A Privacy Conscious Bluetooth Infrastructure for Location Aware Computing

Albert Huang and Larry Rudolph
Massachusetts Institute of Technology
{albert,larry}@csail.mit.edu

July 21, 2004- TODO: remove

Abstract

We present a low cost and easily deployed infrastructure for location aware computing that is built using standard bluetooth® technologies and personal computers. Mobile devices are able to determine their location to room-level granularity with existing bluetooth technology, and to even greater resolution with the use of the recently adopted bluetooth 1.2 specification, all while maintaining complete anonymity. Various techniques for improving the speed and resolution of the system are described, along with their tradeoffs in privacy and ease of deployment.

1 Introduction

Location aware computing provides applications with knowledge of the physical location where the computation is taking place. This allows applications to operate in a more context-sensitive fashion. A mobile device could display maps and provide walking or driving directions to its user. A kiosk could, upon determining its location, configure itself to present information about local events and weather. Upon entering a room, a software agent could adjust the room’s lighting and temperature to its user’s preferences.

Fundamental to the task of location aware computing is determining the location of the computational device. In outdoor environments with unobstructed views, GPS[9] is widely used for this purpose. Indoors, and in crowded city streets, however, the effectiveness of GPS is greatly diminished. A number of approaches have been made towards indoor localization, with varying features and measures of success. We present a system that provides an infrastructure for location aware computing that emphasizes the following key features.

Cost The system must not be prohibitively expensive, and its costs should scale well with its size.

Accuracy In order to provide useful information, the system should strive for as high a resolution as possible. For pervasive computing tasks, it should at least be able to determine which room it’s in.

Privacy A location aware system should preserve the privacy of its users. Users should be able to choose whether or not to reveal their location and identity to others, and should not be actively tracked
without explicit consent. It should not be possible for the system to track users without permission.

Convenience Our system relies on technology that is already widely available and in use today. The hardware is multi-purpose and can be used for a variety of other computing tasks when not being used for localization purposes. Many potential users of our system would not need a significant investment in capital or other resources to take advantage of our infrastructure.

2 Related Work

2.1 Active Bat

In the Active Bat System\[17][10] developed at Cambridge University, small badges are equipped with radio receivers and ultrasonic emitters. Base stations emit radio control signals which cause select badges to respond with a sequence of ultrasonic pulses. These pulses are measured by a network of receivers to determine the badge’s location in three dimensions.

Active Bat achieves a high level of accuracy, enough so to reliably deduce the position of an object in a room. As the name implies, however, no privacy is guaranteed to its users, as they are actively tracked by the base stations and receivers.

2.2 Cricket

Like Active Bat, Cricket\[14] uses a combination of ultrasonic pulses and radio signals to provide short range location information. Cricket uses a passive localization approach, in that the mobile units whose locations are to be determined never need to transmit any data. Instead, statically positioned cricket beacons constantly transmit radio and ultrasonic pulses, allowing the mobile receivers to pinpoint their own location relative to the beacons.

Since mobile Cricket devices are never required to transmit, they cannot be tracked and thus offer full privacy guarantees to its users. Cricket also provides fine grained accuracy. Currently, however, Cricket devices are not widely available and prohibitively expensive for many potential users.

One attribute of both Cricket and Active Badge is that neither system works through walls or other soundproof materials. Depending on the usage, this could be either advantageous or an inconvenience. On the one hand, it is almost impossible for leakage to occur across rooms, and can make beacon placement easier in confined spaces. On the other hand, a simple pane of glass is enough to block the ultrasonic signals both systems rely on.

2.3 WLAN

Location-aware systems have also been built that use 802.11\[11] signals to localize a mobile device. One such system\[12] uses bayesian inference techniques in an area dense with wireless base stations to achieve 1 meter resolution. This approach currently suffers from a few drawbacks, the most inconvenient of which is the need for rigorous training of the device at every possible location before the system is usable. Addition or removal of base stations requires retraining of the mobile device, resulting in a system that does not scale well.
802.11 devices, while inexpensive and prevalent in laptops and desktop systems, are not as popular in smaller devices like cell phones and PDAs, largely due to the high power requirements. With typical usage drawing power in excess of 1W, few devices have batteries that can support sustained localization.

Additionally, in order to use signal strength to aid location inference in 802.11, a device must be an active part of the network. It must also, at the very least, reveal its MAC address to other devices, thus allowing other devices to track its movements.

2.4 Bluetooth

Anastasi et al[3] have also experimented with an indoor bluetooth-based location aware system. In their approach, statically positioned bluetooth devices were used to constantly scan for other bluetooth devices in the vicinity. Detected devices were then entered into a central database which was used to track the location of all moving bluetooth devices. While this approach is as cost effective as ours since it uses the exact same hardware, it allows for no privacy. If a cracker obtained access to the central database, the movements of all users could be tracked without their consent or knowledge.

The Local Positioning Profile[8](LPP) defines a standardized protocol for bluetooth devices to exchange positioning data. A device whose location is known runs a Local Positioning (LP) Server, to which other bluetooth devices can connect. LP Clients can request positioning information from LP Servers, which may be derived from preset configurations, GPS data, cellular data, or automatically generated, and then infer their own positions. The primary purpose of the LPP is to provide a means for devices to exchange data, and leaves much room for techniques to be developed for determining a device’s actual position given the position of other devices. The LPP also does not take privacy into consideration, as it is required for both client and server to have knowledge of both bluetooth device addresses, allowing a well-coordinated network of LP servers to track clients as they issue requests.

3 Our Approach

Our approach centers around the use of Bluetooth technologies to provide a robust, low cost infrastructure for location aware computing. For a review of Bluetooth technology, see Appendix A. Class II Bluetooth devices that function as beacons are placed in key locations throughout a building. These devices can be detected from a distance of approximately 10 meters. Mobile devices equipped with bluetooth radios can then scan for our beacons and determine their location to room-level accuracy.

The rest of the paper is organized as follows: Section 3.1 explains the terminology used for the rest of this paper. Section 3.2 gives a short justification for the use of Bluetooth. Our current implementation is described in section 4. A discussion of our results is given in section 5, and the paper concludes with section 6.

3.1 Terminology

beacon An active bluetooth radio participating in the locator network.

locator The device whose location is to be determined. The locator is also equipped with a bluetooth radio.
identity The identity of a device is a piece of information that can be used to positively identify it. For bluetooth devices, the 48-bit device address is sufficient to serve as this identifier. Through this paper, when we refer to the identity of a locator or beacon, we refer to its bluetooth device address.

3.2 Why Bluetooth?

Given that all we want is for a device to find its location, one might justifiably ask why use a technology as complicated as bluetooth instead of low powered radio transmitters placed in key locations around a building.

Perhaps our greatest justification is that bluetooth is already a stable, mature technology that has met great industry acceptance. According to one market research study, 69 million bluetooth ICs were shipped in 2003, approximately double that of the previous year, and are forecasted to grow to 720 million units by 2008[5]. Laptops, cell phones, and PDAs are increasingly shipped with an integrated Bluetooth radio. All of these devices would be able to participate in our location aware network without any additional hardware modifications. Our goal was to see how much we could achieve with a system that is more or less already in place.

Other candidate technologies include IrDA, IEEE 802.11, and RFID. IrDA is not suitable due to its directional nature and intolerance of optical obstructions. In an uncontrolled environment, it is simply too easy to lose a line-of-sight contact with another device. 802.11 is not subject to this limitation, but is too coarse grained for our purposes. With the restriction that a locator should not reveal its identity, an 802.11 device used in our context would not be able to use signal strength to aid its localization tasks, significantly reducing its resolution. Additionally, 802.11 draws significantly more power than bluetooth.

RFID tags are inexpensive passive devices often used in retail stores as electronic bar codes, and in identification cards to replace magnetic strips. They can be read by the more expensive RFID tag readers from varying distances. If tags were scattered throughout the premises of a building, then a mobile reader could determine its location by reading nearby tags. RFID readers are currently prohibitively expensive, however, and are also unable to provide positioning information more precise than the location of detected tags.

Bluetooth can be cheaply and easily integrated into the design of a mobile device, and USB bluetooth devices for use in personal computers are commercially available for under $20. The number of bluetooth devices needed for our network is proportional to the desired area of coverage, providing a cost effective way to increase the range and capabilities of the network.

4 Implementation

We have distributed 30 beacons throughout the research areas of our newly constructed building. Most of our beacons are USB bluetooth devices hosted by a Linux, Windows, or Macintosh computer. All of the computers were already in use by lab members in CSAIL\(^1\), so no hardware beyond the bluetooth

---

\(^1\)Ironically, the most difficult task in creating our network was convincing lab members to let us insert a bluetooth dongle into their computers. Despite being a hub of research in computer science, many lab members had no idea what
dongles was required. No extra software was needed for the beacons, other than the operating system drivers for the USB devices.

Once the beacons are in place, there are four steps that a locator must take to determine its own location.

1. The locator scans for beacons.
2. The locator determines the location of detected beacons.
3. The locator determines its location relative to the detected beacons.
4. Using information from the previous two steps, the locator determines its absolute location.

### 4.1 Inquiring for beacons

To detect nearby beacons, the locator performs a bluetooth device inquiry. This inquiry is nothing more than repeatedly broadcasting a predefined sequence of bits while hopping channels pseudorandomly. Thus, the locator does not reveal its own identity during an inquiry. Upon hearing a response, the locator must determine if the responder is a beacon in the locator network, or if it’s merely another bluetooth device. This is discussed in the next section.

The ideal beacon would always listen for the inquiry sequence and respond almost immediately upon detection. Unfortunately, the bluetooth specification is much more complicated than this, and a number of factors can cause a beacon to either not respond, or to not detect an inquiry.

- Electromagnetic noise and interference with other devices in the 2.4 GHz range may hinder communications.
- A beacon can not listen for an inquiry all the time. It must allocate time to listen for connection requests, and to participate in active connections.
- Upon first detecting a device inquiry, a beacon will always enter a backoff stage, in which it idles for 0 to 0.33 seconds randomly. Only when the beacon wakes up from the backoff does it respond to an inquiry.
- A beacon, while listening for inquiries, will listen on one of 32 predefined channels at a time. During an inquiry, the locator will inquire on half of these channels for 2.56 seconds, switch to the other half for another 2.56 seconds, and then alternate two more times. Consequently, it is possible that a locator will not even inquire on the same channel that a beacon is listening on for at least 2.56 seconds. Figure 1 illustrates this.

While nothing can be done about noise and interference from other radio sources, we discuss techniques for improving the detection speed of a beacon in section 5.2. One researcher whose computer was not hosting a beacon expressed sincere concern that nearby bluetooth devices would make his workstation vulnerable to crackers. This is a problem addressed in version 1.2 of the Bluetooth specification, which allows for adaptive frequency hopping to avoid channels being used by co-located, interfering devices. However, existing bluetooth devices are not able to take advantage of this ability.
4.2 Determining a beacon’s location

The locator maintains a lookup table, mapping beacon identities (Bluetooth device addresses) to locations. Upon detecting a beacon during an inquiry, the locator checks the lookup table for a mapping. If a mapping exists, then the locator can instantly determine the location of the beacon. If no mapping exists, then the locator can query the beacon for its location at the expense of anonymity.

The Bluetooth specification allows for exactly one situation when information can be sent from a device B (the beacon) to a device A (the locator) without B having knowledge of A’s identity. This is when B is responding to an inquiry made by A, and no further communication between the two devices has taken place. By leveraging this feature of Bluetooth, once a beacon has responded to an inquiry with its device address, we are able to determine its location with the help of the lookup table.

If a beacon B is detected that does not have an entry in the locator’s lookup table, and the locator is willing to reveal its identity, then it can establish a higher level Bluetooth connection with B and request more information, such as the location of B. We chose to embed a beacon’s location inside its Bluetooth friendly name, so that a locator need only issue a remote name request to determine the beacon’s location. For example, B could be given the name “OKN-32-305” to signify that it is in building #32, room 305. “OKN” is used here as a prefix to distinguish our beacons from other Bluetooth devices.3

By issuing the name request, the locator reveals its identity to the beacon, and allows itself to be tracked. If the locator is to remain anonymous, then it must have already populated its lookup table with the relevant entries. This table could be obtained by a centralized distribution service such as a website or a database. It is also worth noting, however, that the extent to which a system can track a locator based on the remote name requests is directly related to the frequency that the locator issues

---

3This is exactly what LPP is designed to do - provide a standardized method for the transfer of positioning information from the beacon to the locator. However, at the time of writing, LPP was still in draft form and we found this method much simpler.
them. If the locator does not issue name requests very often, then it is still difficult to track.

4.3 Determining locator position relative to beacons

Once a locator has detected one or more beacons, it can take the intersection of the areas covered by detected beacons to determine its approximate location. Thus, the precision with which a locator can determine its position is directly related to the number of beacons it detects. Figure 2 illustrates this principle.

Figure 2: a) When the locator L can only detect one beacon, it can only conclude that it is somewhere within the circle b) When two beacons are detected, much greater resolution is achievable, and the locator can conclude it is somewhere in the shaded region **this is a really lame figure**

In general, we can hold the height of the locator and beacons fixed, and compute the locator’s location using just the X and Y coordinates of the beacons. A complication arises, however, when beacons on multiple floors of a building are detected. In this case, knowledge about beacon altitude is needed in order to properly determine the locator’s position.

Bluetooth devices adhering to the Bluetooth 1.2 specification [7] or later support device inquiries that report signal strength of discovered devices. Most bluetooth devices do not yet support this feature, however, so we mention it only for future consideration. Currently, signal strength of another bluetooth device can only be determined after a higher level connection has been established and identities revealed. Section 5.1 discusses this in further detail.

5 Discussion

In this section, we discuss techniques for improving accuracy, the speed of localization, and tradeoffs between privacy and performance.

5.1 Signal strength and Link Quality

The `HCI_Read_Link_Quality` command is supported on some bluetooth devices, and can be used to determine the quality of a bluetooth connection with another device. In the absence of radio noise and obstruction from objects such as wood, metal, and people, the quality of a link between two bluetooth devices is inversely related to their distance.

While we are not able to use link quality to determine exact distance from a beacon, we can use it to establish an upper bound. Factors other than distance can only serve to diminish link quality.
For example, if we know link quality at a 2 meter distance from a beacon has value 200 when there are no other radio devices in the area and nothing between the locator and beacon, and the current link quality measures 200, then we can conclude that the locator must be at most 2 meters from the beacon.

The use of link quality can also be used to resolve which floor of a building the locator is on, with the assumption that signals received from beacons on other floors are much weaker than signals received from beacons on the same floor.

As mentioned above, information about link quality with another device is not supported on all bluetooth devices, and the method for calculating link quality is device specific. Thus, in order to make effective use of link quality information, detailed knowledge about the specific bluetooth device being used must be available. In our limited experience, signal strength and link quality is not available on HP iPaqs, is available on D-Link USB bluetooth devices, and is available on Nokia Series 60 cell phones with the signature of a non-disclosure agreement.

Some devices also support the HCI\texttt{Read\_RSSI} command, which provides direct information about the signal strength of a connected device. In our testing with D-Link DBT-120 USB bluetooth devices, we found link quality to be more closely related to distance than RSSI, so we did not experiment with this command significantly. Additionally, in the Master's Thesis by Nilsson and Hallberg\cite{13}, the authors found that signal strength was poorly correlated with distance. Although the focus of this paper is not algorithms utilizing link quality and RSSI, we believe they may be useful in more precisely determining the locator position, and merit further investigation.

5.2 Multiple bluetooth devices

The bluetooth specification is optimized for the situation where many bluetooth devices are all in the same vicinity. Device inquiry is especially slow because of the pessimistic backoff algorithms used to minimize collisions. The recommended duration for a device inquiry is 10.24 seconds\cite{6}, which is clearly longer than many applications can tolerate. In an environment sparsely populated with bluetooth devices, it is possible to group multiple devices together to form a single beacon with a much faster response time. Similarly, a locator can make use of multiple bluetooth radios to issue queries in parallel.

We equipped a number of PCs with two bluetooth devices each. Both beacons were given the same friendly name to indicate they were in the exact same location, and both beacon identities were entered into a locator’s lookup table. We conducted experiments in different office environments to see how much faster a locator could determine its location when two beacons are used instead of one. The key question here is whether or not the probability of one beacon responding is conditionally dependent on the probability of the other beacon responding. If so, then using multiple beacons is not likely to be helpful. If not, however, then it should help greatly in areas where the chance of receiving a response from a single beacon is low.

In our experiments, we found that the beacons responded independently of each other, providing an ideal increase in response rate. The results of one of these experiments is shown in figure 3. In this run, the locator was placed approximately 8 meters from two colocated beacons, with a closed
Figure 3: When beacons A and B are in the exact same location, and the locator knows that location, the locator can determine its location much more quickly than when in the presence of only one beacon. This is especially beneficial in a noisy environment such as the one this test was conducted in.

doors, some wooden office furniture, and a metal filing cabinet in between the locator and the beacons. Additionally, a host of other active bluetooth and WiFi devices were operating in the vicinity.

Similarly, we found that when a locator was equipped with two bluetooth devices, and performed inquiries with both devices simultaneously, beacons were also discovered more quickly. These results are summarized in figure 4. In order to achieve the improved response rate, however, we found we needed to disable discoverability of the locator’s bluetooth devices. Otherwise, performance actually decreased as the locator began responding to its own inquiries. Response rate of a single bluetooth device in range was still not as fast as when a beacon is equipped with two bluetooth devices and the locator with one. We attribute this to the backoff algorithm used during the inquiry scan process.

While we have no plans to deploy locators with multiple bluetooth devices, devices that need to be able to discover individual bluetooth devices quickly could benefit from perform two simultaneous inquiries. Additionally, the names of devices could be determined twice as rapidly, since the remote name requests could be issued in parallel.

5.3 Early Timeout

If the locator has an incomplete lookup table mapping beacon identities to beacon locations, then it must issue name requests to discovered bluetooth devices in order to determine their locations\(^4\). This entails joining and leaving the piconet of every discovered device, a potentially time consuming process. If a full length inquiry is performed, followed by issuing a name request to each beacon one by one, the cumulative time spent becomes prohibitively expensive.

The bluetooth 1.2 specification introduces a new command, which allows a device to cancel an in-progress remote name request. In a situation where the locator discovers many unrecognized bluetooth devices and needs to query each device for its name, the locator should not spend too much time querying a single device.

\(^4\)If the user desires absolute anonymity, then this step shouldn’t be taken, as it reveals the locator’s identity.
Looking at figure 1, we can see that the majority of Bluetooth devices discovered during an inquiry are discovered early on. In our measurements, 53% of Bluetooth devices discovered during an inquiry were discovered in the first 2.56 seconds. Similarly, figure 5 shows our findings that during a remote name request, if the name of a device was resolved at all, it was resolved during the first 5.12 seconds of the request 87% of the time. A remote name request did not always succeed, and could take up to 20 seconds to fail with a timeout. This could easily happen if the devices are moved out of radio range before the name request completes.

In our experiments, the mean time for a remote name request to complete (either succeed with a name, or fail on a timeout) was 3.4 seconds. The mean time for a device to be discovered during an inquiry was 2.5 seconds. Although these numbers will fluctuate with the environment, devices used, and exact operating conditions, we believe them to be fairly representative of a typical office environment.

Based on these observations, we suggest the following algorithm for localizing the locator as quickly as possible.

The locator performs a device inquiry for 2.56 seconds, during which time any unrecognized devices are queued for name resolution. If there are unrecognized devices after 2.56 seconds, then the locator stops its inquiry. For each unrecognized device, the locator tries to resolve its name by issuing a remote name request for 5.12 seconds. If the name of the device is not determined within 5.12 seconds, then the locator cancels the remote name request and moves on to the next unrecognized device. When all the unrecognized devices have been queried for their names, the locator repeats.

By imposing a 5.12 second timeout on remote name requests, the algorithm prevents the locator from wasting much time on an unresponsive Bluetooth device. The average time for a remote name request decreases to 2.2 seconds.

I think this would work better with experimental evidence instead of this nasty mathematical justification...

Consider a room with \( n \) unknown Bluetooth devices, of which exactly one is a beacon \( b \). Let \( p_r \) denote the probability of successfully resolving the name of a device before the locator times out,
Figure 5: The name request process is similar in nature to the inquiry process, but uses a different set of 32 channels. If the name of a device is resolved, it is usually done so during the first 5.12 seconds of the name request - the amount of time it takes for the locator to iterate through both trains A and B.

$p_s = 0.87 p_r$ be the probability of successfully resolving the name of a device in the first 5.12 seconds of a name request, $p_d$ be the probability of discovering a device during a full 10.24 second inquiry, and $p_e = 0.53 p_d$ be the probability of discovering a device during the first 2.56 seconds of an inquiry. Our goal is to minimize the expected time to localization (i.e. the time to discover $b$ and successfully resolve its name).

Using the simple method of performing a full 10.24 second inquiry, followed by issuing a name request, we can estimate the expected time to localization as follows. 10.24 seconds is spent performing a device inquiry, after which $p_d n$ devices will have been discovered. Following the inquiry, the locator issues remote name requests to $\lfloor p_d n \rfloor$ devices. The name of each device will be successfully resolved with probability $p_r$, with each name request taking an average of 3.4 seconds. On the first pass of this algorithm, the probability of discovering $b$ and successfully resolving its name is $p_d p_r$, and is expected to take $t = 10.24 + 3.4 \frac{p_d n + 1}{2}$ seconds. If $b$ is not detected or its name is not resolved, then the locator tries again. On the next try, however, it will already have successfully resolved the names of $p_r p_d n$ devices, and will not need to query them again, so the number of devices to issue name requests to is $[(1 - p_r) p_d n]$. Generalizing, we find that the expected time needed for the locator to discover $b$ and resolve its name is:

$$t(n) = p_d p_r \left(10.24 + 3.4 \frac{p_d n + 1}{2}\right) + p_d p_r (1 - p_d p_r) \left(10.24 + 3.4 \frac{p_d n + 1}{2} + \sum_{j=0}^{\infty} 10.24 + 3.4 (1 - p_r)^j p_d n \right) \quad (1)$$

$$t(n) = p_d p_r \sum_{i=0}^{\infty} (1 - p_d p_r)^i \left(10.24 + 3.4 \frac{(1 - p_r)^i p_d n + 1}{2} \right) + \sum_{j=0}^{i} 10.24 + 3.4 (1 - p_r)^j p_d n \quad (2)$$

why do I feel like it shouldn’t be this messy?

Using the early cancellation method, we can use the same equations, but with different values. In our measurements, the expected duration of an inquiry drops to 2.6 seconds, and the expected time to complete a remote name request drops to 2.2 seconds.
expected localization times in presence of one beacon

Figure 6: By cancelling inquiries and remote name requests prematurely, the expected amount of time needed for the locator to determine its location decreases significantly.

\[ u(n) = p_n p_s \sum_{i=0}^{\infty} (1 - p_e p_s)^i (2.6 + 2.2 \frac{(1 - p_s)^i p_e n + 1}{2}) + \sum_{j=0}^{i} 2.6 + 2.2 (1 - p_s)^j p_e n \]  

In equations 2 and 3, components of the summation where \( i > 3 \) can be safely ignored, as they are insignificantly small. A comparison of the expected localization times is shown in figure 6.

The technique described here can also be used with LPP[8]. The process of connecting to an LPP server is similar in nature to a remote name request, and cancellation of a remote name request can be replaced by cancellation of a create connection command.

Even though we are able to use remote name requests to achieve reasonable performance in positioning a locator, we believe it should only be used as a last resort, when the locator has no other means of determining its position. Since the amount of time needed is proportional to the number of bluetooth devices in the area, in a crowded lecture hall with many bluetooth cell phones and PDAs, this becomes an unacceptable method. Usage of a cached lookup table is still preferred, as it is faster in all cases, and is relatively independent of the presence of other bluetooth devices.

5.4 Privacy

When a bluetooth device is left in discoverable mode, a host of possibilities for abuse arises. A widely distributed network could easily track a user’s movements[3]. Unrequested advertisements[1], solicitations for sexual encounters[16], and other unwanted messages[4] could arbitrarily be sent to the user’s cell phone or mobile device. While there certainly is an audience that would welcome these actions, there are also many more who would not. One of our major goals is to respect the privacy of our participants.

Our system currently allows a user to retain this privacy, preventing other devices from discovering the locator, while still being able to take advantage of our location aware services. While it is possible for a well-coordinated system to track bluetooth device inquiries, the system has no way of knowing what device made the inquiries. Since bluetooth also has numerous other uses, almost all of which involve
making device inquiries, it is infeasible to track an individual user that does not wish be tracked. Even if the user’s locator performs the occasional name request to identify unrecognized bluetooth devices, a surveillance network would still have a difficult time pinpointing a user’s location.

If the user desires complete anonymity, coarse grained location-aware computing is possible at room-level granularity with existing bluetooth 1.1 hardware. Newer devices compliant with the 1.2 specification can achieve even better resolution while still remaining anonymous. If privacy is not a concern, however, then more precise localization is possible using signal strength of nearby beacons.

5.5 Future Research

At the time of this writing, there is only one USB bluetooth 1.2 device commercially available on the market. Once more devices are available, we believe the additional features made available in the 1.2 specification are worth investigating. Since the method of measuring signal strength is vendor specific, some devices may be able to obtain better precision than others.

Application of noisy inference techniques to the localization problem in our system is also an area that merits further investigation. There are a number of users for whom privacy concerns are not an issue that would benefit from the use of these methods.

We have plans to increase the coverage and density of our network, and deploy more precise client software. Integration with other location aware applications is also planned.

6 Conclusion

We have presented a privacy conscious location aware system that is based solely on inexpensive, off the shelf components. Participants in our system would not require specialized devices, and could simply use the bluetooth enabled cell phones and PDAs they have already been purchasing. The client software needed to take advantage of our system is lightweight and easily deployed. As the infrastructure changes, clients can either obtain centralized updates, or update their cache manually by querying beacon locations in person.

With the use of bluetooth 1.1 devices, our system provides room-level granularity while retaining complete anonymity. Bluetooth 1.2 devices that leverage noisy inference techniques such as kalman filters and bayesian inference are able to obtain even greater resolution while still remaining anonymous. If a user does not require anonymity, then a 1.1 device is sufficient for accurate, meter-level localization.

Appendix A

The bluetooth specification defines a standard for short-range wireless communication in the 2.4 GHz frequency band. Three classes of bluetooth devices are defined, with range and power consumption as listed here.

<table>
<thead>
<tr>
<th>class</th>
<th>range (m)</th>
<th>max power consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>100</td>
<td>100 mW</td>
</tr>
<tr>
<td>II</td>
<td>10</td>
<td>2.5 mW</td>
</tr>
<tr>
<td>III</td>
<td>1</td>
<td>1 mW</td>
</tr>
</tbody>
</table>
Most bluetooth devices embedded in consumer electronics such as cellular phones and PDAs are class II devices.

79 channels\(^5\), or frequency bands, are defined for bluetooth. Communicating devices will hop pseudorandomly across these 79 channels when transmitting and receiving data. Of interest to us is how bluetooth devices discover each other and determine their friendly names.

**Device Discovery**

When a device \(d\) is discoverable by other bluetooth devices, it periodically enters *inquiry scan mode*, in which it hops pseudorandomly across 32 channels, listening for 1.28 seconds on each channel. During the inquiry process, a discovering device \(i\) will split the 32 channels into two trains A and B, each with 16 channels. \(i\) will then inquire on all of the A channels for 1.28 seconds, and then switch to inquiring on all of the B channels. It will switch trains two more times before the inquiry completes after 10.24 seconds.

When \(d\) first detects an inquiry, it will not respond immediately. Instead, it performs a random backoff for 0 - 0.32 seconds. After the backoff, \(d\) will respond to the next inquiry it detects.

**Piconets**

Once a device has discovered another device, they must join the same piconet in order to establish higher level communications. When two devices are on the same piconet, they follow the same channel hopping sequence when receiving and transmitting data. \(d\) will periodically enter *page scan mode*, which is similar to *inquiry scan mode* except that \(d\) listens for pages instead of inquiries. The paging process for \(i\) is similar to the inquiry process - it pages \(d\) on 32 channels, using two trains, one train at a time.

**Remote Name Request**

A bluetooth device is almost always assigned a friendly name by its user, which is used by people to identify the device. In order for \(i\) to determine the friendly name of \(d\), they must be on the same piconet. Thus, \(i\) will typically form a temporary piconet with \(d\) and then disconnect. This is a potentially time consuming process.

**References**


---

\(^5\)In some countries, such as France, only 23 channels are defined. For convenience, we only consider the specification for 79 channels.


