Unobtrusive, Pervasive, and Cost-Effective Communications with Mobile Devices

Dissertation

Presented in Partial Fulfillment of the Requirements for the Degree Doctor of Philosophy in the Graduate School of The Ohio State University

By

Adam C. Champion, B.S., M.S.

Graduate Program in Computer Science and Engineering

The Ohio State University

2017

Dissertation Committee:
Professor Dong Xuan, Advisor
Professor Feng Qin
Professor Ten H. Lai
Abstract

Mobile devices such as smartphones are ubiquitous in society. According to Cisco Systems, there were eight billion mobile devices worldwide in 2016 [1], which surpassed the human population [2]. Mobile devices and wireless network infrastructure form an “electronic world” of signals that is part of daily life. However, navigating this world with users’ devices is challenging. The volume of signals may confuse users, wireless communications often require manual connection establishment, and latency may be large (such as Bluetooth device discovery). Pervasive mobile device communications offer large-scale measurement opportunities when many devices connect to wireless networks. For example, base stations to which devices connect can indicate human mobility patterns. But existing work only studies coarse-grained cellular call records and datasets used in wireless local area network (WLAN) studies typically consist of laptops. Besides, existing vehicular communications technologies tend to be expensive and available for new vehicles only.

This dissertation studies three topics that arise in the electronic world: unobtrusive communications among mobile devices without manual connection establishment; pervasive mobile device communications measurement; and cost-effective vehicular communication among mobile devices. First, we design Enclave that enables unobtrusive communication among mobile devices without network connections or configurations. Unobtrusive communication is efficient wireless communication that
does not require user interruption such as manual device connection or network configuration. Enclave consists of a delegate mobile device (such as an unused phone) that interposes between a user’s “master” device (such as her smartphone) and the electronic world. Enclave communicates between the master and delegate devices using wireless name communication and picture communication. Second, we study a new dataset with over 41 million anonymized (dis)association logs with WLAN access points (APs) at The Ohio State University (via the osuwireless network) over 139 days from January to May 2015. The dataset includes more than 5,000 university students with their birthdays, genders, and majors, which are made available after anonymization. Using mobility entropy as our metric, we find that entropy increases with age (for 19–21-year-olds), students’ entropic rates of change vary with majors, and all students’ long-term entropy follows a bimodal distribution that has not been previously reported. We design a mobile application that localizes users indoors for 73 campus buildings using WLAN site survey information. In general, our app achieves room-level accuracy. Finally, we propose SquawkComm for cost-effective vehicular communication using mobile devices and FM signals. SquawkComm encodes data as audio, which is sent via inexpensive FM transmitters that plug into vehicles’ cigarette lighters and received via vehicle stereos. SquawkComm uses a new physical-layer coding scheme and link-layer mechanisms for channel access. Our experimental evaluation illustrates the promise of our work in the electronic world. We conclude with directions for future work.
This is dedicated to my family, my advisor, and my friends.
Acknowledgments

Numerous people have provided me vital assistance during this long, challenging “research journey.” Here, I express my gratitude to all of them.

First, I thank my advisor, Prof. Dong Xuan, for his insight and support during my Ph.D. study. He taught me the importance of clear problem definition, solving problems among multiple levels of abstraction, and writing with precision and concision. His passion for impactful, high-quality work with practical utility guided my research efforts and it will guide my future work.

Second, I thank several faculty members at The Ohio State University (OSU) for providing me a strong education. Profs. Feng Qin and Srinivasan Parthasarathy taught me a great deal about computer systems and data mining. Prof. Ness Shroff illuminated queueing theory and stochastic processes in computer networks. Prof. Yuan F. Zheng provided useful insights about computer vision, robotics, and machine learning. I am grateful for Profs. Feng Qin, Mikhail Belkin, and Raghu Machiraju’s incisive comments during my candidacy exam. I appreciate Profs. Feng Qin and Ten H. Lai serving on my final dissertation committee.

Third, I thank the Department of Computer Science and Engineering at OSU for my position as a full-time lecturer and the opportunity to “pay forward” to my students.
Fourth, I thank my collaborators for their insight and discussions, including Prof. Paul Y. Cao (from University of California, San Diego), Prof. Linghe Kong (from Shanghai Jiao Tong University), Dr. Sihao Ding, Dr. Jin Teng, Dr. Xinfeng Li, Dr. Boying Zhang, Dr. Zhezhe Chen, Dr. Boxuan Gu, Dr. Sriram Chellappan, Dr. Zhimin Yang, Dr. Wenjun Gu, Dr. Xiaole Bai, Dr. Xun Wang, Gang Li, Ying Li, Qiang Zhai, Guoxing Chen, Cheng Zhang, Steve Romig, and Austin Gilliam. Working with them has taught me a great deal about computer science, electrical engineering, and applied mathematics. I thank Prof. Cao for his insights on large-scale data analysis and mobility entropy. Prof. Kong’s constructive criticism assisted my development of SquawkComm and writing. Dr. Ding and Ying taught me about computer vision and machine learning. Dr. Teng and Guoxing described electrical engineering and subtle mathematical concepts. Drs. Zhezhe Chen and Boxuan Gu sharpened my knowledge of high-performance computing and malicious code detection, respectively. Drs. Teng and Li as well as Gang and Qiang assisted with experiments and writing. Dr. Yang explained Bluetooth, Wi-Fi, and distributed systems design. Dr. Bai provided valuable insights about mathematical rigor and research. Dr. Wang introduced me to graduate study and machine learning. Cheng has illuminated deep learning, neural networks, and artificial intelligence. Steve and Gang played vital roles in obtaining the WLAN log dataset that Chapter 4 analyzes. Austin and Qiang assisted in SquawkComm’s development and experiments. Dr. Xinfeng Li’s dissertation [3] was helpful in writing mine. Dr. Theodore Pavlic’s dissertation [4] provided superb \LaTeX support; Dr. Chris Monson’s \LaTeX Makefile [5] was very helpful. Günter Lorenz provided the FMLIST database [6] of FM radio stations for SquawkComm.

Finally, I thank my family for their continued love and support.
Vita

September 11, 1983 ........................................... Born – Columbus, OH, USA

June 2007 ...................................................... B.S., with distinction, Computer Science and Engineering

June 2012 ...................................................... M.S., Computer Science and Engineering

2007–present .............................................. Univ. Fellow, The Ohio State University

2014–present .............................................. Full-Time Lecturer, Computer Science and Engineering, The Ohio State University

Publications

Research Publications – Conference


Research Publications – Journal


Fields of Study

Major Field: Computer Science and Engineering

Studies in:

Computer Networking
Prof. Dong Xuan
Prof. David Lee

Distributed Systems
Prof. Feng Qin
Prof. Srinivasan Parthasarathy

Artificial Intelligence
Prof. James Davis
# Table of Contents

- Abstract .................................................................................................................. ii
- Dedication ................................................................................................................ iv
- Acknowledgments ........................................................................................................ v
- Vita ............................................................................................................................ vii
- List of Tables ............................................................................................................... xii
- List of Figures ............................................................................................................. xiii

1. Introduction ............................................................................................................... 1
   - 1.1 Unobtrusive Communications with Mobile Devices ........................................ 4
   - 1.2 Pervasive Mobile Device Communications Measurement .................................. 5
   - 1.3 Cost-Effective Vehicular Communication with Mobile Devices ....................... 7
   - 1.4 Organization ...................................................................................................... 8

2. Related Work ............................................................................................................ 9
   - 2.1 Unobtrusive Communication via Mobile Devices ............................................ 9
   - 2.2 Pervasive Mobile Device Communications Measurement ............................. 11
   - 2.3 Cost-Effective Vehicular Communication with Mobile Devices ................... 14

3. Unobtrusive Communication Among Mobile Devices ........................................... 17
   - 3.1 Overview ........................................................................................................... 17
   - 3.2 Design Overview ............................................................................................. 22
     - 3.2.1 NameCast: Enclave-to-Outside Communication ........................................... 23
     - 3.2.2 PicComm: Enclave-to-Master Communication ............................................ 33
List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Organization of the dissertation</td>
</tr>
<tr>
<td>3.1</td>
<td>Average Bluetooth and Wi-Fi discovery times.</td>
</tr>
<tr>
<td>3.2</td>
<td>Master device power consumption in PicComm.</td>
</tr>
<tr>
<td>4.1</td>
<td>Student demographic data.</td>
</tr>
<tr>
<td>4.2</td>
<td>Long-term metrics for each academic major.</td>
</tr>
<tr>
<td>4.3</td>
<td>Short-term daily $S_{td}$ for each academic major.</td>
</tr>
<tr>
<td>4.4</td>
<td>Short-term metrics for each day of a week</td>
</tr>
<tr>
<td>4.5</td>
<td>Number of floors for each building at OSU with WLAN site survey data.</td>
</tr>
<tr>
<td>5.1</td>
<td>Limitations of existing work for vehicular communications requirements.</td>
</tr>
<tr>
<td>5.2</td>
<td>SquawkLink frame types.</td>
</tr>
</tbody>
</table>
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>The electronic world.</td>
<td>18</td>
</tr>
<tr>
<td>3.2</td>
<td>Enclave system architecture and supporting technologies.</td>
<td>22</td>
</tr>
<tr>
<td>3.3</td>
<td>Motivating topology for NameCast.</td>
<td>27</td>
</tr>
<tr>
<td>3.4</td>
<td>Wi-Fi control frame and Bluetooth name for NameCast.</td>
<td>29</td>
</tr>
<tr>
<td>3.5</td>
<td>OCRHash grouping example.</td>
<td>36</td>
</tr>
<tr>
<td>3.6</td>
<td>Average number of received messages for each device.</td>
<td>41</td>
</tr>
<tr>
<td>3.7</td>
<td>Average time needed for devices to receive messages.</td>
<td>41</td>
</tr>
<tr>
<td>3.8</td>
<td>Throughput of PicComm.</td>
<td>43</td>
</tr>
<tr>
<td>4.1</td>
<td>Observed long-term mobility entropy distribution over a 139-day period.</td>
<td>48</td>
</tr>
<tr>
<td>4.2</td>
<td>Location granularity as buildings.</td>
<td>54</td>
</tr>
<tr>
<td>4.3</td>
<td>Stay time estimation for inter-building AP connections in a person’s trajectory.</td>
<td>54</td>
</tr>
<tr>
<td>4.4</td>
<td>Histograms of long-term entropy $S_{td}$ for different age groups.</td>
<td>59</td>
</tr>
<tr>
<td>4.5</td>
<td>Average entropies across age groups for each day of the week.</td>
<td>61</td>
</tr>
<tr>
<td>4.6</td>
<td>Histograms of 20-year-olds’ (sophomores’) $S_{td}$ for six different groups of students based on their majors.</td>
<td>62</td>
</tr>
</tbody>
</table>
4.7 Histograms of 20-year-olds’ (sophomores’) \( S_{ii} \) for six different groups of students based on their majors. ........................................... 63

4.8 Average entropy for each day of the week for males and females. .... 65

4.9 Males’ and females’ average daily entropies during the 139-day period. 67

4.10 Overall long-term entropies \( S_{td} \) and \( S_{ii} \) and their convergence results of students based on cumulative trajectories. ......................... 68

4.11 Average entropy for each day of the week. .......................... 69

4.12 OSU buildings with AP deployment information. .................. 71

4.13 WLAN site survey database schema. .................................. 74

4.14 Screenshot of mobile application for indoor localization. .......... 76

5.1 SquawkComm workflow. ............................................... 85

5.2 Frequency spectrum of FM baseband signal. ........................ 86

5.3 Equal loudness contours vs. frequency. ........................... 88

5.4 Road noise in vehicles. ............................................... 88

5.5 Model of FM communication on roadways \((\ell = 2)\). ............... 93

5.6 SquawkLink frame structure. ....................................... 95

5.7 SquawkComm overall latency. ..................................... 100

5.8 SquawkComm bit error rate. ..................................... 100

5.9 Components of SquawkComm’s latency. .......................... 100

xiv
Chapter 1: Introduction

Today, mobile devices such as smartphones are increasingly pervasive in society and they are an important part of the “electronic world” that permeates daily life. Cisco Systems estimates there were eight billion mobile devices and connections worldwide in 2016 [1], which exceeds the world population of 7.4 billion people [2]. According to the International Telecommunications Union (ITU), 95% of the world’s population lives in areas with at least second-generation (2G) cellular network coverage [7]. Many mobile devices are smartphones with powerful processors, gigabytes of memory, and short-range wireless communications capabilities such as Bluetooth, Wi-Fi, and cellular connectivity.\(^1\) Almost 1.5 billion smartphones were sold in 2016 alone [8] and 77% of all adults in the U.S. own a smartphone [9]. In general, people carry their mobile devices with them everywhere they go. Wireless network infrastructure (such as cellular base stations and Wi-Fi access points) is widely deployed in the physical world, spreading wireless signals. Thanks to mobile devices and infrastructure, these signals form an electronic world with which we interact daily.

Navigating the electronic world effectively with users’ mobile devices presents the following challenges:

\(^1\)In this dissertation, Bluetooth refers to “classic” Bluetooth, which is supported by most mobile devices sold since 2000.
- **Confusion**: The sheer volume and different types of wireless signals may confuse users. Most mobile devices support third-generation (3G) and fourth-generation (4G) cellular communications such as the 4G Long Term Evolution (LTE) standard. In addition, these devices support various short-range wireless communications protocols such as Bluetooth and Wi-Fi.

- **Manual connection establishment requirement**: Wireless communications typically entail tedious connection establishment and network management. For example, Wi-Fi infrastructure requires users to associate with access points (APs) in wireless local area networks (WLANs) and enter credentials such as passwords in order to use the networks. Such procedures are cumbersome for users. For instance, users need to remember multiple passwords to use multiple WLANs if each WLAN requires a separate password.

- **High network latency**: Network latency may be very high, especially when discovering nearby devices. For example, Bluetooth device discovery takes at least 10.24 seconds to find nearby devices’ Media Access Control (MAC) addresses; mapping MAC addresses to human-readable device names takes \(~1\) second for each device. This delay is very irritating to users and inappropriate for dynamic environments such as vehicular communication.

- **Pervasive device communications measurement**: Pervasive mobile devices generate considerable data as they communicate. Effectively measuring such communications is challenging, especially if many devices connect to wireless networks. Since people’s mobile devices accompany them on the move, their
interactions with wireless network infrastructure such as WLAN APs and cellular towers can indicate their mobility. Human mobility has applications in epidemic modeling, urban planning, and resource management for mobile communications [10,11]. In addition, these interactions can guide administrators for facilities planning and emergency management purposes. However, most work on user mobility focuses on call detail records (CDRs) [12,13] that consist of users’ calls among each other. This body of work measures users’ mobility based on the cellular base stations to which their devices associate. As these base stations are often deployed kilometers apart, their data granularity is coarse. Previous studies of devices’ wireless connectivity traces have been carried out on university campuses’ WLANs [14–19], but existing datasets tend to use many laptops and few mobile devices [20]. To our knowledge, these studies do not study pervasive mobile device communications with demographic information.

- **Expensive or specialized communications equipment**: Some electronic communications equipment may be too expensive to be available to a critical mass of users. For examples, radar is usually only available on high-end vehicles [21,22] and some mobile devices cannot receive FM radio signals [23]. Users should not need such equipment in order to interact effectively with the electronic world.

This dissertation studies three topics regarding mobile device communications that arise in the electronic world:
– **Unobtrusive communications with mobile devices**: How can mobile devices communicate efficiently without requiring user interruption (such as manual device connection or network configuration)?

– **Pervasive mobile device communications measurement**: How can we measure users’ pervasive mobile device communications in wireless networks?

– **Cost-effective vehicular communication with mobile devices**: How can users’ mobile devices in vehicles communicate with each other rapidly in various locations, including remote areas in which there is little infrastructure?

We elaborate each topic in the following sections.

### 1.1 Unobtrusive Communications with Mobile Devices

Today, it is very hard for mobile devices to achieve *unobtrusive communications* that do not require user interruption (such as manual connection establishment and network configuration). For examples, users need to enter their account credentials (*e.g.*, passwords) in order to associate with encrypted WLANs and discovering nearby devices using Bluetooth takes at least 10.24 seconds. (External devices often require tedious Bluetooth pairing prior to use.) These examples illustrate the “electronic barrier” between people using mobile devices and the electronic world. This barrier arises for two reasons. First, many current wireless communications protocols are designed for reliable communication between senders and receivers, which entails delivering each byte of content correctly. Second, enabling wireless connectivity on mobile devices may expose users to malicious code [24,25] or indecent content. We propose *Enclave* for secure and unobtrusive communication using mobile devices. Enclave is a
delegate mobile device (“enclave device”) that mediates access to the electronic world from the user’s “master device” (i.e., her primary mobile device). If Enclave is compromised, the user can reset it to a “known good” state. Enclave uses two supporting technologies: NameCast and picture communication (PicComm). NameCast forwards information among mobile devices using Bluetooth device names, Wi-Fi service set identifiers (SSIDs), and erasure coding for reliable forwarding. PicComm displays text on the enclave device’s screen that the master device reads optically via its camera. We design a feedback mechanism using sound that informs the former device of PicComm’s communications success or failure. Our real-world experiments show that Enclave forwards information among mobile devices effectively via NameCast and PicComm.

1.2 Pervasive Mobile Device Communications Measurement

Recently, mobile phone CDR data and WLAN data have become available at large scale [13,26]. Numerous researchers have proposed human mobility models based on these real-world datasets [15,20,27–29]. However, existing datasets of device usage in wireless networks generally do not consider handheld mobile devices as “first-class citizens.” To our knowledge, these datasets do not study mobility in terms of demographic categories such as age and gender [20]. In a seminal work, Song et al. [30] showed that mobility entropy can characterize the predictability of human mobility. A higher entropy for a group of people indicates that the group’s mobility is less predictable on average. In addition, the entropy of a person’s mobility can be interpreted as the amount of irregularity in her mobility. A body of human mobility studies uses the mobility entropy metric [31–33]. In this dissertation, we use this metric to
study mobility differences with respect to demographic categories via a comprehensive WLAN dataset collected at The Ohio State University. Our dataset consists of over 5,000 students’ (dis)associations with APs at OSU for the osuwireless WLAN captured during 139 days from Tuesday, January 13, 2015, to Sunday, May 31, 2015 along with each student’s age, gender, and major. We group student majors as follows: business, education, engineering, health, social science, and science majors. Each student in the dataset is represented via an anonymized identifier (ID). We present the first study of the predictability of human mobility at a public university with respect to demographic differences. Our findings contradict “common-sense” views that mobility entropy of traditional (undergraduate) college students, a group with similar ages and backgrounds, would be similar for students with different ages and majors. Our results challenge the common-sense view that expects students’ mobility patterns to follow those of the general population (i.e., heavy-tail distributions such as Pareto or Weibull distributions [15]). Actually, we discover that students’ overall long-term entropy greatly varies with age for 19–21-year olds. The rate of change across age groups within each major is non-uniform. Students’ overall long-term entropy follows a bimodal distribution that has not been previously reported in the literature. We find no significant difference between males’ and females’ long-term entropies, which confirms the results of Song et al. [30]. However, comparison of entropies on a daily basis reveals that females’ daily entropies slightly exceed males’, especially on Tuesdays and Thursdays, and this holds throughout the entire 139-day observation period. Furthermore, our dataset contains WLAN site survey information for 73 buildings at OSU. We use this information and design a mobile application (app) that localizes users in these buildings via multilateration using their devices’ received
signal strength indicators (RSSIs). Our app shows users’ indoor locations as well as their locations via Mapbox maps [34]. Our experiments show that, in general, users’ locations are mapped with room-level accuracy.

1.3 Cost-Effective Vehicular Communication with Mobile Devices

Wireless vehicular communication in physical proximity has applications such as drivers notifying other drivers of injured passengers they are taking to hospitals and social interaction among passengers in vehicles with common destinations [35]. In order to enable such applications, latency must be minimal (vehicles may only be nearby for a few seconds), manual connection establishment should not be required, and communications should be cost-effective for vehicle occupants whose mobile devices accompany them everywhere. However, as discussed previously, Bluetooth has high latency. Ad hoc Wi-Fi is unavailable on commercial off-the-shelf (COTS) smartphones unless they are “rooted” or “jailbroken” [36]. Wi-Fi Direct requires connection establishment, which may take two minutes, and switching Wi-Fi “mobile hotspots” on and off takes several seconds [37]. A practical cost-effective system would be very desirable for vehicular communication in physical proximity. Such a system should support a wide variety of vehicles (both new and old), use COTS equipment as much as possible, and provide both physical-layer and link-layer support. We propose SquawkComm, a system that uses mobile devices and FM signals for such purposes. SquawkComm does not require connection establishment or wireless infrastructure (e.g., cellular base stations and WLAN APs) that may be unavailable in remote areas. SquawkComm encodes data as audio that smartphones send to inexpensive COTS FM transmitters. SquawkComm uses two supporting technologies, SquawkCode and SquawkLink, for
physical-layer data encoding and link-layer channel access control. Experiments in laboratory and real-world vehicular environments show that SquawkComm achieves latency within a few seconds.

1.4 Organization

Table 1.1 shows the organization of this dissertation.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Solution</th>
<th>Chapter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unobtrusive communication with mobile devices</td>
<td>Enclave</td>
<td>Chapter 3</td>
</tr>
<tr>
<td>Pervasive mobile device communications measurement</td>
<td>WLAN log analysis with demographics; mobile app for indoor localization</td>
<td>Chapter 4</td>
</tr>
<tr>
<td>Cost-effective vehicular communications with mobile devices</td>
<td>SquawkComm</td>
<td>Chapter 5</td>
</tr>
</tbody>
</table>

Table 1.1: Organization of the dissertation

The rest of the dissertation is structured as follows. Chapter 2 reviews related work regarding Enclave, wireless log analysis, and SquawkComm. Chapter 3 presents Enclave for unobtrusive communication among mobile devices. Chapter 4 analyzes OSU students’ pervasive mobile device WLAN logs with their demographics and discusses our mobile app for indoor localization on campus. Chapter 5 presents SquawkComm for cost-effective vehicle communication using mobile devices. Chapter 6 concludes the dissertation and gives directions for future work. Appendix A defines technical terms used in this dissertation.
Chapter 2: Related Work

This chapter reviews related work regarding unobtrusive, pervasive, and cost-effective computing and communications with mobile devices. Section 2.1 discusses work related to unobtrusive communication. Section 2.2 describes related measurements of pervasive mobile device communications. Finally, Section 2.3 presents related work for vehicular communication.

2.1 Unobtrusive Communication via Mobile Devices

Mobile Social Networking

There are many mobile social networking works that we classify as centralized and distributed based on the underlying system architecture. A representative centralized system is Social Serendipity [38], which detects proximate others via Bluetooth and queries a central profile database to match people based on commonalities. Nokia Sensor [39] is a representative distributed system that lets proximal users detect each other and share information via a Bluetooth connection. Existing work does not consider the “gap” between the electronic and physical worlds (i.e., when people meet each other in physical proximity, they cannot easily view related electronic profiles without connection establishment). However, strangers may not want to establish such connections. Our work, Enclave, bridges
this gap via NameCast, which leverages Wi-Fi and Bluetooth names to wirelessly provide social information to nearby people without connection establishment.

**Proximate Communication via Bluetooth and Wi-Fi** Several works aim to control proximate communication using Wi-Fi Service Set Identifiers (SSIDs) and Bluetooth names. Beacon stuffing [40] adds information to Wi-Fi access point (AP) beacons to realize Wi-Fi communication without association. Neighborcast [41] forms multicast groups among disparate Wi-Fi clients regardless of AP association. Researchers have also leveraged Bluetooth device names for various contexts including automatic configuration of mobile ad hoc networks [42, 43], Bluetooth device name interactions [44], and “proactive displays” [45]. The Wi-Fi SSID based approach can broadcast to a wide range of nearby devices, but it can only deliver a small amount of information. The Bluetooth-name-based approach can deliver more information, but it only supports point-to-point communication. E-Shadow [46] uses both Wi-Fi and Bluetooth to publish personal information; it covers a large communication range and delivers a large volume of information. However, E-Shadow does not aim for communication among neighbors. Our Enclave differs from E-Shadow as we piggyback message delivery atop Bluetooth device discovery for rapid unobtrusive communication.

**Wi-Fi Direct** Wi-Fi Direct [47] is a technology enabling Wi-Fi-enabled devices to detect others nearby, after which wireless connections are directly established. It forms peer-to-peer-like groups among devices such as cameras and printers. This technology eases information sharing among neighboring mobile devices. However, content made available over a Wi-Fi Direct group connection remains restricted to
specific applications [47,48]. In this sense, there remains a gap between the electronic and the physical world. Our Enclave work helps close the gap via NameCast.

**Mobile Device Communications** Our Enclave and its supporting technologies can be implemented on a single mobile device such as a smartphone. Many such devices support data encryption mechanisms, Bluetooth and Wi-Fi communication, and cameras that can be used to compartmentalize sensitive data. Electronic communication can access information from physically proximate mobile devices, location information, and Internet servers. Similarly, cameras and optical character recognition (OCR) technology can be used to parse textual information transmitted via a visual channel. Perli et al. [49] proposed using pairs of cameras for fast information exchange. However, they neither consider this a security measure nor explore its security aspects. They use two-way “picture taking” for feedback, which requires front cameras on both devices. Enclave can use wireless signals and sounds as feedback, which is more flexible.

### 2.2 Pervasive Mobile Device Communications Measurement

**Mobile Device Datasets** The collection and analysis of datasets from mobile devices have attracted considerable attention from the research community. We refer the reader to [12,13,20] for surveys of data collection methodologies, results, and applications. There are three main types of mobile device data: devices’ GPS locations, locations determined via cellular towers, and wireless connectivity logs via Bluetooth, wireless local area networks (WLANs), or radio frequency IDs (RFIDs) [20]. The granularity of GPS and WLAN data is on the order of meters whereas that of cellular towers depends on their geographic distribution; typically, distances among towers are.
on the order of kilometers. GPS and cellular networks offer potential coverage over large areas whereas WLAN logs offer relatively accurate measurements of user behaviors in more localized environments. Devices’ WLAN logs are mainly captured on university campuses, via public transportation, and in office buildings, cities, and conferences. Biswas et al. [26] measure wireless interference and application usage on end systems using Cisco Meraki APs deployed throughout the U.S. Several studies have captured devices’ wireless connectivity on campuses such as Dartmouth College [14], North Carolina State University and KAIST [15], University of North Carolina [16], Massachusetts Institute of Technology [17], University of California, San Diego [18], University of Southern California [19], University of Connecticut [50], and University of Wisconsin-Madison [51]. Wei et al. [52] study behaviors of mobile devices on a university campus. In their Reality Mining work [53], Eagle and Pentland examined Bluetooth and cellular connectivity among 100 users with mobile phones. Unlike this body of work, this dissertation measures students’ mobile device WLAN communications with respect to their demographic categories.

Public WLAN Logs Public real-world WLAN logs are available from repositories such as CRAWDAD [54]. However, existing WLAN logs only contain small numbers of mobile devices, most of which are laptops [20]. These WLAN logs were collected between 1999 and 2006 using at most a few hundred APs. Besides, most work focuses on physical measurements and analyses of traces. To our knowledge, no studies have used demographic information along with WLAN logs to infer human mobility. In contrast, this dissertation studies demographic data in much finer detail than these
studies. We examine the predictability of several thousand students’ mobility on a large university campus considering differences in their majors, ages, and genders.

**Mathematical Models** Researchers have proposed various mathematical models to describe human mobility from mobile device datasets. González et al. [28] measure the regularity of human mobility via the radius of gyration and entropy from a country’s cellular logs. Similarly, Song *et al.* [30] derive bounds for predictability of human mobility from a country’s cellular logs and demographic data such as gender. Using entropy as a metric, Qin *et al.* [31] investigate the predictability of people at home and work. Lu *et al.* [32] use entropy measured via cellular log data to study the predictability of West Africans’ mobility. Cho *et al.* [33] analyze location periodicity in human mobility using entropy calculated from two location-based social network datasets and one cellular log dataset. Rhee *et al.* [15] show via GPS logs that human mobility patterns are statistically similar to Lévy walks, which are random walks with self-similar jumps [20]. Tuduce and Gross [55] find that WLAN traces follow a power-law model, which has been observed in many other datasets [56]. Kim *et al.* [57] refine this model to a lognormal model. Hsu et al. [58] propose a time-variant community mobility model that captures characteristics of WLAN traces. In another work by Song *et al.* [59], continuous-time random walks (CTRWs) were shown to conflict with empirical data, and the authors proposed an individual mobility model incorporating the number of unique locations visited and the visitation frequency. Other well-known models include the random waypoint model [18] and the statistical mobility model [60]. This dissertation considers mobile devices as “first-class citizens” along with students’ demographics. We measure the mobility entropy [30] of students’
movements to infer the predictability of their movements. In addition, we present a mobile application for indoor localization using WLAN site survey information.

2.3 Cost-Effective Vehicular Communication with Mobile Devices

Vehicle Communications Communications among vehicles have been studied for a long time and we refer the reader to [61] for surveys of this topic. Some work [62–64] studies sharing Internet connections among vehicles and cluster-based vehicular communication. RoadSpeak [65] lets drivers speak with each other via 3G cellular infrastructure. Vehicular testbeds such as CarTel [66] and DieselNet [67] have been developed for delay tolerant networking where vehicles serve as “data mules” for nearby data transfer. Dedicated Short Range Communications (DSRC) vehicular safety technology is under development; in DSRC, vehicles rapidly send messages to each other (at 1–10 Hz) regarding speed, braking, and turning [61]. Bai et al. measured DSRC’s performance on the road [68]. Some work [69,70] has integrated DSRC with smartphones using cellular networks, WLANs, or custom silicon [71]. NHTSA intends to expedite DSRC development and installation in new vehicles [72,73]. In this dissertation, our SquawkComm complements DSRC broadcast, but we use RTS/CTS with commercial off-the-shelf (COTS) FM transmitters, smartphones, and vehicle stereos to avoid broadcast storms to which DSRC is susceptible [74].

Participatory Sensing Regarding Vehicles Researchers have developed participatory sensing systems using people’s smartphones in vehicles. (We refer the reader to surveys [75,76] for more information.) Nericell [77] senses road and traffic conditions using smartphones’ accelerometers, GPS, and microphones. GreenGPS [78]
uses smartphones’ GPS and vehicular OBD-II data for fuel-efficient routing in cities. SignalGuru [79] predicts traffic signal schedules using smartphones’ cameras and accelerometers. WreckWatch [80] uses smartphones’ GPS, microphones, and accelerometers to detect accidents and report them to first responders. VTrack [81] uses smartphones’ GPS and Wi-Fi to estimate traffic delays. Unlike this body of work, our SquawkComm enables cost-effective low-latency communication among nearby vehicle occupants using their smartphones, FM transmitters, and vehicular stereos.

**Smartphone Communications** Recently, various systems have been developed for communication among smartphones. Better Approach To Ad Hoc Networking (B.A.T.M.A.N.) uses ad hoc Wi-Fi for smartphone communication without infrastructure [82]. FireChat uses Bluetooth and ad hoc Wi-Fi for similar purposes [83]. Smartphone applications such as Zello, Heytell, and Voxer [84] enable voice chat over Wi-Fi or cellular infrastructure. GoTenna uses expensive VHF antennas ($199 per pair) with smartphones for communication over a few kilometers without infrastructure [85]. Hu et al. [37] minimize smartphones’ switching time between Wi-Fi “AP” and “client” modes in order to enable phone-to-phone communication among vehicles, but their latency ($\geq 3\, s$) is too high for our purposes. Su et al. [86] used Wi-Fi for communication between smartphones in vehicles. However, B.A.T.M.A.N. requires rooting or jailbreaking smartphones, which entails specialized knowledge. Ad hoc Wi-Fi is typically unavailable on COTS smartphones [36]. Wi-Fi Direct requires manual connection establishment, which can take up to two minutes [47,87]. Bluetooth requires at least 10 seconds for nearby device inquiry as well as manual connection establishment. In contrast, our SquawkComm uses vehicle occupants’ smartphones
as well as vehicular FM transmitters and stereos for cost-effective low-latency communication among nearby occupants.

Researchers have explored FM technology for communications. Rahmati et al. [88] and Paolini et al. [89] developed systems using RDS for communication among smartphones. Yu et al. [90] proposed a smartphone communication system for nearby people using FM audio. Wang et al. [91] developed a hybrid system using smartphones and FM transmitter circuit boards to share music playlists among people in proximity. In addition, researchers have used RDS for synchronization in wireless sensor networks [92] and among Wi-Fi APs [93]. By contrast, our SquawkComm achieves communication among nearby vehicle occupants using their smartphones as well as automotive FM transmitters and stereos.

**DSRC and Privacy** Vehicles’ rapid DSRC broadcast messages can reveal their MAC addresses to attackers, which is a serious privacy issue. Researchers have investigated various schemes for assigning vehicles non-identifying pseudonyms instead of MAC addresses and updating pseudonyms on the road [94]. In this dissertation, SquawkComm communicates using FM audio, not DSRC, and our messages do not contain geolocations. In SquawkComm, each sender randomly generates a new vehicle ID (VID) upon message transmission, which helps SquawkComm resist fingerprinting via MAC addresses [95,96].
This chapter proposes Enclave, a system that provides unobtrusive and secure communication between a mobile device such as a smartphone and the electronic world of prolific wireless signals that surrounds us. Today, interacting with this world via mobile devices is very tedious and the security of such interactions has room for improvement. As we explain, gathering electronic information unobtrusively and securely is desirable in certain cases.

3.1 Overview

Electronic signals mainly consist of traffic between individual users and base stations such as Wi-Fi access points (APs). In the future, we imagine that people extract useful information from these signals just as our eyes capture visual information from the physical world. As Figure 3.1 shows, electronic signals have two main types of useful information:

- **Information from static objects**: Static objects include buildings. For example, government agencies can broadcast wireless bulletins near their buildings. Stores can broadcast wireless advertisements and distribute electronic coupons to passersby who would like to receive them.
Figure 3.1: The electronic world. Ubiquitous wireless signals are emitted from stationary locations such as historical buildings and stores. Mobile objects like passersby or drivers broadcast personal social networking profiles [38, 39, 46, 97–99], accident notifications, and so on. The person at bottom right uses one wireless device to interact with the electronic world; the person’s smartphone is private.

- **Information from mobile objects**: Mobile objects include people or vehicles. People may want to introduce themselves to each other for socialization or job search purposes. Drivers may warn others of accidents they just witnessed.

In addition, hybrid sources of information may arise. For instance, mobile users may forward bulletins with emergency notices to others. Similarly, nearby drivers may relay accident information and people may forward store coupons to others due to incentive mechanisms [99].

However, life with such wireless communications is not yet reality. Accessing information in the electronic world is cumbersome for two reasons:

- **Reliable communications goal**: Current wireless technology is designed with the goal of reliable information transmission between senders and receivers. Thus, network configuration and connection establishment are necessary in most
cases. Complex mechanisms are enforced for reliability such as error correction for perfect reception accuracy. However, this is overkill for large-scale information dissemination. People cannot freely share information to nearby others without manual configuration and connection establishment. In some cases, such accuracy is not required. These well-intentioned mechanisms actually form an "electronic barrier" among people.

- **Malicious content:** Attackers may exploit users who receive information without filtering. Exchanged wireless information may contain malicious code [24,25] and indecent content. Hence, people tend to shield themselves behind this electronic barrier.

We propose Enclave to safely remove the electronic barriers and achieve unobtrusive communication. We do so via NameCast and we introduce secure communication via PicComm. Enclave is a delegate wireless device that helps people’s *master devices* (*e.g.*, their smartphones) communicate wirelessly (*e.g.*, their smartphones).

Enclave is a separate mobile device that may not have a data plan and runs our software. Separating functionality between two devices may seem inconvenient, particularly as mobile devices tend to consolidate many functions on one device. While Enclave can be implemented on the owner’s device, this approach is actually too heavyweight for the device. Also, it is risky as mobile devices often contain much sensitive data that may be leaked or attacked via malicious code. People can rent mobile devices when they travel to foreign countries or attend conferences; they can also use “second devices” that serve as their Enclaves (*e.g.*, their previously used phones without data plans). Enclave is similar to a sandbox as Enclave aims to protect
the master device from attacks. Unlike a sandbox, Enclave also aims to promote unobtrusive communication with the electronic world.

The following usage scenarios illustrate typical Enclave use:

- **Tourists in a foreign country**: Suppose tourists travel to a foreign country eager to explore local attractions, eat at local restaurants, and so on. These fixed sites broadcast location-specific information such as the area’s history and advertisements to passersby. Tourists view attractions and eat at restaurants carrying their mobile devices, which receive this information and store it. When they return to their hotels, they copy some received information to their personal phones or laptops. The mobile devices act as their Enclaves and their personal phones or laptops act as master devices.

- **Conference attendees**: Suppose academics attend a conference eager to meet new people and explore the surroundings. As they do so, their mobile devices broadcast their names and research work to people nearby and they receive such information from others. Interested in certain people’s information, they connect their personal phones or laptops to their mobile devices and copy information from the latter to the former. Here, the mobile devices act as Enclaves and their personal phones or laptops act as master devices.

We implement our Enclave system on commercial off-the-shelf (COTS) mobile devices. To enable Enclave’s functionality, we propose the following two key supporting technologies that address the two challenges mentioned above:

- **NameCast**: NameCast uses wireless device names to unobtrusively transmit short messages without tedious connection establishment. As its name suggests,
NameCast uses wireless names that are exchanged freely. For example, simple scans provide the service set identifiers (SSIDs) of nearby APs or the names of nearby Bluetooth devices. It effectively “tunnels” through electronic barriers using erasure coding for reliable communication. NameCast enables information dissemination via extensive peer-to-peer sharing.

– **Picture Communication (PicComm):** PicComm is a type of communication based on the transfer of visual images. It is for secure delivery of collected information on the Enclave to the master device. The basic idea is that taking pictures of images is generally safe from attack by malicious code. We encode textual information as visual images on the Enclave. The master device takes a picture of the Enclave’s screen and parses the textual content there via optical character recognition (OCR). This channel’s characteristics help hinder information leakage as potential snoops need to be physically close to the devices. We incorporate a feedback channel that automatically refreshes the Enclave’s textual content once the master device parses it successfully.

With NameCast, we transmit the feedback message via the Wi-Fi SSID, which holds up to 32 bytes of information. However, the master device needs to disclose its Wi-Fi MAC address, which may concern some users. Thus, we propose a second option: sound. Using sound, PicComm can achieve a higher degree of security without disclosing any electronic information. The drawback is the low channel capacity: only 1 or 2 bits per feedback message. PicComm divides the Enclave’s screen into pieces and dynamically updates the pieces’ “resolutions” based on OCR performance. We design a novel OCR hash function that provides resolution feedback to the master device.
Figure 3.2: Enclave system architecture and supporting technologies.

We implement Enclave with NameCast and PicComm on COTS smartphones running Android 2.3. Our experimental evaluation shows Enclave’s potential for reliable, secure, and unobtrusive electronic communication.

To our knowledge, Enclave is the first system that interposes communication between a smartphone and the electronic world using an intermediary device for the purpose of promoting unobtrusive and secure communication.

3.2 Design Overview

This section describes Enclave’s system architecture that Figure 3.2 shows. In general, Enclave has two interfaces and several parts inside as follows:

- **Interface with the electronic world:** Enclave uses NameCast as the interface with the electronic world to unobtrusively collect information from it and to send information to it on the master device’s behalf. NameCast is an option for unobtrusive communication in physical proximity. Users may choose other wireless interfacing methods such as direct connection with APs.

- **Interface with the master device:** Enclave uses the PicComm interface with the master device to securely transmit useful requested data to it.
– **Components inside Enclave**: Enclave contains several components. First, it uses the security filter to check the security of collected information from the electronic world. This component performs lightweight filtering of the received data. All received data are stored in the database. Before Enclave sends the data to the master device, the user policy module further checks the data for security breaches. Also, Enclave has a reset module that can reset it to factory conditions, purging the system of malicious data and security threats.

The components inside Enclave are lightweight as the master often resets the enclave device. We detail the two key supporting technologies, NameCast and PicComm, which enable Enclave’s unobtrusive and secure interfaces to the electronic world and the master device, respectively.

### 3.2.1 NameCast: Enclave-to-Outside Communication

We describe Enclave’s unobtrusive communication between the enclave device and external information sources. We use NameCast to achieve such communication. We describe NameCast’s design rationale and present NameCast’s high-level framework for rapid unobtrusive communication.

**Design Rationale**

NameCast aims to tunnel through the electronic barrier by implementing unobtrusive reliable communication of textual information among mobile devices. Here, *unobtrusive* means that such devices should rapidly discover each other’s presence electronically once they enter communication range without user interruption. Device users should not have to perform manual connection establishment or network configuration in order to find nearby devices. The discovery process should be rapid
and autonomous without requiring many scans to find nearby devices. NameCast’s communication should work with COTS mobile devices with Bluetooth and Wi-Fi requiring minimal configuration.

**Challenges**

To realize unobtrusive communication of textual information using Bluetooth and Wi-Fi, NameCast needs to overcome limitations of their discovery processes. We illustrate this by measuring the time incurred by such processes on COTS mobile devices. Specifically, we measure Bluetooth and Wi-Fi discovery times on two HTC Touch Diamond2s running Windows Mobile 6.1 and two Nexus S phones running Android 2.3. One Diamond2 phone discovers another Diamond2 phone and one Nexus phone discovers another Nexus phone using both Bluetooth and ad hoc Wi-Fi. Table 3.1 shows the results. We find it takes 17–18 seconds and 1.0–1.2 seconds for a Nexus phone to discover a Diamond2 using Bluetooth and Wi-Fi, respectively. Clearly, Bluetooth and Wi-Fi’s discovery processes need improvement in order to achieve unobtrusive communication.

<table>
<thead>
<tr>
<th></th>
<th>Windows Mobile 6.1</th>
<th>Android 2.3</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bluetooth (seconds)</strong></td>
<td>18–22</td>
<td>10–20</td>
</tr>
<tr>
<td><strong>Wi-Fi (seconds)</strong></td>
<td>1.0–1.1</td>
<td>1.1–1.2</td>
</tr>
</tbody>
</table>

Table 3.1: Average Bluetooth and Wi-Fi discovery times.

**Main Ideas**

At its heart, NameCast leverages the strengths of Bluetooth and Wi-Fi’s discovery processes so that they can help each other. Bluetooth can disseminate much more
information at once than Wi-Fi can (248 bytes vs. 32 bytes), but Bluetooth’s discovery process takes much longer than Wi-Fi’s (10.24s vs 1–2s). We exploit Wi-Fi’s fast discovery time to guide Bluetooth’s device discovery process. For this purpose, we embed information about the current state of a device’s Bluetooth discovery process in its Wi-Fi SSID. This information is “forwarded” to nearby devices to speed up their discovery processes. Bluetooth can then discover nearby devices more quickly than if had not received such information.

We observe that Bluetooth device names and Wi-Fi SSIDs can transmit (short) messages without tedious network configuration or user intervention. In NameCast, we leverage this capability using the following two techniques:

- **Using Wi-Fi to control Bluetooth:** We provide discovered Bluetooth device identifiers (IDs) in Wi-Fi SSIDs, which control Bluetooth device discovery. If a device discovers an SSID, it can compare the set of Bluetooth device IDs therein to its set of discovered Bluetooth devices. If there are any matching Bluetooth device IDs, the first device bypasses the lengthy inquiry process and directly obtains the matching Bluetooth devices’ names.

- **Piggybacking message dissemination atop device discovery:** We leverage the message-carrying capability to piggyback message delivery on Bluetooth and Wi-Fi discovery processes. Specifically, for a given device, we place a message and all discovered Bluetooth device IDs in this device’s Bluetooth name. We also create an ad hoc network whose SSID contains (parts of) its Wi-Fi MAC address and all discovered Bluetooth IDs. Once a second device obtains the first device’s name via Bluetooth or Wi-Fi discovery, it also receives that device’s set of all discovered Bluetooth devices. We do not include each device’s
set of SSIDs in its Bluetooth device name or Wi-Fi SSID since Wi-Fi has greater range and faster discovery time than Bluetooth.

To achieve unobtrusive communication, NameCast forwards messages among mobile devices without connection establishment. We first describe a simple case of forwarding without reliability considerations. Next, we describe how we increase the reliability of forwarding via erasure codes.

**Basic Forwarding**

Forwarding messages to mobile devices that have not yet received these messages plays a key role in NameCast. Consider Figure 3.3, which shows a simple network topology. All devices in Figure 3.3 can discover each other via Wi-Fi, but devices A and C cannot discover each other via Bluetooth. All devices transmit control information via Wi-Fi “control frames” and their messages in Bluetooth device names, which can store more information than SSIDs. SSIDs can be discovered within a couple seconds, whereas Bluetooth name discovery takes at least ten seconds. Such control frames contain a device’s message (its Bluetooth MAC address) along with others’ messages. In the topology, B determines that A has not received C’s Bluetooth address and forwards it, along with B’s address, in B’s control frame. Thus, A discovers C’s Bluetooth MAC address and pages it before Bluetooth inquiry is finished. C’s message is piggybacked to A during the process. Similarly, B forwards A’s MAC address to C when it observes C has not received this address.

This forwarding mechanism enables mobile devices to “see” each other’s presence after they “meet.” This surmounts the electronic barriers posed by conventional wireless protocols and achieves unobtrusive communication.
Figure 3.3: Motivating topology for NameCast. Bluetooth inquiry and Wi-Fi scanning begin at time 0 seconds. At 2 seconds, the nodes discover each other's SSIDs. Inquiry finishes at 10.24 seconds; paging 1–2 seconds thereafter. Paged device names are shown in white on a black background. The Wi-Fi control frame provides a list of each node’s discovered Bluetooth addresses; those with double lines are to be paged.

Forwarding with Reliability Enhancements

This NameCast forwarding mechanism is well suited to small-scale short message transmission without reliability. To achieve this at large scale, we enhance our mechanism to incorporate erasure coding. Our motivation is as follows. If many devices are transmitting messages to each other without connections, transmission errors are likely to occur: receivers may miss messages, messages may be garbled, etc. We need to enhance our forwarding mechanism so it can recover from these errors without substantial overhead. We use fountain codes to do so. In the following, we explain our rationale and coding approach.

Fountain codes are sparse graph codes for channels with erasures [100]. Such codes divide long messages into many small pieces and disseminate these pieces to receivers.
Once receivers receive enough pieces, they can reconstruct the messages regardless of the order of the pieces’ delivery. Fountain codes have many desirable properties: they are simple to implement (typically using XOR operation), rateless (meaning a coding rate need not be specified in advance), and largely obviate a feedback channel. Receivers need not inform the sender about missing message pieces, which is tedious.

We use fountain codes at the MAC layer as ACKs are not present there. Acknowledging each successful channel access imposes extra overhead on protocols. Fountain codes transmit encoded messages without requiring feedback. Receivers who have not received enough information need not feed this information to the message sender; as the sender keeps sending, they simply wait until they have received enough information.

Our fountain coding approach uses the following concepts:

- **Bluetooth frame**: A Bluetooth frame is a message consisting of all nearby devices’ Bluetooth names. Each device constructs its own frame and disseminates it to all nearby devices. Each frame has a unique two-digit ID.

- **Encoded chunk**: We encode the Bluetooth frame using fountain codes and split it into equal-size chunks, each of which fits in a Bluetooth device name. When disseminating its Bluetooth frame, each device publishes the frame’s (coded) chunks in its Bluetooth name.

Devices always transmit and receive chunks. If a receiver does not receive a transmitted chunk, it gathers more chunks until it reconstructs the Bluetooth frame. This enables error recovery with low overhead, achieving reliable communication.
Figure 3.4: Wi-Fi control frame and Bluetooth name for NameCast. The ‘/’ denotes erasure coding. Here, 01 is the Bluetooth frame ID and 3 chunks are needed to decode the frame.

Protocol Details  Now we explain the details of our NameCast protocol. First, we describe the Wi-Fi control frame data structure. Next, we describe our erasure coding procedure. Finally, we describe how the NameCast protocol operates.

Figure 3.4 illustrates the high-level Wi-Fi control frame structure. The first field of the control frame is the last two bytes of the device’s Bluetooth address. These bytes link the control frame with the message published in the device’s Bluetooth name: they are used to determine a complete Bluetooth address when paging. The following fields of the control frame show which messages have already been received and which ones have not. The +/- character denotes whether a message has been received. The control frame serves two purposes: (1) Informing neighbors of what message the local device is currently publishing; and (2) Informing neighbors what messages the local device has received and what ones it has not. We expect that neighbors will publish messages with ‘-’ flags for the local device.

Our coding approach works as follows. We construct a Bluetooth frame of length $m$, which we encode using LT codes [101]. The encoded frame is split into $n$ chunks
\( c_1, \ldots, c_n \), where each \( c_i \) has length \( \ell \). We construct a full rank 0-1 coding matrix \( E \) with \((1 + \epsilon) \cdot n \) rows and \( n \) columns and XOR chunks \( i \) and \( j \) if element \( e_{i,j} = 1 \).

In each round of our multiple update NameCast protocol, we place each matrix row and a chunk in the Bluetooth name. Once receivers receive \((1 + \epsilon) \cdot n\) chunks (and rows), they can decode the Bluetooth frame. Let \( T_u \) denote the update interval at the sender and let \( T \) be a random variable describing the message decoding time. Then \( E[T] = (1 + \epsilon) \cdot n \cdot T_u \). We add a header to the Bluetooth name that denotes the use of fountain codes and includes the frame ID and number of chunks needed to decode the frame.

The NameCast protocol is described in Algorithm 3.1. It has two main threads of execution: 1. Wi-Fi; and 2. Bluetooth. The Wi-Fi thread periodically scans for control frames. If a new Bluetooth address ID is found in an SSID, this ID is added to the \texttt{CandidAddrSet}. (This ID comprises the last two bytes of the device's true Bluetooth MAC address; the ID is not the whole address.) This thread also finds the “most desirable” message ID that appears most frequently in the control frames and has not been received. The Bluetooth thread alternates between Bluetooth inquiry/paging and publishing (long) messages in the Bluetooth name. Due to the long inquiry time, we inquire for addresses every other iteration. If \texttt{candidAddrSet} is not empty, we page the Bluetooth addresses corresponding to the IDs there; otherwise, we randomly choose addresses to page. Then we split the message into chunks (if necessary) and publish each chunk in turn. We need to publish the current chunk but also receive new chunks from neighbors. Thus, we alternate between publishing this chunk and paging Bluetooth neighbors for new chunks. Upon successful paging, we mark a Bluetooth device’s IDs (the last two bytes of its address) with a ‘+’ in the control frame;
Algorithm 3.1 NameCast protocol

1: Main Procedure:
2: \( BT_{name} \leftarrow \); \( Wi - Fi_{SSID} \leftarrow \) ctrl frame; make discoverable;
3: Start Wi-Fi thread and Bluetooth thread;
4:
5: Wi-Fi Thread: {
6: Set up ad hoc Wi-Fi network with SSID \( CF \); / /ctrl frame
7: while (user does not terminate) do
8: Do active scan, store received Wi-Fi SSIDs (\( CF_i \)) in \( CFSet \);
9: for each \( CF_i \) in \( CFSet \) do
10: for each addr\(_j\) in \( CF_i \) do
11: if addr\(_j\) \( \notin \) BTAddrSet \( \&\& \) addr\(_j\) \( \notin \) RecvMsgSet then
12: CandidateAddrSet \( \leftarrow \) CandidateAddrSet \( \cup \{addr_j\}\);
13: end if
14: end for
15: end for
16: Select \( MsgID \in \) RecvMsgSet appearing most frequently in \( CFSet \) with flag ‘-’;
17: Update Wi-Fi SSID and Bluetooth name with \( MsgID \);
18: end while }
19:
20: Bluetooth Thread: {
21: iteration \( \leftarrow \) 0; codedBTAddrs \( \leftarrow \) \( \varnothing \);
22: while (user does not terminate) do
23: Inquire nearby Bluetooth addresses, store in BTAddrSet;
24: if CandidateAddrSet = \( \varnothing \) then
25: for each of \( N_r \) randomly chosen addresses addr\(_i\) do
26: PerformPaging(addr\(_i\))
27: end for
28: else
29: for each discovered address in BTAddrSet do
30: PerformPaging(BTAddr\(_i\))
31: end for
32: end if
33: Construct BT frame, encode it, and split into chunks;
34: for each chunk do
35: Place chunk and encoding row in BT name;
36: Page each addr\(_j\) \( \in \) codedBTAddrs
37: end for
38: end while }
39:
40: Subprocedure PerformPaging(addr\(_i\)):
41: Page addr\(_i\), store message in message
42: if paging succeeds and isValidChunk() then
43: Decode chunk (frame)
44: codedBTAddrs \( \leftarrow \) codedBTAddrs \( \cup \{addr_i\}\)
45: Update Wi-Fi SSID accordingly;
46: end if
upon paging failure, we mark the ID with a ‘-’. We maintain $\text{codedBTAddrs}$, the set of currently discovered Bluetooth addresses whose corresponding names are encoded chunks. When we discover a new such Bluetooth address, we add it to $\text{codedBTAddrs}$. In each iteration of the Bluetooth thread, we generate a Bluetooth frame, encode it, and split the encoded frame into chunks. We alternately publish each chunk and page each address in $\text{codedBTAddr}$ to retrieve newly published chunks.

**Remarks** NameCast can only change Bluetooth device names and Wi-Fi SSIDs names programmatically on certain mobile devices such as Android and Windows Mobile 6.x smartphones. At this time, iOS, BlackBerry OS, and Windows Phone do not allow users to *programmatically* change Bluetooth device names and SSIDs. However, these devices can still interact with devices running NameCast without installing the system. Users of such devices can disseminate information by manually changing their devices’ Bluetooth names or Wi-Fi SSIDs.

NameCast implicitly assumes that users are willing to share some information electronically with people in physical proximity. This has potential privacy implications as attackers can fingerprint the master or enclave device via their Bluetooth device names and Wi-Fi SSIDs [102]. We point out that NameCast can be used only for communication between the enclave device and external data sources, which limits such fingerprinting to the enclave device. In such cases, the master device would like to communicate with the enclave device without leaving an electronic “trace.” PicComm provides such functionality, which we discuss in Section 3.2.2. Additionally, incorporating strong privacy and security controls with NameCast forms an important part of our future work. We consider leveraging further coding techniques and
power control to selectively send the NameCast information to a designated group of people.

3.2.2 PicComm: Enclave-to-Master Communication

Enclave can realize secure communication between the master and enclave devices via picture communication (PicComm), a type of visual communication based on taking photos without wireless communications. We detail it in this section.

Design Rationale

We need a secure communication channel between the master device and the enclave device in order to transmit useful information to the master. This is hard to achieve. Such a channel needs to be protected from eavesdropping attackers, but the master device cannot totally trust the enclave device as it is exposed to the outside world where malicious code or indecent content may reside. We argue that wireless-connection-based communication may not be secure enough due to security problems [103] or vulnerabilities [104]. Further, some users may be concerned about disclosing their electronic identities (e.g., MAC addresses) using Wi-Fi or Bluetooth.

Main Ideas

We propose PicComm in Enclave to establish such secure communication. In PicComm, the master device takes pictures of the enclave device’s screen, which displays textual messages, and recognizes their contents using optical character recognition (OCR). In this way, information transmission from the enclave to the master is very secure. Besides security, PicComm needs to achieve high throughput. However, the screen size of a device is limited and OCR results are not 100% accurate. Intuitively, if
the message on the screen has larger font size and letterspacing, OCR performs better, but the message volume on-screen decreases. Thus, we propose a dynamic resolution adjustment mechanism for PicComm to find a good tradeoff between message volume and resolution (i.e., the font size and letterspacing). In dynamic resolution adjustment, the master device sends feedback to the enclave (an ACK or NAK) and hash values indicating the OCR error degree. Based on this information, the enclave device dynamically adjusts its resolution of the message on-screen. We consider two options for the feedback channel: wireless name communication and sound. Wireless name communication is part of NameCast; we use the Wi-Fi SSID to transmit the feedback message (up to 32 bytes). But the master device has to disclose its Wi-Fi MAC address, which may concern some users. Thus, we propose the second option: sound. With sound, PicComm can achieve greater security without disclosing any electronic information. The drawback is low channel capacity (1 or 2 bits per feedback message).

In the following, we discuss the design of our dynamic resolution adjustment mechanism.

**Dynamic Resolution Adjustment** A naïve approach to resolution adjustment is to uniformly change the whole screen. If the enclave device receives an NAK, the resolution setting is enlarged by one (i.e., the font size increases by one point). If the device receives an ACK, the resolution setting will be decreased by one point. However, such an approach is inefficient. We should not enlarge resolution settings for error-free parts of the screen, which yields less space for the total message. Thus, we introduce the concept of a “block,” an isolated area in the screen with its own resolution settings. We perform resolution adjustment for each block.
To check the correctness of the OCR result, we place a cyclic redundancy check (CRC) code at the bottom of the enclave device’s screen. If the CRC check passes, the master simply sends an ACK. Otherwise, a NAK is insufficient for the enclave device to identify the part of the screen containing errors and determine their severity. Thus, we design a hash function that helps evaluate the error degree between the original message and its OCR result. Traditional cryptographic hashing functions like MD5 and SHA-1 do not apply, since they only specify that differences occur, not the difference amount. Moreover, limited feedback channels such as sound do not permit long hash values (e.g., 128 bits for MD5).

We design OCRHash, a hashing function that maps a variable-length message to a small range of values. OCRHash has the following property: the greater the difference between two OCRHash values, the more errors probably occurred. The intuition behind OCRHash is that more probable errors contribute more to the difference amount of OCRHash values. For OCR, most common errors come from interchanged characters [105]. Based on a training dataset, we can determine each character $C$ and all possible interchanged characters $E_i$. Then the probability of recognizing $E_i$ as $C$ is calculated as [105]:

$$P(C|E_i) = \frac{P(C)P(E_i|C)}{\sum_{k=1}^{n} P(C_k)P(E_i|C_k)}. \quad (3.1)$$

We divide all characters into several groups, each of which has a unique group value such as 0 and 1. If a pair of characters $i$ and $j$ has a high probability of interchange (i.e., $P(i|j)$ or $P(j|i)$ is large), $i$ and $j$ will be placed into different groups whose values differ greatly. If the pair’s probability of interchange is low, $i$ and $j$ will be placed into two groups with a small group value difference (perhaps the same group). Consider Figure 3.5 as an example. Characters “m” and “n” are interchanged with
high probability, so their group values differ greatly. But characters “A” and “B” are interchanged with low probability, so their group values are the same. Next, we sum all characters based on their group values and proportionally downscale the sum to the range of values:

$$\text{hash}((m_0, \cdots, m_{l-1})) = \left\lceil \frac{\sum_{i=0}^{l-1} \text{group}(m_i)}{l \times (n-1) \times (2^b - 1)} \right\rceil,$$

(3.2)

where $l$, $n$, and $b$ are the message length, number of groups, and number of bits in the feedback channel, respectively.

Protocol Details

The workflow of PicComm is as follows. First, the enclave device displays a message, denoted $CurMsg$, on the screen. The master device takes pictures of the enclave device’s screen and parses the content via OCR as $RecvMsg$. The master device checks if $RecvMsg$’s CRC matches $RecvMsg$’s content. If yes, the master sends an ACK and null OCRHash values for each block via sound or NameCast. Otherwise, the master calculates the OCRHash values based on each block in $RecvMsg$ and sends a NAK with these OCRHash values. After receiving feedback messages from the master device, the enclave device parses them into two parts: the first bit indicating ACK or NAK and the following bits indicating OCRHash values. If the first
bit is an ACK, the enclave device fetches the next message and displays it on the screen. If this bit is a NAK, the enclave device calculates the differences between the received OCRHash value and the original content’s OCRHash value for each block. Each block’s resolution is changed based on its difference value. If this value is zero, we enlarge the resolution by one step (e.g., one font point size); larger values yield correspondingly larger resolution changes. Algorithm 3.2 shows this protocol.

**Remarks** Ideally, the granularity of block partitioning is as fine as possible, which helps identify the erroneous parts of the screen without affecting the other correct parts. However, the feedback channel from the master to the enclave has only a few bits. As a result, we only apply several blocks in PicComm. Thus, how to partition a screen into blocks (i.e., the location and shape of a block), poses an interesting problem related to the OCR segmentation process. Currently, we use regular blocks that cover one or multiple lines as our OCR tool performs better with lines of similar settings. Irregular block partitioning is part of our future work.

Like parity bits, our current group value design cannot handle cases of interchanged two-character errors. For example, if “man” is recognized as “nam”, the two OCRHash values will be the same. Empirically, we notice that the interchange error probability in OCR is asymmetric. Usually, unidirectional high-probability error is common; thus, interchanged two-character errors rarely occur. We adopt the conservative policy where resolution settings change even if the OCRHash value difference is zero in order to correct such unexpected errors.
Algorithm 3.2 PicComm protocol

1: Enclave Procedure:
2: \hspace{.5cm} CurMsg ← first message in \{message\};
3: \hspace{.5cm} block[0,\ldots,k] ← split CurMsg into k blocks by each block’s setting;
4: \hspace{.5cm} ControlBlock ← CRC(CurMsg);
5: \hspace{.5cm} while \{message\} ≠ ∅ do
6: \hspace{1cm} OnRecvFeedback()
7: \hspace{1cm} ACK,MasterVal[0,\ldots,k] ← information from feedback;
8: \hspace{1cm} if ACK then
9: \hspace{1.5cm} \{message\} ← \{message\} \ {CurMsg};
10: \hspace{1.5cm} CurMsg ← first message from \{message\};
11: \hspace{1.5cm} block[0,\ldots,k] ← split CurMsg by each block’s setting;
12: \hspace{1.5cm} EnclaveVal[0,\ldots,k] ← OCRHash(block[0,\ldots,k]);
13: \hspace{1cm} else if NAK then
14: \hspace{1.5cm} for each block_i do
15: \hspace{2cm} Diff_i ← |EnclaveVal_i - MasterVal_i|;
16: \hspace{2cm} fontSize ← Diff_i + 1; make bold; add 1 kern space;
17: \hspace{1.5cm} end for
18: \hspace{1cm} end if
19: \hspace{1cm} end while
20: 
21: Master Procedure:
22: \hspace{.5cm} while user does not terminate do
23: \hspace{1cm} Take picture of enclave device’s screen;
24: \hspace{1cm} RecevMsg ← perform OCR on picture;
25: \hspace{1cm} if CRC check passes then
26: \hspace{1.5cm} ACK ← 1; MasterVal[0,\ldots,k] ← 0;
27: \hspace{1cm} else
28: \hspace{1.5cm} ACK ← 1;
29: \hspace{1.5cm} block[0,\ldots,k] ← split RecevMsg by delimiter;
30: \hspace{1.5cm} ACK ← 1; MasterVal[0,\ldots,k] ← 0;
31: \hspace{1.5cm} Send feedback \{ACK,block[0,\ldots,k]\} by NameCast or sound;
32: \hspace{1cm} end if
33: \hspace{1cm} end while
34: 
35: Subprocedure:
36: \hspace{.5cm} OCRHash(block):
37: \hspace{1cm} WeightSum ← \sum_{ch\in block} groupValueof(ch);
38: \hspace{1cm} ScaledVal ← downscale WeightSum to \langle 0,\ldots,2^m - 1 \rangle; // m is the number of bits for the feedback channel (NameCast or sound)
39: \hspace{.5cm} return ScaledVal;
3.3 Implementation and Evaluation

We implement Enclave on Google Nexus S smartphones running Android 2.3.3. These phones have both Bluetooth and Wi-Fi functionality. We perform factory reset using standard Android techniques [106].

3.3.1 Implementation

We implement NameCast on the Nexus Ss in Java; our Android package is 52.7 KB. Our system uses the BlueZ binary hcitool [107] for Bluetooth inquiry and paging. We establish an ad hoc wireless network for each smartphone to publish Wi-Fi SSIDs and scan for nearby networks. We implement ad hoc networking using Wireless Tools for Linux [108] and wpa_supplicant [109]. We implement erasure coding using LT codes with $\epsilon = 0.5$, message length $m = 20$, and chunk length $n = 60$.

We implement a similar prototype system running a naïve version of the NameCast protocol that publishes static Bluetooth names and Wi-Fi SSIDs. It has no control frame information or message delivery piggybacking. We use this naïve protocol as a control in the experiments described in Section 3.3.2.

We implement PicComm on the Nexus Ss. The enclave device displays textual content of which the master device takes pictures and processes via OCR. Specifically, the enclave displays text on the screen in a $2 \times 2$ block layout (four blocks in total). In our experiments, the enclave displays text from *Alice’s Adventures in Wonderland* [110]. The master device processes the text via OCR Tool for Android [111], which uses the Tesseract OCR engine [112].
### 3.3.2 Evaluation

We test our naïve and NameCast systems in a university building with 6 Nexus Ss and 14 other mobile devices. We use 4 iPod touches, 3 laptops, and 7 mobile phones. For all experiments, we run each system 10 times.

We evaluate our naïve and NameCast systems using the following metrics:

- **Coverage.** We measure how many devices we discover while running both systems; and

- **Power consumption.** We measure Nexus S power consumption during each run of the naïve and NameCast systems.

Figure 3.6 shows the average number of received messages for the naïve and NameCast systems. For the majority of devices, NameCast discovered more devices in the network than the naïve system. This occurred because NameCast forwarded Bluetooth messages among many hops; the naïve system could only discover devices one hop away. These results are encouraging: they show NameCast can generally discover more Bluetooth devices in a network than a naïve system.

Figure 3.7 shows the average time required for devices to receive messages with the naïve system and NameCast. After three minutes, the number of devices the naïve system discovered levels off, whereas the number of devices NameCast discovers increases over time. The curves cross at about four minutes. This is unsurprising, since NameCast forwards Bluetooth messages among many hops and the naïve system can only discover messages one hop away. We see that, with enough time, NameCast can discover more nodes in the network than the naïve system.
Our results also illustrate NameCast’s ability to achieve unobtrusive communication on a wide range of mobile devices. NameCast users need not install our software on their devices to use our protocol; they set their Bluetooth names following NameCast’s format to communicate with devices running the software.

We test PicComm using Nexus Ss. Two phones, an enclave and a master, are placed face-to-face. The following three different mechanisms are evaluated:

- **Naïve version without any feedback.** The enclave device only uses a default setting and never change its resolution settings.

- **PicComm with 1-bit feedback (via sound or NameCast).** The master device only provides 1 bit of feedback (*i.e.*, ACK or NAK) to the enclave device. The enclave device changes the resolution setting uniformly.

- **PicComm with multiple-bit feedback via NameCast.** We split the enclave device’s screen into 4 blocks and the OCRHash value for each block is
0–3, which takes 2 bits. 1 bit is reserved for a CRC check result. Totally, 9 bits of (NameCast) feedback are used.

We evaluate the metrics of throughput and power consumption. Figure 3.8 and Table 3.2 illustrate the results. Figure 3.8 shows that PicComm performs better with more feedback. This demonstrates that block-based dynamic resolution adjustment helps improve PicComm’s throughput. PicComm’s throughput with multiple-bit feedback is almost twice of that of 1-bit PicComm. We see that PicComm’s throughput varies with different contents but is generally stable. The final throughput is \( \sim 2 \) bytes/second, enough to transmit short messages between the master and enclave. The na"ive version does not work well; only a few messages transmit successfully. Table 3.2 shows average power consumption rates of different mechanisms within a 30-minute test period. PicComm’s power consumption rate is fairly low when only sound is used for feedback. With PicComm, Wi-Fi and the camera consume the most power. With multiple-bit feedback, PicComm’s power consumption rate is lower with 1-bit feedback. We explain this as follows: fewer rounds of feedback are performed for PicComm with multiple-bit feedback, which encounters fewer OCR failures.
3.4 Summary

This chapter proposed Enclave, a delegate wireless device that helps people’s smartphones communicate unobtrusively. We realized Enclave using two key supporting technologies: NameCast and PicComm. We implemented Enclave on Nexus S smartphones. Our experimental evaluation showed Enclave’s potential for smartphone data protection and unobtrusive and secure communication.
Chapter 4: Pervasive Mobile Device Communications
Measurement

This chapter discusses measurement of pervasive mobile device communications via a new dataset of students’ wireless LAN (WLAN) logs collected at The Ohio State University (OSU). Our dataset includes students’ demographic categories (i.e., their birthdays, genders, and majors) that are made available after anonymization. To the best of our knowledge, this dataset is the largest regarding the number of non-laptop mobile devices and access points (APs) in it. We study students’ mobility patterns for different demographic categories. Finally, we leverage the dataset’s WLAN site survey information for 73 buildings on OSU’s campus in order to perform indoor localization. We employ multilateration regarding devices’ received signal strength indicators (RSSI) for APs in these buildings. We design and implement a mobile application (app) on commercial off-the-shelf devices running Android 5.1. In general, our app localizes users with room-level accuracy.
4.1 Overview

Mobile devices such as smartphones and tablets are ubiquitous. Since people carry mobile devices everywhere, device mobility closely approximates human mobility. Recently, large-scale mobile phone log data from cellular networks have become available [13]. Researchers have also used GPS-equipped automobiles to investigate vehicle mobility [27]. Human mobility models have been proposed based on these real-world datasets [15,20,27–29].

Understanding human mobility is critical for various applications such as epidemic modeling, urban planning, and resource management for mobile communications [10,11]. To an observer, human mobility may appear random and unpredictable. However, in a seminal paper, Song et al. [30] used mobility entropy to characterize the predictability of human mobility. A higher entropy for a group of people indicates that this group’s mobility is less predictable on average. Similarly, the entropy of a person’s mobility can be interpreted as the amount of irregularity in her mobility. In human mobility studies, a body of work has used this metric [31–33]. In this chapter, we use mobility entropy to investigate mobility differences across demographic categories by exploiting a new comprehensive wireless local area network (WLAN) dataset collected at OSU.

The WLAN data are collected over 139 days from Tuesday, January 13, 2015, to Sunday, May 31, 2015. OSU’s campus size exceeds 1,500 acres (∼6.07 kilometers$^2$) and has an extensive Wi-Fi network with over 8,000 APs; OSU offers several hundred majors. Over 5,000 students are included in this dataset, which represents a random sample of the overall student body. The dataset consists of the (dis)association records
of all students’ devices that connect to osuwireless, OSU’s encrypted WLAN Service Set Identifier (SSID).

This is the first study of the predictability of human mobility using WLAN log data with demographic information at a large public university. We investigate the underlying mobility entropy differences among groups of students based on their ages, academic majors, and genders. By varying the sampled student size and observation length, we verify the validity of our discoveries. We conclude that the observation length and the number of students in our dataset are sufficient to infer meaningful results. We compare students’ overall long-term mobility entropy across the entire observation period and short-term mobility entropy (such as daily and weekly entropy).

Main Discoveries Using mobility entropy as the primary metric, we make the following surprising discoveries contrary to common-sense beliefs.

– **Common-sense view:** As a group with similar ages and backgrounds, it is often assumed that students’ mobility follows a similar distribution with respect to age. One or two years of age variation should not yield major changes in students’ mobility.

We observe that the overall long-term entropy greatly varies by age for 19–21 year-olds (the group of traditional college students). The distribution of mobility entropy noticeably increases as students’ ages increase. The mean entropies for 19-year-olds and 20-year-olds are 0.89 and 1.16, respectively, which represents a 30% increase. Similarly, the entropy of 21-year-olds is 23% higher than that of 20-year-olds. The rate of entropy increase levels off after age 21. The shift in
mobility entropy across age groups was not previously reported in the literature such as [30,32].

– Common-sense view: The rationale why students choose a major is believed to be largely due to academics. The impact of academic majors on mobility should be minimal.

We observe that the rate of change across age groups within each major is not uniform. The change in mobility entropy from 19- to 22-year-old age groups is markedly different for different majors. For example, the entropic change from 19- to 20-year-old engineering students is much less compared with that of business majors. Health-related majors and engineering majors have lower mobility entropies than students from other majors. When student mobility entropies are compared across majors on a daily basis, undecided majors have lower entropies than every other major.

– Common-sense view: The distribution of mobility for college students is expected to be similar to the distribution of the general population. Previous studies have shown that mobility distributions of people follow heavy-tail distributions such as Pareto or Weibull distributions [15].

Surprisingly, we observe that the overall long-term entropy of students follows a bimodal distribution. The two modes of observed entropies indicate that there are two groups of students, each with its distinctive mobility. Figure 4.1 shows the observed bimodal distribution.

In addition, we verify that there is no significant difference between males and females on their long-term entropies. This finding confirms the results from Song et
Figure 4.1: Observed long-term mobility entropy distribution over a 139-day period.

...
overall mobility entropy indicates that a homogeneous assumption of mobility parameterizations should be revisited. In epidemic studies, if demographic information is known, then the speed of disease spread should consider mobility predictability to reflect a more refined model. In the area of resource management, considering mobility entropy differences in demographics can yield more accurate estimates of resource distribution rates; and

- **Insights for academic administrations:** Our results can also help academic administrations regarding facilities planning and student retention. For example, our study shows that students with health-related and engineering majors tend to have low mobility entropy. Thus, classrooms for those majors might be intentionally allocated among different buildings considering their predictability differences. For retention, university administrations can design better education policies and study plans that incorporate mobility differences across age groups focusing on the 19–22-year-old group who are most likely to be “traditional” undergraduate college students.

In addition, our dataset contains detailed WLAN site survey information regarding AP deployments in 73 buildings on the OSU campus. We leverage this information to build a mobile application (app) that localizes users via multilateration using mobile devices’ received signal strength indicators (RSSIs). The app displays the APs and users’ estimated locations on indoor floor plan imagery as well as Mapbox maps [34]. We implement the app on commercial off-the-shelf (COTS) mobile devices running Android 5.1. We find that the app localizes users with room-level accuracy.
4.2 Dataset

The dataset includes the (dis)association of mobile devices with respect to APs on OSU’s academic campus in Columbus, OH, USA. Based on their authentication credentials, we obtain students’ birthdays, majors, and genders from the university’s student information system. We exclude students who use the unsecured WLAN from the dataset as their demographic information cannot be verified. For privacy reasons, we encrypt student credentials using a two-way hash function. There are 5,096 students in the dataset with 13,549 unique mobile device MAC addresses. The dataset includes (dis)association logs from 8,420 different APs across 225 buildings.

A sample log entry has the following format:

\texttt{timestamp,process,ap-name,student-id,role,MAC,SSID,result}

The fields in the log represent the event’s UNIX timestamp, the process that generated the log entry, the AP name, the encrypted student ID, the device’s role, the anonymized MAC address (with the Organizational Unit Identifier (OUI) preserved), the SSID name, and the authentication result (success or failure), respectively. Each AP is named via the building name, floor number on which the AP resides, and a unique numeric ID for the AP on this floor. For example, the AP name BB-2-4 indicates that this AP is the fourth AP on the second floor in building BB.

There are 41,006,186 log entries in the dataset, and the unit of the timestamp is seconds. Some entries do not include student IDs and AP names; we consider these log entries invalid. After removing invalid entries from the dataset, 39,100,373 log entries remain.
Our dataset has wireless data for 5,096 students with demographic information, which is 12% of the overall student population at OSU. Each student is represented via an anonymized ID associated with the student’s gender, birthday, and major. We also obtained a list of all students at the university including their demographic information. A careful examination of the distribution of students in the dataset reveals that these students are representative of the overall student population at the university. Nearly all demographic categories are within \((12 \pm 2)\%\), which limits potential student under- or over-representation. To verify that our results are significant with respect to student size, we also sample 50–80% of students from the dataset and perform the same calculations. We observe that our original findings remain when over 60% of students are sampled. To ensure the best mobility entropy measurement, we eliminate laptop computers from consideration. This requirement eliminated 720 students whose only device that appeared in the dataset is a laptop. We verify that these students are uniformly distributed across demographic categories, so this adjustment does not distort demographics. We believe that this group is likely to have set up their real mobile devices to use the university’s unsecured WLAN.

Table 4.1 shows student data. We group student majors into seven categories based on the university registrar’s descriptions of majors. The age range of students is 19–58 with over 80% of students in the 19–22-year-old groups. The number of students decreases in the “24 and older” group \((24+)\) as students’ ages increase. In this study, we treat the 24+ group as a single age category. This group of students includes senior citizens who take tuition-free college courses.
<table>
<thead>
<tr>
<th>Majors</th>
<th># of Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business</td>
<td>893</td>
</tr>
<tr>
<td>Education</td>
<td>218</td>
</tr>
<tr>
<td>Engineering</td>
<td>840</td>
</tr>
<tr>
<td>Health</td>
<td>645</td>
</tr>
<tr>
<td>Science</td>
<td>840</td>
</tr>
<tr>
<td>Social Science</td>
<td>803</td>
</tr>
<tr>
<td>Undecided</td>
<td>137</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Age</th>
<th># of Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>19</td>
<td>706</td>
</tr>
<tr>
<td>20</td>
<td>1,116</td>
</tr>
<tr>
<td>21</td>
<td>1,014</td>
</tr>
<tr>
<td>22</td>
<td>676</td>
</tr>
<tr>
<td>23</td>
<td>309</td>
</tr>
<tr>
<td>24 and older</td>
<td>125 (age 24)</td>
</tr>
<tr>
<td>24 and older</td>
<td>105 (age 25)</td>
</tr>
<tr>
<td>24 and older</td>
<td>325 (older than 25)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gender</th>
<th># of Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>Female</td>
<td>2,039</td>
</tr>
<tr>
<td>Male</td>
<td>2,335</td>
</tr>
<tr>
<td>Unknown</td>
<td>2</td>
</tr>
</tbody>
</table>

| Total          | 4,376        |

Table 4.1: Student demographic data. Total reflects the laptop reduction mentioned above.

4.2.1 Data Processing Methodology

The primary metric that we use, mobility entropy, is calculated using a person’s trajectory $T$ that is defined as

$$T = (L_1, t_1, ST_1) \rightarrow (L_2, t_2, ST_2) \rightarrow \cdots \rightarrow (L_N, t_N, ST_N), \quad t_1 < t_2 < \cdots < t_N,$$

where $L_i$ is the $i$-th location in trajectory $T$, $t_i$ is the arrival time of the person at location $L_i$, and $ST_i$ is the stay time of the person at location $L_i$.

The length of a trajectory, $N$, is the total number of locations that a person visits between times $t_1$ and $t_N$. The maximum observation length of our WLAN log is 139 days but by varying $t_1$ and $t_N$, appropriate time intervals of interest can be selected. In this dissertation, we define long-term as the entire 139-day time span and short-term as daily or weekly.
To build the trajectory of each user for entropy calculation, we need to extract the sequence of locations for all students as well their stay times from the WLAN log entries. Each log entry provides information about a user’s identity (user/MAC), the name of the AP with which the student was interacting, and a timestamp $t$. Since a student is likely to have several mobile devices (the number usually ranges from 2 to 5), each MAC address is treated independently when the trajectory is constructed. Thus each log entry can be viewed as an ordered triple $(t, \text{user/MAC}, \text{AP})$. We sort log entries based on the timestamp $t$ to ensure sequential order.

### 4.2.2 Location Granularity

Since an AP can only provide accuracy within its range, and a device’s connection to an AP does not imply that this AP is the closest one to the device, we use buildings as base units in trajectories. We believe this approximation is appropriate since this work focuses on human mobility and buildings are natural base units for outdoor mobility. Thus, all sequential connections to APs inside a building are treated as a single data point in a student’s trajectory with accumulated stay time. In the example shown in Figure 4.2, a device interacts with three APs in building $B_1$ before connecting to $AP_1$ in $B_2$ followed by two APs in $B_3$. Thus, the sequence of locations this device has visited is $B_1, B_2, B_3$. The stay time of a device with an AP is the length of the time interval between its association time and disassociation time. Most devices are not logged when they disassociate with APs. Hence, the stay time of each user inside the same building is calculated as the difference of association times between the current AP and the next AP in the log. For example, in Figure 4.2, $AP_1$’s stay
time is calculated as $t_2 - t_1$. Thus, the overall stay time of a user in building $B_i$ is $\sum_{j=4}^{k} staytime_{AP_j}$, where $AP_j$ denotes APs in the same building.

### 4.2.3 Stay Time Adjustment

We observe that users normally connect to several APs in the same building before leaving it and students’ speeds are inconsistent. The latter observation is similar to that of Kim et al. [57]: speeds calculated between consecutive buildings in the log can be large. We believe that this is due to mobile devices’ ability to connect to a remote AP even when people carrying the devices are far away. Thus, we adjust the travel time from one building to another. Figure 4.3 shows our process of stay time estimation.

![Time Compressed into Stay Time](image)

Figure 4.2: Location granularity as buildings.

![Stay time estimation for inter-building AP connections in a person’s trajectory](image)

Figure 4.3: Stay time estimation for inter-building AP connections in a person’s trajectory.
For a device, suppose \( t_1 \) is the time of the first association between this device and an AP in building \( A \), \( t_2 \) is the time of the last association between this device and an AP in building \( A \), and \( t_3 \) is the time of the first association between this device and an AP in building \( B \). We apply the following rules to compensate a building’s stay time based on the analysis of inter-building travel time.

- If the time interval from building \( A \) to building \( B \), \( t_3 - t_2 \), is smaller than the estimated travel time, we judge that a fast remote connection has been established between the mobile device and an AP in a building. There is no need to compensate building \( A \)'s stay time.

- If the time interval \( t_3 - t_2 \) exceeds the estimated travel time, we judge that the user stays in building \( A \) for a longer period before moving to building \( B \). Thus, the total stay time of building \( A \) is \((t_3 - t_1 - \text{estimated travel time})\).

We calculate the estimated walking time between two buildings using the Google Maps API [113]. We obtain the latitude and longitude of every building that appears in the log and calculate the estimated travel time between each pair of buildings with the “walking” travel mode.

There is no association record following a device’s last AP association entry at the end of the observation period. Thus, the aforementioned formulas cannot be applied to find the stay time of the last AP. Since we focus on non-laptop mobile devices’ trajectories, it is very unlikely that a mobile device simply “disappears” from the extensive campus WLAN. The most plausible explanation is that the device has left the WLAN after a very short connection. Hence, we assume that the last AP’s
connection is small but nonzero. When entropy is calculated, we assume that the last AP’s stay time is less than a predefined constant (i.e., $\Delta t$ as defined in Section 4.3.1).

After data processing, each user/MAC’s trajectory becomes a time series of buildings and their corresponding stay times.

4.3 Measurements and Results

In this section, we define mobility entropies that are used to capture student mobility patterns. Next, we present our findings based on different student demographics.

4.3.1 Entropy Calculation

Entropies addressed in this research are mobility entropies; we refer to them as entropies for short. Given the trajectory $T$ of a person, several types of entropies can be calculated as follows [30]:

1. Random entropy: This entropy ignores the spatial and temporal relationship of locations and is only concerned with the number of unique locations that a person visits. Thus, this entropy is $S_1 = \log_2(N)$, where $N$ is the number of unique locations to which a person has traveled.

2. Non-sequential entropy: This entropy considers the frequency of visited locations to determine the randomness of a person’s mobility. Thus the formula for this entropy is $S_2 = -\sum_i p(i) \log_2(p(i))$, where $p(i)$ is the frequency of location $i$ in a person’s trajectory.

3. Real entropy: This entropy considers the spatial and temporal information of a person’s trajectory. Suppose the location sequence of a person’s trajectory is
\[ L = B_1 \rightarrow B_2 \rightarrow \cdots \rightarrow B_N. \] The real entropy can be calculated as

\[ S_3 = - \sum_{L_i'} p(L_i') \log_2(p(L_i')) , \quad (4.1) \]

where \( L_i' \) is a subsequence of \( L \) and \( p(L_i') \) is the probability of \( L_i' \) appearing in \( L \). Kontoyiannis et al. [114] proposed a fast approximation as

\[ S_3 \approx \left( \frac{1}{N} \sum_{i=1}^{N} \frac{\Lambda_i}{\log_2(n)} \right)^{-1} , \quad (4.2) \]

where \( N \) is the length of the trajectory and \( \Lambda_i \) is the length of the shortest substring starting at location \( i \) that does not appear as a continuous substring between positions 1 and \( i - 1 \). Kontoyiannis et al. [114] prove that when \( N \) is large, \( S_3 \) rapidly approaches the real entropy.

In this dissertation, we choose real entropy as our main measurement instead of random or non-sequential entropies. Our rationale is that real entropy incorporates a person’s spatial and temporal features. We also calculate students’ random and non-sequential entropies. In Section 4.3.2, we show that our findings are consistent with [30].

The location sequence needed for Equation 4.1 can be constructed in two ways. The first approach does not consider students’ stay times inside buildings; hence, the trajectory is a list of buildings that a student visited. The real entropy calculated from such a trajectory is defined as time-independent entropy \( (S_{t_i}) \). It shows the randomness of a person’s mobility given only sequential geographic constraints. The other approach considers both the sequence of locations that a student visits and the time the student stays at each location. Starting from the time of a user’s first association entry on a day, we can “slice” the time of this student on that day into
intervals of $\Delta t$ each. Slicing incorporates the student’s temporal randomness in the sequence of locations. We call this type of entropy *time-dependent entropy* ($S_{td}$).

A student’s trajectory $(B_1, ST_1) \rightarrow (B_2, ST_2) \rightarrow \cdots \rightarrow (B_N, ST_N)$ is changed to $L = (B_1) \rightarrow (B_1) \rightarrow (B_1) \rightarrow \cdots \rightarrow (B_N) \rightarrow (B_N) \rightarrow (B_N)$.

For example, suppose a student’s trajectory and stay time (in minutes) in buildings are $(B_1, 60) \rightarrow (B_2, 30) \rightarrow (B_3, 34) \rightarrow (B_4, 19)$ and $\Delta t$ is 30 minutes. The location sequences in Equation 4.1 are $B_1, B_2, B_3, B_4$ and $B_1, B_1, B_1, B_2, B_3, B_3, B_4$ for $S_{ti}$ and $S_{td}$, respectively.

Students may have several mobile devices such as smartphones and tablets. We construct each mobile device’s trajectory and calculate its entropy according to Equation 4.2. We select the device with the largest entropy to represent the owner’s entropy. We calculate students’ overall long-term time-dependent entropy ($S_{td}$) from cumulative trajectories of all visited locations with a time slice $\Delta t = 30$ minutes throughout the 139-day data collection period. As students’ log entries terminate before 11:59 p.m. each day, we add a “dummy” marker to each student’s last log entry on that day to indicate the end of the day. We treat the marker as a “virtual location” in the student’s trajectory. We investigate short-term entropy on daily and weekly bases to examine students’ predictability and regularity with respect to age, major, and gender. We calculate short-term entropies $S_{td}$ and $S_{ti}$ by limiting the time interval to a shorter scale such as a day or a week instead of the entire 139-day period. In addition, we calculate entropies based on different time intervals from 1 to 9 weeks. We observe that overall entropies ($S_{td}$) for a 6-week time interval converge statistically to the overall long-term entropy measured over the 139-day period. Thus, the saturation point of
our calculated entropy is \(\sim 6\) weeks, which is shorter than the 12-week saturation time interval using cellular data [30].

### 4.3.2 Results

This section presents students’ mobility patterns using the entropy defined in Equation 4.1. We choose \(\Delta t\) as 30 minutes for \(S_{td}\) as the measured average stay time of locations in our dataset is 52.9 minutes; therefore, 30-minute slices provide temporal repetition for locations with longer stay times. In addition, we test other \(\Delta t\) values in our calculations and report the results in Section 4.3.2.

First, we present the demographic differences of entropy categorized by age, academic major, and gender and we summarize the overall patterns. For each demographic category, we discuss interesting findings based on long-term and short-term entropies.

**Differences Among Age Groups**

**Long-Term Entropy** Long-term \(S_{td}\) across age groups shows marked differences. Figure 4.4 shows the distribution of student entropies stratified by age groups.

![Figure 4.4: Histograms of long-term entropy \(S_{td}\) for different age groups.](image)
Traditional college students should be in age groups between 19 and 22. We infer students’ class ranks as follows: 19-year-olds are freshmen, 20-year-olds are sophomores, 21-year-olds are juniors, and 22-year-olds are seniors. As Figure 4.4 shows, 19- and 22-year-olds’ entropies follow a unimodal distribution with 22-year-olds’ average entropy markedly higher than that of 19-year-olds. 20- and 21-year-olds’ entropies are bimodal with peaks centered near those of 19-year-olds and 22-year-olds, respectively. Notably, the mean mobility entropy increases with age until age 22. The 23-year-old group has a similar distribution as that of the 22-year-old group. The 24-and-older (24+) group also follows a bimodal distribution.

**Discussion**  Both spatial and temporal factors may cause the entropic increase as students age. Thus, we also investigate the long-term $S_{ti}$ of students across different age groups. Results show that the distributions of $S_{ti}$ are all unimodal across age groups and the entropic increase as students age is much subtler. For students in the 19–22-year-old age groups, there is a 6% increase for $S_{ti}$ compared with the 67% increase for $S_{td}$. Thus, temporal differences across age groups should have contributed more to the observed phenomenon.

**Short-Term Entropy**  We focus on students’ entropies on each day of the week when short-term entropies are studied. There are $\sim$19 weeks throughout the observation period. We used two approaches to determine the average entropy for every day of the week: (1) For any day of a week, the average entropy for that day is calculated only using students who appeared at least once on that day of the week throughout the observation period. This requirement ensures that a student must be in the log at least once for each day of the week. 1,934 students fit this requirement, which we
call “strict”; and (2) Each student in the log contributes to the average entropy of a day of a week in which the student appears. This approach includes all students since every student in the dataset appears at least once for a day. We call this method of entropy calculation “inclusive.” We calculate entropies using both approaches and obtain similar results.

Daily entropies of students with different ages show consistent differences. Figure 4.5 shows the differences of $S_{td}$ on ages using the two aforementioned approaches. As can be seen, the daily average $S_{td}$ strictly increases as students’ ages increase from 19 to 22 using both the inclusive and the strict approaches. This finding confirms the pattern found in the overall entropy results.

![Figure 4.5](image)

Figure 4.5: Average entropies across age groups for each day of the week. The figure on the left is calculated using the “inclusive” approach, and the one on the right uses the “strict” approach.

**Differences Among Academic Majors**

**Long-Term Entropy** Long-term entropy $S_{td}$ is diverse across academic majors whereas long-term $S_{rt}$ follows similar patterns. Table 4.2 shows the mean entropies for each major.
Table 4.2: Long-term metrics for each academic major.

<table>
<thead>
<tr>
<th>Majors</th>
<th>Engineering</th>
<th>Science</th>
<th>Health</th>
<th>Business</th>
<th>Social</th>
<th>Education</th>
<th>Undecided</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean $S_{td}$</td>
<td>1.22</td>
<td>1.29</td>
<td>1.18</td>
<td>1.30</td>
<td>1.33</td>
<td>1.33</td>
<td>1.09</td>
</tr>
<tr>
<td>Mean $S_{ti}$</td>
<td>2.19</td>
<td>2.24</td>
<td>2.13</td>
<td>2.25</td>
<td>2.24</td>
<td>2.25</td>
<td>2.17</td>
</tr>
</tbody>
</table>

Figure 4.6: Histograms of 20-year-olds’ (sophomores’) $S_{td}$ for six different groups of students based on their majors. The two contours represent the long-term entropy of 19-year-olds (freshmen, solid red lines) and 21-year-olds (juniors, dashed green lines).

$S_{ti}$ values vary within 4% of each other while variations of $S_{td}$ exceed 22%. Undecided majors have the smallest $S_{td}$ and $S_{ti}$ compared to all other majors.

Another interesting observation is that the rate of entropic increase from 19-year-olds (freshmen) to 20-year-olds (sophomores) differs significantly among majors. For example, this rate of increase for engineering majors is small compared with that of business majors. Figure 4.6 shows histograms among six different majors for their 20-year-old group (sophomores). Unlike the rate of change from age 19 to age 20, the rate of entropic change from age 20 to age 21 is similar among majors. Undecided majors are not shown due to limited data after dividing into individual age categories.
Discussion The most dramatic increase occurs with business majors whose entropy mode shifts from left to right when students’ ages increase from 19 to 20.

The entropy mode shift for engineering and science majors is much smaller. We calculate the average stay time of students with respect to their ages (19 and 20) and majors. We find that business majors’ average stay times decreased from 72.9 minutes (age 19 group) to 52.5 minutes (age 20 group), a 28% drop. Engineering majors’ average stay times decreased from 68.8 minutes to 59.6 minutes, a 14% drop.

We calculate the time-independent entropy $S_{ti}$ for all students and observe no dual mode. The observed shifts between 19- and 20-year-olds across majors are minimal. For example, Figure 4.7 shows 20-year-olds’ (sophomores’) $S_{ti}$ for different majors.

![Figure 4.7: Histograms of 20-year-olds’ (sophomores’) $S_{ti}$ for six different groups of students based on their majors. The two contours represent the long-term entropy of 19-year-olds (freshmen, red solid lines) and 21-year-olds (juniors, green dashed lines).](image)

Short-Term Entropy For every day of the week, undecided majors have low entropies compared with those of every other major. Table 4.3 shows $S_{td}$ for each day
of the week for all seven majors using the inclusive approach. The strict approach yields similar results.

**Discussion** We observe that undecided majors have the lowest entropies for every day of the week compared with other majors. This finding is interesting as we would expect undecided majors to have high mobility as they explore classes around the campus, which would increase entropy. Examining the average stay time reveals that undecided majors have the highest stay time (on average) than any other group. We believe that undecided majors are generally from lower class ranks. In our dataset, over 73% of undecided majors are between ages 19 and 20 (i.e., this group has more freshmen and sophomores). Hence, this group has lower entropy than those of other groups. Also, health-related majors have relatively low entropies compared with other majors. The pattern for $S_{td}$ is similar, but undecided majors do not have the lowest $S_{ti}$. There are no distinct patterns across academic majors for daily $S_{ti}$.

Table 4.3: Short-term daily $S_{td}$ for each academic major.
Differences Between Genders

We compare the long-term overall entropies of male and female students and observe no statistical difference between them. This finding confirms the discovery in previous work [30].

Short-Term Entropy When students’ entropies are compared on different days of the week, gender differences appear. Females have relatively larger daily $S_{td}$ while this pattern is not apparent for $S_{tt}$. For each day of the week, males’ and females’ average entropies follow very similar shapes, but there are differences between genders primarily between Tuesday and Thursday.

Figure 4.8 shows that females’ $S_{td}$ entropies are slightly higher than males’ (at least 2% on Tuesdays and Thursdays for both measurements).

![Figure 4.8: Average entropy for each day of the week for males and females. The left figure shows the result using the inclusive approach; the right figure shows that using the strict approach.](image)
There are some nuanced differences between the two approaches as the “strict” approach yields slightly more observable differences between male and females during the middle of the week.

We use the non-parametric Kolmogorov-Smirnov test to determine if males’ and females’ entropies on different days of the week follow the same distribution. The test yielded statistical differences ($p < 0.001$) on Tuesday, Thursday, and Sunday with both the strict and the inclusive approaches.

**Discussion** Though the difference on certain days of a week between males and females is small, we argue that this difference is significant as the pattern holds throughout the entire observation period. Figure 4.9 shows daily average entropies for males and females throughout the 139-day observation period. Females’ daily entropies are usually a bit larger, especially during “normal” weeks (week 9 is spring break, around day 60; week 15 and thereafter are summer break, around day 100).

However, the overall entropies of females and males are statistically similar as stated above. This discrepancy is quite intriguing since females’ larger entropies on a daily basis would imply that females have higher cumulative entropies throughout the 139-day period.

We also calculate the average weekly entropies for males and females for each of the 19 complete weeks in the dataset. The weekly averages are almost identical between males and females (Kolmogorov-Smirnov test with $p = 0.9563$). Thus, females have larger daily entropies, yet the gender difference disappears as the measurement time interval increases to weeks or longer. One possible explanation for this discrepancy is as follows: females’ mobility is more random than males’ on a daily basis, but females
Figure 4.9: Males’ and females’ average daily entropies during the 139-day period. The heat map near the bottom of the figure indicates if females’ average entropy is higher than males’ on that day (gray) or not (black). Days 124–126 mostly contain laptop devices due to infrastructure problems. Average entropies for these three days are omitted.

tend to repeat their daily routines more strictly than males. Thus, when entropies are measured over a longer period, the randomness of females’ daily mobility is countered by their mobility’s long-term regularity.

Overall Patterns

Long-Term Entropy  Long-term entropy $S_{td}$ follows a bimodal distribution. As shown in Figure 4.10(a), the left mode is centered at $\sim 0.8$ and the right mode is centered at $\sim 1.65$, twice the left mode. This means that students whose entropies are near the left peak have (on average) less than $2^{0.8} \approx 1.7$ locations as possible choices if students randomly pick their locations, whereas students whose entropies are near the right peak have $2^{1.65} \approx 3$ locations as possible choices. The overall mean entropy is $1.2655$ for all students.
Discussion  The overall mean entropy calculated from this WLAN dataset slightly exceeds the result from Song et al. [30]. We hypothesize this difference is due to the finer granularity of WLAN data compared to call detail record (CDR) data.

![Graph](image)

Figure 4.10: (a) Overall long-term entropies $S_{td}$ and $S_{ti}$, of students based on cumulative trajectories across the 139-day span. (b) The convergence result of long-term overall entropies ($S_{ti}$ and $S_{td}$).

To our knowledge, the bimodal entropy distribution has not been reported in the literature, although Pappalardo et al. [115] find two distinct profiles in human mobility in their study of CDR and GPS data. We investigate possible reasons for the bimodal distribution and test the correlation between entropy and other metrics such as the radius of gyration, the number of unique locations students visit, and the total number of visited locations. None are strongly correlated with the entropy distribution’s bimodal pattern. We also find that students’ average stay time correlates with their entropy. The average stay times for students whose entropies are near the left and right modes are 80.5 minutes and 29.1 minutes, respectively. This finding reaffirms the concept of entropy. Students with longer stay time (on average) have less random mobility, yielding smaller entropies. Similarly, students with shorter stay
time (on average) change locations along their trajectories more actively, yielding larger entropies.

We increase time slice length $\Delta t$ from 30 minutes to 110 minutes with 10-minute increments. We discover that once $\Delta t$ reaches 110 minutes, the bimodal pattern no longer holds in the time-dependent entropy ($S_{td}$) distribution. This observation indicates that the unit of temporal measurement is an important parameter in the measurement of human mobility.

We also investigate the convergence time for mobility entropies. Figure 4.10(b) shows the convergence of mean $S_{td}$ and $S_{ti}$ when the measurement length spans 1–9 weeks. $S_{td}$ converges at 6 weeks when the measurement length is 6 weeks while $S_{ti}$ converges at 8 weeks.

![Graph showing daily entropy variation for different weekdays.](image)

Figure 4.11: Average entropy for each day of the week. The figure on the left uses the inclusive approach and the one on the right uses the strict approach.

**Short-Term Entropy**  Patterns of average daily entropy (both $S_{td}$ and $S_{ti}$) follow hook shapes similar to those in Figure 4.5 and Figure 4.8. Figure 4.11 shows the average daily entropy for all students.
Discussion  This hook-shaped pattern for days of a week is confirmed by examining weekly patterns during the 139-day period of data collection. The only exceptions are the special weeks of a semester such as spring break and final exams during which student mobility patterns change dramatically.

Changing mobility patterns on weekends can be partially attributed to students lingering longer at locations such as libraries and dormitories, resulting in lower mobility. However, the slightly larger mobility entropy from Tuesday to Thursday is an interesting phenomenon.

We investigate location-related metrics including the average number of (unique and total) locations visited by students as shown in Table 4.4. As can be seen, both the number of unique and total locations visited by students on each day of the week follow a very similar trend to mobility entropies.

<table>
<thead>
<tr>
<th>Days</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
<th>Sat</th>
<th>Sun</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean $S_{ld}$</td>
<td>1.48</td>
<td>1.53</td>
<td>1.54</td>
<td>1.51</td>
<td>1.48</td>
<td>1.10</td>
<td>1.22</td>
</tr>
<tr>
<td>Mean $S_{lt}$</td>
<td>2.14</td>
<td>2.18</td>
<td>2.18</td>
<td>2.17</td>
<td>2.14</td>
<td>1.98</td>
<td>1.99</td>
</tr>
<tr>
<td>Unique Locations</td>
<td>11.79</td>
<td>12.55</td>
<td>12.70</td>
<td>12.42</td>
<td>12.10</td>
<td>9.18</td>
<td>8.83</td>
</tr>
<tr>
<td>Total Locations</td>
<td>17.56</td>
<td>18.64</td>
<td>19.00</td>
<td>18.39</td>
<td>17.86</td>
<td>13.92</td>
<td>13.22</td>
</tr>
</tbody>
</table>

Table 4.4: Short-term metrics for each day of a week

4.4 Extension: Indoor WLAN Localization with Site Survey Data

Our WLAN dataset includes site survey data for current AP deployments in 73 buildings on the OSU campus. These buildings are used mainly for academic and administrative purposes. We leverage these data and design a mobile application
(app) for indoor localization on campus. Figure 4.12 shows these buildings shaded in black. Table 4.5 describes the number of floor plans for each building where AP deployment information is available.

Figure 4.12: OSU buildings with AP deployment information (shaded). Other buildings include: 22 E. 16th Avenue, 53 W. 11th Avenue, Knight House, North Commons, Northwood-High Building, Raney Commons, Riverwatch Tower, and the Wexner Center for the Arts (not shown). We generate the map using Mapzen [116] with OpenStreetMap data [117].
<table>
<thead>
<tr>
<th>Building Name</th>
<th>Number of Floors</th>
</tr>
</thead>
<tbody>
<tr>
<td>209 W. 18th Ave</td>
<td>4</td>
</tr>
<tr>
<td>22 E. 16th Ave</td>
<td>4</td>
</tr>
<tr>
<td>53 W. 11th Ave</td>
<td>1</td>
</tr>
<tr>
<td>Arps Hall</td>
<td>5</td>
</tr>
<tr>
<td>Baker Systems Engineering</td>
<td>6</td>
</tr>
<tr>
<td>Blackwell Inn</td>
<td>9</td>
</tr>
<tr>
<td>Bolz Hall</td>
<td>4</td>
</tr>
<tr>
<td>Bricker Hall</td>
<td>4</td>
</tr>
<tr>
<td>Caldwell Laboratory</td>
<td>5</td>
</tr>
<tr>
<td>CBEC</td>
<td>8</td>
</tr>
<tr>
<td>Celeste Laboratory</td>
<td>5</td>
</tr>
<tr>
<td>Central Services Building</td>
<td>3</td>
</tr>
<tr>
<td>Cockins Hall</td>
<td>5</td>
</tr>
<tr>
<td>Converse Hall</td>
<td>3</td>
</tr>
<tr>
<td>Denney Hall</td>
<td>6</td>
</tr>
<tr>
<td>Derby Hall</td>
<td>5</td>
</tr>
<tr>
<td>Dreese Laboratories</td>
<td>9</td>
</tr>
<tr>
<td>Dulles Hall</td>
<td>4</td>
</tr>
<tr>
<td>Enarson Classroom Building</td>
<td>5</td>
</tr>
<tr>
<td>Evans Laboratory</td>
<td>5</td>
</tr>
<tr>
<td>Faculty Club</td>
<td>3</td>
</tr>
<tr>
<td>Fisher Hall</td>
<td>9</td>
</tr>
<tr>
<td>Fontana Laboratory</td>
<td>3</td>
</tr>
<tr>
<td>French Field House</td>
<td>1</td>
</tr>
<tr>
<td>Gerlach Hall</td>
<td>4</td>
</tr>
<tr>
<td>Hagerty Hall</td>
<td>5</td>
</tr>
<tr>
<td>Hale Hall</td>
<td>5</td>
</tr>
<tr>
<td>Hayes Hall</td>
<td>4</td>
</tr>
<tr>
<td>Hitchcock Hall</td>
<td>5</td>
</tr>
<tr>
<td>Hopkins Hall</td>
<td>5</td>
</tr>
<tr>
<td>Hughes Hall</td>
<td>5</td>
</tr>
<tr>
<td>Ice Rink</td>
<td>1</td>
</tr>
<tr>
<td>Independence Hall</td>
<td>2</td>
</tr>
<tr>
<td>Jesse Owens Recreation Center North</td>
<td>1</td>
</tr>
<tr>
<td>Journalism Building</td>
<td>4</td>
</tr>
<tr>
<td>Knight House</td>
<td>1</td>
</tr>
<tr>
<td>Knowlton Hall</td>
<td>6</td>
</tr>
<tr>
<td>Koffolt Laboratory</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Building Name</th>
<th>Number of Floors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kuhn Honors and Scholars House</td>
<td>2</td>
</tr>
<tr>
<td>Lincoln Tower</td>
<td>5</td>
</tr>
<tr>
<td>Lincoln Tower Park</td>
<td>1</td>
</tr>
<tr>
<td>MacQuigg Laboratory</td>
<td>6</td>
</tr>
<tr>
<td>Maintenance Building</td>
<td>3</td>
</tr>
<tr>
<td>Mason Hall</td>
<td>5</td>
</tr>
<tr>
<td>Math Building</td>
<td>5</td>
</tr>
<tr>
<td>Math Tower</td>
<td>7</td>
</tr>
<tr>
<td>McPherson Laboratory</td>
<td>4</td>
</tr>
<tr>
<td>Mendenhall Laboratory</td>
<td>5</td>
</tr>
<tr>
<td>Mershon Auditorium</td>
<td>6</td>
</tr>
<tr>
<td>Newman and Wolf from Laboratory</td>
<td>5</td>
</tr>
<tr>
<td>North Commons</td>
<td>2</td>
</tr>
<tr>
<td>Northwood-High Building</td>
<td>1</td>
</tr>
<tr>
<td>Ohio Stadium</td>
<td>10</td>
</tr>
<tr>
<td>Orton Hall</td>
<td>3</td>
</tr>
<tr>
<td>Page Hall</td>
<td>4</td>
</tr>
<tr>
<td>Pfahl Hall</td>
<td>4</td>
</tr>
<tr>
<td>Physics Research Building</td>
<td>6</td>
</tr>
<tr>
<td>Raney Commons</td>
<td>2</td>
</tr>
<tr>
<td>Riverwatch Tower</td>
<td>1</td>
</tr>
<tr>
<td>Schoenbaum Hall</td>
<td>4</td>
</tr>
<tr>
<td>Science and Engineering Library</td>
<td>5</td>
</tr>
<tr>
<td>Scott Laboratory</td>
<td>5</td>
</tr>
<tr>
<td>Smith Laboratory</td>
<td>6</td>
</tr>
<tr>
<td>St. John Arena</td>
<td>4</td>
</tr>
<tr>
<td>Stillman Hall</td>
<td>5</td>
</tr>
<tr>
<td>Student Academic Services Building</td>
<td>6</td>
</tr>
<tr>
<td>Sullivant Hall</td>
<td>3</td>
</tr>
<tr>
<td>Tuttle Parking Garage</td>
<td>3</td>
</tr>
<tr>
<td>University Hall</td>
<td>5</td>
</tr>
<tr>
<td>Watts Hall</td>
<td>5</td>
</tr>
<tr>
<td>Weigel Hall</td>
<td>4</td>
</tr>
<tr>
<td>Wexner Ctr. for the Arts</td>
<td>4</td>
</tr>
<tr>
<td>Women’s Field House</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4.5: Number of floors for each building at OSU with WLAN site survey data.
OSU’s WLAN uses Aruba Networks’ APs and network management software [118]. The site survey data are provided in Extensible Markup Language (XML) files for use with Aruba’s AirWave network management software [119]. As XML is a standard file format, we parse these files and extract their internal structures. Each XML file corresponds to a single floor in one building, which is referred to as a site. Each site has one or more aps with x- and y-coordinates, (unique) names, and other information. Each ap has one or more radios with technical WLAN information such as transmitted signal strength and channel, where each channel corresponds to a single radio frequency. The files describe hardware such as Intermediate Distribution Frames (idfs), which route cables from buildings to individual APs, racks, and planning regions. Other XML tags are self-explanatory. We map the XML files’ structures to a MySQL database schema, which Figure 4.13 shows. For brevity and privacy reasons, we omit students’ WLAN (dis)associations with APs from the schema.

Next, we convert the MySQL database to SQLite [120] using the mysql2sqlite utility [121]. (Both the iOS and Android operating systems use the SQLite relational database management system.) The WLAN mobile app uses this database to find locations of nearby APs via their MAC addresses. In general, AP locations are consistent with each other for each floor of a building. The app estimates the user’s location using Algorithm 4.1.

As Algorithm 4.1 shows, the app estimates the user’s current building and floor based on the majority of nearby APs found for each floor and building. (For example, if two APs are discovered nearby that are actually deployed on the first floor of Bolz Hall and one AP is discovered that is actually deployed on the second floor, the app estimates that the user is on the first floor.) The app extracts RSSIs each AP
Algorithm 4.1 WLAN site survey localization algorithm

1: totalRssiWeight ← 0; \((x_{img}^{i_{loc}}, y_{img}^{i_{loc}}) \leftarrow (0, 0); num[b_i][f_j] \leftarrow 0 \) for all buildings \( b_i \), floors \( f_j \)
2: for each AP \( AP_i \) found nearby do
3:  Find RSSI \( rssi_i \), location \((x_i, y_i)\) for \( AP_i \), building \( b_i \), floor \( f_j \)
4:  \( num[b_i][f_j] \leftarrow num[b_i][f_j] + 1 \)
5:  rssiWeight_i \leftarrow \text{convertRssiToWeight}(rssi_i)
6:  totalRssiWeight \leftarrow totalRssiWeight + rssiWeight_i
7: end for
8: Find current building \( b_i \), floor \( f \) based on \( \max_{i,j}(num[b_i][f_j]) \)
9: for each AP \( AP_i \) found nearby do
10:  Weight \( w_i \leftarrow rssiWeight_i/totalRssiWeight \)
11:  \((x_{img}^{i_{loc}}, y_{img}^{i_{loc}}) \leftarrow \text{convertToImgCoords}((x_i, y_i))\)
12:  \((x_{img}^{i_{loc}}, y_{img}^{i_{loc}}) \leftarrow (x_{img}^{i_{loc}}, y_{img}^{i_{loc}}) \ast w_i \)
13:  Draw \( AP_i \) on floor plan image for floor \( f \) of \( b \)
14: end for
15: Draw user location \((x_{loc}^{img}, y_{loc}^{img})\) on image

Figure 4.13: WLAN site survey database schema.

discovered, which determines the weights \( w_i \) for the APs. In order to localize users, we employ multilateration using RSSIs and APs’ locations. For each AP discovered,
the app maps its real-world coordinates to image coordinates and draws the APs on floor plan imagery. The user’s location is estimated as a weighted sum of APs.

We implement the app on a Moto G (3rd generation) running Android 5.1. The app shows the user’s location via offline Mapbox maps [34], which are downloaded via Wi-Fi upon app initialization. Offline maps enable the app in order to function in the absence of network connectivity. Similarly, the app stores all buildings’ floor plans on the mobile device. Initially, the app takes 63.29 MB of space; after offline map download finishes, the app takes $\sim 135$ MB of space. We observe this space overhead is similar to that of other apps. For instance, on a Moto X Pure Edition running Android 6.0, the apps Dropbox, Google, and WeChat occupy 95.48 MB, 180 MB, and 105 MB of space, respectively. In addition, today’s smartphones typically have at least several gigabytes of storage, even on low-end devices. Figure 4.14 shows a screenshot of the app. We test the app in 113 Bolz Hall; it correctly determines the author’s location in the same room. Tests in other locations yield similar results.

4.5 Summary

This chapter presented the results of a human mobility predictability study across demographics (age, gender, and academic major). We inferred results from a comprehensive WLAN dataset that contained over 41 million log entries for over 5,000 students across several hundred buildings. We analyzed the predictability of students’ mobility via long-term and short-term entropies. We found that demographic information warrants consideration in mobility research as mobility patterns vary across demographic categories. We observed large increases in mobility entropy regarding age. Students’ overall long-term mobility entropy showed a bimodal distribution with
Figure 4.14: Screenshot of mobile application for indoor localization.

differences across academic majors. All students’ average daily entropies showed a hook-shaped pattern throughout the week. Students with undecided majors had the lowest daily entropies throughout the week. In addition, we designed and implemented an Android mobile app that localizes users in 73 campus buildings WLAN site survey information. The app uses multilateration with respect to APs’ signal strengths. We showed that location estimation generally achieves room-level accuracy.
Chapter 5: Cost-Effective Vehicular Communications with Mobile Devices

This chapter presents SquawkComm, our solution for cost-effective vehicular communications via mobile devices. SquawkComm leverages inexpensive vehicular FM transmitters that plug into cigarette lighters and vehicle stereos for communications with low latency. Mobile devices encode data to be sent as audio via 3.5mm cable that connects to FM transmitters operating at low unused frequencies. Receiving vehicles tune their stereos to these frequencies; devices receive audio via their microphones and decode the data. At the physical layer, SquawkComm uses SquawkCode for audio encoding and processing. At the link layer, SquawkComm uses SquawkLink for carrier frequency selection and channel access. We implement SquawkComm on commercial off-the-shelf (COTS) mobile devices. Our experiments in laboratory environments show its low latency and bit error rate.

5.1 Overview

Both academia and industry pay close attention to vehicular area networks (VANETs) [61]. However, VANETs are in the development phase and there is no single recognized standard [61]. Currently, diverse technologies try to solve communications problems that arise in different applications.
Wireless communication among nearby vehicles has several uses:

- Suppose someone has an injury and needs to be driven to the hospital. The vehicle driver wants to notify nearby vehicles of the situation in order to expedite medical care. NHTSA estimates that over 2.3 million people were injured on U.S. roads in 2013 alone [122].

- Occupants in nearby vehicles traveling to a common destination converse by sharing commonalities [35], which eases social interaction. A 2014 survey found that 97% of young people post on social media when traveling [123].

- When one driver lets another enter a lane of traffic, the second driver electronically thanks the first driver. Hence, the second driver does not need to wave to the first one or quickly blink emergency flashers, which are customary.

Certain communication requirements need to be met in order to enable such applications. First, latency must be minimal as opportunities for communication among proximate vehicles may be short (e.g., a few seconds). Second, people should not need to set up irritating manual connections, which may lead to missed communication opportunities. Third, communications need to be cost-effective in order to reach a majority of users with limited budgets. Fourth, communication should be convenient for vehicle occupants whose mobile devices accompany them everywhere.

Table 5.1 shows the limitations of existing work on vehicular communication regarding our requirements. We describe each type of work and its limitations:

- **Two-way radios:** Two-way radios ("walkie-talkies") seem promising as they let vehicle occupants talk with low latency and no connection establishment.
<table>
<thead>
<tr>
<th>Existing Work</th>
<th>Latency</th>
<th>Automatic Communication?</th>
<th>Cost-Effective?</th>
<th>Convenient?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Two-way radios</td>
<td>Low</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
</tr>
<tr>
<td>Smartphone FM</td>
<td>Low</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
</tr>
<tr>
<td>Bluetooth</td>
<td>High</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Ad hoc Wi-Fi</td>
<td>Low</td>
<td>x</td>
<td>x</td>
<td>✓</td>
</tr>
<tr>
<td>Wi-Fi Direct</td>
<td>High</td>
<td>x</td>
<td>x</td>
<td>✓</td>
</tr>
<tr>
<td>Mobile hotspot Wi-Fi</td>
<td>High</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Cellular</td>
<td>Moderate</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>DSRC broadcast</td>
<td>Low</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
</tr>
</tbody>
</table>

Table 5.1: Limitations of existing work for our requirements.

However, using two separate devices (two-way radios and mobile devices) is inconvenient.

- **Smartphone FM**: Researchers have used Nokia N900 smartphones’ built-in FM hardware for proximity social networking [88,90], but these old devices have low market share. As some mobile devices do not support FM signals [23], this approach does not fit communication among nearby vehicles.

- **Bluetooth**: Bluetooth takes over 10 seconds to discover nearby devices and requires connection establishment, which is tedious and ill-suited for our purposes.

- **Ad hoc Wi-Fi**: Ad hoc Wi-Fi is unavailable on most COTS mobile devices [36] unless they are rooted or jailbroken. Hence, it is impractical.
- **Wi-Fi Direct**: Wi-Fi Direct enables “grouping” mobile devices for communication [47]. However, it requires connection establishment, which can take up to two minutes [47, 87]. Thus, it does not fit our purposes.

- **Mobile hotspot Wi-Fi**: Mobile hotspot Wi-Fi lets some mobile devices act as Wi-Fi APs to which other devices connect. But prior work [37] showed that switching between these operating modes has high latency ($\geq 3$ seconds). Hence, this technology does not suit communication among nearby vehicles.

- **Cellular**: Cellular networks have been used for driver communication using COTS devices [65]. However, these networks route traffic from senders to receivers, which increases latency, and data plans may be costly. Thus, cellular networks do not fit.

- **Dedicated Short-Range Communication (DSRC) broadcast**: DSRC broadcast is a vehicular standard for rapid safety messages. DSRC has been used with smartphones [69, 70] via cellular networks or WLANs, but Wi-Fi APs are sparse on the road [124, 125]. DSRC’s lack of channel access mechanisms such as RTS/CTS can lead to “broadcast storms” [74].

A practical cost-effective system for communication among vehicles in physical proximity would be very desirable. Such a system should not require external infrastructure in order to enable communication everywhere, including remote areas where infrastructure is unavailable.

Any system for communications among nearby vehicles faces several challenges:
- **Broad Vehicular Support**: The system needs to support various vehicles, both new and old. It should not require expensive equipment such as radar that is only available on high-end vehicles [21,22].

- **Physical Layer**: All vehicles need to send and receive data wirelessly using inexpensive COTS equipment. The system needs to indicate common available channels so vehicles can hear each other’s transmissions.

- **Link Layer**: As data may be corrupted in transit, the system needs to perform error detection. In addition, the system needs to avoid collisions that arise from vehicles communicating simultaneously.

We propose SquawkComm, a novel vehicular communications system using COTS mobile devices and FM signals that achieves low latency, cost efficiency, and convenience without connection establishment or external infrastructure. SquawkComm leverages vehicle stereos’ built-in FM reception and low-cost COTS FM transmitters. It is easy to use: one just tunes transmitters and stereos to the same FM frequencies. Many countries allow unlicensed FM communication within tens of meters [126–129]. This chapter focuses on low data rate applications only. Applications needing higher data rates can use other communications techniques.

SquawkComm sends data among vehicle occupants’ mobile devices as well as data from vehicles themselves (*e.g.*, from Bluetooth dongles plugged into OBD-II ports). This is convenient for users. We identify senders and receivers via numeric vehicle IDs (VIDs). As each sender generates a new VID before sending, SquawkComm resists attempts to identify users via their devices’ MAC addresses [95,96]. SquawkComm uses two supporting technologies:
– **SquawkCode**: At the physical layer, SquawkComm achieves low latency using SquawkCode, which leverages on-off keying (OOK) to encode data as audio signals. For all audio baseband frequencies, we encode 1s as sine waves with these frequencies and 0s as unchanged audio. In the frequency domain, mobile devices detect “spikes” in Fast Fourier Transform (FFT) amplitudes corresponding to frequencies for 1s (and 0s otherwise). FM transmitters send audio to proximate vehicles. Each device chooses the lowest unused carrier frequency to tune transmitters and stereos. Thus, nearby vehicles receive audio as rapidly as that from local radio stations.

– **SquawkLink**: At the link layer, SquawkComm operates without infrastructure via SquawkLink, our distributed channel access algorithm that adapts to the number of nearby vehicle senders. We send data in frames, each of which has source and destination VIDs and a CRC checksum. If there are duplicate VIDs, the receiver notifies the sender, which generates a new VID. SquawkLink sends frames using communication “rounds” with two parts: RTS/CTS exchange and transmission. During transmission, we divide time into slots. Each vehicle sorts its list of received VIDs. The position of the vehicle’s VID in the list determines its time slot.

**Typical Working Scenario**  
Now we give a typical working scenario that shows SquawkComm in action. Suppose Alice and Bob drive vehicles, she is injured on the way to the hospital, and his vehicle is in front of hers. SquawkComm chooses the lowest unused FM carrier frequency (*e.g.*, 88.3 MHz) to which both of them tune
their vehicles’ stereos. Alice sends the message "Injured" via her device. SquawkComm encodes her message using as audio and sends it via her transmitter. Bob’s stereo plays the audio, which his device receives via its microphone, decodes the text, and displays it. Bob changes lanes, helping Alice arrive at the hospital sooner.

We implement SquawkComm on COTS mobile devices and FM transmitters. We evaluate its bit error rate and latency in a laboratory setting. We find that SquawkComm achieves low bit error rate and latency.

In summary, we make the following contributions:

– We propose SquawkComm, the first communications system for nearby vehicles that achieves low latency without infrastructure via pervasive mobile devices, automotive FM transmitters, and vehicle stereos;

– We design SquawkCode, an OOK-based physical layer scheme for encoding data as audio signals and choosing unused FM carrier frequencies;

– We design SquawkLink, a distributed link layer algorithm for channel access among communicating vehicles; and

– We implement SquawkComm on COTS mobile devices and evaluate its performance.

5.2 System Design

In this section, we present SquawkComm’s design rationale in Section 5.2.1 and discuss its system workflow in Section 5.2.2. Section 5.2.3 and Section 5.2.4 detail SquawkCode and SquawkLink, respectively.
5.2.1 Design Rationale

We face several challenges in designing such a system:

- **Carrier frequency selection**: Regulations specify that unlicensed FM transmitters have maximum ranges of tens of meters, which limits transmit power [126–129]. Under such restrictions, vehicles cannot communicate on carrier frequencies used by licensed FM radio stations. In addition, if vehicles communicate on different unused carrier frequencies, they cannot hear each other;

- **Data representation efficiency**: Our system needs to encode data as audio and decode data from audio efficiently, since FM’s audio bandwidth is limited;

- **Error detection**: Transmitted data may be corrupted by errors at the receiver side. Our system needs to detect errors and handle them accordingly; and

- **Channel collisions**: If multiple vehicles send data at the same time, their transmissions will collide, yielding unintelligible results. Our system needs to control channel access in order to avoid collisions.

SquawkComm addresses these challenges as follows. At the physical layer, we design SquawkCode, which encodes data to audio using OOK and finds the lowest unused FM carrier frequency locally on mobile devices. We develop an algorithm for unused carrier frequency selection that all devices run locally. At the link layer, we design SquawkLink, which frames data for transmission and controls channel access to avoid collisions. Recall that we repurpose vehicle stereos for receiving data sent by SquawkComm. Most vehicles have built-in stereos that demodulate FM audio in
hardware and play it via speakers. We piggyback on this functionality: mobile devices receive aud

![Diagram of vehicle communication](image)

Figure 5.1: SquawkComm workflow.

5.2.2 System Workflow

Figure 5.1 shows SquawkComm’s workflow among two vehicles within FM communication range. Each vehicle has four functional components: a mobile device, sent and received data, an in-vehicle FM transmitter, and the vehicle’s stereo. FM is used for transmission only as some COTS FM chipsets such as the Texas Instruments SN7611234 [130] support transmission, but not reception. The device connects to the FM transmitter via a cable connecting the device’s and the FM transmitter’s 3.5 mm audio jacks. People in the vehicle enter data to be sent to the device. The device encodes data as audio and sends this audio out to the FM transmitter. Other vehicles nearby receive the audio (receiving vehicles). SquawkComm repurposes these vehicle stereos for reception: audio plays via the speakers. In receiving vehicles, the
mobile device captures audio via its microphone, decodes the data, and displays it. Figure 5.1 shows the FM “transmission path” in double-line arrows; single-line arrows illustrate the FM “reception path.” We require the FM transmitter and nearby vehicles’ stereos to be tuned to the same (unused) FM frequency. We discuss how SquawkComm achieves this in Section 5.2.3.

The basic workflow can support other usage scenarios. Although Figure 5.1 shows only two vehicles for simplicity, this communications workflow generalizes to any number of vehicles in FM communication range at any point in time. In addition to user-entered text, SquawkComm can send data such as vehicular speed via the vehicle’s OBD-II port.

5.2.3 Physical Layer: SquawkCode

In this subsection, we first provide background information on FM. Next, we discuss SquawkCode, which has two parts: our on-off keying (OOK) mechanism that encodes data as audio and our algorithm that chooses the lowest unused FM carrier frequency for transmission. We discuss each in turn.

![Figure 5.2: Frequency spectrum of FM baseband signal.](image-url)

---

5.2.3 Physical Layer: SquawkCode

In this subsection, we first provide background information on FM. Next, we discuss SquawkCode, which has two parts: our on-off keying (OOK) mechanism that encodes data as audio and our algorithm that chooses the lowest unused FM carrier frequency for transmission. We discuss each in turn.
Background: FM

Figure 5.2 shows the frequency spectrum of baseband FM signals. Monaural audio is transmitted between 30 Hz and 15 kHz. A pilot tone is transmitted at 19 kHz in order to receive stereo audio, which is transmitted as two 15 kHz wide signals from 23–38 kHz and 38–53 kHz, respectively. Stereo receivers reconstruct stereo audio at 38 kHz, the second harmonic of the pilot tone. A $\pm 4$ kHz guard band surrounds the pilot tone. The Radio Data System (RDS) (known as the Radio Data Broadcast System (RDBS) in the U.S.) transmits textual information at 57 kHz ($\pm 6$ Hz). This information may include flags for types of audio (e.g., traffic, music genre), radio stations’ carrier frequencies and callsigns, and the name of the current song playing [131]. While RDS seems appealing for vehicular communication, not all vehicle stereos can receive RDS information.

FM transmitters modulate baseband signals to carrier frequencies ($\sim 88–108$ MHz) that vehicle stereos receive and demodulate. Countries including the U.S., Canada, European states, and Japan allow low-power unlicensed FM transmission within tens of meters [126–129]. SquawkComm’s design follows U.S. FCC regulations.

Encoding Data to Audio

Since all vehicle stereos can receive audio, we design SquawkCode’s scheme for encoding data as audio signals. SquawkCode’s coding scheme achieves a maximum bitrate of 108.75–118.57 bits per second, which suits low data rate applications. We elaborate this scheme as follows.

One solution is to use the entire FM monaural bandwidth (30 Hz–15 kHz) to encode data. However, this solution ignores people’s varying perception of audio loudness
at different frequencies. Figure 5.3 plots ISO 226 equal loudness contours at various phon levels, or levels of perceived loudness [132]. 1 phon is defined as the sound pressure level (in dB) of a 1 kHz sound with equal loudness. As Figure 5.3 shows, for any sound pressure level, frequencies above 10 kHz are perceived as louder than those below 10 kHz. In order to minimize irritation of vehicle occupants, we set 10 kHz as the upper bound of SquawkCode’s audio bandwidth.

Figure 5.4: Road noise in vehicles. (Screenshot colors are inverted for clarity; darker colors show greater intensity.)
Road noise in vehicles can also interfere with audio coding. We conduct an experiment in two vehicles (a 1999 Buick Century and a 2003 Chevrolet Malibu) driving 35–65 mph (∼15.6–29 m/s) on city roads and highways with the windows closed, radio off, and no one talking. We measure the frequency response over ∼90 s using a Samsung Galaxy Nexus running Android 5.1 and the mobile application (app) Analyzer [133]. We set the app’s sampling rate to 44.1 kHz and the block size to 4,096 using a Blackman-Harris window function. Figure 5.4 shows the results; considerable road noise is evident for frequencies below 2 kHz. Typically, the average amplitude of ambient noise was between −15 dB and −12 dB, with maximum amplitudes ranging from −12 dB to −3 dB. Various data spikes occur due to uneven roads and audible noise such as turn signals and vehicles’ climate control systems. We could use lower frequencies down to 30 Hz in our system, but road noise could interfere with audio coding. As a tradeoff, we treat 1 kHz as the lower bound of SquawkCode’s audio bandwidth.

SquawkCode’s coding scheme uses OOK with audio frequencies between 1 kHz and 10 kHz. For each audio baseband frequency \( f \), we encode a 1 as a sine wave \( \sin(2\pi f) \) and a 0 as unchanged audio at that frequency. We superpose the sine waves to generate the encoded audio signal. In the frequency domain, this yields “peaks” and “valleys” in FFT amplitudes corresponding to 1s and 0s, respectively. Mobile devices can easily detect these via the FFT, which decodes the data. Due to these properties, OOK suits SquawkComm well. We use an FFT with sampling rate 44.1 kHz and size 2,048. However, signal aliasing limits the granularity with which we can recover each frequency \( f \). To mitigate aliasing, we consider each sequence of five adjacent bins as a “single unit” and employ a peak detection algorithm for each sequence of bins. As
a result, each frequency 1,098 Hz, 2,048 Hz, …, 10,034 Hz represents a single bit with 84 bits in total.

Now we give an example that shows SquawkCode in action. Sending the message “Hello!” takes 56 bits using ASCII encoding. Our scheme places this message in a 76-bit or 83-bit frame (depending on the CRC checksum length as discussed in Section 5.2.4). The receiver receives the message in 0.7 seconds. Thus, we can achieve a peak bitrate of 108.75–118.57 bits per second.

**FM Carrier Frequency Selection**

Successful system operation requires senders and receivers to use a common carrier frequency for FM transmission. Carrier frequency selection entails avoiding licensed FM radio stations in order to avoid interference. In addition, in order to maximize SquawkComm’s communication range, we need to use the lowest possible unused FM carrier frequency. This follows from the Friis transmission equation:

\[
\frac{P_r}{P_t} = G_t G_r \left( \frac{\lambda}{4\pi R} \right)^2.
\]

Higher carrier frequencies have smaller wavelengths than lower carrier frequencies. If antenna gains \(G_r\) and \(G_t\) as well as the transmitted power \(P_t\) are fixed, the received power \(P_r\) is larger with lower carrier frequencies.

The number of FM transmitters near vehicles influences SquawkCode’s number of possible unused carrier frequencies. Consider \(N\) (isotropic) transmitters on a two-dimensional plane where each transmitter \(i\) is at geolocation \(x_i\) \((i = 1, \ldots, N)\). Each transmitter transmits at a frequency \(f_i\) with a certain power and gain. We have a received power threshold \(P_{th}\) that is needed to receive the transmission. There is a circle around each transmitter \(i\) with center \(x_i\) and radius \(d_{max}\) where
received power equals $P_{th}$. For any point $x$ at time $t$, we receive all FM frequencies $\{f_i\}$ whose transmitters’ Euclidean distance to $x$ is at most $d_{max}$. However, certain frequencies may be so close together (e.g., 89.3 MHz and 89.5 MHz) that there is insufficient “wiggle room” to transmit on the average frequency between them (e.g., 89.4 MHz) without interference. Thus, we cluster frequencies $\{f_i\}$ (including the minimum and maximum FM frequencies). Frequencies whose separation is below a threshold (e.g., 0.3 MHz) are placed in one cluster. Let $N_{unused}(t, x)$ denote the number of carrier frequencies, which is one less than the number of clusters. In addition, traffic patterns upper-bound the number of carrier frequencies. Let $N(t, x)$, $N_{max}(t, x)$, and $\rho(t, x) = N(t, x)/N_{max}(t, x)$ denote the number of vehicles on the road, the maximum number of vehicles corresponding to stationary bumper-to-bumper traffic, and the density of vehicles near $x$ at time $t$, respectively [134]. Thus, the number of used carrier frequencies in SquawkCode is at most \[
\min\{\lfloor N(t, x)/2\rfloor, N_{unused}(t, x)\} = \min\{\rho(t, x)N_{max}(t, x), N_{unused}(t, x)\}.
\]

We design Algorithm 5.1 that finds the lowest unused carrier frequency. In Algorithm 5.1, $\text{candFreqs}$ is the set of all unused FM carrier frequencies, $\text{usedFreqs}$ is the set of all used frequencies, and $\text{freq}$ is the returned FM frequency. $\text{getCandFreqs}(\phi, \lambda)$ takes one parameter, the smartphone’s estimated geolocation $(\phi, \lambda)$, and calls $\text{getUsedFreqs}(\phi, \lambda)$ with this location. $\text{getUsedFreqs}(\phi, \lambda)$ returns all unique FM transmitter frequencies within a bounding box with coordinates $(\phi \pm \Delta\phi, \lambda \pm \Delta\lambda)$, where $\Delta\phi$ and $\Delta\lambda$ are respective latitude and longitude thresholds specifying the size of the box. In practice, we search databases of radio stations such as FMLIST [6]. $\text{findEquidistFreqs}(\phi, \lambda)$ finds frequencies “equidistant” to those in $\text{usedFreqs}$. (For instance, given the input $\{87.5, 93.1, 101.1, 107.9\}$,
Algorithm 5.1 SquawkCode’s FM Carrier Frequency Selection

1: function getCandFmFreq(φ, λ)
2: allFmFreqs ← {87.5, ..., 107.9};
3: usedFmFreqs ← getUsedFmFreqs(φ, λ)
4: usedFmFreqs ← sort(usedFmFreqs ∪ {87.5, 107.9})
5: candFmFreqs ← findEquidistFreqs(usedFmFreqs)
6: candFmFreqs ← sort(candFmFreqs)
7: for fmFreq ∈ candFmFreqs do
8: if not isInUse(fmFreq) then
9:     fmCarFreq ← fmFreq
10:     break
11: end if
12: end for
13: return fmCarFreq
14: end function

15: function getUsedFmFreqs(φ, λ)
16: usedFmFreqs ← ∅
17: fmFreqs ← unique transmitter frequencies in bounding box (φ ± Δφ, λ ± Δλ)
18: for freq ∈ fmFreqs do
19:     usedFmFreqs ← usedFmFreqs ∪ {freq}
20: end for
21: return usedFmFreqs
22: end function

23: function findEquidistFreqs(usedFmFreqs)
24: availFmFreqs ← ∅
25: for each pair of consecutive frequencies freq_i, freq_j ∈ usedFmFreqs do
26:     newFreq ← (freq_i + freq_j)/2
27:     availFmFreqs ← availFmFreqs ∪ {newFreq}
28: end for
29: return availFmFreqs
30: end function
findEquidistFreqs() returns \{90.3, 97.1, 104.5\}. It can be seen that the algorithm yields the lowest unused FM carrier frequency.

5.2.4 Link Layer: SquawkLink

In this subsection, we discuss SquawkLink, which has two parts: framing and channel access. We describe each in turn.

Framing

Before we describe SquawkLink’s frame structure, we estimate how many vehicles on the road need to be addressed at any point in time. Consider a straight length of roadway with \(\ell\) lanes in each direction, each of width \(w\) (where \(\ell > 0\) and \(w < r\)). We model the FM communication range as a disc with radius \(r\) and center \(C\) where the roadway length is much greater than \(r\). Figure 5.5 shows our communication model; the dotted line divides traffic moving in different directions. We assume that \(\ell\) ranges between 1 and \(r/w\). Angle \(\theta\) extends between points \(A\) and \(B\) on the boundary of the disk about its center \(C\), and \(\theta = \min\{\arctan(\ell w/r), \pi/2\}\). We want to find the
roadway area $A$ inside the disk that comprises the areas of sectors $BCI$ and $CEG$ and triangles $BDC$, $DEC$, $CGH$, and $CHI$. It follows that

$$A = 2r^2 \min\{\arctan(w/r), \pi/2\} + (\ell wr)/\sqrt{(\ell wr)^2 + 1}\}.$$ 

In practical situations, while $\ell$ varies for different roadways, $w \geq \sim 2.4 \text{ m}$ [135]. For example, $\ell \approx 2$ on residential streets, but $\ell \geq 4$ on highways. We consider $w = 2.6 \text{ m}$ as this is the maximum width for commercial trucks, $r = 60 \text{ m}$ for regulatory reasons, and $\ell = 4$ [126, 136]. We find $A = 2,465.39 \text{ m}^2$, enough space for $\sim 256 \times 2.695 \text{ m} \times 1.663 \text{ m}$ stationary Smart Fortwos, each of which has 1 m behind it and in front it [137]. (Fortwos are the smallest vehicles we find with built-in FM stereos.) However, real-world roadways have irregularities such as median barriers and vehicles such as trucks are much larger than Fortwos. In addition, traffic patterns vary throughout the day, leading to congestion at “rush hours” and nearly empty roads at night. Consequently, practical roadways tend to have fewer vehicles in FM communication range at one time than in our stationary case. Thus, we design SquawkLink to address 64 vehicles.

Next, we consider how to address each vehicle in FM communication range. At minimum, a vehicle needs to address each vehicle adjacent to it on roadways. One approach uses the 17-character Vehicle Identification Number (VIN) that each vehicle on the road must have by law [138]. However, representing the VIN would take at least 136 bits (in ASCII), which does not fit SquawkComm. Another approach uses the vehicle license plate number. While license plate numbers are short (6–7 characters), administrative regions within countries typically issue license plates separately [139]. This could lead to two different vehicles having the same ID, which is unacceptable. For efficiency, we generate a random vehicle ID (VID) for each vehicle: $VID \sim U(0, N - 2)$, where $N = 64$ is the maximum number of vehicles. We represent
VID using \( [\log_2(N)] \) bits. If a sender and a receiver have duplicate VIDs, the receiver notifies the sender, which generates a new VID.

<table>
<thead>
<tr>
<th>Preamble</th>
<th>Source Address</th>
<th>Destination Address</th>
<th>Frame Type</th>
<th>Urgent</th>
<th>CRC Type</th>
<th>Data</th>
<th>CRC Checksum</th>
</tr>
</thead>
<tbody>
<tr>
<td>3 bits</td>
<td>6 bits</td>
<td>6 bits</td>
<td>2 bits</td>
<td>1 bit</td>
<td>1 bit</td>
<td>56 bits</td>
<td>1 or 8 bits</td>
</tr>
</tbody>
</table>

Figure 5.6: SquawkLink frame structure.

<table>
<thead>
<tr>
<th>Frame Type Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>RTS</td>
</tr>
<tr>
<td>01</td>
<td>CTS</td>
</tr>
<tr>
<td>10</td>
<td>Data frames</td>
</tr>
<tr>
<td>11</td>
<td>Duplicate ID frames</td>
</tr>
</tbody>
</table>

Table 5.2: SquawkLink frame types.

Now we discuss SquawkLink’s frame structure as shown in Figure 5.6. There are eight fields: the preamble, source and destination VIDs, the frame type, the urgent bit, the CRC type bit, the data payload, and a CRC checksum for error detection. SquawkLink’s frame has 76 or 83 bits in total, depending on whether we use 1-bit or 8-bit CRC checksums. The three-bit preamble 010 identifies valid frames; we ignores frames without this preamble. The source and destination VIDs are randomly-generated binary integers with values between 0 and 62. The address 63 (111111 in binary) denotes “broadcast to all vehicles” just as MAC addresses whose binary representations are all 1s denote broadcast in Ethernet and Wi-Fi MAC protocols. VIDs distinguish source and destination vehicles without using static MAC addresses.
that can identify vehicles [95,96]. There are four frame types as shown in Table 5.2, the first three of which are self-explanatory. (We discuss the fourth one shortly.) The urgent bit indicates whether the frame has high priority. High-priority frames are sent without the RTS/CTS handshake; low-priority frames are sent with the handshake. If the CRC type bit is 1, we use an 8-bit CRC checksum; if the bit is 0, we use a 1-bit CRC checksum. We use a 1-bit checksum by default; the 8-bit checksum is used for applications that require higher data integrity.

Algorithm 5.2 Channel Access Protocol

1: Generate random vehicle ID $VID$
2: if vehicle needs to send data then
3: Generate new random $VID$
4: if frame has urgent status then
5: Wait random backoff time $t_{\text{wait}} \sim U(0, t_{\text{max}})$
6: Send frame
7: if collision then
8: $t_{\text{max}} \leftarrow \min\{t_{\text{max}}, 2 \cdot t_{\text{max}}\}$
9: else
10: $t_{\text{max}} \leftarrow \max\{t_{\text{min}}, t_{\text{max}}/2\}$
11: end if
12: else
13: Start RTS timer
14: Wait short random time; send RTS, $VID$, timestamp
15: Wait until RTS timer expires
16: Sort sent, received vehicle IDs $ListVIDs$
17: if $\geq 2$ vehicles and $VID = \max\{ListVIDs\}$ then
18: Send CTS, timestamp
19: end if
20: Wait until time slot
21: Send frame for time slot duration
22: Wait until all time slots finish
23: end if
24: end if
25: if vehicle receives frame with type 11 then
26: Generate new random $VID$
27: end if
Channel Access

SquawkComm faces an important challenge: preventing several vehicle occupants from sending data at the same time. Each vehicle occupant on the road that wants to send data may not see other senders nearby and transmit, leading to collisions. Since no sender can “hear” the data it is sending, collision avoidance is desirable. No one should be starved from sending data due to other senders constantly transmitting. However, urgent frames should be delivered before other frames.

We address these challenges via SquawkLink’s channel access protocol shown in Algorithm 5.2. Although Algorithm 5.2 builds atop CSMA/CA, we tailor it for communication among rapidly moving vehicles. Initially, each vehicle occupant’s mobile device chooses its own VID at random following the approach discussed previously. The remainder of the protocol only applies to senders. For privacy reasons, each sender generates a new temporal VID before sending (line 3). If a frame has high priority, we choose a random wait time $t_{\text{wait}} \sim U(0, t_{\text{max}})$ and wait, where $t_{\text{max}}$ is the longest possible wait time. Next, we send the frame. We double or halve $t_{\text{max}}$ based on whether there is a collision (subject to thresholds $th_{\text{min}}$ and $th_{\text{max}}$). Otherwise, communication takes place in rounds (lines 13–22). At the start of a round, after waiting a short random period of time, each sender transmits an RTS frame while listening for other senders’ RTS frames. Each RTS frame includes the sender’s VID and a timestamp. When the RTS period ends, each sender has a list of all other senders’ VIDs. Each sender sorts its list and determines the position of its VID in the list. If a sender determines its VID is the maximum among all VIDs and there are least two VIDs, the sender sends a CTS frame. Time is then divided into slots; each sender waits for its slot and transmits its data. The round ends once all senders finish transmitting.
If any vehicle sender has more data to send, it needs to do so in the next round. If a sender and a receiver have duplicate temporal VIDs, the receiver notifies the sender via a frame with type 11 (see Table 5.2). The sender then generates a new random temporal VID (lines 22–24). SquawkLink’s time slot allocation mechanism prevents any nearby vehicle occupant from sending data for more time than it is allotted.

Now we analyze this protocol’s collision probability. Suppose there are $n$ vehicles in proximity, each of which has an occupant that transmits independently with probability $p$. Let $t_{RTS}$ and $t_{CTS}$ denote the time periods where vehicles send RTSs and CTSs, respectively, and let $ts$ denote the length of a time slot. We consider two extreme cases:

- **Case 1**: All vehicle occupants transmit urgent frames; and

- **Case 2**: All vehicle occupants transmit frames using time slots.

We consider Case 1 first. Collisions occur when two or vehicles transmit simultaneously (i.e., $Pr\{\text{collision}\} = \sum_{i=2}^{n} \binom{n}{i} p^i (1 - p)^{n-i}$). It follows that $Pr\{\text{collision}\} = 1 - (1 - p)^{n-1}[(1 - p) + np]$. However, in Case 2, collisions only occur before vehicle occupants determine their time slots. Thus, the collision probability only holds during RTS and CTS transmission: $(t_{RTS} + t_{CTS})/(t_{RTS} + t_{CTS} + n \cdot ts)$. As $n$ increases, the number of time slots increases and this fraction decreases. If $k$ vehicle occupants use urgent frames and $n - k$ use time slots, the collision probability “shifts” between Cases 1 and 2, depending on the value of $k$.

### 5.3 Implementation and Evaluation

In this section, we first discuss our implementation of SquawkComm followed by our experimental evaluation.
5.3.1 Implementation

We implement SquawkComm using COTS Samsung Galaxy Nexus smartphones running Android 5.1 as well as two GoGroove FlexSmart X2 FM transmitters for vehicles. Each FM transmitter plugs into a vehicle’s cigarette lighter. Each smartphone connects to an FM transmitter via 3.5 mm audio cable. Our mobile application (app) includes an offline database of FM transmitters for several countries [6]. We use these databases to find nearby transmitters (within 50 kilometers) in order to determine an unused FM carrier frequency. The app’s total size is 10.52 MB, of which 1.59 MB is used for the databases. Due to its size, our app can be installed on a wide variety of mobile devices.

We implement our app’s audio processing using FFTPack [140]. One thread in the app continuously listens for audio, computes the FFT, parses the frame of delivered data, and decodes the data. Another thread encodes user-specified textual data as audio, performs framing, and sends out the data via the smartphone’s 3.5 mm audio jack to the FM transmitter.

5.3.2 Evaluation

We evaluate our SquawkComm app in both laboratory and real-world vehicular environments. We describe each in turn.

Laboratory Environment

We evaluate SquawkComm in a laboratory environment with two Galaxy Nexus devices running Android 5.1: one device is the sender and the other is the receiver. The sender device connects to an FM transmitter whose carrier frequency is 88.3 MHz. (Our app indicates this is the lowest unused frequency.) We tune a C. Crane Pocket...
Radio [141] to this frequency. The receiver phone is placed a few centimeters from a Logitech Z80 computer speaker that connects to the radio via 3.5 mm audio cable. We use 1.25-second RTSs and data frames. We send the message “hello” in ten rounds using both urgent frames and “non-urgent” ones (i.e., our channel access algorithm). For each set of ten rounds, we record the reception latency and bit error rates (BERs) and average the results. Figure 5.7 and Figure 5.8 illustrate SquawkComm’s latency and BERs, respectively, for urgent and non-urgent frames.

![Figure 5.7: SquawkComm overall latency.](image1)

![Figure 5.8: SquawkComm bit error rate.](image2)

![Figure 5.9: Components of SquawkComm’s latency.](image3)
Figure 5.7 shows that SquawkComm’s overall latencies are 0.357 seconds and 3.178 seconds for urgent and non-urgent frames, respectively. Similarly, Figure 5.8 shows that SquawkComm’s BERs are 0.1776 (for urgent frames) and 0.1714 (for non-urgent frames). Figure 5.9 shows the components of SquawkComm’s overall latency for both types of frames. There are five components: the RTS frame duration, the RTS processing latency, the “inter-frame” latency, the data frame duration, and the frame processing latency. The first three components are absent for urgent frames. Figure 5.9 illustrates the 1.25-second frame lengths (for both RTS and data frames). For non-urgent frames, there is a 0.394-second delay between processing RTS and data frames (the inter-frame latency). RTS and data frame processing latencies are 0.146 seconds and 0.138 seconds, respectively.

**Vehicular Environment**

We evaluate our app using two vehicles: a 2017 Subaru Legacy and a 2003 Mazda CX-5. Both vehicles are parked in a parking lot next to each other in December 2016. The Legacy sends messages using one Galaxy Nexus phone and the CX-5 receives them using another Galaxy Nexus phone; both phones run Android 5.1. We synchronize both phones’ clocks via NTPSync [142] over Wi-Fi beforehand. These experiments are similar to those in the laboratory environment except the CX-5 moves one parking space away after each round of ten messages. We observe that SquawkComm sends urgent frames with latency similar to that in the laboratory. Using our channel access algorithm, we observe SquawkComm sends RTSs and frames within a few seconds. Data analysis for these experiments are part of our ongoing work.
5.4 Summary

This chapter presented SquawkComm, the first system for cost-effective vehicle communication using mobile devices and FM signals. At the physical layer, we developed SquawkCode, which included a scheme for encoding data as audio and an algorithm for choosing unused FM carrier frequencies. At the link layer, we developed SquawkLink, which included a framing mechanism and a channel access algorithm. We implemented SquawkComm using off-the-shelf smartphones and FM transmitters and experimentally evaluated it in real-world vehicles.
Chapter 6: Conclusion and Future Work

This dissertation studied three topics regarding communications in the electronic world: unobtrusive communications among mobile devices, pervasive mobile device communications measurement, and cost-effective vehicular communication with mobile devices. This chapter provides concluding remarks and directions for future work.

First, we proposed Enclave, a delegate wireless device that enables unobtrusive communication among mobile devices such as smartphones. We realized Enclave using two key supporting technologies, NameCast and PicComm. NameCast used wireless device names to unobtrusively transmit short messages without connection establishment. PicComm used the transfer of visual images for secure delivery of electronic information among mobile devices. We implemented Enclave on commercial off-the-shelf (COTS) Nexus S mobile devices running Android 2.3.3. Our experimental evaluation illustrated Enclave’s potential for unobtrusive and secure communication among mobile devices.

Second, we presented results of a pervasive mobile device communications measurement study regarding demographic categories (i.e., ages, genders, and academic majors). We examined a comprehensive WLAN dataset with over 41 million log entries for more than 5,000 students at The Ohio State University (OSU) campus. Many of the logs originated from communications among students’ pervasive devices.
We analyzed the predictability of students’ mobility via long-term and short-term entropies and found variation across demographic categories. We observed large entropic increases with age. The distribution of students’ long-term mobility entropy was bimodal with differences among majors. All students’ average daily entropies showed hook-shaped patterns throughout the week. Females’ daily entropies exceeded males’ on a weekly basis, but females showed greater potential regularities over longer periods. Students with undecided majors had lower daily entropies during the week. We offered potential explanations for our findings. In addition, we designed and implemented an Android mobile application (app) for indoor localization at OSU leveraging WLAN site survey information for 73 buildings. The app localizes users via multilateration regarding RSSIs of nearby APs with known locations. Our app generally achieves room-level localization accuracy.

Finally, we presented SquawkComm, a system for cost-effective vehicle communication using COTS mobile devices and FM signals. At the physical layer, we developed SquawkCode, which included a scheme for encoding data as audio and an algorithm for choosing unused FM carrier frequencies. At the link layer, we developed SquawkLink, which included a framing mechanism and a channel access algorithm. We implemented SquawkComm using COTS smartphones and FM transmitters and experimentally evaluated it in laboratory environments. The results showed SquawkComm’s promise for cost-effective communication among nearby vehicles with low latency and bit error rates.

While we have addressed certain challenges in the electronic world, several directions remain for future work. We describe them as follows:
– **Unobtrusive communication with more mobile devices:** Currently, our Enclave system only works with mobile devices running the Android operating system. However, numerous people use Apple iOS devices as well. According to Nielsen, 43% of American smartphones run iOS in the third quarter of 2016 [143]. In future work, we intend to expand Enclave’s unobtrusive communication to iOS devices using technologies such as Apple’s MultipeerConnectivity [144].

– **Further measurement of pervasive mobile device communications:** Our study of student mobility on OSU’s campus only used WLAN log data at building-level granularity. Actually, our dataset logs specify students’ associations with WLAN APs that are deployed within buildings. On average, there are 9–10 APs deployed on each floor of each building in our dataset. In future work, we intend to predict students’ trajectories based on their (dis)association with APs. This will provide campus administrators with fine-grained information that may assist them in facilities planning.

– **Real-world vehicular communication:** Our SquawkComm experiments have studied communication in a laboratory environment. In future work, we will evaluate SquawkComm in real-world vehicular environments and analyze the resulting latencies and bit error rates. We will study the impact of distance among nearby vehicles in stationary and mobile settings.
Appendix A: Glossary of Technical Terms

This appendix defines technical terms used in this dissertation.²

**Access Point (AP)** APs are wireless base stations in WLANs to which devices connect for network access. Each AP provides wireless service within 30–70-meter ranges (depending on the environment and number of concurrent users).

**Bluetooth** Bluetooth is a communications standard for short-range wireless personal area networks (∼10 meters). This dissertation discusses “classic” Bluetooth.

**Dedicated Short Range Communications (DSRC)** DSRC is a vehicular safety technology in which vehicles rapidly send safety messages to each other (at 1–10 Hz) regarding their motion.

**Electronic world** The electronic world consists of all mobile devices and wireless infrastructure that transmit and receive wireless signals.

**Enclave device** In our Enclave work, the enclave device is a delegate mobile device that users employ to interact with the electronic world on their behalf. For example, enclave devices may be “rental” ones for tourists traveling abroad or older devices without data plans. Chapter 3 describes such devices further.

²Trademarks are the property of their respective owners.
Entropy  Entropy quantifies the uncertainty in a person’s or a group’s mobility. Section 4.3.1 describes entropies used in our WLAN measurement study.

Frequency Modulation (FM)  FM is a standard technique that automotive transmitters use to transmit audio from mobile devices that is played on vehicle stereos. Textbooks such as [145] provide details.

Institute of Electrical and Electronic Engineers (IEEE)  IEEE is a professional society of engineers that defines communications standards [146].

Master device  In our Enclave work, the master device is a user’s primary device (such as a smartphone) that interacts with the electronic world via the user’s enclave device. Chapter 3 provides further details.

Multiple Access Control (MAC) address  A MAC address is a 48-bit address that identifies a network adapter at the link layer.

NameCast  NameCast is our supporting technology for unobtrusive communication among mobile devices without connection establishment. Section 3.2.1 explains how it uses Bluetooth device names and Wi-Fi SSIDs for this purpose.

Optical Character Recognition (OCR)  OCR is a computer vision technology that parses textual content from imagery containing such content.

On-Off Keying (OOK)  OOK is a modulation technique that encodes 1s and 0s as the presence and absence, respectively, of sine waves \( \sin(2\pi f) \) with corresponding frequencies \( f \). Section 5.2.3 explains how SquawkComm uses OOK.

Organizational Unit Identifier (OUI)  An OUI comprises the three most significant bits of a network adapter’s MAC address that uniquely identifies the
adapter’s manufacturer. Manufacturers pay IEEE fees in order to use certain OUIs. [147] lists all current OUIs in use.

**Picture Communication (PicComm)** PicComm is our supporting technology for visual communication between the master and enclave devices. Section 3.2.2 explains how our Enclave work uses PicComm.

**Service Set Identifier (SSID)** An SSID is the name that a WLAN uses to advertise network services (such as *osuwireless*). When mobile devices connect to SSIDs, they can access the network via Wi-Fi.

**SquawkComm** SquawkComm is our work for cost-effective communication among mobile devices in nearby vehicles. Chapter 5 provides further details.

**Unobtrusive communication** Unobtrusive communication is efficient wireless communication on mobile devices that minimizes user interruption. For example, device users should be able to discover nearby devices and networks without waiting 10.24 seconds (for Bluetooth) or establishing network connections manually (for Bluetooth and Wi-Fi). Chapter 3 elaborates such communication.

**Vehicular Area Network (VANET)** VANETs are ad hoc networks consisting of vehicles that communicate with roadside infrastructure, other vehicles nearby, or both. These cases are referred to as vehicle-to-infrastructure (V2I) and vehicle-to-vehicle (V2V) communications, respectively.

**Wireless Local Area Network (WLAN)** WLANs are wireless networks that organizations deploy for Internet access on their premises. WLANs have one or more APs. Wi-Fi (IEEE 802.11) is the standard for WLAN communication.
Bibliography


[71] Pilsoon Choi, Jason Gao, Nadesh Ramanathan, Mengda Mao, Shipeng Xu, Chirn-Chye Boon, Suhaib A. Fahmy Fahmy, and Li-Shiuan Peh. A case for


[104] Chloe Albanesi. HTC smartphone vulnerability exposes your personal data. [http://www.pcmag.com/article2/0,2817,2393999,00.asp](http://www.pcmag.com/article2/0,2817,2393999,00.asp).


[140] FFTPack. https://w w w . f e r n a n d e s . o r g / t x p / 2 0 0 6 / 0 5 / 1 7 / f f t p a c k - t r a n s l a t e d - t o - p u r e - i s o - c c .


