of the difference in dB between the transmitted power, Pt, and the received power, Pr. Path loss is difficult to calculate accurately in an indoor environment. This is due to the variety of physical barriers and materials within the building. The path between receiver and transmitter is usually blocked by walls, ceilings and other obstacles, leading to a NLOS path. There are many mathematical models, which help in the estimation of pathloss. The most simple of these models is the Freespace path loss model. This chapter also examines the COST231 indoor models; the One Slope Model (1SM), the Multi Wall Model (MWM) and the Linear Attenuation Model (LAM).

2.5 Indoor Propagation Modelling

2.5.1 Freespace pathloss

Freespace pathloss is the attenuation of a signal radiated from an isotropic source through a vacuum. It is fundamental to indoor path loss analysis, as it is referenced in many other pathloss models. To define free space propagation from an isotropic point source transmitting a power of Pt, a distance d from the source, the power transmitted is spread uniformly on the surface of a sphere of radius d. The power flux density, S, is given by [WAL02]:

$$S_r = \frac{P_t}{4\pi d^2} \qquad \text{(Eqn.2.1)}$$

If the receiving antenna is placed at a distance d from the source, it will receive a power proportional to its effective area Ae. That is Pr = SAe. The power received, Pr, by the antenna is then:

$$P_r = \frac{P_t A_e}{4\pi d^2} \qquad \text{(Eqn.2.2)}$$

Ae, the effective area of the antenna is given by equation 2.3, where Gr is the gain of the receiving antenna and λ is the wavelength in meters.

$$A_e = \frac{G_r \lambda^2}{4\pi} \qquad \text{(Eqn.2.3)}$$

The wavelength of the signal can be calculated by c/f, where c is the speed of light, 3x10m/s, and f is the frequency of the signal. For example, the wavelength of the 2.4GHz sinusoid used in 802.11 WLAN [IEE97] is .125 meters. Coupling equation 2.2 and 2.3 gives:

$$P_r = \frac{P_t G_r \lambda^2}{[4\pi d]^2} \qquad \text{(Eqn.2.4)}$$

By replacing the isotropic source with a transmitting antenna, gain given by Gt, gives:

$$P_r = \frac{P_t G_r G_t \lambda^2}{[4\pi d]^2} \qquad \text{(Eqn.2.5)}$$

In decibels the pathloss is given by:

$$PL(dB) = 10\log_{10}\left(\frac{P_r}{P_t}\right) = -10\log_{10}\left(\frac{G_r G_t \lambda^2}{[4\pi d]^2}\right) \qquad \text{(Eqn.2.6)}$$

By applying a reference distance of 1 meter, assuming unity gain at both antennas and using a frequency of 2.4GHz, the frequency range used in IEEE 802.11b to equation 2.6, the path loss from a reference distance d1 is obtained.

$$PL(d_1) = -10\log_{10}\left(\frac{.125^2}{[4\pi d]^2}\right) = 40.046dB \qquad \text{(Eqn. 2.7)}$$

When the received effect at a reference distance, d, e.g., PL(d) = 40.046dB, is known, the following equation can be used to find the received pathloss at a distance, d. This is the equation used in this thesis for the Freespace propagation model:

$$PL(d) = PL(d_1) + 20\log\left(\frac{d}{d_1}\right) = 40.046 + 20\log(d) \qquad \text{(Eqn.2.8)}$$

The free space propagation model is ideal and can only be applied to satellite systems and short-range line of sight propagation. Obstructed pathloss is much more difficult to predict as it has to account for many different scenarios and materials. Several different path loss models exist to describe unique indoor characteristics. These are based on the free space loss and the propagation phenomenon, reflection, diffraction and scattering, as well as the effects of different building types.

2.5.2 One Slope Model (1SM)

The 1SM is based on free space and takes into account the attenuation of the signal with distance, as well as the effect of the propagation environment. It assumes a linear dependence between the path loss and the logarithmic distance. This model predicts that the pathloss PL(d) measured in dB, at a distance, d from the transmitter will be [COS91]:

$$PL(d) = PL(d_o) + 10n\log_{10}\left(\frac{d}{d_0}\right)(dB) \qquad \text{(Eqn. 2.9)}$$

The reference path loss is given by PL(d0) and n is the path loss exponent. The reference distance should be in the far field of the transmitting antenna, typically 1km for outdoor urban environments, 100m for microcell systems and 1m for indoor environments, [ARY95]. Eqn. 2.7 calculates the pathloss at a reference distance of 1 m to be 40.046dB. The path loss exponent is an empirical constant that varies with the propagation environment; it is often measured empirically but can be derived theoretically. For a value of n = 2, the path loss is that of free space. Table 2.3 gives typical values for n [RAP96].

Table 2.3 Typical Path Loss Exponents

Environment	**Path Loss Exponent, n**
Free Space	2
Urban Area Cellular	2.7 to 4.0
Shadowed Urban Cellular	3 to 5
In building line of sight	1.6 to 1.8
In factory line of sight	1.6 to 2
In building one floor non line of sight	2 to 4
Obstructed in building	4 to 6
Obstructed in factories	2 to 3

The value used in this thesis is n = 3.5. The value given in Table 2.3 for in-building with NLOS and one floor is 2 to 4. This thesis is applied to one floor of a building, with mainly soft partitions. Many different references quote different values for n, for indoor obstructed buildings, [RBX97]. Therefore, the following approach was followed. Modelling was first carried out with a value of n = 3. This model was then compared to the empirical model. Afterward, many different values of n were tried until the value that best allows the model to approach the empirical model was found. The equation used in this thesis is then:

$$PL(d) = 40.046 + 35\log(d) \qquad \text{(Eqn.2.10)}$$

2.5.3 Multi-Wall Model (MWM)

The Multi-Wall Model is another pathloss model, which takes into account the free space loss as well as any floors and walls in the direct line of the transmitter and receiver. The Multi-Wall Model (MWM) is given as follows [COS91]:

$$PL(d) = PL(FS) + L_C + \sum_{i=1}^{I} k_{wi} L_{wi} + k_f^{\left[\frac{k_f+2}{k_f+1} - b\right]} L_f \qquad \text{(Eqn.2.11)}$$

where, PL(d) is the pathloss due to a distance, d,

PL(FS) is the Freespace loss between the transmitter and the receiver,

L_c is the constant loss,

k_{wi} is the number of penetrated walls of type i,

k_f is the number of floors,

L_{wi} is the loss of wall type i,

L_f is the loss between adjacent floors

b is the empirical parameter,

i is the number of wall types.

As the direct path attenuates due to passing through the floors and walls, the multipath become more significant with respect to the direct path. This is taken into account with the use of the constant loss. This value normally approaches zero. Also it has been observed, [TBL93] that the total floor loss is a non-linear function of the number of penetrated floors. This explains the introduction of the empirical parameter, b. The third term gives the total wall loss as a sum of the different wall types between transmitter and receiver. There is two wall types categorised, a light wall, L_{w1}, for example, plasterboard or partitioning, and a heavy wall, L_{w2}, which is made of concrete or brick. From [COS91] L_{w1} has a loss factor of 3.4dB, L_{w2} has a loss factor of 6.9dB and Lf has a loss factor of 18.3dB. The value of b, the empirical parameter was found to be .46, [TBL93].

For this thesis, the fourth term can be ignored as the area is concerned with one floor of the building. The walls in the testbed area are generally light internal walls, which would have a weighted average loss factor of 3.4dB. Table 2.4 gives the number of light internal walls used in the equation for this thesis.

Table 2.4 – Number of Walls

Meters between transmitter and receiver	**Number of walls**
0 – 3	0
4- 7	1
8 – 11	2
> 11	3

The reasoning behind the values for this table are twofold. Firstly, by looking at the given area, this roughly seems to correspond with the area. Secondly, most office block buildings contain offices which are approximately 3.5meters squared in size. There are also four cement pillars scattered throughout the area. With the positioning of the access points in this testbed, there is generally only one cement pillar at any time between the transmitter and receiver. This leads to the following equation:

$$PL(d) = 3.4t + 6.9 + 40.046 + 20\log_{10}(d) \qquad \text{(Eqn.2.12)}$$

where t is equal to the number of light walls between the transmitter and receiver.

2.5.4 Linear Attenuation Model (LAM)

This model, which is referenced in [COS91] and [KAR95] assumes that the excess path loss is linearly dependent on the distance, with an attenuation coefficient of α. The Freespace path loss is also included in this model. It is given as:

Eqn. 2.13)
$$PL(d) = PL(FS) + \alpha * d$$

The value of γ given at a frequency of 1800MHz from [COS91] can be seen in Table 2.5.

Table 2.5 Attenuation Coefficient for Linear Attenuation Model at 1800MHz

Environment	α
Dense – One Floor	.62
Dense - Multi Floor	2.8
Open	.22

As this testbed is conducted over one floor, a α of .62 is chosen. This leads to the following equation:

$$PL(d) = 40.046 + 20\log_{10}(d) + .62d \qquad \text{(Eqn. 2.13)}$$

2.6 Radio Propagation and WLAN

WLAN technologies are an example of radio communications technology. IEEE 802.11b, [IEE97], is the radio technology used in this thesis. It operates in the 2.4GHz Industrial, Scientific and Medical (ISM) equipment band. The frequency used is taken into account in the propagation models using the wavelength as seen in Eqn. 2.7. For more information on IEEE 802.11b WLAN technology see Appendix A.

WLAN technologies are susceptible to two main radio propagation problems. These are RF interference and multipath propagation. These problems give rise to a lower throughput and dropped packets as well as unpredictable propagation of the radio waves. Unpredictable propagation will cause propagation models of an area to be inaccurate.

2.6.1 RF Interference

RF interference involves the presence of unwanted, interfering RF signals that disrupt normal system operations. IEEE 802.11b WLAN technology operates in the 2.4GHz ISM band. This is an unlicensed operating band, which is shared with many other devices such as microwave ovens and cordless phones and as well as other technologies such as Bluetooth [BLU04]. As such, 802.11b is prone to interference from many sources, including other WLAN's operating on the same channel. For example. a microwave oven operating within 10 feet of an 802.11b device will cause the performance of the device to drop [HPI02].

A station which transmits/forwards and receives information in IEEE 802.11 standard is called an access point. An interfering signal from an outside source may appear to an access point as a bogus 802.11 station, which is transmitting a signal. The access point will then wait until the bogus station is finished before transmitting a packet. If the interfering signal transmits at the same time as an access point is transmitting a packet, the receiving station will receive the packet with errors and then will not reply with an acknowledgement. In this case, the transmitting access point will retransmit the packet, adding overheads to the network. In some cases, the 802.11b device will attempt to continue operation in the presence of RF interference by switching to a lower data rate.

Strong WLAN signals are less prone to RF interference. Therefore, by providing adequate coverage the effects of RF interference can be eliminated or reduced. Also tuning the access points to one of the other 11 channels available can help reduce the RF inference. The 2.4 GHz band used by 802.11b is prone to RF interference by several sources. However, the IEEE have also developed the 802.11a standard, which operates in the 5GHz operating range. This range has significantly less RF interference.

2.6.2 Multipath Propagation

Multipath propagation is another problem encountered by WLAN technology. It causes the information symbols represented in the WLAN signal to overlap. This causes the receiving device to demodulate the information incorrectly, giving bit errors in the packet. This is often referred to as intersymbol interference (ISI). The errors that this causes will be picked up by the Cyclic Redundancy Check (CRC) checksum. In response to this, the receiving access point will not send an acknowledgement to the transmitter, which causes the transmitter to resend the packet. The users of the device will then encounter lower throughput because of these retransmissions.

When comparing Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS) and Orthogonal Frequency Division Multiplexing (OFDM), different physical layers used by 802.11, DSSS used by 802.11b networks is the most prone to multipath propagation. FHSS, used by the original 802.11 standard, uses narrow channels, 1 MHz, and changes frequency regularly, making it difficult for multipath to occur. OFDM, which is used by 802.11a and 802.11g, also transmits information on many narrow subchannels, which also reduces the effects of multipath. DSSS transmits continuously over a wide channel, nearly 30MHz. This allows lower frequency elements of the DSSS signal to reflect off elements differently than the higher elements of the signal, thus increasing multipath.

Diversity is the use of two antennas at each radio in order to increase the odds of receiving a better signal on either of the antennas. These antennas are kept physically separate from the radio and from each other, to ensure that one will encounter less multipath propagation than the other. This method can also reduce the negative effects of multipath propagation.

2.7 Summary

Propagation of radio waves affect the quality of the transmission, the area of coverage and the maximum data rate achievable within that area. The main mechanisms that affect propagation are reflection, diffraction and scattering and these all have different effects on the propagation of the wave. In an indoor environment, such as used in this thesis, there is generally less distance between transmitter and receiver and more obstacles per square meter, than an outdoor environment.

Multipath is one of the main problems associated with modelling a radio propagation environment. This is when a transmitted radio wave reaches the receiver by more than one path. Shadowing is another problem associated with radio propagation and this arises from all the obstacles present between the transmitter and receiver. Pathloss, another problem encountered by radio propagation, is the loss of power of a signal as it travels over a distance.

Several indoor radio propagation models are presented in this chapter. These were the models that were used in this thesis and the results of each of the models can be seen in Chapter five. The first model is the Freespace propagation model, which is a good reference model for other models but has little relevance to an indoor environment. The One Slope Model (1SM) takes in the loss of the signal due to the propagation environment as well at the loss due to distance. This model gives a more realistic estimate of an indoor propagation environment.

The Multi-Wall Model (MWM) is based on the Freespace path loss model but also takes into account the loss due to floors and walls. It can give a more realistic estimate than the One Slope Model, however knowledge of the propagation environment is necessary to use this model. The Linear Attenuation Model (LAM) based on Freespace assumes that the excess pathloss is linearly dependent with the distance. The attenuation coefficient is the factor of linear dependency. The accuracy of this model is dependent on the attenuation coefficient.

Finally, this chapter examines some of the radio propagation issues associated with WLAN technologies, such as 802.11b, as used in this thesis. The main issues are RF interference and multipath propagation. Interference is caused by other devices transmitting in the same frequency range, such as Bluetooth enabled devices interfering with 802.11b devices. To overcome interference on WLAN's, ensure that WLAN devices are on individual channels, allocate adequate access points in the coverage area to ensure a strong signal throughout the area and/or move to a different frequency range device, for example switch to 802.11a, which operates in the 5GHz frequency band. DSSS as used by 802.11b is especially susceptible to multipath propagation because it transmits continuously on the same frequency over a wide channel. OFDM, which transmits information over narrow subchannels, reduces the effect of multipath. 802.11a and 802.11g both use OFDM as their physical layer. Another method to reduce multipath is the use of two antennas at each device, called diversity.

The radio technology in use for this thesis is as already stated IEEE 802.11b WLAN technology. This chapter has presented the indoor propagation models based on those recommended by [COS91]. The Freespace propagation model is included as a reference model. For more information on both outdoor and indoor propagation models, see [RAP96]. The models presented in this chapter will be used in this thesis in order to create the propagation models of the given area. These models were chosen for several reasons. They are easy to apply to WLAN technology, e.g. the 1SM only has one input parameter, the distance. They provide a variety of models, which can be compared to each other. Information regarding the input parameters is freely available through [COS91].

3 Location Positioning Technologies

3.1 Introduction

Location aware applications enable applications and devices to detect and make use of changing environmental conditions. These systems, also called context-aware, have knowledge of location, nearby people and accessible resources. Examples of such systems include an interactive tour guide in an art gallery. This type of application would identify the location of the user, the painting in front of them and nearby areas of importance, such as emergency exits. The application can then inform the user about the painting or direct them to the nearest exit etc.

In order to infer the location of a user, a review of location positioning technologies is carried out in this chapter. This review ranges through systems in the following areas: satellite based systems, radionavigation non-satellite based systems, cellular based systems and indoor based systems.

One of the first major technological breakthroughs in position location was Global Positioning System (GPS). The shortcomings of GPS have driven the advancement of many new technologies and improvements to GPS. These shortcomings include high power consumption and time to acquire the initial position. Of the new position location technologies that have appeared in recent years, most can generally be classed as satellite-based systems and cellular based systems. For the purpose of this document, four categories have been classified, satellite-based, cellular based, non-satellite radionavigation based and indoor based methods. Table 3.1 below also lists the availability classifications, which are referred to throughout the thesis.

This chapter gives an overview on position location technologies and how they are evolving. Section 2 is broken into two parts, the Global Navigation Satellite System (GNSS) and the improvements currently being carried out on the GNSS. In section 3, Radionavigation non-satellite systems, including Loran C are reviewed. Section 4 gives an overview of some of the cellular based position location systems that are currently in use or being developed. Section 5 takes a brief look at some of the indoor non-cellular based techniques available, namely, sensor technologies, WLAN positioning and Ultra-Wideband positioning.

Table 3.1 – Availability Classification

Class	Examples
Remote	Open sea, desert, no cellular coverage
Rural	Countryside, residential houses, roads
Urban	High building, constructions, urban canyons
Indoor	Wooden or concrete walls, metal roofed, office buildings
Underground	Concrete constructions, underground car parks

3.2 Satellite Based Systems

3.2.1 Global Navigation Satellite Systems (GNSS)

Transit was the first satellite based navigational system. It became fully operational in 1964. This system provided two-dimensional high-accuracy positioning. Positioning with Transit was accomplished by accumulating measurements of the Doppler shift of signals emitted from the satellites over a period of time [DP90]. Real-time navigation was not possible with this system. The satellites' orbits were low, at approximately 1000km. Transit was developed specifically for the U.S. navy and was not suitable for aircrafts or other high velocity users. Tsikada was the Russian equivalent to Transit [DG86]. Table 3.2 gives the timeframe on US and European GNSS.

Table 3.2 – US and European Global Navigation Satellite Systems

Time Frame	System
1964 -1996	Transit [1]
1995 - Present	NAVSTAR
Expected 2005	GPS Modernization
Expected 2008	Galileo

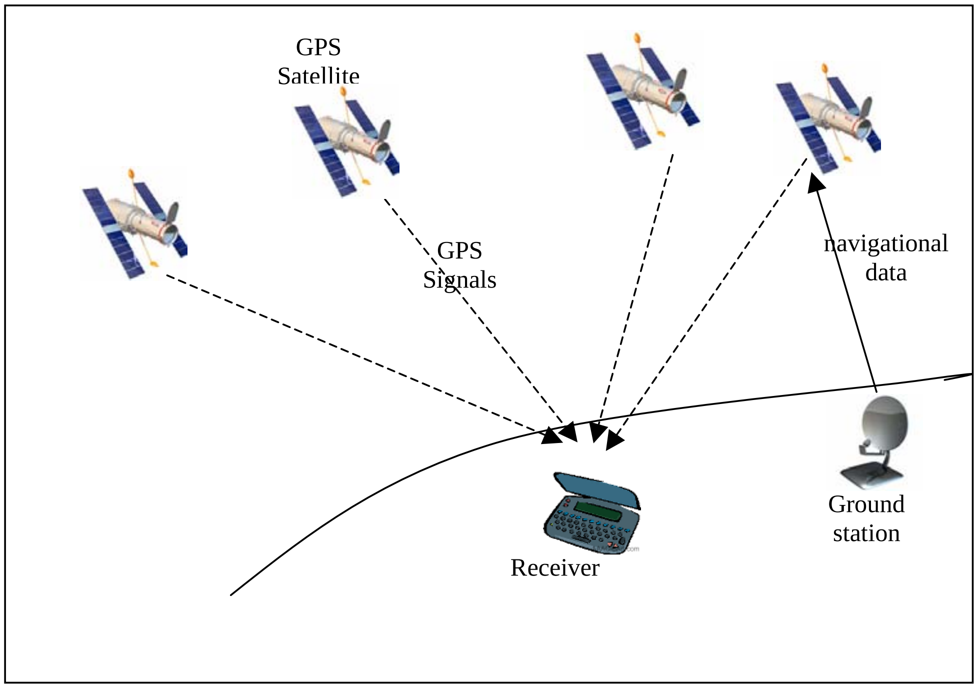
3.2.1.1 NAVSTAR GPS

NAVSTAR Global Positioning System (GPS) was designed specifically to overcome Transit's limitations. GPS is a fully operational navigational and position location system, which also provides Coordinated Universal Time (UTC). GPS is owned and maintained by the Department of Defence (DoD) in the United States [USN04]. The system is split into three segments, the control segment, the user segment and the satellite segment, see Figure 3.1. The control segment consists of five main control stations, which monitor the health and status of the satellites. The control stations upload navigation and other data including ephemeris data to the satellites. The

[1] Tsikada and GLONASS, the Russian equivalents have similar time frames.

satellite system consists of 24 satellites arranged in 6 orbital planes with 4 satellites per plane. The orbital planes are approximately circular, are separated by 60° and are about 20,200km in altitude. The user segment consists of GPS receivers, which receives the GPS signals and translates the signal into position, velocity and time.

Figure 3.1 – Global Positioning System (GPS)



The accuracy and availability of GPS is constantly being improved upon. In May 2003, the DoD improved the performance of GPS to provide accuracy within three meters for use during the Iraq war [BRE03]. This was accomplished by uploading more comprehensive and accurate ephemeris data to the satellites in the GPS system. This helped to better determine the position of the satellites, allowing the satellites to provide more accurate location information to receivers in precision weapons and aircraft as well as receivers used by ground forces. In reality however, receivers used by the public achieve accuracy between 5 and 6 meters 95% of the time [GSC04]. This is because the public only has access to one frequency channel L2, whereas the military also uses a second channel called L5 to improve the accuracy of the system.

3.2.1.2 GLONASS

Global Navigation Satellite System, GLONASS, is the Russian equivalent of GPS. The constellation consists of 3 orbital planes as opposed to 6, with 8 satellites in each plane. The ground stations, which monitor the satellites, are located throughout Russia. GLONASS is operated by Russia’s Ministry of Defence. GLONASS employs Frequency Division Multiple Access (FDMA), in which each satellite transmits on a different frequency. This technique allows the same ranging codes to be broadcast from each satellite. GLONASS, like GPS provides separate civil and military services. Due to economical difficulties, the GLONASS system has suffered severe degradation, leading to poor accuracy and availability. For more information on GLONASS, see [BDJ99].

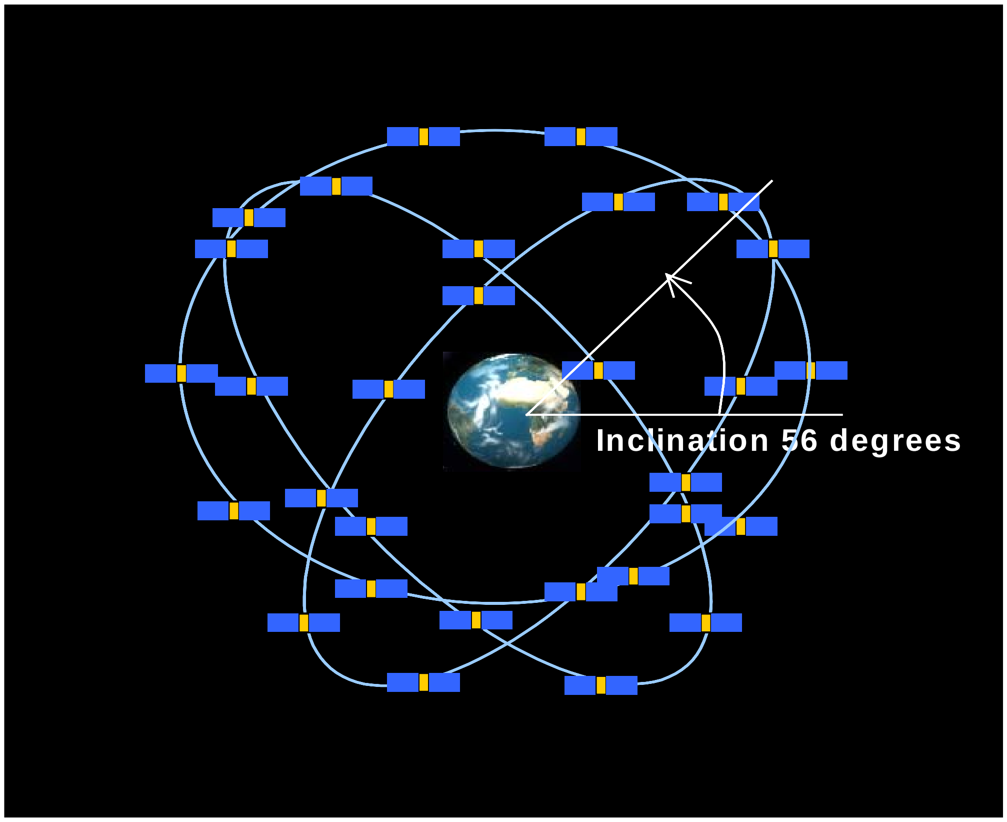
3.2.1.3 GALILEO

Galileo is Europe’s answer to GPS, to rival the existing US NAVSTAR system and the Russian GLONASS system. The European Commission (EC) and the European Space Agency (ESA) are promoting Galileo as a highly accurate global positioning service, which is under civilian control as opposed to military control. It will be usable as a standalone global system but will also be interoperable with NAVSTAR and GLONASS. Galileo will offer dual frequencies as standard, and therefore eliminate or reduce the effect of the ionosphere on the accuracy of the measurement. This will deliver real time positioning accuracy in the meter range. It is expected that Galileo will be fully operational in 2008 [BDG+00].

When it is fully deployed, Galileo will consist of 30 satellites, 27 operational and 3 active spares, in three Medium Earth Orbital (MEO) planes, see Figure 3.2. The orbital planes are 23,616km above the Earth’s surface. There will be 2 Galileo Control Centres (GCC's) on European ground to provide for the control of the satellites and the synchronization of the time signals. Information will be sent to the GCC from a global network of 20 sensor stations, GSS. The data will be sent from the GCC’s to the satellites using up-link stations, which will be located around the globe.

In an emergency situation Galileo will provide a global Search and Rescue (SAR) function, whereby if a receiver sends a distress signal, the satellites will receive it. On receiving this distress signal, the satellite transmits the distress signal to the Rescue Coordination Centre, which will then initiate the rescue operation. The satellite also sends information back to the receiver, informing the user that help is on the way. This is a new feature, which is not currently available on NAVSTAR or GLONASS.

Figure 3.2 – Orbital Planes for Galileo



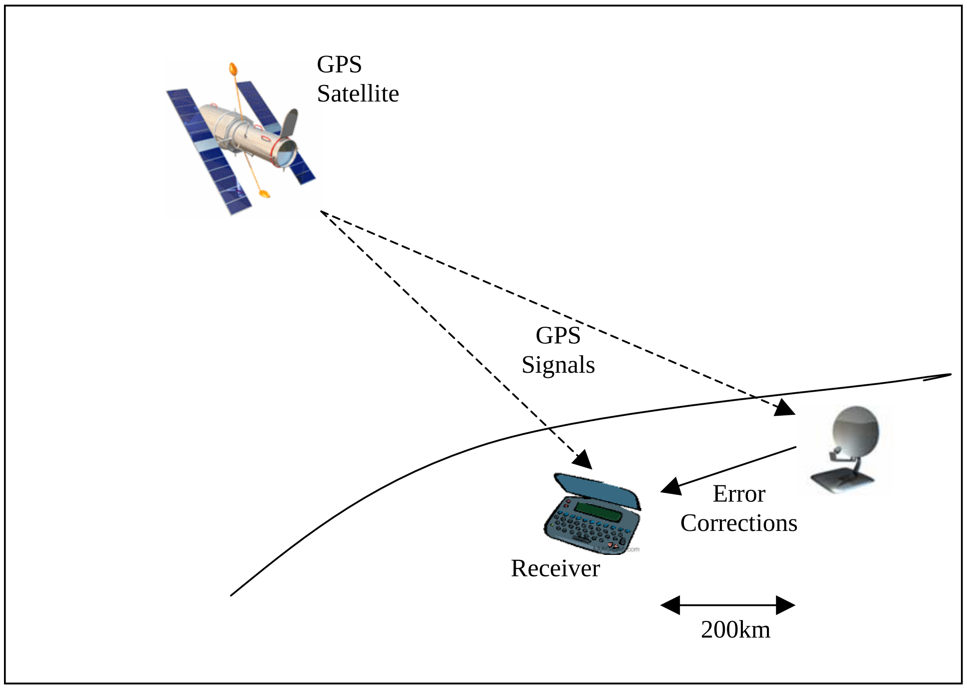
3.2.2 Improvements to Global Navigation Satellite Systems

3.2.2.1 Differential GPS (DGPS)

Transit, Tsikada, GPS and GLONASS were all developed for military use and do not allow great accuracy for civilian users. To achieve greater accuracy for civilian users a technique called Differential GPS or Differential GLONASS, was introduced. DGPS relies on the concept that errors in one location are similar to the errors for all locations within a given area, up to a few hundred kilometres. This is because within that given area, the signals are travelling through approximately the same atmosphere to reach the receivers [SMR03]. Typically DGPS can achieve accuracy between one and three meters, which is a significant improvement over GPS, [USC03].

Accuracy is improved by removing the correlated errors between two or more receivers using the same satellites. By using a reference receiver whose location is known and calculating the position of that receiver using a given set of satellites, a correction error can be calculated, see Figure 3.3. This correction error is known as the differential correction and is the difference between the real location and the calculated location. The differential correction is then transmitted to the roving receiver. The roving receiver can now use this error correction to improve the accuracy of the location. The differential error can be transmitted using two main methods, a geostationary satellite or a ground station.

Figure 3.3 – Differential GPS using a Ground Station



Ground Station DGPS

In this approach, the exact predetermined location of a ground station is obtained. The location of the ground station is then calculated using GPS and a certain set of satellites. The difference between this location and the exact location gives the differential correction. The differential correction for each satellite is then transmitted to the roving receiver. The roving receiver, uses the differential correction for each satellite that it is using to improve the accuracy of it's own GPS determined location.

Geostationary Satellite

This approach of DGPS uses a geostationary satellite to determine the error correction instead of the ground station. It estimates the error of each visible satellite and transmits the errors to the roving receiver. The ground station method of DGPS has been shown to be more accurate than the geostationary satellite [SC00]. This is mainly due to Ionospheric and Tropospheric delays [CKP+99]. However, the geostationary satellite method does allow for more coverage and better compatibility with the GPS roving receiver.

3.2.3 Assisted – Global Positioning System (AGPS)

Network assisted GPS (AGPS) overcomes some of the problems associated with GPS. AGPS uses a base station, which is also a GPS receiver. The base station can then fetch data from the GPS satellites and transmit this data to the receiver. As the base station is at a fixed location, it can act as a differential GPS station and transmit error corrections to the mobile station. This allows the mobile station to make timing measurements from the satellites without having to decode the GPS signals in order to calculate the location.

These enhancements provide accuracy of within 10 to 20m. By placing the base station on top of a tall building, the effects of multipath fading are partially mitigated. The use of the base station as a fixed GPS receiver also allows some indoor coverage and hence indoor position location. AGPS gives improvements over GPS in the following areas, time to first fix (TTFF), battery life, cost and indoor coverage, [SYR01]. For example, AGPS performance provides typical TTFF of 30 seconds compared to about 48 seconds for decoding the navigation data. This is a significant performance increase, which directly impacts customer acceptance and usability of AGPS technology [LDJ02].

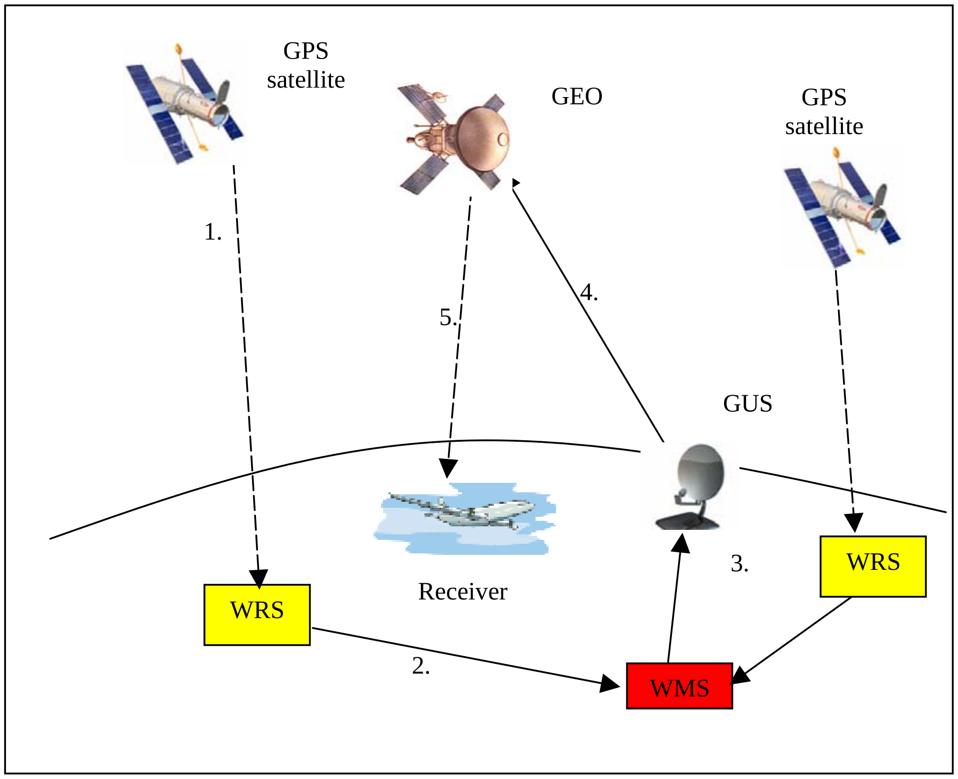
3.2.4 Augmentation Systems

The Federal Aviation Administration (FAA) in the US is working with the NAVSTAR product team to create a GPS-based navigation system to cover all phases of flight from flying en-route through to surface navigation. Two systems are being developed to achieve this; the wide area augmentation system (WAAS) and the local area augmentation system (LAAS). These systems will provide the accuracy, availability and integrity necessary to use GPS as the primary means of navigation in the US. The FAA is working with other organizations to promote the goal of a single integrated Global Navigation Satellite System, GNSS.

3.2.4.1 Wide Area Augmentation System (WAAS)

WAAS is designed to improve the accuracy and ensure the integrity of information coming from GPS satellites [IDA01]. It consists of a network of Wide area ground Reference Stations (WRS) situated across the US, see Figure 3.5. These stations receive signals from GPS satellites and identify errors in the GPS signals. The stations then relay this information to one of the Wide area Master Station (WMS). There are two WMS' in the US. The WMS then compute correctional information for specific geographical areas. This correctional message is then uplinked to a Geostationary Communications Satellite, GEO, using a ground uplink station, GUS. The GEO then transmits the correctional message on the same frequency as the GPS signal to the GPS/WAAS receivers. The accuracy of WAAS is shown to be approximately 2 meters or less [TRI01].

Figure 3.4 - Steps involved in Wide Area Augmentation System (WAAS)

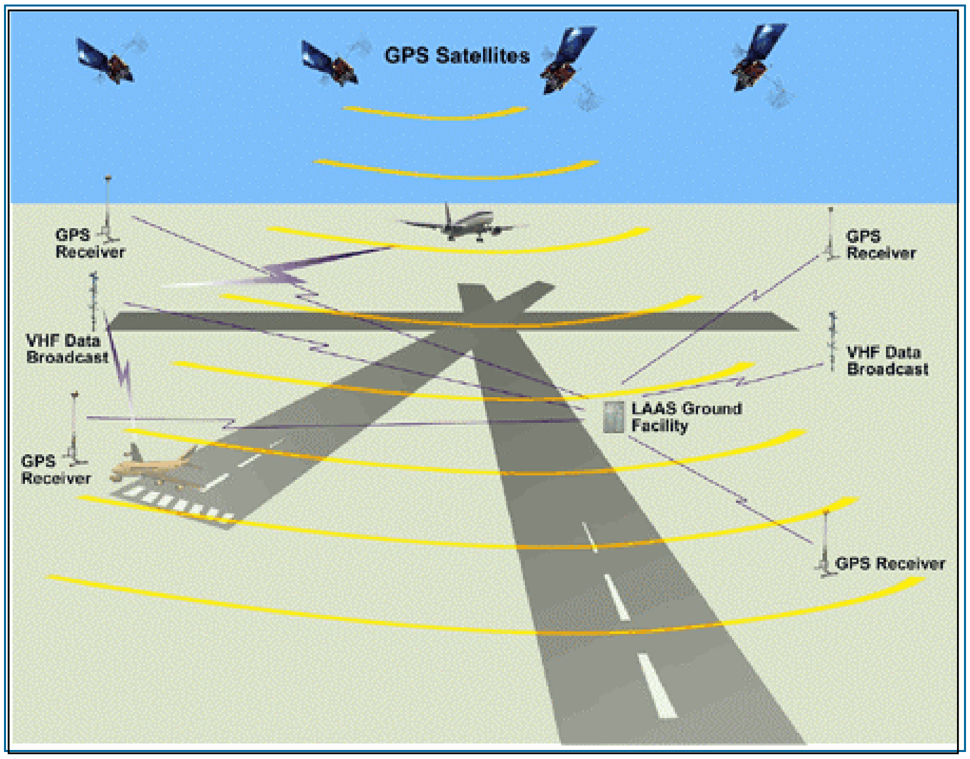


1. WRS receive GPS signal and identifies errors
2. WRS sends this information to the WMS
3. WMS compute correctional information and sends it to GUS
4. GUS sends correctional message to GEO
5. GEO sends correctional message to receiver on airplane

3.2.4.2 Local Area Augmentation System (LAAS)

LAAS is an augmentation to GPS that focuses on a small area, e.g. an airport [DOT02]. It yields extremely accurate positioning and navigation. The LAAS demonstrated accuracy is less than 1 meter in both the horizontal and vertical axis. The correction messages are broadcast on a Very High Frequency (VHF) radio data link from ground based transmitters. LAAS Ground Facility (LGF) equipment, which includes reference receivers and transmitters, is installed on the airport property. The reference receivers receive, decode and monitor GPS signals. Correction messages are then calculated and sent to the GPS/LAAS receivers, using the VHF radio link.

Figure 3.5 – Local Area Augmentation System (LAAS)



3.2.4.3 European Geostationary Navigational Overlay System (EGNOS)

European Geostationary Navigation Overlay System (EGNOS) is Europe’s first contribution to the Global Navigation Satellite System (GNSS) [TRA03]. It will augment the US’ NAVSTAR and Russia’s GLONASS systems, to provide for more reliable and accurate position location. It is expected to given accuracy of 5 meters or less [BMV03]. EGNOS is a joint project by the European Space Agency (ESA), the European Commission (EC), and Eurocontrol, the European organization for the safety of air navigation. It is scheduled to be fully operational by end of 2004. It is being developed with the intention that it will provide accuracy that is good enough to be used for safety critical applications.

EGNOS will consist of three geostationary satellites and a network of ground stations. The satellites will transmit information on the reliability and accuracy of the signals sent out by NAVSTAR and GLONASS. These signals will improve the accuracy of positions from about 20m to 5m, warn of disruptions to a satellite signal and inform users of errors in the position measurements. In effect, the EGNOS system will give a user location and tell the user of the error in the location. The signals sent by the EGNOS signals contain information on the position of each satellite, the accuracy of the atomic clocks on board the satellites and information on disturbances within the ionosphere that might affect the accuracy of the positioning measurement. The EGNOS receiver can then use this information to get a more accurate location and an accurate estimate of errors.

3.3 Non Satellite Radionavigation Systems

Radionavigation is defined as navigation using radio waves for determination of position or a line of position. Speed and direction may also be derived from positional information. Many Radionavigation Systems exist today. In this section, non-satellite radionavigation systems are explored. Two examples are identified and explained, Decca and Loran C.

3.3.1 Decca Navigation System (DNS)

The Decca Navigator System, generally known as Decca, is a hyperbolic radionavigation system, which is now almost obsolete. The system uses groups of at least three ground transmitter stations called chains. Each chain comprises one Master and two or three Slave stations, 80 - 110 km from the Master station. The accuracy of Decca ranges from 50 - 800m and decreases as the distance from the baseline increases. The accuracy is also subject to night and seasonal effects, which generally reduces the accuracy by a factor of 6 to 8. All Irish and U.K. stations were closed down in Year 2000.

3.3.2 Loran C System

Loran-C is an all weather, highly accurate and reliable hyperbolic radionavigation system that covers most of the Northern Hemisphere [FAA02]. The system uses groups of at least three ground transmitter stations called chains. Each chain comprises one Master and two or three Secondary stations, several hundred kilometres from the Master station. Unlike Decca, Loran-C is unaffected by night and seasonal effects and coverage remains the same throughout the year and also by day and night [USC90].

Loran-C is a low frequency/long wave electronic position fixing system using radio signals transmissions @ 100 KHz from 3 or more transmitters, linked in a chain. It gives a latitude and longitude readout position to marine, aero and land receivers. It is transmitted via a vertical mast.

Loran-C was initially an American Department of Defence system covering the Pacific and Atlantic areas for 20 years. In 1974 the system was opened up to civil use. Because the U.S.A. has functional control of GPS, Congress decided they had no need to support two independent overseas marine systems. As a result, all overseas stations were closed down or handed over to host states to operate in 1993.

The Northwest European Loran-C System (NELS) is part of the European Union plan for an independent European Radionavigation System [NEL95]. A predicted accuracy of 463m (0.25 nautical miles) extends up to 1000 km off the south and west coasts of Ireland while an accuracy of greater than 100m is predicted for the Irish Sea, the Celtic Sea and the seas to the north of Ireland as well as all Irish coastal waters. Loran-C accuracy of greater than 100m also extends over the whole of Ireland for aero and civil users. The repeatable accuracy of Loran-C is impressive, allowing a return to a marked position with greater accuracy time and time again. All Loran-C signals are constantly being monitored. A code within the signal, which is known as Blink, will warn users of any abnormality.

3.4 Cellular Based Systems

Cellular networks cover a high percentage of regions in the populated world. Those regions with no cellular networks are predominantly undergoing plans to introduce them. As such, they provide a good infrastructure and freely available resource for global positioning. Cellular based systems can be separated into three categories, pure network based solutions, network based mobile-assisted solutions and mobile-based network assisted solutions, as can be seen in Figure 3.1. Generally, if a positioning system is network based mobile-assisted, then it can also be mobile-based network assisted, depending on implementation.

3.4.1 Cell of Origin (COO) or Cell Identity (CI)

Cellular networks are divided into cells, with each cell being controlled by a base station, which transmits and receives signals from the mobile station located in its cell. Cell of Origin (COO) simply identifies the cell that the mobile station is located in, [LUD01]. Any handset that is connected to the network can be positioned in this way. It is the simplest of all the location position methods. However, accuracy of this method is dependent on the size of the cell. For indoors, cell sizes can be as small as 50m^2, However, in rural areas a cell size of 50km^2 is quite normal. Since modifications are minimal it is a very cost efficient method to deploy. Cell Identity is a pure networked based solution with no interaction necessary from the mobile station.

3.4.2 Timing Advance (TA)

Timing Advance is another pure network based solution. It is a GSM method for position location. It is simply a specific parameter in GSM. This parameter is a measure of the time it takes for a signal to travel from a base station to a mobile station. It is required because multiple devices share the air interface at the same frequency using different time slots. It ensures that the bursts from the mobile station arrive at the base station at the correct intervals. The timing advance parameter can be converted into distance with a precision of 550 meters, [3GP99]. If used in conjunction with COO, the accuracy may be improved dramatically. However, once again the accuracy is dependent on the cell density. 3G systems have an equivalent parameter called Round Trip Time, which can be used for position location in the same manner.

COO and TA can also be implemented as a mobile based solution, in which the base station transmits the necessary information to the mobile station and the mobile performs the calculation to give the approximate location. Timing advance is also a cost efficient method as minimal modifications are necessary in deploying this technology.

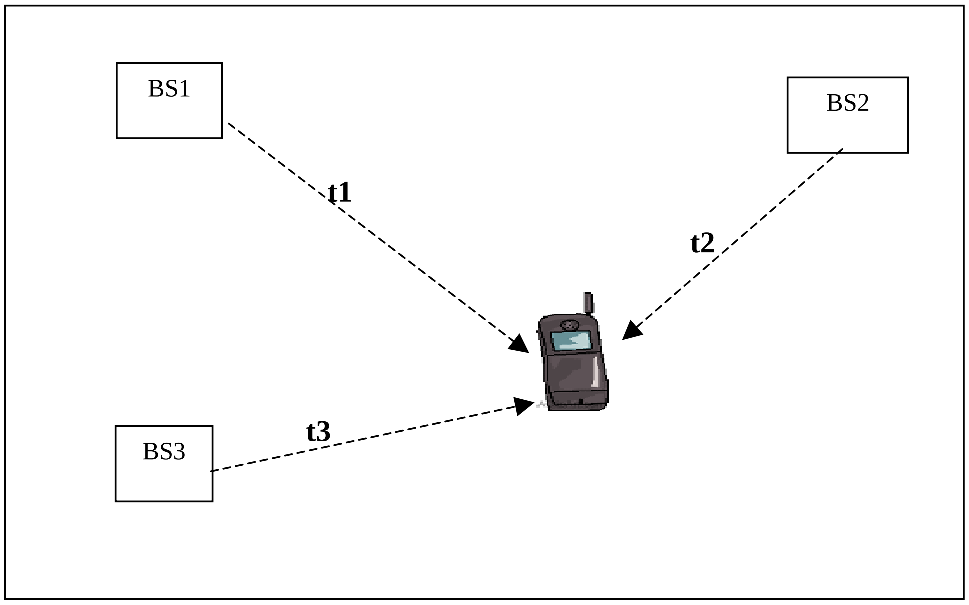
3.4.3 Time of Arrival (TOA)

This method measures the arrival time of a signal transmitted from a mobile station to several base stations. The distance to each of these base stations is then calculated by measuring the time taken by the speed of the signal, which is approximately the speed of light, see Figure 3.6. Triangulation is then used to determine the location of the MS.

Some implementations of TOA use Location Management Units (LMU) with GPS time stampings. When a base station receives a location request, it sends a signal to the mobile station to generate an access burst. It then provides information about the timing of the burst to several LMU's. Each LMU processes the received burst and estimates TOA with respect to GPS time. The TOA values are then sent back to the base stations for location computation of the mobile station.

The accuracy of TOA systems can vary to within 50 meters in rural areas and 150 meters in highly populated urban areas [GOR99], [FKL00]. They do not need line of sight, but multipath propagation can cause errors. Long processing delays can also cause problems. They are very expensive as atomically accurate clocks are necessary in order to synchronize the base stations and the receivers. TOA is one of the methods, which has been standardized by the 3GPP consortium, [3GP99].

Figure 3.6 Time of Arrival (TOA)



3.4.4 Time Difference of Arrival (TDOA)

Time Difference of Arrival is similar to the Time of Arrival method with one major difference; the base stations and the receivers do not need to be synchronized [GG03]. The asynchronous networks are usually tied to a common reference time by measuring the time differences between the base stations. These time differences are called Real Time Differences (RTD) and can be measured from the signalling delays or from additional hardware. Once all the RTD's in a network are calculated, each base station can be synchronized to one particular base station, or to some reference time, for example GPS time. The inaccuracy of the RTD's should not exceed tens of nanoseconds in order to maintain good position accuracy. The RTD's will have to be recalculated at a regular interval, in order to avoid inaccuracies caused by clock drifts.

The time difference between the base stations and the receiver can be removed also. As the base stations are virtually synchronized by the RTD's, the clock bias is the same with respect to all base stations. By subtracting two measurements from each other, the clock bias will cancel out. At least three base stations are necessary for the clock bias to cancel out. However, a fourth base station will improve the accuracy and integrity of the solution.

The accuracy of TDOA is well within E-911 standards, in urban and suburban areas [SAR02]. The availability of TDOA is dependent on signal coverage and the number of base stations that the receiver can receive transmissions from. Multipath propagation, NLOS propagation and errors in clock synchronization are the primary sources of degradation of accuracy. Due to the RTD measurement, the signalling load on the network is increased dramatically, which can affect the performance of the network and the positioning algorithm. On a positive side, TDOA positioning works well indoors and in underground facilities, assuming base station availability. Also, TDOA is directly applicable to legacy phones as no modifications are necessary on the receiver side. The term legacy phone refers to mobile phone that are on the market prior to new applications coming on the market.

3.4.5 Unilateral Timing Methods

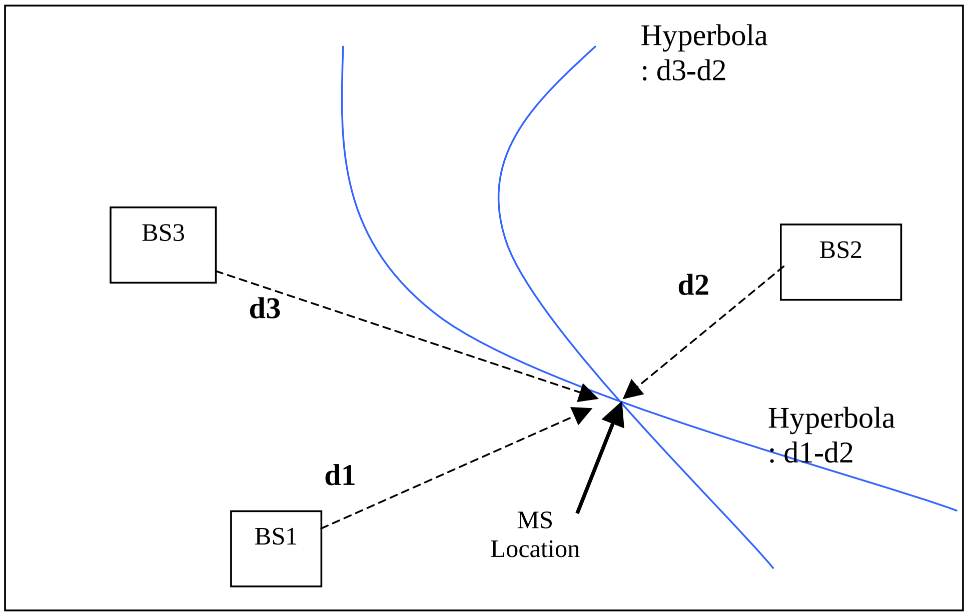
Time of Arrival and Time Difference of Arrival are multilateral positioning methods. This means that the calculation is based on location measurements made by signals at several base stations. The following are three methods of unilateral based positioning where, the position is calculated by the receiver based on signals sent from several base stations.

3.4.5.1 Enhanced Observed Time Difference (E-OTD)

The Enhanced Observed Time Difference (E-OTD) is another method similar to TOA for determining the position of a MS. This method is standardised by GSM, [ETS99]. For this method, the MS needs to be able to receive signals from more than one base station. The mobile station measures the time taken for the signals to travel from the BS to the MS. If the network is synchronised, the MS employs TOA positioning. If the network is asynchronous, TDOA positioning is employed. The location can be determined by comparing the time differences between the stations, see Figure 3.7. The differences in time are combined to produce intersecting hyperbolic lines. A hyperbola is a curve from where the difference in the distances to two fixed points is constant. The intersection point between the hyperbolic lines is the location.

If the base stations in the network are not synchronized, the delays between the base stations must be measured using LMU’s. The LMU’s used in the E-OTD method are simpler and less expensive than those that are used for TOA. Also, they are only necessary in a 3:1 ratio with the base stations, i.e. three base stations for every one LMU. This method is very accurate but is subject to degradation in places where signal interference or weak signals occur. E-OTD is also dependent on the number and location of base stations. Modifications are necessary to the network and to the MS for E-OTD.

Figure 3.7 - The Use of Hyperbola in E-OTD for Position Location.



3.4.5.2 Observed Time Difference of Arrival – Idle Period Down Link

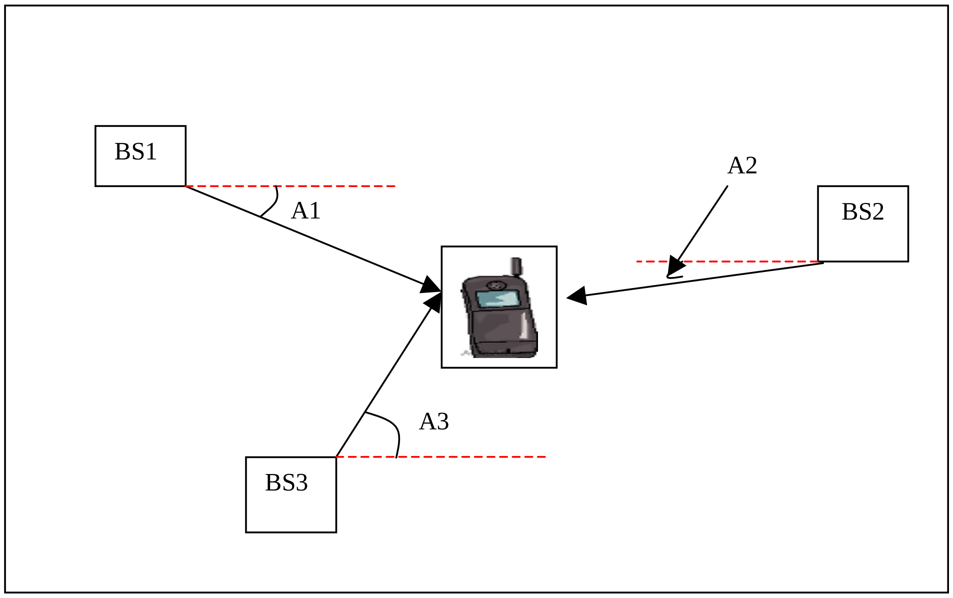
Observed Time Difference of Arrival – Idle Period Down Link (OTDOA-IPDL) is based on TDOA positioning principal. It is being standardized in 3G networks, which are CDMA based systems [3GP99]. In CDMA systems, interference is a problem and causes the near-far effect, which degrades the accuracy of the position. The near-far effect is caused by a transmitter with a strong signal that blocks or suppresses a transmitter with a weaker signal. This problem is compensated by the use of idle periods. The base stations temporarily ceases downlink transmission so that the receiver can measure the signals from neighbouring base stations without interference. The accuracy of OTDOA-IPDL is therefore better than E-OTD. One of the problems with OTDOA-IPDL is the receiver trying to measure the signals from base stations during very short idle periods. Multipath and NLOS propagation are two of the main error sources that impinge on the accuracy of this method.

3.4.6 Angle of Arrival (AOA)

This method uses the angles of the incoming signals at the base stations to estimate the position of the mobile station [3GP99]. The base station needs a set of directional antennas in order to calculate the incoming signal, see Figure 3.8. Once the angles are calculated, triangulation can once again be use to calculate the mobile station position.

A minimum of two base stations is required for 2-d location. Generally more than two are used for improved accuracy. This method does not perform well in urban areas due to effect of reflect signal cause by multipath propagation. A set of highly adaptive phased antenna, called Adcock Array antennas, is needed at each base station for this method [HEK02].

Figure 3.8 - The Use of Angle of Arrival (AOA)



3.4.7 Received Signal Strength (RSS)

This method of location positioning is based on signal power levels. In all cellular systems, the mobile station will make measurements on the air interface in order to facilitate handover decisions. These measurements contain estimated power levels from the serving base station and from neighbouring base stations. These power levels can be used to estimate the distance from the mobile station to the base stations. The following is the formula used:

Eqn. 3.1) $$P_r = P_t + G_t + G_r - PL(d)$$

where P_r is the power that is received at the receiver

P_t is the power transmitted from the transmitter

G_t is the gain of the transmitter antenna

G_r is the gain of the receiver antenna and

PL(d) is the pathloss of the signal at a distance d

Using a known mathematical model, which describes the signal pathloss, the distance can then be estimated from the pathloss if the other parameters are known. Hata-Okumura model is the most commonly used, [HAT80]. The model is represented by:

Eqn.3.2)

$$PL = 69.55 + 26.16\log(f) - 13.82\log(h_t) \\ - a(h_m)[44.9 - 6.55\log(h_t)]\log(d)dB$$

Eqn 3.3)

$$a(h_m) = [1.1\log(f) - 0.7]h_m - [1.56\log(f) - 0.8]dB$$

where f – operating frequency (MHz)

h_t – base station antenna height (m)

h_m – mobile station antenna height (m)

a (h_m) – correction factor for mobile station antenna height (dB)

d – the distance from the BS to the MS (m)

By calculating the distance from the mobile station to several base stations, the method of triangulation can once again be used to calculate the location of the mobile station [GOR99].

3.4.8 Location Fingerprinting (LF)

This method greatly differs from the other cellular positioning methods reported in this text. LF actually makes use of the multipath phenomena. By combining the multipath pattern with other signal characteristics a unique “fingerprint” for each location can be created. These fingerprints have to be calculated in locations all over the coverage area and are stored in a database. By matching the fingerprint of the receiver’s signal with the database of known fingerprints, a geographical location can be identified, [KLH00].

Accuracy and availability of this method depends greatly on the size and resolution of the reference database, as availability in particular is limited to the range of locations surveyed beforehand and to network coverage. The accuracy of the method is also dependent on the environment staying constant, since any changes will cause changes in the multipath and hence the fingerprint of the signal. As there are no changes necessary to the receiver, this method is suitable for legacy phones. Rather minor changes are also necessary on the network end to enable fingerprint measurements and the necessary signalling. One of the major advantages with this technology is the fact that it works well in indoor locations and in urban locations. However, the maintenance of the fingerprinting database is costly, from both a financial and a time point of view.

3.5 Indoor Based Location Methods

3.5.1 Wireless Local Area Network Positioning

WLAN positioning is another form of positioning method and can be used in any WLAN network, indoors or outdoors. Positioning can be carried out in a similar method to cellular networks using signal strength methods or TDOA methods. In general the positioning methods used in WLAN are unilateral or multilateral, i.e. signals from more than one transmitter are necessary. The mobile device that is being located measures the signal strength of each of the Access Points in range. It then searches through the Radio Map database to determine the signal strengths that best matches the signal strengths it has measured.

Alternatively, a Cell ID location method can be used since the location of access points are known. Registration to a certain access point locates the mobile device to a certain area [HKS+97]. Another method of WLAN positioning combines the use of signal strength and radio frequency electronic tags [CG93]. These tags can then be monitored through a network of access points. Clever deployment of access points gives good coverage and accuracy.

The initial work in the area of WLAN positioning was carried out by Microsoft [BP00]. This system, called RADAR, is based on signal strength and fingerprinting as described earlier. A Radio Map of the given area is created, which holds the signals strengths received from each of the access points at each location in the area. An access point is a transmitter in WLAN technology. See Appendix A for more information on IEEE 802.11b WLAN technology.

Several other similar attempts have been made to determine user location from 802.11b access point information including, the Halibut system by Stanford University [KMR01], the Nibble system by UCLA [CCK01], Amulet by the University of Rochester [HAR02], a system developed by the University of Linz [FWN01] and a system developed by Carnegie Mellon University [SSS01].

3.5.2 Sensor Technologies

Sensors can also be used for location positioning. Commonly used sensors in positioning include devices, which measures the receiver’s acceleration, angular velocity, ambient air pressure or magnetic field. One of the most popular of these methods includes MEMS-INS micro-electromechanical sensors with inertial navigation systems to measure acceleration of the receiver. Sensors can only give a local position. In order to achieve global positioning, sensor technologies have to be combined with another positioning method such as GPS. For more information on sensors see, [KEW00], [SCH00].

The 3D-iD RF tag system built by PinPoint Corporation is an example of a sensor technology determining location [WL98]. Antennas placed around a building emit RF signals at 2.4GHz. Tags worn by people or attached to mobile equipment transmit a response signal at 5.8GHz along with an identification code. Various antennas receive the signal and send the results to cell controllers, which triangulate the reflections to determine the tag's whereabouts. The accuracy of this system is dependent on the number and locations of the antennas.

3.5.3 Infrared Technologies

The seminal work in IR based systems is the Active Badge system [WHF92]. A badge, worn by a person or a piece of mobile equipment, emits a unique IR signal every 10 seconds. Sensors placed at known positions within a building pick up the unique identifiers and relay these to a centralized location server. This information is then used to determine the precise location of the user. Another example of infrared technology used to infer location is given in [AZU93]. In this system. IR transmitters are attached to the ceiling at fixed known positions around the building. An optical sensor senses the IR beacons and enables the system software to determine the user' location. These systems provide accurate location information but require specialised hardware.

3.5.4 Ultra-Wideband

Ultra-Wideband is a wireless technology for transmitting digital data over a wide spectrum of frequency bands with very low power. It uses very narrow or fast time pulses to deliver information. In ultra-wideband, very fast pulses that contain the data are directly transmitted. This results in the bandwidth of the pulses having very low energy density. This low energy density reduces its interference potential and its very short pulses make it relatively immune to multipath fading.

Ultra-Wideband sensor networks can be used efficiently in location positioning. The sensors are integrated together in an ad-hoc peer-to-peer network, similar in concept to access points in Wireless LAN's. These sensors then transmit unique identification codes, which are picked up by the receivers. The receiver then uses this information about the transmitters and the method of triangulation to calculate a position.

This technology works well both indoors and outdoors. Outdoors, it performs well over a distance of over a few kilometres in line of sight situations. Indoors, it can operate over a hundred metres from the transmitter, depending on the construction materials. Ultra wideband can give less than one foot accuracy in indoor location and a few inches accuracy outdoors in line of site propagation, [FON01].

UWB is superior to infrared tracking, ultrasonic tracking and other radio-based positioning systems in terms of accuracy. Most systems using radio technologies such as Bluetooth or Wi-Fi can only pinpoint an item within about 3 meters to 5 meters in an indoor environment. One such UWB location positioning product is Ubisense, which is able to track to an accuracy of 6 inches or 15cms within a typical building [LIP03].

This technology is by far the most accurate positioning technology. It is not widely available or deployed to the public yet. However, in the future as this technology is deployed on a widespread basis, its use for location positioning should become widespread.

3.6 Summary

Satellite-Based systems offer an excellent solution for outdoor location positioning. They are constantly being improved and updated. They achieve good accuracy and integrity. However, one of the main disadvantages of satellite-based systems is the availability of the system, i.e. the receiver needs to be able lock onto four satellites in order to perform triangulation. Using the NAVSTAR GPS system, there should be five satellites in view from any position in earth at all times. This is assuming that the user is in a rural area. However, in urban, indoor and built up areas, this is not always the case and the system will degrade significantly with lower accuracy, if any position fix can be obtained at all. The use of Assisted-GPS will obviate or reduce this availability problem. Table 3.3 shows the Satellite based systems and approximate accuracies that can be obtained.

Table 3.3 – Accuracy Improvements of the Global Navigation Satellite Systems

System	**Accuracy**
Original GPS with SA	35 meters
GPS without SA	6 meters
DGPS	3-5 meters
WAAS	< 3 meters
LAAS	< 1 meter
EGNOS	5 meters

Non-satellite based radio navigation systems like Loran-C and NELS provide location positioning on coastlines, in particular. These systems provide good accuracy and 24hr coverage. However, as the accuracy and availability of GPS improves, these systems will become obsolete, as they cannot compete.

Several cellular based location position methods are detailed in section 3.4. There are a number of characteristics studied when deciding which cellular based technology is most suitable. These include qualities such as, accuracy in different classifications such as rural or indoor, whether modifications are necessary to the handset or the network. If there is no modification necessary to the handset, legacy phones can be used to implement the technology. This has the added benefit of quicker and better market penetration. Network modifications can be very time consuming and costly. Table 3.4 below summarises some of the important characteristics for cellular based positioning technologies.

Table 3.4 – Comparison of cellular based technologies

	Handset Modification	Network Modification	Rural Performance	Urban Performance	Indoor Performance
COO	No	Yes	Poor	Moderate	Moderate
TA	No	No	Poor	Moderate	Good
TOA	Yes	Yes	Moderate to Poor	Moderate	Moderate
AOA	No	Yes	Poor	Moderate to Poor	Moderate
EOTD	Yes	Yes	Poor	Moderate	Good
RSS	No	Yes	Poor to Moderate	Moderate	Moderate to Good
A-GPS	Yes	Yes	Good	Moderate	Poor to Moderate

The use of portable computers, PDA's and indeed mobile phones spurs the need for location positioning technologies that operate indoors. Section 3.5 introduced several technologies that can determine the position of a user or device indoors. WLAN positioning is a very useful method of determining position in an indoor environment. If the indoor environment has already deployed a WLAN network, this method is cheap, easy to deploy and provides good accuracy. As the use of WLAN networks becomes ubiquitous, WLAN positioning will become extensive.

The use of sensors for location positioning allows for high accuracy but is dependent on the number and placement of highly specialised equipment. IR technology suffers from the same drawbacks as sensors but once again can provide very good accuracy. Finally, ultra-wideband is a relatively new technology, which also provides location positioning both indoors and outdoors when a network of sensors is added to the area.

Table 3.5 – Technologies Deployed by different Companies

GPS	SnapTrack Inc.
A-GPS	Lucent Technologies
TDOA	Associated Inc. – TruePosition
E-OTD	Nokia & Cambridge Positioning Systems – Cursor
AOA	KSI Inc. - TeleSentinel
Received Signal Strength	GeoMode & ModelSoft Oy
Location Fingerprinting	U.S. Wireless Corp.- RadioCamera™
WLAN	Ekahau Inc.
Ultra-Wideband	MultiSpectral Solutions Inc.

This chapter has provided an overview of several available and developing wireless location technologies. It can be seen that each technology is distinct and has many different advantages and disadvantages. Several companies are developing or have already developed solutions using some of these technologies for various environments, see Table 3.5. It is unlikely that any one system will be able to provide accurate mobile positioning under all circumstances. The final solution to the problem is more likely to be a combination of several methods working together.

Wireless Local Area Network is the radio technology deployed for this thesis. For this reason, the location positioning technique used must be compatible with WLAN technology. The positioning technique chosen for this thesis is WLAN positioning using location fingerprinting and received signal strength. This method is easy to deploy and no special hardware is necessary apart from the WLAN hardware. It is inexpensive, as there is no extra hardware. In fact the only necessary modification is a software installation on the mobile device. Most importantly, it can achieve a high level of accuracy in an indoor environment. The Ekahau Positioning Engine boasts an average accuracy of 1m [EKA02]. In the future, it is envisioned that the use of UWB as the position location method would greatly improve the accuracy of the system. However, as it is not currently available to the public, WLAN technology positioning is the best choice for this thesis.

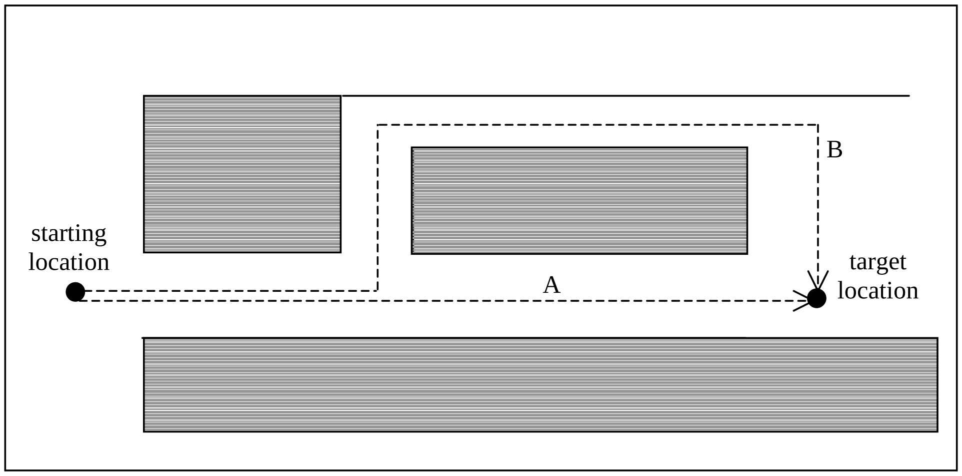
4 Path Finding Algorithms

4.1 Introduction

Path finding algorithms are state space searching algorithms that find a path from a source to a destination. It is a type of searching, which starts at the source and continues until the destination is reached. Given an area in a map and starting at a certain location, the problem is to find the shortest path to the target location. As well as finding a path between the source and destination, the path found should be the optimal path, i.e. the shortest path. The most efficient method of searching for a shortest path makes use of graphs to perform the searching.

Given an area in a map and starting at a certain location, the problem is to find the shortest path to the target location, see Figure 4.1. The shaded areas of the map are obstacles. The dashed lines are paths between the starting position and the target position. Two paths are shown, A and B. In this case A is clearly the optimal path as it is the straight-line distance between the two points.

Figure 4.1 – Example Pathfinding Problem



The types of problem suited to searching usually have a defined initial state, a defined target state and a set of operations that allow the problem to move from one state to another. A state represents the condition of a problem at a given finite time, e.g., the user is initially located at coordinates (5,10), represents a start state. The space or search space consists of all possible states that can represent the problem. An operation is an action performed in order to move the problem to another state, e.g. the user moves from coordinate (5,10) to coordinate (6,10). Any type of problem that makes use of states and operators is said to use a state space approach to problem solving. In this thesis the problem that is being solved is moving from one location on a map to a new location, i.e. path finding.

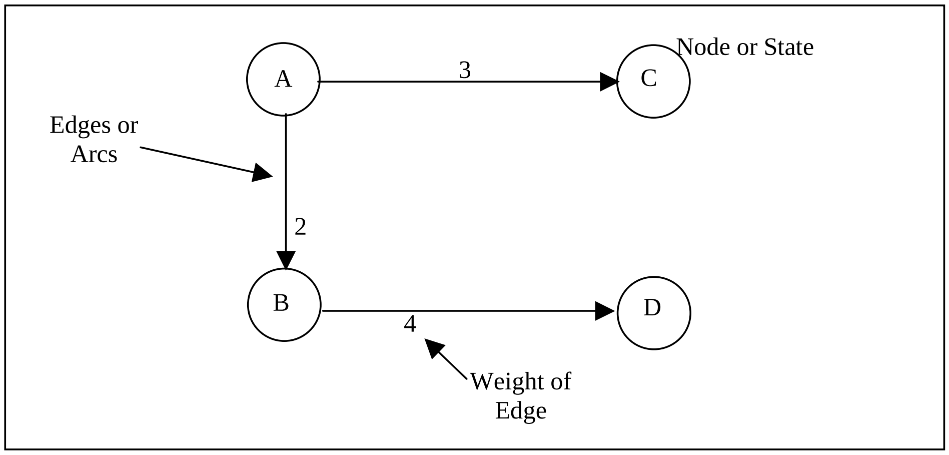
Search algorithms can be generally classified into two simple types. The simplest search methods are uninformed as they have no information about the state space and are left to search blindly through the nodes in a systematic fashion. However, if there is some knowledge about the problem available, the algorithm can be guided to a more efficient conclusion. This type of searching is called informed searching.

The following section introduces the use of graphs and trees for searching. Next, uninformed searches are introduced with an explanation of several well-known examples. Then, informed searches are introduced with three well-known examples given. The chapter draws some conclusions in section 5.6.

4.2 Graphs and Trees

Graphs are very useful in modelling a large number of real life problems, specifically searching problems, i.e. finding the shortest path from one area to another. A graph, G, is a set of nodes or vertices, V, connected together by links called edges, E, or arcs. Figure 4.2 is an example of a graph with nodes, {A, B, C, D} and with edges, {ab, ac, bd}. Typically, a graph is depicted as a set of circles, i.e. the vertices or nodes connected by lines, i.e. the edges. A node is a state that a problem can be in, for example in pathfinding a node is a 2-d coordinate of the present location.

Figure 4.2 – A Directed Weighted Graph

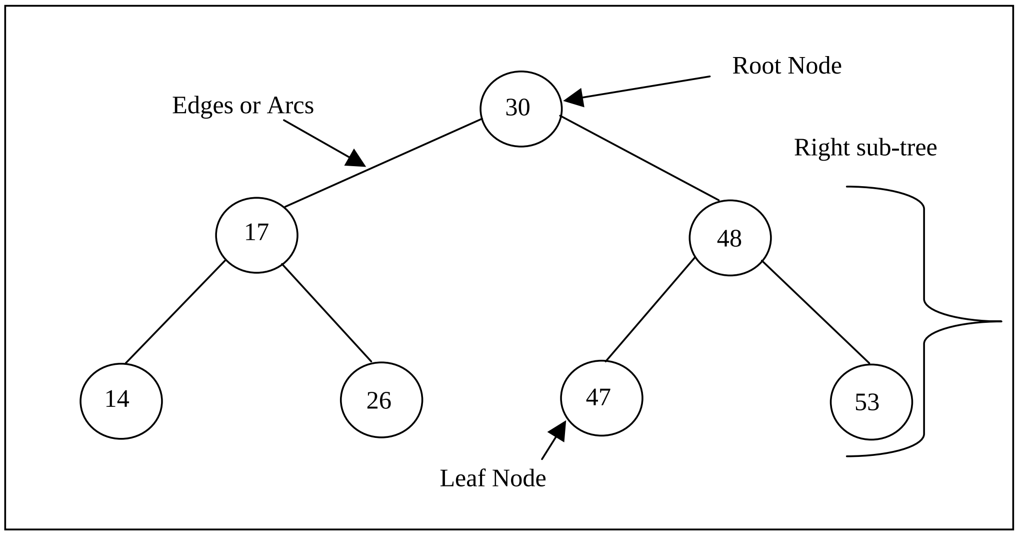


A path in a graph is the edges that are followed to get from one node to another node. The length of the path is the number of edges in it, e.g. in Figure 4.2 there is a path of length 2 from A to D along the edges, ab, bd. A graph is connected if there is a path between any two vertices. A directed graph is a graph with all edges having an ordered pair of vertices and a direction associated, e.g. a one-way road system, which can be travelled in only one direction. In a weighted graph, each of the edges is assigned a weight or value. The weight of an edge is represented by w(e). The weight of the path in the example given from A to D is six, which is the sum of the weight of the edges traversed on the path.

A tree is a graph in which any two nodes are connected by exactly one path. The simplest form of graphs is the binary tree, which consists of nodes, the root node and the children nodes and the left and right sub trees. The nodes at the lowest level of the tree are called leaves. In an ordered binary tree, the elements in the nodes in the left sub tree are less than that of the root and the elements in the nodes of the right sub tree are greater than that of the root, Figure 4.3. The left and right sub trees are themselves ordered binary trees. The height of a tree is the length of the longest path from root to leaf, for example the height of the tree in Figure 4.3 is two. A tree is balanced if all root-to-leaf paths are short. Balanced trees are usually short and wide.

A graph search (or graph traversal) algorithm is an algorithm that systematically goes through the nodes in a graph, often with the goal of finding a particular node, or one with a given property. A binary tree search will in the worst case, traverse a root-to-leaf path. Therefore, keeping a tree balanced reduces the worst-case scenario because the height of the tree is now smaller. The use of trees in search algorithms makes the problem easier to comprehend and then solve as well as increasing the efficiency of the algorithms. For example, in moving from one location to another, the root node of the tree is the start location and the next level of the tree expands to all location that can be moved to in one step. This is continued until the target node or location is reached.

Figure 4.3 – A Binary Search Tree Graph



4.3 *Analysing a Search Algorithm*

The most important quality for any algorithm is correctness. That is proof that the algorithm yields a required result for every legitimate input in a finite amount of time. This is usually completed by mathematical induction or by testing all possible inputs. The next important quality for analysing a search algorithm or indeed any algorithm is efficiency. There are two types of efficiency: time efficiency and space efficiency. Time efficiency indicates how fast the algorithm runs and space efficiency indicates how much memory the algorithm requires.

Time efficiency is the more important of the two and has three important values.

- the average time
- the worst-case time and
- the best possible time.

However, generally with the worst-case time is the most important, as calculations based on worst-case times can lead to guaranteed performance predictions. Conveniently, the worst-case times are generally easier to calculate than average times.

Taking a sequential search as an example, which just searches a list of items in sequential order. If there are n items in a collection - whether it is stored as an array or as a linked list, then it is obvious that in the worst case, when there is no item in the collection with the desired key, then n comparisons of the key with keys of the items in the collection will have to be made.

To simplify analysis and comparison of algorithms, a dominant operation is found and then the number of times that operation has to be performed is counted. In the case of searching, the dominant operation is the comparison, since the search requires n comparisons in the worst case, we say this is an O(n) (pronounce this "big-Oh-n" or "Oh-n") algorithm. The best case in which the first comparison returns a match requires a single comparison and is O(1). The average time depends on the probability that the key will be found in the collection this is something that we would not expect to know in the majority of cases. Thus, in this case, as in most others, estimation of the average time is of little utility. If the performance of the system is vital, i.e. it's part of a life-critical system, then the worst case analysis must be performed in our design calculations as it represents the best guaranteed performance.

4.4 Uninformed Searches

There are a number of ways to search a graph, including breadth-first, depth-first and greedy algorithms. The first two examples are uninformed or blind searches that do not know the cost of moving to the next node. Breadth-first search always expands a shallowest unexpanded node reached so far, i.e. it searches the nodes on one level of the graph before moving to the nodes on the next level of the graph. Therefore, it finds a minimum-depth solution. For example in Figure 4.3, if the search algorithm was searching for number 47, it would start at the root node and then move to the children of the root and compare the number 17 and 48. After testing these numbers, it would move to the children of 17, before expanding the children of 48 and finding the target node. In comparison, depth-first search always expands the deepest node reached so far (and therefore searches one path to a leaf before following up any other path). Thus, it finds the "left-most" solution. These two search types are the simplest form of searching a graph and the idea behind them is used in many other algorithms. For example, Dijkstra's algorithm [DIJ59], uses a breadth-first type of searching with associated weights.

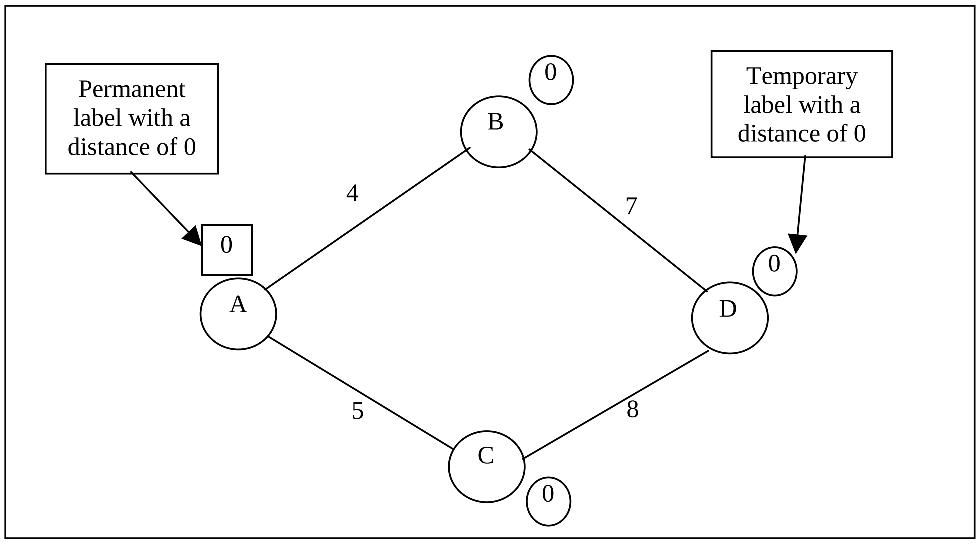
Breadth-first search was discovered by Moore, [MOO59], while trying to find a path through a maze and independently by Lee, [LEE61] in the context of routing wires on circuit boards. Depth first searches have been used since the 1950's in artificial intelligence programs. However, Hopcroft and Tarjan, [HJ73], were the first to recognize the widespread importance of depth first searches.

Iterative deepening is a graph search algorithm, which combines the properties of depth first searching and breadth first searching. It is a space state search strategy that visits each node in the search tree in the same order as depth first search but does so by gradually increasing the maximum depth limit of the search iteratively. At each iteration the maximum depth is increased and the search is re-run. This is repeated until the depth limit reaches the depth of the target node.

4.4.1 Dijkstra’s Algorithm

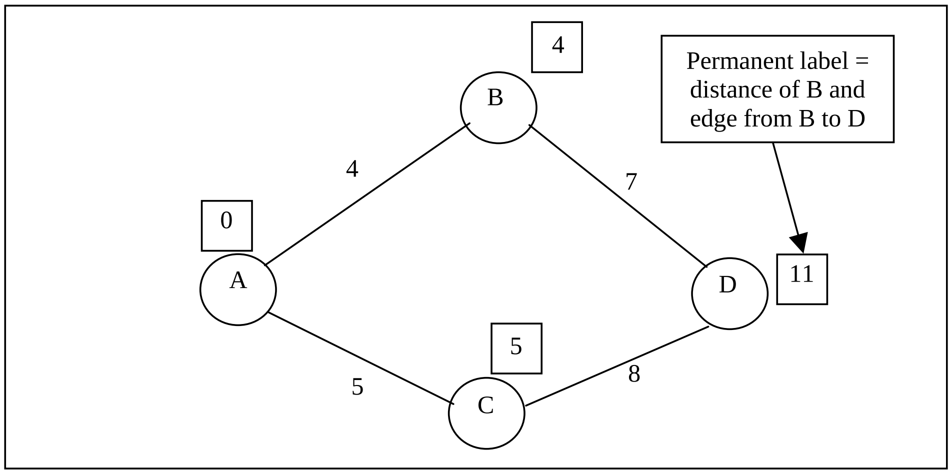
Dijkstra’s algorithm is another example of a shortest path algorithm. However, it also determines the distance between the start point and all other points on the graph. It is named after its inventor, the Dutch scientist Edsger Dijkstra, [DIJ59]. It creates labels associated with nodes. These labels represent the distance or cost from the source node to the current node. There are two types of labels used in Dijkstra’s algorithm, one of which must be attached to each node in the graph: permanent and temporary labels. The temporary labels are given to nodes that have not been reached and their distance to the root node is unknown. Permanent labels are given to nodes that have been reached and their distance to the root node is known.

Figure 4.4 Initialisation of a sample graph using Dijkstra’s Algorithm


Permanent label with a distance of 0
0
B
Temporary label with a distance of 0
4
7
0
0
A
D
5
8
C
0

The algorithm begins by initialising all nodes in the graph, the root node with a permanent label with the value 0 and all other nodes with temporary labels, also with the value 0. See Figure 4.4 for an example. The algorithm then proceeds to select the least cost edge connecting a node with a permanent label to a node with a temporary label. The label attached to the new node is then updated to a permanent label and the value stored in the label is the sum of the value in the label of the previous node and the cost on the edge between the two nodes. In the example in Figure 4.4, node B is selected and its permanent label gets the value 4. The next step is to find the next least cost edge extending from a node with a permanent label to a node with a temporary label. This process is repeated until all the nodes in the graph have permanent labels, see Figure 4.5.

Figure 4.5 – A Sample Graph after Dijkstra's Algorithm



4.4.2 Bellman-Ford Algorithm

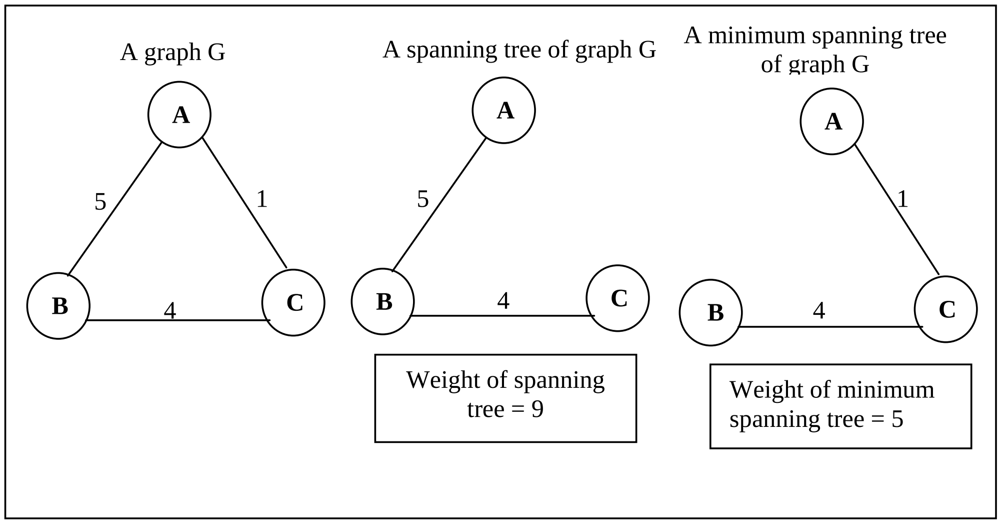
Dijkstra's algorithm does not allow for negative cycles or negative weight edges. An example of a negative edge could be cycling downhill as the cyclist builds up speed without pedalling. The Bellman-Ford algorithm is an example of a shortest path algorithm, which allows for negative weight edges. This algorithm is based on separate algorithms by Bellman, [BEL56], Ford, [FF62] and Moore, [MOO59].

The algorithm checks if there is a negative weight cycle reachable from the source. If there is such a cycle, the algorithm indicates that no solution exists by returning a Boolean value of false. If not, the algorithm produces the shortest paths and their weights and returns a Boolean value of true. It uses the technique of relaxation similar to Dijkstra's algorithm by progressively decreasing an estimate d[v] on the weight of the shortest path from the source, s, to each vertex, v, until it produces the actual shortest path weight from s to v. This algorithm has since been improved on by Goldberg, [GR93], Pallantino, [PAL84] and Pape, [PAP74], but the underlying principles remain the same.

4.4.3 Minimum Spanning Tree (MST)

A spanning tree of a connected graph, G, is a connected subgraph of G having no cycles, see Figure 4.6. Every graph has at least one spanning tree. The weight of a subgraph is the sum of all the edge weights in the subgraphs. Therefore, different spanning trees of the same graph have different weights. The minimum spanning tree is the spanning tree with the minimum weight; therefore from this the shortest path between any two points can be deduced.

Figure 4.6 – Minimum Spanning Tree


A graph G
A
B
C
5
1
4
A spanning tree of graph G
A
B
C
5
4
Weight of spanning tree = 9
A minimum spanning tree of graph G
A
B
C
1
4
Weight of minimum spanning tree = 5

From the example in Figure 4.6, it can be seen that finding the minimum spanning tree is a problem very similar to finding the shortest path between two points. The first algorithm for finding a minimum spanning tree was developed by a Czech scientist Otakar Boruvka in 1926. The purpose of the algorithm was to find an efficient electrical coverage for Bohemia and it is called Boruvka's algorithm, [NMN00]. Currently there are two main algorithms used to solve the minimum spanning tree problem. These are Kruskal's algorithm and Prim's algorithm.

4.4.4 Kruskal's Algorithm

Kruskal's algorithm finds the minimum-spanning tree for a connected weighted graph, [KRU56]. That is, given a list of connected nodes with weighted edges, it finds the path that covers all nodes using the least-cost edges. It works as follows:

- *Create a forest F, a set of trees, where each vertex in the graph is a separate tree*
- *Create a set S, containing all the edges in the graph*
- *While S is nonempty repeat the following –*
    - *Remove the edge with minimum weight from S*
    - *If that edge connects two different trees, then add it to the forest combining two trees into a single tree*
    - *Otherwise discard it*

When this algorithm is completed the forest F has only one tree, which is the minimum spanning tree of the graph.

4.4.5 Prim's Algorithm

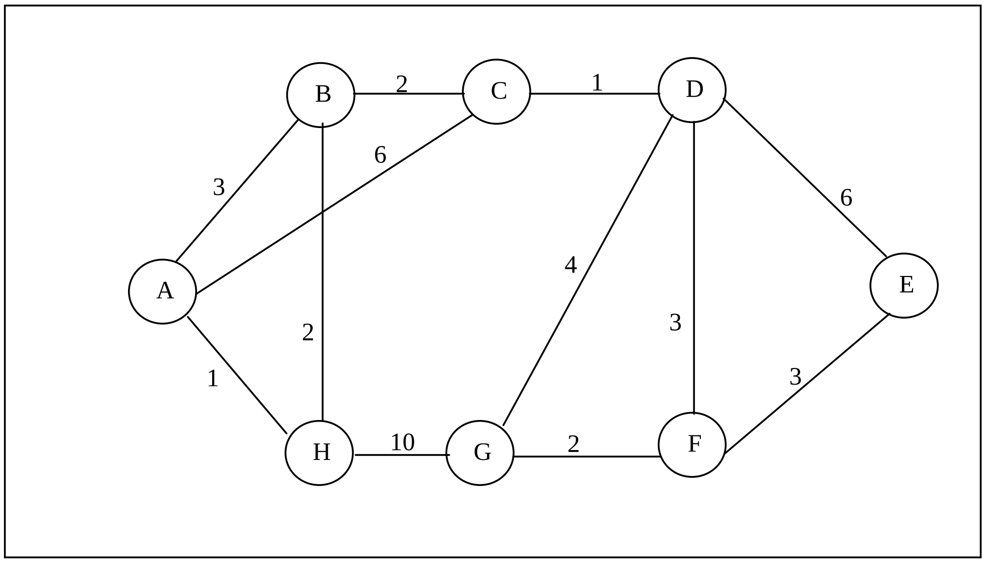
Prim's Algorithm is another example for solving the minimum spanning tree problem. The MST starts at an arbitrary node with a new node being added at each stage until all the nodes of the graph are part of the tree. This algorithm was conceived in 1957 by a computer scientist named Robert Prim, [PRI57]. It works as follows:

- *Create a tree containing a single vertex, chosen randomly from the graph*
- *Create a set, S containing all the edges in the graph*
- *OLP until every edge in the set connects two vertices in the tree*
    - *Remove from the set, S an edge with minimum weight that connects a vertex in the tree with a vertex not in the tree*
    - *Add that edge to the tree*

On termination, the tree that was created is the minimum spanning tree of the graph.

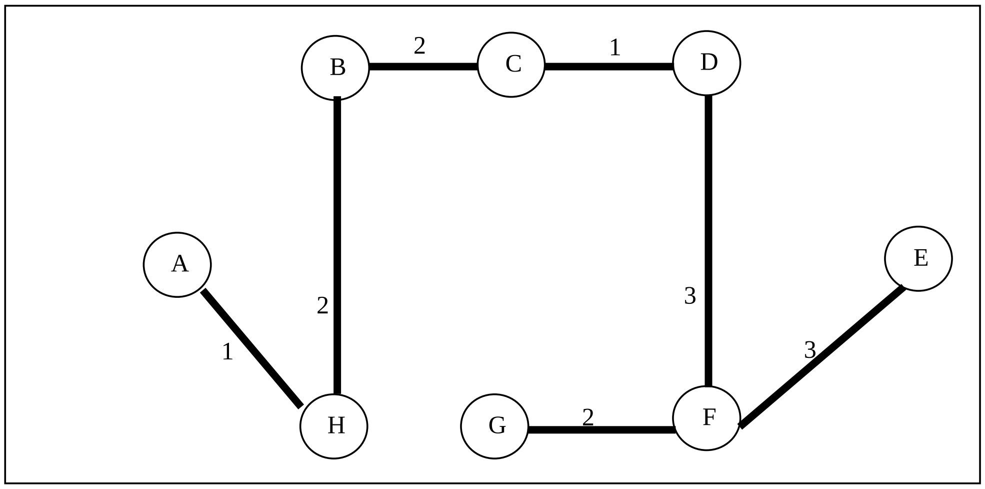
As an example of this algorithm, see Figure 4.7, which shows a graph of possible bus routes through a city. The letters are bus stops; while the edges represent the time it takes to get from one bus stop to another.

Figure 4.7 – Graph showing possible bus routes



Starting from vertex A, there is three edges that can be chosen, moving to vertex B to vertex C or to vertex H. The edge going to vertex H is chosen first as this has the lowest weight of 1. From vertex H, there is three possible edges, moving to G, moving to B or moving to C back from A. Therefore the next edge that is chosen is the path from H to B, as this has a weight of 2. At this stage, the possibilities for the next move are either from vertex B to vertex C or back to vertex H and moving to vertex G. In this case, the edge moving from B to C has the lowest weight and is chosen next. For the same reason, D is chosen as the next vertex after that. Now the algorithm has four possible edges to take next. These are from H to G, with weight 10, from D to G with weight 4, from D to F with weight 3 and from D to E with weight 6. In this case the edge from D to F is chosen next. After this, the edge from F to G is chosen and finally the edge from F to E is chosen. The final MST graph can be seen in Figure 4.8.

Figure 4.8 Minimum Spanning Tree for graph 4.7



4.4.6 Dynamic Programming

Dynamic programming is a principle used in solving search problems. It works by breaking a problem into smaller problems, as is done in the divide and conquer principle and solving each of these smaller problems independently. In dynamic programming, the principle is to solve all the smaller problems and store the results to be used later in solving larger problems. Floyd's algorithm, [FLO62], to find the all-pairs shortest path is an example of dynamic programming, as is Warshal's algorithm, [WAR62], to find the transitive closure of a graph. The transitive closure of a graph answers the question, "is there a path from one vertex to another?" for all pairs of vertices in the graph. For weighted graphs, a table can be built which gives the shortest path between all pairs of vertices in the graph. This is called the all-pairs shortest path problem and the Floyd-Warshal algorithm and Johnson's algorithm for sparse graphs [JOH74] both solve this problem efficiently.

4.4.7 Floyd-Warshal Algorithm

This is an all-pairs algorithm that finds the shortest path from all vertices to every other vertex. It is a combination of Floyd's algorithm, designed by R. Floyd in 1962, [FLO62] and Warshal's algorithm, proposed by S. Warshal in 1962, [WAR62]. The idea is to first find all the minimal distances for pairs without using any intermediate vertices. These distances will equate to infinity if no edge exists between a pair and the edge weight if an edge exists. Next, allow paths that use the first vertex as an intermediate. If using this intermediate vertex decreases the weight, then use this new weight, or else keep the original value. Continue with this process until all the vertices of the graph have been included as intermediate paths. At this stage, there will be a table containing the shortest path between all pairs of vertices.

The algorithm can be defined as follows:

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{if } \mathrm{k} = 0, \\ \min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{k-1}) & \text{if } \mathrm{k} \geq 1. \end{cases} \qquad \text{(Eqn.4.1)}$$

Where k is equal to the number of intermediate vertices and $d_{ij}(k)$ is equal to the weight of the shortest path from vertex i to vertex j. See Figure 4.9 for an example implementation of Floyd's algorithm. D(4) gives the shortest paths from all vertices to every other vertex.

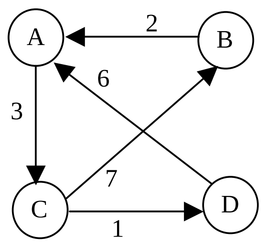
Figure 4.9 Example Implementation of Floyd-Warshal algorithm

D(0) =

	a	b	c	d
a	0	∝	3	∝
b	2	0	∝	∝
c	∝	7	0	1
d	6	∝	∝	0

D(2) =

	a	b	c	d
a	0	∝	3	∝
b	2	0	5	∝
c	9	7	0	1
d	6	∝	9	0



D(1) =

	a	b	c	d
a	0	∝	3	∝
b	2	0	5	∝
c	∝	7	0	1
d	6	∝	9	0

D(3) =

	a	b	c	d
a	0	10	3	4
b	2	0	5	6
c	9	7	0	1
d	6	16	9	0

D(4) =

	a	b	c	d
a	0	10	3	4
b	2	0	5	6
c	7	7	0	1
d	6	16	9	0

4.4.8 Johnson's Algorithm

Johnson's algorithm is another all-pairs shortest path algorithm, [JOH74]. The algorithm either returns a matrix of shortest path weights for all pairs or reports that the input graph contains a negative weight cycle. This algorithm calls Dijkstra's algorithm and the Bellman-Ford algorithm as subroutines of the algorithm.

The algorithm uses a technique called reweighting. If a path has a negative edge, then all the edges in the graph are given a new weight, such that there is no negative weighted edge in the graph anymore. This reweighting is done throughout the graph and thus does not change the shortest path, as can be seen in the proof in [CLR90]. The algorithm works by firstly adding a new node with zero weight edges from it to all other nodes. Then calls the Bellman-Ford algorithm to check for negative weight cycles and find d(u) the least weight of a path from the new vertex to vertex u. Then, the edges are reweighted based on these values and Dijkstra's algorithm is run for each of the vertices. The final weights of the paths are adjusted to take into account the reweighting process.

4.5 Informed Searches

Although the above search algorithms solve the shortest path problem efficiently, it is often infeasible to use them as they can expand a lot of nodes before a path is found. There are always limits on the amount of time and storage available to expend on the search. A greedy algorithm is an algorithm in which, at each stage, a locally optimal choice is made. A greedy algorithm is not very forward planning as it only examines the next step and picks the best choice from the possible options. An example of a greedy algorithm is a climber trying to climb to the top of Everest, at every step they move in the direction that is upwards.

These search algorithms can be similar to Breadth-first search. However, instead of comparing the left-most child node first, it picks the node to compare first based on heuristics. An heuristic is an attempt to predict how close to the target node is the current node. It is a rule of thumb, simplification, or educated guess that reduces or limits the search for solutions in domains that are difficult and poorly understood. In the example above, the heuristic would simply be the difference between the value of the node and the target node. Paths, which are judged to be closer to a solution, are extended first. Heuristics do not guarantee the optimum path. However, the search algorithm runs more efficiently with the use of heuristics.

4.5.1 Beam Search

The beam search is an informed search based on the breadth first search. It uses an heuristic to choose the next node to be expanded. Before moving to the next level of the graph, the algorithm expands a number, B, of the most promising nodes and discards the other nodes. B is the so-called beam factor. Therefore, there is a danger that the target will not be found. For example, if backtracking is preformed the branch that contained the target could be removed as it was not one of the most promising nodes. Although this search method is very fast, it does not always find the target node. Beam search was first used in the artificial intelligence community for speech recognition [LOW76] and image understanding problems [RUB78].

However, one of the benefits of this search method is that it avoids the combinatorial explosion problem. Some search algorithms might write out the entire graph before finding the optimal path. This approach is sufficient for small numbers of objects in a graph. However, imagine a tree where each node has three children. One level down, there are three nodes open. Two levels down there are nine open. Three levels down there are now 27 nodes open. We only have to get seven levels down and there are more than two thousand nodes open. This is combinatorial explosion. In real life systems, most problems will have a much higher branching factor. For example, in a coordinate system in a map, there is a branching factor of eight.

4.5.2 Hill Climbing Search

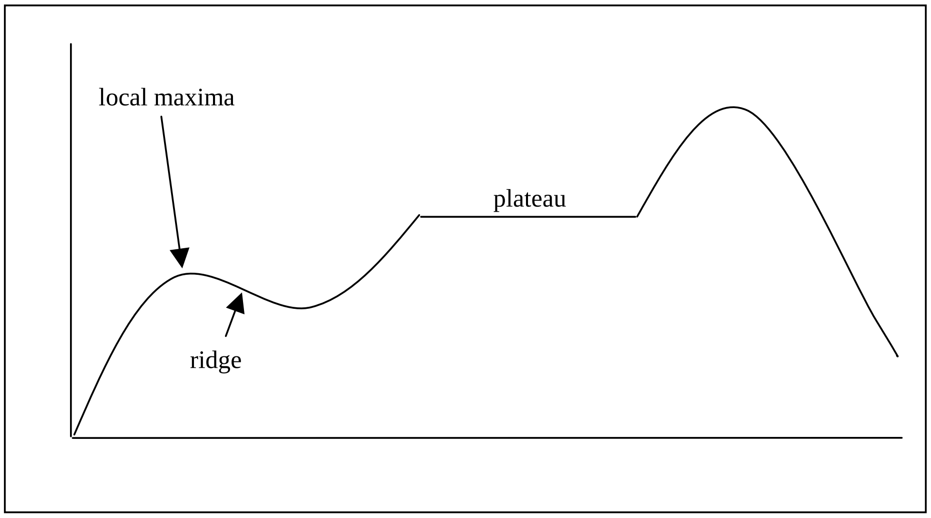
Hill Climbing search is based on the depth first search [ACK87]. However, it is an informed search as a heuristic is used to improve the efficiency of the search. At each step, the node with the best heuristic is chosen. The pseudocode of the algorithm is given as follows:

```
Function HILL-CLIMBING(Problem) returns a solution state
Inputs: problem
Local variables: Current = a node, Next = a node
Current = MAKE-NODE(INITIAL-STATE[Problem])
OLP do
Next = a highest-valued successor of Current
If VALUE[Next] < VALUE[Current] then return Current
Current = Next
End
```

It should be noted that this algorithm does not maintain a search tree. It only returns a final solution. Also, if two neighbours have the same evaluation and they are both the best quality, then the algorithm will choose between them at random.

The problems associated with this search include the following: "local maxima", "plateau's or shoulders" and "ridges". These are best explained by visualising the hill climbing analogy as shown Figure 4.10. A climber starts in the foothills and spends time climbing these foothill summits before realising that these summits are not the target but just local maxima. A local maximum is a peak that is lower than the highest peak in a state space. Once on a local maximum, the algorithm will halt even though the solution may be far from satisfactory. If the climber reaches a flat plain somewhere along the climb, local comparisons will not determine the best direction and they will wonder around aimlessly until an uphill direction is found. This problem is referred to as a plateau. When the climbers find themselves on a gently sloping ridge, most moves will take them downwards. This can be misleading, as the climber may believe they have reached the summit.

Figure 4.10 – Problems Encountered by Hill-Climbing Search Algorithm



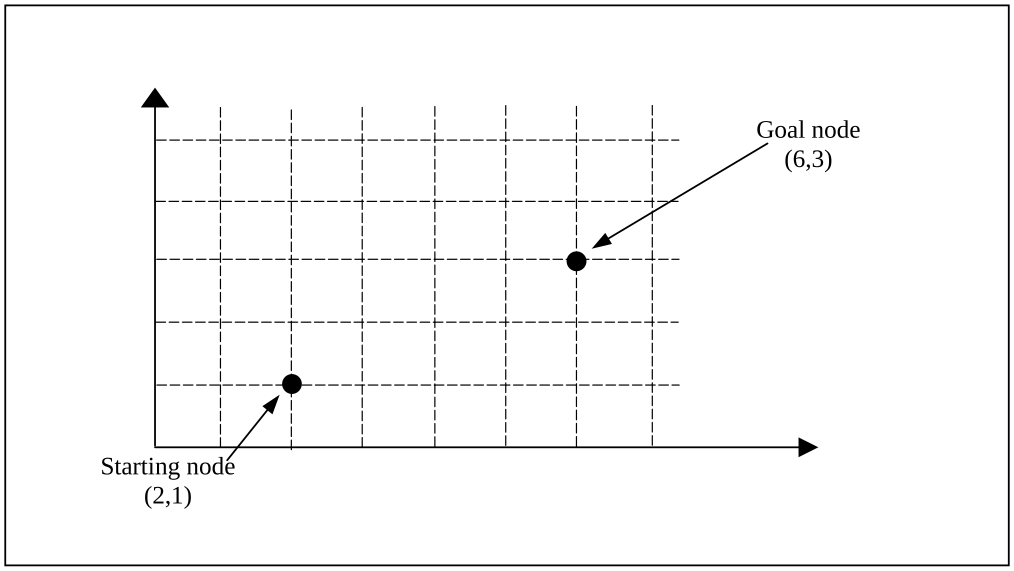
Hill climbing and beam search both have inherent problems, that need to be solved or there is a chance the shortest path will not be found. A* is an algorithm that solves these problem and uses the heuristic knowledge to guarantee a path being found.

4.5.3 A* Search

The A* Search is an example of a greedy search algorithm, as it uses heuristics to determine which node to test next [HNR68], [NIL80]. It improves upon the above search methods by also using an evaluation function. The purpose of the evaluation function is to provide a means for ranking those nodes that are candidates for expansion to determine which one is more likely to be on the best path to the goal.

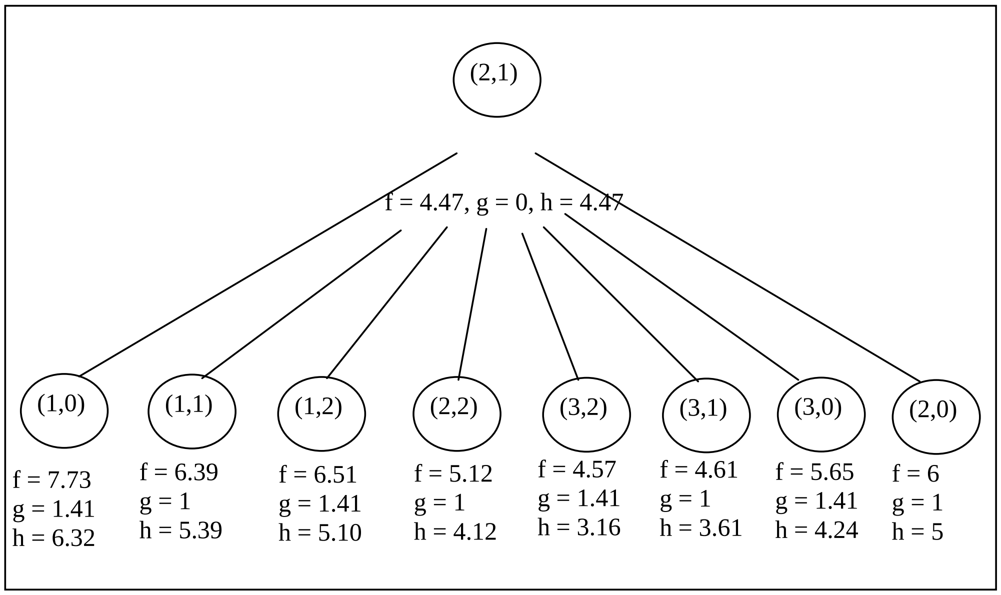
A* search begins at the start node and determines how many nodes can be moved to from this node. As an example of implementing the A* algorithm see Figure 4.11, which shows a grid on a map. The starting point or node is (2,1) and the goal or finishing point is (6,3). From this starting node, it can be seen that nodes on the next level of the graph would be: (1,0), (1,1), (1,2), (2,2), (3,2), (3,1), (3,0) and (2,0). The next step is to decide, which of these nodes in the graph to move to next. In order to achieve this, we associate a heuristic with each of the nodes. In this example, the heuristic will be the straight-line distance from the node to the goal node. For example, the straight-line distance from (1,0) to the goal node (6,3) is 6.3246.

Figure 4.11 – An Example Search Problem



As well as the heuristic, which is denoted by the letter h in the A* algorithm, each node also has two other values associated with it. These are represented by the letters f and g, where g is the sum of all the costs it took to get to the current node and f is the sum of g and h, i.e. f = g + h, i.e. it is the evaluating function of the algorithm. For the starting node, the value of g would be 0. In moving to the next level of the graph, the value of g would increase. If moving in a straight line horizontally or vertically in the example in Figure 4.11, the cost of moving is value 1. However, if moving diagonally, the cost of moving is 1.414. Figure 4.12 shows the first two levels of the graph that would be created for the example in Figure 4.11 using the A* algorithm, including the f, g and h values for each of the nodes.

Figure 4.12 – Graph Generated by A* Algorithm



After calculating the f, g and h values for each of the nodes in the next level, the algorithm chooses the node with the lowest value of f. In the example above, this is node (3,2). This node is expanded to show all the possible moves available from this point. This process is repeated, with the f, g and h values being calculated and the node with the lowest value of f moved to next, until the target node is reached.

If one of the new nodes on the graph has already been visited, then it is discarded and no further processing is done on that node. For example, when expanding the graph in 5.4 further, the node (2,1) is not included below the expanded node (3,2) as it has already been visited. This algorithm is a very efficient and fast method of calculating the shortest path between two points. It can be altered to take obstacles into account by increasing the cost of certain nodes.

The heuristic can be used to control the behaviour of the A* algorithm. It controls the balance between the speed and the accuracy of the algorithm.

- If h is zero, then only g plays a role in the algorithm and A* turns into Dijkstra's algorithm, which is guaranteed to find a shortest path
- If h is always lower than the cost of moving from the start node to the goal node, then A* is guaranteed to find a shortest path. The lower h is, the slower the running time of the algorithm

- If h is equal to the cost of moving from the start node to the goal node, then A* will only expand the best path, making it very fast. This is not always possible due to obstacles, etc.
- If h is greater than the cost of moving from the start node to the goal node, then A* is not guaranteed to find the shortest path but it can run faster
- If h is very high relative to g, then only h plays a role in the algorithm and A* turns into the Best-First Search (BFS) algorithm

In order for the A* algorithm to find the optimum shortest path between two points an admissible heuristic must be used. An admissible heuristic is one that must underestimate the distance to the goal node. This is essential in order that the actual distance to the goal node is never shorter than the estimate. A reasonably reliably underestimate of distance might be the Euclidian distance between the points, as no path can ever be shorter than the straight line between two points. This is the heuristic that is used in this thesis. Although it could make the running time of the algorithm slightly slower, this thesis is not concerned with running time.

4.6 Summary

Path finding algorithms are searching algorithms that find a path from a source to a destination. In this thesis, the shortest path between two locations must be found, i.e. the optimal path. Search algorithms are generally classified into two types, informed and uninformed. Several examples from each class were presented in this chapter. Dijkstra's algorithm is an example of a well-known uniformed search algorithm, which takes the cost of the edges into account. The Bellman-Ford algorithm is similar to Dijkstra's algorithm except that a negative weight or cost is allowed on edges. Dijkstra's algorithm cannot solve for negative weight problems.

Next, the idea of a minimum spanning tree was introduced. MST is another method of finding the shortest path between two points. Two algorithms which solve this problem were presented, Kruskal's and Prim's algorithms. The use of dynamic programming, which breaks a problem into smaller problems and then solves for the smaller problems were described after this, including two example algorithms: Floyd-Warshal and Johnson's algorithms.

Informed searches are search algorithms that employ forward planning to choose the next node to move to. This forward planning is based on a heuristic. Three examples of informed searches are specified, the beam search, the hill climbing search and finally the A* search.

The algorithms shown in this chapter are a subset of the better-known possible algorithms available to find the shortest path between two points. The algorithm used for certain applications can vary and depends on the properties of the application. For example, if looking for all the paths to a given point as well as the shortest path, Dijkstra’s algorithm would be one choice. The algorithm used in this thesis is A* search algorithm, as this algorithm finds the shortest path and then stops and can easily be altered to take obstacles into account.

5 Local Optimal User Position (LOUP) System

5.1 Introduction

The Local Optimal User Position system proposed in this thesis determines the location of the user in a Wireless Local Area Network and then ascertains the closest position that will improve the service to the user. The location of the user is identified using signal strength information. This information, together with the propagation model of the locality, allows the system to determine if the user is located in an area with a low class of service, for example, behind a cement pillar. In this situation, the system directs the user to move to another location, which will have a better signal strength value, in order to receive a better class of service. The class of services for IEEE802.11b are defined in Table 5.1.

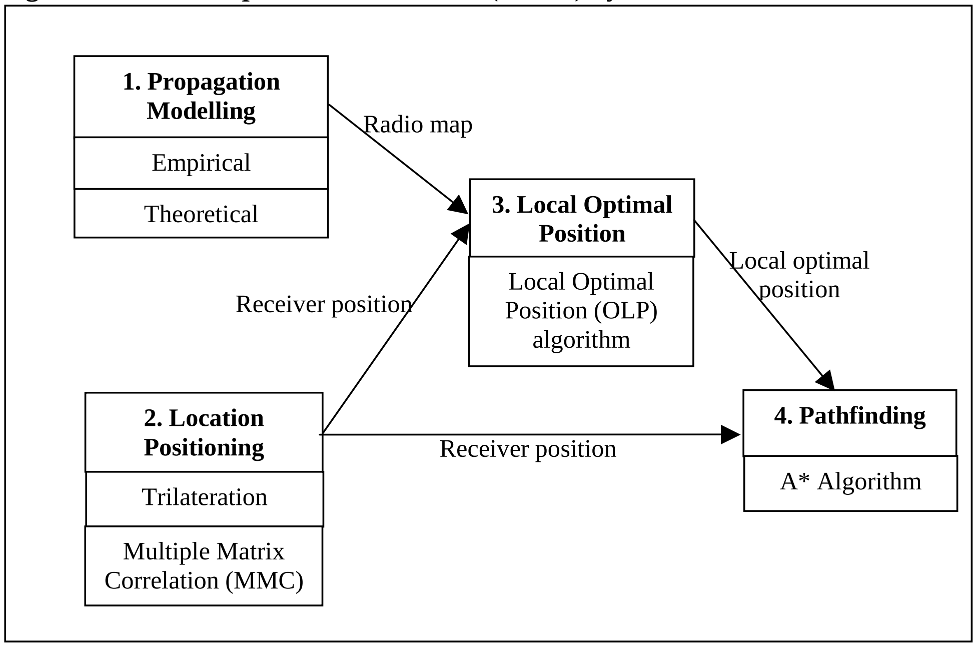
Table 5.1 – Classification of Service for IEEE802.11b

Class of Service	**Data Rate**
Class A	11Mbits/sec
Class B	5.5Mbits/sec and over
Class C	2Mbits/sec to 5.5Mbits/sec
Class D	1Mbits/sec to 2Mbits/sec
Class E	No service

The signal strength received at the user device determines the quality of the link, e.g. if the signal strength is below 60dB, a data rate of 11Mbits/s will not be achievable. 11Mbits/s is the maximum attainable data rate for IEEE802.11b, [IEE97]. Therefore, in order to improve the class of service, the user may have to be directed to an area with higher signal strength even if it means connecting to a different access point.

The local optimal solution is reached by discovering signal strength for a given area around the user, selecting the highest signal strength location and then moving to that location. The first step of the process is to create a radio map of the area. Once this is created, the location of the user can be determined using the signal strength received at the user device and the known fixed positions of the user. The signal strength information received by wireless devices is readily available by the WLAN software on the devices, e.g. the Client Manager from Orinoco [ORI]. After the location of the user is derived, the Optimal Local Position can be obtained. Finally, the user is directed from their current position to the new optimal position by the shortest path. This is achieved by the A* shortest path search algorithm [HNR68]. This chapter will explain the implementation details behind the above steps in detail, starting with stage 1, the propagation modelling stage. Figure 5.1 reiterates the LOUP system from Chapter 1.

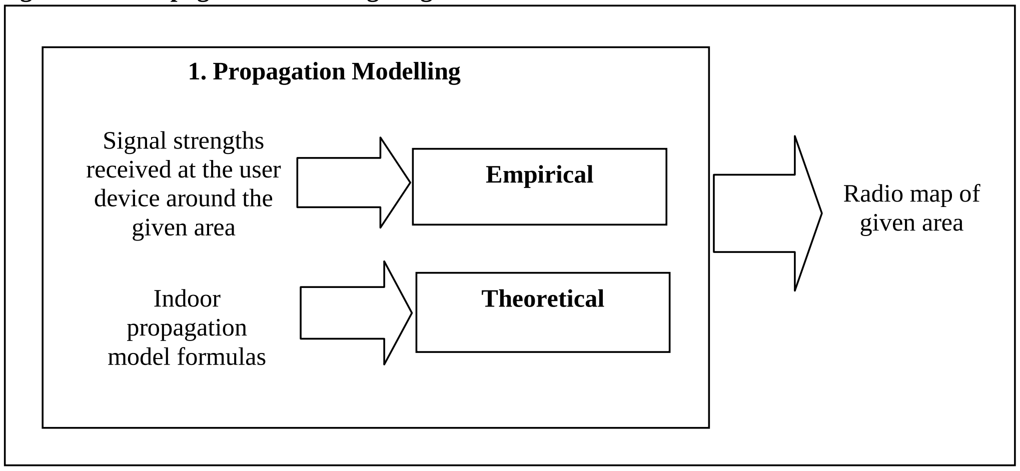
Figure 5.1 – Local Optimal User Position (LOUP) System



5.2 Propagation Modelling

A propagation model is used to give an estimate of the signal strength received at a distance from a transmitter. Figure 5.2 shows the propagation modelling stage of the LOUP system. The empirical method is based on a location fingerprinting method, which measures the signal strength values received from each of the transmitters in the area, at different locations around the given area. The theoretical method uses indoor signal propagation models to estimate the signal strength profile for a given area. Given the propagation model a radio map of the area is created, this is a map of the area and the given signal strength that is received at each location around the area.

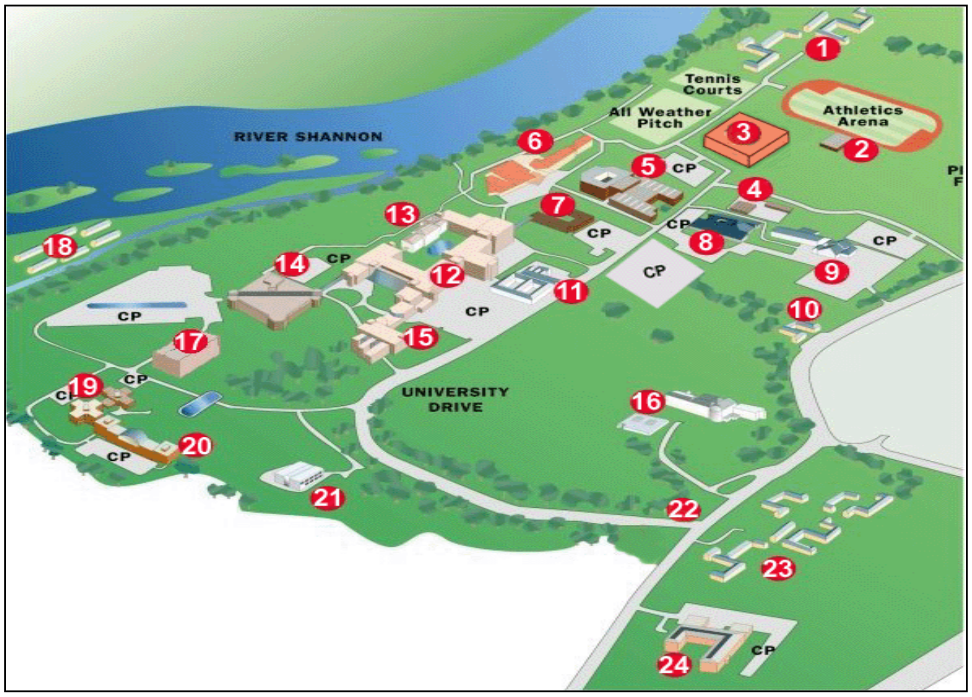
Figure 5.2 – Propagation Modelling Stage


1. Propagation Modelling
Signal strengths received at the user device around the given area
Empirical
Indoor propagation model formulas
Theoretical
Radio map of given area

5.2.1 Empirical Propagation Model

The empirical propagation model of an area is obtained by taking signal strength values at given locations around the area. The experimental testbed for this thesis is located on the first floor of the foundation building in the University of Limerick as shown by number 14 in the UL campus map in Figure 5.3.

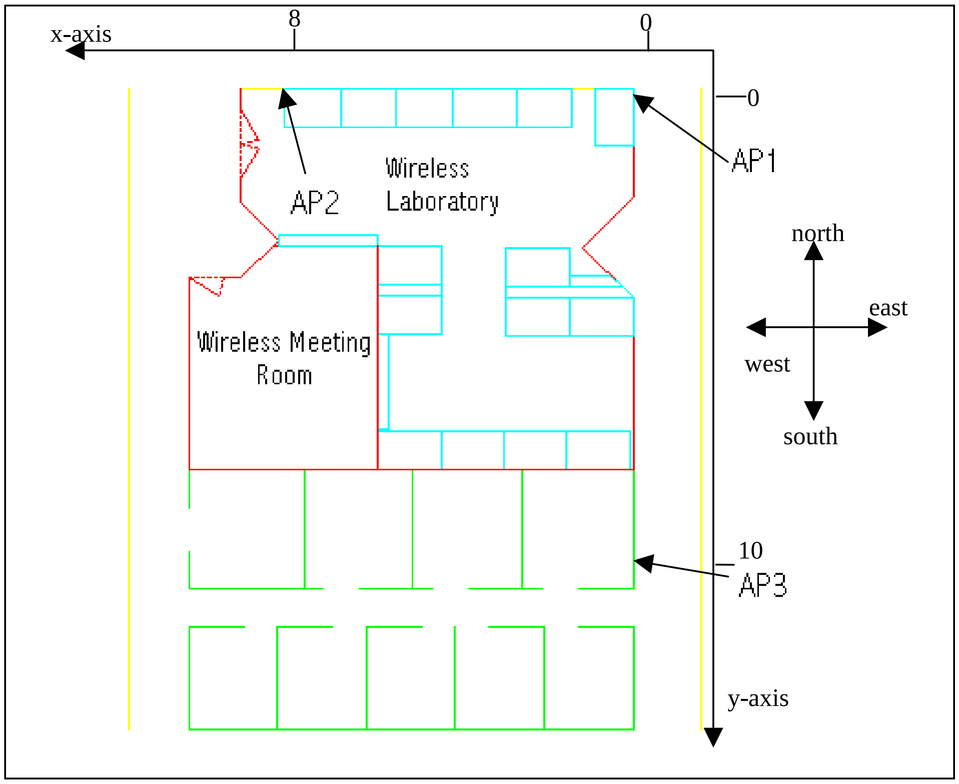
Figure 5.3 – UL Campus Map showing the Foundation Building, no. 14.



The layout of the area that is used for this thesis is shown in Figure 5.4, it has a dimension of 15m by 13m and an area of 195m^2. It consists of a laboratory area, a meeting room, corridors and nine cubicles. The three transmitters, AP1, AP2 and AP3, are located respectively at coordinates (0,0), (8,0) and (0,10). There is a scale of 1 to 1m on the coordinate system used. The transmitters used are Orinoco AP1000 access points [ORI]. The mobile host used to take the measurements is an IBM ThinkPad T23 laptop. Both the access points and the laptop are equipped with Lucent Technologies Network Interface Cards (NIC) supplied by Orinoco.

The Orinoco AP1000 access points come supplied with the Orinoco Client Manager software. This software reads the signal strength information from all access points in the area and can log this information. This is set to automatically log data every second and the results are read into a file. Measurements were taken every 1m^2 in the coverage area. Each location in the Testbed area gets a log file created for it. An example of a log file can be seen in Appendix C. It is the signal strength measurement that is of interest for this thesis. The noise level or Signal to Noise Ratio (SNR) could also be used but the signal strength is the most useful measurement for inferring location [BP00].

Figure 5.4 – Testbed layout of the First Floor of the Foundation Building



Different colours shown in Figure 5.4 represent different materials. The yellow line represents a block cement wall. The blue line represents wooden furniture, e.g. desks or bookshelves. The red line represents floor to ceiling internal partitions and the green line represents cubicle partitions, which are 1.47m high.

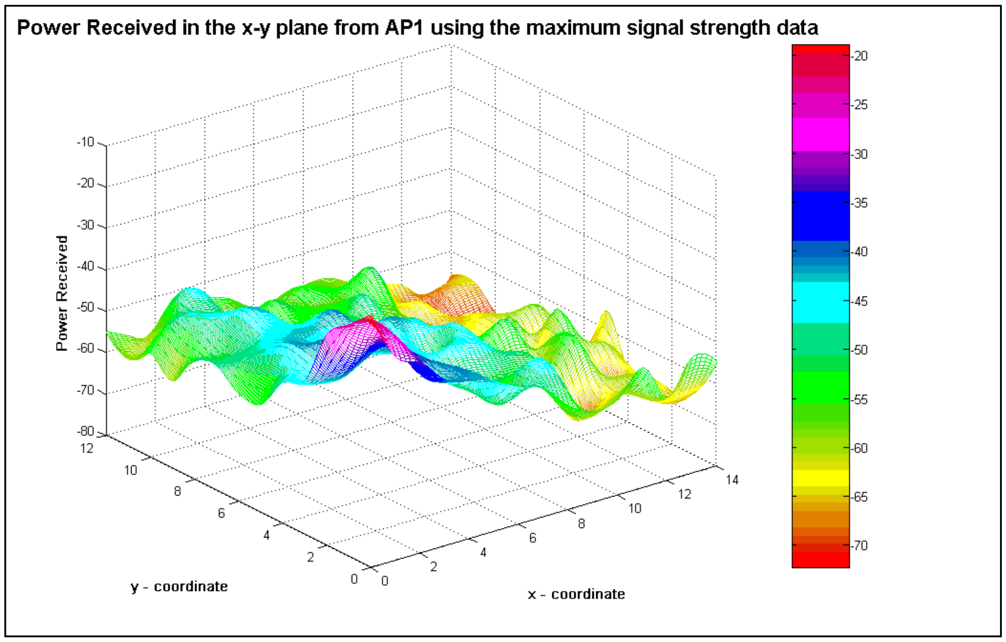
For each location in the grid area a total of 40 measurements were taken, 10 in each of four directions, which for simplicity are called north, south, east and west. These directions are shown in Figure 5.4. Signal strength at a given location can vary by up to 5dB depending on the user's orientation. For each of these locations, two filtering methods are used to determine the final signal strength measurement to store in the database. These are the maximum filter and the average filter, which work as expected by taking either the maximum signal strength received at a location or taking the average signal strength received. A matrix of the maximum and the average value has been created for each of the access points. For example, the maximum signal strength received at each location from AP1 for a portion of the room is given in Table 5.2. As AP1 is located at position (0,0), the signal strength is at its highest at this point and decreases as you move away from the access point.

Table 5.2 - Maximum Signal Strength received from access point, AP1, in dB

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
0	-19	-31	-34	-40	-42	-44	-47	-49	-55	-58
1	-24	-25	-32	-42	-41	-45	-54	-47	-47	-62
2	-29	-38	-35	-38	-42	-50	-52	-52	-54	-59
3	-43	-45	-43	-39	-43	-47	-45	-46	-50	-52
4	-46	-43	-39	-44	-46	-45	-51	-44	-54	-57
5	-54	-42	-43	-44	-45	-42	-44	-48	-54	-54
6	-54	-45	-43	-43	-41	-50	-50	-48	-54	-57

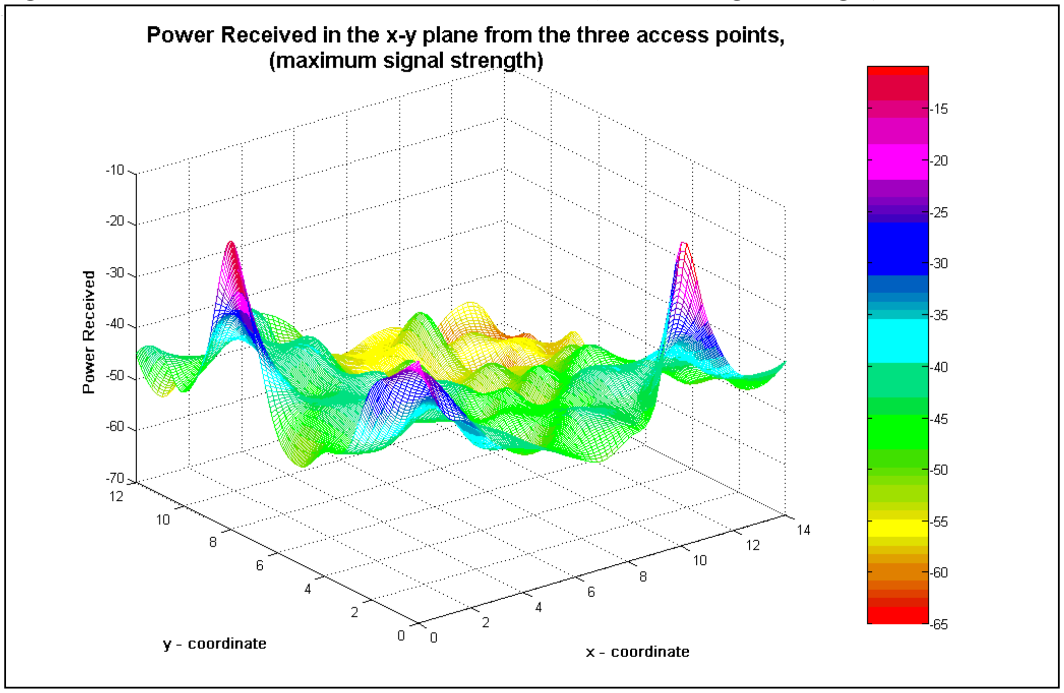
Similar matrices were created for each of the three access points. As well as the maximum signal strength, the average signal strength at each location is calculated and three matrices, one from each access point were created. These values were then modelled using Matlab. Figure 5.5 shows the 3-d signal strength model for the access point AP1 around the given experimental area.

Figure 5.5 – Power Received from AP1, (maximum signal strength)



A NIC connected to a laptop or other device, given a choice of several access points will connect to the access point that it receives the highest signal strength from. For this reason, the maximum signal strength from the three access points is chosen when modelling all access points on the same model, i.e. for each location on the map the three access points are compared and the one with the highest signal strength is connected to and the signal strength of that AP is modelled. Figure 5.6 gives the empirical data collected from the three access points, by modelling the maximum signal strength that is received at each location.

Figure 5.6 – Power Received from three Access Points, (maximum signal strength)



5.2.2 Theoretical Propagation Modelling

As well as the empirical propagation modelling, a theoretical model of the given area is used. A mathematical model of indoor signal propagation is used to generate a set of estimated signal strength data. The matrices that are created as in section 5.2.1 are again created but with estimated values instead of real measurements this time.

As given in Chapter 3 this thesis considers 4 different propagation models:

- Freespace
- One Slope Model (1SM)
- Multi-Wall Model (MWM)
- Linear Attenuation Model (LAM)

The power received, P_r, at a receiver is given by the following formula:

$$P_r = P_t + G_t + G_r - PL(d) \qquad \text{(Eqn. 5.1)}$$

where P_t is the power transmitted, in the case of the Orinoco silver card this is given as 15.05dB [ORI],

G_t is the gain of the transmitter antenna, in this case of this thesis set to 1dB,

G_r is the gain of the receiver antenna, in this case of this thesis set to 1dB and

PL(d) is the pathloss at distance d.

5.2.2.1 Freespace Propagation Model

Taking the equation for the pathloss in Freespace from eqn. 3.8 and adding it to eqn. 5.1 gives:

$$P_r = P_t + G_t + G_r - 40.046 - 20\log_{10}(d) \qquad \text{(Eqn. 5.2)}$$

This then gives the power received at a distance, d, from the access point using the Freespace propagation model. From this formula, the equivalent matrix for each of the access points can be produced. The resultant model from the maximum signal strength of the three matrices is given in Figure 5.7, with Figure 5.8 showing the model in 2-d.

Figure 5.7 – Power Received from the three AP's with the Freespace Model.

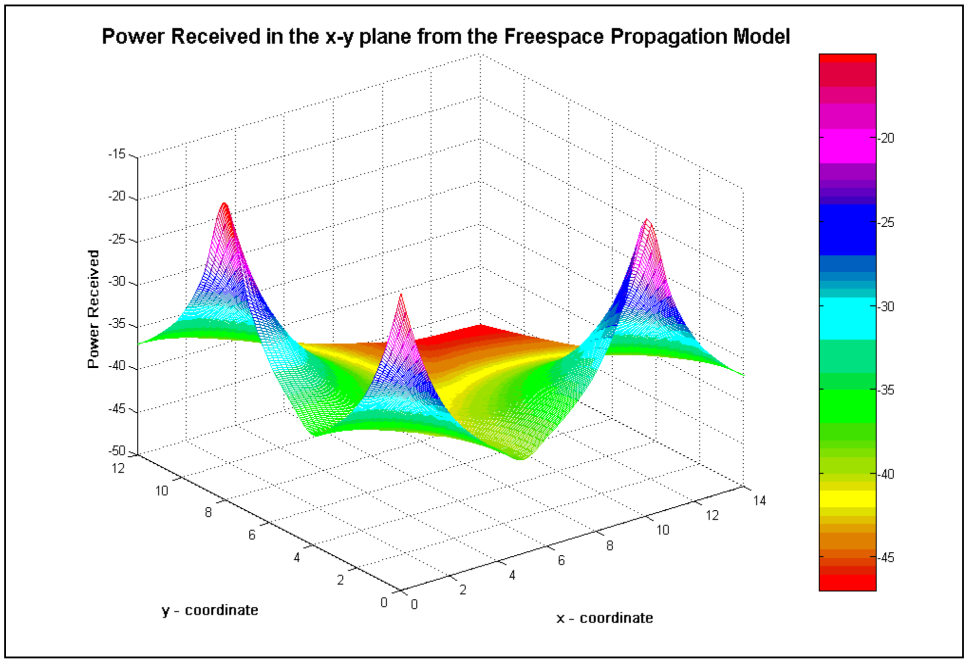
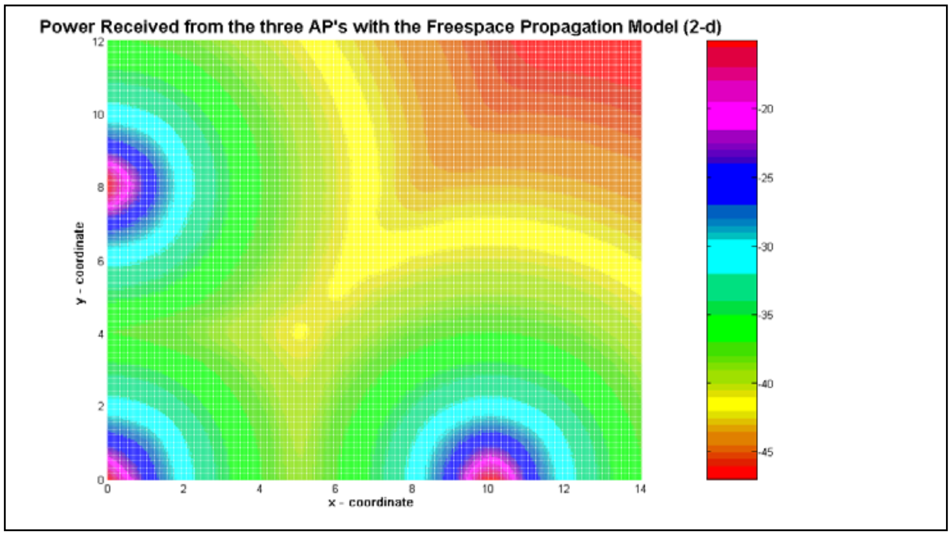


Figure 5.8 – Power Received from the three AP's with the Freespace Model in 2-d



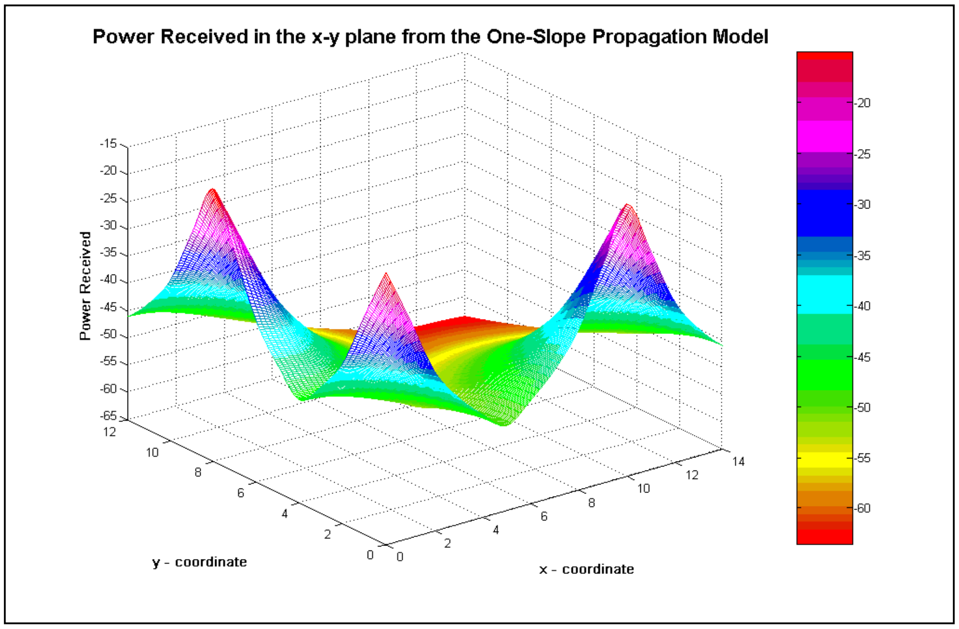
5.2.2.2 One Slope Model (1SM)

Taking the formula for the one slope model or log distance model as given in Eqn. 3.10 and eqn. 5.1 gives the power received from the access point at a distance d.

$$P_r = P_t + G_t + G_r - 40.046 + 35\log_{10}(d) \qquad \text{(Eqn. 5.3)}$$

Again, this will give the power received at a distance, d, from the access point and can be used to produce the matrices of power received for each of the access points. Figure 5.9 gives the resultant model from the three access points.

Figure 5.9 – Power Received from the three AP's with the One Slope Propagation Model



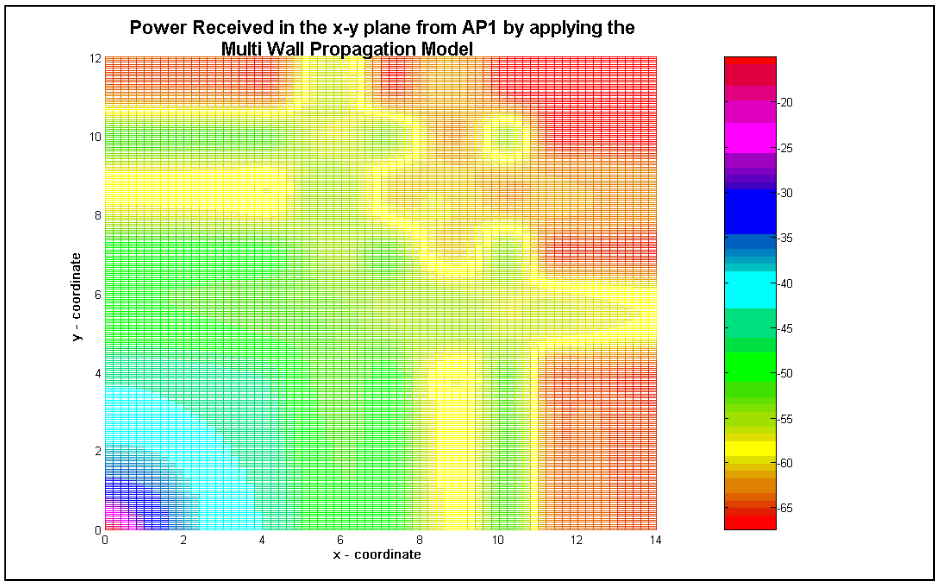
5.2.2.3 Multi-Wall Model (MWM)

Combining eqn. 2.12, which gives the pathloss for the MWM, with eqn. 5.1 gives the following:

$$P_r = P_t + G_t + G_r - 3.4 * t - 6.9 - 40.046 - 20\log_{10}(d) \quad \text{(Eqn 5.4)}$$

where t is the number of light walls between transmitter and receiver. Its value is taken from Table 3.4. This model can be more clearly shown in 2-d, to show the effect and position of the walls, as illustrated in Figure 5.10.

Figure 5.10 – Power Received from the three AP's with the Multi-Wall Propagation Model in 2_d



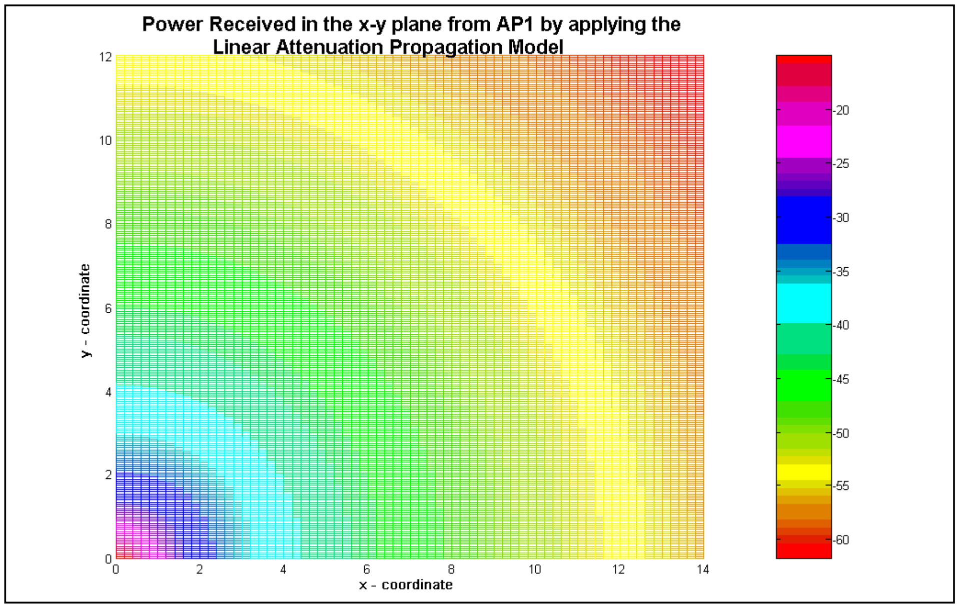
5.2.2.4 Linear Attenuation Model (LAM)

The pathloss for the linear attenuation model as proposed in [COS99] is given in eqn. 3.14. Adding this equation to Eqn. 5.1 gives the following:

$$P_r = P_t + G_r + G_t - 40.046 + 20\log_{10}(d) + \gamma * d \qquad \text{(Eqn.5.5)}$$

where γ is an attenuation coefficient, with value chosen as 0.62 from Table 3.5. The above equation gives the power received from the access point at a distance d. Figure 5.11 gives the resultant model from access point 1.

Figure 5.11 – Power Received from AP1 with the Linear Attenuation Propagation Model



5.2.2.5 Modelling in Matlab

In order to model the data in Matlab, the following steps are carried out:

- Create an empty matrix of same size as the coverage area, i.e. 15x13
- For each position in the matrix, calculate the Euclidian distance to the access point, for example position (3,4) in the matrix, which corresponds to the same coordinate in the map is a distance of 5m from the access point[2]
- This distance is put into the corresponding position in the matrix, now called the distance matrix

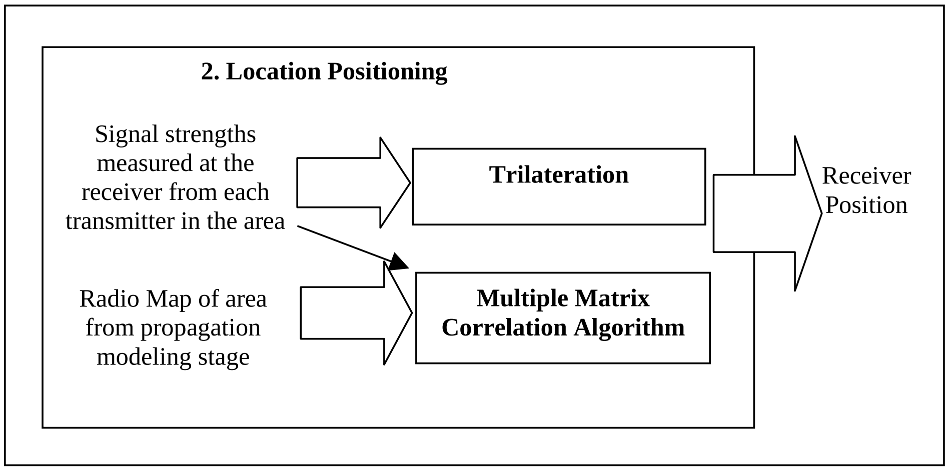
[2] From the equation $d = \sqrt{\left((x_2 - x_1)^2 + (y_2 - y_1)^2\right)}$, where the access point coordinates, i.e. (0,0) = (x_1,y_1) and the position coordinates, i.e. (3,4) = (x_2,y_2).

- Create a new matrix of same size
- For each position in the new matrix, take the distance in the corresponding position of the distance matrix and use this distance to calculate the power received from one of the propagation models
- Place this power in its corresponding position in the new matrix, now called the power matrix
- Interpolate the data with a resolution of 0.05. Interpolation is a process for estimating values that lie between known data points.
- Plot the resultant data using a mesh graph as shown in Figure 5.7

5.3 Location Positioning

After creating the propagation models for the area the user's location can be determined. The Location Positioning stage of the LOUP system is shown in Figure 5.12. In previous work, this has generally been achieved by the use of triangulation or trilateration. As well as using trilateration to solve for the user location, this thesis introduces a novel algorithm, called Multiple Matrix Correlation (MMC) to determine the location position.

Figure 5.12 – Location Positioning within the LOUP System


2. Location Positioning
Signal strengths measured at the receiver from each transmitter in the area
Radio Map of area from propagation modeling stage
Trilateration
Multiple Matrix Correlation Algorithm
Receiver Position

Before this can be achieved, the WLAN NIC card at the user device must measure the signal strength received from each of the transmitters in the area. Once again, 40 measurements are taken at the receiver position. The average or maximum of these measurements is then calculated as appropriate, depending on which filter method was used in creating the empirical propagation model. These values are stored in an array and then used for determining the location position.

Trilateration has been used predominately to solve position for this type of application. The signal strength received at the user device from each of the access points is known. Using this information, the distance to each of the transmitters can be calculated by working backwards with the signal propagation formula. From these distances and the known fixed positions of the transmitters, triangulation gives the user location.

The MMC algorithm takes the signal strengths received at the user device from each of the transmitters and correlates them with the databases of the signal strengths from each of the transmitters in the area. These databases are already available from the propagation modelling, as previously discussed. The closest match is given as the user location.

5.3.1 Trilateration

Trilateration is similar to the navigational technique called triangulation. Both techniques make use of the geometry of triangles to determine a relative position between points. Triangulation uses both distances and angle measurements, while trilateration uses only distance measurements to determine position [TET01]. At the users device, the Client Manager software and the NIC records the signal strength from each of the access points in the area. These signal strengths are first converted back into distances by using the appropriate propagation formula. The equations used for each of the models to convert the power received to the distance are given below.

- Freespace Model

$$d = 10^{\left(\frac{P_t + G_r + G_t - P_r - 40.046}{20}\right)} \qquad \text{(Eqn. 5.6)}$$

- One-Slope Model

$$d = 10^{\left(\frac{P_t + G_r + G_t - P_r - 40.046}{35}\right)} \qquad \text{(Eqn. 5.7)}$$

- Multi-Wall model

$$d = 10^{\left(\frac{P_t + G_r + G_t - P_r - 40.046 - 3.4 * t - 6.9}{20}\right)} \qquad \text{(Eqn. 5.8)}$$

In this thesis there are three transmitters or Access Point and therefore there is an array of three signal strengths, one for each of the AP's. These values are then converted into distances using the appropriate equation above. The equation chosen is the corresponding model that is used in the theoretical propagation modelling stage. Once the three distances are calculated, the position of the user can be determined by trilateration. The trilateration method uses the known fixed positions of the access points and the distance the receiver is from these three access points. If there are two access points, the position can be determined in 2-d. With three access points, the position can be determined in 3-d. This thesis is mainly concerned with 2-d, as it is assumed that the user device will be located at desk height. To demonstrate how trilateration works, the experimental testbed is used, see Figure 5.13. The access points are located as follows:

- AP1 – (0,0)
- AP2 – (0,10)
- AP3 – (8,0)

The unknown position of the user device has coordinates, (X, Y). The distances from the user device to the access points are calculated as shown above to be as follows:

- From AP2 – √13 = L1
- From AP1 – √53 = L2
- From AP3 – √85 = L3

The position of the device is first calculated using AP1 and AP2. The distance between AP1 and AP2, called Lb1 is calculated to be 10m. The line between the two points is defined as the x-axis as shown in Figure 5.13. The line perpendicular to this line and running through the first reference point, i.e. AP1, in the same plane, is defined as the y-axis. The altitude of the unknown position is the Y value. By applying Pythagoras theorem to each of the two triangles formed, the following two equations are formed.

$$L1^2 = Y^2 + (Lb_1 - X)^2 \qquad \text{(Eqn. 5.9)}$$

$$L2^2 = Y^2 + X^2 \qquad \text{(Eqn. 5.10)}$$

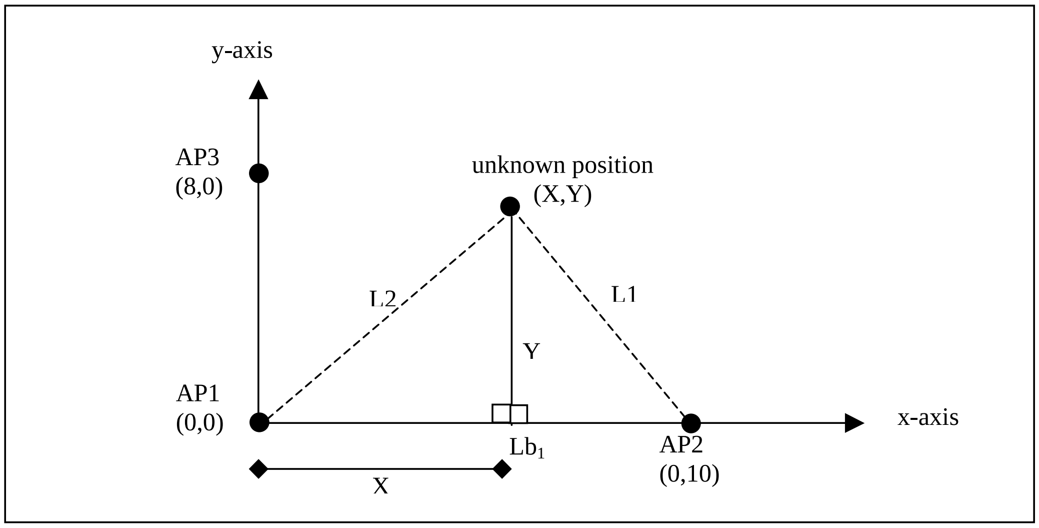
Solving for X and putting the results equal to each other gives the following equation.

$$L2^2 - X^2 = L1^2 - (Lb_1 - X)^2 \qquad \text{(Eqn. 5.11)}$$

This equation can then be solved for X.

$$X = \frac{\left(L2^2 - L1^2 + Lb_1^2\right)}{2Lb_1}$$ (Eqn. 5.12)

Figure 5.13 – Trilateration using two known fixed positions



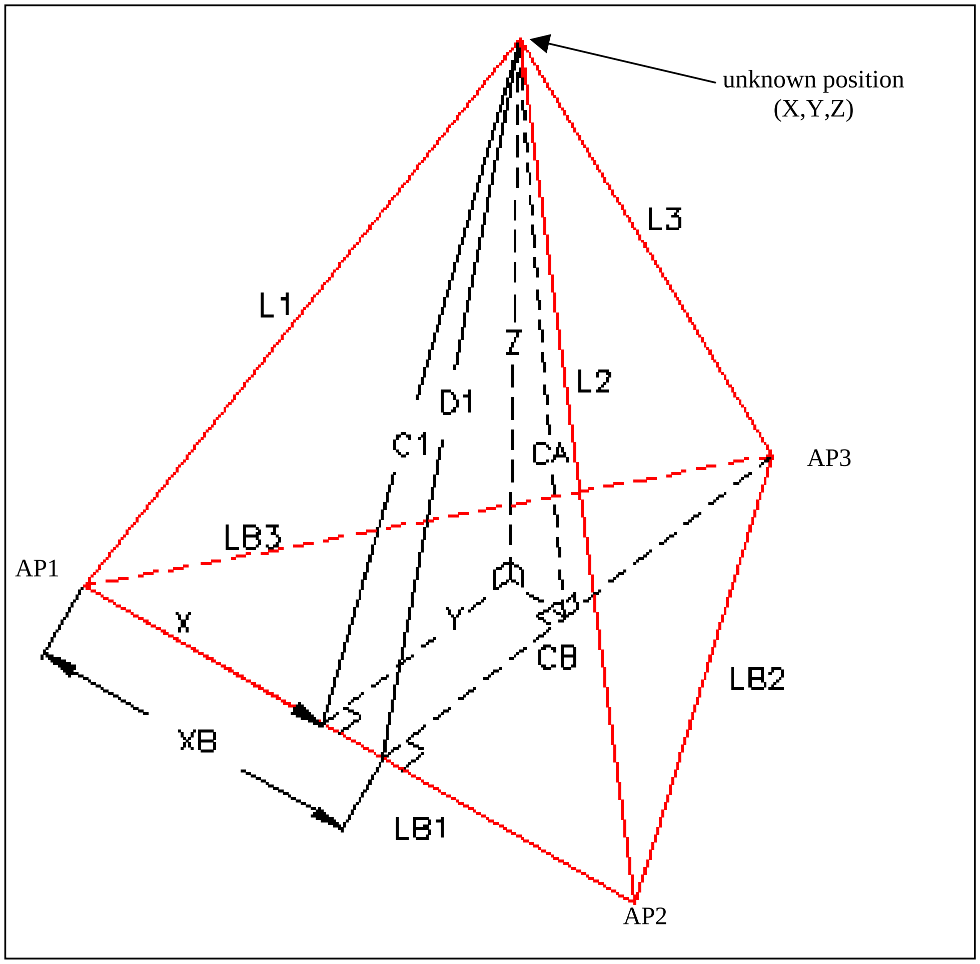
Solving X for the values given above, gives X = 7. With X known, Y can be calculated from Eqn. 5.11.

$$Y = \sqrt{L1^2 - X^2}$$ (Eqn. 5.13)

The above equation gives a value of Y = 2. This method could also be used with a combination of any of the two access points to determine the x and y position of the user.

In this thesis, there are three known fixed positions available. The three fixed positions are utilized to determine the coordinates of the unknown position. This is accomplished in a similar manner as above. The reason for using the three fixed positions is for possible extension of the thesis at a later date to include 3-d coordinates, which can easily be done using the three fixed positions as will be shown next.

Figure 5.14 – Tetrahedron formed by four points



For a 3-d coordinate system, the x-axis is once again a line between the first two reference points, i.e. AP1 and AP2. The y-axis is a line perpendicular to the x-axis running through the first reference point in the same plane as the three points and in the direction of the third reference point, AP3, as shown in Figure 5.13. The z-axis is defined by a right-handed coordinate system between the x and y-axes. By connecting the four points, that is the three reference points and the unknown point, (X, Y, Z), with lines a geometric entity known as a tetrahedron is formed. Figure 5.14 taken from [TET01], shows the tetrahedron that would be produced.

The distance between AP2 and AP3, shown as Lb2 is calculated to be $\sqrt{164}$ and the distance between AP1 and AP3, shown as Lb3 is calculated to be 8. The X coordinate is determined in the same manner as before from equation 5.13. The Y coordinate is determined in the following way:

$$C_1 = \sqrt{L1^2 - X^2}$$ (Eqn. 5.14)

The formula arises from Pythagoras theorem, as C1 is perpendicular to X. C1 is then equal to 2. Xb is calculated in a similar manner as X as seen.

$$Xb = \frac{Lb_3^2 - Lb_2^2 + Lb_1^2}{2Lb_1}$$ (Eqn. 5.15)

This gives Xb equal to 0.

$$Cb = \sqrt{Lb_3^2 - Xb^2}$$ (Eqn. 5.16)

This gives Cb equal to 8.

$$D_1 = \sqrt{C_1^2 + (Xb - X)^2}$$ (Eqn 5.17)

D1 equals to √53. Finally the value of Y can be determined from the following equation.

$$Y = \frac{\left(D_1^2 - L3^2 + Cb^2\right)}{2Cb}$$ (Eqn. 5.18)

This gives the value of Y = 2, which is the same as the value calculated for Y from the 2-d example. The Z coordinate if needed can then be calculated from the following equation.

$$Z = \sqrt{C_1^2 - Y^2}$$ (Eqn. 5.19)

The file trilat.m in Appendix B gives the Matlab implementation of the use of trilateration in a 2-d coordinate system to solve for an unknown position.

5.3.2 Multiple Matrix Correlation (MMC)

The MMC algorithm given in this thesis was developed as an alternative to trilateration. The trilateration method is based on using an indoor propagation model to determine the distances to the transmitters. Therefore, it can only ever be as accurate as the propagation model. The MMC algorithm does not rely on propagation models for positioning. Indeed, it can work on a radio map that is created from either a propagation model or from empirical data. It solves the X, Y coordinates of an unknown position. Each position in a given area that has three or more access points will have a unique fingerprint. The fingerprint is based on the different signal strengths received from the transmitters.

In this thesis, there are three access points in the given experimental area. Therefore, a fingerprint in each position in the area will consist of three different signal strengths, one from each of the access points. As a receiver moves into one of these positions, it will take a fingerprint of that area, i.e. measure the signal strength from each of the access points. Thus, in order to determine its position, a match has to be found between its fingerprint and the fingerprints already calculated for the given area. In Figure 5.15 is an outline of the algorithm. For a full flow chart, see Appendix D and for the implementation in Matlab see mmc_method.m in Appendix B.

Figure 5.15 – Outline of the MMC Algorithm

- inputs – three matrices holding the signal strength reading for the given area from the three access points, called PR1, PR2 and PR3 and three signal strength readings from the receiver from each of the access points, called pow_rec_1, pow_rec_2 and pow_rec_3
- outputs – the X, Y coordinates of the receiver position
- Initialise a matrix, called test, that is the same size as PR1 and populate the matrix with 1's.
- Initialise a counter to 2, which will hold the number of matches at a given time.
- Initialise an integer, power_diff_1 to 13, to hold the allowable difference between the signal strength received, pow_rec_1 and the signal strength stored in the database.
- Initialise an integer, power_diff_2 to 9, to hold the allowable difference between the signal strength received, pow_rec_2 and the signal strength stored in the database.
- Initialise an integer, power_diff_1 to 2, to hold the allowable difference between the signal strength received, pow_rec_2 and the signal strength stored in the database.
- Initialise an integer inc_1 equal to 2, to hold a value that will increase the power_diff_1 if it gets too small
- Initialise an integer inc_2 equal to 2, to hold a value that will increase the power_diff_2 if it gets too small
- Initialise an integer inc_3 equal to 2, to hold a value that will increase the power_diff_3 if it gets too small
- While count > 0 do the following
    1. count equals 0
    2. Compare pow_rec_1 with all values in PR1, for values of PR1 that are within power_diff_1 of pow_rec_1, place a 1 in the corresponding position in the test matrix and increment count. Otherwise place a 0 in the corresponding position.
    3. if count is greater than 1 do the following
        - power_diff_1 equals power_diff_1divided by 1.4
    4. else if count is less than 1 do the following
        - power_diff_1 equals power_diff_1 multiplied by inc_1
        - test equals to 0 for all values in the matrix
        - inc_1 equals inc_1 plus 1
    5. count equals zero

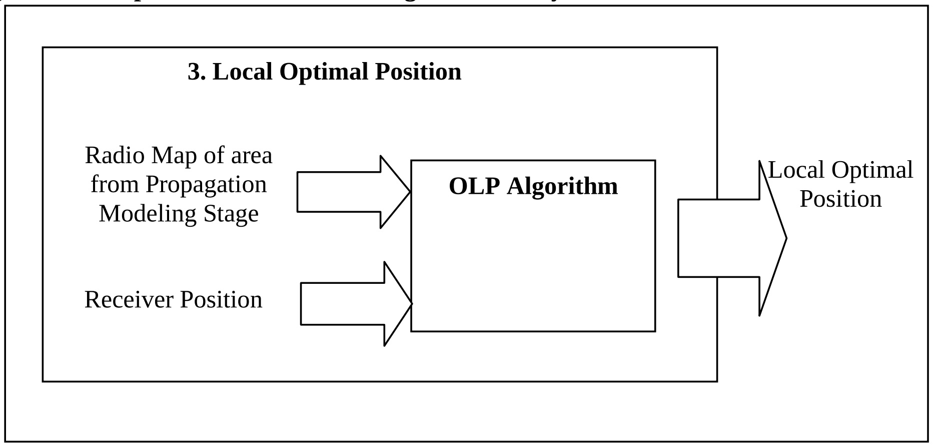
6. For all values of 1 in the test matrix, compare the value that is located in the position of PR2, corresponding to the position of value 1 in the test matrix, with pow_rec_2. If they are within power_diff_2 apart, place a 1 in the corresponding position of the test matrix and increment count. Otherwise, place a 0 in the corresponding position.
7. if count is greater than 1 do the following
    - power_diff_2 equals power_diff_2divided by 1.4
8. else if count is less than 1 do the following
    - power_diff_2 equals power_diff_2 multiplied by inc_2
    - test equals to 0 for all values in the matrix
    - inc_2 equals inc_2 plus 1
9. count equals zero
10. Repeat step 6, 7 and 8 with PR3 and pow_rec_3.
11. If the counter is equal to 1, return the position of the 1 in the test matrix as the coordinate of the receiver and end of algorithm.

5.4 Optimal Local Position

The next step is to determine the local optimal location for the user, as shown in Figure 5.16. An innovative algorithm called Optimal Local Position (OLP) is introduced to solve this problem. At this stage, the user location is ascertained and the received signal strength from each of the transmitters given, i.e. the radio maps of the area from each of the transmitters. OLP first determines if the signal strength at the current location is within a given acceptable threshold, e.g. within 60% of the transmitted power level. If this is the case, the user does not have to move, otherwise, the nearest location that is within this given threshold is computed.

The user on start-up of the system can set the threshold to a certain percentage. These percentages could be mapped onto certain classes of service, see Table 5.1. If, for example, the user desired a Class A service, the threshold would be set to 80%.

Figure 5.16 – Optimal Local Position Stage of LOUP System


3. Local Optimal Position
Radio Map of area from Propagation Modeling Stage
Receiver Position
OLP Algorithm
Local Optimal Position

The algorithm searches the area immediately around the user position and finds the maximum signal strength in this area. This signal strength is then tested to determine if it is within the given threshold. If so, the algorithm finishes and returns the coordinates of the new position. If the maximum signal strength is not within the given threshold, the search area is widened. This will continue until a value within the threshold is found. The pseudocode for this algorithm is given in Figure 5.17, a flowchart of the algorithm is shown in Appendix E and the implementation of the algorithm in Matlab in Appendix B.

Figure 5.17– Pseudocode for the OLP Algorithm

```
•  initiate search_area equal to one
•  if current_signal_strength is greater than threshold
      o  test equals true
      o  new_signal_strength equals current_signal_strength
•  else
      o  test equals false
•  while test equals false
      o  new_signal_strength equals maximum of signal strengths in
         search_area
      o  if new_signal_strength is greater than threshold
            ▪  test equals true
      o  search_area equals search_area plus one
•  return coordinates of new_signal_strength
```

5.5 Pathfinding

Finally, the system will inform the user how best to move from their current position to the new position, taking obstructions into account. This is achieved using the heuristic search algorithm A*, which finds the shortest path between two points [HNR68]. The implementation of this can be seen in Appendix F. Figure 5.18 shows the pathfinding stage in the LOUP system.

Figure 5.18– The Pathfinding Stage of the LOUP System

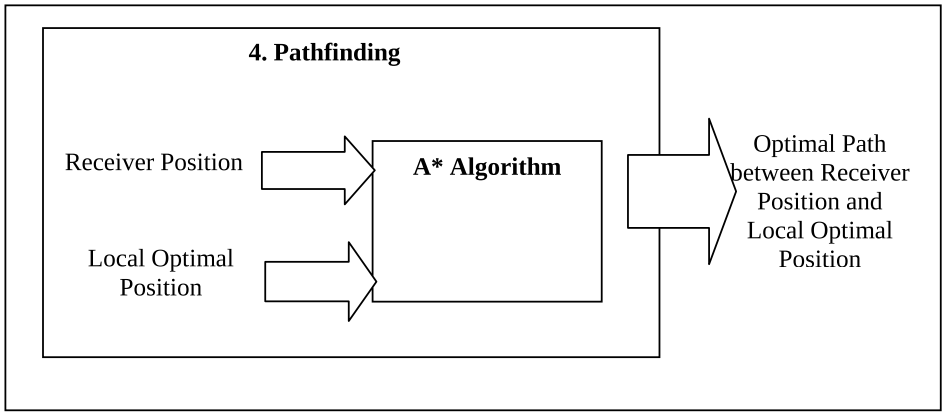
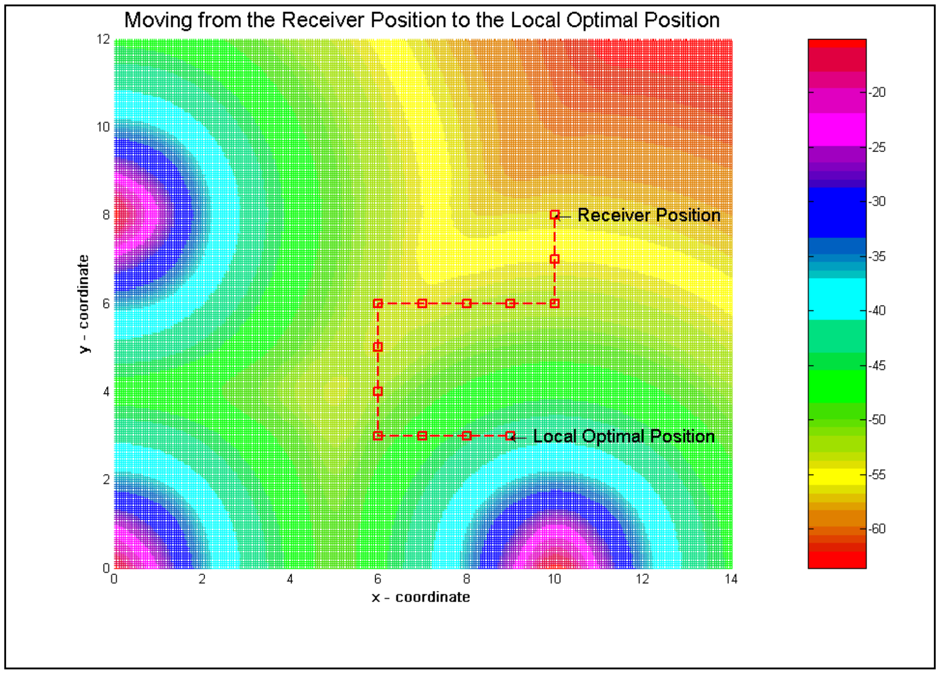

4. Pathfinding
Receiver Position
Local Optimal Position
A* Algorithm
Optimal Path between Receiver Position and Local Optimal Position

Figure 5.19 below demonstrates the A* algorithm directing the user from the receiver position to the Optimal Local Position. The red dashed line indicates the path that should be followed by the user. The model that is used in this example is the One Slope Model. The receiver position is initially calculated to be at coordinates, (10,8). The OLP algorithm then determines the local optimal position to be at (9,3).

Figure 5.19 - Demonstrating the A* algorithm directing the User from the Receiver Position to the Optimal Local Position



The A* algorithm then determines the shortest path between these two points to be along the coordinates as shown in Figure 5.20, which gives the output of the A* algorithm.

Figure 5.20 – Output from the A* Algorithm

```
Search found goal state
Node position : (10,8)            Node position : (10,7)
Node position : (10,6)            Node position : (9,6)
Node position : (8,6)             Node position : (7,6)
Node position : (6,6)             Node position : (6,5)
Node position : (6,4)             Node position : (6,3)
Node position : (7,3)             Node position : (8,3)
Node position : (9,3)
Solution steps: 25                SearchSteps : 18
```

The obstacles on the map are represented by a matrix of same size as the coverage area for the building, i.e. 15 by 13, in the A* algorithm. A value of 1 in this matrix represents a terrain in the map that is easy to travel over, for example, an area with no obstacles. A value of 9 in the matrix represents a wall in the building. For clarity in the example shown in Table 5.3, which is using the one-slope propagation model, there is only one wall on the x-axis and one wall on the y-axis as can be seen below.

Table 5.3 – Matrix used by A* algorithm to represent the map

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	1	1	1	1
4	1	1	1	1	1	1	1	1	1	1	1	1	1
5	1	1	1	1	1	1	1	9	9	9	9	9	9
6	1	1	1	1	1	9	1	1	1	1	1	1	1
7	1	1	1	1	1	9	1	1	1	1	1	1	1
8	1	1	1	1	1	9	1	1	1	1	1	1	1
9	1	1	1	1	1	9	1	1	1	1	1	1	1
10	1	1	1	1	1	9	1	1	1	1	1	1	1
11	1	1	1	1	1	9	1	1	1	1	1	1	1
12	1	1	1	1	1	9	1	1	1	1	1	1	1
13	1	1	1	1	1	9	1	1	1	1	1	1	1
14	1	1	1	1	1	1	1	1	1	1	1	1	1

5.6 Summary

This chapter introduced the Local Optimal User Position (LOUP) system and demonstrated how the concepts of the system are implemented to produce the final system. The LOUP system identifies the location of the user, determines an optimal local position for the user to achieve an increased class of service and then directs the user to this optimal location.

The first step is propagation modelling of the indoor environment. This is achieved using empirical data and using theoretical propagation models. The empirical data propagation model is more accurate than the theoretical models. However, the theoretical modelling removes the need to perform many cumbersome measurements in the indoor environment.

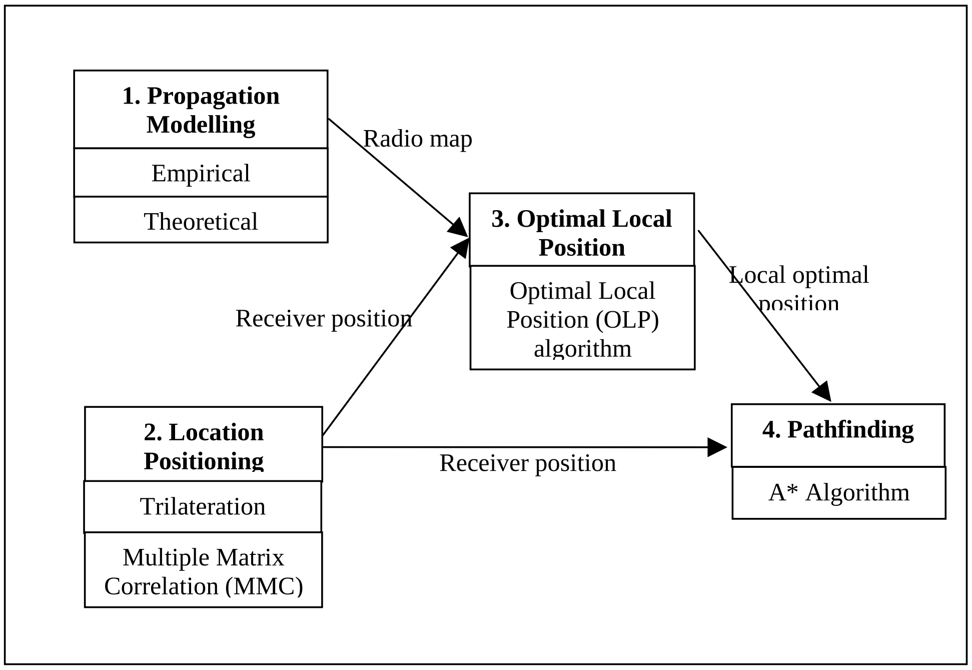
The position of the user is identified, first by trilateration and then by the MMC algorithm. The OLP algorithm determines the Optimal Local Position for the area that the user is located in. Finally, the user is directed to this optimal position with the A* search algorithm.

6 Performance of the Local Optimal User Position (LOUP) System

6.1 Introduction

This chapter presents results obtained during the experimental phase of this thesis. Firstly, the empirical propagation models of the area that were created are given. Next, the theoretical models are compared to the empirical data. Results from location positioning using both trilateration and the MMC algorithm are then imparted. The resulting Optimal Local Position from certain receiver positions is then demonstrated. Finally the chapter gives examples of the shortest path between the two points.

Figure 6.1 – Local Optimal User Position (LOUP) System



6.2 Propagation Modelling

The propagation models for the area in this thesis as stated in Chapter 5, were created in two different manners. The empirical method is based on taking measurements from different locations around the area, while the theoretical method is based on estimating the signal strength using indoor signal propagation models.

6.2.1 Empirical Propagation Model

The empirical based model takes 40 signal strength measurements from each location. The measurements are taken every 1m apart, with 10 measurements taken in every direction, north, south, east and west. The model created from each of the user directions are shown. Figure 6.2 gives the model of the area, with the user-facing north, from the three access points.

Figure 6.2 – Power Received from Empirical Data with the User facing north using maximum filter

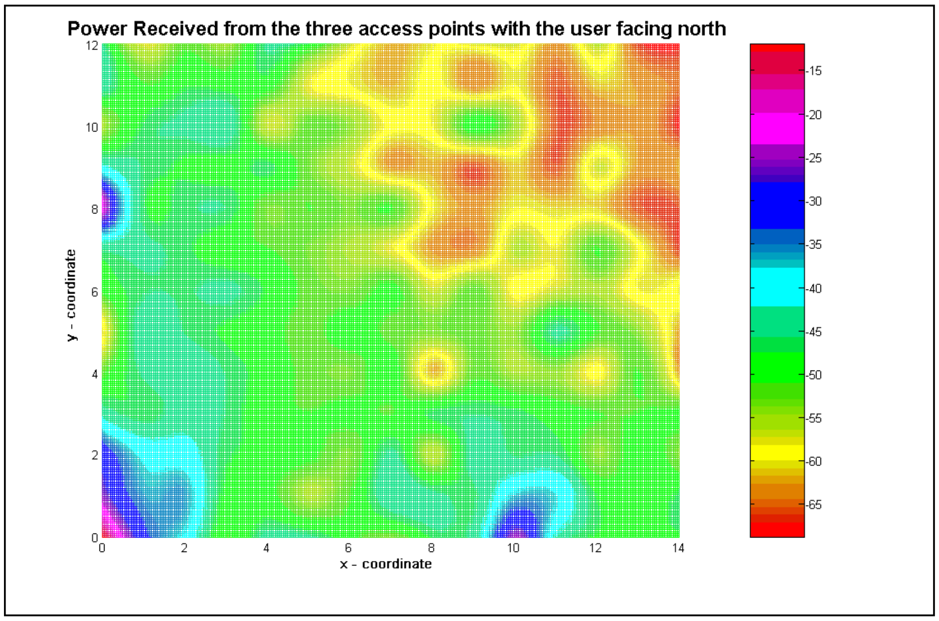


Figure 6.3 – Power Received from Empirical Data with the User facing east

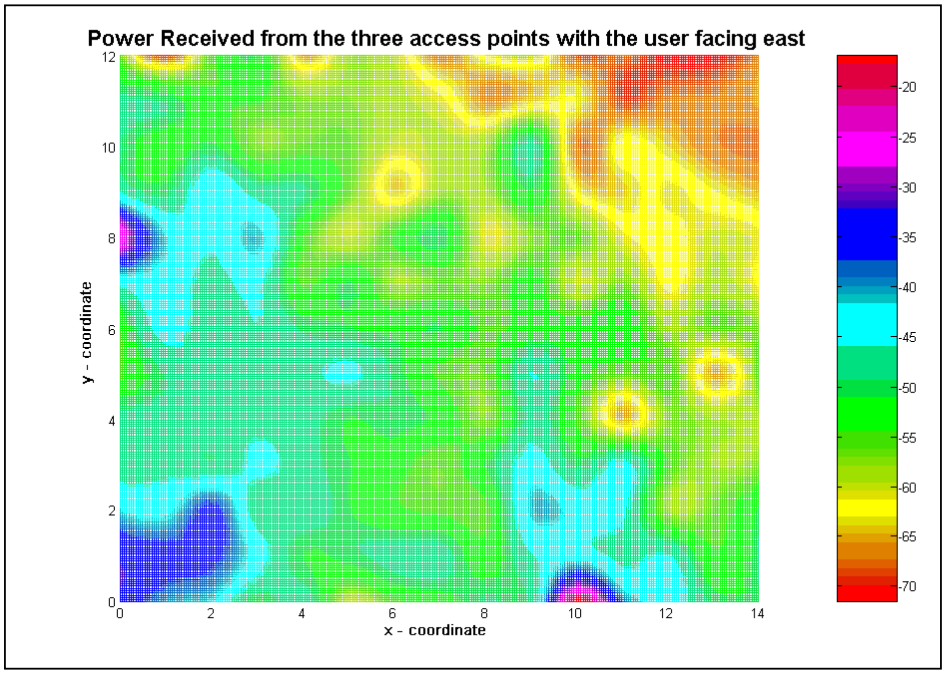


Figure 6.4 – Power Received from Empirical Data with the User facing south

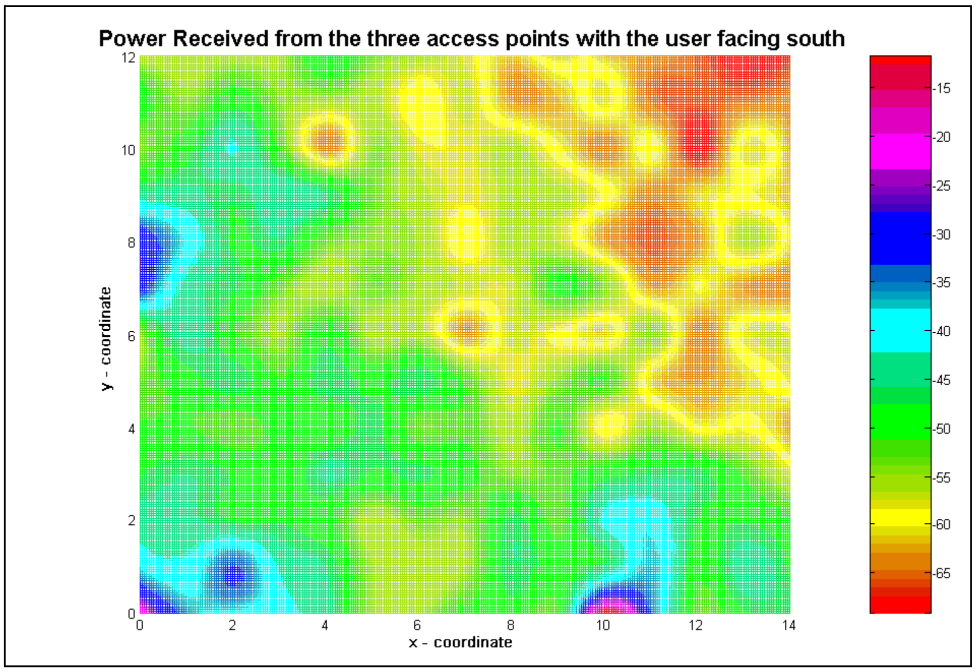
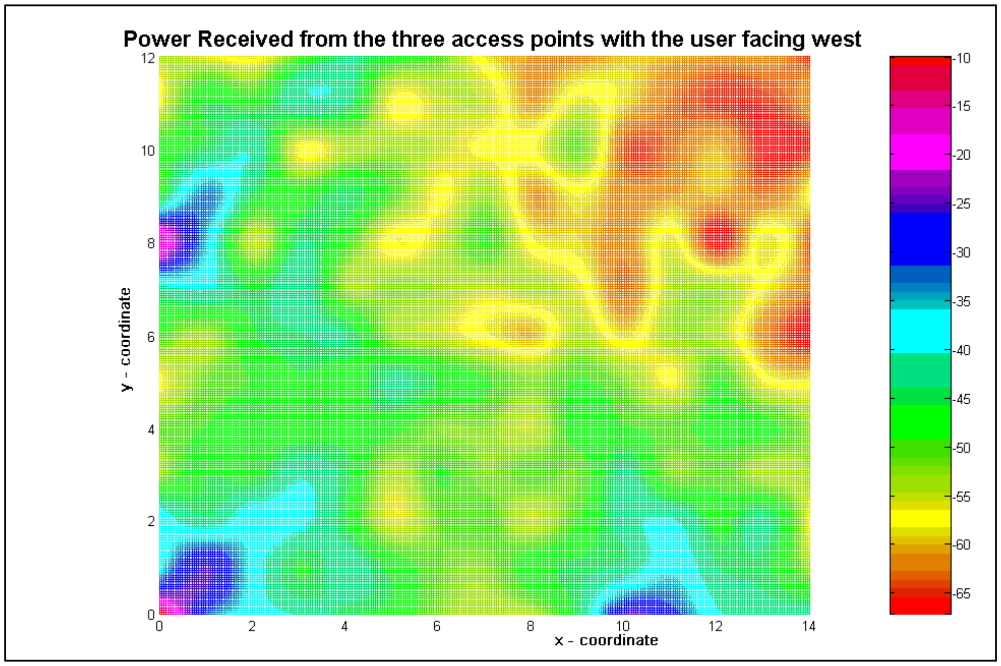


Figure 6.5 – Power Received from Empirical Data with the User facing west

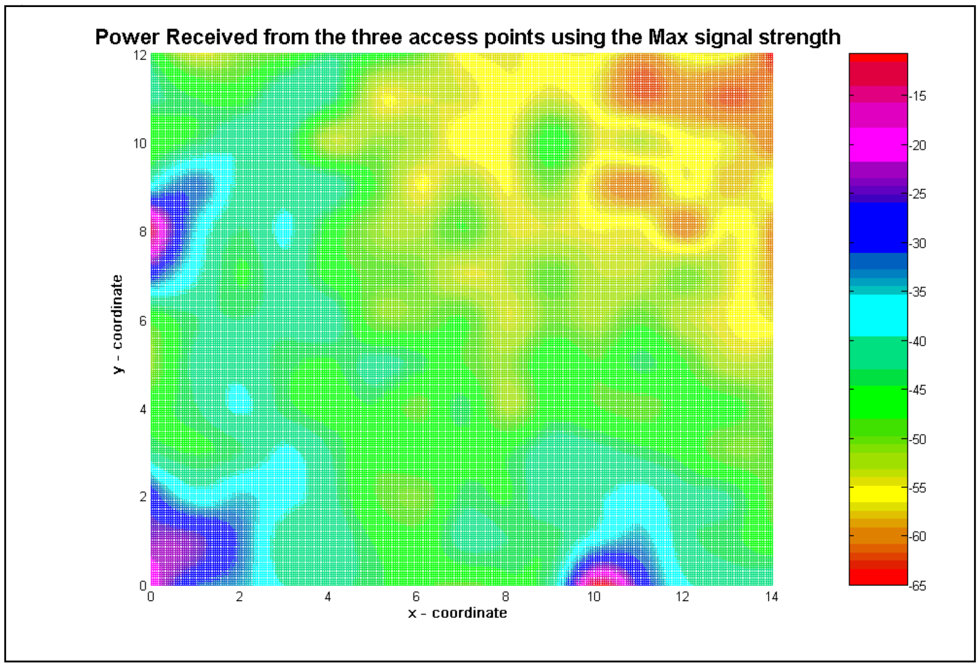


The above models are based on the average signal strength at each location used as a filtering method. For each location, 10 different measurements were taken in each direction. The average of these 10 measurements was then calculated and this value was stored in a matrix. Each access point has four matrices, one for each direction, i.e. ap1_north, ap1_east, ap1_south and ap1_west.

A more useful filtering method used the maximum signal strength in each location. The reason for this is that any abnormal deviation in the radio propagation environment could lead to a significant deviation in the signal strength measured. This measurement then altered the average signal strength for that location. Taking the maximum signal strength received at each location will give a more accurate propagation model, assuming that the maximum is also chosen at the receiver for determining the location position. Although the signal strength chosen will still be affected by deviations giving rise to spikes in the signal strength, it will not be affected by dips in the signal strength.

For each location, there are still 40 different measurements, one for each direction. The maximum of these 40 measurements is now calculated and stored in a matrix. In this case, each access point has one matrix, containing the maximum signal strengths received around the given area. Figure 6.6 shows the empirical data model created from the three access points taking the maximum signal strength.

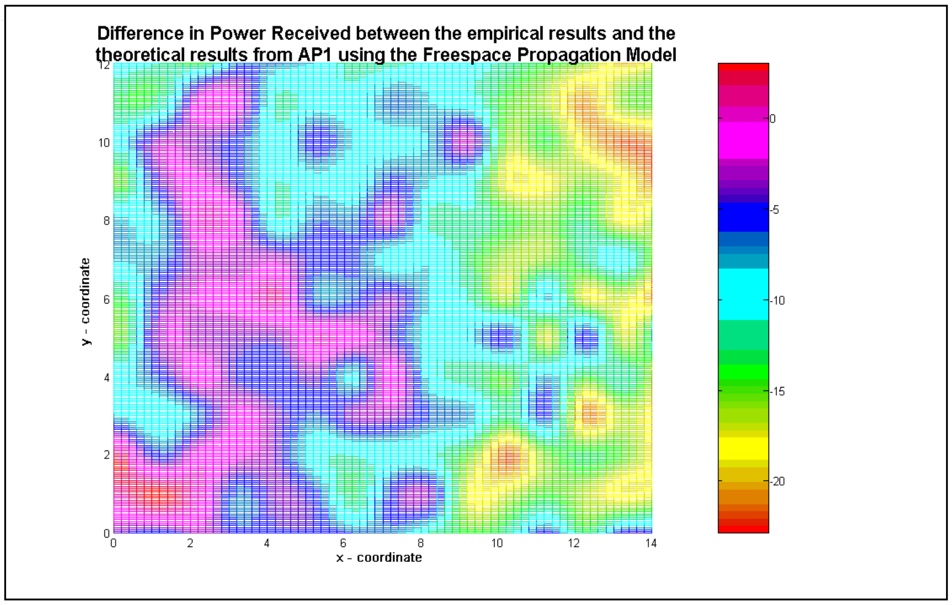
Figure 6.6 – Power Received from Empirical Data with the Maximum Data at each Location



6.2.2 Theoretical Modelling

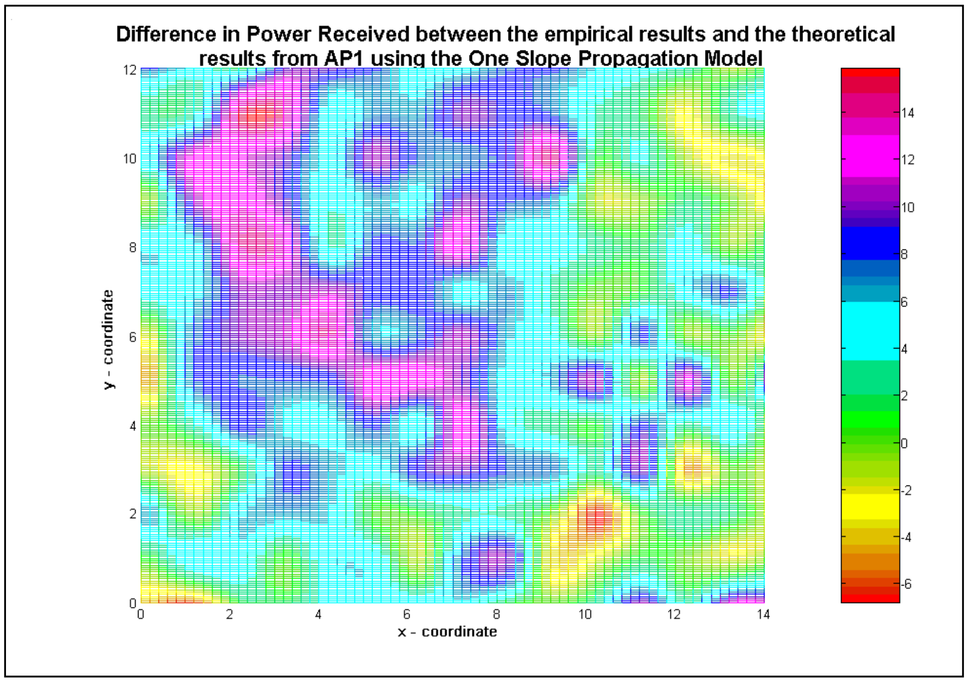
The models created from the four different propagation models are shown in Chapter 5. These models were compared with the empirical model given in Figure 6.6 to find the model that best represented the indoor environment. For clarity and ease of comparison, the models are compared using one access point in the environment. This is AP1 at position (0,0). Figure 6.7 shows the differences in the power received from the empirical data and the power estimated by the Freespace propagation model.

Figure 6.7 – Difference in Power between Empirical Data and Freespace Propagation Model



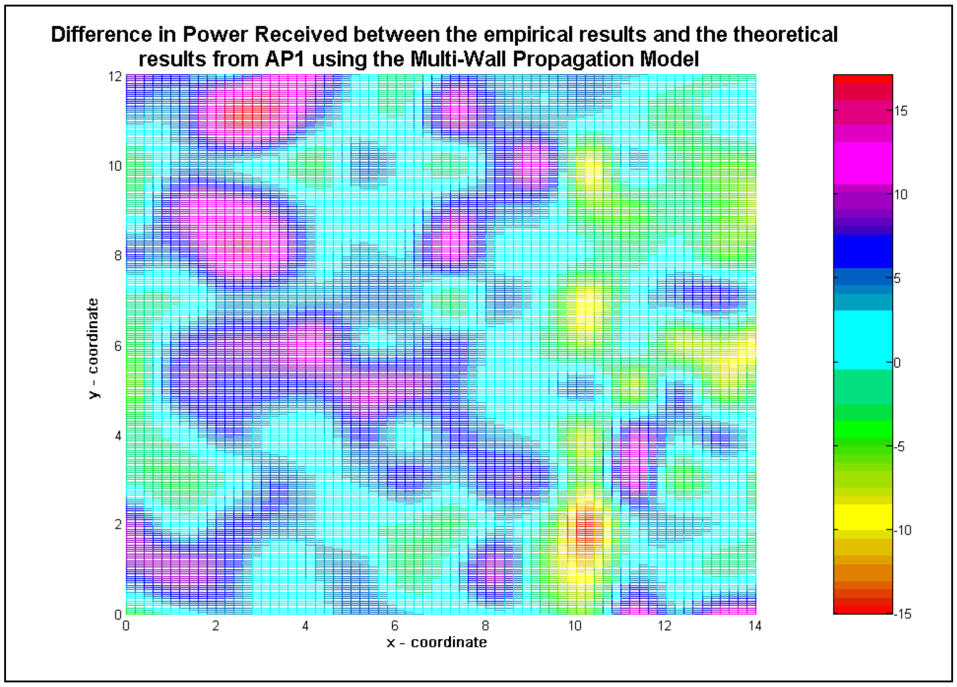
Close to the access point position, there is little or no difference but as the distance to the access point increases, so does the difference between the empirical and theoretical data. The inconsistencies in this arise from the unpredictability of the indoor radio environment, for example, pathloss due to multipath and pathloss due to shadowing. This model shows that the Freespace propagation model is not a suitable model for the given indoor environment. Next, the one-slope model is used and the difference between this and the empirical model is given in Figure 6.8.

Figure 6.8– Difference in Power between Empirical Data and One-Slope Propagation Model



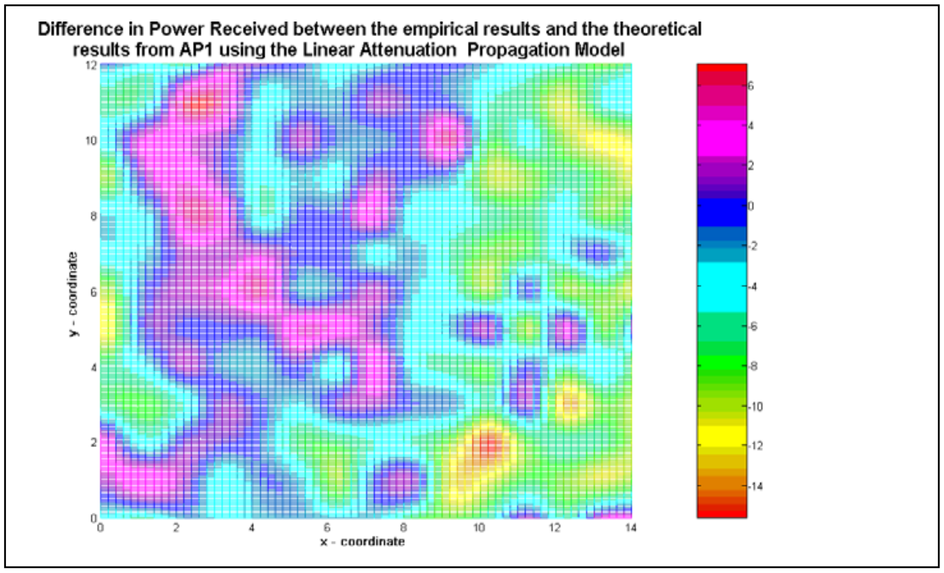
The one-slope model gives a more accurate estimation of the indoor propagation environment. However, there is still a difference of up to 14dB between the empirical data and the theoretical data in certain locations. The difference between the Multi-Wall Model and the empirical data is modelled in Figure 6.9.

Figure 6.9 – Difference in Power between Empirical Data and Multi-Wall Propagation Model



The Multi-Wall model shows much improvement over the One-Slope Model in this indoor environment. In general, the Multi-Wall Model is reasonably consistent with the empirical data. However, there are still several locations that have a difference of up to 15dB and one location that has a difference of –15dB. Finally, the difference between the Linear Attenuation Model (LAM) and the empirical data is shown in Figure 6.10.

Figure 6.10 – Difference in Power between Empirical Data and Linear Attenuation Propagation Model



The LAM propagation model does not closely match the empirical data and is therefore not suitable for modelling the particular indoor environment in this thesis. As can be seen from the above Figures, the one-slope model and the Multi-Wall model are most suitable for modelling the indoor environment in this thesis, with the Multi-wall model giving a slightly improved estimation over the One-Slope Model.

The inaccuracies between the empirical data and the theoretical data are caused by a number of factors including interference due to noise, inaccuracies in the measurement technique and multipath due to the movement of people. In order to infer the location, the signal strength that is measured at the receiver needs to be approximately within 3dB of the signal strength recorded in the matrix. As a result of this, it is determined that the empirical data measured for this thesis is not accurate enough to use for location positioning. Since the aim of this thesis is not to prove radio propagation models but to identify a Optimal Local Position for a user to move to in order to gain an improved class of service, the use of the theoretical propagation models will suffice.

6.3 Location Positioning

The location of the user can now be determined using the propagation model and the signal strength received at the user device. The location is solved using two different methods, trilateration and database correlation. At the receiver, when determining the location, several measurements are also recorded from each of the access points. The maximum is then determined and used for inferring the location of the receiver. Some results from both these methods are presented next.

6.3.1 Trilateration

The data modelled both empirical and theoretical can differ from one position to the next by as little as 0.5 of a decibel, particularly as the distance to all access points increases. In the trilateration method, a small difference in the signal strength can cause the distance to the access points to change. As an example, a user located at position (4,5) in the map is examined. According to the empirical data measured, the 802.11b device recorded a maximum signal strength of –45dB received from access point, AP1. Using the One-Slope Model, the signal strength estimated at this position is -49.46dB. Before trilateration is performed the distance to the access points must be calculated. This is done using the signal strengths and the indoor propagation models.

Using the signal strength obtained from the One-Slope Model, the distance is correctly calculated to be 3m. The trilateration method then correctly determines the position of the receiver to be (3,4). However, if using the value of –45dB the distance is calculated to be 3.7286m. Assuming the rest of the signal strengths are calculated accurately from the One Slope Model, to be -53.9272dB and -53.2197dB, the trilateration method determines the position of the receiver to be at the coordinates (2.3064, 3.4451). This example shows that a small discrepancy in the signal strength received and the signal strength given in the model can cause an error in the accuracy of the position.

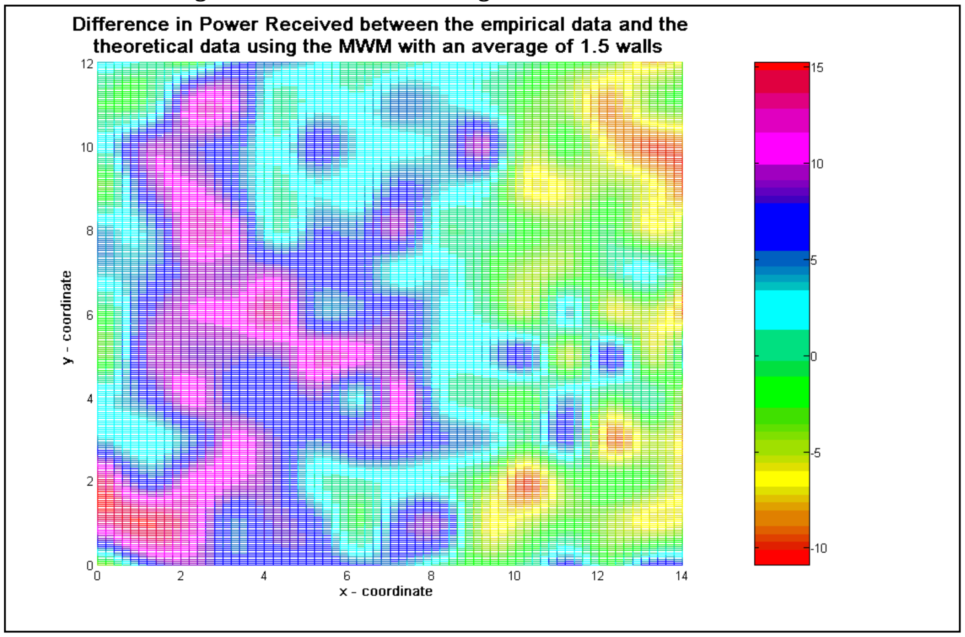
Given the signal strength as calculated by the theoretical propagation model, the trilateration method determines the position of the device accurately every time. Table 6.1 below shows the results for different sets of signal strength. The first three columns show possible signal strengths that could be received at a certain position from the three access points. The last two columns show the X and Y coordinates as determined by the trilateration method. The first set is the signal strengths that would be calculated at position (3,5) on the map using the One-Slope Model. As the Table shows, the trilateration accurately determines the position to be (3,5) using this data. The data is varied by one or two decibels in order to show the deviance that this causes in the determination of the (X, Y) coordinates by the trilateration method.

Table 6.1 – Trilateration method to determine location given certain signal strengths

Signal Strength Received at Receiver			Trilateration Method	
AP1	AP2	AP3	X	Y
-51.7969	-51.7969	-54.728	3	5
-52	-52	-55	2.9437	5
-51	-51	-54	3.0739	5
-50	-51	-55	2.4387	4.8113
-52	-53	-54	3.343	4.7545

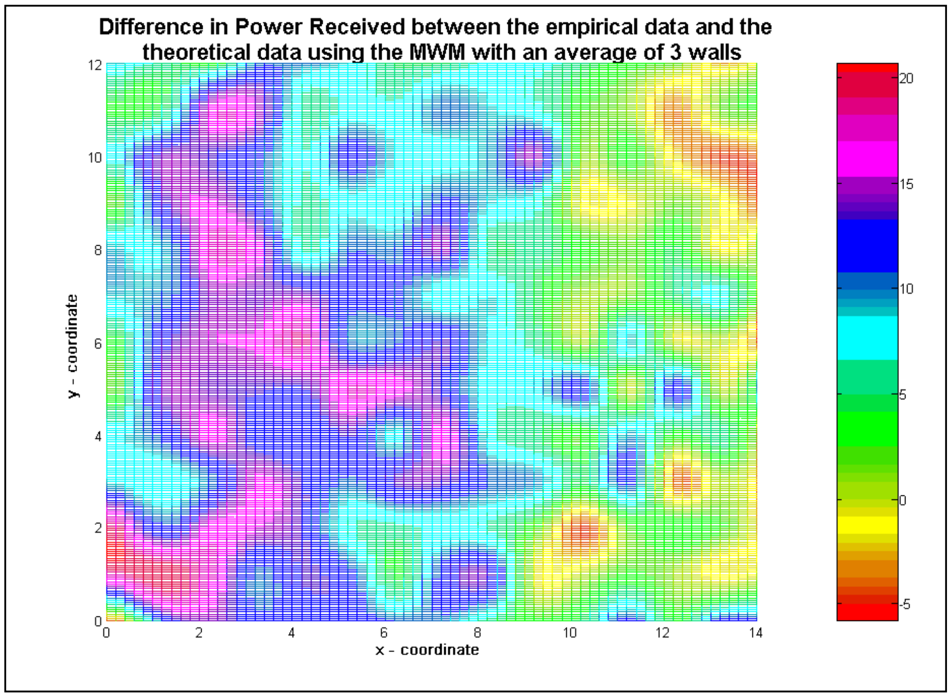
The above example uses the One-Slope Propagation Model. If the Multi-Wall Model is used for the propagation model an average number of walls has to be used throughout the model. Otherwise, the distance will not be calculated accurately. In order to determine the best average number of walls to use throughout the area, several values were tried and the compared to the empirical data to determine which value allows the model best approach the empirical data. This is illustrated in Figure 6.11.

Figure 6.11 – Difference in the power received from the empirical data and the theoretical data using the MWM with an average of 1.5 walls



The above Figure takes an average of 1.5 walls in the area. If this example is compared to Figure 6.9, which determines the number of walls from the distance between the transmitter and receiver, it can be seen that Figure 6.9 performs significantly better. Figure 6.12 will take the same example with an average of 3 walls.

Figure 6.12 – Difference in the power received from the empirical data and the theoretical data using the MWM with an average of 3 walls



The above Figure shows that an average of 3 walls gives a worse approximation that using an average value of 1.5 walls for the area. Therefore, this is the average number of walls that will be used in the MWM, to infer the location of a user.

6.3.2 MMC Algorithm

The Multiple Matrix Correlation algorithm is an alternative method to trilateration for solving the (X, Y) coordinates of an unknown position. Some results from the MMC algorithm with the same signal strength values as used in Table 6.1 are given in Table 6.2.

Table 6.2 – MMC method to determine location given certain signal strengths

Signal Strength Received at Receiver			MMC Method	
AP1	**AP2**	**AP3**	**X**	**Y**
-51.7969	-51.7969	-54.728	3	5
-52	-52	-55	3	5
-51	-51	-54	3	5
-50	-51	-55	2	5
-52	-53	-54	4	5

Table 6.2 shows that given the signal strengths at the receiver correct to within a few decibels of the propagation data, the MMC method will generally determine the correct position for the user. The last two sets are closer in value to the signal strengths received at position (2,5) and (4,5) than to position (3,5), which is why the MMC algorithm returns these values.

Table 6.3 – Comparison Example between the MMC method and the Trilateration Method

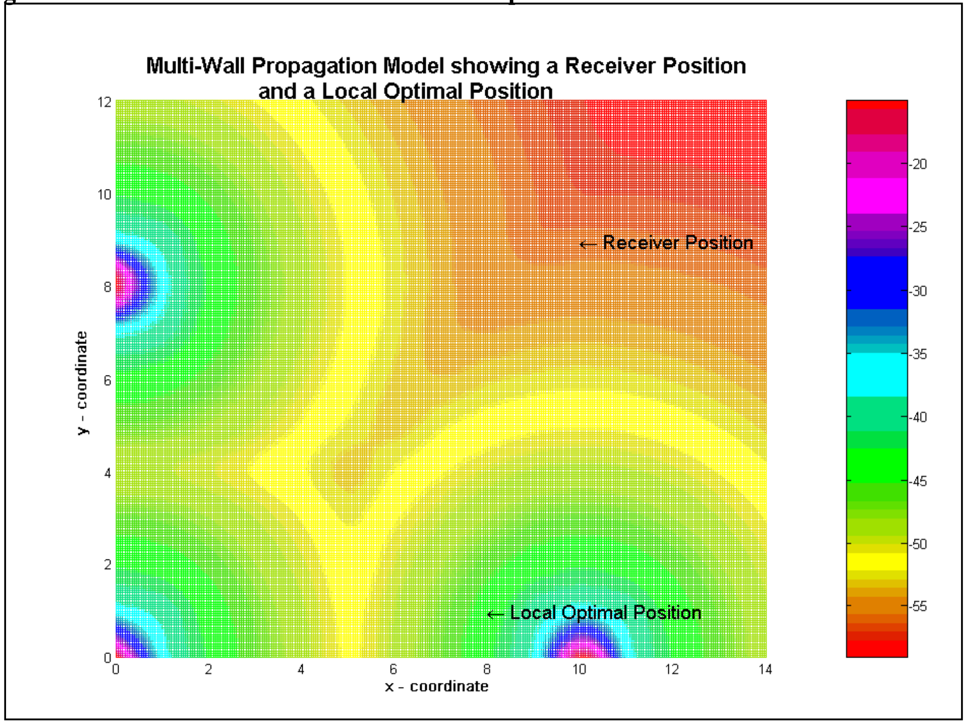
	Signal Strength Received at Receiver			**MMC Method**		**Trilateration Method**	
	AP1	**AP2**	**AP3**	**X**	**Y**	**X**	**Y**
1	-69.2969	-63.5681	-65.7069	12	14	12	14
2	-70	-64	-66	12	14	13.5387	15.18
3	-69	-63	-66	12	14	10.6649	13.9249
4	-64.9933	-55.1708	-62.8199	7	12	7	12
5	-65	-55	-63	7	12	6.7934	12.0675
6	-64	-56	-63	7	12	5.305	10.5123

Table 6.3 gives a possible set of signal strengths received at the receiver and the location that is calculated from these signal strengths using both the MMC method and the trilateration method. In these examples, the MMC method clearly performs better than the trilateration method, as the deviance between the signal strengths is generally less than 0.5 of a decibel. Taking sets 4 and 6 in the above table, it can be seen that the signals strengths between the two closely match and therefore the locations determined by each should be very close. The MMC algorithm determines the location to be (7,12) in both cases. However, the trilateration method correctly determines set 4 to be at position (7,12) but determines set 6 to be at (5.3, 10.5). Using the signal strengths given in set 6, the trilateration method determines the distance to the three access points to be 13.0137, 7.6883 and 12.1851. From these distances, it then calculates the X Y position, with less accuracy than the MMC method.

6.4 Optimal Local Position

The Optimal Local Position is calculated by the OLP algorithm as explained in Chapter 5, section 5.4. Figure 6.13 below shows an example receiver position and the calculated Optimal Local Position on the coverage model of the area. As can be seen from this Figure, the receiver is initially in position, (10,9), with a signal strength of approximately –55dB. The Optimal Local Position is then determined by the OLP algorithm to be (8,1), which has a signal strength according to the model of approximately –40dB.

Figure 6.13 – Receiver Position and Calculated Optimal Local Position



The OLP algorithm accurately calculates the Optimal Local Position for any given receiver position. In the example, in Figure 6.10, the threshold for the Optimal Local Position is set to 40% of the transmitted power. The transmitted power is 15.05dB, therefore the threshold for the Optimal Local Position is –37.625dB. This threshold can be easily changed to other percentages in order to suit the quality of service required by the user. Appendix G gives a result file, showing all possible receiver locations and the Optimal Local Position calculated for each of these positions. A subset of these results are shown in Table 6.4. In this set of test results, the 1SM is the propagation model and the MMC algorithm determines the receiver position.

Table 6.4 – Example of Optimal Local Positions calculated from Receiver Positions

Receiver Location		Optimal Local Position	
X	**Y**	**X**	**Y**
12	0	10	0
12	1	10	0
12	2	10	0
12	3	10	1
12	4	9	1
12	5	9	2
12	6	8	2
12	7	8	2
12	8	8	2
12	9	8	2
12	10	8	2
12	11	8	2
12	12	8	2
12	13	2	10
12	14	2	10

6.5 Shortest Path

The final step is to inform the user how to move from the receiver location to the local optimal path. This is done by using the A*, [HNR68], shortest path algorithm, which basically finds the shortest path between two locations. The shortest path algorithm is dependent on obstructions in the given area.

Given the receiver position at (10,9), as in figure 6.13, the local optimal position is calculated to be at (8,1). The map used is given in Figure 5.15, as presented to the A* algorithm and the resultant path as calculated by the A* algorithm is shown in Figure 6.14. The position of the walls is shown on the model. The red dashed line shows the path that the user should follow.

Figure 6.14 – Moving from the Receiver Position to the Optimal Local Position

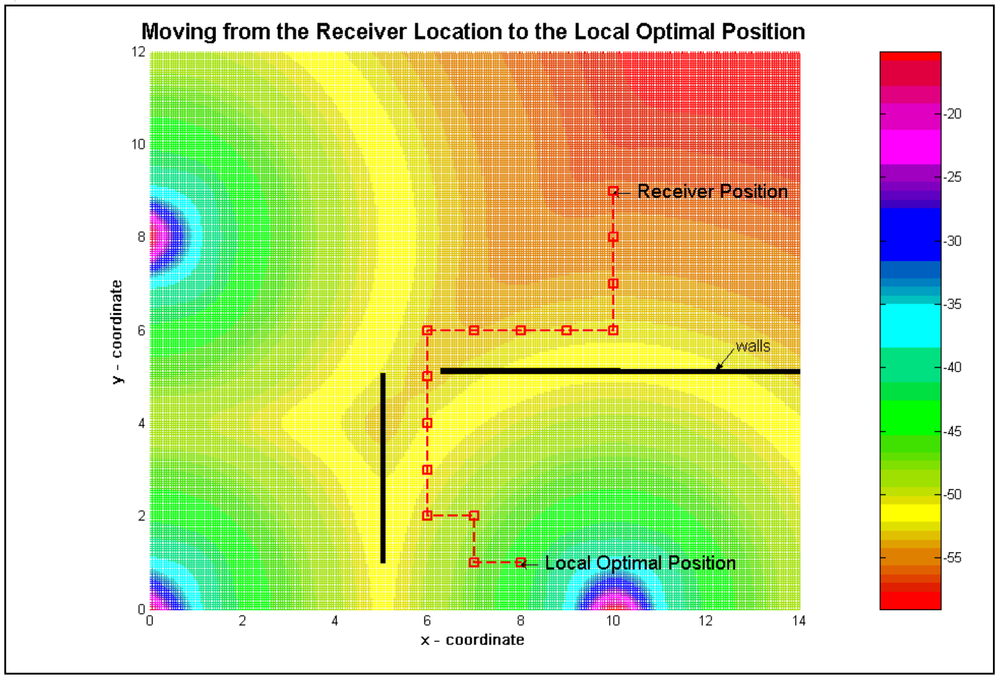
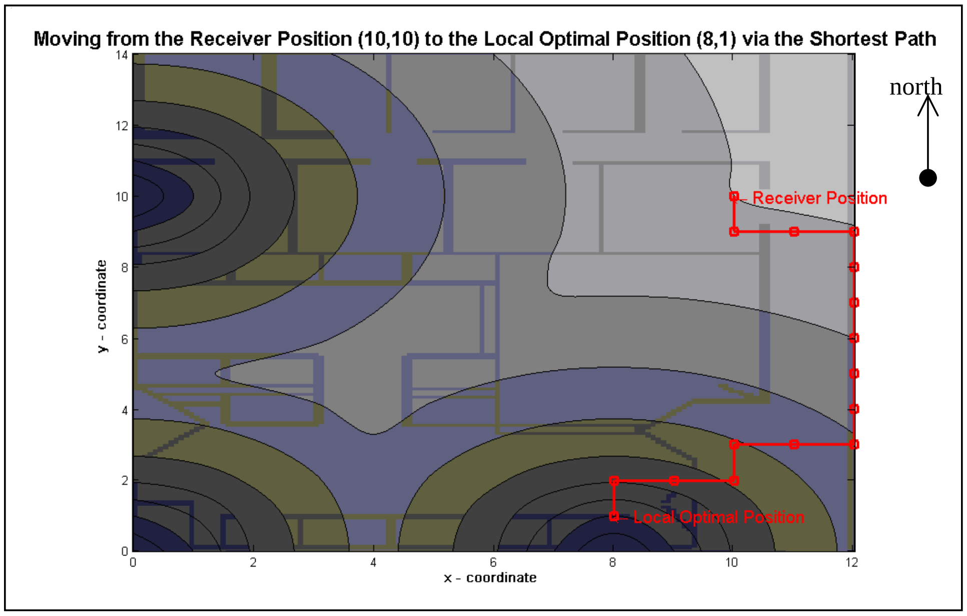


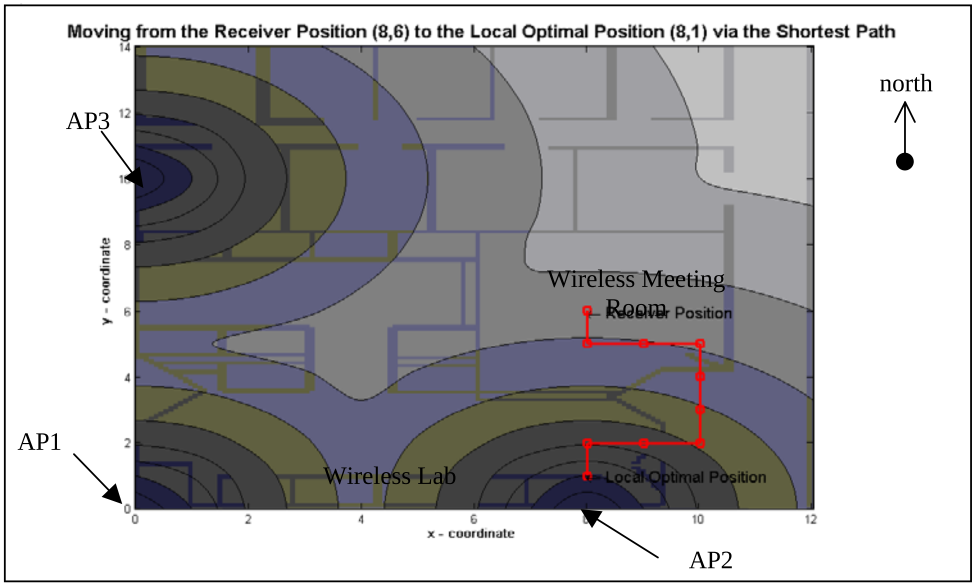
Figure 6.15 demonstrates the area used in this thesis, with the A* algorithm finding the shortest path between the receiver position and the local optimal position. The receiver position is at (10,10). Therefore the Optimal Local Position is calculated to be (8,1). The red line once again shows the optimal path. The coverage of the area is overlaid onto the map, with the more darkly shaded area showing the better signal strengths.

Figure 6.15 - Moving from (10,10) to (8,1) on the map of the Foundation Building



Another example of the A* finding the shortest path in the Foundation Building is given in Figure 6.16. The user is originally located at coordinates (8,6), which is in the wireless access group's meeting room. The Optimal Local Position is then calculated to be (8,1), i.e. in the wireless access group's laboratory, by the OLP algorithm. The A* algorithm identifies the shortest path between the two points using the given map, see Table 6.5 for the A* map. As can be seen in Figure 6.16, the user is directed out of the meeting room through the door into the corridor and from there into the wireless lab through the double doors and then to the coordinate (8,1).

Figure 6.16 – Moving from the Receiver Position to the Optimal Local Position via the Shortest Path



The A * algorithm correctly determines the shortest path between any two given points on the map. One drawback of this system is that the map being used by the A* algorithm currently has to be hard coded in the C++ code and then compiled and linked back into the Matlab directory. The matrix that approximately represents the above map is given in Table 6.5.

Table 6.5 – Matrix representing map in Figure 6.12 above

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**
0	1	1	1	1	1	1	1	1	1	9	1	1	1
1	1	1	1	1	1	1	1	1	1	9	1	1	1
2	1	1	1	1	1	1	1	1	1	1	1	1	1
3	1	1	1	1	1	1	1	1	1	9	1	1	1
4	1	9	9	9	1	9	9	9	9	9	1	9	1
5	1	1	1	1	1	1	9	1	1	1	1	9	1
6	1	1	1	1	1	1	9	1	1	1	1	9	1
7	1	1	1	1	1	1	9	1	1	1	1	9	1
8	9	9	9	9	9	9	9	9	9	9	9	9	1
9	1	1	1	9	1	1	9	9	1	1	1	1	1
10	1	9	1	9	9	1	9	9	1	9	9	9	1
11	1	1	1	1	1	1	1	1	1	1	1	1	1
12	9	1	9	9	1	9	9	1	9	1	9	9	1
13	1	1	9	1	1	9	1	1	9	1	1	9	1
14	1	1	9	1	1	9	1	1	9	1	1	9	1

7 Conclusions

7.1 The LOUP System

The Local Optimal User Position (LUOP) system proposed in this thesis locates a user within an area and directs the user to a new position in order to receive improved service. The results, shown in Chapter six, demonstrate the performance of the system in a given area, i.e. the first floor of the foundation building in the University of Limerick.

The value of such a system in indoor wireless communications is vast. Wireless technology users are constantly looking for improvements in the service that is obtainable. Indeed, this is the goal of many research projects. This thesis has combined position location and propagation modelling to achieve such an end in a simple but efficient manner.

7.2 Propagation Modelling

The modelling stage performed in this thesis is carried out in two different manners. The empirical modelling uses two different types of filters to choose the signal strength that represents each position in the map, the average and the maximum. The maximum is a more reasonable value to choose, as this is not affected by dips in the signal strength. If enough measurements are taken both for the offline empirical phase and at the receiver for the actual determination of the location position, the maximum signal strength at the receiver should be close enough to the modelled value to accurately determine the position of the receiver.

The indoor propagation models were compared with the empirical data to determine which model best approaches the given indoor radio environment. Both the One Slope Model and the Multi Wall Model give reasonable approximations of the empirical data. The Freespace model and the Linear Attenuation Model do not accurately represent the given indoor environment.

The theoretical model used for location positioning is the One Slope Model. In order to use the MWM, an average number of walls has to be used, in order to determine the distance between the transmitter and receiver from the signal strength. The best value for the average number of walls is determined to be 1.5 for the given indoor environment. Using an average number of walls for the MWM makes the model less accurate when compared to the empirical data. This is why the One Slope Model is used in this thesis to infer a location.

The empirical data measured for this thesis is not accurate enough to be used for inferring the location of a user. The inaccuracies between the empirical data and the theoretical data are caused by a number of factors including interference due to noise, inaccuracies in the measurement technique and multipath due to the movement of people. Therefore, while the empirical method is superior in terms of accuracy for a particular building, assuming an accurate measurement technique, the theoretical method makes deployment quicker and easier. Several suggestions to improve this data and to obtain usable empirical data are made in the further research section below. As the aim of this thesis is to demonstrate the local optimisation system working correctly, the use of theoretical data is adequate.

In summary this thesis demonstrates how to create a propagation model of a given area. For a similar type of building as this thesis, the One Slope Model should be used for theoretical modelling. For greater accuracy an empirical propagation model should be created, using the maximum signal strength received at each location.

7.3 Location Positioning

The location of a user can be determined by trilateration or the MMC method in this thesis. The MMC method makes use of the MMC algorithm, which was designed and developed as part of this thesis. Given the theoretical propagation model and the signal strengths received by the user from each of the access points to be the same as the signal strengths in the model, both methods correctly determine the user position every time. By allowing a deviation of 1 or 2 dB's in the signal strengths received at the receiver, it is shown that the MMC algorithm generally calculates the position of the receiver more accurately than the trilateration method.

The trilateration method is based on using an indoor propagation model to determine the distances to the transmitters. Therefore, it can only ever be as accurate as the propagation model. The MMC algorithm does not rely on propagation models for positioning. Indeed, it can work on a radio map that is created from either a theoretical propagation model or from empirical data.

7.4 Optimal Position

The Optimal Local Position is the closest position to the user that gives an improved class of service to the user. The amount of improvement is dependent on the threshold used. In this thesis, a threshold of 40% is used, that is the power received at the Optimal Local Position has to be within 40% of the power transmitted. This threshold could be set by the user depending on the class of service required. The Optimal Local Position is calculated using the OLP algorithm, as explained in Chapter 5. This algorithm was designed and developed specifically for use within this thesis. The OLP algorithm accurately calculates the Optimal Local Position from any given receiver position.

7.5 Pathfinding

The final step in the system is to direct the user from the receiver position to the Optimal Local Position. The path that the user should follow is determined by the A* shortest path algorithm. The A* algorithm correctly finds the shortest path between the receiver position and the Optimal Local Position if one exists and then graphs the path onto the map of the area using Matlab.

7.6 Overall Conclusions and Further Research

In summary, this thesis demonstrates how to create a propagation model of an indoor environment, how to use this model along with the signal strengths received from three or more access points to accurately determine the location of a receiver within the area and how to direct the user to a nearby location in order to obtain a better class of service.

Although the data obtained for the empirical model is too inconsistent for determining the location position in the given area, the theoretical models used, namely the 1SM and the MWM, perform very well in modelling the given area and indeed an accurate position can be determined when using these models. Obtaining a real dataset from an outside source and further validating the work in this thesis would be an interesting area of further research.

The OLP algorithm to find the Optimal Local Position in the area, performs as expected and returns the closest position with a signal strength within the threshold value every time. The outcome from the A* algorithm in returning the shortest path between the two points is also very good. A drawback of the system is that the OLP algorithm works independently to the A* algorithm. The shortest path is not found until after the Optimal Local Position is determined. In order to reach this optimal position, several different rooms might have to be passed through. There could be an optimal position, which is easier to reach but is further away in terms of Euclidean distance. One important area of further research would be to combine these two algorithms into one algorithm in order to obtain a better result. In this situation the A* algorithm takes the improved signal strength as its heuristic instead of the Euclidean distance.

Another important area of further research for this thesis would be to determine the location position of the user with more accuracy by using different technology. For example, the use of UWB instead of WLAN technology in the future would improve the accuracy of the system.

8 References

[3GP99] 3GPP TSG-RAN Working Group, "Report on Location Services, 3G TR 25.923 v1.0.0", May 1999

[ACK87] D. Ackley, "A Connectionist Machine for Genetic Hill-Climbing", Kluwer Academic Publishers, Boston, 1987.

[ARY95] Jorgen Bach Andersen, Theodore Rappaport and Susumu Yoshida, "Propagation Measurements and Models for Wireless Communications Channels", IEEE Communications Magazine, 1995.

[AZU93] R. Azuma, “Tracking Requirements for Augmented Reality”, Communications of the ACM, Vol. 36, July 1993.

[BEL58] Richard Bellman, "On a routing problem", Quarterly of Applied Mathematics, 16, 1958.

[BDG+00] J. Benedicto, S.E. Dinwiddy, G. Gatti, R. Lucas, M. Lugert, "Galileo: Satellite System Design and Technology Developments" European Space Agency, November 2000.

[BDJ99] J. Börjesson, F. Darin and J. Johansson, "GLONASS: experiences from the first global campaign", RVK99, Karlskrona, Sweden, June 14-17, 1999.

[BMV03] J.Benedicto, P.Michel, J.Venutra-Traveset "EGNOS The first European implemenation of GNSS – Project Status Overview", European Space Agency, 2003.

[BP00] Paramvir Bahl, Venkata Padmanabhan, "RADAR: An In-Building RF-Based User Location and Tracking System", Microsoft Research, In Proc. of IEEE INFORCOM 2000.

[BRE03] Bob Brewin, "Pentagon tweaked GPS accuracy to within three meters during Iraq war", Computer World, June 24 2003.

[BRI01] Roger Bridgeman, "Guiglielmo Marconi: radio star", Institute of Physics, Physics World, December 2001.

[BLU04] Bluetooth Special Interest Group, "Bluetooth Core Specification v1.2", Bluetooth Special Interest Group, 2004.

[CCK01] Paul Castro, Patrick Chiu, Ted Kremenek, Richard Muntz, " A Probabilistic Room Location Service for Wireless Networked Environments", UCLA, 2001

[CG93] T.W. Christ, P.A. Goodwin, “A Prison Guard Duress Alarm Location System”, Proceedings IEEE International Carnahan Conference of on Security Technology, 1993.

[CKP+99] Jock Christie, Ping-Ya Ko, Boris S. Pervan, Per K. Enge, David Powell, Bradford W. Parkinson, "Analytical and Experimental Observations of Ionospheric and Tropospheric Decorrelation Effects for Differential Satellite Navigation during Precision Approach" Dept of Aeronautics and Astronautics, Stanford University, 1999.

[CLR90] Thomas Cormen, Charles Leiserson, Ronald Rivest, "An Introduction to Algorithms", The MIT Press, 1990

[COL01] Kent Colling, "Ultra-Wideband Communications and Sensor Networks", Innovative Wireless Technologies, 2001.

[COS91] European Cooperation in the Field of Scientific and Technical Research EURO-COST 231, "Urban Transmission Loss Models for Mobile Radio in the 900 and 1800MHz Bands", Tech. Report, The Hague, 1991.

[DAV97] John S. Davies, "Indoor Propagation at 2.4GHz – Measurements in Cory Hall at UC Berkeley", Wireless Communication CD-ROM Edition, 1997.

[DEA98] de Aquino, Marcio H., Regional Approach to Wide Area DGPS, University of Nottingham, Institute of Surveying & Space Geodesy, November 1998.

[DF00] P. Deng, Pz. Fan, "An AOA Assisted TAO Positioning System", Southwest Jiao Tong University, 2000.

[DG86] P. Daly and G. E. Gerry, "Recent Developments with the Soviet Union's VHF Satellite Navigation System", Space Communication and Broadcasting, 1986.

[DIJ59] E. W. Dijkstra, "A Note of two problems in connexion with graphs" Numerische Mathematik, 1959.

[DOT01] US Dept. of Transport, Federal Aviation Administration, "GPS Modernization – The Future of GPS", Published by the Federal Aviation Administration, FAA, 2001

[DOT02] US Dept. of Transport, Federal Aviation Administration, "Local Area Augmentation System Specification", Published by the Federal Aviation Administration, FAA, 2002.

[DP90] Robert J. Danchik and L. Lee Pryor, "The Navy Navigation Satellite System (TRANSIT)", APL Technical Digest, Volume 11, 1990.

[EKA02] Ekahau, "Ekahau Positioning Engine 2.1 User Guide", 2002

[ETS99] ETSI GSM LCS TS 101 528 V8.1.0, "Broadcast Network Assistance for Enhanced Observed Time Difference and Global Positioning Systems Positioning Methods", Release 1999.

[FAA01] Statement from FAA, "WAAS and its Relation to Enabled Hand-Held GPS Receivers", March 2001

[FAA02] US Dept. of Transport, Federal Aviation Administration, "Evaluation of Loran C", Published by the Federal Aviation Administration, ICNS Conference Briefing, May 2002.

[FB01] Alois Ferscha, Wolfgang Beer, Wolfgang Narzt, Location Awareness in Community Wireless LAN’s, 2001.

[FF62] Lester R. Ford and D. R. Fulkerson, "Flows in Networks", Princeton University Press, 1962.

[FG02] Robert Fontana, Steven Gunderson, "Ultra-Wideband Precision Asset Location System", IEEE Conference of Ultra-Wideband Technologies, May 2002.

[FKL00] Sven Fischer, Havish Koorapaty, Erik Larsson and Ari Kangas, "System Performance Evaluation of Mobile Positioning Methods", 2000.

[FLO62] R. Floyd, "Algorithm 97 (SHORTEST PATH)", Communications of the ACM, 5, 1962.

[FON01] Robert Fontana, "Experimental Results from an Ultra-Wideband Precision Geolocation System", 2001.

[FWN01] Alois Ferscha, Wolfgang Beer, Wolfgang Narzt, "Location Awareness in Community Wireless LAN's", University of Linz, 2001.

[GG03] Fredrik Gustafsson, Fredrik Gunnarsson, "Positioning Using Time-Difference of Arrival Measurements", Department of Electrical Engineering, Link¨oping University, 2003

[GOR99] Swedberg Goran, "Ericsson’s Mobile Location Solution", Ericsson Review no. 4, 1999.

[GSC04] GPS Support Center, "2003 2-d Error", GPS Support Center, May 2004.

[GR93] Andrew V. Goldberg and Tomasz Radzik "A Heuristic Improvement of the Bellman-Ford algorithm", Amlets, Applied Maths Letters, 7, 1993

[HAI99] Brig J. Haigh, "The Services Textbook of Radio, Volume 7, Radiolocation Techniques", Wireless World, 1999

[HAR02] Blake Harris, "Amulet: Approximate Mobile User Location Tracking System", University of Rochester.

[HAT80] Hata M., "Empirical Formula for Propagation Loss in Land Mobile Radio Services", IEEE Trans. Vehicle Technology, Aug 1980.

[HAT02] Dale N. Hatfield, "A Report on Technical and Operational Issues Impacting the Provision of Wireless Enhanced 911 Services", Federal Communications Commission, 2001.

[HEK02] Aki Hekkala, "Radio Direction-Finding Antennas and Systems", Telecommunication Laboratory, University of Oulu, Finland, 2002

[HJ73] John Hopcroft and Robert E Tarjan, "Efficient algorithms for graph manipulation", Communications of the ACM, 16, 1973.

[HKS+97] T.D. Hodes, R.H. Katz, E.S. Schreiber and L. Rowe, “Composable Ad Hoc Mobile Services for Universal Interaction”, MobiCom ’97 Proceedings, Sept. 1997.

[HNR68] P. Hart, N. Nilsson and B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", IEEE Trans. Sys. Sci. Cybernetics, vol. no.2, 1968.

[HP102] Hewlett Packard Invent, "Wi-Fi™ and Bluetooth™ –Interference Issues" January 2002.

[IDA01] Institute of Defence Analysis, "Wide Area Augmentation System – Independent Review Board Report", Published by the Federal Aviation Administration, FAA, 2001

[IEE97] IEEE 802.11 Standard for Wireless LAN, Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, Jan 1997.

[ITU01] International Telecommunications Union – Radio Sector, "Radio Regulations", Edition of 2001.

[JOC+99] R.I. Jock, et al, "Analytical and Experimental Observations of Ionospheric and Tropospheric Decorrelation Effects for Differential Satellite Navigation during Precision Approach, Dept of Aeronautics and Astronautics", Stanford University, 1999.

[JOH74] Donald S. Johnson, "Efficient algorithms for shortest paths in sparse networks", Journal of Computer and System Sciences, 1974.

[JON04] Justin Heyes-Jones, A* Algorithm Tutorial, http://www.geocities.com/jheyesjones/astar.com

[KAP96] Elliot D. Kaplan, "Understanding GPS Principles and Applications", Artech House Publishers, 1996.

[KAR95] P. Karlsson, "Indoor Radio Propagation for Personal Communications Services", PhD Thesis, University of Lund, 1995.

[KEW00] C. Kreye, B. Eissfeller, J.O. Winkel, "Improvements of GNSS Receiver Performance Using Deeply Coupled INS Measurements". ION GPS 2000.

[KLH00] Hiroaki Koshima, Locus Corp. & Joseph Hoshen, "Personal Locator services emerge", IEEE Comms, 2000.

[KM90] J. M. Keenan and A. J. Motley, "Radio Coverage in buildings", British Telecom Technology Journal, 1990

[KMR01] Arun, Kishan, Michael Michael, Sahil Rihan, Rahul Biswas, "Halibut: An Infrastructure for Wireless LAN Location-Based Services", Stanford University, June 2001.

[KOR85] Richard E. Korf, Iterative-Deepening A*: An Optimal Admissible Tree Search. In Proceedings of the International Joint Conference on Artificial Intelligence, Los Altos, California, 1034-1036., 1985.

[KOR98] Richard E. Korf, "Artificial Intelligence Search Algorithms", UCLA, 1998.

[KRU56] J.B. Kruskal, On the Shortest Spanning Subtree of a Graph and the travelling salesman problem", Proc. Amer. Maths. Soc. 7, 1956.

[LDJ02] Jimmy LaMance, Javier DeSalas, Jani Jarvinen, "Assisted-GPS A Low Infrastructure Approach", GPS World, March 1, 2002.

[LEE61] C. Y. Lee, "An algorithm for path connection and its applications", IRE Transactions on Electronic Computers, 1961.

[LIP03] Vikki Lipset, "Sensing Location with UWB", UltraWideband Planet, Issue Dec 2003.

[LM04] Nicola Lenihan, Sean McGrath, " Local Optimal Position for Indoor Wireless Devices", May 2004, Accepted for Proceeding of WPMC Conference 2004.

[LOW76] B.T. Lowerre, "The HARPY Speech Recognition System", PhD. Thesis, Carnegie Mellon University, April 1976.

[LUD01] Ludden Brendan, "Location Technology", Vodafone Review, 2001.

[MK98] A. J. Motley and J. P. Keenan, "Personal Communication Radio Coverage in Buildings at 900MHz and 1700MHz", 1998

[MOO59] Edward F. Moore, "The Shortest Path through a Maze", Proc. of the Int. Symp. of the Theory of Switching, Harvard University Press, 1959.

[NEL95] Northwest European LORAN-C System, "NELS Specification of the transmitted Loran-C signal", Document S-006-2, October 1995.

[NIL80] N. J. Nilsson, "Principles of Artificial Intelligence", Tioga Publishing Company, 1980

[NMN00] Jaroslav Nesetril, Eva Milková, Helena Nesetrilová, "Otakar Boruvka on minimum spanning tree problem (translation of the both 1926 papers, comments, history)", 2000.

[ORI01] Orinoco, Orinoco AP-1000 User Guide, 2001

[PAL84] S. Pallotino, "Shortest Path Methods: Complexity, Interrelations and new Propositions", Networks, 14, 1984.

[PAP74] U. Pape, "Implementation and Efficiency for Moore Algorithms for the shortest root node", Math. Prog., 7, 1974.

[PRI57] R.C. Prim, "The shortest connecting network and some generalisations", Bell. Syst. Technical Journal 36, 1957.

[PS99] Greg Perkins and Diane Souvaine, "Efficient Radio Wave Front Propagation for Wireless Radio Signal Prediction", 1999.

[RAP96] Theodore S. Rappaport, Wireless Communications Principles and Practice, IEEE Press/Prentice Hall PTR, New Jersey, 1996.

[RBX97] Theodore S Rappaport, Keith Blankenship and Hao Xu, "Propagation and Radio System Design Issues in Mobile Radio Systems for the GloMo Project", DARPA GloMo Project, 1997.

[RR00] Christ Roger, Lavigne Robert, "Radio Frequency Based Personnel Location Systems", IEEE Trans. VT Technology, Aug 2000.

[RUB78] S. Rubin, "The ARGOS Image Understanding System", PhD. Thesis, Carnegie Mellon University, April 1978

[SAR02] Tomi Sarvanko, "Positioning Standards, E911, E112 and UMTS", Telecommunication Laboratory, University of Oulu, Finland, 2002.

[SC00] Samer S. Saab, Dany S. Charbachy, "Differential GPS Performance: Ground Versus Geostationary Satellite", IEEE, 2000.

[SCH00] B.M. Scherzinger, "Precise Robust Positioning with Inertial/GPS RTK", ION GPS 2000.

[SMR03] Manuel G. Soares, Benedita Malheiro and Francisco J. Restivo, "Implementation of a Campus Wide DGPS Data Server", Dept. of Electrical Engineering, ISEP, Portugal, 2003.

[SSS01] Jason Small, Asim Smailagic, Daniel Siewiorek, "Determining User Location for Context Aware Computing Through the Use of a Wireless LAN Infrastructure", Carnegie Mellon University, 2001

[STEIN] John S. Stein, "Indoor Radio WLAN Performance Part II: Range Performance in a Dense Office Environment", Intersil Corporation.

[SYR01] Jari Syrjarinne, "Studies of Modern Techniques for Personal Positioning", Thesis at Tampere University of Technology, 2001.

[TBL93] C. Tornevik, J. E. Berg, F. Lotse, "900MHZ propagation measurements and path loss models for different indoor environments", Proc. IEEE VTC93, 1993.

[TET01] Tetra Precision Inc., "http://www.tetraprec.com/trilatef.htm", Last Modified December 2001.

[TRA03] Dr. Javier Ventura-Traveset "European Geostationary Navigational Overlay Service", EGNOS Project Office, European Space Agency, 2003.

[TRI01] Trimble Navigation Ltd., "Wide Area Augmentation System, Frequently Asked Questions and Answers", 2001.

[USC90] U.S. Coast Guard, "LORAN-C User's Handbook, COMDTPUB P16562.6" United States Coast Guard Navigation Centre, 1990.

[USC03] U.S. Coast Guard Navigation Center, "Differential Global Positioning System – General Information" United States Coast Guard Navigation Centre, 2003.

[USN04] United States Naval Observatory (USNO), "GPS System Description", March 2004.

[WAL02] Bernhard H. Walke, "Mobile Radio Networks – Networking, Protocols and Traffic Performance", Second Edition, John Wiley and Sons Ltd., 2002.

[WAR62] Stephen Warshal, "A theorem on Boolean matrices", Journal of the ACM, vol. 9, 1962.

[WHE02] WhereNet, "WhereLan, Location Sensor Locating Access Point", 2002

[WHF92] R. Want, A. Hopper, V. Falcao, J.Gibbons, "The Active Badge Location System," ACM Transactions on Information Systems, Vol. 10, January 1992.

[WL98] J. Werb, C. Lanzl, "Designing a Positioning System for Finding Things and People Indoors," IEEE Spectrum, September 1998.

Appendix A

A 802.11 WLAN Networks

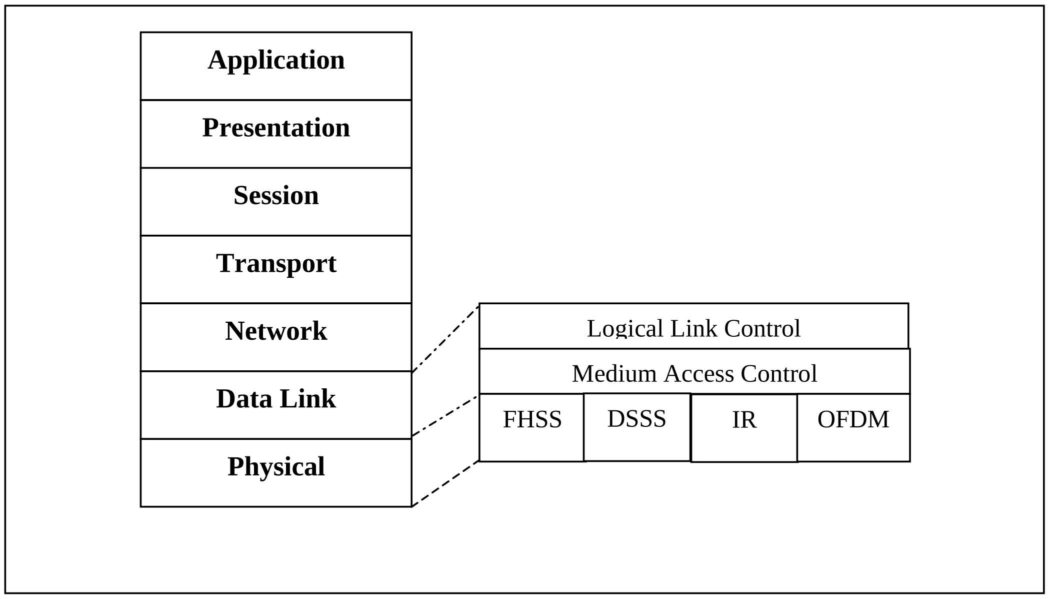
A.1 Introduction

Wireless Local Area Network (WLAN) technologies uses radio frequency technology to receive and transmit data over the air. It is an alternative to prevalent wired technology such as Ethernet, which offers data connectivity with mobility. Typical WLAN configurations consist of a transmitter/receiver device called an access point, which allows a small group of users to connect to each other or to a backbone system connected to the access point. End users access the WLAN through adapters or network interface cards (NIC). Examples of WLAN technologies include, IEEE 802.11, HomeRF and HyperLan/2. This thesis is only concerned with IEEE 802.11, but the principles contained within could be applied to any WLAN technology.

802.11 Wireless Local Area Networks is a standard that was first adopted by IEEE in 1997 to define the media access control (MAC) and the physical (PHY) layers for a LAN with wireless connectivity. It is officially titled the IEEE Standard for Wireless LAN Medium Access Control (MAC) and Physical Layer Specifications. It addresses networking issues associated with devices that are connected over the air. These issues or characteristics include mobility, range limitations, unreliable media, dynamic topology, interference from outside sources and lack of ability for every device to hear every other device within the LAN.

The standard is required to appear to higher layers of the topology as normal LAN, i.e. from the logical link control (LLC) layer up it is not required to distinguish between a wired or wireless network, the functionality is the same. In order to achieve this, some functionality has been moved from the LLC down to the MAC layer. See Figure A.1 for the reference model.

Figure A.1 Reference Model used by 802.11 Architecture



A.2 802.11 Alphabet Soup

In June 1997, the Institute of Electrical Engineers (IEEE) published the initial standard for WLAN, IEEE 802.11. This standard specified a 2.4GHz operating frequency with data rates of 1 and 2 Mbps. The IEEE 802.11 Working Group (WG) has made several revisions to these standards through various task groups, which has led to the 802.11 alphabet soup. A particular letter corresponds to each standard such as 802.11a, 802.11b, 802.11f and so on. The standards that are relevant to WLAN installers are: 802.11a, 802.11b and 802.11g and Table 3.1 compares the three standards. The other standards have improved and added different features, e.g. 802.11e is working on improving the QoS within WLAN standard. Altogether, there are 14 different task groups and the progress that each these groups can be seen at [9]. The equipment used in this thesis is 802.11b compliant.

Table A.1 Comparison between different 802.11 Standards

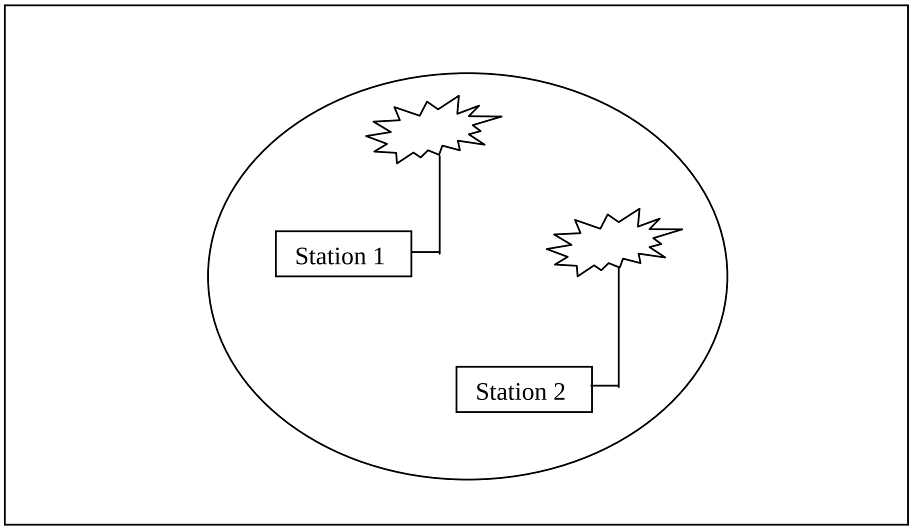
Standard	Completed	Data Rata	Frequency	Modulation Scheme
802.11a	Sept 2001	54Mbps	5 GHz	OFDM
802.11b	Sept 1999	11MBps	2.4GHz	DSSS/CCK
802.11g	July 2003	54Mbps	2.4GHz	DSSS/PBCC

A.3 802.11 Topology

The standard supports two topologies for wireless device interaction. These are independent basic service set (IBSS) and infrastructure service set configurations. A set of wireless devices is defined by the standard as a basic service set (BSS). These wireless devices or stations are any devices that contain the functionality of the 802.11 protocols, that being the MAC, the PHY and a connection to the wireless media. This functionality is usually contained in a card called a network interface card (NIC), which can be plugged into the device.

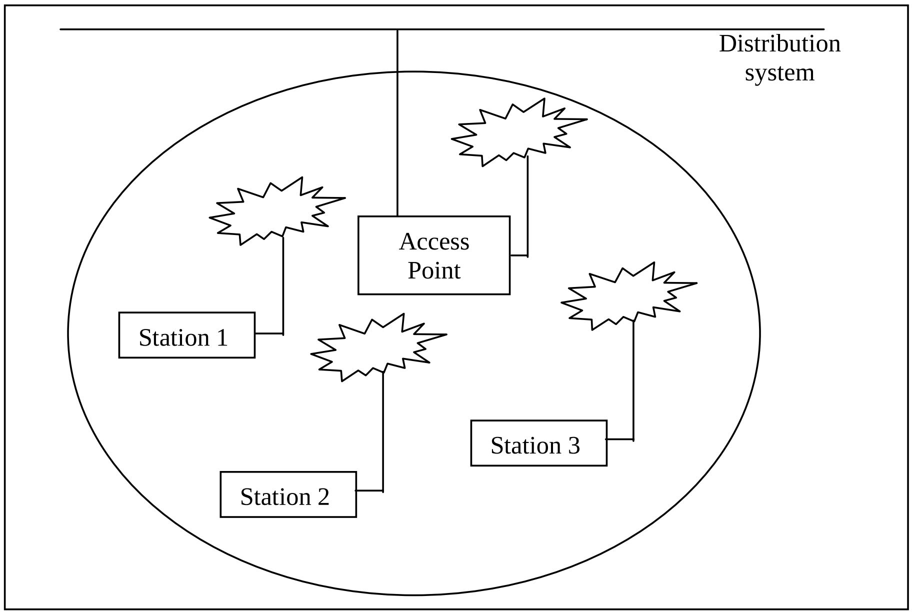
The independent basic service set (IBSS) networks are made up of wireless devices communicating in a peer-to-peer mode without the use of any other communicating station. This form of network topology is also sometimes referred to as an Ad-hoc network. In an IBSS, the mobile stations communicate directly with one another and there is no relay function. Therefore, all stations that need to communicate must be within range of each other. An IBSS is shown in Figure A.2, with station 1 and station 2 communicating.

Figure A.2 – Independent Basic Service Set (IBSS)



An infrastructure basic service set is a BSS with a component called an access point (AP). This is normally the topology that is used everyday in offices and buildings. The access point provides the relay function to the BSS. All the stations now communicate through the AP. The local relay function provided by the AP increases the range of the IBSS. The access point can also be connected to a distribution system (DS). Figure A.3 shows an infrastructure basic service set with the access point connected to a distribution system.

Figure A.3 – Infrastructure Basic Service Set

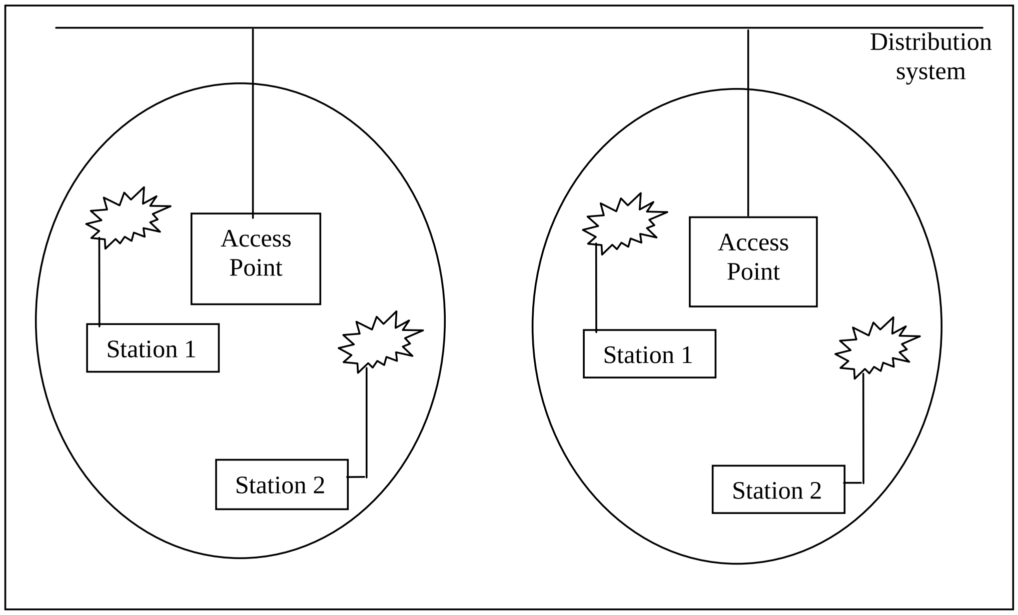


The distribution system (DS) is the means by which the access point communicates with the rest of the world. It can communicate with another access point to exchange frames for stations in other infrastructure basic service sets, forward frames to follow mobile devices that move from one BSS to another and exchange frames with a wired network. There are no restrictions from the IEEE 802.11 standard on how to implement the DS, only on what services it must provide. It can be a wired network, such as Ethernet or a special purpose box that interconnects the access points and provides the required distribution services.

An Extended service set (ESS) is a set of infrastructure BSS', where the access points communicate amongst themselves via the distribution system, to enable movement of stations between BSS'. The distribution system determines if traffic should be relayed back to a destination in the same BSS, forwarded on the distribution system to another BSS or sent into the wired network to another destination. Frames received by the access point from the distribution system are transmitted to the destination station in the BSS over the air. Figure A.4 gives an example of an ESS.

The ESS is seen as a single MAC-layer network from all stations outside it. This level of transparency hides the mobility of the network and allows existing protocols to operate and interact with the wireless LAN architecture.

Figure A.4 – Extended Service Set (ESS)



A.4 Physical Layer

Most WLAN devices operate in the license free ISM, Industry, Science and Medical, frequency band at 2.4GHz. 802.11a and HiperLan/2 are examples that operate in the 5GHz band, which enables faster transfer rates but decreases the coverage area. The transmission bandwidth ranges from 2Mbits/s to approx. 50Mbits/s, e.g. 802.11b operates at the 11Mbits/s range. The IEEE 802.11 standard defines transmission over three different physical layers, PHY: infrared, frequency hopping spread spectrum, FHSS, and direct sequence spread spectrum, DSSS. As an exception to this, the

802.11a and HiperLan/2 uses a physical layer called Orthogonal Frequency Division Multiplexing, which operates at the 5GHz band.

A.4.1 Infrared

The Infrared PHY layer is based on the 850 – 950 nm range, which is just below visible light in the electromagnetic spectrum. The commercial uptake of this layer is not expected to be widespread due to the fact that it requires free sight between the communicating nodes. The transmission range is about 10 meters and it is therefore limited to a small area within a room or using diffuse transmission around a portion of a room.

A.4.2 Frequency Hopping Spread Spectrum (FHSS)

Frequency hopping transmission method transmits and receives on one frequency for a short time and then jumps to another frequency. IEEE 802.11 does a 1 MHz jump once every tenth of a second. The hopping frequency is based on a psuedo-random hopping algorithm and must be know to both the transmitter and receiver. The main intent of the hopping pattern is to avoid interference by not spending very much time on any particular frequency. If interference is present on any of the channels in the hopping pattern, the received signal will have a small amount of noise interference from time to time that will be minimised by the small amount of time transmitting on that frequency.

Within the ISM band, there are 78 frequencies available for IEEE 802.11 and since each access point has its own unique hopping sequence, it creates its own LAN segment. This means that in one coverage area, there could be up to 78 access points having ongoing transmissions with numerous devices. This increases the number of users that can be supported in one coverage area. One drawback of FHSS is the maximum bandwidth of 1MHz leads to a transmission data rate of only either 1 or 2 Mbits/s.

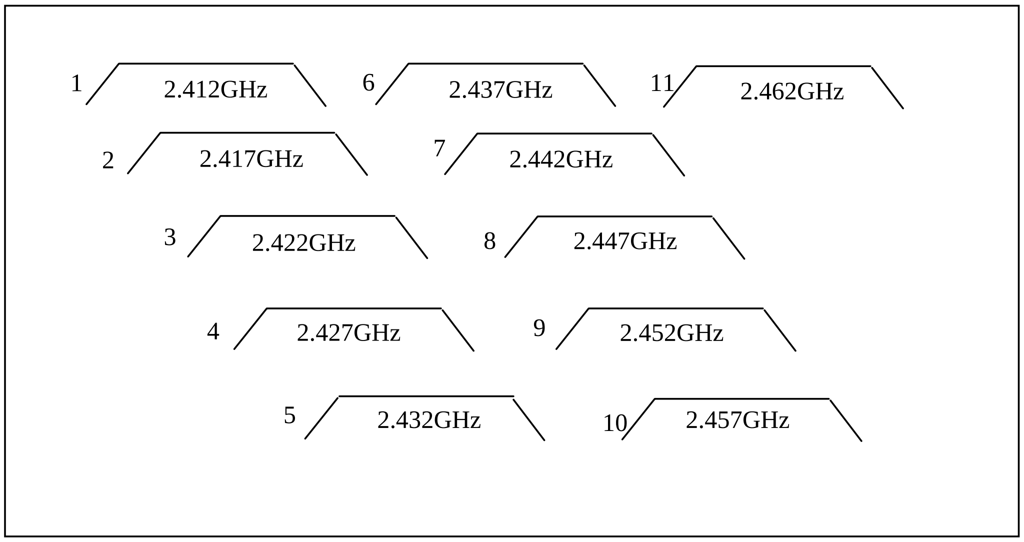
A.4.3 Direct Sequence Spread Spectrum (DSSS)

Direct Sequence Spread Spectrum, DSSS is the transmission rate used by 802.11b. The original data stream to be sent over the air interface is first XOR-ed with a spreading factor, also called a chipping code. This results in the data stream being broken into multiple sub-bits or chips, which are represented by a 1 or a 0. All of these chips are then transmitted over a much broader frequency range than the original data stream. At the receiver end, the received signal is fed into a decoder with the same chipping code to reassemble the original data.

Since the sequence is spread over several frequencies, this provides more immunity towards interference. A typical interference source, such as a Bluetooth device, which also operates in the 2.4GHz range, would only interfere with a small portion of the frequency band. This may cause some of the chips to be corrupted. However, at the receiver end, the signal will be fully recovered using the chipping code.

In 802.11, there are 11 channels or frequency ranges available for DSSS but only three of these are non-overlapping, see Figure A.5. This means that in an 802.11 network, such as 802.11b networks, there can only be three AP's that have overlapping coverage areas. If more than three access points are deployed in one coverage area, there will be higher bit error rates and lower data transfer rates. Therefore, while DSSS supports greater transfer rates, FHSS supports a greater density of users.

Figure A.5 Frequency Channel overlap in the ISM band



A.4.4 Orthogonal Frequency Division Multiplexing (OFDM)

This is the technique used by IEEE802.11a and HiperLan/2 at the 5GHz frequency band. The original data stream is divided into several interleaved parallel bit streams where each stream modulates a separate sub-carrier. By modulating the channel in this way, the channel frequency is divided into several independent sub channels. This method is very efficient in its use of the radio spectrum. However, due to the high frequency band, 5GHz, the transmission range is quite small, 30 to 150m only.

A.5 Medium Access Control Layer (MAC Layer)

A.5.1 Introduction

The main function for this layer is to control access to the wireless environment. The MAC layers supports other functions include fragmentation, encryption, power management, synchronization and roaming support.

A.5.2 Access Scheme

IEEE802.11 supports two different access methods. Distributed Coordinator Function (DCF) is the basic mandatory scheme and Point Coordinator Function (PCF) is the optional scheme. DCF uses the basic access method called Carrier Sense Multiple Access/ Collision Avoidance (CSMA/CA) to mediate access to the shared medium. In simple terms the method works as follows: if a device wishes to transmit, it listens to the medium and if the medium is not busy it begins transmission. If the medium is busy it waits a certain amount of time before trying again. In this situation two devices may sense that the medium is free and start transmitting at the same time, causing a collision. To overcome this problem, if a device senses that the medium is free, it waits a given time, called Distributed Inter Frame Space (DIFS) and then sends a ready to send (RTS) message across the medium to the intended receiving device, which responds with a clear to send (CTS) message. The medium is then reserved for this transmission for a while and all other devices have to wait a given time before trying to transmit again.

This method is similar to the access method used on 802.3, Ethernet, which is called Carrier Sense Multiple Access/ Collision Detection (CSMA/CD). This is one of the most pervasive access methods for wired networks. However, it cannot be used for wireless networks, as a full duplex medium is needed, which is capable of transmitting and receiving data simultaneously for this method. The Collision Avoidance method used by 802.11 overcomes this problem. The receiving device checks the CRC of the received data and sends an acknowledgement packet (ACK). On receiving the ACK, the transmitting device discerns that no collision occurred. If no ACK, is received the transmitting device retransmits the data for a given number of retransmissions, after which the data is discarded.

The PCF is an optional part of the standard, which gives time critical or delay sensitive packets priority over regular data transmissions. If PCF is present, it coexists with DCF with both schemes sharing the use of the medium. It uses a polling procedure to setup a contention free period, which takes higher priority than the DCF procedure. The contention free period (CFP) is started by the access point broadcasting a special beacon frame. The access point keeps a list of devices that have requested to send data using PCF. During the CFP, the AP sends poll frames to the devices when they are clear to access the medium. This allows higher priority to be given to delay sensitive data such as voice or video data.

A.5.3 Fragmentation and Reassembly

The MAC layer also provides the functionality of fragmentation and reassembly. Wireless LANs prefer smaller packets because the smaller the packet, the less chance it has of being corrupted. Also if it is corrupted and has to be resent, a smaller packet requires less overhead. The fragmentation mechanism breaks up larger packets such as Ethernet packets that could be up to 1518bytes long into smaller packets before transmitting them. After a smaller packet has been transmitted, the transmitting device waits for an ACK before transmitting the next packet. If it does not receive an ACK, it resends the packet. If no ACK is received after resending a packet a set amount of times, the whole frame is dropped. On the receiving side, the packet is reassembled into the original larger packet at the MAC layer.

A.5.4 Security

The MAC layer provides the basic security for the wireless network. It does this using several different methods. These are the use of a Service Set Identifier (SSID), MAC Address Filtering and Wired Equivalent Protocol (WEP). To achieve the best security for a WLAN network, all of these methods should be implemented.

A.5.4.1 Service Set Identifier (SSID)

The SSID is the identifier associated with an access point or a group of AP's. The access point is programmed with an SSID corresponding to a specific wireless network. In order to access this network, client devices must be configured with this SSID. Therefore the SSID acts as a password to provide a minimal level of security. Configuring the access point to broadcast its SSID can compromise this level of security.

A.5.4.2 MAC Address Filtering

A client device can be identified by a unique MAC address, which is stored in the 802.11 network card. As an added security measure, each AP can be programmed with a list of the MAC addresses of the devices, which are associated with it. If a devices MAC address is not in the list, the device will not be allowed to access the wireless network.

A.5.4.3 Wired Equivalent Protocol

The MAC layer provides protection for the wireless network by use of an encryption method called Wired Encryption Protocol (WEP). This is necessary as wireless networks are easy to connect to and be eavesdropped on. WEP is a single shared key system in which a basic 40-bit encryption is applied to all packet transmissions on the network and then decrypted using the same key. The key is stored in the access points in the network and in the client devices computer. The use of the WEP slows down transmissions and increases the overhead of packet transmissions. Also several security flaws have been found in the algorithm.

Appendix B

B Matlab Implementation Files

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OLP_empirical - This program takes a given area, e.g. the floor of a building, with three
% access points in it, and graphs a signal strength graph for that floor. It then takes the
% power received at the receiver from 3 access points and from this information calculates
% the position of the receiver. Next it checks if the power received at this position is above a
% certain threshold value, e.g. 75% of the poser received. If it is not, then it searches the area
% around the receiver and finds the position with the highest signal strength closest to the
% receiver that is above the threshold.
%
% Input  PR1  database of signal strength around given area from AP1
%        PR2  database of signal strength around given area from AP2
%        PR3  database of signal strength around given area from AP3
%        pow_rec_ap1 power received at the receiver from AP1
%        pow_rec_ap2 power received at the receiver from AP2
%        pow_rec_ap3 power received at the receiver from AP3
% Output    None
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function OLP_empirical (PR1, PR2, PR3, pow_rec_ap1, pow_rec_ap2, pow_rec_ap3)


clc;       % Clear the screen


% Get the maximum signal strength at each position
PR_temp = max(PR1,PR2);
PR = max(PR_temp,PR3);


% Call the function to plot the graph
power_graph(PR);


rec_pow = [pow_rec_ap1, pow_rec_ap2, pow_rec_ap3];


% Call the function which calculates the distance the receiver is from the three access points.
 Dist_to_APs = power_to_distance(rec_pow);


% Calls the trilat function, which takes the distances and calculates the position of the
% receiver
```

```
 rec_pos_trilat = trilat(Dist_to_APs)

% Calls the mmc_method function, to calculate the position of the receiver
 rec_pos_dcm_1 = mmc_method(rec_pow, PR1, PR2, PR3);

% Find the position of the local optimal power level
% First find the current power level
power_value = PR(rec_pos_dcm_1(1),rec_pos_dcm_1(2));

% Check if this power value is within the given threshold
t = get_threshold(power_value);

% Set optimal_position = -1 this is to check if a move is necessary
optimal_position = -1;

% Call the function to find the local optimal position if necessary
if t ==1
   optimal_position = OLP_alg(PR, rec_pos_dcm_1);
end

% subtracts 1 to change values to 0 x 0 grid system.
rec_pos_dcm = [rec_pos_dcm_1(1)-1 rec_pos_dcm_1(2)-1]
text(rec_pos_dcm(1), rec_pos_dcm(2), '\leftarrow Receiver Position', 'Fontsize', 14);

% Calls the A* algorithm to find the shortest path
if optimal_position ~= -1
   path = a_stars(rec_pos_dcm(1), rec_pos_dcm(2), optimal_position(1), optimal_position(2));

   [len,width] = size(path);
   w = 2*width;
   x = [path(1:2:w)];
   y = [path(2:2:w)];



    % Plots the shortest path
   plot(x,y, '--rs', 'LineWidth',2);
```

```
    text(optimal_position(1), optimal_position(2), '\leftarrow Local Optimal Position',
'Fontsize', 14);


end % of if statement
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OLP_theoretical - This program takes a given area, e.g. the floor of a building, with three
access  points in it, and graphs a signal strength graph for that floor. It then takes the power
received at the receiver from 3 access points and from this information calculates the position
of the receiver. Next it checks if the power received at this position is above a certain
threshold value, e.g. 75% of the poser received. If it is not, then it searches the area around
the receiver and finds the position with the highest signal strength closest to the receiver that
is above the threshold.
%
% Input  PR1  database of signal strength around given area from AP1
%        PR2  database of signal strength around given area from AP2
%        PR3  database of signal strength around given area from AP3
%        pow_rec_ap1 power received at the receiver from AP1
%        pow_rec_ap2 power received at the receiver from AP2
%        pow_rec_ap3 power received at the receiver from AP3
% Output    None
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function OLP_theoretical (pow_rec_ap1, pow_rec_ap2, pow_rec_ap3)


clc;       % Clear the screen


% Calculate power emitted from Access Point 1
AP1 = [0, 0];
D = get_distance(AP1);
PR1 = get_power(D);


% Calculate power emitted from Access Point 2
AP2 = [10, 0];
D = get_distance(AP2);
PR2 = get_power(D);


% Calculate power emitted from Access Point 3
AP3 = [0, 8];
D = get_distance(AP3);
PR3 = get_power(D);


% Get the maximum signal strength at each position
PR_temp = max(PR1,PR2);
```

```
PR = max(PR_temp,PR3);
% Call the function to plot the graph
power_graph(PR);

rec_pow = [pow_rec_ap1, pow_rec_ap2, pow_rec_ap3];

% Call the function which calculates the distance the receiver is from the three access points.
 Dist_to_APs = power_to_distance(rec_pow);

% Calls the trilat function, which takes the distances and calculates the position of the
receiver
 rec_pos_trilat = trilat(Dist_to_APs)

% Calls the mmc_method function, to calculate the position of the receiver
  rec_pos_dcm_1 = mmc_method(rec_pow, PR1, PR2, PR3);

% Find the position of the local optimal power level
% First find the current power level
power_value = PR(rec_pos_dcm_1(1),rec_pos_dcm_1(2));

% Check if this power value is within the given threshold
t = get_threshold(power_value);

% Set optimal_position = -1 this is to check if a move is necessary
optimal_position = -1;

% Call the function to find the local optimal position if necessary
if t ==1
    optimal_position = OLP_alg(PR, rec_pos_dcm_1);
end

% subtracts 1 to change values to 0 x 0 grid system.
rec_pos_dcm = [rec_pos_dcm_1(1)-1 rec_pos_dcm_1(2)-1]
text(rec_pos_dcm(1), rec_pos_dcm(2), '\leftarrow Receiver Position', 'Fontsize', 14);

% Calls the A* algorithm to find the shortest path
if optimal_position ~= -1
```

```
   path = a_stars(rec_pos_dcm(1), rec_pos_dcm(2), optimal_position(1), optimal_position(2));


   [len,width] = size(path);
   w = 2*width;
   x = [path(1:2:w)];
   y = [path(2:2:w)];


   % Plots the shortest path
   plot(x,y, '--rs', 'LineWidth',2);
   text(optimal_position(1), optimal_position(2), '\leftarrow Local Optimal Position',
'Fontsize', 14);


end % of if statement
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% get_distance - Calculate the distance to the access point from each point in the area.
%
% Input     AP            -- The coordinates of the access point.
% Output    Dist_AP   -- A matrix the size of the area, with the distance from the access point
at each %                              point
% Fixed No  m, n        -- The size of the area, m is the x-coordinate, n
%                              is the y coordinate
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function Dist_AP = get_distance(AP)

for m=1:15       % Choose the x-coordinate of the map.

   for n = 1:13     % Choose the y-coordinate of the map.
       m = m-1;   % Start at location [0,0]
       n = n-1;
       % Calculate the distance from each point to the access point.
       d = sqrt((AP(1) - m).^2 + (AP(2) - n).^2);

       m = m+1;   % Change back for array indexing
       n = n+1;
       Dist_AP(n,m) = d; % Place distance into array, distance = .1m

   end     % of for OLP for y-coordinate

end        % of for OLP for x-coordinate
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% get_power - Get the received power at each point in the grid using a propagation model.
% The following propagation models are implemented:
% - Freespace Propagation Model
% - One Slope Propagation Model
% - Multiwall Propagation Model
% - Linear Attenuation Propagation Model
%
% Input    D -- A matrix the size of the area, with the distance from the access point at each
point
% Output   PR -- A matrix of the size of the area, which holds the power received from the
%                      access point at each point in the grid.
% Fixed No
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function PR = get_power(D)


Gt = 0;            %Gain at the transmitter antenna
Pt = 15.05;        %Power at the transmitter
Gr = 0;            %Gain at the receiving antenna
lm = .125;         %Wavelength of the signal
t = 0;             %Will hold the number of walls between transmitter and receiver
delta = .62;       % As given by COST 231

[r,c] = size(D);    % Get the row and column length of the matrix.

%For indexing through the matrix D
for m = 1:r

   for n = 1:c

      if D(m,n) == 0      % At access point power PR =Pt

      PR(m,n) = -(Pt + Gt + Gr);

      else
         % Calculate the power received
```

```
            %FREESPACE Model
            %PR(m,n) = Pt + Gt + Gr - (40.046 + 20*log10(D(m,n)));

            % COST 231 One Slope Model also called Log Distance Model
            PR(m,n) = Pt + Gt + Gr - (40.046 + 35*log10(D(m,n)));

            % COST 231 Multi Wall Model
            t=0;
             if ((n > 5 & n < 8) | ( m > 5 & m < 8) )
                 t = 1;
             elseif (( n > 8 & n < 11) | ( m > 8 & m < 11))
                t = 2;
            elseif (( n > 11 ) | ( m > 11))
                t = 3;
            end

%           t =1.5;
             %PR(m,n) = Pt  + Gt + Gr - (3.4*t + 6.9 + 40.046 + 20*log10(D(m,n)));

            %Linear Attenuation Model COST 231
            %PR(m,n) = Pt + Gt + Gr - (40.046 + 20*log10(D(m,n)) + delta*D(m,n));



        end     % of if D

    end         % of for OLP through rows of database

end             % of for OLP through columns of database
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% power_graph - plots the power received in a mesh or surf graph
%
% Input  PR -- A matrix of the size of the area, which holds the maximum power received
%              from the access points at each point in the grid.
% Output       -- None
% Fixed No     -- The resolution values for interpolation
% Author   Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function power_graph(power_rec)

[r,c] = size(power_rec);
xgrid  = 0:c-1;       %x-coordinates for the 2-d plane
ygrid = 0:r-1;        %y-coordinates for the 2-d plane
xx= 0: 0.05 : c-1;     %Choose resolution for interpolation
yy = 0: 0.05 :r-1;

yy1 =yy';          % Change array into vector for interp2

zz = interp2(xgrid, ygrid, power_rec, xx, yy1, 'cubic');  % interpolates the data

mesh(xx,yy1,zz);       %plots a mesh plot, can use surf
%contourf(xx,yy1,zz);
xlabel('x - coordinate', 'FontSize', 12);
ylabel('y - coordinate', 'FontSize', 12);
zlabel('Power Received', 'FontSize', 12);
title('Power Received in the x-y plane from the modelled data', 'FontSize', 16);
colormap(hsv)          %can be set to hsv,gray,hot, bone, prism,etc
axis('xy');
grid on;
hold on;
colorbar;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% power_to_distance - Calculates the distance from each access point the receiver is, by
% using the reverse  of the appropriate propagation model,
% - Freespace Propagation Model
% - Indoor Propagation Model
% - Wall Attenuation Factor Model
% - Random Propagation Model
%
% Input     pow_ap      -- The power received from each of the access points at the receiver.
% Output    D_AP        -- The distance to each access point from the receiver.
% Fixed No
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function D_AP = power_to_distance(pow_AP)

 Gt = 0;            % Gain at the transmitter antenna
 Pt = 15.05;           % Power at the transmitter
 Gr = 0;            % Gain at the receiving antenna
[r,s] = size(pow_AP);    % Gets the size of the  matrix passed in for use in the for OLP

for m = 1:s

   % Freespace Model
   %t = 1.5;
   %D_AP(m) = 10.^((Pt   - pow_AP(m) - 40 - 3.4*t - 6.9)/25);

   % COST 231 One Slope Model
   D_AP(m) = 10^((Pt + Gt + Gr - pow_AP(m) - 40.046 )/35);

   % COST 231 Multi Wall Model
   %t = 1.5;
   %D_AP(m) = 10.^((Pt +Gt + Gr  - pow_AP(m) - 40.046 - 3.4*t - 6.9)/20);
   %Linear Attenuation Model
   %D_AP(m) = 10.^((Po - pow_AP(m) - 40.046)/x);
end % of For OLP
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% trilat - Calculates the position of the receiver from the given distances.
%
% Input     AP_dist          -- The distance to each access point from the receiver.
% Output    rec_pos_trilat  -- The position of the receiver.
% Fixed No  -- The position of the access points.
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function rec_pos_trilat = trilat(AP_dist)

AP1 = [0; 0];     % Access Point 1
AP2 = [10; 0];    % Access Point 2
AP3 = [0; 8];     % Access Point 2

Lb1 = sqrt((AP2(1) - AP1(1)).^2 + (AP2(2) - AP1(2)).^2); % Distance between AP1 and AP2
Lb2 = sqrt((AP3(1) - AP2(1)).^2 + (AP3(2) - AP2(2)).^2); % Distance between AP2 and AP3
Lb3 = sqrt((AP1(1) - AP3(1)).^2 + (AP1(2) - AP3(2)).^2) ; % Distance between AP3 and AP1

X = (AP_dist(1).^2 - AP_dist(2).^2 + Lb1.^2)/(2*Lb1); % Calculate X coordinate

C1 = sqrt(AP_dist(1).^2 - X.^2);

Xb = (Lb3.^2 - Lb2.^2 +Lb1.^2)/(2*Lb1);

Cb = sqrt(Lb3.^2 - Xb.^2);

Dg1 = sqrt(C1.^2 + (Xb - X).^2);

Y = (Dg1.^2 - AP_dist(3).^2 + Cb.^2)/(2*Cb);  % Calculate Y coordinate

rec_pos_trilat = [ Y X];          % Returns the position
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mmc_method - Multiple matrix correlation - checks the three power databases to find a
% location that matches where those powers are received.
% Input  P1     - The power received from AP1 at all positions in the area.
%          P2     - The power received from AP2 at all positions in the area.
%          P3     - The power received from AP3 at all positions in the area.
%          pow_rec- Contains the power received at the receiver from each
%                  of the three access points.
% Output    rec_pos_dcm -- Contains the position of the receiver.
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function rec_pos_dcm = mmc_method_empirical(pow_rec, P1, P2, P3)


[r,c] = size(P1);   % get the size of the databases
test(1:r,1:c) = 1;  % Will hold the result of the pass through the first database P1
power_diff_search_1 = 13;  % half the diffence between high value and next highest value,
                      % biggest difference possible
power_diff_search_2 = 9;
power_diff_search_3 = 9;
count = 2;              % Holds the number of positions that pass through
inc_1 = 2;
inc_2 = 2;
while count > 1        % Continue until only one value left.
   count = 0;          % Reinitialise the counter


   % Searching database P1 for matches
   for m = 1:c        %index for the colums of the database


      for n = 1:r     % index for the rows of the database


         if test(n,m) == 1


            % is the power value in P1 close to the power value of the
            % receiver by a difference of power_diff_search.
           if ((pow_rec(1) >= P1(n,m) - power_diff_search_1) & (pow_rec(1) <= P1(n,m) +
           power_diff_search_1))


               test(n,m) = 1;
```

```
                count = count + 1;

              else

                test(n,m) = 0;

              end     % of if pow_rec

            end % of if OLP

          end           % of for OLP through rows

        end             % of for OLP through columns

        if count > 1
          power_diff_search_1 = power_diff_search_1/1.4;
        elseif count < 1
          power_diff_search_1 = power_diff_search_1*inc_1;
          test(1:n,1:m) = 0;
          inc_1 = inc_1 + 1;
        end

        count = 0;

        % Searching database P2 for matches
        for q = 1:c        %index for the colums of the database

          for p = 1:r     % index for the rows of the database

            if test(p,q) == 1

              if ((pow_rec(2) >= P2(p,q) - power_diff_search_2) & (pow_rec(2) <= P2(p,q) +
      power_diff_search_2))

                test(p,q) = 1;
                count = count + 1;
              else
```

```
                test(p,q) = 0;

              end     % of if pow_rec

           end        % of if test

        end           % of for OLP through rows

     end              % of for OLP through columns

     if count > 1
        power_diff_search_2 = power_diff_search_2/1.4;
     elseif count < 1
        power_diff_search_2 = power_diff_search_2*inc_2;
        test(1:n,1:m) =1;
        inc_2 = inc_2 + 1;
     end
         count = 0;

     % Searching database P3 for matches
     for s = 1:c        %index through the colums of the database

        for t = 1:r     % index through the rows of the database

           if test(t,s) == 1

              if ((pow_rec(3) >= (P3(t,s) - power_diff_search_3)) & (pow_rec(3) <= (P3(t,s) +
  power_diff_search_3)))

                 test(t,s) = 1;
                 count = count + 1;
                 rec = [t,s];

              else

                 test(t,s) = 0;
        end   % of if pow_rec

      end     % of if test

    end       % of for OLP through rows

  end         % of for OLP through columns

  if count > 1
    power_diff_search_3 = power_diff_search_3/1.4;
  elseif (count < 1)
    power_diff_search_3 = power_diff_search_3*2;
    count =2;
    test(1:n,1:m) = 1;
  end

end           % of while OLP

rec_pos_dcm = [rec(1) rec(2)];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% get_threshold - Tests if the power is above a threshold value
%
% Input     pow         -- The power at a position in the grid
% Output    t           -- The test value, t = 0, pow is above threshold
% Fixed No  Pt          -- The power transmitted at an access point
%           threshold   -- The threshold percentage
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function t = get_threshold(pow_rec)

% set the percentage for the threshold
threshold = 50;
Pt = -15.05;

% The threshold power is 75% of the power transmitted.
pow_threshold = 18 * 100/threshold ;
t =0;

% Test if power received is less than the threshold
if (pow_rec < -pow_threshold)
   t = 1;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OLP_alg - Get the local maximum power.
%
% Input     PR - A matrix of the size of the area, which holds the maximum power received
from the access % points at each point in the grid.
%           rec_pos     -- The position of the receiver
% Output    new_pos     -- The optimal local position
% Fixed No  none
% Author    Nicola Lenihan
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function new_pos = OLP_alg(PR, rec_pos)

% If the power value is within a given threshold will hold 0 else will hold 1.
test = 1;
test_size = 0;      % Sets the size of the local area to be tested;
[r,c] = size(PR);    % Get the row and column length of the matrix.
count = 0;
rec_local_row = 1;
rec_local_col = 1;

while (test == 1) & (count < 10)

   % Increases the size of the local area to be tested
   test_size = test_size+1;

   % Getting the area for the local area -
   % Number of rows - Top of local area
   if (rec_pos(1) - test_size) < 1
      local_area_row_top = 1;

   else
      local_area_row_top = rec_pos(1) - test_size;
      rec_local_row = rec_local_row + 1;

   end

   % Bottom of local area
   if rec_pos(1) + test_size > r
      local_area_row_bottom = r;

   else
      local_area_row_bottom = rec_pos(1) + test_size;
     end

   % Number of columns - left of local area
   if rec_pos(2)-test_size < 1
      local_area_column_left = 1;

   else
      local_area_column_left = rec_pos(2) - test_size;
      rec_local_col = rec_local_col +1;
   end

   % Number of columns - right of local area
   if rec_pos(2) + test_size > c
      local_area_column_right = c;

   else
      local_area_column_right = rec_pos(2) + test_size;

   end

   % Storing the area to be tested into local_area
   local_area = PR(local_area_row_top:local_area_row_bottom,
local_area_column_left:local_area_column_right);

   % Getting the maximum value in the local area
   [h,y] = max(local_area);
   [z,x] = max(h);

   % Testing if this value is within the threshold
   test = get_threshold(local_area(y(x),x ));
   % x and y here will give you the directions and distance to move by.
   count = count + 1;

end % of While OLP

% Finds the indices of the receiver within the local area.
local_area_pos = [y(x), x];

% Convert back from local_area to full grid system.
% The Y-coordinate, the row value in the matrix.
if rec_local_row == rec_pos(1)
   new_pos(1) = local_area_pos(1);

elseif rec_pos(1) > rec_local_row
   new_pos(1) = local_area_pos(1) + (rec_pos(1) - rec_local_row);

elseif rec_pos(1) < rec_local_row
   new_pos(1) = local_area_pos(1) + (rec_local_row - rec_pos(1));
end

if rec_local_col == rec_pos(2)
   new_pos(2) = local_area_pos(2);

elseif rec_pos(2) > rec_local_col
   new_pos(2) = local_area_pos(2) + (rec_pos(2) - rec_local_col);

elseif rec_pos(2) < rec_local_col
   new_pos(2) = local_area_pos(2) + (rec_local_row - rec_pos(2));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

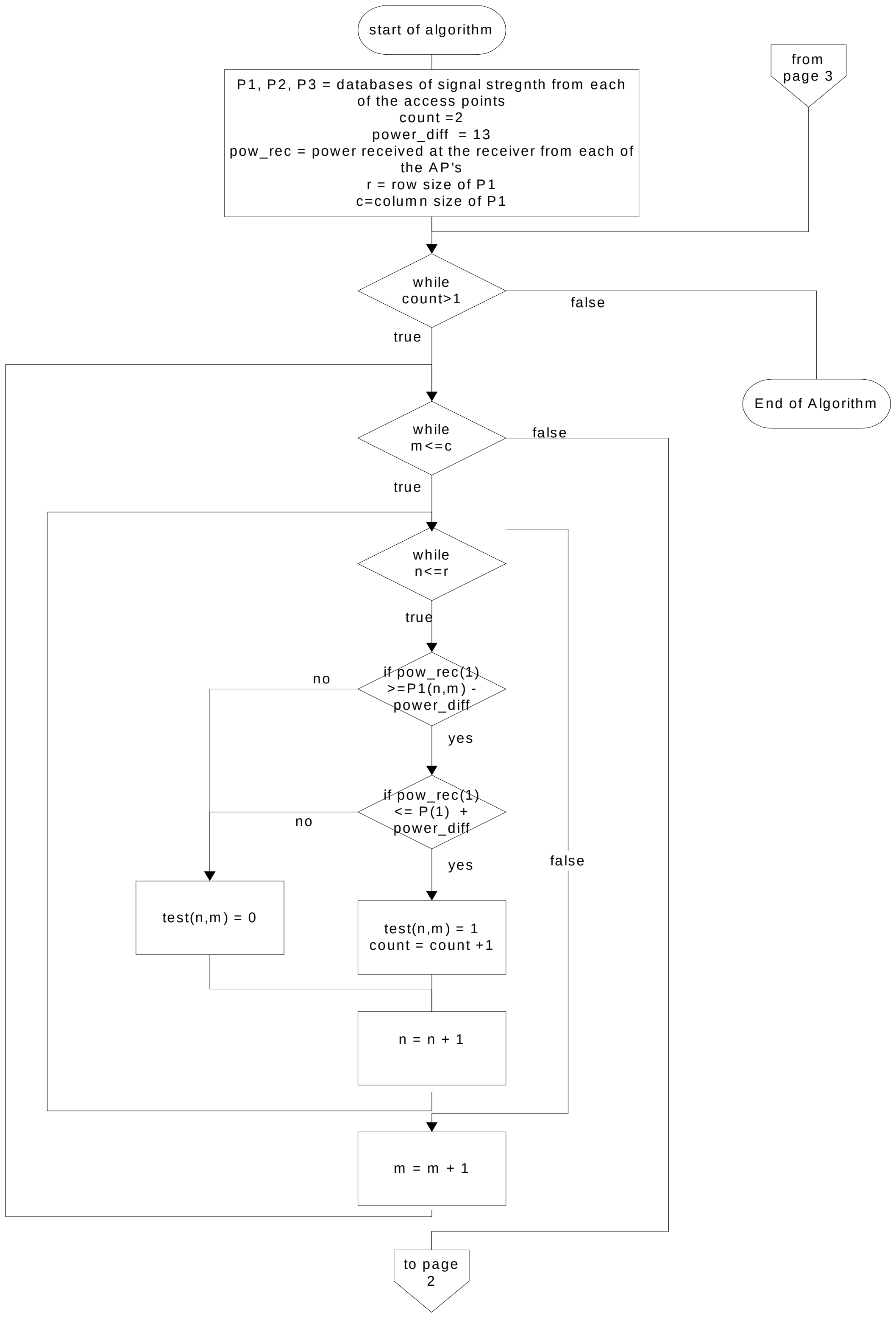
Appendix C

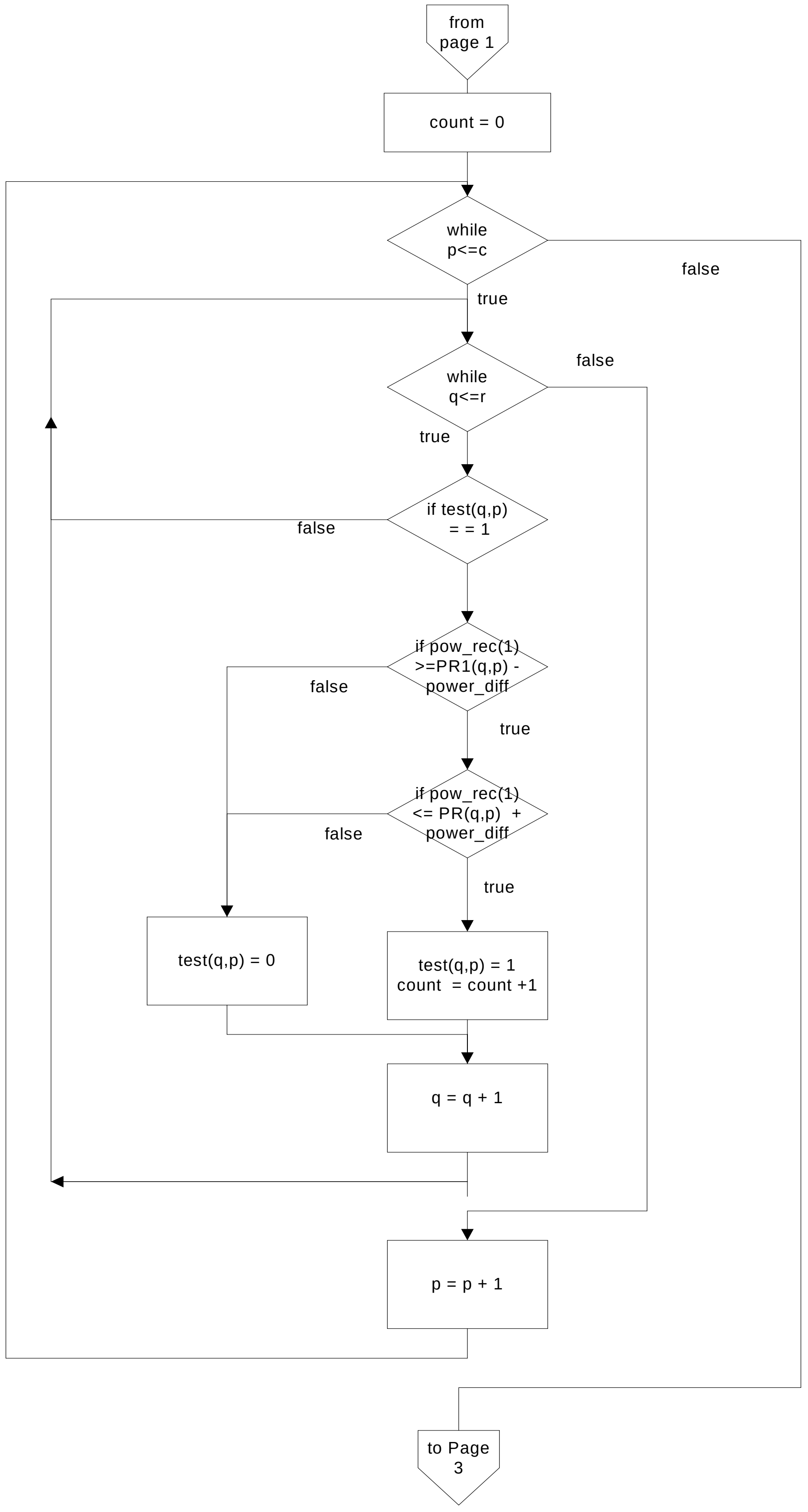
C Log file for position (10,12)

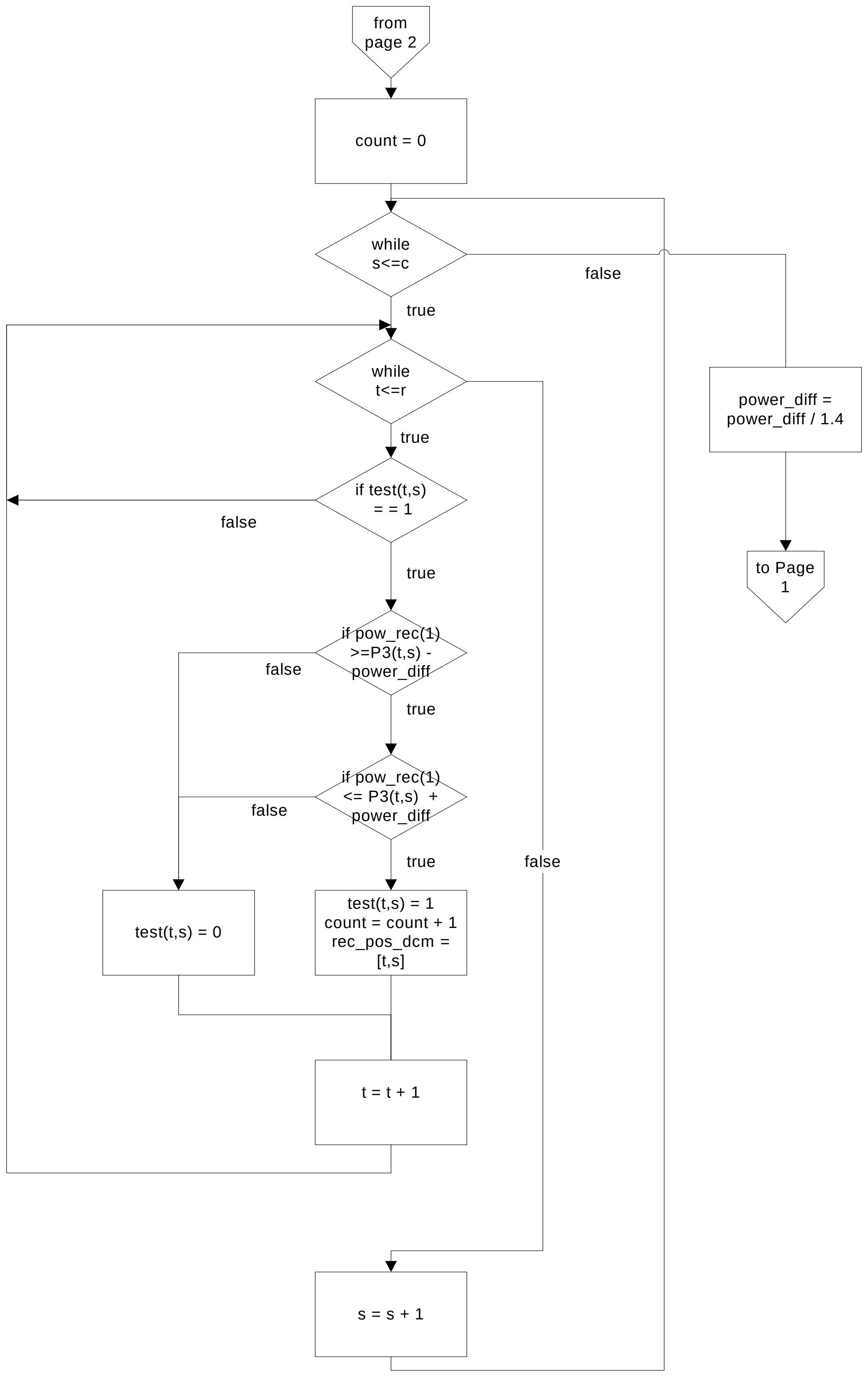
```
ORiNOCO Client Manager Log

Client Manager Variant 1 Version 2.90
2003/11/20-12:29
Monitor Log Session
Run on Station: RUGGED address: 00022D8B81D7 (This station)
Selected SSID: WAR Network
Acronyms:
BSSID=MAC address
Ch=Channel
Sl=Signal level
Nl=Noise level
SNR=Signal to Noise Ratio
Ap Name=Access Point Name

End Monitor Log Session
2003/11/20-12:30

Automatic log
,,Start time      ,End time        ,BSSID       ,  Ch,  Sl,  Nl,  Snr,AP Name
,,2003/11/20-12:29,2003/11/20-12:30,00022D8B81D5,   7, -65, -92,   27,WAR_AP_03
,,2003/11/20-12:29,2003/11/20-12:30,00022D8B845D,  10, -69, -94,   25,WAR_AP_02
,,2003/11/20-12:29,2003/11/20-12:30,00022D9DFA3A,   3, -66, -95,   29,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -64, -92,   28,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -70, -94,   24,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -65, -96,   31,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -62, -92,   30,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -70, -94,   24,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -63, -96,   33,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -62, -92,   30,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -70, -94,   24,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -63, -96,   33,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -62, -92,   30,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -70, -94,   24,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -62, -95,   33,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -61, -92,   31,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -71, -95,   24,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -61, -95,   34,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -60, -92,   32,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -69, -95,   26,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -61, -96,   35,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -62, -92,   30,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -70, -95,   25,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -61, -94,   33,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -62, -92,   30,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -70, -95,   25,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -61, -94,   33,WAR_AP_01
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B81D5,   7, -60, -92,   32,WAR_AP_03
,,2003/11/20-12:30,2003/11/20-12:30,00022D8B845D,  10, -69, -95,   26,WAR_AP_02
,,2003/11/20-12:30,2003/11/20-12:30,00022D9DFA3A,   3, -62, -96,   34,WAR_AP_01
```

Appendix D

D MMC Algorithm Flow Chart

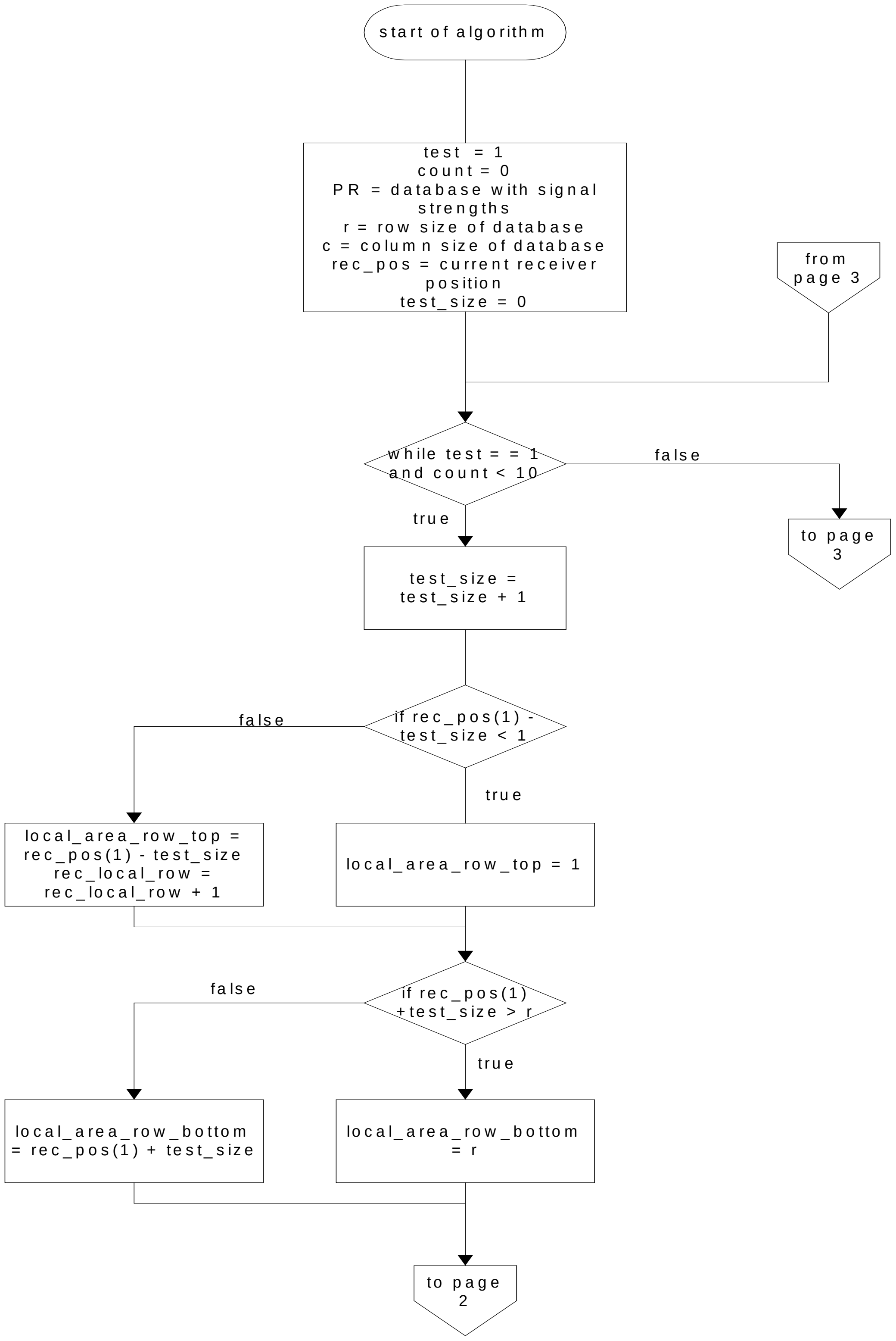


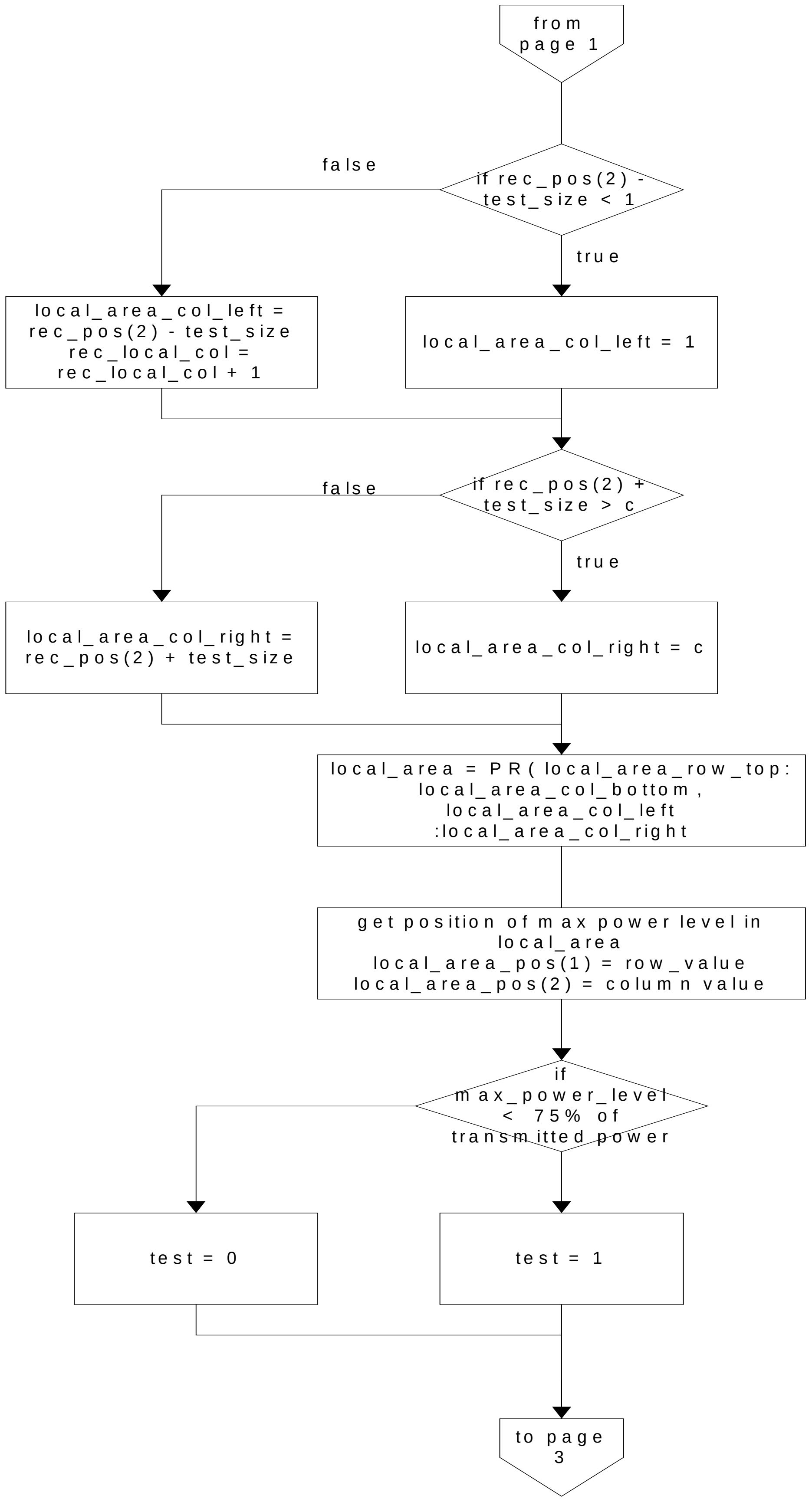

from page 1
count = 0
while p<=c
true
false
while q<=r
true
false
if test(q,p) = = 1
false
if pow_rec(1) >=PR1(q,p) - power_diff
true
false
if pow_rec(1) <= PR(q,p) + power_diff
true
false
test(q,p) = 0
test(q,p) = 1 count = count +1
q = q + 1
p = p + 1
to Page 3


from page 2
count = 0
while s<=c
false
true
while t<=r
true
if test(t,s) = = 1
false
true
if pow_rec(1) >=P3(t,s) - power_diff
false
true
if pow_rec(1) <= P3(t,s) + power_diff
false
true
false
test(t,s) = 0
test(t,s) = 1 count = count + 1 rec_pos_dcm = [t,s]
t = t + 1
s = s + 1
power_diff = power_diff / 1.4
to Page 1

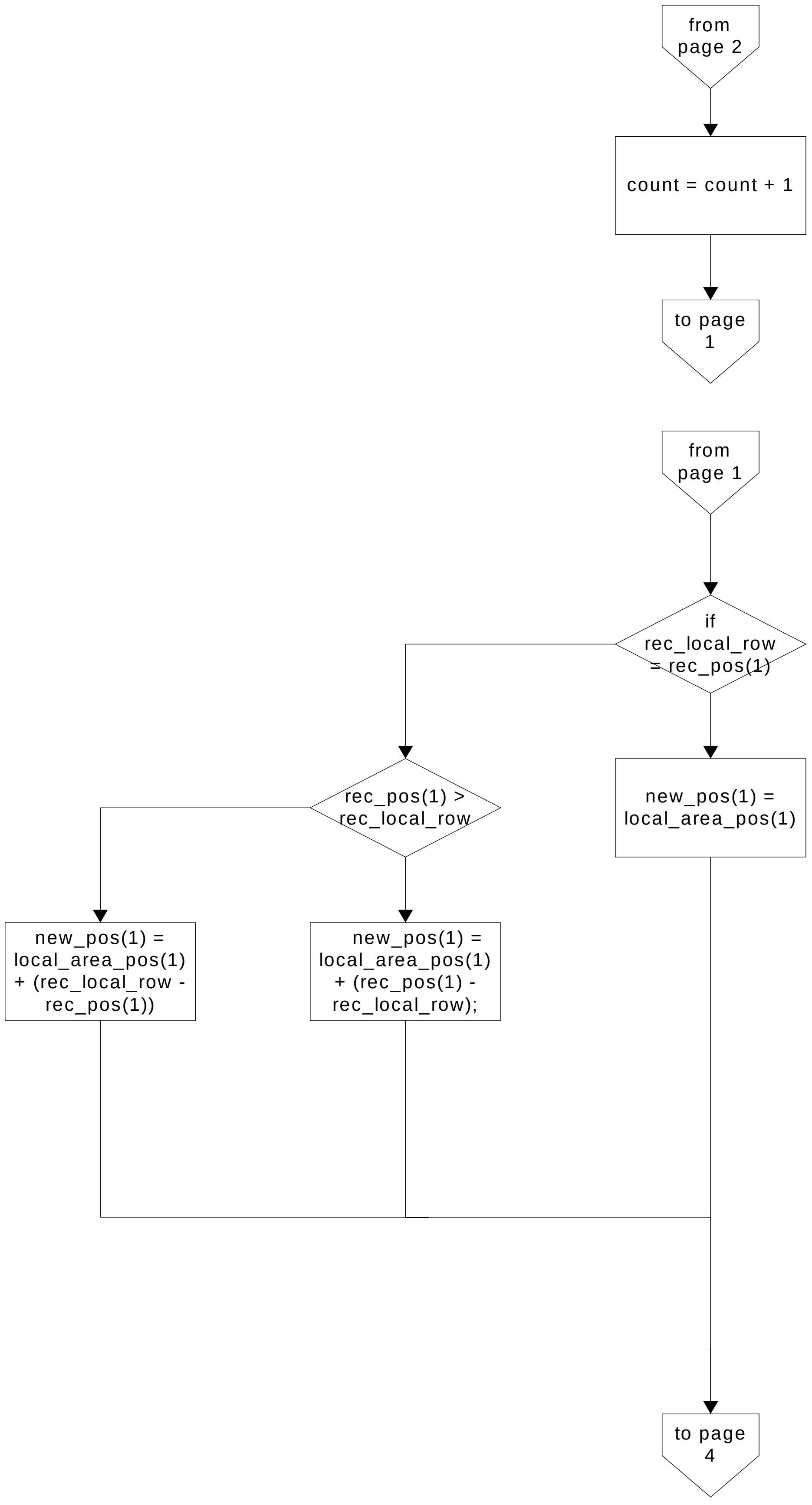
Appendix E

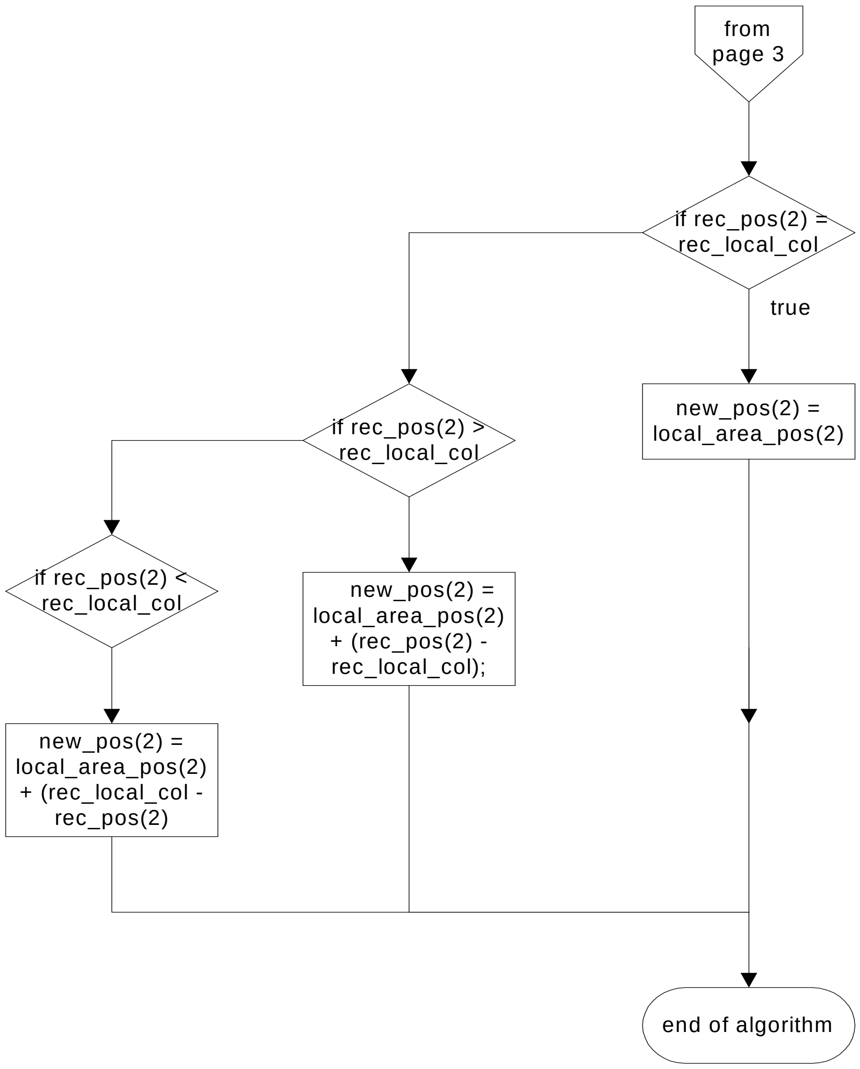
E OLP Algorithm Flow Chart


start of algorithm
test = 1
count = 0
PR = database with signal strengths
r = row size of database
c = column size of database
rec_pos = current receiver position
test_size = 0
from page 3
while test = = 1 and count < 10
false
to page 3
true
test_size = test_size + 1
if rec_pos(1) - test_size < 1
false
true
local_area_row_top = rec_pos(1) - test_size
rec_local_row = rec_local_row + 1
local_area_row_top = 1
if rec_pos(1) +test_size > r
false
true
local_area_row_bottom = rec_pos(1) + test_size
local_area_row_bottom = r
to page 2



from page 1
if rec_pos(2) - test_size < 1
false
true
local_area_col_left = rec_pos(2) - test_size
rec_local_col = rec_local_col + 1
local_area_col_left = 1
if rec_pos(2) + test_size > c
false
true
local_area_col_right = rec_pos(2) + test_size
local_area_col_right = c
local_area = PR(local_area_row_top: local_area_col_bottom , local_area_col_left :local_area_col_right
get position of max power level in local_area
local_area_pos(1) = row_value
local_area_pos(2) = column value
if max_power_level < 75% of transmitted power
test = 0
test = 1
to page 3


from page 2
count = count + 1
to page 1
from page 1
if rec_local_row = rec_pos(1)
rec_pos(1) > rec_local_row
new_pos(1) = local_area_pos(1)
new_pos(1) = local_area_pos(1) + (rec_local_row - rec_pos(1))
new_pos(1) = local_area_pos(1) + (rec_pos(1) - rec_local_row);
to page 4


from page 3
if rec_pos(2) = rec_local_col
true
new_pos(2) = local_area_pos(2)
if rec_pos(2) > rec_local_col
new_pos(2) = local_area_pos(2) + (rec_pos(2) - rec_local_col);
if rec_pos(2) < rec_local_col
new_pos(2) = local_area_pos(2) + (rec_local_col - rec_pos(2)
end of algorithm

Appendix F

F A * Algorithm Implementation in C++

```
//////////////////////////////////////////////////////////////////////////////////////////////////////////
// a_star.cpp : Defines the entry point for the DLL application.
// Finding a path on a simple grid maze
// This shows how to do shortest path finding using A*
//////////////////////////////////////////////////////////////////////////////////////////////////////////
#include "mex.h"
#include "a_star.h"
#include <iostream.h>
#include <stdio.h>
#define DEBUG_LISTS 0
#define DEBUG_LIST_LENGTHS_ONLY 0
#define DISPLAY_SOLUTION 0

using namespace std;
// Global data
// The Foundation Building map
const int MAP_WIDTH = 13;
const int MAP_HEIGHT = 15;
int map[ MAP_WIDTH * MAP_HEIGHT ] =
{
// 00 1  2 3  4  5  6  7  8  9  10 11 12
        1,1, 1,1, 1, 1, 1, 1, 1, 9, 1, 1, 1,    // 00
        1,1, 1,1, 1, 1, 1, 1, 1, 9, 1, 1, 1,    // 01
        1,1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1,    // 02
        1,1, 1,1, 1, 1, 1, 1, 1, 9, 1, 1, 1,    // 03
        1,9, 9,9, 1, 9, 9, 9, 9, 9, 1, 9, 1,    // 04
        1,1, 1,1, 1, 1, 9, 1, 1, 1, 1, 9, 1,    // 05
        1,1, 1,1, 1, 1, 9, 1, 1, 1, 1, 9, 1,    // 06
        1,1, 1,1, 1, 1, 9, 1, 1, 1, 1, 9, 1,    // 07
        9,9, 9,9, 9, 9, 9, 9, 9, 9, 9, 9, 1,    // 08
        1,1, 1,9, 1, 1, 9, 9, 1, 1, 1, 1, 1,    // 09
        1,9, 1,9, 9, 1, 9, 9, 1, 9, 9, 9, 1,    // 10
        1,1, 1,1, 1, 1, 1, 1, 1, 1, 1, 1, 1,    // 11
        9,1, 9,9, 1, 9, 9, 1, 9, 1, 9, 9, 1,    // 12
        1,1, 9,1, 1, 9, 1, 1, 9, 1, 1, 9, 1,    // 13
        1,1, 9,1, 1, 9, 1, 1, 9, 1, 1, 9, 1,    // 14
};

// map helper functions
int GetMap( int x, int y )
{
        if( x < 0 || x >= MAP_WIDTH || y < 0 || y >= MAP_HEIGHT)
        {
                return 9;
        }
        return map[(y*MAP_WIDTH)+x];
}

// Definitions
```

```
class MapSearchNode
{
public:
	unsigned int x;	// the (x,y) positions of the node
	unsigned int y;
	MapSearchNode() { x = y = 0; }
	MapSearchNode( unsigned int px, unsigned int py ) { x=px; y=py; }
	float GoalDistanceEstimate( MapSearchNode &nodeGoal );
	bool IsGoal( MapSearchNode &nodeGoal );
	bool GetSuccessors( AStarSearch<MapSearchNode> *astarsearch, MapSearchNode
*parent_node );
	float GetCost( MapSearchNode &successor );
	bool IsSameState( MapSearchNode &rhs );
	void PrintNodeInfo();
};

bool MapSearchNode::IsSameState( MapSearchNode &rhs )
{
	// same state in a maze search is simply when (x,y) are the same
	if( (x == rhs.x) && (y == rhs.y) )
	{
		return true;
	}
	else
	{
		return false;
	}
}

void MapSearchNode::PrintNodeInfo()
{
	char str[100];

	sprintf( str, "Node position : (%d,%d)\n", x,y );
	mexPrintf("%s", str);
//	cout << str;
}

// Here's the heuristic function that estimates the distance from a Node
// to the Goal.
float MapSearchNode::GoalDistanceEstimate( MapSearchNode &nodeGoal )
{
	float xd = float( ( (float)x - (float)nodeGoal.x ) );
	float yd = float( ( (float)y - (float)nodeGoal.y) );
	return ((xd*xd) + (yd*yd));
}

bool MapSearchNode::IsGoal( MapSearchNode &nodeGoal )
{
	if( (x == nodeGoal.x) &&
		(y == nodeGoal.y) )
	{
		return true;
	}
	return false;
}
```

```
// This generates the successors to the given Node. It uses a helper function called
// AddSuccessor to give the successors to the AStar class. The A* specific initialisation
// is done for each node internally, so here you just set the state information that
// is specific to the application
bool MapSearchNode::GetSuccessors( AStarSearch<MapSearchNode> *astarsearch,
MapSearchNode *parent_node )
{
        int parent_x = -1;
        int parent_y = -1;
        if( parent_node )
        {
                parent_x = parent_node->x;
                parent_y = parent_node->y;
        }
        MapSearchNode NewNode;
        // push each possible move except allowing the search to go backwards
        if( (GetMap( x-1, y ) < 9)
                && !((parent_x == x-1) && (parent_y == y)))
        {
                NewNode = MapSearchNode( x-1, y );
                astarsearch->AddSuccessor( NewNode );
        }
        if( (GetMap( x, y-1 ) < 9) && !((parent_x == x) && (parent_y == y-1)))
        {
                NewNode = MapSearchNode( x, y-1 );
                astarsearch->AddSuccessor( NewNode );
        }
        if( (GetMap( x+1, y ) < 9)
                && !((parent_x == x+1) && (parent_y == y)))
        {
                NewNode = MapSearchNode( x+1, y );
                astarsearch->AddSuccessor( NewNode );
        }
        if( (GetMap( x, y+1 ) < 9)
                && !((parent_x == x) && (parent_y == y+1)))
        {
                NewNode = MapSearchNode( x, y+1 );
                astarsearch->AddSuccessor( NewNode );
        }
        return true;
}

// given this node, what does it cost to move to successor. In the case
// of our map the answer is the map terrain value at this node since that is
// conceptually where we're moving
float MapSearchNode::GetCost( MapSearchNode &successor )
{
        return (float) GetMap( x, y );
}

int findsteps (double x_val_start[], double y_val_start[], double x_val_end[], double y_val_end[])
{
        // Our sample problem defines the world as a 2d array representing a terrain
        // Each element contains an integer from 0 to 5 which indicates the cost
        // of travel across the terrain. Zero means the least possible difficulty
```

```
		// in travelling (think ice rink if you can skate) whilst 5 represents the
		// most difficult. 9 indicates that we cannot pass.
		// Create an instance of the search class...
		AStarSearch<MapSearchNode> astarsearch;
		// Create a start state
		MapSearchNode nodeStart;
		nodeStart.x = x_val_start[0];
		nodeStart.y = y_val_start[0];
		// Define the goal state
		MapSearchNode nodeEnd;
		nodeEnd.x = x_val_end[0];
		nodeEnd.y = y_val_end[0];
		// Set Start and goal states
		astarsearch.SetStartAndGoalStates( nodeStart, nodeEnd );
		unsigned int SearchState;
		unsigned int SearchSteps = 0;
		do
		{
			SearchState = astarsearch.SearchStep();
			SearchSteps++;
#if DEBUG_LISTS
			int len = 0;
			mexPrintf( "Open:\n");
			MapSearchNode *p = astarsearch.GetOpenListStart();
			while( p )
			{
				len++;
#if !DEBUG_LIST_LENGTHS_ONLY
				((MapSearchNode *)p)->PrintNodeInfo();
#endif
				p = astarsearch.GetOpenListNext();
			}
			mexPrintf( "Open list has %d nodes\n", len);
			len = 0;
			mexPrintf("Closed:\n");
			p = astarsearch.GetClosedListStart();
			while( p )
			{
				len++;
#if !DEBUG_LIST_LENGTHS_ONLY
				p->PrintNodeInfo();
#endif
				p = astarsearch.GetClosedListNext();
			}
#endif
		}
		while( SearchState ==
AStarSearch<MapSearchNode>::SEARCH_STATE_SEARCHING );
		if( SearchState == AStarSearch<MapSearchNode>::SEARCH_STATE_SUCCEEDED )
		{
				MapSearchNode *node = astarsearch.GetSolutionStart();
#if DISPLAY_SOLUTION
				mexPrintf( "Displaying solution\n");
#endif
				int steps = 0;
				for( ;; )
```

```
                        {
                                node = astarsearch.GetSolutionNext();
                                if( !node )
                                {
                                        break;
                                }
                                steps ++;
                        };
                        // Once you're done with the solution you can free the nodes up
                        astarsearch.FreeSolutionNodes();
                        return steps;
        }
        return 0;
}

int findpath (double test[], double x_val_start[], double y_val_start[], double x_val_end[], double
y_val_end[])
{
        // Our sample problem defines the world as a 2d array representing a terrain
        // Each element contains an integer from 0 to 5 which indicates the cost
        // of travel across the terrain. Zero means the least possible difficulty
        // in travelling (think ice rink if you can skate) whilst 5 represents the
        // most difficult. 9 indicates that we cannot pass.
        // Create an instance of the search class...
        AStarSearch<MapSearchNode> astarsearch;
        // Create a start state
        MapSearchNode nodeStart;
        nodeStart.x = x_val_start[0];
        nodeStart.y = y_val_start[0];
        // Define the goal state
        MapSearchNode nodeEnd;
        nodeEnd.x = x_val_end[0];
        nodeEnd.y = y_val_end[0];
        // Set Start and goal states
        astarsearch.SetStartAndGoalStates( nodeStart, nodeEnd );
        unsigned int SearchState;
        unsigned int SearchSteps = 0;
        do
        {
                SearchState = astarsearch.SearchStep();
                SearchSteps++;
#if DEBUG_LISTS
                mexPrintf( "Steps: %d \n", SearchSteps);
                int len = 0;
                mexPrintf( "Open:\n");
                MapSearchNode *p = astarsearch.GetOpenListStart();
                while( p )
                {
                        len++;
#if !DEBUG_LIST_LENGTHS_ONLY
                        ((MapSearchNode *)p)->PrintNodeInfo();
#endif
                        p = astarsearch.GetOpenListNext();
                }
                mexPrintf( "Open list has %d nodes\n", len);
                len = 0;
```

```
                    mexPrintf("Closed:\n");
                    p = astarsearch.GetClosedListStart();
                    while( p )
                    {
                              len++;
#if !DEBUG_LIST_LENGTHS_ONLY
                              p->PrintNodeInfo();
#endif
                              p = astarsearch.GetClosedListNext();
                    }
                    mexPrintf( "Closed list has %d nodes\n", len);
#endif
          }
          while( SearchState ==
AStarSearch<MapSearchNode>::SEARCH_STATE_SEARCHING );
          if( SearchState == AStarSearch<MapSearchNode>::SEARCH_STATE_SUCCEEDED )
          {
                    mexPrintf( "Search found goal state\n");
                    MapSearchNode *node = astarsearch.GetSolutionStart();
#if DISPLAY_SOLUTION
                    mexPrintf( "Displaying solution\n");
#endif
                    int steps = 0;
                    test[steps] = node->x;
                    test[++steps] = node->y;
                    node->PrintNodeInfo();
                    for( ;; )
                    {
                              node = astarsearch.GetSolutionNext();
                              if( !node )
                              {
                                        break;
                              }
                              node->PrintNodeInfo();
                              //steps ++;
                              test[++steps] = node->x;
                              test[++steps]= node->y;
                    };
                    mexPrintf( "Solution steps: %d \n", steps);
                    // Once you're done with the solution you can free the nodes up
                    astarsearch.FreeSolutionNodes();
          }
          else if( SearchState == AStarSearch<MapSearchNode>::SEARCH_STATE_FAILED )
          {
                    mexPrintf( "Search terminated. Did not find goal state\n");
          }
          // Display the number of OLPs the search went through
          mexPrintf("SearchSteps : %d", SearchSteps, "\n");
          return 0;
}

void mexFunction( int nlhs, mxArray* plhs[], int nrhs, const mxArray* prhs[] )
{
          int n =2, m=20;
          double *results_vector, x;
          double *start_x, *start_y, *end_x, *end_y;
```

```
        if(nrhs!=4)
        {
                mexErrMsgTxt("Wrong number of inputs.");
        }
        start_x = mxGetPr(prhs[0]);
        start_y = mxGetPr(prhs[1]);
        end_x = mxGetPr(prhs[2]);
        end_y = mxGetPr(prhs[3]);
        x = mxGetScalar(prhs[0]);
         m = findsteps(start_x, start_y, end_x, end_y);
        plhs[0] = mxCreateDoubleMatrix(n /* 1 */, m+1, mxREAL);
        results_vector = mxGetPr(plhs[0]);
        findpath(results_vector, start_x, start_y, end_x, end_y);
}
A_Stars.h
//////////////////////////////////////////////////////////////////////////////////////////////////////////////
/ The following ifdef block is the standard way of creating macros which make exporting
// from a DLL simpler. All files in this DLL are compiled with the A_STAR_EXPORTS
// symbol defined on the command line. this symbol should not be defined on any project
// that uses this DLL. This way any other project whose source files include this file see
// A_STAR_API functions as being imported from a DLL, wheras this DLL sees symbols
// defined with this macro as being exported.

// used for text debugging
#include <iostream.h>
#include <stdio.h>
#include <conio.h>
#include <assert.h>

// stl includes
#include <algorithm>
#include <set>
#include <vector>
using namespace std;
// fast fixed size memory allocator, used for fast node memory management
#include "fsa.h"
// Fixed size memory allocator can be disabled to compare performance
// Uses std new and delete instead if you turn it off
#define USE_FSA_MEMORY 1
// disable warning that debugging information has lines that are truncated
// occurs in stl headers
#pragma warning( disable : 4786 )
// The AStar search class. UserState is the users state space type
template <class UserState> class AStarSearch
{
public: // data
        enum
        {
                SEARCH_STATE_NOT_INITIALISED,
                SEARCH_STATE_SEARCHING,
                SEARCH_STATE_SUCCEEDED,
                SEARCH_STATE_FAILED,
                SEARCH_STATE_OUT_OF_MEMORY,
                SEARCH_STATE_INVALID
        };
```

```
	// A node represents a possible state in the search
	// The user provided state type is included inside this type
	public:
	class Node
	{
		public:
Node *parent; // used during the search to record the parent of successor nodes
Node *child; // used after the search for the application to view the search in reverse
			float g; // cost of this node + it's predecessors
			float h; // heuristic estimate of distance to goal
	float f; // sum of cumulative cost of predecessors and self and heuristic
			Node() :
				parent( 0 ),
				child( 0 ),
				g( 0.0f ),
				h( 0.0f ),
				f( 0.0f )
			{
			}
			UserState m_UserState;
	};

	// For sorting the heap the STL needs compare function that lets us compare
	// the f value of two nodes
	class HeapCompare_f
	{
		public:
			bool operator() ( const Node *x, const Node *y ) const
			{
				return x->f > y->f;
			}
	};
public: // methods
	// constructor just initialises private data
	AStarSearch( int MaxNodes = 1000 ) :
		m_AllocateNodeCount(0),
		m_FreeNodeCount(0),
		m_FixedSizeAllocator( MaxNodes ),
		m_State( SEARCH_STATE_NOT_INITIALISED ),
		m_CurrentSolutionNode( NULL ),
		m_CancelRequest( false )
	{
	}

	// call at any time to cancel the search and free up all the memory
	void CancelSearch()
	{
		m_CancelRequest = true;
	}

	// Set Start and goal states
	void SetStartAndGoalStates( UserState &Start, UserState &Goal )
	{
		m_CancelRequest = false;
		m_Start = AllocateNode();
		m_Goal = AllocateNode();
```

```
			m_Start->m_UserState = Start;
			m_Goal->m_UserState = Goal;
			m_State = SEARCH_STATE_SEARCHING;
			// Initialise the AStar specific parts of the Start Node
			// The user only needs fill out the state information
			m_Start->g = 0;
m_Start->h = m_Start->m_UserState.GoalDistanceEstimate( m_Goal->m_UserState );
			m_Start->f = m_Start->g + m_Start->h;
			m_Start->parent = 0;
			// Push the start node on the Open list
			m_OpenList.push_back( m_Start ); // heap now unsorted
			// Sort back element into heap
			push_heap( m_OpenList.begin(), m_OpenList.end(), HeapCompare_f() );
			// Initialise counter for search steps
			m_Steps = 0;
		}

		// Advances search one step
		unsigned int SearchStep()
		{
			// Firstly break if the user has not initialised the search
			assert( (m_State > SEARCH_STATE_NOT_INITIALISED) &&
					(m_State < SEARCH_STATE_INVALID) );
		// Next I want it to be safe to do a searchstep once the search has succeeded...
			if( (m_State == SEARCH_STATE_SUCCEEDED) ||
				(m_State == SEARCH_STATE_FAILED))
			{
				return m_State;
			}
			// Failure is defined as emptying the open list as there is nothing left to
			// search...
			// New: Allow user abort
			if( m_OpenList.empty() || m_CancelRequest )
			{
				FreeAllNodes();
				m_State = SEARCH_STATE_FAILED;
				return m_State;
			}
			// Incremement step count
			m_Steps ++;
			// Pop the best node (the one with the lowest f)
			Node *n = m_OpenList.front(); // get pointer to the node
			pop_heap( m_OpenList.begin(), m_OpenList.end(), HeapCompare_f() );
			m_OpenList.pop_back();
			// Check for the goal, once we pop that we're done
			if( n->m_UserState.IsGoal( m_Goal->m_UserState ) )
			{
				// The user is going to use the Goal Node he passed in
				// so copy the parent pointer of n
				m_Goal->parent = n->parent;
				// A special case is that the goal was passed in as the start state
				// so handle that here
				if( n != m_Start )
				{
					//delete n;
					FreeNode( n );
			// set the child pointers in each node (except Goal which has no child)
					Node *nodeChild = m_Goal;
					Node *nodeParent = m_Goal->parent;
					do
					{
						nodeParent->child = nodeChild;
						nodeChild = nodeParent;
						nodeParent = nodeParent->parent;
					}
		while( nodeChild != m_Start ); // Start is always the first node by definition
				}
				// delete nodes that aren't needed for the solution
				FreeUnusedNodes();
				m_State = SEARCH_STATE_SUCCEEDED;
				return m_State;
			}
			else // not goal
			{
				// We now need to generate the successors of this node
				// The user helps us to do this, and we keep the new nodes in
				// m_Successors ...
				m_Successors.clear(); // empty vector of successor nodes to n
		// User provides this functions and uses AddSuccessor to add each successor of
				// node 'n' to m_Successors
				bool ret = n->m_UserState.GetSuccessors( this, n->parent ? &n-
>parent->m_UserState : NULL );
				if( !ret )
				{
					// free the nodes that may previously have been added
					for( vector< Node * >::iterator successor =
m_Successors.begin(); successor != m_Successors.end(); successor ++ )
					{
						FreeNode( (*successor) );
					}
				m_Successors.clear(); // empty vector of successor nodes to n
					// free up everything else we allocated
					FreeAllNodes();
					m_State = SEARCH_STATE_OUT_OF_MEMORY;
					return m_State;
				}
				// Now handle each successor to the current node ...
				for( vector< Node * >::iterator successor = m_Successors.begin();
successor != m_Successors.end(); successor ++ )
				{
					//	The g value for this successor ...
					float newg = n->g + n->m_UserState.GetCost( (*successor)-
>m_UserState );
			// Now we need to find whether the node is on the open or closed lists
			// If it is but the node that is already on them is better (lower g)
				// then we can forget about this successor
					// First linear search of open list to find node
					vector< Node * >::iterator openlist_result;
					for( openlist_result = m_OpenList.begin(); openlist_result !=
m_OpenList.end(); openlist_result ++ )
					{
```

```
                                                if( (*openlist_result)->m_UserState.IsSameState(
(*successor)->m_UserState ) )
                                                {
                                                        break;
                                                }
                                        }
                                        if( openlist_result != m_OpenList.end() )
                                        {
                                                // we found this state on open
                                                if( (*openlist_result)->g <= newg )
                                                {
                                                        FreeNode( (*successor) );
                                        // the one on Open is cheaper than this one
                                                        continue;
                                                }
                                        }
                                        vector< Node * >::iterator closedlist_result;
                                        for( closedlist_result = m_ClosedList.begin(); closedlist_result
!= m_ClosedList.end(); closedlist_result ++ )
                                        {
        if( (*closedlist_result)->m_UserState.IsSameState( (*successor)->m_UserState ) )
                                                {
                                                        break;
                                                }
                                        }
                                        if( closedlist_result != m_ClosedList.end() )
                                        {
                                                // we found this state on closed
                                                if( (*closedlist_result)->g <= newg )
                                                {
                                                        // the one on Closed is cheaper than this one
                                                        FreeNode( (*successor) );
                                                        continue;
                                                }
                                        }
                                        // This node is the best node so far with this particular state
                                        // so lets keep it and set up its AStar specific data ...
                                        (*successor)->parent = n;
                                        (*successor)->g = newg;
                                        (*successor)->h = (*successor)-
                                >m_UserState.GoalDistanceEstimate( m_Goal->m_UserState );
                                        (*successor)->f = (*successor)->g + (*successor)->h;
                                        // Remove successor from closed if it was on it
                                        if( closedlist_result != m_ClosedList.end() )
                                        {
                                                // remove it from Closed
                                                FreeNode(  (*closedlist_result) );
                                                m_ClosedList.erase( closedlist_result );
                                        }
                                        // Update old version of this node
                                        if( openlist_result != m_OpenList.end() )
                                        {
                                                FreeNode( (*openlist_result) );
                                                m_OpenList.erase( openlist_result );
                                                // re-make the heap
```

```
                                                    make_heap( m_OpenList.begin(), m_OpenList.end(),
HeapCompare_f() );
HeapCompare_f() );
                                        }
                                        // heap now unsorted
                                        m_OpenList.push_back( (*successor) );
                                        // sort back element into heap
                        push_heap( m_OpenList.begin(), m_OpenList.end(), HeapCompare_f() );
                                }
                                // push n onto Closed, as we have expanded it now
                                m_ClosedList.push_back( n );
                        } // end else (not goal so expand)
                        return m_State; // Succeeded bool is false at this point.
                }

                // User calls this to add a successor to a list of successors
                // when expanding the search frontier
                bool AddSuccessor( UserState &State )
                {
                        Node *node = AllocateNode();
                        if( node )
                        {
                                node->m_UserState = State;
                                m_Successors.push_back( node );
                                return true;
                        }
                        return false;
                }

                // Free the solution nodes
                // This is done to clean up all used Node memory when you are done with the
                // search
                void FreeSolutionNodes()
                {
                        Node *n = m_Start;
                        if( m_Start->child )
                        {
                                do
                                {
                                        Node *del = n;
                                        n = n->child;
                                        FreeNode( del );
                                        del = NULL;
                                } while( n != m_Goal );
                                FreeNode( n ); // Delete the goal
                        }
                        else
                        {
                // if the start node is the solution we need to just delete the start and goal nodes
                                FreeNode( m_Start );
                                FreeNode( m_Goal );
                        }
                }

                // Functions for traversing the solution
                // Get start node
```

```
UserState *GetSolutionStart()
{
        m_CurrentSolutionNode = m_Start;
        if( m_Start )
        {
                return &m_Start->m_UserState;
        }
        else
        {
                return NULL;
        }
}

// Get next node
UserState *GetSolutionNext()
{
        if( m_CurrentSolutionNode )
        {
                if( m_CurrentSolutionNode->child )
                {
                Node *child = m_CurrentSolutionNode->child;
                m_CurrentSolutionNode = m_CurrentSolutionNode->child;
                        return &child->m_UserState;
                }
        }
        return NULL;
}

// Get end node
UserState *GetSolutionEnd()
{
        m_CurrentSolutionNode = m_Goal;
        if( m_Goal )
        {
                return &m_Goal->m_UserState;
        }
        else
        {
                return NULL;
        }
}

// Step solution iterator backwards
UserState *GetSolutionPrev()
{
        if( m_CurrentSolutionNode )
        {
                if( m_CurrentSolutionNode->parent )
                {
                        Node *parent = m_CurrentSolutionNode->parent;
                        m_CurrentSolutionNode = m_CurrentSolutionNode->parent;
                        return &parent->m_UserState;
                }
        }
        return NULL;
}
```

```
// For educational use and debugging it is useful to be able to view
// the open and closed list at each step, here are two functions to allow that.

UserState *GetOpenListStart()
{
	float f,g,h;
	return GetOpenListStart( f,g,h );
}

UserState *GetOpenListStart( float &f, float &g, float &h )
{
	iterDbgOpen = m_OpenList.begin();
	if( iterDbgOpen != m_OpenList.end() )
	{
		f = (*iterDbgOpen)->f;
		g = (*iterDbgOpen)->g;
		h = (*iterDbgOpen)->h;
		return &(*iterDbgOpen)->m_UserState;
	}
	return NULL;
}

UserState *GetOpenListNext()
{
	float f,g,h;
	return GetOpenListNext( f,g,h );
}

UserState *GetOpenListNext( float &f, float &g, float &h )
{
	iterDbgOpen++;
	if( iterDbgOpen != m_OpenList.end() )
	{
		f = (*iterDbgOpen)->f;
		g = (*iterDbgOpen)->g;
		h = (*iterDbgOpen)->h;
		return &(*iterDbgOpen)->m_UserState;
	}
	return NULL;
}

UserState *GetClosedListStart()
{
	float f,g,h;
	return GetClosedListStart( f,g,h );
}

UserState *GetClosedListStart( float &f, float &g, float &h )
{
	iterDbgClosed = m_ClosedList.begin();
	if( iterDbgClosed != m_ClosedList.end() )
	{
		f = (*iterDbgClosed)->f;
		g = (*iterDbgClosed)->g;
		h = (*iterDbgClosed)->h;
		return &(*iterDbgClosed)->m_UserState;
```

```
			}
			return NULL;
		}
		UserState *GetClosedListNext()
		{
			float f,g,h;
			return GetClosedListNext( f,g,h );
		}
		UserState *GetClosedListNext( float &f, float &g, float &h )
		{
			iterDbgClosed++;
			if( iterDbgClosed != m_ClosedList.end() )
			{
				f = (*iterDbgClosed)->f;
				g = (*iterDbgClosed)->g;
				h = (*iterDbgClosed)->h;
				return &(*iterDbgClosed)->m_UserState;
			}
			return NULL;
		}
		// Get the number of steps
		int GetStepCount() { return m_Steps; }
	private: // methods
		// This is called when a search fails or is cancelled to free all used
		// memory
		void FreeAllNodes()
		{
			// iterate open list and delete all nodes
			vector< Node * >::iterator iterOpen = m_OpenList.begin();
			while( iterOpen != m_OpenList.end() )
			{
				Node *n = (*iterOpen);
				FreeNode( n );
				iterOpen ++;
			}
			m_OpenList.clear();
			// iterate closed list and delete unused nodes
			vector< Node * >::iterator iterClosed;
			for( iterClosed = m_ClosedList.begin(); iterClosed != m_ClosedList.end();
	iterClosed ++ )
			{
				Node *n = (*iterClosed);
				FreeNode( n );
			}
			m_ClosedList.clear();
		}

	// This call is made by the search class when the search ends. A lot of nodes may be
	// created that are still present when the search ends. They will be deleted by this
	// routine once the search ends
		void FreeUnusedNodes()
		{
			// iterate open list and delete unused nodes
			vector< Node * >::iterator iterOpen = m_OpenList.begin();
			while( iterOpen != m_OpenList.end() )
			{
				Node *n = (*iterOpen);
				if( !n->child )
				{
					FreeNode( n );
					n = NULL;
				}
				iterOpen ++;
			}
			m_OpenList.clear();
			// iterate closed list and delete unused nodes
			vector< Node * >::iterator iterClosed;
			for( iterClosed = m_ClosedList.begin(); iterClosed != m_ClosedList.end();
iterClosed ++ )
			{
				Node *n = (*iterClosed);
				if( !n->child )
				{
					FreeNode( n );
					n = NULL;
				}
			}
			m_ClosedList.clear();
		}

		// Node memory management
		Node *AllocateNode()
		{
#if !USE_FSA_MEMORY
			Node *p = new Node;
			return p;
#else
			Node *address = m_FixedSizeAllocator.alloc();
			if( !address )
			{
				return NULL;
			}
			m_AllocateNodeCount ++;
			Node *p = new (address) Node;
			return p;
#endif
		}

		void FreeNode( Node *node )
		{
			m_FreeNodeCount ++;
#if !USE_FSA_MEMORY
			delete node;
#else
			m_FixedSizeAllocator.free( node );
#endif
		}
private: // data
		// Heap (simple vector but used as a heap, cf. Steve Rabin's game gems article)
		vector< Node *> m_OpenList;
		// Closed list is a vector.
		vector< Node * > m_ClosedList;
```

```
        // Successors is a vector filled out by the user each type successors to a node
        // are generated
        vector< Node * > m_Successors;
        // State
        unsigned int m_State;
        // Counts steps
        int m_Steps;
        // Start and goal state pointers
        Node *m_Start;
        Node *m_Goal;
        Node *m_CurrentSolutionNode;
        // Memory
        FixedSizeAllocator<Node> m_FixedSizeAllocator;
        //Debug : need to keep these two iterators around
        // for the user Dbg functions
        vector< Node * >::iterator iterDbgOpen;
        vector< Node * >::iterator iterDbgClosed;
        // debugging : count memory allocation and free's
        int m_AllocateNodeCount;
        int m_FreeNodeCount;
        bool m_CancelRequest;
};
```

Appendix G

G Optimal Local Positions calculated from Receiver Positions

Using 1SM as propagation model and MMC method to determine Receiver Position

Receiver Location		Optimal Local Position		Receiver Location		Optimal Local Position	
X	Y	X	Y	X	Y	X	Y
0	0	no move necessary		3	0	1	0
0	1	no move necessary		3	1	1	0
0	2	0	1	3	2	1	0
0	3	0	1	3	3	0	0
0	4	0	1	3	4	0	1
0	5	0	1	3	5	0	1
0	6	0	9	3	6	0	9
0	7	0	9	3	7	0	10
0	8	0	9	3	8	1	10
0	9	no move necessary		3	9	1	10
0	10	no move necessary		3	10	1	10
0	11	no move necessary		3	11	1	10
0	12	0	11	3	12	1	10
0	13	0	11	3	13	0	10
0	14	0	11	3	14	0	11
1	0	no move necessary		4	0	1	0
1	1	0	0	4	1	1	0
1	2	0	1	4	2	1	0
1	3	0	1	4	3	1	0
1	4	0	1	4	4	0	0
1	5	0	1	4	5	0	1
1	6	0	9	4	6	0	10
1	7	0	9	4	7	1	10
1	8	0	9	4	8	1	10
1	9	0	10	4	9	1	10
1	10	no move necessary		4	10	1	10
1	11	0	10	4	11	1	10
1	12	0	11	4	12	1	10
1	13	0	11	4	13	1	10
1	14	0	11	4	14	0	10
2	0	1	0	5	0	7	0
2	1	1	0	5	1	7	0
2	2	0	0	5	2	7	0
2	3	0	1	5	3	8	0
2	4	0	1	5	4	8	1
2	5	0	1	5	5	8	1
2	6	0	9	5	6	1	10

2	7	0	9	5	7	1	10
2	8	0	10	5	8	1	10
2	9	1	10	5	9	1	10
2	10	1	10	5	10	1	10
2	11	1	10	5	11	1	10
2	12	0	10	5	12	1	10
2	13	0	11	5	13	1	10
2	14	0	11	5	14	1	10

Receiver Location		Optimal Local Position		Receiver Location		Optimal Local Position	
X	Y	X	Y	X	Y	X	Y
6	0	no move necessary		9	0	no move necessary	
6	1	no move necessary		9	1	no move necessary	
6	2	7	1	9	2	no move necessary	
6	3	7	2	9	3	8	2
6	4	8	2	9	4	8	2
6	5	8	2	9	5	8	2
6	6	8	2	9	6	8	2
6	7	2	10	9	7	8	2
6	8	2	10	9	8	8	2
6	9	2	10	9	9	8	2
6	10	2	10	9	10	2	10
6	11	2	10	9	11	2	10
6	12	2	10	9	12	2	10
6	13	2	10	9	13	2	10
6	14	2	10	9	14	2	10
7	0	no move necessary		10	0	no move necessary	
7	1	no move necessary		10	1	no move necessary	
7	2	no move necessary		10	2	9	1
7	3	8	2	10	3	9	2
7	4	8	2	10	4	8	2
7	5	8	2	10	5	8	2
7	6	8	2	10	6	8	2
7	7	8	2	10	7	8	2
7	8	2	10	10	8	8	2
7	9	2	10	10	9	8	2
7	10	2	10	10	10	8	2
7	11	2	10	10	11	2	10
7	12	2	10	10	12	2	10
7	13	2	10	10	13	2	10
7	14	2	10	10	14	2	10
8	0	no move necessary		11	0	10	0
8	1	no move necessary		11	1	10	0
8	2	no move necessary		11	2	10	1
8	3	8	2	11	3	9	1

8	4	8	2	11	4	9	2
8	5	8	2	11	5	8	2
8	6	8	2	11	6	8	2
8	7	8	2	11	7	8	2
8	8	8	2	11	8	8	2
8	9	2	10	11	9	8	2
8	10	2	10	11	10	8	2
8	11	2	10	11	11	8	2
8	12	2	10	11	12	2	10
8	13	2	10	11	13	2	10
8	14	2	10	11	14	2	10

Receiver Location		**Optimal Local Position**	
X	**Y**	**X**	**Y**
12	0	10	0
12	1	10	0
12	2	10	0
12	3	10	1
12	4	9	1
12	5	9	2
12	6	8	2
12	7	8	2
12	8	8	2
12	9	8	2
12	10	8	2
12	11	8	2
12	12	8	2
12	13	2	10
12	14	2	10

Appendix H

H Local Optimal Position for Indoor Wireless Devices

The following paper has been accepted for the WPMC conference in Italy in Sept 2004. The paper is subject to review by the board of the conference and therefore may differ slightly to the paper that appears in the conference proceedings.