Honeycomb: Indoor location estimation based on Wi-Fi signal strength

APPROVED BY

SUPERVISING COMMITTEE:

Christine Julien, Supervisor

William Bard

Honeycomb: Indoor location estimation based on Wi-Fi signal strength

by

Michael Linder, B.A.

REPORT

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

Master of Science in Engineering

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2015

Dedicated to my wife, Dana, whose support made this possible.

Honeycomb: Indoor location estimation based on Wi-Fi signal strength

Michael Linder, M.S.E.
The University of Texas at Austin, 2015

Supervisor: Christine Julien

This paper presents Honeycomb, an indoor location estimation product based on Wi-Fi signal strength. Wireless Local Area Networks are ubiquitous today, and most people carry Wi-Fi capable devices in their pocket. This existing infrastructure can thus be leveraged for purposes of location estimation. Using Wi-Fi signal strength fingerprinting, Honeycomb harnesses existing Wi-Fi infrastructures as a means to track the movements of individuals through an indoor space. Fingerprinting is a method by which Wi-Fi signal strengths are mapped at regular intervals in a bounded space. Once a space is fingerprinted, a given node must simply sample Wi-Fi signal strengths as it moves through the same space and Honeycomb's algorithm will determine the node's path in an offline manner. Because Honeycomb only requires nodes to passively measure Wi-Fi signal strengths rather than send out its own beacon, it prevents malicious third parties from gaining access to any real time data, and thus maintains the security and privacy of the user. By performing location

estimations on the data collected on an independent platform, and not on the device itself, it saves the user from spending the computing power, and thus the device's battery. We believe Honeycomb to be a product unlike any other, which is suitable for deployment in multiple real world scenarios.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

In recent years wireless LAN technology has become ubiquitous. Wi-Fi access points have become virtually trivial to install, and nearly everyone carries a Wi-Fi capable mobile device in their pocket. It is also the case that much research has been done on various methods of location estimation. It follows, then, that location estimation that leverages Wi-Fi would be a valuable topic, and in fact much research has already been done in the space, including [9], [11], [10], [7], and [12].

The benefits of using Wi-Fi for location estimation are manifold. For instance, while the Global Positioning System is in many ways the premier method for location estimation in the world [3], GPS signals are often unreliable indoors [19], making it a poor choice for any indoor location estimation. Location systems that use other mechanisms such as RFID [17], ultrasound [13], or geomagnetism [5] are difficult to setup, require specialized hardware, and ultimately can only be used for a single purpose. Wi-Fi based location estimation solves all of these problems. Wi-Fi signals are readily available indoors. Wi-Fi is relatively cheap and easy to setup, and in many cases existing access points can be leveraged.

1.1 Definitions

There are a few terms that will be used throughout this paper that it is important to define early. Understanding these definitions will help make clear the purpose of this paper and its contributions.

Signal Strength vs. RSSI Much of the research involving location estimation with Wi-Fi signal strength refers to the measured power present in the radio signal as the Received Signal Strength Indicator (RSSI). While in general terms this moniker is good enough, in truth, the IEEE 802.11 specifications [8] do not indicate a specific relationship between RSSI and the actual power level as measured in either milliwatts (mW) or Decibel-milliwatts (dBm). As such, manufactures are free to provide their own arbitrary units, and RSSI measurements are generally found to be integer values greater than 0. Because of this inconsistency, Honeycomb does not use RSSI, favoring instead what we refer to as simply "signal strength". For our purposes, signal strength is a measure of the power present in the Wi-Fi signal as measured in dBm. dBm is a measurement relative to 1 mW of power, where 0 dBm is equal to 1 mW. Because dBm measurements are made on a logarithmic scale, we find our measurements to be negative integers between 0 and 100, where the measured value is the exponent in the logarithm. So, where 0 dBm is equal to 1 mW, -10 dBm is equal to .1 mW, -20 dBm is equal to .01 mW, and so on. Measuring signal strength in this way allows Honeycomb to maintain consistent tracking across signal strength measurement platforms, and thus makes Honeycomb a more diverse and viable product.

Fingerprint Throughout this paper will we refer to fingerprints. In this context, a fingerprint is a set of Wi-Fi signal strength measurements taken from a set of Wi-Fi access points at a specific point in a given space.

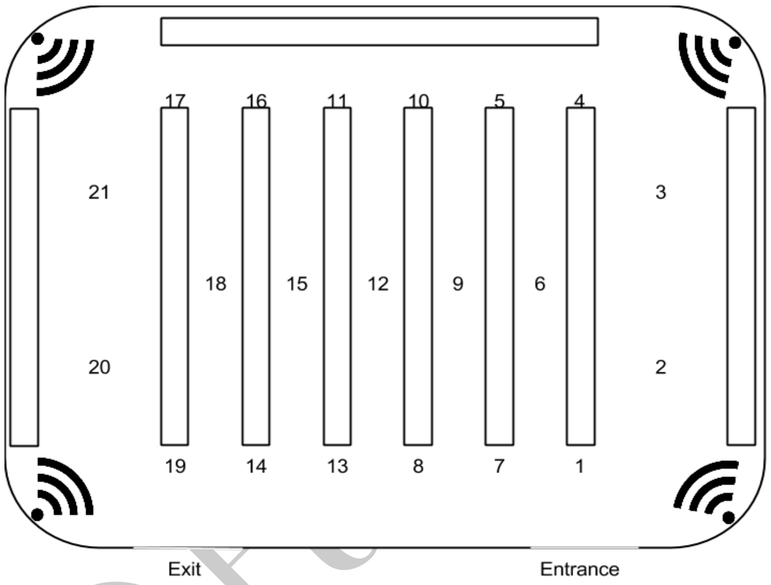

Figure 1.1: An example of a location with labeled measurement points

Fingerprint Session While the concept of the Wi-Fi fingerprint is relatively common, here we introduce a new concept that will enable a Honeycomb installation to maintain its value over time: the fingerprint session. A fingerprint session is the complete set of fingerprints from all measurement points in a space at measurement time. Because the internal layout of any given space can change over time, causing issues with blocking and reflection of Wi-Fi signals, Honeycomb has built in the ability to re-fingerprint the entire location at any time so that the measurements can be as precise as possible. Figure 1.1 shows an example location with aisles like that of a grocery store. It includes four Wi-Fi access points and numbered labels for points that may be considered relevant for location estimation. A single fingerprint session in this space would be the complete set of Wi-Fi signal strength measurements from all four access points at every numbered location.

User Track A user track differs from a fingerprint in that it represents a single user's movement through the space. Thus, a user track is composed of a set of timestamps, each of which is associated with a set of signal strength measurements for each of the access points in the space. Figure 1.2 shows an example of a user track. In this example, the small dots represent the user's actual path through the space, while each large dot represents a timestamped set of signal strength measurements. Thus it can be seen that in this example that the user walked at a relatively constant pace through the space, slowing down four times near locations 5, 7, 14, and 17. For location estimation purposes, Honeycomb will compare a given user track to the most recent fingerprint session for the given space.

1.2 Motivation

Wi-Fi based location estimation is a well researched topic [11]. The goal of Honeycomb is to leverage that research and build an indoor location

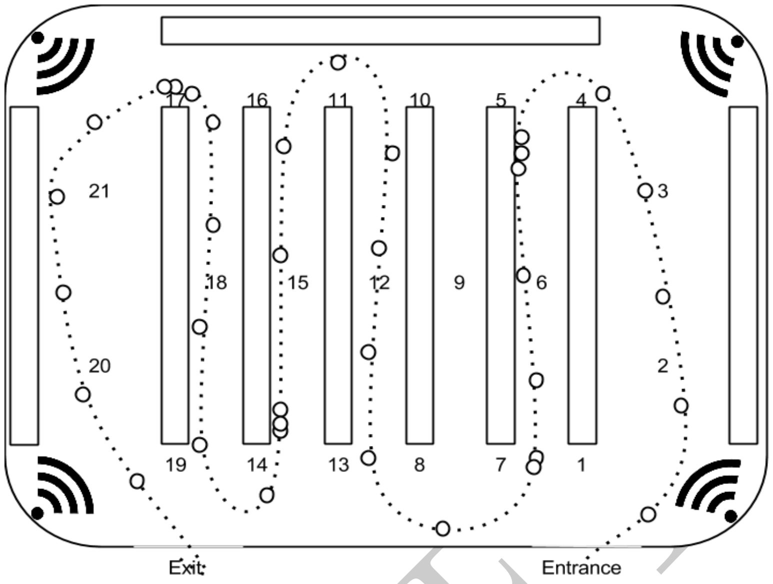

Figure 1.2: An example user track

tracking system which is suitable for deployment in a real world scenario. As such, Honeycomb includes an Android application capable of fingerprinting a space, and an API which is deployed to the web for uploading both fingerprints and user track data. The web application also executes the location estimation algorithm, and provides a user interface for browsing the user track data. Honeycomb itself remains agnostic of the mechanism used to gather the user's Wi-Fi signal strength data. By decoupling Honeycomb in this way, we allow Honeycomb to be used in multiple scenarios in which a specialized user track gathering mechanism is desired.

User privacy is also a major motivating factor in our work. By only performing location estimation based on signal strength and timestamp data passively gathered on the Wi-Fi capable device, we have avoided the pitfalls of systems that require the mobile device to send out a beacon [9] [19] which can be intercepted by malicious third parties. Additionally, the offline nature of the location estimation algorithm greatly simplifies the entire process, as real time location estimation is still a highly volatile field [18]. It also helps provide an additional layer of privacy protection for the user, as the data can easily be decoupled from any identifying information before processing.

1.3 Contribution

While there has been much research done in the space, to the author's knowledge, there does not yet exist a product on the market that achieves true location estimation via Wi-Fi signal strength measurements. Honeycomb is such a product. Honeycomb provides tools on the web for site administrators to manage their locations and view individual user tracks. It also includes an API through which fingerprints and user tracks can be uploaded, and an Android application capable of doing the fingerprinting and uploading the results to Honeycomb through the API.

1.4 User Stories

We envision Honeycomb being deployed in multiple different scenarios. Essentially, wherever there is a desire to track a person's movements through a bounded space, we believe Honeycomb to be part of a viable solution. In this section, we describe two such scenarios.

1.4.1 The Grocery Store

The canonical example, and the one to which we will refer throughout this paper, is the retail establishment that wishes to track customer movements through their space. In this case, we use the example of the grocery store. The grocery store lends itself well to this scenario due to the fact that stores are generally rather large in size and that there is a general expectation that its customers will spend most of their time moving around the space. In this scenario, we see two major benefits of customer location tracking. Although we've chosen the grocery store for this scenario, these same benefits could be applied to similar scenarios, such as large conferences with multiple rooms and displays. In this scenario, we see two major benefits to the grocery store:

Visibility High visibility of products is a valuable commodity in any retain environment. Each store can use aggregate data about its customers movement through the space to identify key, high traffic areas, and sell shelf space or ad space accordingly. Additionally, [16] shows that customers respond to engaging store layouts, which can be facilitated by customer movement data. Similarly, a conference can identify high traffic areas and place sponsor ads, or other information valuable to attendees, at the site.

Flow Control Data about how people move through a space can be used to identify bottlenecks or other poorly designed traffic areas and improve them in order to provide a better user experience for patrons. In the context of a grocery store, this could result in a generally happier clientele, which means more repeat business [16]. At a conference, this data could be used to identify popular booths, and rearrange them in such a way that will cause traffic to flow in desired patterns, either to eliminate bottlenecks or to direct traffic flow past more sponsors.

1.4.2 Security Guards

For security companies, a critical component of their service is often regular patrolling of the space by a human being. For this reason, it is crucial for the security company to make absolutely sure that the security guard actually goes on their patrols. This is often accomplished via RFID stations or QR codes located throughout the space that the guard must scan in order to prove that they were there. However, this scanning requires the security guard to be both mentally and visually distracted for the length of time required to make the scan, and therefore creates a weak point in their security that can be exploited. Passive tracking of the security guard via Wi-Fi signal strength polling eliminates this distraction, while still maintaining the necessary tracking.

Note that in the above examples, the method by which the polling data is transferred from the individual's Wi-Fi capable device into Honeycomb may be dramatically different. In the case of the grocery store, there may be some desire on the part of store management to evaluate the data before transferring it to Honeycomb, for example to credit the customer's account for their incorporated rewards system, which may be necessary as a motivation for the user to allow themselves to be tracked. A grocery store's general patterns of ingress and egress provide a natural place for the data to be collected, possibly over Wi-Fi itself, so as to be as unobtrusive to the customer as possible. Conversely, in the example of the security guard, there may not be a convenient area in which to place a data collector, since you may be tracking multiple security guards through multiple spaces, and it is not worth setting up a data collector for one individual in a given space. Additionally, obtrusiveness is not an issue, since reporting their position data is part of the security guard's job. It is for this reason that Honeycomb remains agnostic of the user data gathering mechanism, in order to provide benefit in a wider variety of areas.

1.5 Structure Of This Report

The goal of this report is to provide context for the value of a Wi-Fi signal strength based indoor location tracking system and to describe the particular implementation of Honeycomb. In Chapter 2 we discuss the state of Wi-Fi based location tracking and explain why we feel that the methods we chose were the best choices for Honeycomb. In Chapter 3 we present BumbleBee. Because Honeycomb remains agnostic of user track gathering mechanisms, we needed to choose a product that is capable of gathering user track data. BumbleBee is an independent, previously unpublished Wi-Fi signal strength measurement tool used to collect user tracks, and was co-written by the author of this paper. In Chapter 4 we discuss the architecture of

Honeycomb and the technologies on which it was built. In Chapter 5 we present the testing procedures that were implemented and their results. In Chapter 6 we discuss the results of our tests and the future of Honeycomb as a product.

Chapter 2

Background and Related Work

Several key decisions were made in designing Honeycomb. Chief among these were the basing of our location estimation system on Wi-Fi signal strength, the fingerprinting of the space, and the subsequent offline location estimation algorithm. Our choice to base our system on Wi-Fi signal strength was an easy one. Systems based on RFID [17], ultrasound [13], or geomagnetism [5] require single purpose hardware, and can be costly to install, and were thus rejected outright. In this section, we review the state of Wi-Fi location estimation and explain why we made the choices that we made.

2.1 High Level Location Estimation Schemes

Liu [11] categorizes three high level location estimation schemes: triangulation, proximity, and scene analysis.

2.1.1 Triangulation

In triangulation schemes, like [20] and [19], nodes are tracked based on the time of arrival (TOA), angle of arrival (AOA) or roundtrip time of flight (RTOF) of Wi-Fi signals. These methods, while potentially extremely precise, require knowledge of the locations of access points themselves, as well as the distances between them. We consider the near plug-and-play ability of Honeycomb to be a benefit to its adopters, and thus consider this requirement to be a significant blocker to adoption. Additionally, in order to gather precise TOA, AOA, and RTOF measurements, a line of sight must be maintained between the access point and the mobile node, which is not possible in the scenarios in which we expect Honeycomb to be deployed. Thus triangulation schemes were rejected immediately.

2.1.2 Proximity

Proximity schemes, like [4], generally consist of a dense array of antennas which are capable of detecting mobile nodes, and the location of the mobile node is considered to be whichever antenna detects it. These schemes thus require significant extra infrastructure, which we believe would be a barrier to entry for Honeycomb's expected customers. Additionally, these schemes require the mobile node to send out a beacon for the antenna to detect, which we consider to put the mobile node carrier's security and privacy at risk.

2.1.3 Scene Analysis

Thus we are left with scene analysis schemes. Scene analysis schemes generally consist of two phases: a training phase and an estimation phase. In the training phase the location is mapped, usually via fingerprinting, and in the estimation phase a mobile node gathers its own measurements which are then compared to the fingerprints. There are many methods for doing this comparison, which we will discuss in the next section. While scene analysis schemes are not without their own overhead, they are far preferable to both triangulation schemes and proximity schemes for Honeycomb's intended uses. For these reasons, we chose a scene analysis scheme, which we will describe further here.

2.2 Scene Analysis Scheme Approaches

While all scene analysis schemes involve a training phase and an estimation phase, the particulars of these phases can vary. Most approaches, including all of those cited in this section, employ fingerprinting in the training phase, but vary dramatically in the estimation phase. There are three basic approaches to the estimation phase in this scheme, as described by [18]: probabilistic matching, Bayesian networks, and nearest neighbor.

2.2.1 Probabilistic Matching

Probabilistic approaches take a user track measurement and calculates the probability that the measurement was taken at each of the fingerprinted points in the space. These approaches requires complex calculations for determining probability, but do not generally perform better at location estimation than other approaches[7].

2.2.2 Bayesian Networks

Bayesian network approaches [9] [15] are a special subset of probabilistic approaches. They attempt to build a probability model in the training phase, and estimate the evolving state of the system based on that model and the previous state estimate. These approaches can be among the most accurate, but require a trade off for computational overhead. While these approaches are quite promising, we fear that the computational overhead will present problems when run at a large scale, and therefore have chosen not to employ them at this time.

2.2.3 Nearest Neighbors

Nearest neighbor methods [14] [12] are the simplest approaches, and provide an acceptable level of accuracy for Honeycomb. In nearest neighbor approaches, the estimation phase implements a heuristic, often Euclidean distance (Figure 2.1), is applied to the gathered signal strength measurements, and the fingerprint which most closely matches the gathered measurement is considered to be the location of the mobile node. While nearest neighbor approaches are subject to the Wi-Fi signal multipath problem, the fact that each estimate does not rely on the state of the estimate before it means quick recovery in cases of error. Additionally, accuracy of estimations can be greatly influenced by density of fingerprints.

2.3 Offline Location Estimation

Some location estimation systems perform real time user tracking [2] [1] [9]. These real time tracking algorithms generally fall into two categories: those in which the location estimation is done on the mobile device itself, and those in which it is not. Systems which perform the location estimation somewhere other than the mobile device require direct communication with the device in order to gather real time data. As stated previously, these approaches were rejected due to privacy and security concerns. While approaches in which the estimation is done on the mobile device avoid these concerns, they violate Honeycomb's goal of staying decoupled from the user track data gathering mechanism. Additionally, by performing calculations on a central server, battery and computational power of mobile devices is conserved.

2.4 The Honeycomb Approach

Following the decisions made in this chapter, the Honeycomb approach to indoor location estimation is a scene analysis scheme with a nearest neighbor approach. For the training phase, Honeycomb includes an Android mobile application capable of fingerprinting a space for signal strength measurements from a given set of BSSIDs. It then uploads those measurements as raw data to a web server. The number and density of fingerprints taken can be determined by the site administrator based on desired accuracy of location estimates. For the estimation phase, Honeycomb decouples itself from the actual user track gathering tool, but exposes an API for timestamped user tracks to be uploaded.

It then runs a Euclidean distance algorithm (Figure 2.1) to determine which fingerprint the user track was nearest to for each timestamp, and thus achieves continuous location estimation for the duration of the user track.

$$\sqrt{(FP_1 - UT_1)^2 + (FP_2 - UT_2)^2 + ... + (FP_n - UT_n)^2}$$

Figure 2.1: The Euclidean distance algorithm between a fingerprint (FP) and a user track (UT) with N signal strength measurements.

Chapter 3

BumbleBee

Because one of Honeycomb's goals is to decouple the user track gathering mechanism from the location estimation mechanisms, we needed a tool with which to gather user tracks in order to prove Honeycomb's effectiveness. Fortunately, in a previous, unpublished project, the author of this paper co-wrote precisely that tool, called BumbleBee. BumbleBee's only purpose is to provide a viable user track gathering tool in order to feed user track data into a location estimation system. Here, we describe BumbleBee in more detail.

3.1 Infrastructure

Bumblebees major contribution is its novel infrastructure (Figure 3.1), in which there exists a Wi-Fi network host, the Gatekeeper, at the store entrance. As mobile devices, the bumblebees, enter the store, they connect to the Gatekeeper and request work. The Gatekeeper provides the bumblebee the BSSIDs addresses of N Wi-Fi access points located throughout the store, where N is the number of different access points from which the Gatekeeper wants signal strength measurements. When the bumblebee leaves the store, and detects that it is once again in range of the Gatekeeper, it makes another connection to the Gatekeeper and hands over its data. Employing the Gatekeeper as data sink allows the bumblebee to drop its collected data off and free its memory. The Gatekeeper can then aggregate the data and deliver it to the site specific location estimator, allowing both the bumblebees and the Gatekeeper to be agnostic of the location estimation method.

We believe that the real novelty of BumbleBee lies in the introduction of the Gatekeeper. Serving dual roles, both as distributor of work and as data sink, the Gatekeeper is the driving force in the system. Because bumblebees are mobile and the Gatekeeper is not, employing the Gatekeeper as the distributor of work allows for site specific configurations, and allows the bumblebee to easily move into and out of various deployments without any a priori knowledge of the site specific deployment save for the identity of the Gatekeeper.

3.2 Implementation

3.2.1 The Gatekeeper

The gatekeeper is implemented in the Python programming language so that it can remain relatively platform independent. It is comprised of three main components: the gatekeeper network service, a GUI, and a plugin interface to allow extending the systems functionality. We have implemented plugins for saving/restoring results to/from disk, graphing received signal strength from all BSSIDs across time, logging, and exporting results to a portable format for use outside BumbleBee. As part of the Honeycomb project, we also implemented a Gatekeeper plugin for uploading user tracks to the Honeycomb

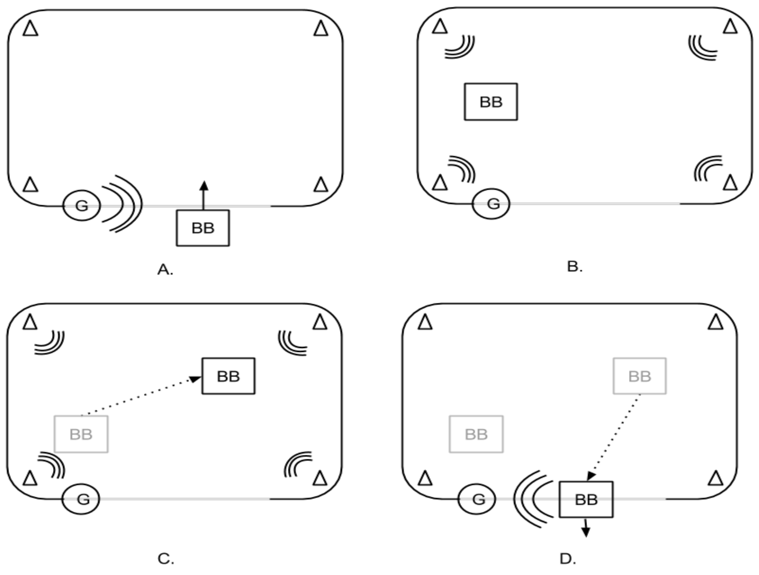

Figure 3.1: BumbleBee Architecture

API. The Gatekeeper server listens on a fixed TCP port (0xb33) for requests from clients through remote procedure calls (via Pythons SimpleXMLRPC-Server). Information exchange with the bumblebees is done in JSON data format. The gatekeeper administrator configures the BSSIDs addresses and minimum polling interval.

3.2.2 The Bumblebee

The bumblebee component is implemented as an Android application written in the Java programming language. The application performs three main tasks: automatic network detection of the Gatekeeper, negotiation with the Gatekeeper, and data collection related activities. These operations require no user interaction and thus we consider this application unobtrusive. As part of the data collection process the bumblebee application will wake up on the negotiated time interval, observe broadcasting wireless networks, and collect and store signal strength values for all requested BSSIDs. It then stamps the collected data with the offset from the time it collected the work from the Gatekeeper and returns to sleep. The periodic waking process is also used to rediscover the Gatekeeper. When the Gatekeeper is rediscovered the application once again negotiates a connection, but this time instead of accepting work it submits the collected data. The data is then discarded from the mobile device.

3.2.3 Communication Mechanism

The bumblebee mobile device communication model is shown in Figure 3.2. The initial state of the bumblebee is Sleeping / Inactive (with respect to BumbleBee activities). Upon discovery of the Gatekeeper the application enters the Handshake state. In this state the application may choose to not accept the requested task, in which case it moves back to the Sleeping / Inactive state. If work is accepted then the application moves into the Data Collection state. The data collection process was described in the previous section. The handshake process is described below. When the bumblebee once again discovers the Gatekeeper is once again enters the Handshake state, but this time it transmits the collected data to the Gatekeeper. Once this is completed, the bumblebee moves back into the Sleeping / Inactive state.

The handshake between Gatekeeper and a bumblebee client happens within the context of a single connection-oriented session and involves a simple two-way handshake. The handshake happens after the bumblebee discovers the Gatekeeper and involves either the requesting of a new data collection information or the submission of collected data. In both cases the bumblebee initially transmits its unique ID to the Gatekeeper. This provides the Gatekeeper an opportunity to perform validation of the ID (perhaps to reference a user database or possibly black/white list of IDs). Assuming the ID is validated, and the client is making a request, a response is sent listing the BSSIDs to monitor, the maximum acceptable interval between samples, and the Gatekeepers current timestamp. The bumblebee is now in the Data Collection state as described above. If the client is unable to support the request it silently drops the request and transitions back to the Sleeping / Inactive state.

If the bumblebee successfully transitioned to the Data Collection state and the Gatekeeper is once again discovered, the handshake process again takes place. After validation the bumblebee then transmits an overall status, the original request timestamp, and the series of collected timestamped signal strength values for all BSSIDs. Regardless of the status of the request the bumblebee moves to the Sleeping / Inactive state. The Gatekeeper will store the request, regardless of status, for later analysis. In our case, this analysis actually occurs on Honeycomb hardware.

cite these or delete them: [6] [21]

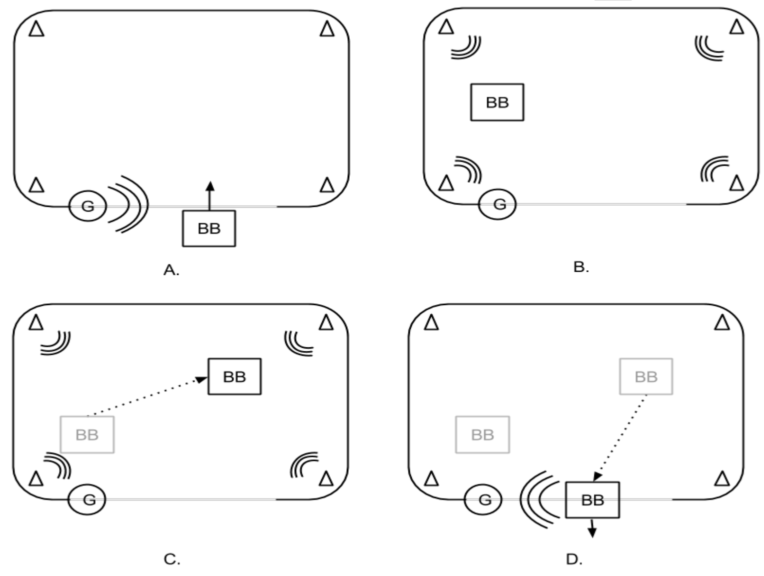

Figure 3.2: BumbleBee State Diagram

Chapter 4

Tech Overview

4.1 Components

4.2 Technologies

4.3 Architecture

Something about averaging measurements to get the fingerprint and how the mobile app can be adjusted for poll frequency and length Something about euclidean distance algorithm

Chapter 5

Testing and Results

5.1 Testing Setup

5.2 Test Variants

5.3 Results

Chapter 6

Discussion

6.1 Interpretation of Results

6.2 Future Work

Do more with the track data instead of just viewing individual tracks better user interface more security

Bibliography

[1] Paramvir Bahl and Venkata N Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, volume 2, pages 775–784. Ieee, 2000.

[2] Paramvir Bahl, Venkata N Padmanabhan, and Anand Balachandran. Enhancements to the RADAR user location and tracking system. Technical report, technical report, Microsoft Research, 2000.

[3] Rashmi Bajaj, Samantha Lalinda Ranaweera, and Dharma P Agrawal. GPS: location-tracking technology. *Computer*, 35(4):92–94, 2002.

[4] Gaddi Blumrosen, Bracha Hod, Tal Anker, Danny Dolev, and Boris Rubinsky. Continuous close-proximity rssi-based tracking in wireless sensor networks. In *Body Sensor Networks (BSN), 2010 International Conference on*, pages 234–239. IEEE, 2010.

[5] Jaewoo Chung, Matt Donahoe, Chris Schmandt, Ig-Jae Kim, Pedram Razavai, and Micaela Wiseman. Indoor location sensing using geomagnetism. In *Proceedings of the 9th international conference on Mobile systems, applications, and services*, pages 141–154. ACM, 2011.

[6] Prabal K Dutta and David E Culler. System software techniques for low-power operation in wireless sensor networks. In *Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design*, pages 925–932. IEEE Computer Society, 2005.

[7] S Hotta, Y Hada, and Y Yaginuma. A robust room-level localization method based on transition probability for indoor environments. In *Indoor Positioning and Indoor Navigation (IPIN), 2012 International Conference on*, pages 1–8. IEEE, 2012.

[8] IEEE 802.1: Wireless LANs. `http://standards.ieee.org/about/get/802/802.11.html`.

[9] Seigo Ito and Nobuo Kawaguchi. Bayesian based location estimation system using wireless LAN. In *Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops. Third IEEE International Conference on*, pages 273–278. IEEE, 2005.

[10] Nobuo Kawaguchi. WiFi location information system for both indoors and outdoors. In *Distributed Computing, Artificial Intelligence, Bioinformatics, Soft Computing, and Ambient Assisted Living*, pages 638–645. Springer, 2009.

[11] Hui Liu, Houshang Darabi, Pat Banerjee, and Jing Liu. Survey of wireless indoor positioning techniques and systems. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 37(6):1067–1080, 2007.

[12] Tomotaka Nagaosa and Hironori Iguchi. Performance evaluation of a Wireless LAN positioning system using spot information. In *ITS Telecommunications (ITST), 2012 12th International Conference on*, pages 512–516. IEEE, 2012.

[13] Nissanka Bodhi Priyantha. *The cricket indoor location system*. PhD thesis, Massachusetts Institute of Technology, 2005.

[14] Michael Quan, Eduardo Navarro, and Benjamin Peuker. Wi-Fi Localization Using RSSI Fingerprinting. 2010.

[15] Vinay Seshadri, Gergely V Zaruba, and Manfred Huber. A bayesian sampling approach to in-door localization of wireless devices using received signal strength indication. In *Pervasive Computing and Communications, 2005. PerCom 2005. Third IEEE International Conference on*, pages 75–84. IEEE, 2005.

[16] Paurav Shukla and Barry J Babin. Effects of consumer psychographics and store characteristics in influencing shopping value and store switching. *Journal of Consumer Behaviour*, 12(3):194–203, 2013.

[17] Edip Toplan and Cem Ersoy. RFID based indoor location determination for elderly tracking. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pages 1–4. IEEE, 2012.

[18] Daniel Turner, Stefan Savage, and Alex C Snoeren. On the empirical performance of self-calibrating wifi location systems. In *Local Computer*

Networks (LCN), 2011 IEEE 36th Conference on, pages 76–84. IEEE, 2011.

[19] Jie Xiong and Kyle Jamieson. Towards fine-grained radio-based indoor location. In *Proceedings of the Twelfth Workshop on Mobile Computing Systems & Applications*, page 13. ACM, 2012.

[20] Jie Xiong and Kyle Jamieson. ArrayTrack: A Fine-Grained Indoor Location System. In *NSDI*, pages 71–84, 2013.

[21] Rong Xu, Zhiyuan Li, Cheng Wang, and Peifeng Ni. Impact of data compression on energy consumption of wireless-networked handheld devices. In *Distributed Computing Systems, 2003. Proceedings. 23rd International Conference on*, pages 302–311. IEEE, 2003.

Vita

TODO: VITA

Permanent address: mplinder@utexas.edu

This report was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.