Implementing Location Based Reasoning into the Software of the AIBO Robot, Using Existing 802.11 Technology as the Source of Direction and Location

Dominic Hugh Jones

Dissertation submitted to
the University Of Dublin,
in partial fulfilment of the requirements
for the degree of
M.Sc. in Computer Science

Department of Computer Science,
University of Dublin, Trinity College

September 2006

Declaration

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed: _____

Dominic Hugh Jones

11th September 2006

Permission to lend and/or copy

I agree that Trinity College Library may lend or copy this dissertation upon request.

Signed:_____

Dominic Hugh Jones

11th September 2006

Acknowledgements

I would like to thank greatly all the staff of Manchester Metropolitan University for their support and guidance through my undergraduate degree, and for their inspiration and desire for me to study at M.Sc. level. I also give great thanks to the Computer Science Department of Trinity College which has pushed me outside of my comfort zone but brought me to the cutting edge of Computer Science.

Many thanks to Meriel Huggard as project supervisor. She has not only provided the inspiration for the project but provided the much needed support and direction throughout. I'm also eternally grateful for the help received from Professor Peter Stone and Dr Nate Kohl from the University of Texas at Austin. Their work from the AIBO World Cup has been of great benefit.

Vinny Reynolds, member of the DSG group Trinity College aided me on the path from a Java to C++ programmer and his help is really appreciated. Special thanks also go to Prof . Padraig Cunningham who's inspiration led to the implementation of K-Nearest Neighbour as the standard way to implement location estimation, which increase consistency, reliability and accuracy.

Both my parents and my Fiancée must not be forgotten for their continual belief in myself and my studies, they are all amazing But most importantly Desmond Kelly the Security guard has been a friend, a companion, a moral booster and provided a welcome break from the computer screen.

Abstract

This report investigates whether 802.11b can be used in conjunction with the Sony AIBO ERS-7 to create an indoor GPS like system in which the AIBO can self position, navigate and locate itself within the modern home. Using detailed signal analysis combined with position recording the project aimed in the first section to create a detailed map of signal values from which the AIBO could estimate which zone within the room, it was placed. Once established, movement was integrated into the AIBO allowing for self navigation from zone to zone. Additional experimentation showed the AIBO navigating from placement within the test area, to the point in the room with the strongest signal value. Thus creating an AIBO which could find the wireless Access Point (AP), or any other position with a unique set of wireless characteristics.

The AIBO implements a variant of the K-Nearest Neighbour Algorithm to estimate and move from area to area, having two sets of training data from which the estimation is calculated. The second experiment showed how the AIBO was basing movements upon a decreasing or increasing signal value, with no pre-recorded training data, thus showing how navigation to a specific set of co-ordinates was possible.

Table of Contents

Illustration Index

Index of Tables

1 Introduction

The project aims to deduce the location of an AIBO robotic dog using a standard single Wireless Access Point which complies with ISO 802.11b. [1] comments how easily this can be done using satellites and typical GPS type systems yet is difficult to replicate this indoors without the use of "pusedolittes" The project in effect is aiming to work with wifi to create a pusedolitte network, based on wireless access signals, within a defined area, at a low cost and in a way which is easily configurable.

The AIBO used in the project is a ARS-7 running a CPU with a clock speed of 576 MHz based around 64 MB RAM and a removable Magic Gate Pro memory card in which both the Operating System and the user defined Programs are stored, these programs are based on Open – R which is variant of C++. This allows enough processing power to be able to cope with the task of location reasoning, positioning and movement. The AIBOs potential computing power has led to a number of groups working towards design and the implementation of various programs for use on the AIBO. It has been shown in [3] and the RoboCup competition that this dynamic localization can be achieved on a football team scale, however this uses image recognition and position tracking as the main source of navigation, not 802.11.

The AIBO is designed around a Dog, both looking and operating using the same characteristics as a Dog. However Dogs can navigate based on the known areas which they have learnt through association. Two areas of a internal room will be defined using relative signal strengths. After this point the AIBO can monitor the strength of the access point and move around the room relocating itself dependent upon on the signal strength. Success in the project will show the AIBO navigating between two areas of a room, based solely on the wireless signal received and analysed by the AIBO.

There are currently two ways in which location is determined using wifi signals. Firstly the robot can pass back to a central server the encountered wifi signals therefore allowing the server to calculate the robots position and return the result to the mobile device. This may be beneficial when the usage of this technique is over a large scale using multiple low powered devices, as shown in [8]

Another technique involves the mobile devices both receiving the signals and therefore being able to process the location. Triangulation is the normal and most reliable method through which location can be established using wireless signals. However this is not possible using the AIBO. The reasons why will be discussed later in the project, the AIBO in this project navigates based on a single Access Point (AP.)

The human ability to know its way around and associate parts of a room with different actions is pivotal to life, hence the importance of guide dogs to a blind person. What this project aims to do is not report back where an AIBO is or where it is aiming to go but to enable the AIBO to establish where it is and where it needs to go through the incremental and decremental effect movement has upon measured signal strength.

The project initially seemed to deal with wireless communications and network properties yet as the project developed it developed into a mixture of both the above and Artificial Intelligence (AI) adding to the reality of the AIBO and another aspect of functionality relating to wifi networks.

It can be argued that the AIBO is a perfect basis for the proof of point. Can basic equipment be used to guide a robot and provide a machine with human characteristics.

2 Initial Experiment Ideas

It became clear early in the project's development that the standardization of the layout and installation of the system would be paramount if the system was to provide accurate localisation. One question which was raised early in the project related to how to best link the AIBO to a wireless infrastructure. Setting in relation to the $2000 cost of an AIBO a budget for equipment of $100 would be justified. The AIBO uses 802.11b, the most common implementation in Wireless devices. It will be linked up to a signal access point, which will not be moved, this can be argued to be in-line with current 802.11 set-up where once installed an AP is typically not moved and not switched off.

For the implementation and testing of the project what is deemed important is that a standard and defined test area is created for use in the project. For this reason a computer lab which is not being used during the projects developmental period and a single AP, will be combined to create a wireless network where wireless signals are relatively static allowing for locational measurements to be calculated and recorded allowing the AIBO to navigate from two defined zones within the room.

The Access Points will use no security initially as they will not be connected to any of the college's infrastructure. This is the same as when most home Wireless Networks are set-up, leaving these open to all, yet in the set-up of the experiment this is of no concern. After successful demonstration using an insecure network the WEP access protocol will be introduced (this is the only security protocol the AIBO supports.) And the other security specifications as outlined in section 3.9 will be implemented. Like Blue tooth pairing, the AIBO and the access points will create a secure link. This will allow future integration into a home wireless network.

During the Initial stages of development how best to set up the test area needed to be clearly defined. It was clear that the initial design of any configuration may change, need tweaking and develop into a more detailed methodology.
As the research included in the project's state of the art section grew, and my knowledge of the AIBOs operation developed, many a brainstorming session occurred. It was necessary to incorporate the limitations of the AIBO with the project's key aim.

Initially the room was flooded with the signal from three AP operating with a unique SSID and on a less used wifi channel; a wifi signal strength indicator was used to show how the wireless signal varies in different sections of the room. It is possible that any type of home this is installed in will already have a wireless access point installed for general Internet usage, as in the test environment in college. Hence the separation of the AIBO's AP using a less used channel, the default from most manufactures is channel 6, there being 12 in total.

The AIBO can be seen as a novelty toy or fully working robot, really depending upon the user and their desired usage of the AIBO. Using the shop bought access points allows for a number of additional aims to be added to the project. Simplicity in installation, ease of configuration, robustness in design and clarity in operation. It became clear as both the requirements of the project and aims for the user were established that a very clever and detailed technical specification should take no time to install and should be easy for most users to do.

This is an area which as the project develops will be sure to become more difficult to manage. Getting the locational awareness to work is a major target, but getting the robot to move in relation to the signal strength and the variants will be more of a challenge yet pivotal to the successful completion of the project. However the AIBO RoboCup will be used as an excellent source of reference for the project where the AIBOs are challenged to win a game of robotic football. How the project's findings could be integrated in a RoboCup game will then be evaluation.

There are a number of freeware wifi analysis software applications on the market and these will be used within the Windows XP environment to monitor and measure the signals from each AP within the test area. Initial usage of the software showed the main measurement field to be strength measured in -db. This is measured over a scale from 0 to -100, 0 being the strongest signal and the lowest audible sound in relation to the human ear, and -100 being the worst.

Windows XP provides a Very Low....Excellent scale upon which its wireless access strength is measured, this will be an categorization of a signal in db related to a scale. Using the Net Stumbler when the hardware is purchased each AP can be set up in the test bed and its various signal strengths noted at various positions across the room. It will be possible to simulate how the AIBO will locate places within the room based upon the various signal strengths seen from each AP at a specific point.

3 State of the Art

This sections looks at a number of papers and research projects that are relevant to the project's main aim and were deemed to be of help to the projects development. These papers were reviewed in the very beginning of the project so a broad range of understanding regarding wireless location was accumulated. This then helped greatly when the AIBO project started to link the AIBO with the wireless signals and therefore create an electronic map of the test area.

3.1 Robot Localization using Wifi signal without Intensity Map [6]:

This Spanish written paper cites an NDS dissertation [7] from 2004 as its main source of reference. The paper uses computer simulation to predicate wireless signal propagation within an area of wifi coverage. This data is then used by a simulated robot to calculate the estimated environmental position. The paper argues this level of estimation is as accurate as the supervised machine learning techniques as used in other wireless measurement methodologies. They use a predictor as to average signal strength and signal degradation over an area increasing in two square meters.

Like the Net Stumbler software which allows for detailed analysis of a signal strength the group proposes using the wireless network card as the wifi signal for use in the calculation of position based upon sensor readings, assessing the numbers of visible APs, signal and energy levels as key characteristics for measurement. As proposed in this project, two methods were tested. One used an a priori compiled energy map and the other used a map of theoretical WiFi propagation. Using the test area of a computer simulator allowed them to predict signal degradation over an area as opposed to actually measuring readings over a known coverage area. The robot's world was split into 2x2 meter cells.

The paper argues as others do that the map building phase is tedious and time consuming. The average error level achieved in both the propagation and energy

level models is no lower than 1.08m, as is shown to be achieved later in this project. To have the AIBO navigate from one area to another and be within 1 meter of the target zone, using one AP, would show accuracy which could only be increased with the introduction of more APs.

3.2 Peer-to-Peer Determination of Proximity Using Wireless Data [1]

By removing the requirement for the positioning of a device and adding the proximity to other devices the paper argues that in some applications proximity can be as useful as locational information. As the name suggests, this method works upon each mobile peer, monitoring Wireless Signal strength, noise level, and the MAC address of base stations. This information is then transmitted using UDP over the WLAN thus allowing comparison of data to lead to evaluation of proximity.

The implementation of this theory revolves around four main filters. The first being the mobile devices being on the same LAN/WLAN and the second comparing the set of seen APs by each mobile device. If these are a perfect match relative proximity can be deduced. They finally measure the distance in the radio space of the prospective characteristics. Using the “so called Manhattan distance” which can be seen as “the sum of the gap in terms of signal strength and noise level both measured in db for each channel.” Each mobile agent transmits packets containing a string containing perceived APs, signal strength, noise level and peer identity.

The most interesting development this paper introduces to the project is the possible deduction by an AIBO of its relative position in relation to other wifi enabled devices. The paper mapped the proximity to three levels, close, far, and nearby. Could an agent on a user's PC send a signal which when the AIBO needs to be recharged allows the AIBO to navigate to the electrical and request recharging? This would add another level of realism to the AIBO - the ability to find!

3.3 Sensor-Assisted WiFi indoor Location System [2]

This very detailed paper acknowledges the benefits of wifi based indoor location systems as both cost-effective and accurate. However as previously mentioned in [2] they identify instability in positioning accuracy due to changes in environmental factors.

The paper summarizes very clearly the two steps involved in the configuration of a mobile agent to locate where it is from a known electronic map. A human finding themselves in a new environment will have difficulty locating directions without the use of some sort of map or instructions. This is identified as the first stage in WiFi based location awareness, "the off-line phase." This is done by a human operator conducting a survey of the proposed movable area and measuring the Received Signal Strength Indicator (RSSI) of various signals. In the sense of the field of AI this is seen as the "machine learning phase." These signals and variations are recorded onto a Radio Map, the machine version of a road map, where streets and areas are seen as APs and signal strengths. Phase two involves calculation of the remote agents position based upon the seen electrical signals compared to the previously recorded variables. This is described as the "on line estimation phase."

Ekahau [8] is described on its site as "the worlds leading wifi location technology company." They say "the Ekahau system requires 80RSSI samples to be taken every 3 meters to attain average accuracy of 3 meters in a 1000m2 environment.

A considerable portion of the paper deals with the environmental issues relating to wifi signal distribution and proposes a solution to these problems. The paper suggests and supports with experiments the effect that a human has upon a wireless signal. After reading this an experiment was set up using the NetMON software and a colleague. The laptop was set-up in line of site to the AP and the RSSI measured and averaged. My colleague jumped in between the AP and the Laptop and created movement therefore decreasing the wireless strength and

proving the point that humans affect signals. The paper quotes open and closed doors having the same effect on wireless signals as a change in the layout of the room creating an obstacle on which signals can bounce and be deflected. This is of course can be deemed a problem in an environment where cross room tracking is an aim.

This project will initially be aimed at working in a single room and working in the best possible environment. Once this has proved to be successful the integration of the known environmental factors affecting RSSI will be slowly reintroduced. 2.4GHz, the publicly available frequency at which 802.11 operates also happens to be “the resonant frequency of water”. Thus meaning an enviroment with a high relative humidity (RH) is an environment where 802.11 signals find it harder to travel. This would be a possible problem in more humid places than Dublin and more difficult to test for without travel, European climates tending to be a more dry than Dublin's humid weather. However with the very warm summer which Ireland observed this came to be a problem.

These three factors are studied in great detail in the paper and, with time permitting, will be studied further in respect to their effect upon the AIBO and its ability to deduct location.

3.4 Centimetre-Accuracy Navigation Using GPS-Like Pseudolites [9]

There seems to have been a previous drive for a replacement GPS like system for use indoors. Developed before the cost-efficient and widespread introduction of 802.11 access points and coverage. A reader of this article, published in 2001, may well have felt that by today's date the coverage and usage of Pseudolites would be widespread and pivotal in the use of providing location awareness to mobile devices.

What seems to have happened is that the field has changed direction from that of using GPS simulation to using standard 802.11 for detection and discovery of

mobile devices. However the results of this study at the Seoul Nation University GPS lab, as the name of the paper suggests, gained an extremely high level of accuracy. The paper suggest that if Pseudolites are used "GPS navigation is possible in indoor environments" the question that begs to be asked is at what cost can this be provided. The whole aim of WiFi triangulation is to provide location awareness based upon existing and cheap to supply signals. All wifi triangulation needs is the provision of signals, no access or connectivity to the source and those signals is needed nor required to perform location based reasoning.

The Pseudolites use an algorithmic methodology using carrier phase differential GPS (CDGPS). This is very similar to outdoor CDGPS. The aim with the AIBO project and the other projects referenced in this paper is not to create a GPS-Like environment indoors, but to harness the usefulness of the 802.11 signal. Commercial Pseudolites would allow for the AIBO to perform self navigation. However what this project aims to do is allow for accuracy of a meter using off the shelf 802.11 APs.

3.5 Robot Localization using wireless networks [10]

A precursor to [6] this paper describes in detail the way in which a robot can locate itself based on wifi signals. The paper describes two main approaches to robot localization: global localization relates to the position of the robot in its electronic world, where it is and where it wishes to be. The paper quotes this as being important for a robot that is currently unaware where it is in relation to the global map. Built into and directly related to this is "local localization" or as I like to call it "Local Locality Localization." This is the human equivalent of knowing where you are at a specific point, but not knowing where you need to go. Its based around keeping track of movement and location estimation, aiming to predicate where you may end up. Like in sailing, direction and speed can be used to calculate movement. So there are two very different ways in which a robot can predicate movement. 1: Know where everything else is and how you need to get from A->B based upon reasoned position estimation. The second, which surely

must be related back to the first to provide more accuracy, relates to keeping track of where you have been and where you may therefore be going.

In comparison to some of the other papers reviewed in this section, the paper takes into account three obstructions to radio wave propagation, not previously specifically mentioned. Reflection, diffraction and diffusion. Refraction is described as an "electromagnetic wave impinging on an object which is larger than the wave's wavelength." This can be seen as a problem as it both can reduce the power of the reflected signal and can also cause the signal to take a very different path to one which had been previously predicted. Diffraction is described as "an electromagnetic wave which is obstructed by a surface with irregular edges."This is described as allowing for a signal to take a path other than one of line of sight (LoS) which is a benefit to standard wifi coverage patterns but a hindrance when trying to calculate position based upon signal strength from point A -> B. Finally diffusion is described as being "obstruction to an electromagnetic wave by an object with dimensions smaller than the signals wave length." As with Reflection this can lead to the multiple propagation paths.

The combined effect of the three above mentioned factors, not taking into account those mentioned in [1], lead to the "Multipath" problem. The paper describes this as the RF signal taking a different path from Point to Point, So as one works using wifi in an office environment, even with a fixed workstation position and fixed AP, the signal may take various paths from the wireless network card to the AP. How that is countered and its effects on wifi triangulation is a massive area of research to which a solution has still to be found.

The project uses a robot physically connected to a Laptop and wireless card which is remotely controlled. What is unique about the AIBO project is that the robot will be completely independent from any external data or control source. Once finished the robot, when manually configured, will be able to navigate and locate data sources with reliance placed only upon the installed wireless access points. Another very different point is the area in which the robot will navigate.

This paper conducted its experiments in the computer science corridors in a square shape, whereas the AIBOs home will be a square room. Inter room movement is an advanced area of the AIBO project, which was not finally implemented but referenced in this reports future work section.

The paper critiques three different techniques which will be briefly reviewed here. Approach 1, possibly the most common technique implemented, uses a map built from measurements incorporating number of APs, Connected AP, quality, and, signal error. The robot computes the difference between the known four factors and the current four signals. Using the previously compiled Map this allows the location to be calculated based upon the values stored previously in the map. This method can be seen as a calculation based on what the robot has seen in the previous "learning phase" and what is being seen at a present point.

The second method uses signal level as the measure for distance from the source, the AP, to the destination, the WLAN card. The team connected data from only one of the APs at a time. One AP is connected to and the measurement of the signal and noise level was measured at intervals of two meters. Using the sliding scale of signal and noise error over distance, this allowed the group to calculate position. The group found this method to be inaccurate as the change in Signal and noise levels were too insignificant to justify any assessment as to position.

The third and final approach is based around the simulated theory discussed in [6] where the predicted signal strength is arrived at using calculations based upon simulated wireless propagation. As discussed earlier the level of accuracy gained by using this type of method is so low that it is not feasible for the AIBO project.

The optimistic tri-movement approach as discussed in the next section of this report will take the best of the three above approaches and some new ideas to create a hybrid way in which the location can be calculated using the same hardware as above.

3.6 Measuring the wireless performance of an AIBO [13]

The fellow NDS group spent a period of time investigating the wireless connectivity of the AIBO and found some Interesting results. Having never really used the AIBOs I was interested in how the group, who were in the same position as my self, found working with the AIBO. Their main aim was to assess TCP packet transmission between the AIBO and a laptop PC using the 802.11b protocol.

They investigated the AIBO's wireless connectivity through two architectures; firstly they investigated the AIBO connecting to an AP and the laptop connecting to the AP using an Ethernet cable, in a simulated client server model. In a simulated client to client model they investigated data transmission in an Ad-HOC wireless network created between the AIBO and the laptop. The group used the Ethereal software package as their main network management and monitoring aide. The AIBO is capable of sending and receiving TCP and UDP packets. The group focused solely on the TCP protocol and specifically looked at packet loss, latency and thus effective transfer rates.

They used the Sony Open-R software development environment in which they modified some of the program templates so that the AIBO, acting as a server, would accepts files and be able to send files. With reference to the group's paper, the implementation of at least basic AIBO wireless connectivity became more clear. A modified program would be used to enable the AIBO to connect to and monitor various wireless characteristics, saving them to the external memory stick allowing for the later analysis of these results on a remote computer. This step, once the hardware was purchased, would be the first level of connectivity between the two agents.

Additionally the group wrote software for both the AIBO and the laptop which sent a received 4KB buffers over whichever connection was active. The experiments were conducted in the two architectures as mentioned above. These allowed good comparison for the group's work.

For relevance, the group's results working in the client -> server architecture will be reviewed here as this is the architecture in which the current project will be designed and tested. With the AIBO connecting to the laptop through the wireless router and the average data transfer speed of around 500KB/s, there was a relatively high drop in data transfer speeds with an increase in the AIBO -> router distance. However this was much less affected than when in ad-hoc mode. The group also discovered that the AIBO seemed not to be greatly affected by the introducing of a wall in between the router and the AIBO. However what type of wall was used is vital, plasterboard will have less of an effect than brick.

This paper is supporting in the sense that it shows how potential wireless connectivity built in to the AIBO can be harnessed to allow typical network tasks to be performed. The project is now at the point where the APs can be purchased, a test area can be found, the room have areas marked off to show the operate areas and then the wireless characteristics of those areas be measured. Once done the project will start to investigate wireless interactions between the AIBO and a single AP. Firstly a meeting with the previous AIBO will be arranged, to discuss the project further.

3.7 Place Lab [4]

Place Lab uses as many readable APs as possible thereby assuring that the highest level of accuracy is achieved. The machine learning phase involves the documenting of APs signal strength at various locations and the relation of these APs to relative landmarks, or in the case of PlaceLab, GPS position. The proposed system implemented in this project would work around a single AP structure within a set area, and it would be interesting to see whether the project can work using a single access point. If the accuracy of the one hotspot could be assured to +/- 2% then initial predictions show that the system would work from the located AP away from the AP in length and would not be able to calculate the relative width and length of a remote item.

PlaceLab works in a different yet slightly similar way to the proposed system. Placelab collects access points SSID (the Security ID assigned to each AP) and the GPS location at which point a specific SSID is seen. Once added to a centrally managed database the location of a wifi enabled device can be calculated using the received signal strength of a number of access points, known to the PlaceLab project. [4] shows the level of accuracy achieved using this method even in the continually changing environment of a moving car in the metropolitan environment of Dublin City Centre.

The PlaceLab method for location awareness is not dissimilar to the method proposed here. The pre-usage stage allows for the collection of wifi access points data and has this data's association with positions taken from the internal position readings, as and when an AP is seen. The main difference with the Placelab software is that the wifi strength is used to return GPS co-ordinates aimed to provide a wireless GPS type coverage in a city area. Seattle has a coverage rate of around 98% and allows for positioning to be collected using monitored wifi signals only. The PlaceLab system, being open source, allows for modification and analysis of the locational calculations which will be of great benefit to the project. However these calculations are performed on a laptop not a reduced capacity robot in which both location has to be calculated and movement implemented.

There is a company Ekahau [8] which markets tractable devices which can be attached to a valuable piece of machinery and tracked as they move across a building using a wireless network. The AIBO project aims to show how the robot can use known references and its ability to move to find places in a set area.

3.8 Optimistic tri-movement approach (OTMA)

As the initial stages of the project developed and the research increased, it was easy to come to the conclusion that the methods in place for location awareness are not as efficient or easy to implement as one would have initially thought. Ekahau [8] does offer commercial applications of wireless location awareness, but this is at a cost, both financially and in terms of man-hours [2]. As well as being closed source, Ekahau allows for only purchased equipment to be used in relation to wireless location, not off the shelf 802.11 access points and wireless cards.

The whole point of this project is to show location awareness at the lowest cost and with tools which most modern homes own. Many of the solutions proposed by others and reviewed in the Literature survey take the approach of creating an environment where the robot can compare data collected to a pre-compiled map which allows for location to be calculated based upon the compared results of current and past data. What the OTMA approach aims to do is provide a mobile agent with a set of values from which a known location can be stored and remembered for later navigation. Take two values and you have two data sets to compare current and desired co-ordinates. Where the OTMA method differs from other methods is that it uses and stores only the desired co-ordinates for positioning. These are used to move from where ever in the wireless coverage the agent is, to where the agent desires to be.

This can be thought of as a movement and effect relationship. Using two co-ordinates any movement in any direction will have an effect and change all three current values. What OTMA enables the agent to do is, through trial and error, is establish movements which increase the similarity between desired and current values. The agent makes incremental movements. If a movement increases, the similarity between desired and current values it will be furthered; if it decreases the value the movement will be adapted and the new direction tested.

A modern Jet plane using auto land at a capable airstrip uses a similar technique. The perfect line of approach is know by the Flight Computer, the plane makes movements in all directions to establish itself on the correct approach. No movement, with variable environmental conditions, can be guaranteed, so movements with a negative effect upon position are compensated with movements of a positive effect. The Flight Computer is more efficient at this than the pilot, in most cases, and hence when a pilot makes a manual landing the "bounce" effect can be felt by the passengers. Slight left, right, up and down movements are initialized till successful landing. Every movement until then is aimed to establish a correct course.

What effect can this have upon robotics? Imagine the AIBO seeing a set of wireless characteristics as a landing strip. The OTMA expects the mobile agent to move and assess what effect a movement has upon desired position. By making incremental movements the AIBO will be currently assessing what effect the last movement had upon the current position and whether that movement brought the AIBO closer to the desired location. What the OTMA enables the project to do is to allow the wireless characteristics of a specific location to be measured and monitored for a period of time. Once an average set of data has been collected it can be used to allow movement with an area of coverage. All the AIBO needs to know is where it needs to be and then it will allow movement from where the AIBO is to where it needs to be. How successful this is will be judged in its implementation and evaluation.

The OTMA approach was, in the end, not implemented in the initial experiments, but a variation of the OTMA was implemented in the "finding the AP" experiment, which is discussed in section 5.

3.9 Security Review

The AIBO uses IEEE 802.11b wireless to communicate with either a based station or in an ad-hoc environment. The built in wireless card complies with WEP 64 (40bits) and WEP 128 (104bits.) The AIBO is capable of UDP and TCP transmission. The Trinity College, Computer Science Network uses PEAP to connect users to the main departmental network. It was an important decision from the offset that the project would be run separately from any connection to the Trinity network.

The project's design aims to enable the user to run the AIBO using standard Wifi hotspots, not specifically connected to any external network. The safest way to run a computer is to never connect to or open any files. This may be possible, and a main selling point of the final system is that the user doesn't need an Internet connection to initially use the product, yet the security integrated must be enough to assure safety when used with an AP connected to the Internet.

The Laptop used to connect to the AIBO through the router is configured to connect to the TCD network. A major security consideration was what effect would connecting up to the AIBO network have upon the Integrity of the Laptop. As much security as possible should be applied to the AIBO allowing as much confidence as possible to be applied to the integrity of the system as a whole. The aim here is not to fix every problem, but to reduce the chance of there being a problem in the first place. No system is 100 % safe: how can one try to make the AIBO project as safe as possible?

Some threats specific to the use of a wireless network as outlined in [12] include:

1: Malicious entities may gain unauthorized access to a computer network through their wireless network, potentially bypassing any firewall protections.

The number of open Wireless Access Points that provide no security and the number of laptops with enabled ad-hoc wireless networking, unsecured, is a large proportion of the wireless solutions in circulation. Ad-hoc networking on the laptop being used for the project is disabled at the moment so that no external users can automatically connect to the network and the laptop only connects to known networks. This will however change when connecting the AIBO to the router from which the AIBO will be controlled.

There are two main methods by which this security threat can be lessened. Firstly, the Laptop will run an Internal Firewall configured to allow connections only from the AIBO and from no other IP address. Secondly, the wireless router used has MAC ID filtering enabled allowing only the unique MAC ID of the AIBO and the controlling to be allowed connectivity. There are MAC address cracking / masking tools out available but this adds one layer of Authentication.

2: Sensitive information that is not encrypted (or is encrypted with poor cryptographic techniques) and that is transmitted between two wireless devices may be intercepted and disclosed.

Using 104 bit WEP decreases the problems outlined in point 1. There are however some simple ways in which the security of WEP can be increased and known attacks avoided as mentioned in [13]. Using non dictionary words for the creation of the Hexadecimal WEP key prevents dictionary attack in which attacker uses long list of words, converted into Hexadecimal format to try to break the system. The limitations of the AIBO in its wireless communications may therefore limit the implementation of a Encrypted Key Exchange such as DE-EKE or PAK. The main step involved in the creation of secure keys will be the inclusion of 20 random lower and upper case letters with combined numbers and punctuation marks aiming to create an as secure as possible encryption key.

There are also issues identified in [13] regarding WEP only supporting per packet encryption. Given a known packet it is possible to recover the RC4 stream, al-

lowing spoofing of packets. This is the context of the data being sent to the AIBO and is not a great concern to the Integrity of the network.

3: Viruses or other malicious code may corrupt data on a wireless device and be subsequently introduced to a wired network connection.

The AIBO itself has no built in Anti-Virus software. There are only two ways to get data onto the AIBO. One is through the smart media card inserted into the robot, the second is to transfer files to the device across network medium, the FTP transfer requires specialist software to be loaded onto the AIBO, which in the case of the project is not loaded. The AIBO however is unable to execute any of the traditional forms of Windows Virus attacks. With only 150 thousand of them being sold there has been no real desire within the Virus producing communities to try to pose an attack upon a robotic dog.

The main threat within any system is the chance of a Virus being introduced to the PC used through the project. Being a college laptop the Symatec suite of anti virus software is used and combined with regular Windows updates should pose no relevant threat to the system or that of the Trinity network.

4: Internal attacks may be possible via ad-hoc transmissions.

This is a very valid point and at times the Laptop will possibly be open to attacks from other users within a range of the wireless device. It has therefore been decided, in the aim of increasing security as much as possible, to run the following: the AIBO will communicate with the Wireless HotSpot running 104bit WEP. The router will have all new passwords set and WEP keys will be created using the methods as mention above. The Laptop used to connect to the AIBOS and the only possible way into Trinity Colleges network, will have its wireless card disabled before going anywhere near the AIBO.

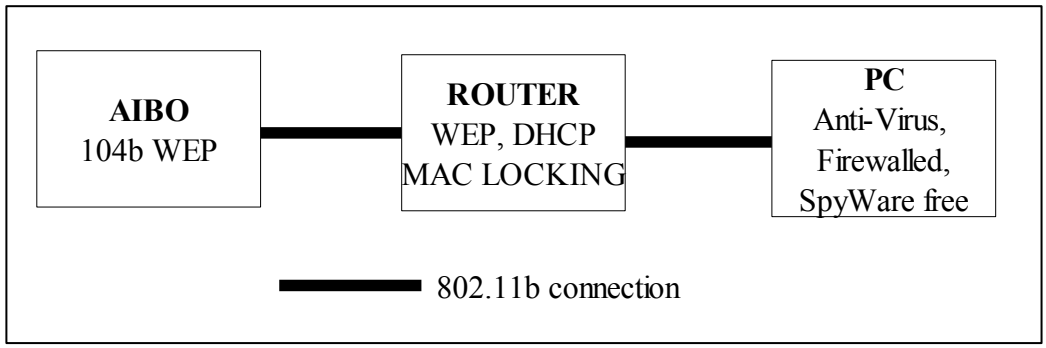

Illustration 1: AIBO Security Set-up

Illustration 1 shows the limits the point of entry to the network from both the PC and the AIBO to just the PC. By securing the Router as much as possible, it ensures that the network behind the router, in this case just the single Laptop, is kept as secure as possible. The possibilities of an internal attack is dramatically reduced. In the final version of the system the Laptop will be removed completely and the AIBO will operate with a single hot spot... The above set-up allows a semi-ad-hoc network to be created without the security limitations of a typical ad-hoc network. Having looked specifically at what effect the set-up of the hardware will have upon the systems security, a more broad overview of the system using the Confidentiality Integrity and Authentication (CIA) methodology, will allow further aspects of the security in place to be identified. It is important to remember that any entry point to the system is a possible security threat.

Confidentiality

The easiest way to quickly identify what level of confidentiality is required is answered by the question "What does the data contain, who would want to see it?" It is an easy assumption to make in the case of this project that the data is not important and that any possible viewing by a third party will introduce no immediate threat to anything.

Integrity

The integrity of the system is a much more relevant and important aspect to the project's security. The assurance that the implementation of the project will have

no effect, whatsoever, upon the Integrity of the Underlying network is one which I have kept in the forefront of my mind. Included in this is the expectation that any data passed between the AIBO and the Laptop is from the AIBO and is meant for the Laptop.

Authorisation

By linking securely the AIBO and the PC through the AP and securing, as much as possible, the AP, I will be creating a relatively de-militarized zone allowing cross communication to occur whilst limiting possible intrusion. As mentioned above the AIBO and wireless hotspot will be configured with 104bit WEP, based on a password both alpha-numerical, lower and upper case and including punctuation. The other aspects of security in place on both the Laptop and the router , as mentioned above, allow the creation of a relatively secure zone.

This security review was carried out before any implementation or connectivity with the AIBO was established. In assessing the security of the project before its actual implementation the relative security issues relevant in the project have been identified and prepared for. The project can now be completed knowing that there are six layers of security in place between the AIBO and the Trinity Network. A user is never able to be assured that any named system is 100% secure. What a user can be 100% assured of is that as many of the possible threats posed to the system have been assessed and the possible solutions to those threats proposed.

It is advised that any future programmer uses the AIBO WLAN Manager 2 software package to edit the wireless configuration file on the MIND Memory card, then copy the configuration over to the user programmable memory stick. In doing this you are assured that the configuration conforms to the Open R standard. Also its is advised that the AP is configured using 128bit WEP, Disable DHCP (assign manual IP address to all clients), MAC filtering enabled and ICMP Ping disabled. Assigning IP addresses to clients allows easier connectivity between the PC and AIBO through telnet, as the AIBO has a static IP address.

4 Design and Implementation

4.1 Initial Design Plan

The first step, without stopping to think, was to purchase three wireless APs. After discussion with the project's supervisor it was established in the initial stages that the localization would be best suited using three identical APS. Each identical AP will map a slightly different area of coverage, thus identical hardware will allow in the initial stages of the project for standardization across the project.

For this reason I purchased three 802.11g Belkin Cable routers. Using 2.4Ghz the router allows for up to 32 wireless clients and four wired. The routers use 1.9cm aerial and operate in both b and g coverage modes. None of the three APs will be connected to the Internet, so they will be used with a Dell Latitude D400 containing a Dell TrueMobile 1300 WLAN Mini PCI card.

A randomly selected Password was selected to assure that only users associated with the project can change any of the settings in the router. Each of the three APs was assigned an SSID of Ap1, Ap2 and Ap3. Each was assigned a channel ID of 1,2 or 3. The channel ID of Ap1 had to changed to 5 as there was a conflict with the Computer Science Wireless AP. Each of these was shown to be working and shown to be unique using the Net Stumbler software. The AP used is a standard low cost, high performance AP which are similar to those installed in many homes and businesses.

In discussion with the project supervisor it was agreed that there needed to be a room in which the project could be conducted and in which the experiment once set-up could be left. For this reason a computer lab which was not being used over the summer was allocated to the project. This allowed for two areas of the room to be defined and the AP, once installed left to operate in an environment

that was least likely to change. The detailed description of the laboratory and its set-up is included later in this report.

4.2 Single Access Point as opposed to Multiple

Fellow TCD student, Gilles Reant, [18] W/BO project main aim was to “create a program that would allow the AIBO to find the access point to which it is connected (using a gradient of quality of signal), and move towards it until it is close enough”– Wi Fi over AIBO states that it is only possible to connect the AIBO to a single predefined AP. The group of fellow NDS students also confirmed this as being true. The problem can be seen in Linux Distributions, the WPA_Supplicant application takes a text file defined by the user and connects to the defined network. This can be supplemented by the installation of third party wifi management software which allows for the user to scan for APs, pivotal in the process of “War Driving.”[4]

Initially the project was based on the predicted ability of the AIBO to monitor all networks and based upon the data revived from the scan, plan its movement and interaction within the environment. What needed to be created for use on the AIBO is a wireless utility which allows for the user to scan all available wireless bands and store the results or act upon the results. There is a well known utility for Windows known as Net Stumbler as used for the signal monitoring in this project. What this allows the user to do is view all available APs and the signal characteristics of the specific AP. Gilles was trying to achieve this and deemed it to be impossible, which after several hours researching I believe to be true, with the current software available.

Initially for there to be any chance of achieving the project aims there needed to be a way to monitor the seen APs and for these results to be able to be recorded. Easily done on all other operating systems and implemented in many areas, but yet to be done with the AIBO. This had been pointed out to me in meeting with the project supervisor. However it was not completely clear until a day was spent working with the AIBOs and the APs. I at least now own enough APs to create a

wide covering home network. I will also leave one of the APs with the Computer Science Dept. so that any future AIBO/Wireless work can be conducted using the same hardware infrastructure as used in this project.

4.3 Programming environment

Initially the main aim of the project was to monitor and record all network traffic. The sample programs as used with the AIBO are coded in C++ and compiled using gcc running under cygwin, being a Linux Simulator. C++ Builder Version 6 was used for my Undergraduate Final year project and for that reason the project was designed using Microsoft Visual Studio.NET 2003. The initial application which when run simply searches for all APs and records the basic characteristics of all them that are within the area was edited in this way.

4.4 Coding and running code on the AIBO

The AIBO development environment allows the user to develop applications allowing for the complete operation of a specific task. In the case of the project, it could allow for the monitoring of a specific AP's statistics and then the acting upon those results, combined with other operational tasks. When the AIBO is switched on it could monitor an AP's signal strength and then move to where a pre-defined set of variable matches the current variables.

There are two runnable options for use with the AIBO, the basic one allows for the AIBO to perform its task, like flashing all its lights, independently from any wireless partner. The second mode allows an operator to Telnet into the AIBO and view output over the wireless connection. The simplest version of this is the Hello World application, which was modified to say “Hello world from Dominic”. This application when loaded onto the program memory stick and run on the AIBO sends out to a telnet client the phrase identified above.

There is a wealth of Information as to how the AIBO works, which all seem to teach nothing. The four most important are the Model Information for the ERS-7, the Level Two Reference Guide for the ERS-7, the Programmers Guide and the

Internet Protocol V4. All of which have been produced by Sony for use with the AIBO. One major issue which may seriously hold the project back is the ability of a network card to work in Monitor Mode which allows a wireless card to capture packets without associating with a AP or ad-hoc network. Most, but not all, Access cards allow this function, however there are a few that do not. However with the work of Gilles, in hindsight it seems that the AIBO can only work with one AP and therefore the project will focus on using a single AP where an ideal operation would be based around multiple APs.

It seems since Sony discontinued the AIBO the number of available applications and the support for these applications has diminished. There is a sample program down loadable from Sony which reports Statistics from the connected network to a telnet session. As a precursor to a final implementation, the file WLANCONF allows for connection to an AP upon start up. This allows for the AIBO to be configured to access a single AP.

It has been discussed with the NDSAIBO group how they found it frustrating that only one program can be executed on the AIBO at one time unless using the MIND O/S. The Open R / C++ code is compiled into a binary file. The AIBO memory stick can contain multiple .bin files, it is however OBJECT.CFG which tells the AIBO which of the user compiled Binary files to load. This is how to execute multiple separate binary s at once on a single AIBO.

4.5 Sample program ERD201DInfo

With the AIBO come various sample software programs, all of which are C++ / Open R code which can be edited and re-compiled. As an initial step in the right direction I opened communication with the members of the Computer Science Dept. University of Texas at Austin. They advised me of a program relating to the AIBO which monitors wireless signal strength, noise level, packet collision counts (etc) which is the code Gilles URBI code [18] is based upon. With reference to the wireless statistics this code allows for the AIBO to monitor and make judgements on the wireless statistics received by accessing the code base.

The ERD201DInfo software, when loaded, reports to a telnet session the wireless characteristics of the current authenticated AP. EtherDriverGetMACAddressMsg, EtherDriverGetStatisticsMsg, EtherDriverGetWLANSettingsMsg, EtherDriver-GetWLANStatisticsMsg, are all Open -R methods for managing and monitoring the wireless link and, it seems at this point, the pivotal functions that will enable this project to be successful. They act as the bridge between the AIBO's wireless card and the outside world. A variable msg is related to the function calls. Using the msg packet the link, signal and noise values, among other things, can be extracted.

This allows for the monitoring of the wireless environment and can be used as a way in which the AIBO can calculate predicted position. Gilles worked using the URBI scripting language for his projects wireless monitoring, he found there to be a drop in the values monitored using the code in his project. It seems from initial test that the WLAN software allows for the continued monitoring of a signal, It can be argued that adding another layer in the form of the URBI API creates a delay between low level operation and high level scripting. Over the six months of testing there has never been a drop in the AIBO's signal

Initially the ERD201DInfo code was edited so that the code fetched and printed to screen the wireless characteristics in a never ending while (true) loop. This meant that the values were read, printed to screen and again read printed to screen... this showed a continuous varying set of values relating to the AIBOs current position of the AIBO. This basic code creates a basic wireless signal monitor which could be used as the AIBO walks around to monitor in real time the value of a specific AP's signal strength. Therefore the basis of navigation could be created from the AIBO's reading of the wireless statistics. This it was felt was a great first step towards achieving the goal of having a self navigating AIBO.

4.6 URBI and other AIBO coding options

Even with the successful compilation and installation of the wireless monitoring, software on the AIBO and the actuate persistent results created through the use of the software, the project became increasingly frustrating. There are a number of problems associated with all of the Programming solutions provided for use with the AIBO. Gilles [18] used URBI and I can see why. URBI, like RCODE is a scripting language which allows for basic commands to be entered into a simple text file and then run on the AIBO.

URBI is a very simple, effective and easy to use/learn way in which one can control and personalise the AIBO. It is perfect for the home user who wishes to be able to type a command like "robot.walk(0.5);" and see the equivalent instant action. What is not so easy in such a language is more complicated lower level control of the AIBO and its associated functions. This is where I feel that the erroneous results that were shown in Gilles project came from. The code which Gilles used was the ER201D1info program, integrated into URBI.

The successful compilation and installation of the ER201DInfo software on the AIBO was a great achievement, but to fully master and take advantage of the AIBOs features it seems more and more important that complete understanding of the Open R language and the way in which it can be optimised was vital to the success of the project. I feel strongly that if Gilles had used the OpenR language his results would not have had the drop in noise level as his did. The Tekksuto framework again has a wireless monitoring application which Gilles was directed towards by forum members on the OpenR site.

There are three options for development of wireless monitoring tools for use with the AIBO. OpenR, Tekksuto and URBI. URBI has been used by Gilles and been shown to be inconstant, Initial OpenR tests have been shown to be well developed and accurate and Tekksuto provides code very similar to OpenR yet implemented in Tekksuto. It is the choice any prospective programmer makes as to

whether they wish for the easy to use format of one of the scripting languages or to use one of the higher level fully functioning languages.

```
#include "ERA201D1Info.h"...
EtherDriverGetWLANStatisticsMsg wlanStatMsg;
status = ERA201D1_GetWLANStatistics(&wlanStatMsg);
```

Illustration 2: OpenR code

```
#include "ERA201D1Info.h"...
EtherDriverGetWLANStatisticsMsg msg;
EtherStatusstatus = ERA201D1_GetWLANStatistics(&msg);
```

Illustration 3: Tekksuto code

The above illustrations show the Open R code in Illustration 2 and Tekksuto code in Illustration 3, both of which you can are relatively identical with reference to the way in which the wireless statistics are collected.

4.7 Test Area Layout

The room in which the AIBO experiments took place is roughly 12 meters long and 6 meters wide. The room was split into two zones, Area A and Area B with the aim of the project getting the AIBO to move from Area B to Area A straight across the room. The AIBO at full speed can move at quite a pace, for this reason I will be spreading the two areas over the full space. Spreading the AP and the areas which the AIBO associates with a different zone will allow for a more dramatic variation in signal strength and hence in theory make navigation easier.

An area 1.2m sq was marked closest to the AP and called Area A. The AIBO was placed in the centre of Area A facing the AP. I booted up the sample wireless monitoring software and read the reading, the Link value was a steady 50. I then modified the sample program so that rather than the statistics being printed out, the program checked the current value of the position against the known value in area A, being in this case 50. When the value of the link signal equalled 50 the AIBO printed out to the telnet session “AIBO is in area A.” This showed the most basic level of reasoning based upon a single value used as a benchmark for correct placement.

The next task was to mark out an area and name it area B. I specifically made this area further away and in a dissimilar position to Area A. By doing this I was able to show two distinct areas of the room as regards wireless signals. I then took ten measurements from each area using only the Link statistic. The data is shown below:

Link	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
AreaA	45	57	55	40	48	59	57	55	53	57
AreaB	44	31	40	42	30	35	38	49	45	32

Table 1: Basic Readings from Area A & B

It's easy to see that area A is typically in the region of 50-60 and Area B is typically in the region of 30-40. Using these figures and a simple if-else statement the AIBO initially prints out to a telnet session which area it is in. This allowed for a user to be connected to the wireless network in another room and monitor where in the main room the AIBO was, almost like a tracking device. This is however not 100% effective, the ranges shown above are large and this tends for over judging of distances as shown below:

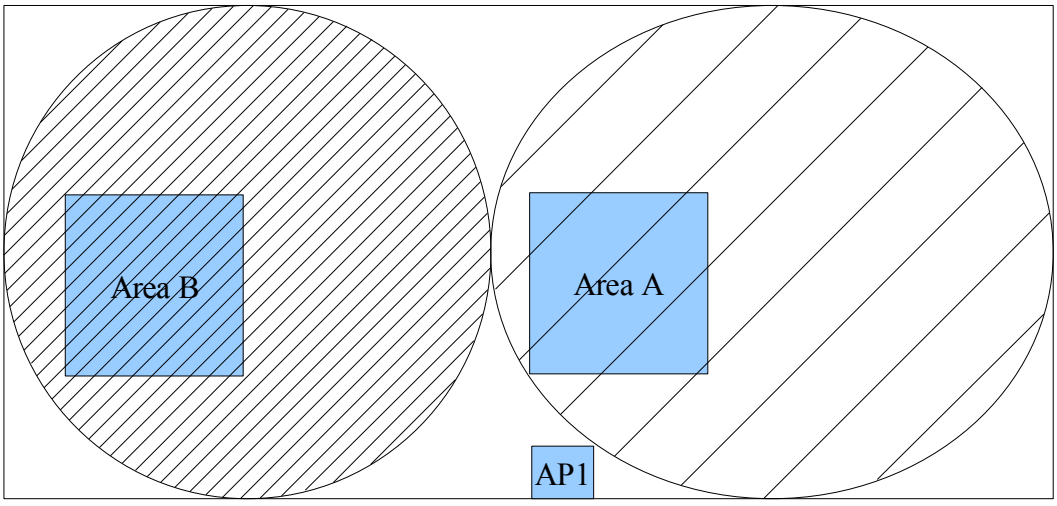

Illustration 4: Initial Wireless Zones

What is shown in Illustration 4 are the areas covered using a basic 10 integer range and created a huge area of coverage and did not represent a valid location estimation. The area in which the AIBO thought it was in is 6-10 times larger

than the actual area! The next step in the project was to look into the refinement of the area's values over a period of time and the inclusion of signal and noise values in the judgement of areas. The set of data shown in Table 2 measures Link, Signal and Noise Value from each of the areas as to electronically identify them. It was thought that the introduction of an extra two sets of data would increase the accuracy of the AIBO is position prediction.

	1	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***10***
Area A										
Link	53	51	53	53	53	53	53	48	51	52
Signal	78	77	78	78	78	79	79	76	76	78
Noise	43	43	43	43	43	43	43	43	43	43
Area B										
Link	36	35	36	36	41	40	43	47	46	34
Signal	68	67	68	68	71	71	73	76	75	67
Noise	44	44	44	44	44	44	44	44	44	44

Table 2: Detailed Link, Signal and Noise Readings from Area A & B

Table 2 shows the second round of data collection in which the Link, Signal and Noise values for both areas were measured ten times. These measurements were taken from ten random positions within the two different zones.

4.8 The Problem with Noise and Detailed Signal Collection

The code in illustration 5 shows how the initial AIBO area was predicted. This noise signal stayed, suspiciously, the same during the initial ten tests within the area. In Area A the figure was 43 and in Area B the figure 44. During the testing I let the AIBO run for 1 min and then paused the software and collected the values for each measurement. This way I could look at any erroneous measurements in comparison to correct values.

```
 ERA201D1Info::PrintWLANStatistics(const  EtherDriverGetWLANStatisticsMsg&
msg)
   {
   if (msg.statistics.link >= 51 && msg.statistics.link <=53 &&
       msg.statistics.signal >=76 && msg.statistics.signal <=79
       && msg.statistics.noise == 43)
       {
       OSYSPRINT(("AIBO is in area A"));
       OSYSPRINT(("  link             : %d\n", msg.statistics.link));
       OSYSPRINT(("  signal           : %d\n", msg.statistics.signal));
       OSYSPRINT(("  noise            : %d\n", msg.statistics.noise));
       }else
       {if (msg.statistics.link >= 36 && msg.statistics.link <= 46 &&
              msg.statistics.signal >=68 && msg.statistics.signal <=76
              && msg.statistics.noise == 44)
       {
       OSYSPRINT(("AIBO is in area B"));
       OSYSPRINT(("  link             : %d\n", msg.statistics.link));
       OSYSPRINT(("  signal           : %d\n", msg.statistics.signal));
       OSYSPRINT(("  noise            : %d\n", msg.statistics.noise));
       }else
       {
       OSYSPRINT(("AIBO IS NOT IN AN AREA"));
       OSYSPRINT(("  link             : %d\n", msg.statistics.link));
              OSYSPRINT(("  signal           : %d\n",
        msg.statistics.signal));
              OSYSPRINT(("  noise            : %d\n",
        msg.statistics.noise));
              }}
```

Illustration 5: Initial Coding

In reference to Gilles work [18], he came across a similar problem relating to the continuous noise value. His URBI script was based upon, and used, the ERA201D1Info program. The URBO code was using the lowest level OpenR with a scripting language added on top. He experienced periods where all the signals would drop and no data would be received and this is, I presume, due to the Interaction between the URBI and Open R code. The code used in this project is

an adapted version of the ERA201D1Info program and gives me direct access to the data without adding a layer of middle ware which passes the data between the Scripting and the lower level programming language language.

The AIBO once booted will continuously produce varying results for Link Signal and Noise Values, However the noise value was static, as referenced by Gilles. Before I used the noise value I wanted to be assured that the noise values reported from the ERA201D1Info program were changing and were valid.

To address the problem of the static noise, I edited the code so that only the noise value was printed to a telnet session. This way I was able to look at 15 different samples of the AIBO's noise, from 15 different places within Oriel House (where the project is being developed.) The places where the samples were taken were completely random and was only to show whether or not the noise signal changes, once the AIBO was booted. The results of these tests are shown in table 3.

Sample	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Noise	46	47	46	46	46	46	46	47	46	46	46	46	46	46	46

Table 3: Noise Tests

The results of the experiment above prove that, as Gilles found, the AIBO does not refresh the noise value. The AIBO only seems to register the noise on its initial boot up. I, as Gilles did, left the AIBO in the same position in sample 15 [the stairs in in-between floors 1-2 Oriel House] and rebooted. The sample changed from 46 to 42. This seems to be a problem to which, with the ceasing of AIBO development, will not be fixed in the near future.

	Round 1		Round 2	
Area	***Link***	***Signal***	***Link***	***Signal***
A1	46	75	47	76
A2	54	80	50	77
A3	57	82	48	76
A4	54	80	51	78
A5	46	75	57	82
A6	52	78	46	75
A7	50	77	53	80
A8	55	81	53	79
A9	45	74	58	83
A10	50	78	43	73
A11	51	78	52	78
A12	50	77	53	80
A13	47	76	52	79
A14	52	78	53	79
A15	55	81	56	81
A16	44	73	58	83
1	56	81	50	77
2	60	84	58	83
3	52	78	59	84
4	52	78	48	76
5	44	73	52	78
6	47	76	44	73
7	48	77	45	74
8	47	76	48	76
9	35	68	47	76
10	40	71	39	70
B1	40	71	45	74
B2	45	74	35	68
B3	42	72	26	68
B4	45	74	40	71
B5	34	68	46	75
B6	35	68	34	67
B7	40	71	43	73
B8	36	68	39	70
B9	35	68	41	71
B10	40	71	39	69
B11	39	70	42	72
B12	41	71	40	71
B13	46	75	41	70
B14	43	73	42	72
B15	30	64	43	73
B16	40	73	32	65

Table 4: Detailed Data Collection

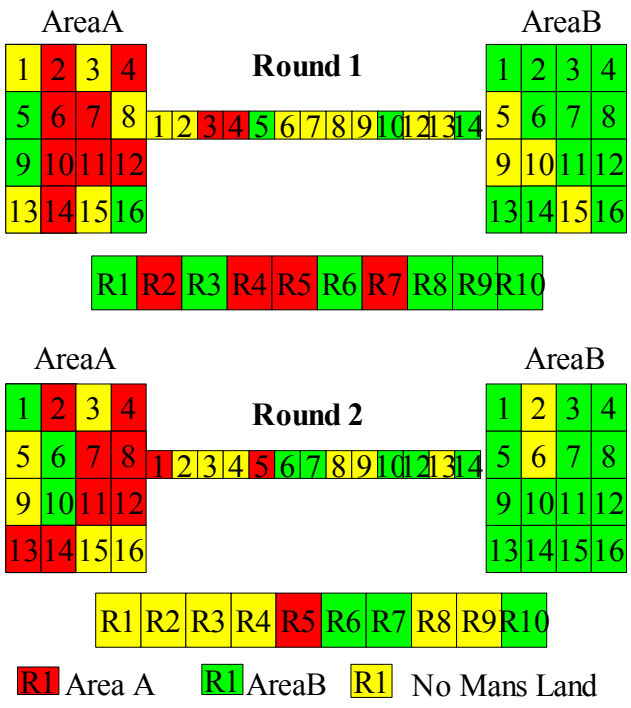

Illustration 6: Initial Area tests

Shown in Illustration 6 is the data collected in Table four in graphical representation. This data was collected from the code shown in Illustration 5, page 32. Both areas, A and B, were split into 16 squares of equal size, and a path between the two area created. Combined with this ten random squares across the room were marked. This allowed for the detailed testing of the signal strength in each squar, in the space in between the two areas and the areas outside A and B. This showed how the AIBO could identify, in the broadest sense area from area. The results were not at all perfect, and deemed that they could be improved.

4.9 Calculating Mean Signal Values

Every time the ER201D1info, EtherDriverGetStatisticsMsg, function is called the AIBO reads the wireless statistics. Up until this point previously the location reasoning has been based on a single reading from the AIBO wireless card. It was concluded that the mean value from a sample of wireless signals would create a more reliable and more valuable signal reading from which location could be deducted. The calculation for the mean value is shown below, in Illustration 7. The value was initially calculated in a spreadsheet and then the theory transferred into C++ code for use on the AIBO.

Mean**LINK**ResultAreaA = (round1AreaAResults) + (round2AreaAResults) / 32

SpreadSheetCalc =(SUM(A1:A15)+SUM(Round2.A1:A16))/32

Illustration 7: Calculating Mean Signal values

The results shown in Table 5 are the mean values taken from the set of data in Table 4, on the previous page. The results show how the epicentre of each area is defined using standard mathematics and was, in the initial stages of the project, used for all navigation.

Both Rounds	AreaA	Path	AreaB
Link	52	39	39
Signal	78	62	70

Table 5: Mean Values for each area

Each area had its mean value calculated from two rounds of data collected shown in table 4, the values were rounded to 1 (d.p.) The next stage was to re-write the locational code within the AIBO to include more accurate location based awareness using the above results and a +/- 5 range. These ranges are shown in table 6, below:

	AreaA	AreaB
Link	47<- 52 ->57	34<- 39 ->44
Signal	73<- 78 ->83	65<- 70 ->75

Table 6: Area Ranges

The above ranges shown in Table 6 were set as the criteria for location awareness. Illustration 8 below shows the results of the Initial testing using the ranges above, accuracy of 80% or higher was set as a benchmark Illustration 8 shows how using the mean values dramatically increases this accuracy when estimation the area in which the AIBO is in, this however is not perfect and could be improved.

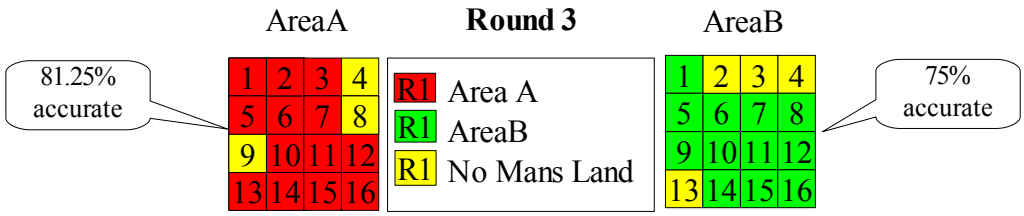

Illustration 8: Third round results

4.10 No Man's Land

For much of the project I decided to have two areas within the room in which the AIBO would be able locate itself. Outside of these two areas I wanted to show the AIBO outside of any area. The aim was initially to teach the AIBO where the two areas were and when the AIBO was not in either area for the AIBO to notify the user of such a position.

Ten areas across the room were created in a random pattern outside of both areas. Again, using the key on the previous page, these areas were tested using the AIBO. Initial predictions were that the AIBO would confuse these areas with either area A or area B as these areas have such a large catchment area. This was proved by the test results, as shown below. Table 7 below also shows the distance from the centre of each random area to the centre of each main area [a & b.] This measurement has been taken as a straight line from point to point. The predicted area was predicted by myself based on the previous tests,

Random area:	***Distance to AreaA (m)***	***Distance to AreaB (m)***	***Predicted Area AreaA / AreaB / ?***	***Link Value***	***Signal Value***
R1	3.35	8.43	**AreaA**	**43**	**71**
R2	2.77	8.13	**AreaA**	**52**	**77**
R3	1.67	6.73	**AreaA**	**44**	**72**
R4	1.79	6.33	**AreaA**	**55**	**79**
R5	1.90	3.86	**AreaA**	**54**	**78**
R6	2.98	3.08	?	**38**	**72**
R7	3.45	2.86	?	**47**	**78**
R8	3.91	1.61	**AreaB**	**35**	**71**
R9	4.85	2.59	**AreaB**	**44**	**76**
R10	6.47	2.32	**AreaB**	**40**	**73**

Table 7: Random area results

The results shown in Table 7 were deemed poor. As a backup I redid both sets of tests to see if the areas A and B still had a high level of accuracy compared and how the random areas were predicted, a second time.

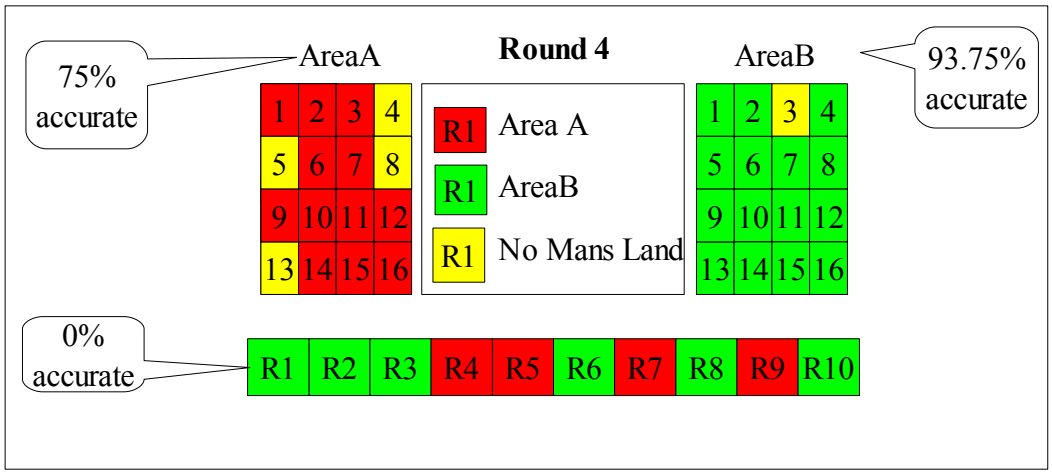

Illustration 9: Area A/B & Random results round 4

The good news was then that the AIBO saw areas A and B with a level of accuracy greater than 75% in both rounds. The bad news was that the AIBO saw the ten random squares as either Area A and B in 20 tests 20 times which is 0% accurate. This was obviously unacceptable. The problem was that any increase in accuracy in the random areas leads to a decrease in accuracy in the main Areas. What has to be decided is what overall level of accuracy is required. Is a level of accuracy at 50% all over better than the above scenario? This is a question which eventually solved using the K Nearest Neighbour section, of this report as described in section 4.13.

4.11 Developing an Understanding of how the AIBO moves

Getting the AIBO to walk in Open R is an extremely difficult task which was neither easy nor quick to complete. However its is a compulsory part of the project and the Wireless signal analysis section was left on the "back burner" whilst a month was dedicated to getting the AIBO to walk. Using a scripting language like URBI allows a command similar to "robot.walk(0.5);" to get the AIBO to perform basic movements, a reason I think why Gilles [18] chose to use URBI. The most pressing question at this point was how to get the AIBO to walk in Open R. As with all other aspects of the AIBO, searching the web for help or tutorials on this, showed no results and lead the searcher in every decreasing circles. I therefore again consulted Dr. Nate Kohl, Department of Computer Sciences, University of Texas at Austin, associated with the UT Austin Villa robot soccer team, I was advised to look at the "MovingLegs7" sample code. This code when run enables all the AIBOs Motors [16 in total] to move the AIBO to a resting position and flashes all the AIBOs lights and moves the head / ears.

Although in theory simple, this code enabled me to base the AIBOs movement in Open R and showed me how to go about movement using the Open R system. Again however a problem that was dealt with in the initial stages of the project was one which asked how to get the AIBO to work with two programs in Parallel. I therefore consulted Vinny Reynolds of TCDs DSG group, having worked on a C++ project with him previously in the year. With his help I was shown how the C++ Make file needed to be amended to include the .h files of any additional classes in the project.

This allowed the AIBO to run the movinglegs7 program and print a simple text message to screen. Although all the wireless code is integrated, the first step was to show how basic code could be integrated with complex code. This was proved by configuring the AIBO when booted up to flash all its lights and move all its joints whilst printing a simple text message to a remote telnet session. This is the basis of the whole project: the integration between the movement and the wireless signal is pivotal in the process. An important aspect here was that the AIBO

was not doing any user configured movement, it was performing the movement coded into the sample code. It was another hurdle to change the code so that the AIBO performed a walking movement.

The next stage of Integration involved running the whole wireless statistic measurement code in parallel with the AIBO, so that the area which was predicted was printed to screen whilst all the AIBO's lights and motors flashed. Nate Kohl also directed me towards the code releases of the AIBO teams from the 2005 AIBO world cup, (yes there is an AIBO world cup!) What is encouraging to know is that all the world cup teams code bases use the Open R language, in effect C++ code showing that the code produced for this project could be easily integrated into existing AIBO world cup code. This proved, to some extent, that the focus of the project solely on the Open R code base was the best choice to make, even when it was unknown as to which of the languages was most used. I believe using Open R allows for a solid and reliable code base to be created, therefore allowing for successful localization based upon a single AP.

At this point a re-assessment of the project's aims was necessary, if only to revisit all the knowledge gained so far. The project's demonstration was visualised as the AIBO being placed in a starting position, the centre of Area B and navigating to Area A. When the AIBO estimates it is in the centre of Area A it will stop, print to a telnet session informing the user it is in the centre of Area A. This, I felt was an achievable and acceptable goal which would have met, in the broadest sense, the initial goals of the project.

The project was in a very secure position as regards completion and achieving the main aims as set out back in November. The two programs MovingLegs7 and the heavily edited ERA201D1info program have been fully integrated and allowed for the final push towards movement based on the wireless signal monitoring. Finding out with regard to AIBO programming that "There is no multithreading. You should adopt a design based on multiple objects to achieve parallelism." **http://tinyurl.com/mcld5** was a devastating blow, the ideal operation of

the code would be a continual loop reading wireless signals and the movement being based upon those signals read. Yet again there was another aspect of the project identified in which the AIBOs limitations had dramatic affects on the projects developments. When compiled all code is combined in a .bin binary file. Each .bin is seen as a separate program and an internal scheduler based within the AIBO alternates between the .bin files. This allows for some basic parallelism but does not allow for dynamic allocation of threads and inter thread operability.
It was therefore necessary to develop a scheduler which in effect was a switch statement within a while (true) loop which allocated each method a chance of running and then switched to the next method. This code allowed for the wireless signal to be read and then the movement of the AIBO to be carried out. Although basic this allows for various tasks to be alternated between and in theory run a very basic parallel operation.

The MovingLegs7 program is actually a collection of seven separate programs which when run create an AIBO whose ears and head move, the Legs reposition and all the LEDs flash. The first stage in the AIBO's movement involved removing all the .bin files that were not needed and editing the config file so only the desired parts of the code operated. Second was the dissection of the Moving Legs program. In the most basic sense each AIBO has four legs each of which has three joints. See the Diagram below for a further explanation:

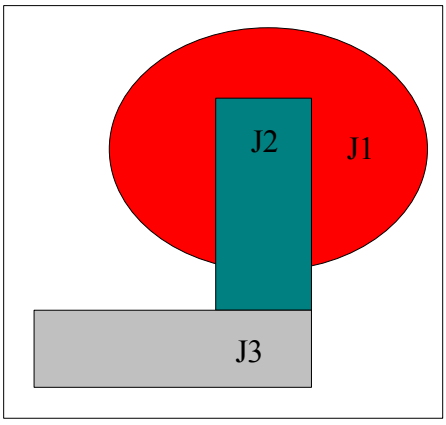

Illustration 10: AIBO Joints

As shown in Illustration 10 there are three joints in each of the AIBOs legs. Each joint can have its values set within a minimum and maximum value range. The values for each joint are stored in an array allowing for the code to loop through assigning a value to each of the joints and simulating movement. The MovingLegs program has two main sets of joint angles, the first moves the AIBO into a stretched out position, the second moved the AIBO into a sleeping position. Sony provides a Motion editor which allows for simulation of the AIBOs position before hard coding it into the program. However this program is very lightweight and often what is shown on the computer screen is very different to what is shown on the actual AIBO. To make the AIBO walk will require 12 variables to move in a pattern know to produce a safe stable walking movement.

There were two main tasks which needed to be completed before the AIBO could walk and move based solely on wireless signals. Firstly a method which when called implements a set of movement variables needed to be developed. This would allow for the AIBO to pass through a range of movements. This can be completed whilst the AIBO is still taking wireless readings. An initial requirement was that the AIBO, when started, walked in a straight line across the room, taking wireless signals and reporting them back to a telnet session.

A large amount of the time was spent developing a detailed understanding of the MovingLegs7 program and the way in which it controls the AIBOs movement. The project initially, when the AIBO is booted, processes four movement requests. The first two MoveToBroadBase and MoveToSleeping were coded by Sony and provided in the sample code. The second two MoveToKneeling and MoveToStanding were coded by myself.

To summarise briefly how the AIBO controls movement: The motor power is set to ON by one of the other MovingLegs7 programs, BlinkingLED7 to be precise. Once this has been set the MovingLegs7 code opens each of the Joints in the AIBO allowing for movement commands to be sent to the AIBO, a Command Vector is created to store the requested movement pattern. One major debugging

issue was that if this command Vector is set to only receive 2 commands if a third is sent to the vector, the program **will** compile, but when run, the "error noise of death" sounds and the AIBO switches of. For this reason the command vector is set to 150 allowing for multiple movements to be implemented. So with the joints enabled and the command vector set up the AIBO moves through a selection of code which sets the values of the AIBO to equal the broad base and then to the sleeping position.

When the set-up code has been established and correctly implemented, the scheduler() is called. This was initially a simple switch which alternates between the wireless statistics program and the two motions designed to get the AIBO standing. This structure changed with the development of the project but initially the AIBO prints the wireless signal to the screen and then calls the InitialiseStanding() method. The states designed by Sony are labelled MLS_SOMESTATE whereas my states are named DHJ_SOMESTATE. Using states is a very clever way to distinguish where the AIBO is and what it is meant to be doing. It would be easy to go into detail in this section as to how this initial build works, but this is going to drastically change. What is important to understand is the following lines of code:

```
1  static double start[NUM_JOINTS];
2  static double end[NUM_JOINTS];
3
4
5  for (int i = 0; i < NUM_JOINTS; i++) {
6     start[i] = SLEEPING_ANGLE[i];
7     end[i] =   KNEELING_ANGLE[i];
8  }
9
10  RCRegion* rgn = FindFreeRegion();
11
12 for (int i = 0; i < NUM_JOINTS; i++) {
13    SetJointValue(rgn, i, start[i], end[i]);
14 }
15
16
17 subject[sbjMove]->SetData(rgn);
18 subject[sbjMove]->NotifyObservers();
```

Illustration 11: Basic movement code

Looking at the code shown in Illustration 11 you can see how the AIBO uses SLEEPING_ANGLE as the known joint position and the KNEELING_ANGLE as the desired position and therefore moves between the two. If the AIBO is prescribed a specific angle and that angle is blocked, meaning it can't therefore be achieved then the AIBO crashes and informs the user in a telnet session that the desired angle is unachievable. The Sony code deals with timing, which for the simplicity of the implementation I removed. Therefore when the AIBO switches between the KNEELING and STANDING angle, it does this at such speed one could blink and miss it. The correct operation of the code is shown using telnet print lines. This, however, is an issue which needed to be resolved and was later in the project.

```
const double SLEEPING_ANGLE[] =
{

    59,    // RFLEG J1
    0,     // RFLEG J2
    30,    // RFLEG J3

    59,    // LFLEG J1
    0,     // LFLEG J2
    30,    // LFLEG J3

    -119,  // RRLEG J1
    4,     // RRLEG J2
    122,   // RRLEG J3

    -119,  // LRLEG J1
    4,     // LRLEG J2
    122    // LRLEG J3
};
```

```
const double KNEELING_ANGLE[] =
{

    29,   // RFLEG J1
    0,    // RFLEG J2
    15,   // RFLEG J3

    29,   // LFLEG J1
    0,    // LFLEG J2
    15,   // LFLEG J3

    -60,  // RRLEG J1
    2,    // RRLEG J2
    61,   // RRLEG J3

    -60,  // LRLEG J1
    2,    // LRLEG J2
    61    // LRLEG J3
};
```

Illustration 12: Sleeping and Keeling Angles

Shown in Illustration 12 are the two sets of angles as would be used by the code in Illustration 11. They are stored in a simple array, yet the AIBO relates each of the positions in the array with a joint, as shown in the green comments. By loading the set of angles to the corresponding joints the AIBO simulates movement.

The project was on schedule and the next main aim was to implement a basic, and possibly crude, walking algorithm integrated with the wireless functions,

showing as the AIBO walks across the room, combining the wireless statistics and active movement. This will allow time for fully implementing movement based upon the wireless signals, finally showing the AIBO moving from area A to area B based solely on the wireless signals received from one AP. Absorbing myself in the project showed me how, implementing the above goal would be a great achievement for me, one of which I at times thought would not have been possible.

4.12 How the AIBO walks

This section was the most testing to achieve and was crucial if the main aims as set out in the dissertation proposal were to be met. Getting the AIBO to walk was one of the many obstacles that had been in my way throughout the project. It was clear that to progress with any other part of the project would require teaching the AIBO to walk, and by the end of this section I will show how it this has been achieved.

The first stage in developing walking was to strip down the MovingLegs7 program so that a greater and deeper understanding of the code could be developed. I removed all of the state transition code and the code which seemed to have little impact on the AIBO. From this point it was easy to see how the joints of the AIBO had their values set, remembering there are three joints on each leg and four legs. To control AIBO movement it is necessary to know where the joint is and where the joint needs to be.

Initially I began trying to map movements using the AIBO motion editor which allows the values of each joint to be set and a simulated model be shown. To make the AIBO walk using this method is a painful and unique project in itself, as the software does not make movement easy. After discussions with both my project supervisor and Dr Ciaran Mc Goldrick, of the TCD Computer Science Dept, it was decided that movement should be measured by hand and recreated in the simulator, to create a crude movement. URBI (Universal Real-time Behaviour Interface), which is what Gilles had previously used allows for a “ro-

bot.walk(0.5);" command to be entered via a Telnet session. It was only after this meeting that I remembered URBI allowed movements to be easily created, yet as previously mentioned in my opinion lacks some of the powerful functionality of C++ and OpenR.

By loading URBI onto one of the AIBO memory sticks and stepping through each movement I could use the "JOINTNAME.val();" function to print to screen the specific value of each joint after each forward movement was called. These were entered into a spreadsheet and after the AIBO had made 12 movements the figures reset themselves and started on the whole cycle again. I now had a set of known joint angles for standard forward movement and the method to change them.

The number of different sets on angles required by URBI to make the AIBO walk equals 12, 12 angles in each set allowing the AIBO take 1 simple step forward means a combined set of 144 angles. This figure shows how painstaking it would be, by hand, to have collected all the data. As with everything else regarding the AIBO there are no open sources where this type of data can be collected or referenced. For this reason after completion of the project I will make these angles publicly available.

12 arrays were created and stored in each of the arrays were a different set of the 12 joint angles needed to implement movement. To simply prove my point that the sets of joint values worked I added another 12 states to the project and in each state got the AIBO to load a set of values in a corresponding array, therefore simulating movement. Each state had a corresponding method which loaded the hard coded angles into the AIBO, this created an additional 12 states and functions, all of which were very similar. When I loaded the values into the AIBO and created a telnet session into the dog, I saw that each function had been called. However the movement was unrecognisable as regards walking. It was only then that I realised that by stripping down the code I had removed a major part of the code, which is explained on the next page.

To move a set of joints you call the following method:

```
MovingResult r = MoveToStanding();
```

When the method has finished its returns a variable r, this being decided by the following ternary switch :

```
return (counter == BROADBASE_MAX_COUNTER) ? MOVING_FINISH : MOVING_CONT;
```

The ternary switch above can be seen in the following traditional if – then – else structure below:

if { (counter == BROA...) **then** (MOVING_FINISHED) **else** (MOVING_CONT)**}**

The counter is incremented each time the function is run and the way the Ready() method works in OpenR is very much like a thread. The BROADBASE_MAX_COUNTER when set high increases the time it takes for the movement to be implemented. The method only returns (r = MOVING_FINISHED) when the counter is equal to the timer therefore by including this code the AIBO was seen to move at a slower pace taking 3072 ms to compete one movement.

At last there was an AIBO which when loaded made a set of twelve movements each taking three seconds and making the AIBO walk one movement. It was the AIBOs first walk, which I had coded, and a monumental occasion.

As part of our M.Sc. major coding project, we worked an eXtreme Programming project in which we learnt to pair-program and continually peer review code. For this reason and for the development of “good quality code” I asked one of my peers, Zef Hemel, an extremely competent programmer, to spend and hour with me and review my code and discuss changes and additions which may add to the quality of the project. We initially looked at the sets of joint values and realised one multidimensional array with all the sets of joint angles within it would be neater and allow a single function to call any of the values in the array.

However, it is important the reader understands how Open R deals with the joint values. The AIBO requires two sets of joint angles for each movement, referred to in the AIBO as the start and delta set, the specific angle joints are stored in arrays and when needed added into a set of new arrays, this can be seen below:

```
for (int i = 0; i < NUM_JOINTS; i++) {
          start[i] = BROADBASE_ANGLE[i];
          delta[i] = (FORWARD[0][i] - start[i]) / ndiv;
      }
```

BroadBase angle is the initial stretch angle which the AIBO performs to assure that joints are in the correct position. All the sets of movements, are stored in a Multidimensional array**.** The above therefore loads the set at position forward[0] and stores the 12 values from this set in the delta array. So in start[] the current position of the AIBOs joints are stored in delta[] the desired next values are stored. Once this has been done the joints need to be moved to the desired angles as shown below:

```
for (int i = 0; i < NUM_JOINTS; i++) {
     SetJointValue(rgn, i, start[i], start[i] + delta[i]);
     start[i] += delta[i];
  }
```

What was needed next was to look at how each of the joints is set in OpenR. This is done using the SetJointValue() function which requires four values to be passed into it, the region, the second is the index or Joint value which is prescribed by the order of values fed into the arrays. In other words a specific position in an array corresponds to a specific joint, the start position, then the start + end position. The value has been divided in the first code segment by ndiv which can be seen as a timer. Varying the value ndiv in the .h file alternates the speed in which the joint moves. So it was now possible to remove the 12 methods which set each joint and replace them with one method for forward motion as shown below:

```
for (int i = 0; i < NUM_JOINTS; i++) {
    start[i] = FORWARD[step][i];
    delta[i] = (FORWARD[step+1][i] - start[i]) / ndiv;
}
```

Combined with this is the new state diagram where the function is called 12 times, each call passes an incremental variable to the one function it can be seen above as step. Remember above start[] is where the AIBOs joints are and delta[] is where they need to be. How the new method works is each time it is called it uses the current position and the current position + 1 in the array to select the data set. The next variable [i] then loads the specific values from the array. It is a much more efficient and well designed way to achieve the same thing. Once the loop reaches 12 it is reset and the wireless statistics function is called, which now locally calculates position and prints this to screen. So one movement and one wireless reading was achieved, the the process started again.

The AIBO was now in a position to walk across the room, on a semi-straight line, whilst pausing and reading the wireless signals. The next step was to code the AIBO so it walked across the room from Area B to Area A and stopped when it reaches the epicentre of Area A, showing location awareness and movement!

4.13 Introduction to K-Nearest Neighbour

There was deemed to be a more reliable way in which location estimation could be calculated, referred to by my postgraduate Artificial Intelligence lecture Prof. Pádraig Cunningham from the Computer Science Dept. Trinity College. Two interesting developments arose from this meeting regarding both the location estimation and the identification of which are the AIBO was in. Before meeting with Pádraig location estimation was based on a three case if else statement using so called "catchment areas" as the identifier for an area. What Pádraig proposed was integrating the K-Nearest Neighbour (K-NN) algorithm in which the AIBO has reference to sets of training date for each area.

Taking on board the teaching from my M.Sc. Artificial Intelligence course I refreshed my knowledge of K-NN and considered how it could be integrated into the project. K-NN allows a set of training data to be integrated into an equation, meaning data in which the results is not known, can have the predicted result estimated using K-NN. In the case of the AIBO there are two areas divided into 16 squares. The initial phase is the training phase, the AIBO was placed in each of the squares and then the average mean reading in a "still" electrically speaking environment was taken. This created a reading for each square in each of the areas and is seen as the training data for both Signal and Link values.

The results collected in the training round are then used as estimates for the actual calculation. In the case of this project the data included in the training data set is the average Signal and Link measurement from the 16 squares in each area. The data is hard coded into 4 arrays, 2 for area A one, for Signal and one for Link, and two for area B, one Signal and one for Link.

When a real time measurement is taken by the AIBO, rather than this being passed through the previous if-then-else statement, the measurements are compared against the two arrays and the nearest number of matches identifies which of the areas the AIBO is in. By doing it this way if the experiment layout was to be changed the code would not have to be changed, just the training set of data.

The initial task was to develop a training set of data and enter this into an array. Once the data was stored in the array the AIBO will compare the real time data from the Network and find the closest match from the four arrays, thus predicting the area in which the AIBO is in. Using an array structure which clearly defines which sections of the array are for which area in the room any student extending this work can enter new values from a new environment yet use the same code.

The current code takes 1000 samples from the AIBO and creates a mean value of the samples. For this reason, when calculating the training set of data, I took ten measurements for both the signal and link values of each sub square of each area. The mean value of these samples is used as the training set. This value is the sum of a 1000 results. So even though the array contains 10 values those ten values are the combined value of 10,000 measurements. The first step was to implement this in a spreadsheet followed by implementation in the AIBO's code base, followed by testing. The very helpful tutorial provided by [Teknomo, K. (2005). K-NN Tut. Which can be seen at http://tinyurl.com/fdobk] allowed me to develop in a spreadsheet the methodology to be used in the C++ program. It also allowed for the results of the AIBO's calculations to be checked against a known reliable source. Although the spreadsheet is copyrighted I requested from its creator permission to change it for use in this non-profit project and received such permission The spreadsheet stores a set of variables and the associated result associated with the variables, allowing prediction of new variables based on known results.

Link	Signal	Result
40	70	A
44	73	A
...	...	...
43	73	B
42	71	B
..	..	..
43	72	B

Table 8: KNN predictions

Table 8 shows a sample of the training data used. Link and Signal values are associated with a result, either A or B. The training data is 32 entries long. However each entry is the mean values from 1000 samples. Every Signal and Link reading is compared again the mean value of 1000 samples. The last row shows, highlighted yellow, unknown values entered into the spreadsheet and the result predicted, in this case truly.

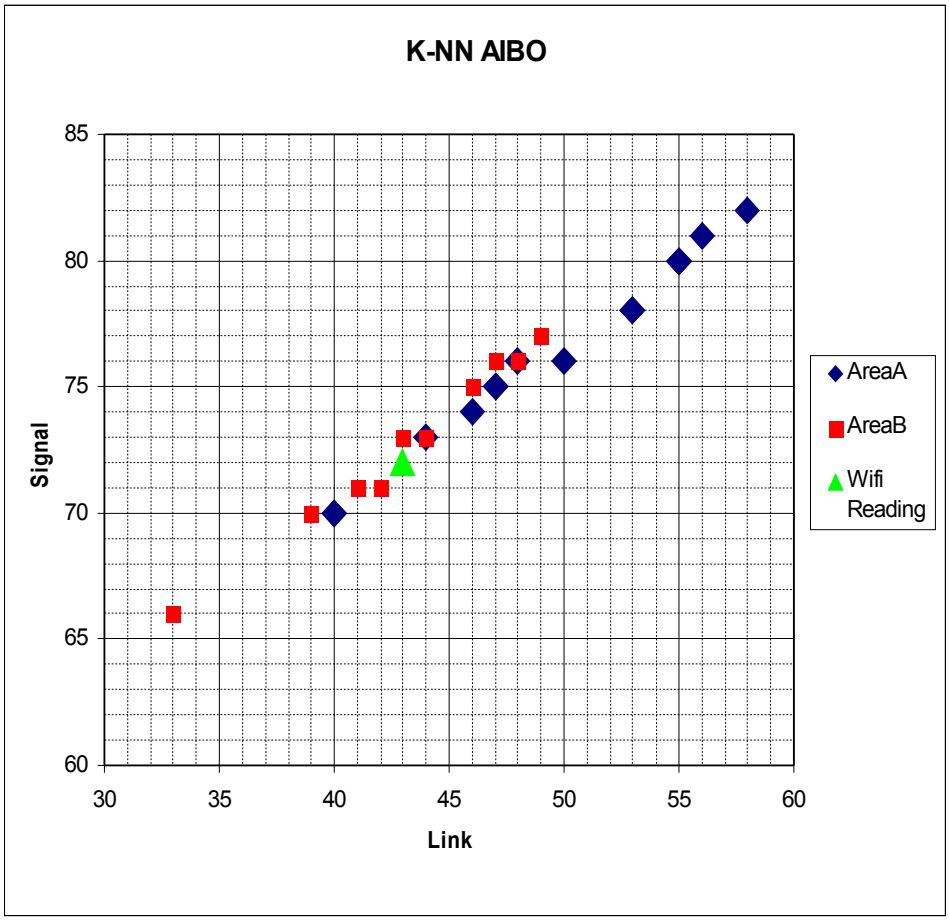

Illustration 13: KNN Predictions based on Training data

Illustration 13 shows the output results from the K-NN calculations, performed in a standard spreadsheet calculation. The training results are represented in the blue diamonds, for Area A training data. The red squares represent Area B training data. The green triangle shows the plot of the (Link value = 43) and the (Signal value = 72.) Just by visual estimation it is easy to see that the result is closest to the Area B training data, which is what the K-NN calculation predicted. The reading is in fact from AreaB and shows how clever K-NN is at predicting the outcome from a training set. Each time the AIBO makes one movement it computes the query data based on the training data and estimates position.

The initial spreadsheet tests show how the KNN method for location awareness provides a massive increase in accuracy. The prediction was that integration into

the code base would provide a higher level of accuracy in general location estimation.

4.14 Steps involved in calculating K-Nearest Neighbour

1: Calculate for each entry in the training data set the Distance from train-data to queried data:

$$Distance = ((LinkTrainingVal - LinkQueryVal)^2) + ((SignalTrainingVal - SignalQueryVal)^2)$$

2: Calculate for each entry whether the distance value is one away from the lowest value in the training set. If this is true set a flag to 1, otherwise set a flag to 0.

$$if\ (Distance \leq setOfTrainingData - 1)\ then\ (KNNSign = 1)\ else\ (KNNSign = 0)$$

3: The 32 training data examples are split into two 16 partitions, one which represents Area A's results and one which represents Area B's results. The Final step is to count which of the two partitions have the most positive flags and assign this value as the predicted value for the query.

$$if\ (Count(TrainingSetA)) > (CountCount(TrainingSetB)) \\ then\ (Result = A)\ else\ (Result = B)$$

Following these three steps allows for the allocation of a result to a queried set of data. The next step was to code this into the C++ / Open R code. During the pseudo codes integration relevant checks were made against the spreadsheet to show correct operation. The AIBO was then left to walk from Area B to Area A and the predicted area printed to a telnet session. This led to there now being no "No mans land" just area A and area B. This will split the room into two electronic zones and allowed for the AIBO to navigate from one zone to another. There is a point half way between the two zones in which the AIBO swaps over from Area B to Area A and again this is shown on the AIBO's return to the other zone. It is not 100% accurate. There are still some erroneous results, but the

gradual fade from one area to another shows location awareness. The next step is to get the AIBO to stop when it reaches the epicentre of an area, this will be discussed in the next sub-section of the report.

4.15 Working out where to stop

Each time an area is estimated using K-NN the AIBO increments an internal counter, as an initial judgement of where the AIBO is. Again after consultation with Pádraig Cunningham, it was decided that when the AIBO saw the integer count for a known area, it could estimate quite accurately that it was in a specific area. Using this analogy when the number of A readings is greater than the number of B readings, the AIBO rationalises that it is actually in area A, the same with Area B. An additional step was added, when the AIBO receives a value for Area B, it decrements the counter for Area A, so only when the AIBO picks up area A signals does it start counting results for Area A.

Although simple the theory had to be tested. The AIBO was left to walk from Area B To Area A. Looking at the data it was clear to see that after five consecutive Area A results had been collected and printed to screen, the AIBO is almost certainly closer to Area A. The Initial round of testing involved ten walks from Area B to Area A. Each time 5 consecutive Area A readings were taken it was marked on the strip running from area A to area B.

It is important to note that these experiments were being conducted in a very hot and humid laboratory. In an ideal world the lab would be air conditioned, allowing for the heat to be gradually increased thus allowing for what effect this has on the AIBO, The AIBO felt physically hot, as everything in the room did. For this reason the first five readings were taken using the AIBO with the MAC address ending 4AF and the last five readings were taken using the AIBO with the MAC address ending 4B2. This allows for a new, fully charged and cool AIBO to be used as well as making sure there were as many testing agents as possible. By using the two AIBOs in rotation I was able to assure that the test agent was a cool as possible.

Once the ten measurements had been made the distance between the furthest and closest readings to Area A were taken and halved. This is the electronic divide between Area A and Area B. All the readings were within 1.07 meters, showing a reasonably accurate estimation to where the dividing line between the two areas is. It is Interesting that the area which is represented by Area B is roughly half the total length of the room meaning the Area A space is again roughly half the space of the room. What also doesn't particularly help is that Area B is slightly raised in comparison to the area between Area B and Area A. Area A is again slightly lowered meaning the AIBO slants after walking across the room. This however can be re-adjusted. Also, the masking tape which marks the areas and the centre strip seems to have an impact on the AIBOs movement. For this reason the AIBO is moved to the edge of the masked areas to start its walk across the room. This initially seems to have little or no effect on the location awareness.

Further testing based on the initial results showed the AIBO to be accurate, but only accurate to within 4-5 meters, I slightly amended the code which is used for location awareness and ran ten walks from Area B to Area A again. Each time I marked the place at which the AIBO stopped, using this method I was able to see exactly where over a spread of ten tests the AIBO thought the epicentre of an area was defined. These tests were extremely accurate, Area A is 1.15 meters by 1.15 meters. In nine of the ten tests the AIBO stopped within the 1.15m width of Area A. In one of the tests the AIBO stopped 80cm outside of Area A.

5 How an AIBO can find an AP

5.1 Introduction and Link to Gilles project

The AIBO project developed into two main areas. The first related to the wireless monitoring of the signals within the room, combined with locational awareness based on an adapted K-NN algorithm. The second established movement based on the wireless signal detected. By combining forward movement with signal analysis the AIBO was able to navigate from A -> B.

There was the desire to develop further movement based on the wireless signal from the Internal wifi card built into the AIBO. To refresh, Gilles Reant's [18] work aimed to create “a program that would allow the AIBO to find the access point to which it is connected (using a gradient of quality of signal), and move towards it until it is close enough” Gilles work, due to time constraints, was limited to coding based on the URBI Scripting language, Gilles found a number of errors with the coding and found movement easily implemented, whereas the wireless signal movement was prone to signal drop outs and false readings.

During the initial stages of the project an interesting discussion with the Project supervisor was whether the problems encountered by Gilles were being noted in the current development. Although many errors were discovered with the AIBO and rectified along the way wireless signal problems were never identified. The reasons for this have been previously discussed in section 4.8, and for this reason will not be discussed again here.

The second aim of the project, introduced in early August, was to repeat Gilles' experiments using the Open R programming language and assess the ability of the AIBO to self navigate from a point within the test area to the access point. Unlike the previous work this experiment would allow for the AP to be placed anywhere within any room and navigate based on the stronger signal to the AP, showing not only location awareness but reasoning based upon the collected

wireless signals. As regards demonstration of the additional work, the main aim was to allow any viewer to select a starting point for the AIBO and then observe the AIBO navigating to the AP, finding its way home.

The current software solution provided with the AIBO allows for the AIBO to self navigate from a position to the base charger when its battery runs low. Both this solution and the current AIBO world-cup solutions, allow for navigation based on image recognition and not on wireless signal strength.

5.2 Programming progress and structure

The first step was to again turn to URBI as regards the movements required for the AIBO to side step and turn 90°, thus allowing the AIBO to rotate in increments of 90°. The aim of using URBI was to collect joint angles allowing for these to be transferred over to the Open R software package. Although this requires the manual recording of each set of angles it would have been a painstaking and time consuming process if manually completed.

The latest build of URBI was complied and stored on a memory stick. Once booted, the AIBO was connected to using the URBI-Lab software, the commands:

```
motor on;
robot.turn(0.5);
JointNAME.val();
```

were activated showing the AIBO being in the process of turning to the right. After each turn had been commanded, each of the 12 joint angles could be collected in one go and entered into a spreadsheet. To turn the AIBO 90° requires 18 sets of 12 angles, meaning a total of 216 different angles are needed to perform a basic turn. However this now meant that a set of 4 right turns returned the AIBO to the position in which it started, and this was to be shown to be pivotal to finding the AP. Whilst the AIBO was in URBI mode I took the measurements for a left turn as well, although these were later to be shown to not be needed.

With the angles needed to make the AIBO turn on the spot coding was required to implement the movement. The initial aim was to have the AIBO make one forward step followed by one right step. This was easily implemented using the existing code base.

What provided a more taxing problem was how to rotate between forward and right movements without the AIBO resting in the idle position. An additional two states were added to the State transitional diagram allowing for a right and a left movement the code was adapted from the forward movement and can be seen below:

```
1 else if (movingLegsState == DHJ_RIGHT) {
2           MovingResult r = Right(stepCounter);
3           if (r == MOVING_FINISH)       {
4                 stepCounter++;
5                 if(stepCounter == 19){
6                       stepCounter = 0;
7                       MovementSched();
                  }}
      }
```

Illustration 14: State Movement Diagram

What is shown in Illustration 14 is the state transition code which is called when the state is changed to DHJ_RIGHT, as shown in Line 1. Line 2 is the call to the Right movement method, which is similar to the forward movement method, except the angles and number of steps to be taken are different. The step counter is passed into the method, this being set by the last state to 0, when its movement was complete. The Step counter is used within the movement code to access the relevant position in the array and incremented when the AIBOs' joints are set to the current values, as stored in the array.

Each time a movement is complete the method returns the variable: if r = Moving Finished as shown on line 3 the Step counter is incremented and again the movement called with the next set of angles. When the Step counter == 19 (the total number of sets of movements requires to turn the AIBO 90°, forward only

requires 12 sets) the step counter is reset and the movement scheduler is called. Previously the next state would have been set in line 7 above, however the aim was to create movement, state setting and wireless reasoning in completely separate sections hence why movement scheduler is called, as seen below:

```
1 void
2 MovingLegs7::MovementSched()
3 {
4
5     int move = er.checkPosition();
6
7     switch (move)
8     {
9     case 0:
10           movingLegsState = DHJ_FORWARD;
11           OSYSPRINT(("FORWARD Called \n"));
12           break;
13     case 1:
14           movingLegsState = DHJ_RIGHT;
15           OSYSPRINT(("Right Called \n"));
16           break;
17    case 2:
18           movingLegsState = DHJ_LEFT;
19           OSYSPRINT(("LEFT Called \n"));
20           break;
21   }
22
23}
```

Illustration 15: Movement Scheduler code

What is important to remember is that the Ready() method within the Open R infrastructure is permanently running. It is like an open thread, so when a change in program state is made in another area of the code the Ready function immediately implements that change. Illustration 15 shows the Movement Scheduler, which is called on line 7 of Illustration 14. The Scheduler immediately calls the code in the program ERD201DInfo method check Position(); as shown on line 5. The most important aspect of the checkPosition() method is that it returns an integer to MovementSched() this integer is stored in the local Integer “move” as shown on line 5. The switch shown from line 7 is based upon this Integer and then the relevant state is set in the case of 0,1,2 this allows for the state, and therefore the prescribed movement to be randomly changed.

The check Position() method initially returned variables looping from 0-2, thus meaning the AIBO walked forward right and then left in a continuous loop, showing how three separate movements could be activated using a remote method. The next stage was to look at how the check Position code would calculate the navigation from the AIBOs placement to the AP.

5.3 Checking position and basing movement on wireless Stats

```
1   int
2 ERA201D1Info::checkPosition()
3 {
4
5
6     collectData();
7     int CurrentLinkVal = divresult_Link.quot;
8     int CurrentSignalVal = divresult_Signal.quot;
9     clearData();
10
11    OSYSPRINT(("CurrentLinkVal = %d\n", CurrentLinkVal));
12    OSYSPRINT(("CurrentSignalVal = %d\n", CurrentSignalVal));
13
14    int reqPos;
15
16    if ((CurrentLinkVal >= PreviousLinkVal)  &&
17    (CurrentSignalVal >= PreviousSignalVal))
18    {
19          Position = 0;//FORWARD
20          OSYSPRINT(("Current reading > than Previous \n"));
21    } else {
22          Position = 1;//RIGHT
23          rightCalled = true;
24          OSYSPRINT(("Current reading < than Previous \n"));
25    }
26
27      PreviousLinkVal = CurrentLinkVal -2;
28      PreviousSignalVal = CurrentSignalVal -2;
29
30
31    return Position;
32 }
```

Illustration 16: Check / Set position

The code in Illustration 16 is called by the Movement Scheduler, when called the method calls collectData() which is the same as the previous code above and calculates the mean value from a number of data samples. Once complete this data is stored in the CurrentLinkVal and CurrentSigVal variables as shown on line 7 & 8. Once called the data sets are cleared in preparation for the next call. The code on line 16 as shown in Illustration 17 is then called.

```
if     ((CurrentLinkVal >= PreviousLinkVal)  &&
       (CurrentSignalVal >= PreviousSignalVal))
```

Illustration 17: Forward or Right code

The above code in Illustration 17 shows how the CurrentLinkVal is compared to the PreviousLinkVal and the CurrentSigVal is compared against the PreviousSignalVal. Both the Previous values are set to zero in the initial running of the code, if the above criteria are met then the Integer Position is set to zero, if they are not met the Integer Position is set to one.

The code on lines 27 and 28 show the PreviousLinkVal and PreviousSignalVal being set as the current values minus 2, the 2 was calculated to allow for some errors or slips in the signal values. Meaning that if after a movement the reading is one or two places less than the last reading. This is seen as a positive, not negative result. This was implemented as sometimes when moving forward towards the AP the AIBO does receive a lesser value which is then increased greatly in the next reading.

Once the previous values have been set the Integer which prescribes the desired movement is retuned to the movementSched() code which in turn sets the state which the ready() method implements. The AIBO compares the new reading with the latest reading and if the AIBO sees a greater value in the latest reading it heads forward. However if the latest reading is less than the previous reading the AIBO turns right and takes another reading. This method allows for the AIBO to navigate towards the strongest signal in the room, which is the AP. Over many tests the AIBO has been observed to walk in a circle and take a path forward when the greatest reading is recived. This allows the AIBO if walking away from the AP to do an about turn and head back towards the AP. The AIBO at times walks in circle after circle but **does, (**unless obstructed by a wall or desk,) find the AP and “kicks the AP,” which is deemed in the case of the project as a success. The detailed tests of this and the Area B to Area A navigation are included in section 7.

6 Statistics and Measurement Calculations

6.1 Standard Deviation

During the initial stages of the project, catchment areas were used to estimate position; before the K-Nearest Neighbour algorithm was implemented. The wireless signal ranges for both areas needed to be re-assessed. This had been on hold whilst the walking has been developed, and was the precursor to developing the K-Nearest Neighbour algorithm implementation.

For the accuracy of the AIBOs position to increase, it was felt necessary to start looking at the wireless signal position estimation and integrating this within the AIBO. The first aspect which was changed was the "Thinking in squares as opposed to circles". The testing which was done previously introduced two square blocks, called Area A and Area B. I also marked 10 random square across the room.

The initial idea was to show that the AIBO could see each of the areas and each of the random squares (outside of either area) as unique, something which could be possible if more than one AP was able to be connected to. The random squares were removed yet the two areas were left and the centre of the room was marked as the split between the two areas.

Initial classification of the original areas were done using paper and pen averaging the variance in signal values. These values were deemed both inaccurate and unscientific. Therefore a set of values was collected and Standard deviation was performed on the data to show the normal distribution and the average drift from the mean of the set. As a refresher, each area (square) was divided into 16 squares, each of them 625cm2. This allowed for the AIBO to be placed into the centre of each individual square and a measurement to be taken. Two round were taken in which each square had both its signal and Link value measured, as you can see below in Tables 9 & 10 below:

Area A		**Area A**	
Link Vale	Average	Signal	Average
50	8.62890625	76	5.0625
45	63.00390625	73	27.5625
55	4.25390625	80	3.0625
61	65.00390625	84	33.0625
52	0.87890625	78	0.0625
50	8.62890625	77	1.5625
53	0.00390625	78	0.0625
59	36.75390625	83	22.5625
48	24.37890625	76	5.0625
55	4.25390625	80	3.0625
53	0.00390625	73	27.5625
56	9.37890625	81	7.5625
38	223.1289063	68	105.0625
55	4.25390625	80	3.0625
59	36.75390625	83	22.5625
58	25.62890625	82	14.0625
Mean Link Value:	**52.9375**	Mean Signal Value:	**78.25**
Standard Deviation:	***5.859109716***	Standard Deviation:	***4.32820209***

Table 9: Area A Signal & Link Standard Deviation

Area B		Area B	
Link	Average	Link	Average
31	74.39063	34	21.97266
40	0.140625	35	13.59766
33	43.89063	42	10.97266
47	54.39063	43	18.59766
39	0.390625	32	44.72266
45	28.89063	42	10.97266
39	0.390625	43	18.59766
41	1.890625	35	13.59766
40	0.140625	40	1.722656
42	5.640625	41	5.347656
34	31.64063	37	2.847656
34	31.64063	41	5.347656
44	19.14063	38	0.472656
47	54.39063	44	28.22266
36	13.14063	30	75.47266
42	5.640625	42	10.97266
Mean Link Value:	**39.625**	Mean Signal Value:	**38.6875**
Standard Deviation:	***4.93795***	Standard Deviation:	***4.34693***

Table 10: Area B Signal and Link Standard Deviation

Tables 9 and 10 show the standard deviation from the mean values of signal and link for both area A and area B respectively. By taking area A's signal results, one can see that the data collected is 5.8 +/- from the mean, being 53. This shows the data is within 4 standard deviation points from the mean.

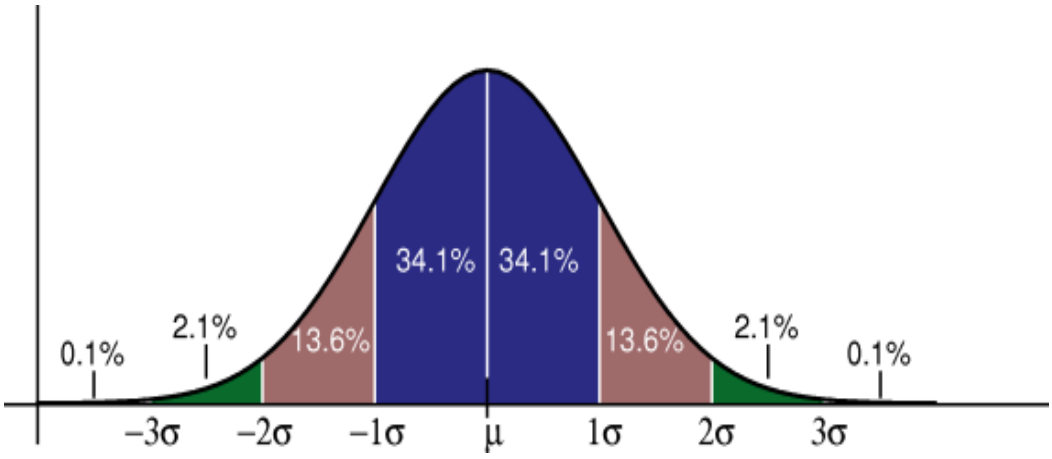

Illustration 18: Normal distribution - GNU license http://tinyurl.com/nrpqm

Illustration 18 states that "Dark blue is less than one standard deviation from the mean. For the normal distribution, this accounts for 68.27% of the set; while two standard deviations from the mean (blue and brown) account for 95.45%; and three standard deviations (blue, brown and green) account for 99.73%." **http://tinyurl.com/nrpqm.**

The mean value is represented by the mu symbol whereas the distance from the mean is represented by the +/- sigma values. Illustration 18 shows the data Area A Link Value, the mean being 53 (1d.p.) The standard deviation for this data set, being 6 (1d.p.), is within 4 deviation points from the mean, the ranges where the data is plotted is shown in the green catchment areas shown on Illustration 18.

Chebyshev rules, taken from **http://tinyurl.com/nrpqm** state that the data is normally distributed within 94% when the mean is within 4 points. This shows that 94% of the values for the Area A Link data set are 4 standard deviations away from the mean. The catchment area would be set to 13% +/- from the mean. K-Nearest Neighbour was later shown to be more accurate than this.

$$(eachResult - AVERAGE(setOfResults))^2$$

$$SumSD = \sum (EachResult)$$

$$\sqrt{(SumSD - (COUNT(setOfResults) - 1))}$$

Illustration 19: Standard Deviation Calculation

The equation shown in Illustration 19 was used to calculate the standard deviation. This calculation was checked against the Spreadsheet STDEV(setOfResults) and was shown to be correct. It was replicated across the two rounds of data collection and in each round for the Link and Signal. The SD for the Link in both rounds was between 5.9 and 7.6. The same for the signal was between 4.3 and 9.5. These major differences can be accredited to two or three erroneous results. For example in the second round of results the first measurement gave a reading of 45, whereas the average reading for that round was 77.8. Its is easy to see how one reading of the scale effects all other results. The next step in the Statistical analysis is to show how these readings fit onto a normal distribution graph, as can be seen below:

The calculations showed correct catchment areas and the spread of the data collected, but did not increase the accuracy of the position estimation. Within the catchment areas, which were initially calculated using pen and paper, standard deviation only proved that the estimated areas were correct.

Once the standard deviation was calculated there were no changes to be made to the catchment areas. This is why further investigation was conducted to establish whether there was a more efficient and accurate way in which location estimation could be calculated. Ultimately, this is why the K-Nearest Neighbour algorithm was used. The calculations do prove that the two areas of the room hold different signal strengths and statistical values, which was seen as necessary for estimation to be correct.

6.2 Are 1000 samples too many or too few?

The AIBO both navigates and calculates position based on the mean value of 1000 Signal and Link value readings. The 1000 was an arbitrary sample count which was chosen to get as broad a sample of results, as possible. The training data is a set of readings each of which is the mean value from 1000 readings... So what effect does taking 1000 readings have on the program? Does this value degrade the performance of the code? Firstly the hardware of the AIBO needs to be explained. The AIBO uses a 64bit RISC Processor MIPS R7000 which has the following performance capabilities:

- Speed rated at 450MFlops , 600MFlops for floating point calculations
- 50MHz external clock, 100MHz internal clock with 10ns cycle time
- Thirty-two 64-Bit registers
- Physical Address Space: 4 GBytes 32-Bit mode / 64 GBytes 64-Bit mode
- Floating Point Unit Single & Double precision operations
- Primary Cache - 16KBytes Instruction and 16KBytes data
- Secondary Cache – 256KBytes

taken from **http://tinyurl.com/o6tj5**

The processor based within the AIBO is by far fast enough to collect 1000 samples, which are integers. There are two aspects to the calculating of the mean Signal and Link Values. Firstly the values are collected, the while loop in which the values are collected does not pause. Once called, it may collect 1000 of the same values or they may be different, it depends on what the values are at that specific collection point.

However it is going to very quickly collect the data set then perform the mean calculation on the values which is the sum of the 1000 values divided by 1000. The RISC processor is a MIPS processor which allows, as the name suggests, for a Million Instructions Per Second (MIPS) to be carried out.
To collect the values, for both Link and Signal, is 2000 instructions, to store the values is another 2000 instructions and to calculated the mean is 2001 instruc-

tions, give or take another 2000 instructions for in-between processing, altogether 8002 instructions. Even if the processor is running at 80% capacity the time it takes to calculate the mean value is minute. What may take more time is the calculation of the KNN algorithm. There is also 64MB of RAM for use in the calculations.

Again there are issues with benchmarking the AIBO as the typical tools available are not able to be run on the AIBO. During Undergraduate study a team on the course investigated the time it took to perform multiple calculations using commercial benchmarking software. This however is unable to be done using the AIBO as these tools are Windows based and as with other aspects of the AIBO there are no other resources available which could perform the same task.

Therefore there seems to be no easy answer to the question which opened this section of the report. Is 1000 samples too many or too few? Looking, in detail, at the hardware specifications of the AIBO it can be argued that the AIBO is more than capable of performing the required calculations without impacting on either the Network or the speed of the other areas of the project.

7 Trial and Evaluation

7.1 Introduction

The test area which has been used throughout the project is a computer lab, which was free over the summer months. The day before extensive testing was carried out on the two experiments all the PCs, 14 in total, were removed from the area, in preparation for upgrades. This actually allows for an interesting question to be answered. Will the removal of 14 PCs and 14 CRT monitors, with their various metallic and magnetic components, have any effect on the way in which the results are shown? As the room is semi-clear for the final tests, all chairs will be removed, creating a completely clear test area. It is predicted, before testing begins that the second test, in which the AIBO finds the AP will not be affected. There may be a small possibility that the training data previously used in the project will need to be re-collected.

When the AIBO tries to find the AP it is simply looking for the strongest signal value in the room. By navigating from the start position to the AP the AIBO disregards known values and is only concerned by the stronger signal. Whether that signal is deflected by machinery already present in the test area is irrelevant, and goes towards the aim that when the AP is placed in another environment the AIBO will still be able to navigate to the strongest signal within the test area.

The best project demonstration is deemed to be an actual demonstration with the observers physically watching the AIBO performing the desired tasks.

For this reason the Trial and Evaluation section of this report cannot be as detailed as other projects, due to the organic nature of location estimation and navigation. Combined with this is the supporting evidence included in section four of this report. There were many test hours carried out, with and without external observers in which the main aims of the project were shown to be in practice through actual observation.

7.2 Area B -> Area A

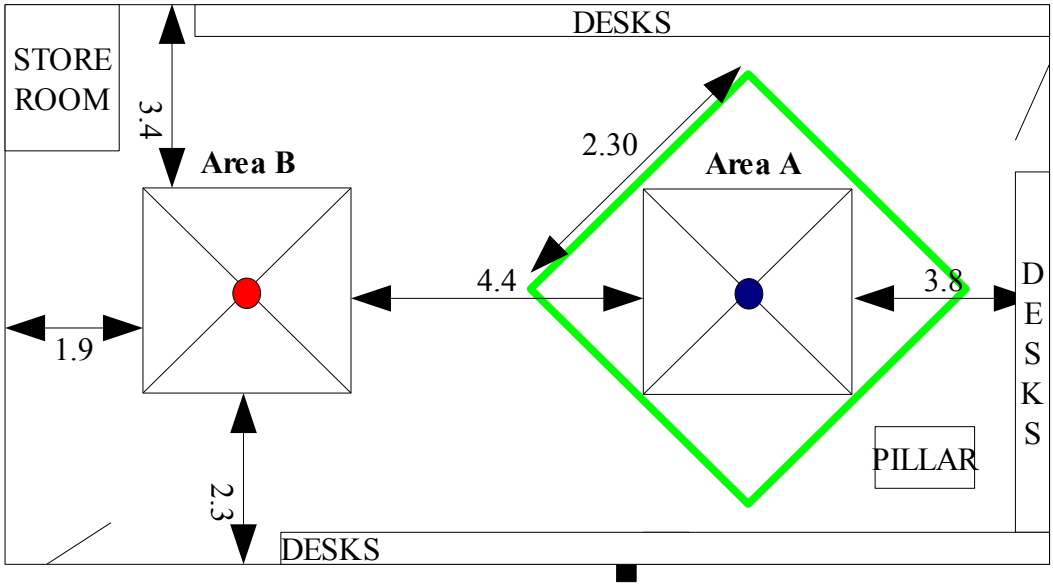

Illustration 20: Room Layout - Experiment 1 (Measurements in meters)

Shown in Illustration 20 is the layout of the test area which was used throughout the project. The two areas A and B are shown in relation to the other furniture and is the exact measured position within the test area. The AP is represented in yellow with the position of the wireless antenna represented in black.

The red dot in the centre of Area B shows the starting position for the Area B -> Area A tests. The AIBO was placed on the red dot with its head facing area A, from this point the AIBO, when booted, navigates from the starting point across the room.

The blue dot in Area A represents the epicentre of Area A, the green diamond represents the desired stopping position. Having started from Area B the aim of the tests were for the AIBO to be at least within the green area and as close as possible to the blue epicentre of Area A.

Ten tests were carried out where the AIBO self navigates from Area B to Area A, the distance from the epicentre of area A was recorded in each test, shown below:

Test #	*Distance from epicentre (m)*
1	+0.14
2	+0.67
3	+0.22
4	-0.64
5	+0.57
6	-0.52
7	+0.38
8	-0.44
9	+0.64
10	-0.22

Table 11: Stopping distance

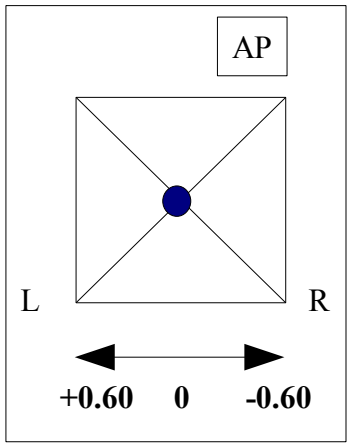

Illustration 21: Epicentre Distance

Illustration 21 shows the way in which the results in Table 11 were calculated. Once the AIBO had stopped walking and identified the epicentre of Area A the distance from the back leg of the AIBO to the epicentre was measured and recorded as either a +/- figure, a positive reading meaning the AIBO had passed the AP and a negative reading meaning the AIBO had yet to meet the epicentre of area A.

The results, as shown in table 11 show that all ten tests were within 1 meter +/- the epicentre and all within the target stopping zone, as represented by the green diamond in Illustration 21.

The AIBO showed, constantly that movement based upon K-Nearest Neighbour estimation creates a level of accuracy which is both acceptable and accurate in location estimation and prediction.Watching output from the AIBO over a telnet session the position estimation is extremely accurate. There is a point in the room 1.5 meters from Area A and 2.78 meters from Area B which is the electronic divide between the two areas. This divide accurately shows the separation of the areas and as the AIBO passes from one area to another. The results are pleasing and show how a level of accuracy has been developed, using standard 802.11.

7.3 Finding the AP

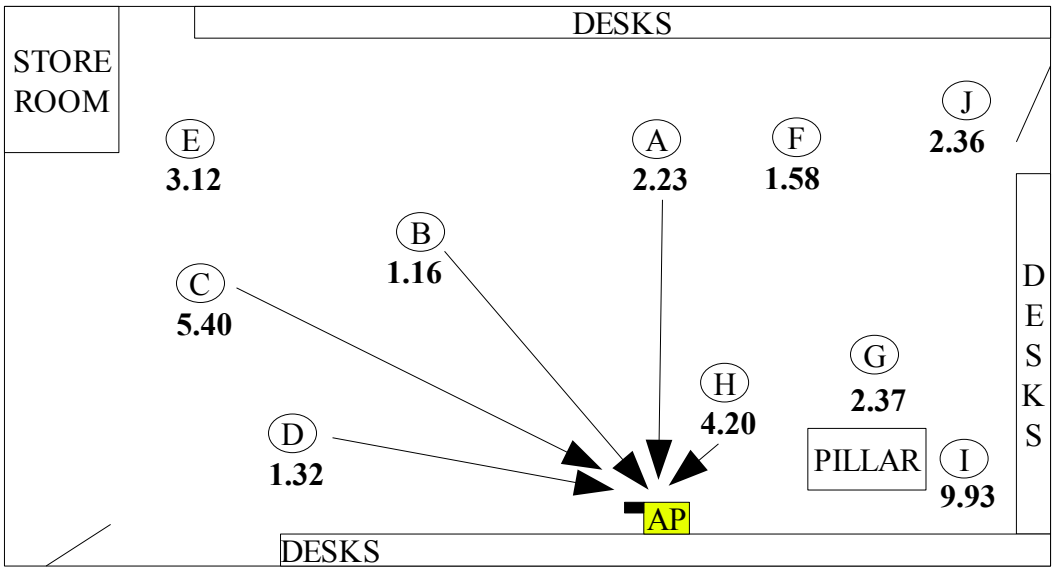

Illustration 22: Rough Positioning of AP finding points

Shown in Illustration 22 are the rough positioning of the ten random test areas, they are shown to spread across the room. The number associated with each position is the time taken to travel from the start point to the AP. This data is shown in Table format shown in table 12, on page 73.

There are a number of factors that are relevant to mention at this point regarding the testing of the finding AP code. As the testing was carried out after the movement from Area B -> Area A testing the battery's were replaced to assure that the AIBO's operation was not compromised by low power. Another reason for changing the battery is the heat given off by both the battery and the back leg joints, with the wireless card being directly above the battery. This heat was deemed to have an affect on the AIBO's ability to navigate.

An additional aspect which was shown to affect the AIBO's movement was the position of the tail in relation to the AP. The wireless card is located near the AIBOs rear, meaning when the tail was facing the AP the AIBO receives a stronger signal than when the head is facing the AP. This error corrects itself when the AIBO makes a number of forward movements.

Position	***Distance from AP (m)***	***Time taken to "Kick" AP (mins)***
A	2.50	2.23
B	2.80	1.16
C	4.10	5.40
D	3.30	1.32
E	4.90	3.12
F	3.23	1.58
G	3.30	2.37
H	0.79	4.20
I	3.58	9.93
J	4.60	2.36
Mean Distance = 3.31 meters		Mean Time = 3.37 mins

Table 12: from ten random Points to the AP

Table 12 shows the distance from the AP to the ten random areas and the time taken to navigate from the randomly chosen position to the AP. The times vary depending on distance and position in relation to the AP. The data shown in table 12 relate to Illustration 22.

The results in Table 12 allow for the calculation of an average time taken to travel from any position to the AP.

$$MeanDistance / MeanTime = Meters\ per\ Min$$
$$3.31/3.37 = 0.98\ meters/min$$

Illustration 23: Average time per to find AP

Illustration 23 shows how the result of the meters per minute was calculated, the times were converted into seconds and the mean results of both time and the mean value of the total distance travelled were dived. The calculation supports the argument that when placed Y meters away from the AP, the AIBO will take Y *0.98 minutes to travel to the AP.

As an extension of the initial testing it was established that a number of tests from the a single starting point should be conducted and provide an average time over a standard distance. This is shown in table 13:

Starting Point Position B – 2.80 m from AP – first time = 71 seconds	
	Time taken to "Kick" AP (Seconds)
1	48
2	46
3	66
4	63
5	56
6	61
7	51
8	70
9	48
10	39

Table 13: Ten Tests from same start point

The meters / minute calculation for these results averages 0.51meters/minute. However these results were calculated using position B, shown in Illustration 22 as the start point, this is a lot closer than other starting points and shows how the closer the AIBO is to the AP, the quicker it finds the AP. When placed further away from the AP, the AIBO takes longer to find the AP.

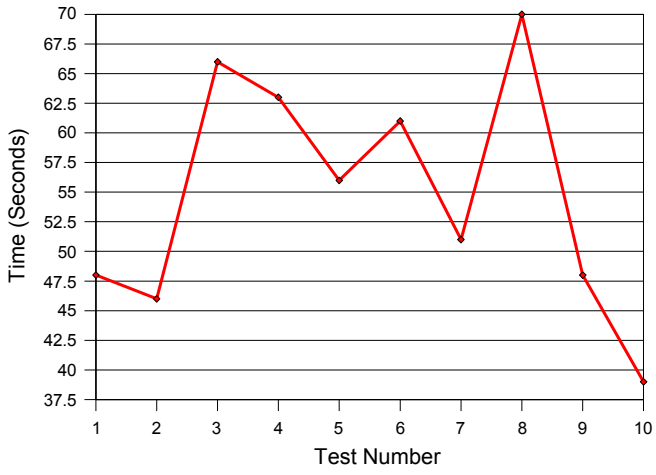

Illustration 24: Spread of data over ten identical tests

Shown in Illustration 24 is the spread of data from the ten tests. The majority of results are within the 50-65 second range. There are issues regarding the layout of the test area in which the AIBO gets lost underneath desks and takes additional time to navigate back into the main test area.

The additional testing does however support the ability of the AIBO to consistently find the same point, from the starting position.

8 Conclusion

8.1 How successful has the project been?

The project has been a great success. The aim from the outset was one in which location based reasoning would be shown to be implemented in the AIBO programming language and within the AIBO hardware specification. The AIBO has been shown to use the wireless signal within a large test area to perform two very different tasks, one in which the AIBO reports generalised position based on current Artificial Intelligence techniques, the second in which the AIBO has been shown to reason based on current readings and make movements towards a desired target zone.

There have been many set-backs and many times in which these two aims have been thought to be impossible and a complete lack of Information regarding the AIBO has hindered often very simple tasks, which in the future can hopefully be explained using the contents of this report. The success of the project can be seen in two respects. The first relates to the actual demonstration of the results and whether the AIBO achieves what was set out to achieve in the project's proposal.

The second aspect refers to the potential shown for wireless based reasoning and navigation using standard equipment. The work shown here opens many different areas in which further experimentation can produce further interesting results.

8.2 What aims have been met?

(The *Italic sentences* are the aims as outlined in the original project proposal, followed by an assessment as to whether they have been met or whether they require further investigation.)

1: *"Establish which of the three programming interfaces (YART, RCodePlus, OpenR SDK) available for use with the AIBO are most suited for the project."*

The initial project work not only focused on evaluating the State of the Art but also involved detailed investigation into which of the programming languages was best suited for the project. Throughout the project I have compared my results to those of Gilles Reant's who used the URBI scripting language. It is with complete confidence that I report Open R to be the most powerful, structured and stable coding environment in which the full power of an Object Orientated Programming language can be combined with the highly technical control of a modern robot. It was an aim of the project to produce code and quality work of the same level as leading Universities currently working with the AIBO. For this reason Open R from the offset was the chosen development environment. Although it is fair to state at this point that without the help of URBI the movements required for use in this project would still be being collected. The comparison of the languages and the similarities and differences can be seen in section 4.6 of this report.

2:*"Develop location based reasoning within the AIBO using the in-built 802.11 antenna and Wireless Access Points."* Two types of location based reasoning have been shown to be implemented in this project. The first regards the location prediction, the second regards the navigation towards the epicentre of signal strength within a test area.

Both these achievements have been developed using standard 802.11 signals and standard AIBO programming, the key with IEEE 802.11 is that it is standard. This standardization was a requirement in the programming of the code, with some translation into another language and hardware reading adjustments the code is required to work. With the AIBO being withdrawn from production during the initial stages of the project, the theory behind the project was seen as more important than the actual code. These achievements meet and extend over the initial aims of the project.

3:"*Develop and test this in a Lab environment and then see how the implementation in different environments effects the reliability and usability of the dog's loc-*

ational features." This is one of the areas in which the evaluation and testing had to be pulled back. There was an extended period in which the ability of the AIBO to move and navigate was assessed, which was not anticipated. The next student to work on the project will have a head start. For this reason and time restraints regarding the final report writing, it has been concluded that this aim cannot be met.

There is also the fact that a multitude of research has been carried out on the exact topic regarding the environmental effects of wireless signal transmission. This area is one of great interest especially when an air conditioning unit placed within the test area had a negative effect when the AIBO was trying to find the AP. The signal seemed to bounce off or be drawn towards the air-con unit and create an area further away from the AP which had a stronger signal in comparison to the space between the unit and the AP. Once removed the AIBO navigated in a much efficient manner.

4: *"Establish whether the current version of both the AIBO and 802.11 are developed enough so that the dog can accurately navigate within a known environment thus finding its way to specific grid references."* The second experiment in which the AIBO navigates to the AP supports the implementation of this aim by showing how combined movements can be used to navigate the AIBO to a specific set of co-ordinates, in this case, the strongest set of signals in the room. This is later discussed in section 9.6 of this report.

The AIBO has shown navigation based on the ability to read a signal value, assess where it is in relation to known, learnt, areas and from this relocate itself to the required area.

The 802.11 technology is by far capable of allowing localisation. This is still an area which in under researched and underdeveloped. The implications from the common user, to the Enterprise environment are wide-ranging and impressive. The AIBO has done the best that it can, the resources regarding AIBO operation

and few and far between, the development has ceased and the code base available is limited. What the project can conclude is that using a single AP, accuracy, reliability, navigation and localisation can be created in a real world environment. After working with 802.11 for an extended period I feel, especially with the longer range introduction of (802.11-N) in Jan 07 that wireless will continue to provide more and more locational aware devices.

8.3 How could the work fit into the greater academic community?

There are many aspects of the project's work which are of great benefit to both the academic and Computer Science community. The prospect of an Internal wireless GPS system has been, in a sense, proven through the work of this project. Through the completed work it has been shown that, using a single wireless AP, an indoor wireless network can be used for position reasoning and wireless based navigation.

Others have looked at how wireless maps can be created in which an enabled entity can locate itself. Using the initial results of the project, it has been shown that areas, using a small sample of data, can be mapped and the current location of a wireless device be reported back to the user. The second experiment shows how a device, in this case a robot, can navigate to a specific point within an electronic environment based on the current and desired positions.

What makes the work of this project different from the work of others is that the above two results have been created using a single AP. Whereas all other previous work has focused on multiple APs, the project, not through choice, has been forced to carry out all experiments using a single AP. It is the equivalent of a GPS system based on a single satellite. From the outset of the project I aimed to either prove or disprove this could be done using a single AP, which has been proved to be possible.

The achievements using a single AP only go to show that with the introduction of other APs the level of accuracy and the performance of the system would only further increase.

8.4 What has the project taught the writer?

The project, and the work completed before the project started, aimed to investigate how successful 802.11 could be as a location estimation and navigation tool. The number of wireless APs in the home environment has grown exponentially over the last three years: standard Eircom broadband is now supplied with a wireless ADSL router, so each new Eircom broadband customer is installing an AP in their home. The background aim of the project was to show how this wireless technology could be harnessed within the home to give added value to the existing installation of wireless technology. Using the AIBO as a demonstration tool not only shows how the robot can use the wireless technology but how all wireless enabled devices can benefit from an existing infrastructure.

After completing the project I would confidently argue that the ability of 802.11 is only just being tapped into. There are many more areas in which both robotics and wireless can be further integrated.

From a personal prospective, there have been many hurdles and set backs which have met and solved over the period of the project. To take a technology which is poorly documented, low-level and widely unknown and to achieve so much furthers the thought in my mind that there is a solution to every problem to which we are exposed. The solutions sometimes takes more or less time, but can always ultimately be solved.

8.5 Why has the AIBO production ceased?

Sony announced the AIBO production was to cease in March 2006, from this point on the only sales would be remaining stock. BSkyB Sky news in mid November 2005 reported the AIBO as the “must-have” Christmas product. Why did the seemingly great interest in the robotic pet lead to its demise?

After working with the AIBO over an extended period I feel that the cost (around $2000) and the high level of technical expertise needed to use the AIBO both killed the robot in the academic and home environment. If one was to forget the third-party scripting languages built to run on top of the AIBO then the core coding option is the Open R C++ hybrid. Open R is so low-level that most competent programmers need to take step back to take a step forward.

The market for electronic devices is so cutthroat that the AIBO, in my opinion was doomed from the start. Using the Mind Software the user had an option of pre-defined programs which would allow the AIBO to conduct neat tricks, but nothing else.

The other aspect is the cost which leads to a small niche market in which the technical development can be of benefit to the user. If the cost were lower the market share of the AIBO would have increased and the popularity of the robotic pet could have been in line with the possibilities of the AIBO.

The AIBOs production has ceased but there are a multitude of other robotic developments in circulation. These although at a higher cost, show that the AIBO was just the start. What seems a shame about the AIBO's demise is that it was the first mass-produced consumer product, which if successful could have seen robotics in many homes, thus increasing the popularity of the pet.

Although the code in this project was created using the Open R / C++ format, the theory behind the code is the same in all languages. The AIBO has been used to show proof of concept, which it has done very successfully.

8.6 How will robots and wireless combine in the future?

With the hindsight of the project, I hope that the development of wireless navigation using robotics will be continued. It seems that most of the work in both the AIBO World Cup and the current development of robotics aims to show how image recognition can be computed by a robot and how a electronic human can be

created. I would argue that image recognition techniques should combine with electronic signal analysis to create an overall location estimation opposed to a single recognition technique.

Most new devices are becoming 802.11 equipped. The Microsoft "Zune" MP3 player is to be the first MP3 player which synchronises using 802.11 wireless technology. I see future wireless and robotic product development as a way in which the Zune could report to its user where in their home it is. This kind of Integration is far more organised and realistically implementable in the near future.

It is arguable that the robotic developments will more and more be integrated with navigation and that the introduction of new technology will be costly so existing sources will be used. Testing in this project shows this statement to be possible. How and when it becomes standard for wireless devices to know where they are or where they need to be is still to be seen.

8.7 What advice would you offer future students?

The main lesson I have learnt from the project is that the possibilities made available to the user through 802.11 are endless. As soon as one project is complete the next project appears and the way in which the project develops is only relevant to the specific task at hand.

There are a multitude of possibilities regarding 802.11 wireless navigation and localisation, even using a single AP. Whereas other technology is mature, the use of wireless as a navigation tool is still in its infancy. The project work completed using the AIBO is unique, having never before been implemented. The possibility of further development is completely in the hands of the program developer.

Future students who aim to look at this work, and the work of others, must initially have the confidence in their own abilities to push and question each technological development. There are key areas in which this work can be implemented and those areas are changing dramatically. The application of wireless reasoning

must not be completed using proprietary or closed code techniques, in order that the theory of the work be available to all.

It is important to Look at what your aiming to achieve from a human perspective and then, and only then, try to translate the human aims into the robotic world. It in necessary to always keep at the forefront of your mind how your work will fit into the larger picture. You must always have faith in your results and use your signals valuably. The signal which the mobile device uses is the eyes and ears of the system. Without well calculated signal values, the project is pointless. Use statistics as and when needed but combine what you see with common sense to achieve what is clear in your mind. Remember this field is relatively young, you wont find answers to even your simplest problems, but once solved, each problem becomes your own work which is something to be proud of.

The most vital lesson I have learnt through completion of this project is patience. When testing the second round of testing where the AIBO finds the AP it was shown that if the programmer thinks the AIBO is failing to find the AP, then with time the AIBO will, every time, find the AP. Using wireless technology allows for the additional constraint of variants, which you can use to your advantage.

9 Future Work

9.1 How can the project develop further?

There is now in this project a solid base from which the AIBO project can develop further and the methodologies implemented in the project can be further refined. Having cracked how to get the AIBO walking using Open R this allows for a much more and detailed analysis and testing of the wireless signals.

In the initial days of the project it was presumed to be easy to make the AIBO walk, however in the end it was the most difficult aspect of the project. Continuing at TCD to PhD level study I can advise any future students on the way in which the AIBO code is built and thus allow any future student to fully investigate the wireless signal strength whilst avoiding all the pointless stress necessary to get the AIBO to walk. With more time the level of accuracy and simulated intelligence available to a user through the AIBO offers great potential. There are many things that can be done using the new electronic eyes provided to the AIBO. Through the work in this project, there are many future possibilities for wireless robotic navigation and the perfect project for a future NDS student.

9.2 How can the work completed be used in other environments?

A summary of the work completed in this project will be forwarded to the University of Texas at Austin where hopefully the “AustinVilla” robocup team will look at how both programs can be integrated into playing football with the AIBOs. A football pitch is split into two halves. Take training data sets from each area and then the AIBO can predict whether it is in its own half or the other team's half. Combined with this an AP was placed behind the opponents goal, then the strongest signal within the pitch would be the back of the net.

Whether or not the AustinVilla take up my work and integrate it into their code base is entirely up to them. In initial discussion with the University they had not

implemented this kind of wireless navigating. It would be very interesting to see how this work could go towards and contribute to the overall reliability of AIBO football would be of great interest.

9.3 How many samples to take?

The reasons behind this question being so important are outlined in section 6.2 and for this reason will not be commented on here. However it would be a very interesting first project for a future student to look at benchmarking the software to see how well the number of samples work and whether less or more samples would have an effect on the wireless reasoning of the project.

9.4 Can the AIBO navigate from area to area across a day?

One of the initial aim of the project was to have the AIBO develop a more real approach to navigation introducing animal like behaviours to a binary robot. This has been shown in both aspects of the project, but specifically in the ability of the AIBO to find the AP.

If the AP is moved around in the test area the AIBO still navigates towards the APs new position. If the AP was installed in a new room and the AIBO placed in the room the AIBO would still find the AP. There are no restrictions on the second code base, the code is completely mobile and what is so amazing during the code's operation is the way in which the AIBO makes real decisions based on wireless readings which end with the AIBO finding its way home, to the AP.

If both sets of code were integrated, the AIBO could be shown to navigate within a room. Using the left and right movements and the reasoning based on the average signal strength the AIBO could, instead of moving from A->B, move around multiple areas which could be seen as the AIBO's home. Integration of the image recognition software, for finding the AIBOs charging base, would allow for permanent navigation and localization where the AIBO could show real dog behaviour. This was one of the aims of the project but the extended period where the project's main aim was getting the AIBO to move has impacted on this being im-

plemented. There is definitely enough work in this aspect to provide a six month project.

9.5 Can the AIBO connect to multiple APs or at least see APs?

Having spent six months working with both the AIBO and the Open R OS this is a project which I do believe is possible to code. Each user defined program is compiled into a binary file which the AIBO loads upon start-up and in which the C++ code is stored. On the user programmable memory stick there are two folders MW and SYSTEM. MW is where the AIBO stores the user code whereas SYSTEM is where the Open R default binaries are stored, there are a number of binaries in this system folder, some of which seem interesting, these include: WLANDR.BIN, WLANDRVM.BIN and NETCONFS.BIN.

Each of these files relates to a C++ source file, of course in line with other aspects of the AIBO there is no Information on what these files do. The source code to Open R has been released and Open R is now Open Source. There are a number of files in the source code which seem of great interest and a confident C++ coder, with more time, could I'm sure develop a wifi sniffer. However this is again a project for another year.

9.6 Can the AIBO navigate from room to room?

The key to all future developments regarding the AIBO is to move and vary the movements based upon readings taken from the connection to the wireless AP. Being able to find the AP shows a level of reasoning that, combined with the K-NN algorithm would allow for the AIBO to navigate to a door way or entrance and make the transition from room to room.

The tools which have been developed for use in this project not only allow for location reasoning but allow for movement towards a required position. Currently the AIBO navigates towards the area in the room in which the wireless signal is the strongest, but its is very possible that the AIBO could navigate towards a specific point, in the form of a doorway. The initial tests showed how the AIBO

can access two specific areas and show which area the AIBO is in. However to pin-point an area as concentrated as a door way would require the combination of both the location reasoning as shown in the first experiment and movement towards a desired location based upon reasoned movement as shown in the second experiment.

9.7 Can the AIBO become a robotic guide dog?

The AIBO has the ability to be trained, to perform set tasks and is not the only robotic development currently being carried out. It would be an Interesting development of the AIBO to develop human interaction in the form of a guide.

A previous NDS student looked at Location based services using 802.11, further development for this project would be investigated to show how the AIBO can be combined with location based reasoning so that the AIBO can inform a user as to where they are and what it near by. Using the current hardware and software solution provided by the AIBO the level of accuracy required to create a robotic guide dog is extremely difficult. What is not difficult is the ability of a programmer to show how the AIBO can recognise, over an extended location, various areas within a building.

9.8 Could the AIBO Locate and track expensive machinery?

The initial project idea came from a desire to track 802.11 enabled equipment within a localised area. This was shown to be already completed in various industrial fashions. It would be a great starter project using the AIBO a single AP and three rooms on the same floor integrated with the K-NN algorithm. Combining these four elements would allow for the AIBO to be physically moved from room to room whilst reporting which room the AIBO is in. Demonstration of this project would allow for a multitude of AIBOS to be placed by the observer, without the knowledge of the programmer, in random rooms from which the AIBO can be connected up to and their location reported back to the remote session.

10 End Note

The results from the experiments and work developed in this project support the idea that 802.11 can be used for wireless positioning and that using a single access point the location of a mobile device can allow tracking and position estimated of 802.11 devices.

So many of the current robotic position techniques are based upon image recognition and previous position recording. There seems to be an aim to integrate into robotics the same level of image recognition and interpretation as is processed by the human brain.

The work conducted in this project proves that using a single AP, wireless location and positioning can be calculated using standard and widely available equipment. This is not to say that the author argues that wireless locational should replace the existing image location techniques, as shown in [17.] The author argue that wireless location should be used in conjunction with current image recognition, adding to the ability of a robot, or device, to locate itself.

The author predicts successful location awareness, indoors, will develop using 802.11 to the point where most capable devices will operate and position themselves using techniques based on wireless signal strength.

11 References

[1] Meunier, J.-L *Peer-to-Peer Determination of Proximity Using Wireless Network Data,* Pervasive Computing and Communications Workshops, March 2004

[2] Yi-Chao Chen, Ji-Rung Chiang, (et al), *Sensor-Assisted Wi-Fi Indoor Location System for Adapting to Environmental Dynamics* International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems, 2005

[3] Peter Stone, *CS395T Autonomous Robots online teaching notes / references* http://tinyurl.com/l7csf**,** University of Texas at Austin, 2006

[4] Shane Brennen, Dominik Haug, Dominic Jones - *Comparing the Accuracy of Tracker Algorithms for Location Based Services,* Computer Science Dept, Trinity College, Dublin, 2005

[5] Zef Hemel, Javier Loriente, Leonard Raphael, *Measuring the wireless performance of an AIBO,* Computer Science Dept, Trinity College, Dublin, '05

[6] Oscar Serrano, Jose Maria Carias, (et al), *Robot localization using WiFi signal without intensity map,* University Rey Juan Carlos, Mostoles, Spain, 2004

[7] Andrew Clarke, *A Reaction Diffusion Model for Wireless Indoor Propagation,* M.Sc. Dissertation, Computer Science Dept. Trinity Col lege, Dublin, 2002

[8] http://www.ekahau.com/ - *T301 Ekahau Wi-Fi Location Tag*

[9] Chandon Kee, Doohee Yun, (et al), *Centimetre-Accuracy Indoor Naviga tion Us`ing GPS-Like Pseudolites, www.gpsworld.com* 2001

[10] Oscar Serrano Serrano, *Robot Localization using wireless networks,* Uni versity Rey Juan Carlos, Mostoles, Spain, 2003

[13] Hemel, Loriente, Raphael, *Measuring the Wireless Performance of an AIBO*, Trinity College NDS Paper, March 2006.

[12] Karygiannis, Owens, *Wireless Network Security 802.11, Bluetooth and Hand held Devices,*Recommendations of the National Institute of Stand ards and Tech nology, 2002

[13] Simon, Aboba, Moore, *IEEE 802.11 Security and 802.1X*, Microsoft Corporation, 2000

[14] Miguel Silva Rentes, *Testing OPEN-R Samples for the SONY AIBO ERS-7,* Artificial Intelligence and Computer Science Laboratory, University of Oporto, Portugal, 2005

[15] Jean-Christophe Baillie, *AIBO programming using OPEN-R SDK Tutorial*, The URBI team, www.urbiforge.com, 2004

[16] Erik Kaltofen, *Generic Programming: C++ vs. Java,* North Carolina State University,

[17] David Cohen, Yao Hua Ooi, (et al), *The University of Pennsylvania RoboCup 2003 Legged Soccer Team*, University of Pennsylvania, 2003

[18] Gilles RÉANT, Meriel Huggard, *W/BO project - Wi-Fi over AIBO,* Trinity College, Dublin, Ireland, 2006