Providing Precise Indoor Location Information to Mobile Devices

by

Nissanka Bodhi Priyantha

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

January 2001

Author . . . .
Department of Electrical Engineering and Computer Science
January 12, 2000

Certified by . . . .
Hari Balakrishnan
Assistant Professor
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Providing Precise Indoor Location Information to Mobile Devices

by

Nissanka Bodhi Priyantha

Submitted to the Department of Electrical Engineering and Computer Science
on January 12, 2000, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

This thesis describes the design and implementation of a location support system called Cricket, which provides mobile applications with indoor location information. The system has no central point of control, which enables easy, decentralized management. Unlike traditional indoor location tracking systems, Cricket preserves user privacy by not tracking the locations of its users. It uses a combination of radio frequency (RF) and ultrasound signals, providing spatial information about the region of space in a which a user or a device resides, rather than simply providing location information as a point within some coordinate system. Cricket enables context-aware location-dependent applications; this thesis describes a few such applications.

Thesis Supervisor: Hari Balakrishnan
Title: Assistant Professor

Acknowledgments

I am grateful to my advisor, Professor Hari Balakrishnan for being a continuous source of advice and inspiration to me. His guidance and understanding has helped me both academically and otherwise. His excellent communication skills and technical input have helped me greatly in my research work.

I am grateful to Professor John Guttag for the help and understanding he showed me during the start of my life at MIT. Dr. Steve Garland provided valuable advice and comments relating to my work. I thank John Ankcorn, Dorothy Curtis, Anit Chakraborty and Allen Miu for their help in the Cricket project.

I am grateful to my office mates, Deepak Bansal, Suchitra Raman, and Jorge Rafael for the endless discussions on various topics we have had. I would also like to thank the other graduate students in the Networks and Mobile Systems group at LCS for making my life at MIT both exciting and interesting.

Some of the text in this thesis has been taken from my paper with Anit Chakraborty and Hari Balakrishnan, which appeared at the ACM Mobicom Conference in August 2000. The Floorplan application was developed by Anit Chakraborty. My research at MIT was primarily supported by grants from NTT Corporation and DARPA.

I dedicate this thesis to my parents whose love and support have been the reasons I have gotten this far.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Over the past few years, we have seen a proliferation of computing devices with varying degree of computational power, ranging from powerful server systems to embedded net-enabled devices. A natural progression of this trend is leading toward an era when computing will be widely available and embedded in our environments, where it will be “ubiquitous” [27] or “pervasive” [8, 4]. The number of users of mobile computing devices has also been growing dramatically spanning the entire spectrum from laptop computers to hand-held personal digital assistants, to more specialized devices such as cellular phones or portable music players.

For applications running on mobile devices to truly benefit from the large number of services available in the environment, there need to be a mechanism for the application to determine its spatial location and the resources in its vicinity, and adapt its behavior accordingly. This in turn calls for a system to determine the physical location of both the fixed and mobile devices. In addition to discovering services in its vicinity, a mobile application can change its behavior depending only on the location information itself, an example being a cellular phone that does not ring loudly inside a conference room.

Applications that adapt to their physical location in outdoor environments using Global Positioning System(GPS) [10] have started becoming available, such as navigation utilities for cars. Since a large class of users and important applications will almost always be located indoors, a system that enables devices and the applications running on them to learn their location inside buildings will be an important step in enabling a number of mobile computing applications to adapt to their location.

The design and deployment of a system for obtaining location and spatial information in an indoor environment is a challenging task for several reasons, including the preservation of user privacy, administration and management overheads, system scalability, and handling the rather unpredictable nature of indoor wireless channels.

Cricket is a location support system that provides indoor location information. Cricket consists of a collection of *beacons* that are spread across a building. Each beacon periodically transmits a text string corresponding to the name assigned for its location. Various devices in the environment, such as static and mobile computers, printers, cameras etc., have Cricket *listeners* attached to them. By listening to the beacon advertisements, each listener determines the space it is in, and passes this information on to the associated device. Once this information is obtained, services advertise themselves to a resource discovery service such as the MIT Intentional Naming System (INS) [2], IETF Service Location Protocol [25], Berkeley Service Discovery Service [7], or Sun's Jini discovery service [16]. In the active map application we developed using the INS infrastructure, services advertise their existence to a map server application with information of their location and the service they offer, and the users discover the services in their vicinity by querying the map server for services at a location.

1.2 Related work

There are several systems that are in existence today for providing location information to users in both indoor and outdoor environments. Some of these systems are mature technologies used by many users while others are still confined to research laboratories. The systems developed for indoor environments determine location information by tracking the objects and users. Furthermore, these systems are built around a central controller which builds a location data base by querying individual objects using a network of wired base stations. Unlike these traditional systems, Cricket provides a location service without tracking its users.

1.2.1 Global Positioning System (GPS)

GPS is an outdoor location system [10]. Although primarily designed for defense purposes, it is used today in a number of non-military applications [11, 14]. GPS infrastructure consists of 28 satellites orbiting the earth at an altitude of about $20,000$ km. These satellites carry very accurate atomic clocks. These clocks are continuously monitored by ground stations operated by the US Air Force and any offsets of the satellite clocks are corrected by issuing appropriate ground-control commands. Each satellite transmits a unique data stream as an RF signal.

When a GPS receiver receives a bit stream corresponding to a particular satellite, it uses the time of flight of the RF signal from the satellite to determine its position. If the receiver's clock is perfectly synchronized with the satellite's clock, the receiver can determine its location using the signals from three different satellites. Because it is extremely difficult to perfectly synchronize the receiver's clock with the satellite's clock, signals from *four* satellites are used to compensate for any offsets in the receiver and satellite clocks. The accuracy of the inferred location information improves if the receiver can receive signals from more than four satellites. Since the GPS satellites are not geo-stationary, the receiver contains a map of all the GPS satellites, which allows it to determine the location of individual satellites at any given time.

While GPS provides a precision of $5 - 10$ meters for outdoor environments using low-cost receivers, it is possible to sometimes achieve sub-centimeter accuracy by using very sophisticated receivers in outdoor environments. However the low RF signal strength, high RF noise, and the reflections of RF signals due to the presence of metallic objects make GPS unsuitable for indoor environments.

1.2.2 GPS-less outdoor localization system

This is a low cost location system for outdoor use, consisting of a number of fixed RF stations with overlapping coverage regions, where each RF station periodically transmits its unique ID and position [5]. The receiver nodes that listen to these signals measure their connectivity, expressed as the fraction of RF signals received from a given reference RF station. The receiver estimates its location as the centroid of the reference points for which it has "sufficient" connectivity, where the centroid is defined as the region of intersection of connectivity regions of the set of reference points under consideration. The system claims to have an accuracy of $2-3$ meters [5]. The authors of the paper mention that this system cannot be used in indoors due the highly unpredictable nature of RF propagation in such environments.

1.2.3 Active badges

The Active badge[1] system, developed at Olivetti labs, was one of the earliest indoor location tracking systems, whose architecture has influenced subsequent systems too [26]. Objects are tracked by attaching a badge, which periodically emits its unique ID using an infrared transmitter. A fixed infrared receiver placed in each room, picks up this information and relay it over a wired network to the central database. The walls of the rooms act as natural boundaries for infrared transmissions, thus enabling a receiver to safely assume that any badge it hears from is located within the same room. Thus, the badge is associated with the room in which the corresponding fixed receiver is located.

The tracking nature of this architecture raises several thorny privacy issues [24]. The wired infrastructure adds to the system's deployment and maintenance costs, while infrared suffers from dead spots in rooms.

[1]Active Badge is a registered trademark of Ing. C. Olivetti &C., S.p.A.

1.2.4 The Bat system

The Bat system, developed at AT&T Research Labs, provides indoor location information by tracking the whereabouts of its users. Various objects and users within a region have uniquely identifiable wireless transmitters or tags attached to them. A centralized location database storing the position of these transmitters is built by periodically tracking each object.

The infrastructure consists of a carefully laid out matrix of receiver elements. Each receiver element consists of an ultrasonic and RF receiver, laid out to form a 1.2m by 1.2m grid. They are typically mounted on the ceiling of a room, and are interconnected using a serial wire network. This wired network also connects to one or more RF base stations and to the central location database.

The wireless tags being tracked consist of an RF and an ultrasonic transmitter, and have globally unique identifiers. Each tag is queried periodically, one at a time, by broadcasting messages addressed to it from the central controller.

A tag, upon hearing a message addressed to it, responds with an ultrasonic pulse. Each receiver element, which also receives the original RF message from the base station measures the time difference between the arrival of that RF message and the ultrasonic response from the tag. This time difference is used to obtain the distance between the tag and the receiver element under consideration. This data is then sent to a central station where processing is done to remove inaccuracies caused by factors such as reflected ultrasonic signals. By obtaining at least three accurate distance measurements, it is possible to estimate accurate position of the tag, and hence the position of the object being tracked.

The Bat system achieves an accuracy of 3-4cm for distance estimation as a consequence of its tightly controlled and centralized architecture, and the accurately laid out grid of sensors [13, 12]. However, its architecture is fundamentally based on *tracking* the users and devices, while its wired infrastructure causes the deployment and maintenance cost of the system to be high.

1.2.5 RADAR indoor location system

The Radar system developed at Microsoft Research implements an indoor location service by leveraging an already existing RF data network [3]. Here, the RF signal strength is used as a measure of distance between RF transmitter and a receiver. This information is then used to locate a user using triangulation. The system uses one of two approaches to determine the position information.

In one scheme, in an off-line phase, an RF signal strength map of the whole space is generated by placing transmitters at various locations and measuring the signal strength at a number of fixed RF receiver stations. Then, during normal operation, the signal strength due to a transmitter is measured at these receiver stations and the transmitter position is inferred by a best fit.

In the second scheme, the system uses an RF propagation model that takes into account factors such as number of walls between a receiver and a transmitter to compute the RF signal strengths due to placement of transmitters at various positions. This information is then used to obtain the best fit for a given transmitter's position.

The position calculation can be done either at a central controller or at the receiver itself. If the latter approach is used, the system preserves user privacy. RADAR depends on the RF signal strength to determine the distance to RF base stations, but the highly unpredictable nature of the RF propagation within buildings, coupled with the dynamic nature of the enviorenment itself, causes the accuracy of distance measurements to be only about 3 meters [3]. Further, for the first scheme described above, the generation of the off-line RF signal strength map becomes a cumbersome procedure.

1.2.6 PinPoint Local Positioning System

This is a tracking system that enables locating both mobile and fixed items within an environment [21]. A region such as a business organization is divided in to number of cells. Each cell has a cell controller to which several RF antennas are attached. And RFID tags are attached to various device that are to be tracked. The cell controller generates a spread spectrum radio signal that is broadcast via the antennas. Each tag, after receiving the signal, responds with a message containing its unique ID. The signals received by different antennas are sent to the controller. The controller uses the time of flight (the difference between the transmit time and receipt time, perhaps compensating for any internal delays of the TAG) of the RF signals from the tag, received at different antennas, to calculate the distance to the tag from these antennae. These distances are used to identify the location of the tag to an accuracy of 3m. The tracking nature and the relatively low accuracy of this system makes it unsuitable for the type of environments the Cricket system is designed for.

System	Bat	Active Badge	RADAR	PinPoint	*Cricket*
User privacy	No	No	Possible, with user computation	No	*Yes*
Decentralized	No	No	Centralized RF signal database	No	*Yes*
Heterogeneity of networks	Yes	Yes	No	Yes	*Yes*
Cost	High	High	No extra components	High	*Low*
Ease of deployment	Difficult; requires matrix of sensors	Difficult; requires matrix of sensors	RF mapping	Difficult; requires matrix of sensors	*Easy*

Table 1.1: Qualitative comparison of other in-door location-tracking systems with Cricket.

1.3 Design goals

The Cricket location support system was designed as a part of Project Oxygen at MIT's Laboratory for Computer Science, as an aid for context-aware systems and applications in pervasive computing environments [20]. Its design centered around the following set of goals, which were motivated by the needs of Oxygen's pervasive computing scenarios:

1.3.1 Preserve user privacy

The previous systems discussed above for providing location information *track* the whereabouts of its users to build a location database. However, this leads to a serious violation of user privacy, as users generally tend not to like their whereabouts tracked and logged [9]. Indeed, there are actual reports of tension caused in the workplace by the presence of such tracking devices [24]. Cricket effectively handles this issue by letting its user learn their own location rather than tracking them, listeners in Cricket are *passive* devices and only infer and provide information to applications running on the attached device.

1.3.2 Recognize spaces rather than position

Location information that describes a space, such as a room or a portion of a room, has more meaning and relevance to practical applications, compared to a description based on point coordinates in some frame of reference. For instance, location information in the form of being inside a particular room of a building is more convenient than that given as being at a certain (x, y, z) co-ordinates within that building. In particular the system should be able to accurately identify the *boundaries* between spaces of practical interest. For instance, a location inaccuracy of several centimeters can easily cross the boundary between two logically distinct domains, such as the eastern and western portions of a room.

The technologies used in Cricket enable the accurate detection of natural boundaries such as walls between spaces, while logical boundaries such as those used to divide rooms in to sections and doorways can be detected with an accuracy of 20cm from the boundary.

1.3.3 Operate inside buildings

Most pervasive computing environments and users are expected to be *indoors*, which implies that Cricket must work well inside buildings. Systems like GPS, that does not track its users, do not work indoors.

1.3.4 Easy administration and deployment

Easy administration and deployment are important aspect of any system. Cricket has a highly decentralized architecture which makes it very easy to administer and allows scalable and incremental deployment. The absence of an interconnecting wired network in Cricket compared to previous systems such as Active Badge and Bat also makes the deployment easier and less costly. The architecture of Cricket adds flexibility by allowing the user to select the proper naming convention for spaces. The name used can be any text string such as a room number, an intentional name, or a URL of a server to access for more information regarding the particular space [2, 15].

1.3.5 Low cost

Due the nature of the Cricket architecture, the fixed cost of deploying the system is small. The total system cost depends primarily only on the number of beacons and listeners used. However, since a large number of Cricket beacons would be needed to provide location support within a single building or an organization, it is important to keep the unit cost as small as possible. Cricket is built using off-the-shelf components, without custom hardware.

Table 1.1 compares the relative benefits and limitations of in-door location technologies discussed above with Cricket.

1.4 Contributions and roadmap

This thesis presents the design, implementation, and evaluation of Cricket, a location-support system for in-building, mobile, location dependent applications. Cricket consists of a collection of beacons spread across a building, and listeners attached to both mobile and fixed hosts. Cricket is the result of several design goals, including user privacy, decentralized administration and architecture, and low cost.

We describe the use of careful system engineering, randomization, and inference algorithms that makes the system robust against interference caused by the uncoordinated beacon transmissions. We also describe an active map application that uses the location information provided by the Cricket system.

The rest of of the thesis is organized as follows. Chapter 2 describes the architecture of Cricket, the use of ultrasound and RF to achieve accurate boundary detection, our solutions to the problems caused by highly uncoordinated nature of the system, and practical beacon configuration issues. The chapter ends by presenting experimental results that demonstrate the correctness and the robustness of the design.

In chapter 3, we study the effects of different transmission schemes on RF channel utilization, which is closely related to the responsiveness and scalability of the system. We use both analytical and simulation techniques to determine the channel utilization under different transmission schemes and show that it is possible to achieve good system performance under large and varying beacon densities.

Chapter 4 discusses various implementation issues including data packet formats and practical issues in deploying ultrasound transmitters to achieve good system performance. We end the chapter by presenting an active map navigation utility built using Cricket.

Chapter 2

System architecture

Cricket consists of a set of beacons that are located at fixed positions and set of listeners that are attached to various devices. Each beacon is associated with a particular space and it periodically broadcast this location information using RF signals. Typically, a beacon is obtained by the "owner" of the location (e.g., the occupant of a room in an office or home, or a building administrator)and is attached to the ceiling or the wall . And the location string to be disseminated by the beacon is configured using a special RF transceiver unit. To enable the use of multiple beacons within a given space for both redundancy and better coverage, each beacon is assigned a unique identifier within that space. This identifier, together with the location string enables each beacon within the system to be identified uniquely.

Listeners are attached to both mobile and fixed devices using an `RS 232` interface. Each listener listens to beacon transmissions and interferes the closest beacon from the set of beacons it heard from. And it associates itself with the space advertised by the closest beacon.

2.1 Determining the distance to beacons

Cricket uses a combination of RF and ultrasound to enable listeners determine the closest beacon. The Beacons advertise the location string using RF signals. It uses an RF transmitter operating at 418MHz with a data rate of 5kb/s. The data to

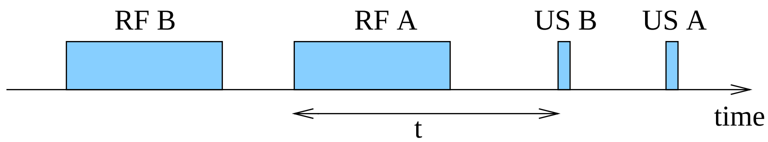

Figure 2-1: Inaccurate distance estimate because RF_A and US_B are associated with each other.

be transmitted is encapsulated into a data packet. Each data packet is transmitted as a sequence of bytes using an asynchronous communication scheme similar to `RS-232`. Before transmitting the actual data packet, the beacon transmits a sequence of synchronization characters. Following this, it transmits a narrow (500μs) pulse of ultrasound at a frequency of 40kHz. It then transmits the actual RF data packet, one byte at a time.

The velocity of RF in air is approximately 3×10^8m/s, while the velocity of ultrasound in air is 344m/s at 68°F. This difference of six orders of magnitude in velocities causes the ultrasound signal from the beacon to lag behind the RF signal as they propagate through air. Thus, by measuring the time difference between the receipt of the first RF bit and the ultrasonic pulse, a listener can estimate the distance to the beacon. Of the set of beacons heard, the one with the smallest distance is the closest to the listener. The listener associates itself with the location-string advertised by that beacon.

2.2 Handling interference

While Cricket has the attractive property that the collection of decentralized beacons is potentially easy to configure and manage, it comes at the absence of explicit coordination of beacon transmissions. This lack of coordination can cause signals from different beacons to interfere at the listener. Consider RF signals RF_A, RF_B and ultrasonic signals US_A, US_B of two beacons A and B respectively. The two RF signals RF_A and RF_B can be distinguished at the listener due to the unique location strings encoded in these signals. However, because the ultrasonic signals from A and B are just pulses, the listener cannot differentiate the two ultrasound signals US_A

and US_B. This, together with the lack of explicit coordination among the beacons, can cause inaccurate distance estimates at the listener. For example, it cause the signals from two beacons to interfere at a listener, as shown in Figure 2-1. Here, the ultrasound pulse US_B of beacon B arrives immediately after the RF signal RF_A of beacon A, and before the arrival of the ultrasound pulses US_A of beacon A. Since the listener cannot differentiate between the two ultrasonic signals US_A and US_B, it simply uses the time difference t between RF_A and US_B to represent the distance to beacon A, thus making an incorrect distance estimate to A.

One possible solution to this would be to modulate the ultrasound signals with the beacon location-string (or some unique identifier), thus enabling the listener to unambiguously correlate the corresponding RF and ultrasound signals. However, the relatively low frequency of the ultrasound signal would result in an extremely slow data rate. Furthermore, ultrasonic signals suffer from severe multi-path effects caused by reflections from walls and other objects, and these are orders of magnitude longer in time than RF multi-path because of the relatively long propagation time of sound in air. These two factors make it practically impossible to modulate ultrasonic signals to carry useful amounts of data.

Rather than preventing beacon interactions completely, the Cricket system uses a combination of three different mechanisms to overcome the effect of such interactions. These are: (i) the use of well-engineered system parameters, (ii) randomization, and (iii) inference algorithms at the listener. Together, these techniques progressively reduce the effects of beacon interference and enable the accurate identification of the closest beacon by the listener.

2.2.1 Engineering system parameters

Carefully engineering the parameters of the hardware used in Cricket can significantly reduce adverse inter-beacon interactions. Consider the RF signal RF_A and ultrasonic pulse US_A of a beacon A, being received by a listener. Any ultrasonic pulse US_I arriving between RF_A and US_A will cause an inaccurate distance estimate to beacon A by the listener (Figure 2-2).

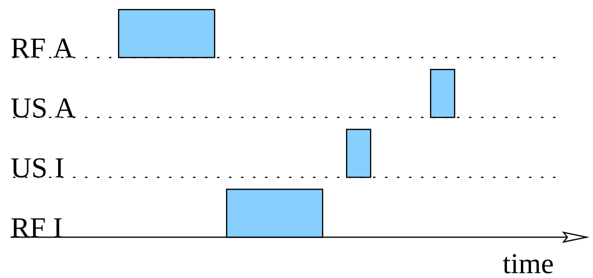

Figure 2-2: A larger RF range compared to the range of ultrasound ensures the receipt of the RF signal corresponding to an intrefering ultrasound signal US_I at the listener.

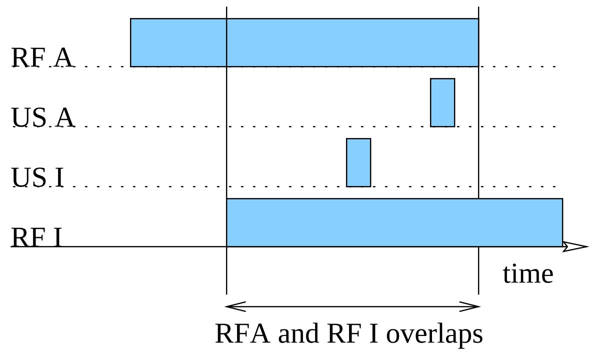

Figure 2-3: Two overlapping RF signals, RF_A and RF_I, and the corresponding ultrasonic signals.

Let the line-of-sight range of RF and ultrasonic signals within the Cricket system be r_{RF} and r_{US} respectively. In cricket these are selected to satisfy the condition $r_{RF} > r_{US}$. In general, this guarantees that a listener will detect the corresponding RF signal, if it detects an ultrasonic signal from a beacon. Thus, in the above example, the RF signal RF_I associated with the ultrasonic signal US_I will also be received at the listener (Figure 2-2).

Let the velocities of ultrasound and RF in air be v_{US} and v_{RF} respectively. The time difference between the arrival of these two signals at a listener at a distance r from the beacon is given by :

$$T = \frac{r}{v_{US}} - \frac{r}{v_{RF}}$$

Since $v_{RF} \gg v_{US}$;

$$T \simeq \frac{r}{v_{US}}$$

Since the maximum distance the ultrasound can travel is r_{US} and $r_{US} < r_{RF}$, the maximum time difference occurs when the listener is at a distance r_{US} from a given beacon, and is given by :

$$T_{max} = \frac{r_{US}}{v_{US}}$$

Let the length of the location string advertised by a beacon be L, and the bit rate of RF transmitter be r. The transmission time of the location string is therefore $\frac{L}{r}$. The duration of the ultrasound signal is t_{US}. Now, if the inequality :

$$\frac{L}{r} - t_{US} > \frac{r_{US}}{v_{US}}$$

is always maintained, the ultrasound signal from a particular beacon will always be enveloped by the corresponding RF signal. The above equation determines the minimum length of the location string advertised by beacons for a given set of system parameters. This minimum length is given by :

$$L_{min} = (\frac{r_{US}}{v_{US}} + t_{US})r$$

As Figure 2-3 shows, these conditions will result in the two RF signals RF_I and RF_A overlapping. This overlap will cause bit-errors in the RF data received at the listener, which are usually detected by the block-parity error check done by the listener on received data. Hence this selection of the minimum length location string considerably reduces the ill effects of inter-beacon interference. In Cricket, the probability of bit-errors due to overlapping RF signals is increased due to our use of On-Off-Key (OOK) modulation for RF data transmission, as opposed to a more robust modulation scheme such as Frequency Modulation (FM).

2.2.2 Randomization

Although the above scheme reduces the possibility of wrong distance estimates at the listener, it cannot totally eliminate them. This is due to several reasons. The unpredictable nature of RF propagation prevents us from giving hard guarantees on the inequality $r_{US} < r_{RF}$. If one RF signal is much stronger than the other due to temporal or spatial fading, the stronger RF signals will be received without any bit errors. In addition, there will always be some bit-error combinations that cannot be detected by the block parity error check.

To further reduce the possibility of the repeated occurrence of any undetected errors we randomize the transmission from each beacon. The time interval between the consecutive transmissions of a particular beacon is not a fixed value, but is a random value which is uniformly distributed between values T_1 and T_2. Because of this randomization, any inaccurate distance estimates to a given beacon caused by repeated interactions with other beacon transmissions, will reduce exponentially with the number of transmissions. This property is used by the listener inference algorithms to correctly identify the closest beacon to a listener.

2.2.3 Inference algorithms

With the assumption that any inaccurate distance estimates at a listener will have a very low probability of recurrence while correct estimates have a very high probability of recurrence, we analyze the performance of three inference algorithms at the listener for determining the closest beacon.

- *Majority.* This is the simplest algorithm, which pays no attention to the distance estimates and simply picks the beacon with the highest frequency of occurrence in the data set. This algorithm does not use ultrasonic signals for determining the closest beacon, but as we find in our experiments, this does not perform well. We investigate this primarily for comparison with the other algorithms.

- *MinMean.* Here, the listener calculates the mean distance from each unique beacon for the set of data points within the data set. Then, it selects the beacon with the minimum mean as the closest one. The advantage of this algorithm is that it can be computed with very little state, since a new sample updates the mean in a straightforward way. The problem with this algorithm is that it is not immune to multipath effects that cause the distance estimates to

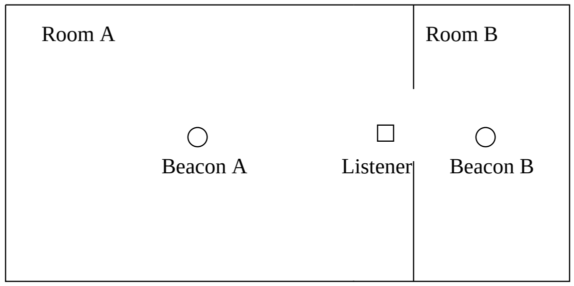

Figure 2-4: The nearest beacon to a listener may not be in the same geographic space.

display modal behavior, where computing a statistic like the mean (or median) is not reflective of any actual beacon position.

- *MinMode.* Since the distance estimates often show significant modal behavior due to reflections, our approach to obtaining a good estimate is to compute the per-beacon statistical modes over the past n samples (or time window). For each beacon, the listener then picks the distance corresponding to the mode of the distribution, and uses the beacon that has the minimum distance value from among all the modes. We find that this is robust to stray signals and performs well in both static and mobile cases.

Section 2.4 discusses the results of our experiments. It should be noted that these algorithms serve as the basic building blocks for implementing other inference algorithms, where these can be combined in different ways to achieve much richer inference algorithms.

2.3 Beacon positioning and configuration

The positioning of a beacon within a room or space plays a significant role in enabling listeners to make the correct choice of their location. For example, consider the positioning shown in Figure 2-4. Although the listener is in room A, the listener finds the beacon in room B to be closer and assumes incorrectly that it is in room B.

One way of overcoming this is to maintain a centralized repository of the physical locations of each beacon and provide this data to listeners. Systems like the Bat

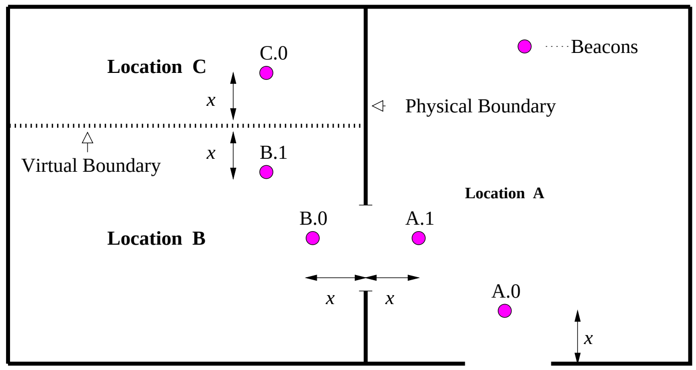

Figure 2-5: Correct positioning of beacons.

essentially use this type of approach, where the central controller knows where each wall- or ceiling-mounted device is located, but it suffers from two problems that make it unsuitable in general pervasive computing environments. First, user privacy is compromised because a listener now needs to make active contact to learn where it is. Second, it requires a centrally managed service, which defeats our goal of building a decentralized system.

There is a simple engineering solution to this problem that preserves privacy and is decentralized. Whenever a space is demarcated by an open boundary, where an open boundary refers to a boundary that lets ultrasound signals go through, the two beacons corresponding to the two spaces separated by the boundary *must* be placed at equal distances from the boundary. Figure 2-5 shows an example of this in a setting with both real and virtual boundaries. The Cricket system is aimed at determining the space a user is in, and the placement of beacons is done to accurately detect the boundaries between such spaces. For spaces with multiple open boundaries such as space B of Figure 2-5, it is necessary to have multiple beacons advertising the location string of a given space. For correctly inferring the closest beacon, the inference algorithms must be able differentiate between individual beacons. Cricket achieves this by using a unique identifier for each beacon within a given space.

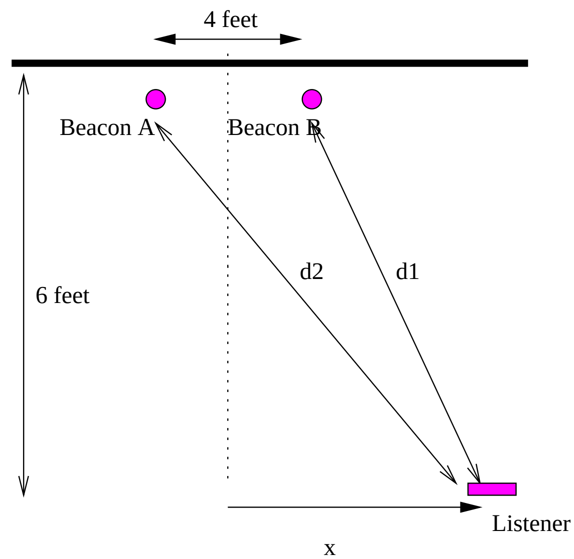

Figure 2-6: Setup for experiment 1, evaluating boundary performance. The dotted line shows the virtual boundary between the spaces advertised by beacons A and B.

2.4 Experimental results

We conducted several experiments to investigate the performance of our Cricket implementation. The first experiment examines the listener performance near location boundaries, and shows that Cricket can detect boundaries accurately. The second experiment investigates the robustness of the system to interference amongst beacons, and evaluates the performance of the MinMode inference algorithm and compares it to two other simple algorithms. The third experiment does the same for the more challenging case of a mobile listener.

2.4.1 Boundary performance

Figure 2-6 shows the setup for this experiment. The aim of this experiment is to investigate the the ability of the listener to detect a boundary, which determines the precision of the system. Two beacons, A and B, advertising different location strings were placed 4 feet apart on the ceiling, giving rise to a virtual boundary in the middle. Distance samples (in the form of ultrasonic pulse propagation time) were taken at 0.5-feet intervals along the x direction as shown in Figure 2-6, starting from the virtual boundary.

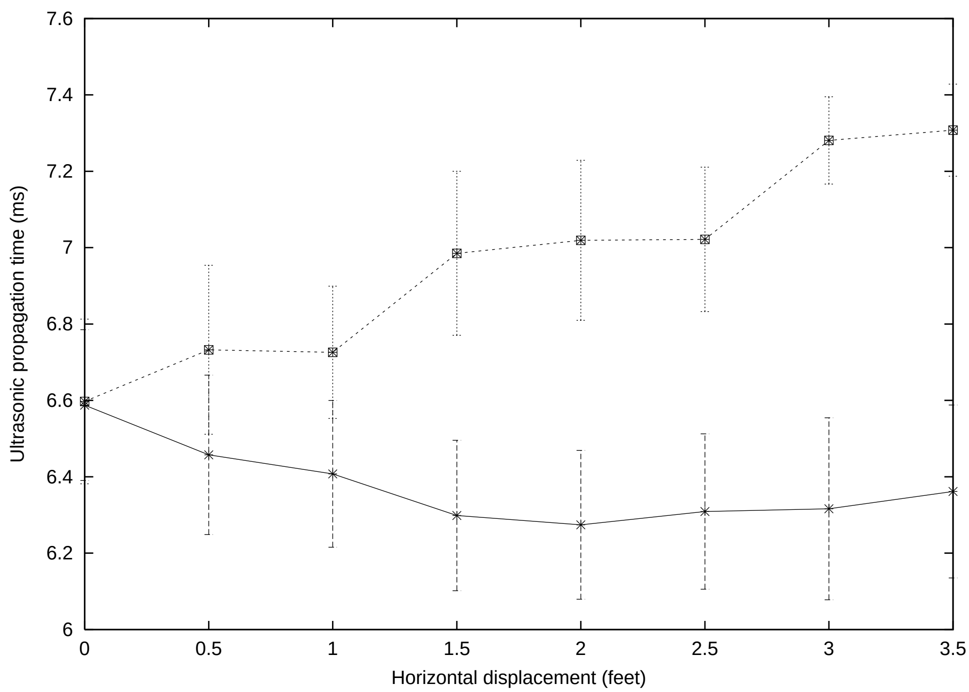

Figure 2-7: Average and standard deviation (the errorbars) of ultrasonic propagation time as a function of the horizontal displacement of a listener from the boundary of two beacon regions. When the displacement is over about 1 foot, the errorbars do not overlap.

Figure 2-7 shows the results of this experiment, plotting the mean and the standard deviation of the ultrasonic propagation times from the two beacons as a function of the displacement from the boundary x. This shows that when the listener is more than about 1 foot away from the boundary, the closest beacon can be determined accurately from the estimated distances, thus enabling the listener to determine its location accurately. Furthermore, the difference of the two average distances increases as the listener moves away from the boundary, which causes the probability of making a wrong decision by the listener to decrease as it moves away from the boundary.

2.4.2 Static performance

In the second experiment, we examine the robustness of Cricket against interference from nearby beacons. The results of the experiment show that it is indeed possible to achieve good system performance, despite the absence of any explicit coordination amongst the beacons. We also compare the performance of the three listener inference

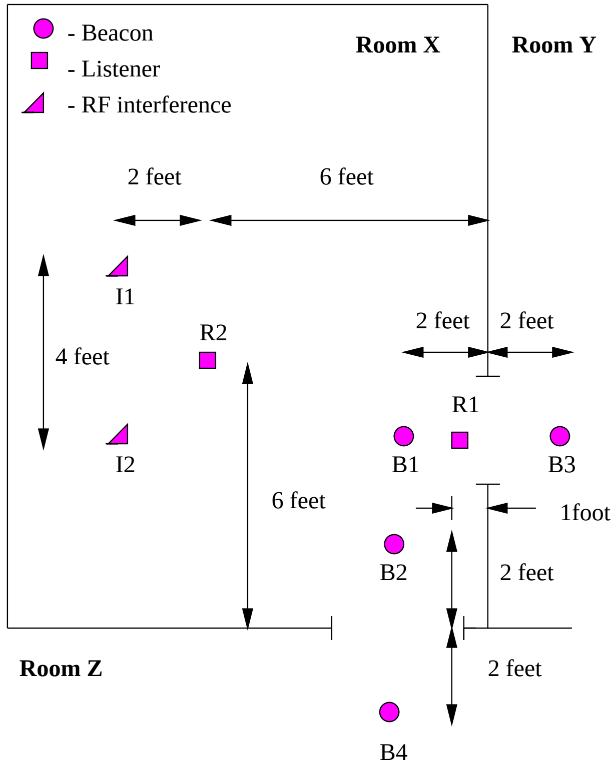

Figure 2-8: Setup for experiment 2, evaluating the robustness of Cricket in the presence of interfering beacons.

algorithms presented in Section 2.2.3.

Figure 2-8 shows the setup for this experiment. Beacons $B1$ and $B2$ provide location information within room X. Beacons $B3$ and $B4$ provide location information for rooms Y and Z. The ultrasonic transmissions of all these beacons are within range of each other. To provide RF interference with no corresponding ultrasonic signals (since the range of RF exceeds that of ultrasound in Cricket), we use beacons $I1$ and $I2$, which have their ultrasonic transmitters disabled.

All the beacons were attached to the ceiling with the ultrasonic transmitters facing their respective spaces. We gathered distance samples at locations $R1$ and $R2$ for a static listener. Observe that $R1$ is closer to the interfering sources $I1$ and $I2$ than to the legitimate beacons for the room, which models the presence of severe RF interference. In contrast, $R2$ is only 1 foot away from the boundary separating the rooms X and Y, showing the performance close to a boundary.

First, we determined the degree of interference caused by $I1$ and $I2$ by collecting

Interference Source	I1	I2
Interference at R1	0.0%	0.0%
Interference at R2	0.3%	0.4%

Table 2.1: Degree of interference at $R1$ and $R2$ caused by $I1$ and $I2$, showing the effectiveness of the randomized beacon transmissions and system parameters.

1000 samples of distance estimates at $R1$ and $R2$ and counting the number of values corresponding to each RF source (beacon or interferer). When the listener was at $R1$, somewhat farther from the interfering sources, there were no distance samples corresponding to the interfering RF sources. On the other hand, at $R2$, we received a total of only 7 samples corresponding to both $I1$ and $I2$, despite the fact that $R2$ is closer to $I1$ and $I2$ relative to the legitimate beacons. Table 2.1 summarizes these results.

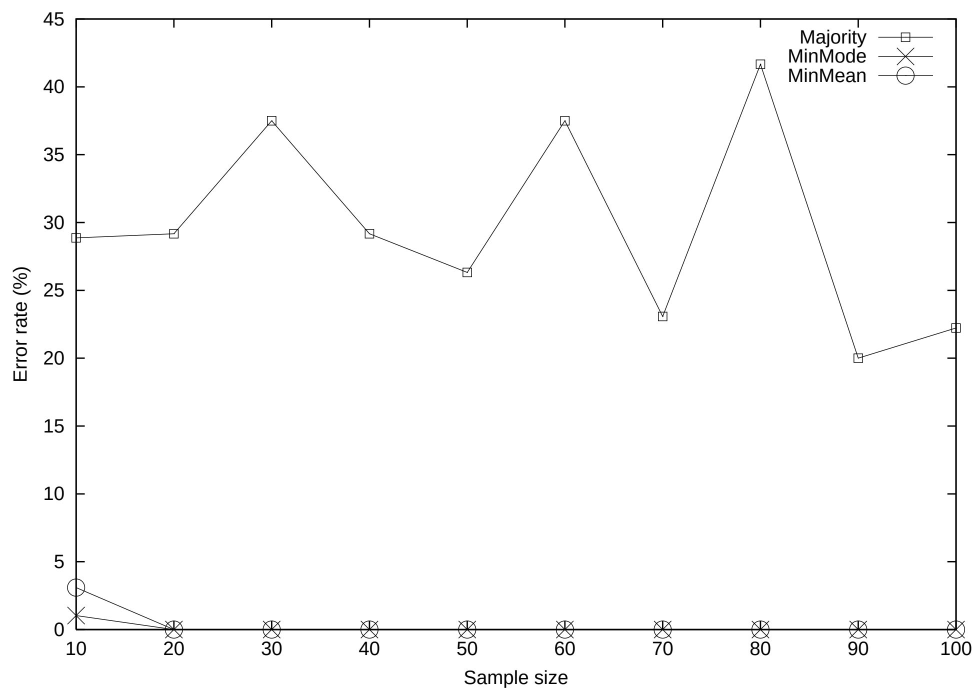

Figure 2-9: Error rates at Position 1 as a function of the sample size.

The samples corresponding to $I1$ and $I2$ are due to the incorrect correlation of these RF signals with ultrasonic pulses from other beacons in the vicinity of the listener. However, the randomized transmission schedule together with the well-engineered system parameters reduces the occurrence of such interference to a very small fraction of the total. This validates our claims in Section 2.2 and our design choices.

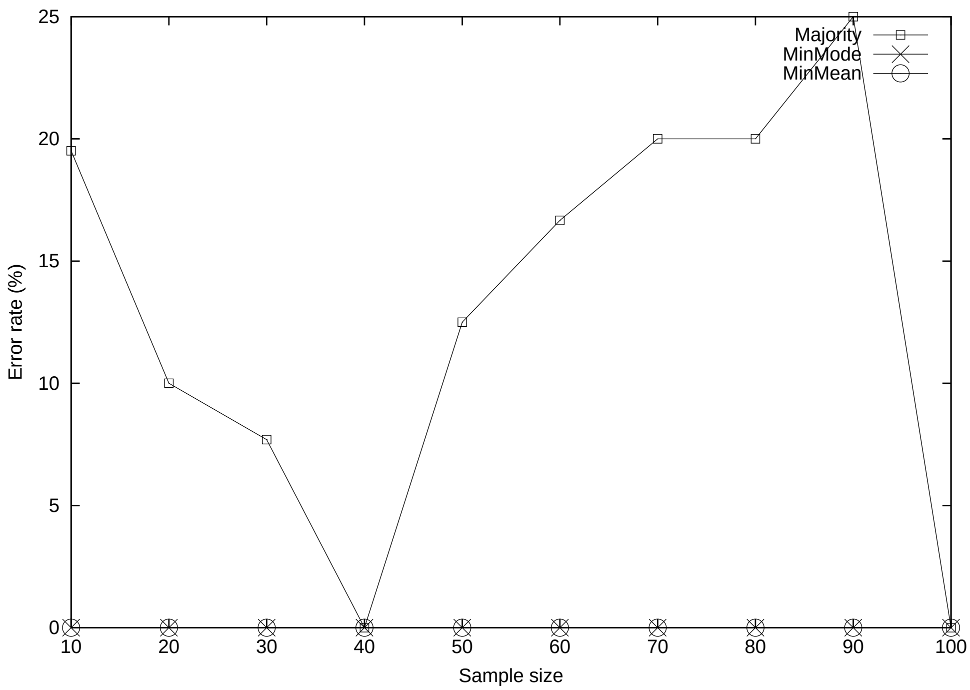

Figure 2-10: Error rates at Position 2 as a function of the sample size. The error rates for both MinMean and MinMode are zero.

We now investigate the performance of the three inference algorithms, *Majority*, *MinMean*, and *MinMode*, when the listener is at $R1$ and $R2$. We calculate the error rate (in percent) in inferring the location using these three inference algorithms, varying the number of distance samples used for inference. The results, shown in Figure 2-9 (for position $R1$) and Figure 2-10 (for position $R2$), demonstrate that both *MinMean* and *MinMode* perform very well even when the sample size is small, and even when a listener ($R1$) is close to a boundary.

2.4.3 Mobile performance

This experiment is aimed at determining the system performance when the listener is mobile. For a mobile listener, being able to obtain accurate location information within a short time (say, a few seconds) is important. Figure 2-11 shows the

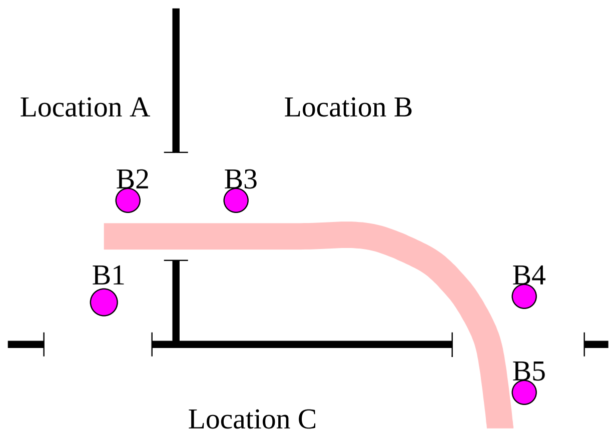

Figure 2-11: Setup for experiment 3, evaluating the mobile performance of Cricket.

configuration of the beacons and the path followed by the mobile user while taking measurements. The listener was moved through each boundary at approximately the same speed each time, emulating a user's typical walking speed in a building. Each time the listener crossed a boundary, a transition event and a timestamp was logged. Once through the boundary, the listener remained stationary for a short period of time to determine how long it takes to stabilize to the correct value, and then the experiment was repeated again through the next boundary. When analyzing the data, we used the logged transition event to determine the user's actual location with respect to the location being reported by the listener. Note that in this experiment, the listener is always located relatively close to the boundaries.

Figure 2-12 shows the location error-rate at the listener for the experiment. The error-rate is calculated over the time period during which the listener moves around a location, after crossing a boundary. The *MinMode* performs the best among the three inference algorithms. From the results, it is evident that larger time intervals provide better results over smaller intervals, which is not surprising since a larger interval gives the algorithm more samples samples to work with. Another interesting point is that *MinMean* and *MinMode* both perform about the same over small time windows. As the time interval gets smaller the probability that a distance value sample containing only a single value per beacon increases. A small number of samples therefore causes both the mean and the mode to be the roughly the same.

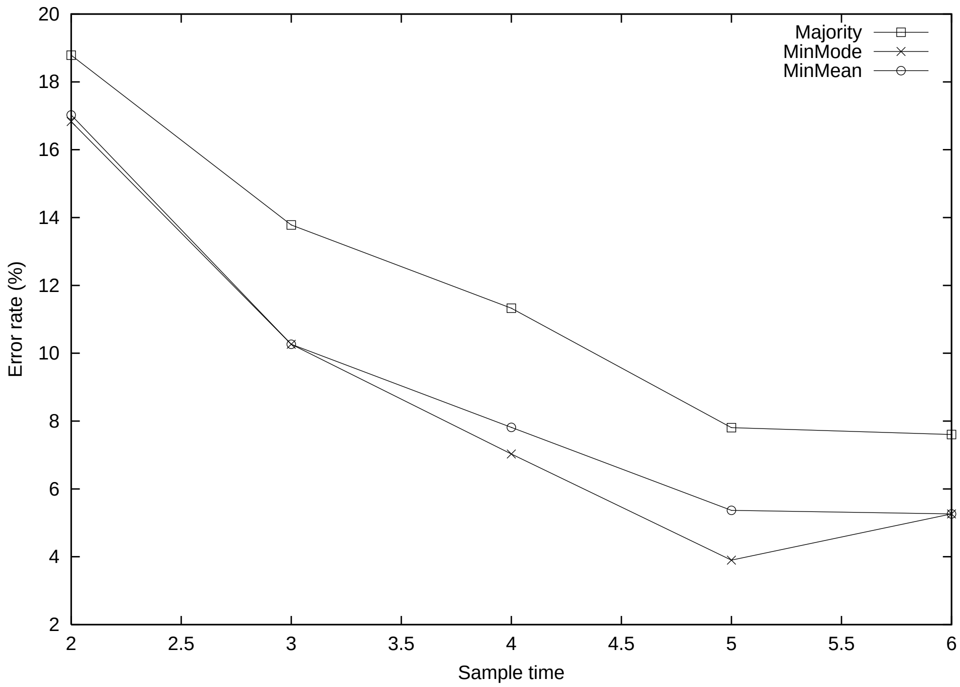

Figure 2-12: Error rates for a mobile Cricket listener as a function of the amount of time (in seconds).

To summarize, by measuring the distance individual beacons and by correct placement of beacons, Cricket enables the identification of a users current space by accurately detecting the boundary between spaces. Unlike traditional location systems that use centralized and tightly coordinated beaconing or querying to track objects, Cricket uses uncoordinated and independent beacons transmissions to provide location information. The combination of three simple mechanisms - bounding stray interference using well engineered parameters, randomized transmissions, and inference algorithms - progressively reduces the effects of the adverse inter-beacon interactions caused by the lack of coordination. Our several experiments showed that the system is in fact robust against such interactions.

Chapter 3

RF channel utilization

Chapter 2 showed that the Cricket system can operate successfully and reasonably accurately with uncorrelated beacon transmissions. It described how Cricket manages to perform correctly in the presence of inter beacon interactions caused by the lack of explicit coordination.

Although correctness is ensured, the system responsiveness in terms of the rate at which location inference can be achieved at a listener is dependent on the degree of collaboration among beacons within close range. This is due to the dependency of RF channel utilization on the nature of beacon transmission interactions. A low channel utilization results in a smaller number of distance samples at a listener for a fixed time. This could result in the poor performance of listener inference algorithms in terms of both the delay and the accuracy, due to the listener using only a small number of samples in the analysis. At low beacon transmission frequencies the channel utilization drops because the channel is idle for most of the time. On the other hand, a very high transmission rate leads to a large number of wasteful collisions, which again reduces the channel utilization.

When discussing channel utilization, it is important to draw a distinction between the RF channel utilization perceived by a beacon and that perceived by a listener. This is due to the fact that, in general, the set of a beacons within the RF range of a given beacon is different from that of a listener. As an example, in Figure 3-1 the channel utilization perceived by beacon A only depends on the transmissions

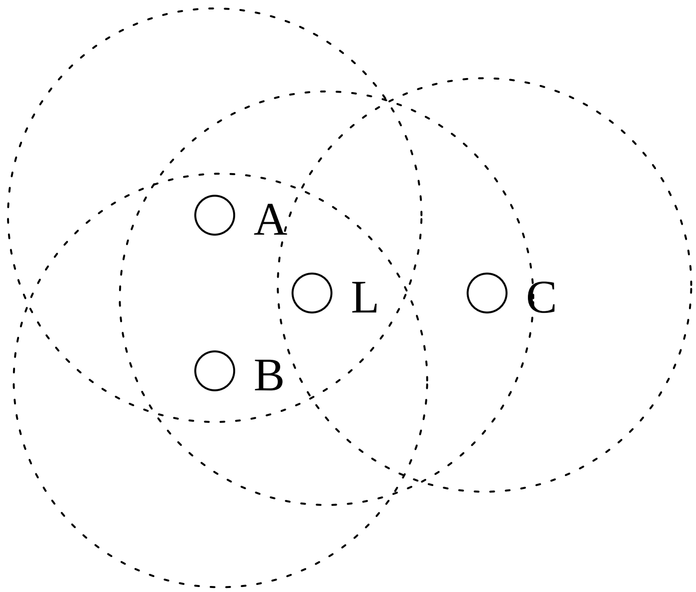

Figure 3-1: Different channel utilizations at the beacons and at a listener.

from B and A itself, while the channel utilization at the listener L depends on the transmissions from all three beacons, A, B, and C. It should be noted that this cannot be solved simply by having a set of uniformly distributed beacons and equal ranges for RF receivers at both beacons and listeners. The utilization depends on the beacon transmission pattern as well, where two beacon transmissions colliding at the listener may not be detected by either of the beacons due to the classical "hidden terminal problem" in wireless communication [23]. Unless otherwise specified, we always use the term "utilization" to refer to the utilization seen at the listener.

This chapter analyzes the dependency of the RF channel utilization on beacon transmission profile using analysis and simulation.

3.1 Randomized transmissions without carrier-sense

For this analysis we assume the following beacon transmission scheme. Each beacon sends a message of fixed duration (transmission time) and these transmissions are separated by a random interval I, which is uniformly distributed between two values I_l and I_h. There are n beacons within the listener's range. We neglect the effects of RF propagation delay since this is several orders of magnitude smaller than the other time intervals under consideration.

Define:

I_e = *expected value of the interval between two transmissions*

T_x = *message duration (transmission delay)*

The probability that a given beacon is transmitting at a given instance:

$$p = \frac{T_x}{I_e}$$

For a successful transmission exactly one of the n beacons should be transmitting at a given time. The probability that only one given beacon is transmitting at a particular time,

$$P_1 = p(1-p)^{n-1}$$

When there are n beacons, the probability that exactly one beacon is transmitting at a given time is the channel utilization of the RF channel, U.

$$U = np(1-p)^{n-1}$$

For maximum utilization;

$$\frac{dU}{dp} = n(1-p)^{n-1} - np(n-1)(1-p)^{n-2} = 0$$

$$\Rightarrow p = \frac{1}{n}$$

$$\Rightarrow I_e = nT_x$$

If we assume that there are n beacons within the radio range of a listener above, this analysis predicts the perceived utilization at the listener. The optimum utilization U_{opt} is given by:

$$U_{opt} = (1 - \frac{1}{n})^{n-1}$$

It can be easily shown that this is a global maximum for the channel utilization. Figure 3-2 plots U_{opt} against the population size n. It can be seen that this drops

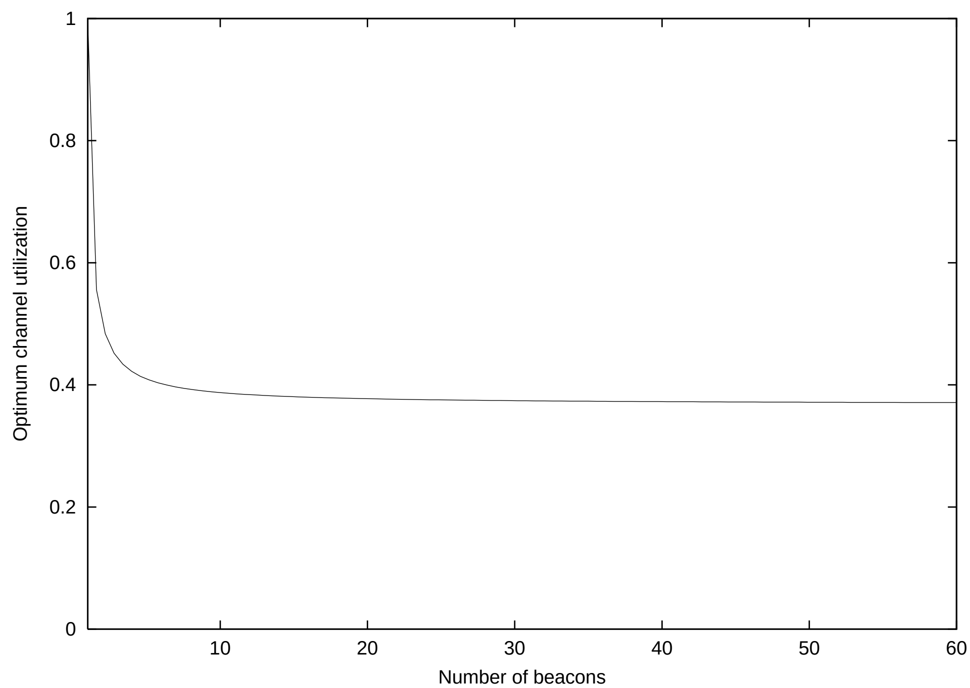

Figure 3-2: Variation of U_{opt} with n.

quickly with increasing n and then slowly decreases from 0.41 to 0.37 as n varies from 5 to 60. A value of $n = 60$ corresponds to a beacon per every 2 square meters with a radio range of ≈ 6 meters, which is representative of the high end of beacon densities in most practical situations. Hence it is safe to assume that the RF channel utilization will be around 0.37 for this transmission scheme. The theoretical limit of U_{opt} as $n \to \infty$ is $\frac{1}{e}$, or 36.8%.

Although this scheme results in reasonably good utilization of the channel, several practical reasons stand in the way of adopting it in Cricket. First, the users of the Cricket system are free to configure any text string as the message, making it difficult to use a single value to represent T_x, thus making the proper selection of I_e difficult. Second, the need for decentralized and incremental deployment of beacons prevents the use of a fixed value for n.

One solution would be to enable beacons to dynamically adjust their transmission rate to satisfy the above condition. Assuming a relatively uniform beacon distribution, and attaching an RF receiver to the beacons (similar to the RF receiver used by the listener), we can enable the beacons to determine the number of different beacons

$\hat{n}$ they hear from. $\hat{n}$ can be used as an approximation to n, the number of beacons a nearby listener hears from. The beacon can also calculate the average length of messages it hears from $\hat{T}_x$, an approximation to T_x, and set the mean transmission interval as $\hat{I}_e = \hat{n}\hat{T}_x$.

Continuously monitoring the RF channel by the beacon can consume a significant amount of energy. The need for continuous monitoring of the channel can be avoided by statistically sampling it, since the number of beacons and their message sizes will be relatively stable over reasonably long periods of time in any given region.

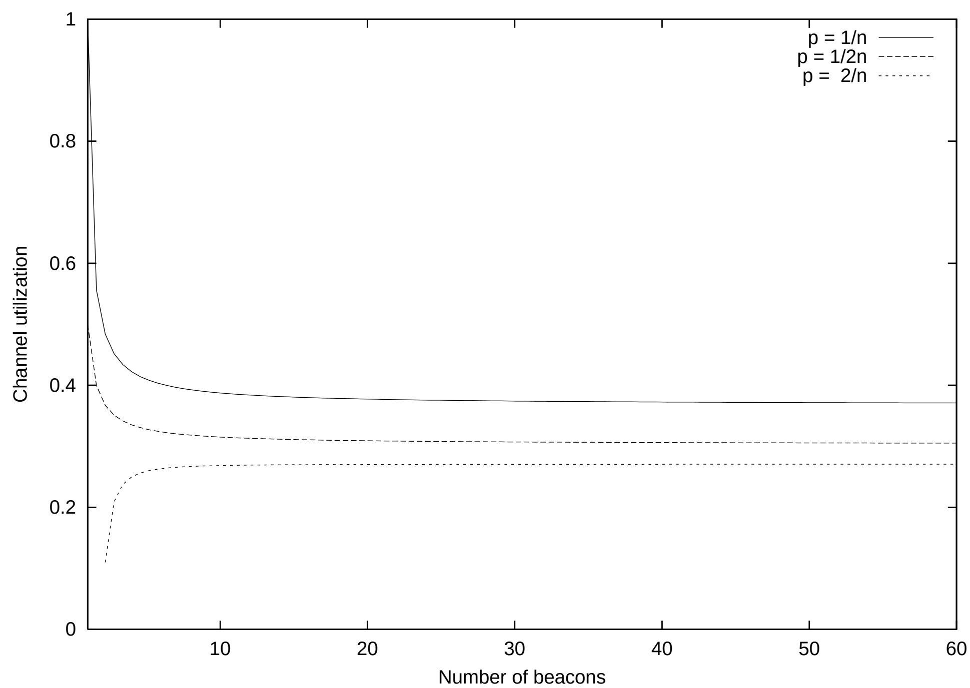

Figure 3-3: Channel utilization for different transmission rates.

However, the use of statistical sampling to determine the number of beacons might make it hard to achieve the above-mentioned optimum transmission rate per beacon. Figure 3-3 shows the effects of using a sub-optimal transmission rate. Specifically, this compares the expected channel utilization achieved when the average transmission rate is half and twice that of the optimum value. This indicates that even under such suboptimal transmission rates, it is possible to achieve a significant ($\approx$ 27% compared to 37% for $n = 10$) channel utilization. This also shows that penalty for underestimating the population size ($p = \frac{2}{n}$) is higher than that for overestimating it

$(p = \frac{1}{2n})$.

3.2 Randomized transmissions with carrier sense

Another possible scheme is to use *RF carrier sense* to prevent collisions between nearby beacon transmissions. We consider the following transmission algorithm at each beacon. Each beacon periodically checks for the presence of the RF carrier, and transmits its data only if the carrier is absent. A random variable is used as the interval between successive transmission attempts to prevent undesirable interactions among beacons.

We investigate the behavior of a set of beacons following the above simple transmission scheme, in terms of both the RF channel utilization and the degree of RF collisions. First, we consider the situation where all the beacons are within each other's range and where the beacons have perfect carrier sensing ability. This enables us to investigate the performance of the algorithm under highly favorable conditions.

In practice, however, it is almost impossible to achieve perfect carrier sensing even when beacons can hear each other's transmissions. There is a certain duration, called *detection threshold*, during which a beacon fails to detect the transmission from another beacon. This would normally be a fraction of a bit-period and is related to the maximum bit-rate at which an RF receiver can receive data. Therefore, we also examine the performance of the above algorithm under imperfect carrier sensing mechanisms. Then we relax the assumption that all beacons are within each other's range, and examine the RF channel utilization when the above transmission scheme is adopted when all the beacons are not within each others range.

3.3 Simulation results

The different beacon transmission schemes were simulated using the ns-2 network simulator with suitable extensions [19]. Our extensions to the simulator were guided by the CMU wireless extensions to ns-2 [18].

3.3.1 Randomized transmissions without carrier-sense

Here, each beacon follows the transmission algorithm shown in Figure 3-4, where each beacon waits a time interval R, which is randomly distributed between T_1 and T_2.

```
loop{
    r ← Uniform[T1,T2]
    delay(r)
    xmit(RF,US)
}
```

Figure 3-4: Randomized beacon transmission algorithm without carrier sense.

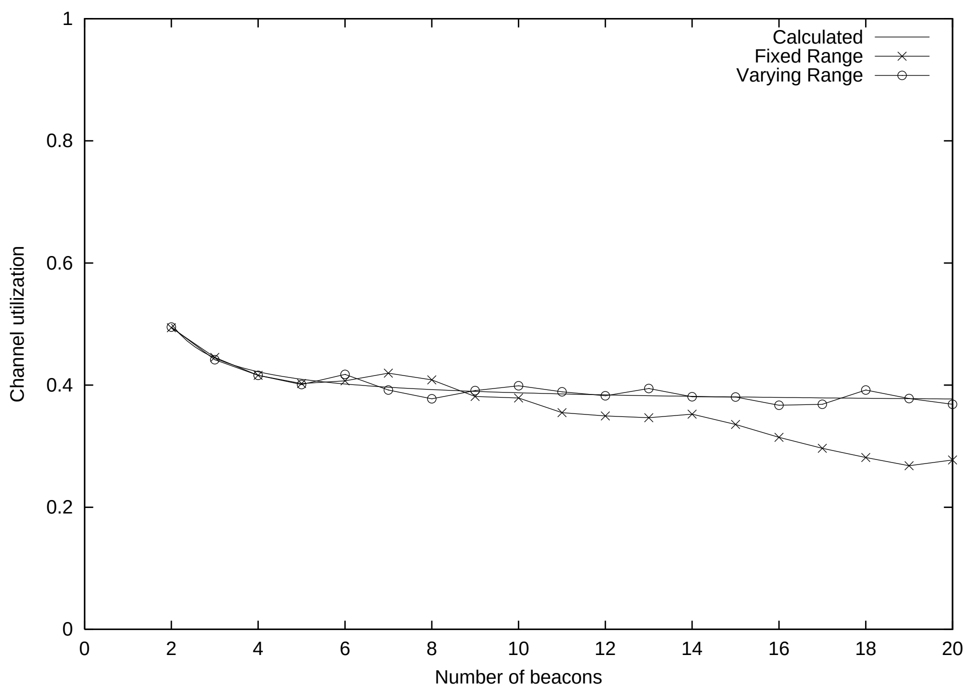

Figure 3-5: Simulation results for randomized transmissions without carrier sense.

Figure 3-5 plots the channel utilization for different assumptions of values R and T_1. The graph named "calculated" plots the optimum channel utilization as obtained in Section 3.1. The graph "fixed range" plots the channel utilization when T_1 is varied, while keeping the value of R fixed at 100ms, such that the expected time between transmissions is equal to the value of T_{opt} as derived in section 3.1. The other graph, "varying range", plots the channel utilization when we vary both T_1 and R such that the expected interval between transmissions is the optimum value T_{opt} and the value of $R = n \times 50$ms, where n is the number of beacons. The random value generation is implemented to reflect the actual implementation within the Cricket beacons which have limited processing power and memory. We use an array of 100 elements that is initialized with random integers uniformly selected between the range T_1 to T_2, and the beacon traverses this array one at a time for each data transmission. The simulations are run for 100 simulated seconds and a fixed message duration of 50ms is used.

From Figure 3-5, we can see that the graph corresponding to varying random range follows the curve representing optimum channel utilization more closely than the graph with fixed random range. Since the optimum utilization curve always achieves a channel utilization greater than 36.8%, by selecting T_1 and R to reflect the average number of beacons within a listener's range, we can always achieve a reasonably good channel utilization.

3.3.2 Randomized transmissions with carrier sense

For the carrier-sensed transmissions, the beacons follow the pseudocode given in Figure 3-6. The algorithm is almost identical to the no-carrier sense scheme described above, other than that each beacon check for the presence of carrier before transmitting data. A beacon transmits data only if the carrier is not detected. As before, we implement the random number generator as a fixed-length array of integers.

```
loop{
    r ← Uniform[T1,T2]
    delay(r)
    if no_carrier()
        xmit(RF,US)
}
```

Figure 3-6: Randomized beacon transmission algorithm with carrier sense.

Figure 3-7 plots the channel utilization for n beacons following the above algorithm, when all the beacons are within each other's range. The graph "perfect cs" plots the channel utilization when the carrier sensing is perfect, while the graph "imperfect cs" plots the channel utilization when there is a carrier detection threshold of duration 2% of the message duration. The simulation is done for 100 simulated seconds and a message length of 50ms.

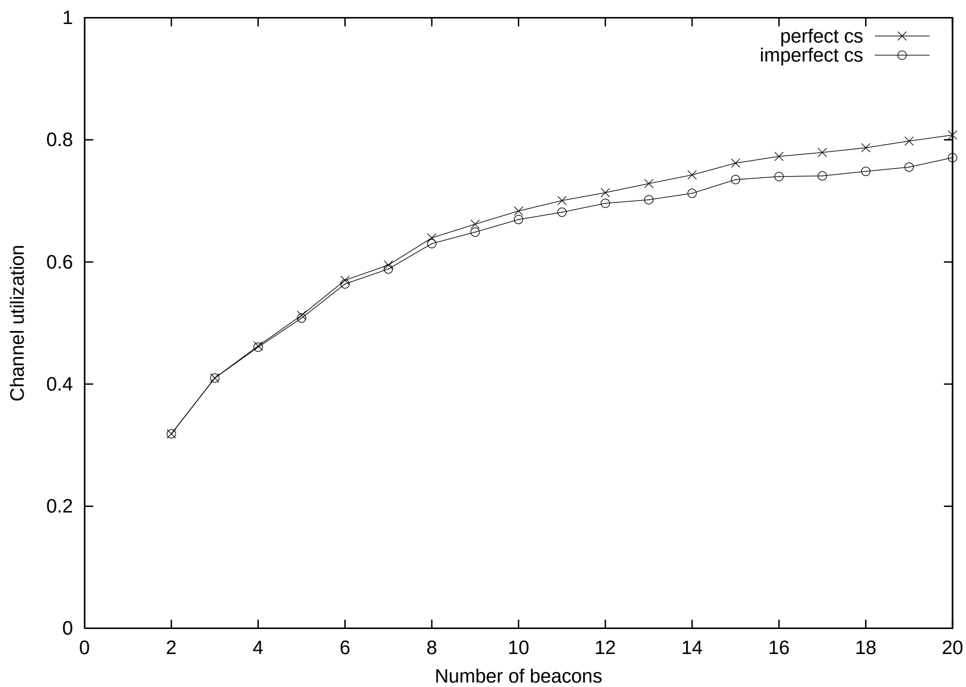

Figure 3-7: Channel utilization results with carrier sense when all the beacons and listeners are within each other's range.

Figure 3-7 shows that it is possible to achieve very high channel utilization when there are enough beacons that are actively trying to transmit data, and with a large number of beacons, it is possible to reach almost 100% channel utilization. However, the detection threshold causes the channel utilization to drop as expected due to increased collisions, and the effects of this increases with the number of beacons in the vicinity. Hence, it is not possible to continuously improve the utilization by just increasing the number of beacons or the aggressiveness of beacon transmissions, since it is impossible implement perfect carrier detection.

Next, we consider the situation where beacons are uniformly spread across a region, and the beacons and the listeners have a *limited* radio range. In the simulations, we overcome the edge effects by using a matrix of 100×100 nodes (a total of 10000 nodes) and determining the channel utilization at a listener in the middle of this matrix.

Figure 3-8 plots the RF channel utilization as we vary the number of beacons within the *listener's* range. We consider two cases: "Case 1" consider beacons and the listeners having the same radio range, while "Case 2" assumes that the RF range of beacons is *twice* the range of listeners. For Case 2, under assumptions of perfect carrier sensing and ideal RF propagation conditions, there are no RF collisions at the listener. Also, for "Case 2", the number of beacons in each other's range is approximately four times the number of beacons within a listener's range.

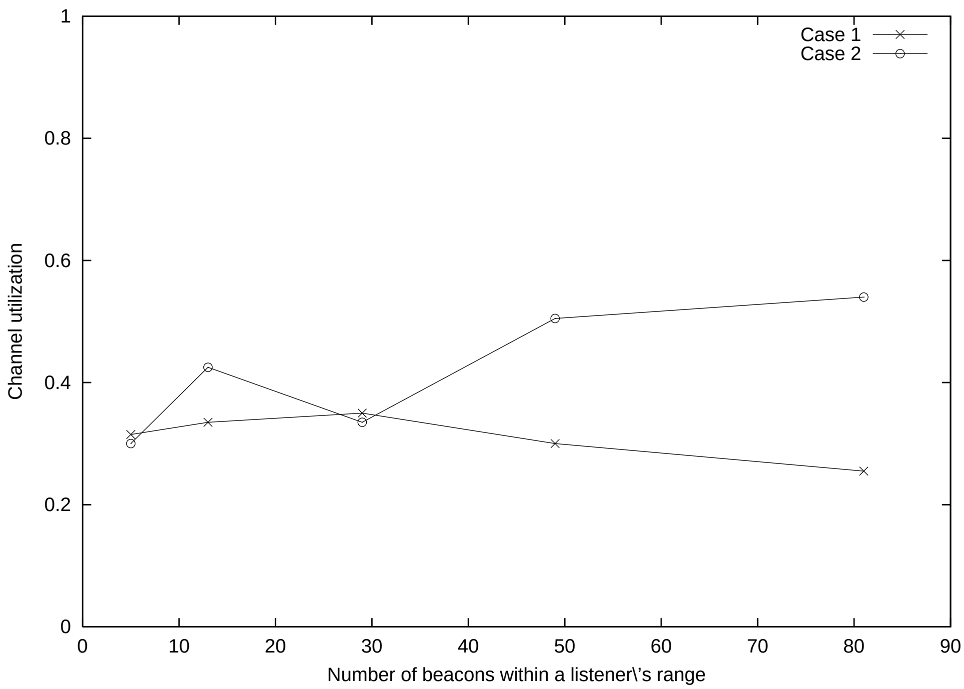

Figure 3-8: Channel utilization with carrier sense under limited RF range for beacons and listeners.

We can observe that this scheme also achieves good channel utilization, and that a listener with smaller RF range performs better than a listener with a larger RF range. In Cricket, we use the scheme based on carrier-sensing, because of the ease of implementation and the reduced number of collisions at the listener.

Chapter 4

Implementation

This chapter describes some salient features of our implementation of the design described in the previous chapters, and discusses the implementation of an active map application.

4.1 Protocols and packet formats

This section explains the beacon transmission protocol in more detail, and presents the various packet formats used in Cricket.

4.1.1 Beacon transmission protocol

Figure 4-1 shows the algorithm used by the beacon. The beacon reads random values from an array of integers, `Random[]`. During `sleep(d)`, the beacon enters a low power mode and sleeps for a duration `d`. Before each transmission, the beacon checks for the RF carrier, and transmits data only if the carrier is free.

After R transmissions, the beacon indicates that it is willing to change the advertised location string by transmitting a special message. A controller that is trying to change the location string associated with a given beacon, detects this message and send a special control message. The beacon, after transmitting the special message, waits a small duration for the arrival of such a control message.

```
i ← 0;
loop{
   d = Random[i];
   sleep(d);
   if no_carrier()
      xmit(RF,US);
      if (i%R == 0)
         xmit(ready_to_change);
         if control_msg();
            change_location();
   i ← (i+1)%M;
}
```

Figure 4-1: The Cricket beacon transmission algorithm. R is the periodicity with which the beacon listens for changes for its advertised space. M is the number of entries in its random array. M≫R.

4.1.2 Packet formats

SYN1	SYN1	SD	Text Strin	FIN	UID	Parity

Figure 4-2: Beacon data packet format.

Figure 4-2 show the format of the data packet transmitted by the beacon. This starts with two `SYN1` characters followed by a `DS` character. The text string corresponds to the string of 7-bit printable `ASCII` characters advertised by the beacon, set to even parity. The field separator, `FIN`, separates this string from the unique identifier `UID` of the beacon. The `UID` is 7 bit binary value with an offset of 17; the offset prevents the misinterpretation of the `UID` as a control character, and also has even parity. A block parity byte, which is calculated over the data packet excluding the `SYN1` and `DS`, is appended after the `UID`.

SYN1	SYN1	SD	Text Strin	FIN	UID	Parity	Wait

Figure 4-3: Special data packet from the beacon.

Figure 4-3 shows the format of the special data packet transmitted by the beacon to indicate it's readiness to accept a control packet. This is identical to the conventional data packet other than the `WAIT` character added at the end, and this character will be neglected by the conventional listeners. However, the controller interprets this byte and reacts by sending location-change control packet if it is trying to change the string advertised by the beacon. Unlike the other characters, the `WAIT` character has an odd parity.

SYN2	SYN2	CS	Current String	FIN	UIDc	Parity	New string	FIN	UIDn	Parity	Password

Figure 4-4: Format of the location change message from controller.

Figure 4-4 shows the format of the control packet sent by the controller to the beacon. The packet starts with two `SYN2` characters followed by a `CS` character. This is followed by the current text string, the `UID`, and the block parity byte of the beacon. A beacon receiving a control packet will compare these fields with current values, one byte at a time, as the packet is read in. If there is any discrepancy, the beacon neglects the packet. The rest of the packet contains the new text string, the `UID`, and the block parity packet, followed by a password. If the block parity is correct and the password matches the beacon's internal password, it starts transmitting the new information. A controller must repeatedly try changing the string until it succeeds. It is possible for the beacon to change its location to a different value than the one requested by the controller if any bit errors caused in the new information is not detected by the block parity check. The controller must detect this and continue to issue control messages until the information is updated correctly.

All the control characters used in RF data packets are selected such that their parity bit is clear when even parity is used, to enable the easy distinction of control characters from other characters in the data packets.

STX	ASCII text string	FIN	UID	FIN	dis. MSB	dis. LSB	ETX

Figure 4-5: Format of data sent from listener to the attached host.

Figure 4-5 shows the format of the location data transmitted from the listener to its attached host. The packet consists of the `STX` character, the `ASCII` string received from the beacon, a `FIN` character, the beacon `UID`, another `FIN` character, two bytes (`dis.MSB`, `dis.LSB`) representing the distance to the beacon, and the `FIN` character

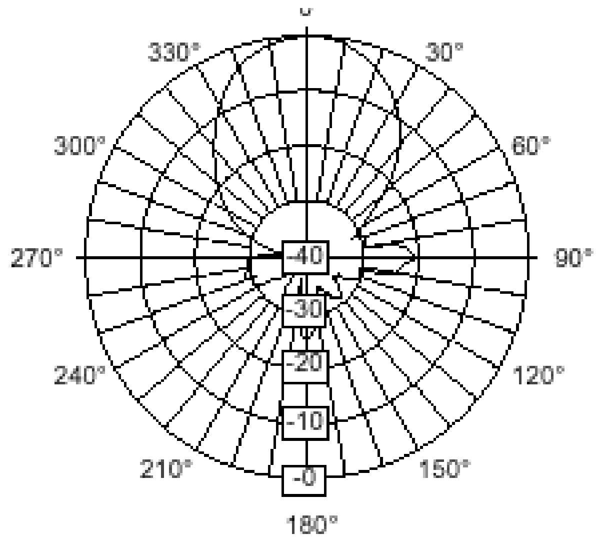

Figure 4-6: The radiation pattern of an ultrasonic transmitter.

indicating the end of the packet. When the attached host looks for the start of the packet, it looks for the (ETX,STX) sequence. This is necessary to prevent the distance information, in the form of a binary value, from being mistakenly interpreted as either STX or ETX. If the (ETX, STX) combination occurs in (dis.MSB, dis.LSB), dis.LSB is incremented by one to prevent any misinterpretations. The percentage error caused to the distance by this adjustment is negligible. This information is provided to the attached host across a standered RS-232 interface.

Control Character	SYN1	DS	FIN	WAIT	SYN2	CS	STX	ETX
Hex value	0x0F	0x09	0x011	0x1f	0x0C	0x0A	0x01	0x03

Table 4.1: Equivalent Hex values of the various control characters used in Cricket.

Table 4.1 lists the various control characters used and their equivalent hex values.

4.2 Ultrasound deployment issues

As described in Section 2.2, ultrasonic interference at the receiver can lead to incorrect distance measurements. It is therefore important to reduce ultrasonic leakage to other locations while trying to provide full coverage to the location served by a Cricket beacon. We achieve this by proper alignment of the ultrasonic transmitters.

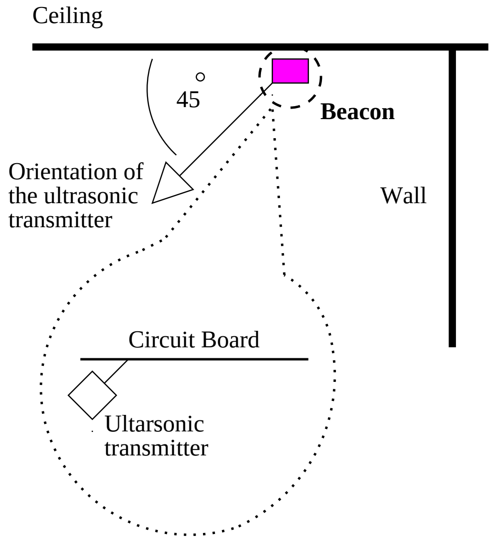

Figure 4-7: Correct alignment of a Cricket ultrasonic transmitter.

Figure 4-6 shows the radiation pattern of the ultrasonic transmitter used in the Cricket beacons. This is shown in (r, θ) polar coordinates, where r corresponds to the signal strength in dB and θ corresponds to the offset in degrees from the front of the ultrasonic transmitter. From the radiation pattern, it can be seen that the direction the ultrasound transmitter faces ($0°$) has the maximum signal strength, while the signal strength drops to 1% (-20 dB) of the maximum value at $\pm 50°$ away from the $0°$ direction.

We align the ultrasonic transmitter such that the direction of its peak signal strength is at $45°$ to the horizontal. The beacon is mounted such that the ultrasonic transmitter faces the location intended to be covered by the beacon. This causes the amount of ultrasonic energy transmitted towards distant locations to be small compared to where it is intended. This alignment is easily accomplished by positioning the transmitter at an angle of $45°$ to the circuit board of the beacon and mounting the board flat on the ceiling or wall of the room, as shown in Figure 4-7.

Another interesting observation we made during the deployment of Cricket is the discovery of various interfering ultrasonic sources present in the environment. Cricket uses 40KHz ultrasonic signals, and some equipment such as faulty fluorescent lamps, computer cooling fans, and air-conditioning equipments generate various amounts of ultrasonic signals in this range. For successful deployment of Cricket, it is necessary to reduce such interferences as much as possible. One solution might be to use an adaptive ultrasonic receiver at the listener that can adapt the sensitivity according the amount of ultrasonic signals present, but it is hard to avoid problems due to strong ultrasonic interference.

4.3 Applications

Cricket provides location information to applications developed using the Intentional Naming System (INS), which handles service discovery and device mobility within the naming system [1, 2]. The Floorplan application, an active map navigation utility, was implemented using the building blocks and services provided by INS [6]. Cricket is currently being used as the location service in the INS environment [22] and in MIT's Project Oxygen [20].

4.3.1 Using virtual spaces in INS

INS uses the concept of a virtual space (vspace), which is a collection of applications/services that can communicate with each other [17]. Each vspace has a set of name resolvers that resolves name requests for entities in that vspace; each entity is described using an *intentional name*, which is a hierarchical collection of application-defined attributes and values.

The overhead for creating a vspace in INS is small. For our location-dependent applications, we create a vspace for every location of interest (e.g., a room or a floor of a building) and identify it by a string. Each beacon advertises the name of the vspace of the corresponding location, and each listener uses this name to bootstrap into its environment by contacting INS and learning about the other existing services in that vspace.

Users and devices can also register their intentional names with the vspace for that location, which enables other entities in that vspace to detect their presence. This way the user can easily determine all the services that are located in their vspace. A user does not necessarily have to be limited to only one vspace at a time, and can select arbitrary services to use. For example, one vspace can correspond to the set of printers in a building while another corresponds to the services located on a specific floor. A user can determine the least loaded printer by querying the printer vspace, or the physically-closest, least-loaded printer by querying the vspace representing the particular floor of the building.

4.3.2 Floorplan application

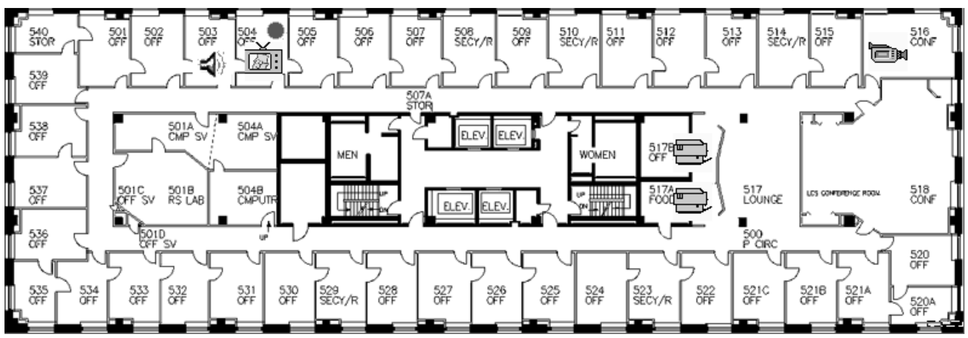

Figure 4-8: The Floorplan active map interface.

The Floorplan is an active map navigation utility that uses Cricket and a map server to present a location-dependent "active" map to the user, highlighting her location on it as she moves. It also displays the set of services that are located in the vicinity of the user, which are dynamically updated as the user moves. Floorplan loads map images from the map server, which also provides the values of (x, y) coordinate on the map corresponding to the user's current vspace position. As the user moves around the building, the listener infers its location and asks the map server to provide the location on the map. Floorplan also learns about various services in the vspace, and contacts those services and downloads a small icon representing each service. These icons are displayed on the map; when the user clicks on an icon, Floorplan uses INS to download a control script or program for the application represented by that icon, and load the controls into a new window so the user can control the application. Figure 4-8 shows an active map displayed by Floorplan; we see that the user (represented by the dot) is in room 503. It also displays four services it has found in the environment,:an MP3 service (represented by the speaker icon) in room 503, a TV service (represented by the TV icon) in room 504, and two printers (represented by the printer icons) in room 517. Using this, a user with no knowledge of her environment or software to control services within it can bootstrap herself with no manual configuration.

Bibliography

[1] W. Adjie-Winoto. A Self-Configuring Resolver Architecture for Resource Discovery and Routing in Device Networks. Master's thesis, Massachusetts Institute of Technology, May 2000.

[2] Adjie-Winoto, W. and Schwartz, E. and Balakrishnan, H. and Lilley, J. The design and implementation of an intentional naming system. In *Proc. ACM Symposium on Operating Systems Principles*, pages 186–201, Kiawah Island, SC, December 1999.

[3] P. Bahl and V. Padmanabhan. RADAR: An In-Building RF-based User Location and Tracking System. In *Proc. IEEE INFOCOM*, Tel-Aviv, Israel, March 2000.

[4] G. Banavar, J. Beck, E. Gluzberg, J. Munson, J. Sussman, and D. Zukowski. An Application Model for Pervasive Computing. In *Proc. ACM MOBICOM*, Boston, MA, August 2000.

[5] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less Low Cost Outdoor Localization For Very Small Devices. Technical Report 00-729, Computer Science Department, University of Southern California, April 2000.

[6] A. Chakraborty. A Distributed Architecture for Mobile, Location-Dependent Applications. Master's thesis, Massachusetts Institute of Technology, May 2000.

[7] S. Czerwinski, B. Zhao, T. Hodes, A. Joseph, and R. Katz. An Architecture for a Secure Service Discovery Service. In *Proc. ACM/IEEE MOBICOM*, pages 24–35, Seattle, WA, August 1999.

[8] M. Dertouzos. The Future of Computing. *Scientific American*, August 1999. Available from http://www.sciam.com/1999/0899issue/0899dertouzos.html.

[9] Federal trade commission privacy initiatives. http://www.ftc.gov/privacy.

[10] I. Getting. The Global Positioning System. *IEEE spectrum*, 30(12):36–47, December 1993.

[11] Garmin gps receiver homepage. http://www.garmin.com/products/, May 1997.

[12] A. Harter, A. Hopper, P. Steggles, A. Ward, and P. Webster. The Anatomy of a Context-Aware Application. In *Proc. ACM/IEEE MOBICOM*, Seattle, WA, August 1999.

[13] Harter, A. and Hopper, A. A New Location Technique for the Active Office. *IEEE Personal Communications*, 4(5):42–47, October 1997.

[14] Hertz Services: Hertz NeverLost. http://www.hertz.com/serv/us/prod_lost.html, 2000.

[15] HP cooltown home page. http://www.cooltown.hp.com.

[16] Jini (TM). http://java.sun.com/products/jini/, 1998.

[17] J. Lilley. Scalability in an Intentional Naming System. Master's thesis, Massachusetts Institute of Technology, May 2000.

[18] Monarch project home page. http://www.monarch.cs.cmu.edu/cmu-ns.html.

[19] NS-2 network simulator. http://www.isi.edu/nsnam/ns/.

[20] Oxygen home page. http://oxygen.lcs.mit.edu/.

[21] Pinpoint home page. http://www.pinpointco.com.

[22] N. Priyantha, A. Chakraborty, and H. Balakrishnan. The Cricket Location-Support System. In *Proc. ACM/IEEE MOBICOM*, pages 32–43, Boston, MA, August 2000.

[23] F. Tobagi and L. Kleinrock. Packet Switching in Radio Channels: Part I - CSMA modes and their throughput Delay Characteristics, Part II - The Hidden Terminal Problem in CSMA and Busy-Tone Solution. *IEEE Transactions on Communications*, 23(12):1400–1433, December 1975.

[24] Privacy international survey. `http://www.privacyinternational.org/survey/technologies.html`.

[25] J. Veizades, E. Guttman, C. Perkins, and S. Kaplan. *Service Location Protocol*, June 1997. RFC 2165 (`http://www.ietf.org/rfc/rfc2165.txt`).

[26] R. Want, A. Hopper, V. Falcao, and J. Gibbons. The Active Badge Location System. *ACM Transactions on Information Systems*, 10(1):91–102, January 1992.

[27] M. Weiser. The computer for the 21st century. *Scientific American*, September 1991.