SIGNAL PROCESSING AND MACHINE LEARNING APPROACHES TO ENABLING ADVANCED SENSING AND NETWORKING CAPABILITIES IN EVERYDAY INFRASTRUCTURE AND ELECTRONICS

By

Kamran Ali

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Computer Science – Doctor of Philosophy

2020

ABSTRACT

SIGNAL PROCESSING AND MACHINE LEARNING APPROACHES TO ENABLING ADVANCED SENSING AND NETWORKING CAPABILITIES IN EVERYDAY INFRASTRUCTURE AND ELECTRONICS

By

Kamran Ali

Mainstream commercial off-the-shelf (COTS) electronic devices of daily use are usually designed and manufactured to serve a very specific purpose. For example, the WiFi routers and network interface cards (NICs) are designed for high speed wireless communication, RFID readers and tags are designed to identify and track items in supply chain, and smartphone vibrator motors are designed to provide haptic feedback (e.g. notifications in silent mode) to the users. This dissertation focuses on revisiting the physical-layer of various such everyday COTS electronic devices, either to leverage the signals obtained from their physical layers to develop novel sensing applications, or to modify/improve their PHY/MAC layer protocols to enable even more useful deployment scenarios and networking applications - while keeping their original purpose intact - by introducing mere software/firmware level changes and completely avoiding any hardware level changes. Adding such new usefulness and functionalities to existing everyday infrastructure and electronics has advantages both in terms of cost and convenience of use/deployment, as those devices (and their protocols) are already mainstream, easily available, and often already purchased and in use/deployed to serve their mainstream purpose of use.

In our works on WiFi signals based sensing, we propose signal processing and machine learning approaches to enable fine-grained gesture recognition and sleep monitoring using COTS WiFi devices. In our work on gesture recognition, we show for the first time that WiFi signals can be used to recognize small gestures with high accuracy. In our work on sleep monitoring, we propose for the first time a WiFi CSI based sleep quality monitoring scheme which can robustly track breathing and body/limb activity related vital signs during sleep throughout a night in an individual and environment independent manner.

In our work on RFID signals based sensing, we propose signal processing and machine learning approaches to effectively image customer activity in front of display items in places such as retail stores using commercial off-the-shelf (COTS) *monostatic* RFID devices (*i.e.* which use a single antenna at a time for both transmitting and receiving RFID signals to and from the tags). The key novelty of this work is on achieving multi-person activity tracking in front of display items by constructing coarse grained images via robust, analytical model-driven deep learning based, RFID imaging. We implemented our scheme using a COTS RFID reader and tags.

In our work on smartphone's vibration based sensing, we propose a robust and practical vibration based sensing scheme that works with smartphones with different hardware, can extract fine-grained vibration signatures of different surfaces, and is robust to environmental noise and hardware based irregularities. A useful application of this sensing is symbolic localization/tagging, *e.g.* figuring out whether a user's device is in their hand, pocket, or at their bedroom table, etc. Such symbolic tagging of locations can provide us with indirect information about user activities and intentions without any dedicated infrastructure, based on which we can enable useful services such as context aware notifications/alarms. To make our scheme easily scalable and compatible with mainstream COTS smartphones, we design our signal processing and machine learning pipeline such that it relies only on built-in vibration motors and microphone for sensing, and it is robust to hardware irregularities and background environmental noises. We tested our scheme on Android smartphones.

In our work on powerline communications (PLCs), we propose a distributed spectrum sharing scheme for enterprise level PLC mesh networks. This work is a major step towards using existing COTS PLC devices to connect different types of Internet of Things (IoT) devices for sensing and control related applications in large campuses such as enterprises. Our work is based on identification of a key weakness of the existing HomePlug AV (HPAV) PLC protocol that it does not support spectrum sharing, i.e., currently each link operates over the whole available spectrum, and therefore, only one link can operate at a time. Our proposed spectrum sharing scheme significantly boosts both aggregated and per-link throughputs, by allowing multiple links to communicate concurrently, while requiring a few modifications to the existing HPAV protocol.

This dissertation is dedicated to my mother, my wife Haneya, and my father who passed away during my PhD while fighting a long hard battle with cancer. Thank you for always believing in me and for encouraging and supporting me in all my endeavors.

ACKNOWLEDGEMENTS

Working towards a Ph.D. has been a deeply enriching and rewarding experience. Looking back, many people have helped shape my journey. I would like to extend them my thanks.

- First and foremost, my advisor, Prof. Alex X. Liu. My work would not have been possible without his constant guidance, unwavering encouragement, his many insights, and his exceptional resourcefulness. And most importantly, his friendship. I have been very fortunate to have an advisor who has also been a close friend. For all of this, Alex, thank you.

- I would also like to thank the rest of my dissertation committee Profs. Eric Torng, Yunhao Liu, Mi Zhang, and Guan-Hua Tu for their encouragement and insightful comments during my qualifier and comprehensive exams.

- I would also like to thank Drs. Yiannis Pefkianakis (HP Labs, Apple), Eugene Chai (NEC Labs), Kazuhito Koishida (Microsoft Applied Sciences Group), and Mohammed Alloulah (Nokia Bell Labs). I really enjoyed working with them during my summer research internships and during our collaboration after the internships.

- Throughout my Ph.D., I was supported by various NSF research grants. Thanks NSF!

- I would also like to thank Michigan State University, and specifically Department of Computer Science and Engineering for providing me financial support to attend various conferences during my Ph.D.

- Many thanks to my colleagues in Systems and Security Lab at Michigan State University. In particular, I would like to thank Faraz Ahmed, Salman Ali, Ali Munir, and Xinyu Le for numerous insightful discussions and collaborations on various projects.

- I am also deeply indebted to Dr. Ijaz Haider Naqvi, who advised my undergraduate thesis, mentored my undergraduate research at ADCOM Lab, LUMS, Pakistan, and encouraged me

to pursue Ph.D. His passion for science and scholarly pursuit has been an inspiration to me (and surely, many of his other students) and helped set me on the path on which I find myself today. I feel fortunate to have him as a close friend and an advisor. For all of your support, Dr. Ijaz, Thank you.

- Finally, I do not know how I can thank my family enough: my parents and wife, from whom I realized that kindness and devotion is endless; my siblings, who always support me no matter what, and the rest of the family members who were always supportive of my studies.

- I dedicate my dissertation to my parents, my wife Haneya, and Dr. Ijaz Haider Naqvi. Thank you all for being an amazing part of my journey!

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ALGORITHMS

CHAPTER 1

INTRODUCTION

Internet of Things (IoT) applications are becoming extremely popular and will play a key role in making different types of environments around us - such as homes, offices, schools, hospitals, factories, shopping plazas, and more - smarter. However, many IoT applications require dedicated communication and/or sensing hardware and/or infrastructure, which can often be cumbersome to deploy. My dissertation focuses on revisiting the physical-layer of of various such everyday COTS electronic devices, either to leverage the signals obtained from their physical layers to develop novel sensing applications, or to modify/improve their PHY/MAC layer protocols to enable even more useful deployment scenarios and networking applications - while keeping their original purpose intact - by introducing mere software/firmware level changes and completely avoiding any hardware level changes. Enabling such new usefulness and functionalities to existing everyday infrastructure and electronics brings advantages both in terms of cost and convenience of use, as the underlying devices and their protocols are already mainstream, easily available, and often already purchased and deployed to serve their mainstream purpose of use. However, developing such applications poses unique challenges as most mainstream COTS electronic devices of daily use are not designed to be used for purposes other than their original purpose of use.

In this dissertation, I present my research on WiFi signals based sensing, RFID signals based sensing, smartphones' vibration based sensing, and distributed spectrum sharing in Powerline Communications (PLCs) - a mechanism which leverages existing power distribution network in a building for communication. In my works on WiFi signals based sensing, I developed signal processing and machine learning approaches to enable fine-grained gesture recognition and sleep monitoring using COTS WiFi devices. In my work on RFID signals based sensing, I developed an RFID signals based imaging scheme to track customer activity in front of display items - in places such as retail stores - using COTS RFID tags and readers. In my work on smartphones' vibration based sensing, I developed a robust and practical vibration based surface recognition scheme that

works with smartphones with different hardware, can extract fine-grained vibration signatures of different surfaces, and is robust to environmental noise and hardware based irregularities. This work finds its applications in indoor localization, for example, training your smartphone to turn off lights when you put it on your bed table. And finally, as communication and sensing go hand in hand, I worked on PLC technology where I developed a distributed spectrum sharing scheme to make enterprise level PLCs based IoT networks faster.

1.1 Contributions

This dissertation takes an in-depth look at the following research problems.

1.1.1 Fine-grained Gesture Recognition Using Everyday WiFi Devices

In this work, we show for the first time that WiFi signals can also be leveraged to recognize small gestures such as keystrokes. The intuition is that while typing a certain key, the hands and fingers of a user move in a unique formation and direction and thus generate a unique pattern in the time-series of Channel State Information (CSI) values, which we call CSI-waveform for that key. In this work, we propose a WiFi signal based keystroke recognition system called WiKey. WiKey consists of two Commercial Off-The-Shelf (COTS) WiFi devices, a sender (such as a router) and a receiver (such as a laptop). The sender continuously emits signals and the receiver continuously receives signals. When a human subject types on a keyboard, WiKey recognizes the typed keys based on how the CSI values at the WiFi signal receiver end. We implemented the WiKey system using a TP-Link TL-WR1043ND WiFi router and a Lenovo X200 laptop. WiKey achieves more than 97.5% detection rate for detecting the keystroke and 96.4% recognition accuracy for classifying single keys. In real-world experiments, WiKey can recognize keystrokes in a continuously typed sentence with an accuracy of 93.5%. WiKey can also recognize complete words inside a sentence with more than 85% accuracy. In this work, we have shown that fine grained activity recognition is possible by using COTS WiFi devices. Thus, the techniques proposed in this work can be used for several localized HCI applications. Examples include zoom-in, zoom-out, scrolling, sliding, and rotating gestures for operating personal computers, gesture recognition for gaming consoles,

in-home gesture recognition for operating various household devices, and applications such as writing and drawing in the air. Other than recognizing keystrokes on conventional keyboards, our WiKey technology can be potentially used to build virtual keyboards where human users type on a printed keyboard.

1.1.2 Understanding and Modeling WiFi Signals Based Sleep Monitoring

Long-term sleep monitoring is crucial for patients with sleep disorders, as well as, for the general population so that people can keep track of their sleep quality and improve their sleeping habits. Moreover, continual sleep monitoring can help with early identification of sleep disorders and related illnesses which would otherwise go undiagnosed. Recently, WiFi Channel State Information (CSI) signals based methods have emerged as an effective approach to low-cost and easily adoptable sleep monitoring for in-home environments. The idea is to track breathing and other body/limb activity, which are closely related to sleep quality in humans, by leveraging the changes caused by those bodily motions in CSI signals. However, their key limitation lies in the lack of a model that can correlate the changes introduced in CSI with those bodily motions. In this work, we propose Serene, a WiFi CSI based sleep quality monitoring scheme which can robustly track breathing and body/limb activity related vital signs during sleep throughout a night in an individual and environment independent manner. We develop two models based on which we design Serene's signal processing pipeline: a *breath-multipath* model, and a *breath-subspace* model. Our breath-multipath model quantifies the effect of small breathing movements on the CSI signals, and allows Serene to robustly extract breathing waveforms. Our breath-subspace model quantifies how breathing affects the subspace formed by WiFi subcarriers very differently compared to other bodily motions, and allows Serene to robustly differentiate between breathing and body/limb activity during sleep. We implemented Serene with commodity off-the-shelf (COTS) WiFi hardware, and tested on 5 different individuals, where we collected more than 550 hours (80 nights) of CSI data at their apartments. 55% of our dataset corresponds to NLOS deployment scenarios, and 45% to LOS. Our results demonstrate that Serene can track breathing with an average error of <0.59 BPM breaths per minute

(BPM) for controlled sleep experiments and an average error of <1.19 BPM for real-world full-night in-home sleep experiments, respectively. We conclude with qualitative sleep assessments for our study participants based on a light weight sleep scoring algorithm.

1.1.3 Monitoring Browsing Behavior of Customers via RFID Imaging

In this work, we propose to use commercial off-the-shelf (COTS) *monostatic* RFID devices (*i.e.* which use a single antenna at a time for both transmitting and receiving RFID signals to and from the tags) to effectively image customer activity in front of display items in places such as retail stores. To this end, we propose *TagSee*, a multi-person tracking system based on monostatic RFID imaging. TagSee is based on the insight that when customers are browsing the items on a shelf, they stand between the tags deployed along the boundaries of the shelf and the reader, which changes the multi-paths that the RFID signals travel along, and both the RSS and phase values of the RFID signals that the reader receives change. Based on these variations observed by the reader, TagSee constructs a coarse grained image of the customers. Afterwards, TagSee identifies the items that are being browsed by the customers by analyzing the constructed images. The key novelty of this work is on achieving multi-person activity tracking in front of display items by constructing coarse grained images via robust, analytical model-driven deep learning based, RFID imaging. To achieve this, we first mathematically formulate the problem of imaging humans using monostatic RFID devices and derive an approximate analytical imaging model that correlates the variations caused by human obstructions in the RFID signals. Based on this model, we then develop a deep learning framework to robustly image customers with high accuracy. We implement TagSee scheme using a Impinj Speedway R420 reader and SMARTRAC DogBone RFID tags. Our experimental results show that, on average, TagSee can achieve a true positive rate (TPR) of ~90% and a false positive rate (FPR) of ~10% using training data from just 2-3 users. Moreover, TagSee can achieve a TPR of more than ~80% and a FPR of less than ~15% in multi-person scenarios using training data from just 3-4 users.

1.1.4 Fine-grained Vibration Based Sensing Using a Smartphone

Vibration based sensing has been shown to be a low-cost and effective approach to recognizing different surfaces. The key intuition is that different surfaces respond to the same vibration differently. Previous schemes either use custom hardware for creating and sensing vibration, which makes them difficult to adopt, or use inertial (IMU) sensors in commercial off-the-shelf (COTS) smartphones to sense movements produced due to vibrations, which makes them coarse-grained because of the low sampling rates of IMU sensors. In this work, we propose VibroTag, a robust and practical vibration based sensing scheme that works with smartphones with different hardware, can extract fine-grained vibration signatures of different surfaces, and is robust to environmental noise and hardware based irregularities. The key intuition is that as the vibrating mass inside a smartphone's vibrator motor repeatedly moves to and fro, the vibrating mass causes the whole smartphone structure and the hardware inside it to vibrate in a peculiar pattern, which depends upon the vibration response (or absorption properties) of the surface that the smartphone is placed on. These vibrations produce peculiar sound waves that VibroTag detects using the smartphone's microphone. To make VibroTag easily scalable and compatible with COTS smartphones, we design VibroTag's signal processing and machine learning pipeline such that it relies only on built-in vibration motors and microphone for sensing, and it is robust to hardware irregularities and background environmental noises. We implemented VibroTag on two different Android phones and evaluated in multiple different environments. Our results show that VibroTag achieves an average accuracy of 86.55% while recognizing 24 different surfaces, which is more than 37% higher than the average accuracy of 49.25% achieved by the state-of-the-art IMUs based scheme, which we implemented for comparison with VibroTag.

1.1.5 Distributed Spectrum Sharing for Enterprise Powerline Communication Networks

As powerline communication (PLC) technology does not require dedicated cabling and network setup, it can be used to easily connect multitude of IoT devices deployed in enterprise environments for sensing and control related applications. While PLC technology has the potential to improve

connectivity and allow for new applications in enterprise settings, it has been mainly deployed in home networks and its deployment in enterprise settings has been largely overlooked. IEEE has standardized the PLC protocol in IEEE 1901, also known as HomePlug AV (HPAV) [3, 5], which has been widely adopted in mainstream PLC devices. A key weakness of HPAV protocol is that it does not support spectrum sharing. Currently, each link in an HPAV PLC network operates over the whole available spectrum, and only one link can operate at any time within a single collision domain. In this work, through an extensive measurement study of HPAV PLCs in a real enterprise environment using commodity off-the-shelf (COTS) HPAV PLC devices, we discover that spectrum sharing can significantly benefit enterprise level PLC networks. To this end, we propose a distributed spectrum sharing technique for enterprise HPAV PLC networks, and show that fine-grained distributed spectrum sharing on top of current HPAV MAC protocols can boost the aggregated and per-link throughput by up to 60% and 250% respectively, by allowing multiple PLC links to communicate concurrently, while requiring a few modifications to the existing HPAV devices and protocols.

CHAPTER 2

FINE-GRAINED GESTURE RECOGNITION USING EVERYDAY WIFI DEVICES

2.1 Introduction

Keystroke privacy is critical for ensuring the security of computer systems and the privacy of users as what being typed could be passwords or privacy sensitive information. The research community has studied various ways to recognize keystrokes in context of computer systems and the privacy of users, which can be classified into three categories: acoustic emission based approaches, electromagnetic emission based approaches, and vision basead approaches. Acoustic emission based approaches recognize keystrokes based on either the observation that different keys in a keyboard produce different typing sounds [13, 170] or the observation that the acoustic emanations from different keys arrive at different microphones at different time [169]. Electromagnetic emission based approaches recognize keystrokes based on the observation that the electromagnetic emanations from the electrical circuit underneath different keys in a keyboard are different [143]. Vision based approaches recognize keystrokes using computer vision technologies [16].

In this work, we show for the first time that commodity WiFi devices can also be used to recognize keystrokes. The key intuition is that while typing a certain key, the hands and fingers of a user move in a unique direction and formation, generating a unique pattern in the time-series of Channel State Information (CSI) values, which we call *CSI-waveform* of that key. The keystrokes of each key introduce relative unique multi-path distortions in WiFi signals and this uniqueness can be leveraged to recognize keystrokes. Due to the high data rates supported by modern WiFi devices, WiFi cards provide enough CSI values within the duration of a keystroke to construct a high resolution CSI-waveform for each keystroke.

In this work, we propose a WiFi signal based keystroke recognition system called WiKey. WiKey consists of two Commercial Off-The-Shelf (COTS) WiFi devices, a sender (such as a router) and a receiver (such as a laptop), as shown in Figure 4.2. The sender continuously emits signals and

the receiver continuously receives signals. When a human subject types in a keyboard, on the WiFi signal receiver end, WiKey recognizes the typed keys based on how the CSI value changes. CSI values quantify the aggregate effect of wireless phenomena such as fading, multi-paths, and Doppler shift on the wireless signals in a given environment. When the environment changes, such as a key is being pressed, the impact of these wireless phenomena on the wireless signals change, resulting in unique changes in the CSI values.

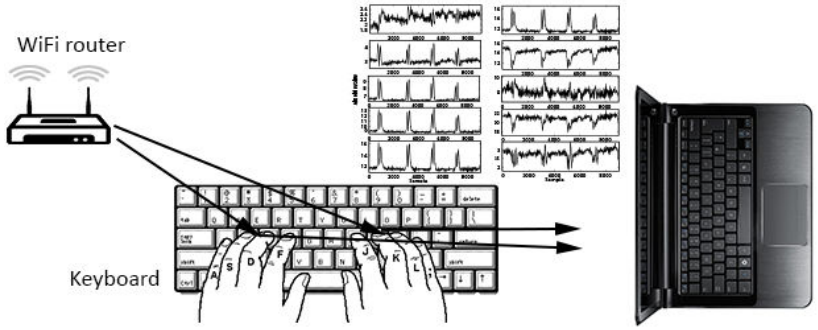


Figure 2.1: WiKey System

There are three key technical challenges. The first technical challenge is to segment the CSI time series to identify the start time and end time of each keystroke. We studied the characteristics of typical CSI-waveforms of different keystrokes and observed that the waveforms of different keys show a similar rising and falling trends in the changing rate of CSI values. Based on this observation, we design a keystroke extraction algorithm that utilizes CSI streams of all transmit-receive antenna (TX-RX) pair pairs to determine the approximate start and end points of individual keystrokes in a given CSI-waveform by continuously matching the trends in CSI time series with the experimentally observed trends using a sliding window approach.

The second technical challenge is to extract distinguishing features for generating classification models for each of the 37 keys (10 digits, 26 alphabets and 1 space-bar). As the keys on a keyboard are closely placed, conventional features such as maximum peak power, mean amplitude, root mean square deviation of signal amplitude, second/third central moment, rate of change, signal energy or entropy, and number of zero crossings cannot be used because the values of these features for adjacent keys are almost identical. To address this challenge, we use the CSI-waveform shapes of

each key from each TX-RX antenna pair as features. As the waveforms for each key contain a large number of samples, we apply the Discrete Wavelet Transform (DWT) technique on these waveforms to reduce the number of samples while keeping the shape preserving time and frequency domain information intact. We use the waveforms resulting from the DWT of individual keystrokes as their shape features.

The third challenge is to compare shape features of any two keystrokes. The midpoints of extracted CSI-wavforms of different keystrokes rarely align with each other because the start and end points determined by extraction algorithm are never exact. Moreover, the lengths of different keystroke waveforms also differ because the duration of pressing any key is often different. Consequently, the midpoints and lengths of shape features do not match either. Another issue is that the shape of different keystroke waveforms of the same key are often distorted versions of each other because of slightly different formation and direction of motion of hands and fingers while pressing that key. Thus, two shape features cannot be compared using standard measures like correlation coefficient or Euclidean distance. To address this challenge, we use the Dynamic Time Warping (DTW) technique to quantify the distance between the two shape features. DTW can find the minimum distance alignment between two waveforms of different lengths.

The key novelty of this work is on proposing the first WiFi signal based keystroke recognition approach. Some recent work uses CSI values to recognize various *macro* aspects of human movements such as falling down [48], household activities [148], detection of human presence [168], and estimating the number of people in a crowd [156]. These schemes extract coarse grained information from the CSI values to recognize the macro-movements such as falling down or recognizing fullbody/limb gestures. They cannot be directly adapted to recognize keystrokes because such coarse grained information does not capture the minor variations in the CSI values caused by human *micro*-movements such as those of hands and fingers while typing. Some recent work, namely WiHear, uses CSI values to extract the micro-movements of mouth to recognize 9 syllables in the spoken words [145]. However, WiHear uses special hardware including directional antennas and stepper motors to direct WiFi beams towards speaker's mouth and extract the micro-movements.

We implemented the WiKey system using a TP-Link TL-WR1043ND WiFi router and a Lenovo X200 laptop. In the evaluation process, we first build a keystroke database of 10 human subjects with IRB approval. WiKey achieves more than 97.5% detection rate for detecting the keystroke and 96.4% recognition accuracy for classifying single keys. In real-world experiments, WiKey can recognize individual keystrokes in a continuously typed sentence with an accuracy of 93.5%. Moreover, WiKey can recognize complete words in a sentence with more than 85% accuracy. In this work, we have shown that fine grained activity recognition is possible by using COTS WiFi devices. Thus, the techniques proposed in this work can be used for several HCI applications. Examples include zoom-in, zoom-out, scrolling, sliding, and rotating gestures for operating personal computers, gesture recognition for gaming consoles, in-home gesture recognition for operating various household devices, and applications such as writing and drawing in the air. Other than recognizing keystrokes on conventional keyboards, our WiKey technology can be potentially used to build virtual keyboards where human users type on a printed keyboard.

2.2 Related Work

2.2.1 Device Free Activity Recognition

Device-free activity recognition solutions use the variations in wireless channel to recognize human activities in a given environment. Existing solutions related to our work can be grouped into three categories: (1) RSS based, (2) CSI based and (3) Software Defined Radio (SDR) based.

RSS Based: Sigg *et al.* proposed activity recognition schemes that utilize RSS values of WiFi signals to recognize four activities including crawling, lying down, standing up, and walking [123, 124]. They achieved activity recognition rates of over 80% for these four activities. To obtain the RSS values from WiFi signals, they used USRPs, which are specialized hardware devices compared to the COTS WiFi devices that we used in our work. While RSS values can be used for recognizing macro-movements, they are not suitable to recognize the micro-movements such as those of fingers and hands in keyboard typing because RSS values only provide coarse-grained information about the channel variations and do not contain fine-grained information about small

scale fading and multi-path effects caused by these micro-movements.

CSI Based: CSI values obtained from COTS WiFI network interface cards (NICs) (such as Intel 5300 and Atheros 9390) have been recently proposed for activity recognition [48, 94, 145, 148, 156, 168] and localization [120, 157, 159]. Han *et al.* proposed WiFall that detects fall of a human subject in an indoor environment using CSI values [48]. Zhou *et al.* proposed a passive human detection scheme which exploits multi-path variations for detecting human presence in an indoor environment using CSI values [168]. Zou *et al.* proposed Electronic Frog Eye that counts the number of people in a crowd using CSI values by treating the people reflecting the WiFi signals as "virtual antennas" [156]. Wang *et al.* proposed E-eyes that exploits CSI values for recognizing household activities such as washing dishes and taking a shower [148]. Nandakumar *et al.* leverage the CSI and RSS information from off-the-shelf WiFi devices to classify four arm gestures - push, pull, lever, and punch [94]. The fundamental difference between these schemes and our scheme is that these schemes extract coarse grained features from the CSI values provided by the COTS WiFi NIC to perform these tasks while our proposed scheme refines these CSI to capture fine grained variations in the wireless channel for recognizing keystrokes. Wang *et al.* propose WiHear that uses CSI values recognizes the shape of mouth while speaking to detect whether a person is uttering one of a set of nine predefined nine syllables [145]. While WiHear can capture the micro-movements of lips, it uses special purpose directional antennas with stepper motors for directing the antenna beams towards a person's mouth to obtain a clean signal for recognizing mouth movements. In contrast, our proposed scheme does not use any special purpose equipment and recognizes the micro-movements of fingers and hands using COTS WiFi NIC.

SDR Based: Researchers have proposed schemes that utilize SRDs and special purpose hardware to transmit and receive custom modulated signals for activity recognition [8, 64, 85, 110]. Pu *et al.* proposed WiSee that uses a special purpose receiver design on USRPs to extract small Doppler shifts from OFDM WiFi transmissions to recognize human gestures [110]. Kellogg *et al.* proposed to use a special purpose analog envelop detector circuit for recognizing gestures within a distance of up to 2.5 feet using backscatter signals from RFID or TV transmissions [64] . Lyonnet

et al. use micro Doppler signatures to classify gaits of human subjects into multiple categories using specialized Doppler radars [85]. Adib *et al.* proposed WiTrack that uses a specially designed frequency modulated carrier wave radio frontend to track human movements behind a wall [8]. Recently, Chen *et al.* proposed an SDR based custom receiver design which can be used to track keystrokes using wireless signals [23]. Compared to these schemes, our scheme does not use any specialized hardware or SDRs rather utilizes COTS WiFi NICs to recognize keystrokes.

2.2.2 Keystrokes Recognition

To the best of our knowledge, there is no prior work on recognizing keystrokes by leveraging variations in wireless signals using commodity WiFi devices. Other than the SDRs based keystroke tracking approach proposed in [23] which uses wireless signals to track keystrokes, researchers have proposed several keystrokes recognition schemes that are based on other sensing modalities such as acoustics [13, 25, 169, 170], electromagnetic emissions [143], and video cameras [16]. Next, we give a brief overview of the other existing schemes that utilize these sensing modalities to recognize keystrokes.

Acoustics Based: Asonov *et al.* proposed a scheme to recognize keystrokes by leveraging the observation that different keys of a given keyboard produce slightly different sounds during regular typing [13]. They used back-propagation neural network for keystroke recognition and fast fourier transform (FFT) of the time window of every keystroke peak as features for training the classifiers. Zhuang *et al.* proposed another scheme that recognizes keystrokes based on the sounds generated during key presses [170]. They used cepstrum features [25] instead of FFT as keystroke features and used unsupervised learning with language model correction on the collected features before using them for supervised training and recognition of different keystrokes. Zhu *et al.* proposed a context-free geometry-based approach for recognizing keystrokes that leverage the acoustic emanations from keystrokes to first calculate the time difference of keystroke arrival and then estimate the physical locations of the keystrokes to identify which keys are pressed [169].

Electromagnetic Emissions Based Vuagnoux *et al.* used a USRP to capture the electromagnetic

emanations while pressing the keys [143]. These electromagnetic emanations originated from the electrical circuit underneath each key in conventional keyboards. The authors proposed to capture the entire raw electromagnetic spectrum and process it to recognize the keystrokes. Unfortunately, this scheme is highly susceptible to background electromagnetic noise that exists in almost all environments these days such as due to microwave ovens, refrigerators, and televisions.

Video Camera Based Balzarotti *et al.* proposed ClearShot that processes the video of a person typing to reconstruct the sentences (s)he types [16]. The authors propose to use context and language sensitive analysis for reconstructing the sentences.

2.3 Channel State Information

Modern WiFi devices that support IEEE 802.11n/ac standard typically consist of multiple transmit and multiple receive antennas and thus support MIMO. Each MIMO channel between each transmit-receive (TX-RX) antenna pair of a transmitter and receiver comprises of multiple subcarriers. These WiFi devices continuously monitor the state of the wireless channel to effectively perform transmit power allocations and rate adaptations for each individual MIMO stream such that the available capacity of the wireless channel is maximally utilized [46]. These devices quantify the state of the channel in terms of CSI values. The CSI values essentially characterize the *Channel Frequency Response* (CFR) for each subcarrier between each transmit-receive (TX-RX) antenna pair. As the received signal is the resultant of constructive and destructive interference of several multipath signals scattered from the walls and surrounding objects, the disturbances caused by movement of hands and fingers while typing on a keyboard near the WiFi receiver not only lead to changes in previously existing multipaths but also to the creation of new multipaths. These changes are captured in the CSI values for all subcarriers between every TX-RX antenna pair and can then be used to recognize keystrokes.

Let M_T denote the number of transmit antennas, M_R denote the number of receive antennas and S_c denote the number of OFDM sub-carriers. Let $\mathbf{X}_i$ and $\mathbf{Y}_i$ represent the M_T dimensional transmitted signal vector and M_R dimensional received signal vector, respectively, for subcarrier i and let $\mathcal{N}_i$ represent an M_R dimensional noise vector. An $M_R \times M_T$ MIMO system at any time

instant can be represented by the following equation:

$$\mathbf{Y}_i = \mathsf{H}_i \mathbf{X}_i + \mathcal{N}_i \quad i \in [1, S_c] \tag{2.1}$$

In the equation above, the $M_R \times M_T$ dimensional channel matrix H_i represents the Channel State Information (CSI) for the sub-carrier i. Any two communicating WiFi devices estimate this channel matrix H_i for every subcarrier by regularly transmitting a known preamble of OFDM symbols between each other. For each Tx-Rx antenna pair, the driver of our Intel 5300 WiFi NIC reports CSI values for $S_c = 30$ OFDM subcarriers of the 20 MHz WiFi Channel [47]. This leads to 30 matrices with dimensions $M_R \times M_T$ per CSI sample.

2.3.1 WiKey Overview

To recognize keystrokes from CSI time series, WiKey needs classification models for all keystrokes. WiKey first generates these classification models using the following four steps and then uses them to classify previously unseen keystrokes.

The first step is to remove noise from the time series of CSI values. The CSI time series reported by WiFi NICs contain a large amount of noise even when the environment is static. WiKey removes the noise in two steps. First, it passes CSI time series of all subcarriers for each TX-RX antenna pair through a low-pass filter to remove high frequency noises. Second, it leverages our observation that the variations in the CSI time series of all subcarriers due to the movements of hands and fingers are correlated and applies Principal Component Analysis (PCA) on the filtered subcarriers to extract the signals that only contains variations caused by movements of hands with acceptable levels of noise.

The second step is to detect the starting and ending points of keystrokes and extract the CSI waveforms for individual keystrokes. WiKey uses our keystroke extraction algorithm to identify the starting and ending points of individual keystrokes in a given CSI time series by leveraging the observation that CSI waveforms of different keystrokes show similar trends in the rates of change in the CSI values at the start and end of any keystroke. Our keystroke extraction algorithm takes into account the variations in the CSI time series of all subcarriers for all TX-RX antenna pairs

during keystroke extraction to minimize chances of detection errors, including missed keystrokes, false positives and detection of the same keystroke multiple times.

The third step is to extract appropriate features from the CSI waveforms of keystrokes to generate classification models. For this, WiKey applies Discrete Wavelet Transform (DWT) on those waveforms to obtain shape features of keystrokes. These shape features obtained from DWT preserve both frequency and time domain information of the CSI waveforms and while at the same time reduce the number of samples in the CSI waveform, which helps in reducing the computational cost.

The fourth step is to generate classification models using these shape features for keystrokes. WiKey trains an ensemble of classifiers to generate classification model for each key using the training data of the user. We chose k-Nearest Neighbor (kNN) classifier because it essentially searches the entire feature space to match the shape features of one keystroke with others, and thus is most suited for this particular application. To compare the shape features of any two keystrokes, WiKey uses Dynamic Time Warping (DTW) based distance metric while training the kNN classifier.

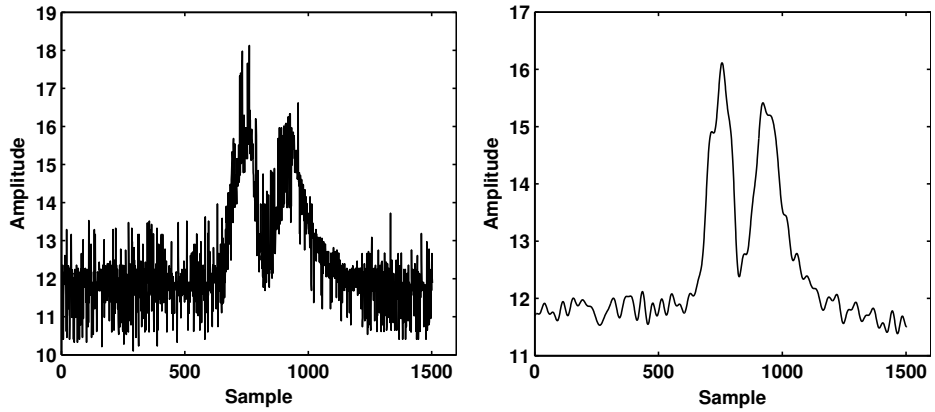
2.4 Noise Removal

The CSI values provided by commodity WiFi NICs are inherently noisy because of the frequent changes in internal CSI reference levels, transmit power levels, and transmission rates. To use CSI values for recognizing keystrokes, such noise must first be removed from the CSI time series. For this, WiKey first passes the CSI time series from a low-pass filter to remove high frequency noises. Unfortunately, a simple low pass filter does not denoise the CSI values very efficiently. Although strict low-pass filtering can remove noise further, it causes loss of useful information from the signal as well. To extract useful signal from the noisy CSI time series, WiKey leverages our observation that the variations in the CSI time series of all subcarriers due to the movements of hands and fingers are correlated. Therefore, it applies Principal Component Analysis (PCA) on the filtered subcarriers to extract the signals that only contain variations caused by movements of hands. Next, we first describe the process of applying the low-pass filter on the CSI time series and then explain

how WiKey extracts hand and finger movement signal using our PCA based approach.

2.4.1 Low Pass Filtering

The frequency of variations caused due to the movements of hands and fingers lie at the low end of the spectrum while the frequency of the noise lies at the high end of the spectrum. To remove noise in such a situation, *Butterworth* low-pass filter is a natural choice which does not significantly distort the phase information in the signal and has a maximally flat amplitude response in the passband and thus does not distort the hand and finger movement signal much. WiKey applies the Butterworth filter on the CSI time series of all subcarriers in each TX-RX antenna pair so that every stream experiences similar effects of phase distortion and group delay introduced by the filter. Although this process helps in removing some high frequency noise, the noise is not completely eliminated because Butterworth filter has slightly slow fall off gain in the stopband.



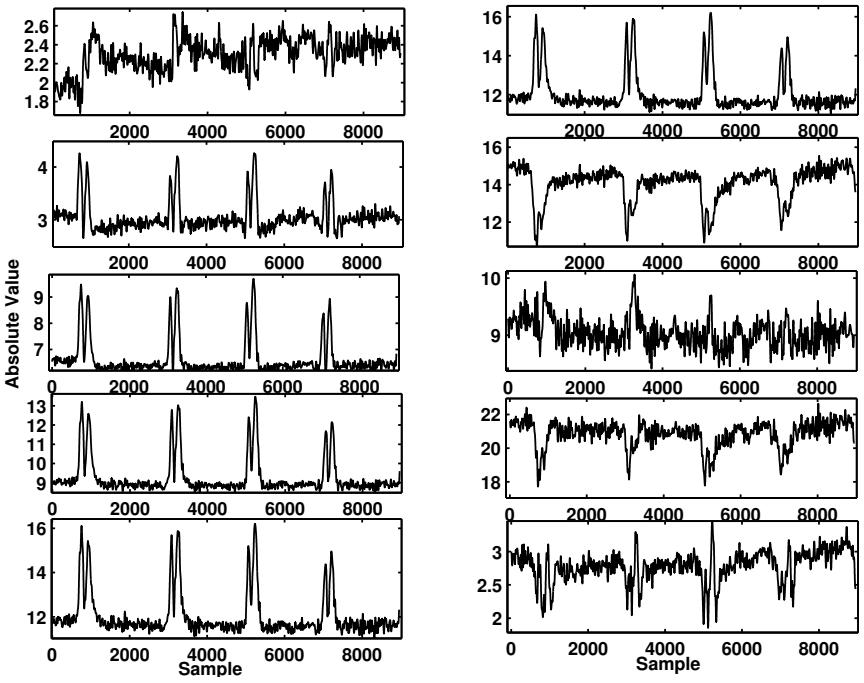
(a) Original time series (b) Filtered time series

Figure 2.2: Original and filtered CSI time series

We observed experimentally that the frequencies of the variations in CSI time series due to hand and finger movements while typing approximately lie anywhere between 3Hz to 80 Hz. As we sample CSI values at a rate of $F_s = 2500$ samples/s, we set the cut-off frequency ω_c of the Butterworth filter at $\omega_c = \frac{2\pi * f}{F_s} = \frac{2\pi * 80}{2500} \approx 0.2$ rad/s. Figure 2.2(a) shows the amplitudes of the unfiltered CSI waveform of a keystroke and Figure 2.2(b) shows the resultant from the Butterworth filter. We observe that Butterworth filter successfully removes most of the bursty noises from the CSI waveforms.

2.4.2 PCA Based Filtering

We observed experimentally that the movements of hands and fingers results in correlated changes in the CSI time series for each subcarrier in every transmit-receive antenna pair. Figure 2.3 plots the amplitudes of CSI time series of 10 different subcarriers for one transmit-receive antenna pair while a user was repeatedly pressing a key. We observe from this figure that all subcarriers show correlated variations in their time series when the user presses the keys. The subcarriers that are closely spaced in frequency show identical variations whereas the subcarriers that farther away in frequency show non-identical changes. Despite non-identical changes, a strong correlation still exists even across the subcarriers that are far apart in frequency. WiKey leverages this correlation and calculates the principal components from all CSI time series. It then chooses those principal components that represent the most common variations among all CSI time series.



(a) # 1,2,3,4,5 (b) # 5,10,15,20,25

Figure 2.3: Correlated variations in subcarriers

There are two main advantages of using PCA. First, PCA reduces the dimensionality of the CSI information obtained from the 30 subcarriers in each TX-RX stream, which is useful because using information from all subcarriers for keystroke extraction and recognition significantly increases

the computational complexity of the scheme. Consequently, PCA automatically enables WiKey to obtain the signals that are representative of hand and finger movements, without having to devise new techniques and define new parameters for selecting appropriate subcarriers for further processing. Second, PCA helps in removing noise from the signals by taking advantage of correlated varations in CSI time series of different subcarriers. It removes the uncorrelated noisy components, which can not be removed through traditional low pass filtering. This PCA based noise reduction is one of the major reasons behind high keystroke extraction and recognition accuracies of our scheme.

2.5 Keystroke Extraction

WiKey segments the CSI time series to extract the CSI waveforms for individual keystrokes. For this, WiKey operates on the CSI time series resulting from the butterworth filtering. Let $\mathsf{H}_{t,r}(i)$ be an $S_c \times 1$ dimensional vector containing the CSI values of the S_c subcarriers between an arbitrary TX-RX antenna pair $t - r$ for the i^{th} CSI sample. Let $\mathbf{H}_{t,r}$ be an $N \times S_c$ dimensional matrix containing the CSI values of the S_c subcarriers between an arbitrary TX-RX antenna pair $t - r$ for N consecutive CSI samples. This matrix is given by the following equation:

$$\mathbf{H}_{t,r} = [\mathsf{H}_{t,r}(1)|\mathsf{H}_{t,r}(2)|\mathsf{H}_{t,r}(3)|...|\mathsf{H}_{t,r}(N)]^T \tag{2.2}$$

The columns of the matrix $\mathbf{H}_{t,r}$ represent the CSI time series for each OFDM subcarrier. To detect the starting and ending points of any arbitrary key, WiKey first normalizes the $\mathbf{H}_{t,r}$ matrix such that every CSI stream has zero mean and unit variance. We denote the normalized version of $\mathbf{H}_{t,r}$ by $\mathbf{Z}_{t,r}$. WiKey then performs the PCA based dimensionality reduction and denoising (as described in Section 2.4.2) on $\mathbf{Z}_{t,r}$ and the resultant waveforms are further processed to detect the starting and ending points of the keystrokes from this particular TX-RX antenna pair. WiKey repeats this process on the CSI time series for all antenna pairs and obtains values for starting and ending points for keys based on the CSI time series from each antenna pair one by one. Finally, WiKey combines the starting and ending points obtained from all TX-RX antenna pairs to calculate a robust estimate of starting and ending points of the time windows containing those keystrokes.

Next we explain these steps in more detail.

2.5.1 PCA on Normalized Stream

Let $\Phi_Z^{\{1:p\}}$ be an $S_c \times p$ dimensional matrix that contains the top p principal components obtained from PCA on $\mathbf{Z}_{t,r}$. We remove the first component from those top p principal components based on our observation that the first component captures majority of the noise, while subsequent components contain information about movements of hands and fingers while typing. This happens because PCA ranks principal components in descending order of their variance, due to which the noisy components with higher variance gets ranked among top principal components. Due to correlated nature of variations in multiple CSI time series, the removal of this PCA component does not lead to any significant information loss as remaining PCA components still contain enough information required for successfully detecting starting and ending points of the keystrokes.

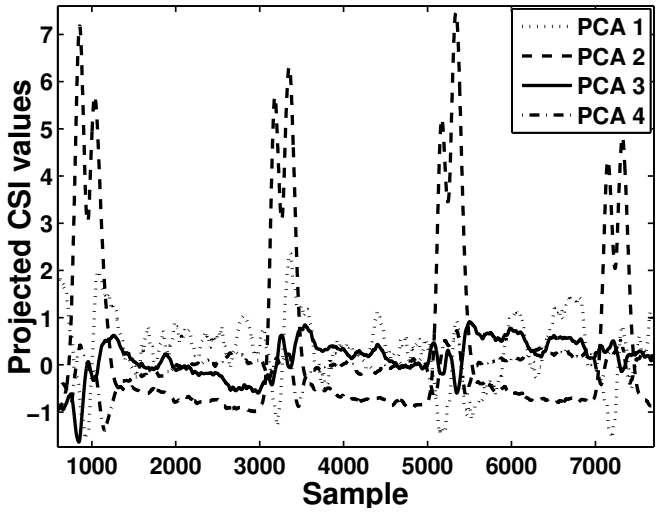
If we exclude the first component, the projection of the CSI stream $\mathbf{Z}_{t,r}$ of t-r transmit-receive antenna pair onto the remaining principal components $\Phi_Z^{\{2:p\}}$ can then be written as:

$$\mathbf{Z}_{t,r}^{\{2:p\}} = \mathbf{Z}_{t,r} \times \Phi_Z^{\{2:p\}} \tag{2.3}$$

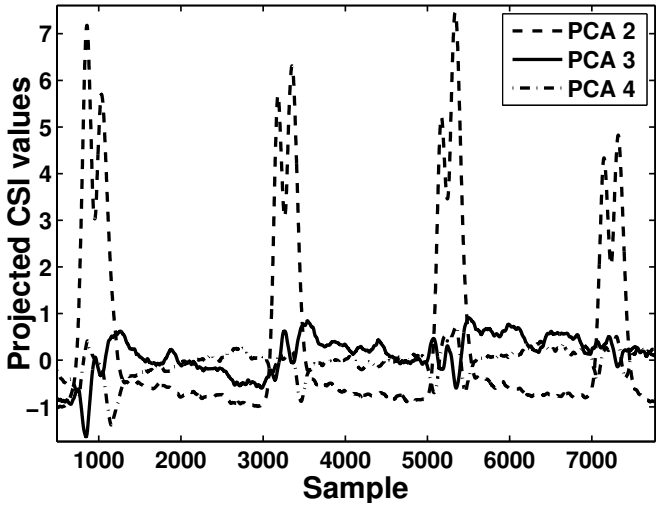
where $\mathbf{Z}_{t,r}^{\{2:p\}}$ is an $N \times (p-1)$ dimensional matrix containing the projected CSI streams in its columns. We choose the $p = 4$ in our implementation based on our observation that only top 4 principal components contained most significant variations in CSI values caused by different keystrokes. Figure 2.4(a) shows the result of projecting normalized CSI time series $\mathbf{Z}_{t,r}$ onto its top 4 principal components. We observe from Figure 2.4(b) that by removing the first principle component, we essentially remove the most noisy projection among the all 4 projections of $\mathbf{Z}_{t,r}$.

2.5.2 Keystroke Detection

Although existing DFAR schemes propose techniques to automatically detect the start and end of activities, they can not be directly adapted for use in detecting the start and end of keystrokes. Existing schemes use simple threshold based algorithms for detecting the start and end of activities. While, threshold based schemes work well for macro-movements, they are not well suited for micro-



(a) Top 4 projections



(b) Projections 2, 3 & 4

Figure 2.4: PCA of Z-normalized CSI stream $\mathbf{Z}_{t,r}$

movements such as those of hands and fingers while typing, where we need to precisely segment time series of keystrokes that are closely spaced in time. Unlike general purpose threshold based algorithms, we propose a keystroke detection algorithm that provides better detection accuracy, since it is strictly based on the experimentally observed shapes of different keystroke waveforms. The intuition behind our algorithm is that the CSI time series of every keystroke shows a typical increasing and decreasing trend in rates of change in CSI time series, similar to the one shown in Figure 2.2. To detect such increase and decrease in rates of change in CSI time series, our algorithm uses a moving window approach to detect the increasing and decreasing trends in rates of change

in all $p-1$ time series for each transmit-receive antenna pair *i.e.*, on each column of $\mathbf{Z}_{t,r}^{2:p}$. Our algorithm detects the starting and ending points of keystrokes in following six steps.

First, the algorithm calculates the mean absolute deviation (MAD) for each of the $p-1$ time series for each window of size W at j-th iteration. This is done primarily to detect the extent of variations in the values of a given time series. The main reason behind choosing MAD instead of variance is that in calculating, the deviations from the mean are squared which gives more weight to extreme values. In cases where a time series contains outliers, this results in undue weight given to those outlying values and that significantly corrupts the measure of deviation. The MAD is calculated using following equation.

$$\triangle m_j[k] = \frac{\sum_{i=j}^{j+W} |\mathbf{Z}_{t,r}^{\{k\}}(i) - \overline{\mathbf{Z}}_{t,r}^{\{k\}}(j : j+W)|}{W} \tag{2.4}$$

where $\overline{\mathbf{Z}}_{t,r}^{\{k\}}(j : j+W)$ represents the vector of means of the kth projected CSI stream in j-th window. It calculates the value of $\triangle m_j$ for each sample point j and for the principle components $2 \leq k \leq p$.

Second, the algorithm adds the mean absolute deviations in each waveform to calculate a combined measure $\triangle M_j$ of MAD in all $p-1$ waveforms, which is calculated in the following equation.

$$\triangle M_j = \sum_{k=2}^{p} \triangle m_j[k] \tag{2.5}$$

Third, the algorithm compares $\triangle M_j$ to a heuristically set threshold $Thresh$. Let $\delta_j = \triangle M_j - Thresh$, then $\delta_j > 0$ shows that the current window j contains significant variations in CSI amplitudes.

Fourth, the algorithm compares δ_j to its value in last window δ_{j-1} to detect increasing or decreasing trend in detected variations. When $\delta_j - \delta_{j-1} > 0$, there is an increasing trend in the rate of change in combined MAD ($\triangle M_j$) of CSI time series and vice versa. These increasing and decreasing trends are captured in variables i_u and d_u, respectively. The algorithm increments the value of i_u by 1 whenever $\delta_j - \delta_{j-1} > 0$ and d_u by 1 whenever $\delta_j - \delta_{j-1} < 0$. Let σ represent *forgetting factor*, which is used to "forget" the variations caused by noise to avoid false positives.

To forget such variations, the algorithm decrements both i_u and d_u by 1 if $\triangle M_j < Thresh$ for a duration of σW.

Fifth, as soon as the values of i_u and d_u exceed empirically determined thresholds I_u and D_u, respectively, the algorithm detects the start of the keystroke. As soon as the algorithm detects a keystroke, it estimates the starting point s_m and ending point e_m of the keystroke waveforms using following equations.

$$s_m = j - \beta W - B_{left} \tag{2.6}$$

$$e_m = j - \beta W + \mathrm{t}_{avg} + B_{right} \tag{2.7}$$

where t_{avg} is the average number of data points spanned by waveforms of different keystrokes, β is the *span factor* which determines the estimated starting point of the keystroke and B_{left} and B_{right} are *guard* intervals on both sides of the estimated keystroke interval. The guard intervals ensure that the detected keystroke waveforms are complete.

Last, our algorithm calculates the sum of powers in all waveforms lying within those starting and ending points and then compares this combined power with a *sum power threshold* (P_{avg}) to confirm the presence of a complete keystroke within that interval. This ensures that the training models are built using only those waveforms which contain complete shapes of the keystrokes. Once keystroke detection is confirmed, the algorithm finally returns the starting point (s_m) of the detected keystroke and jumps $\triangle \mathrm{t}_{avg}$ data points ahead of s_m to look for next keystroke, where $\triangle \mathrm{t}_{avg}$ is the average number of data points between arrival of two consecutive keystrokes. From the CSI data set we collected from our volunteers, we observed that on average the waveforms of a keystroke spanned $\mathrm{t}_{avg} \approx 650$ data points and average number of data points between arrival of two consecutive keystrokes was $\triangle \mathrm{t}_{avg} \approx 1250$ data points at the CSI sampling rate of $F_s = 2500$ samples/s. We empirically determined appropriate values for the remaining constants including W, D_u, I_u, σ, β, B_{left}, B_{right}, $Thresh$ and P_{avg}, as described in Algorithm 1.

Algorithm 1: Keystroke Detection from single TX-RX stream

initialize $\mathbf{S}_{t,r}$ /*Start indices of keystrokes*/
initialize $W = 200$ /*Window Size*/
initialize $Thresh = 0.8$ /*Threshold*/
initialize $P_{avg} = 0.35$ /*Sum Avg. Pwr Threshold*/
initialize i_u /*Duration of increasing trend*/
initialize d_u /*Duration of decreasing trend*/
initialize $I_u = 100$ /*Avg. duration of increasing trend*/
initialize $D_u = 100$ /*Avg. duration of decreasing trend*/
initialize C /*Counts repetitive insignificant changes*/
initialize K_{start} /*Flags possible start of a keystroke*/
initialize $j = 0$ /*Iteration count*/
initialize $B_{left} = W$
initialize $B_{right} = \mathrm{t}_{avg} + W$
initialize $\sigma = 0.1$ /*Forgetting factor*/
initialize $\beta = 1.5$ /*Keystroke span factor*/
while current window is within {1:length($\mathcal{Z}_{t,r}^{\{2:p\}}$)} **do**
    $\triangle m_j[k] \leftarrow \frac{\sum_{i=j}^{j+W} |\mathbf{Z}_{t,r}^{\{k\}}(i) - \overline{\mathbf{Z}}_{t,r}^{\{k\}}(j{:}j+W)|}{W}$
    $\triangle M_j \leftarrow \sum_{k=2}^{p} \triangle m_j[k]$
    if $(j = 0)$ **then** $\delta_j \leftarrow \triangle M_j - Thresh$
    else
        if $\delta_j \geq 0$ **then**
            $K_{start} \leftarrow 1$
            if $(\delta_j - \delta_{j-1}) < 0$ **then** $d_u = d_u + 1$
            else $i_u = i_u + 1$
            end if
            if $(d_u > D_u)$ & $(i_u > I_u)$ **then**
                $O \leftarrow \mathcal{Z}_{t,r}^{\{2:p\}}(j - floor(\beta W) - B_{left} : j + floor(\beta W) + B_{right})$
                $p \leftarrow sum(mean(O^2))$
                if $p > Pavg$ **then**
                $\mathbf{S}_{t,r} \leftarrow \mathbf{S}_{t,r} \cup (j - floor(\beta W) - B_{left})$
                $i_u \leftarrow 0, d_u \leftarrow 0$
                $j \leftarrow j + \triangle \mathrm{t}_{avg}$
                end if
            end if
        else
        /*Removing spurious counts of i_u and d_u if insignificant changes detected in waveforms*/
            if $(K_{start} = 0)$ **then**
                $C \leftarrow C + 1$
                if $(C > \sigma W)$ **then**
                    if $i_u > 0$ **then** $i_u \leftarrow i_u - 1$
                    end if

```
                if d_u > 0 then d_u ← d_u − 1
                end if
            end if
        else
            K_start ← 0
        end if
    end if
    δ_j ← △M_j − Thresh
  end if
  j ← j + 1
end while
Return S_{t,r}
```

2.5.3 Combining Results from Antenna Pairs

As mentioned earlier, we obtain the starting points of keystrokes independently from each TX-RX antenna pair. Let $\mathbf{S}_{t,r}$ represent the set containing the starting points of all keystrokes obtained from the keystroke detection algorithm applied on the antenna pair $t - r$. First, we obtain the set $\mathbf{S}_{t,r}$ for each $t - r$ pair. Second, we take the average of all the starting points that are within $\triangle \mathrm{t}_{avg}$ of each other in all sets $S_{t,r}$ to obtain a robust estimate of starting points of keystrokes. Third, based on experimentally measured average span t_{avg} of different keystrokes, we calculate the ending points of all keystrokes by simply adding t_{avg} to the corresponding starting point.

2.5.4 Extracting Keystroke Waveforms

Once the algorithm calculates the set of starting and corresponding ending points for keystrokes, we use those points to extract the waveforms from CSI matrix $\mathbf{H}_{t,r}$. Let $\mathbf{K}_{m,t,r}$ represent the CSI waveform of m^{th} keystroke extracted from the antenna pair t-r. Let $\overline{s}_m$ represent the average of the starting points for the m^{th} keystroke from all antenna pairs. We can express $\mathbf{K}_{m,t,r}$ in terms of $\mathbf{H}_{t,r}$ follows.

$$\mathbf{K}_{m,t,r} = \mathbf{H}_{t,r}(\overline{s}_m : \overline{s}_m + \mathrm{t}_{avg}) \tag{2.8}$$

After extracting the CSI waveforms $\mathbf{K}_{m,t,r}$ from all subcarriers of the t-r antenna pair, we apply PCA on those CSI waveforms to remove the noisy components and obtain the components that represent the variations caused by movements of hands and fingers.

Unlike principle components derived from normalized streams, it is difficult to decide which PCA component represents noise and should be removed from the top p principal components for the case of $\mathbf{K}_{m,t,r}$. The difficulty arises because $\mathbf{K}_{m,t,r}$ contains the set of waveforms for a specific keystroke instead of the whole CSI stream, due to which the variance of noisy component often becomes small. We observe that the noisy PCA component keeps changing positions between 1^{st} and 2^{nd} place among the sorted PCA components for different extracted keystroke waveforms. In order to get rid of this problem, we first project $\mathbf{K}_{m,t,r}$ onto all top q principal components. Let $\Phi_K^{\{1:q\}}$ be an $S_c \times q$ dimensional matrix that represent the top q principal components in $\mathbf{K}_{m,t,r}$ obtained after applying PCA and $\mathbf{K}_{m,t,r}^{\{1:q\}}$ be an $L \times q$ dimensional matrix containing the projected CSI streams in its columns, where L is the length of segmented keystroke waveform. Thus, $\mathbf{K}_{m,t,r}^{\{1:q\}}$ is given by the following equation.

$$\mathbf{K}_{m,t,r}^{\{1:q\}} = \mathbf{K}_{m,t,r} \times \Phi_K^{\{1:q\}} \tag{2.9}$$

In our implementation, we choose $q = 4$. This choice is again based on the observation that the top 4 principal components contain enough information about keystrokes required to achieve high accuracy during classification.

To detect which waveform in $\mathbf{K}_{m,t,r}^{\{1:q\}}$ represents the noisy projection, we chose the top 2 projected waveforms and divide each of them into R bins and calculate the variances in those bins. We then compare the variances calculated for different bins of one waveform with the corresponding bins of the other waveform. The waveform that has larger number of higher variance bins is considered to be the noisy projection, which we remove from $\mathbf{K}_{m,t,r}^{\{1:q\}}$ to finally get $q-1$ waveforms. Here we leverage the fact that although overall variance of a noisy projection may be smaller than the variance of other waveforms, but if the waveform is divided into appropriate number of smaller bins then the number of bins in which the variance of the noisy projection is higher than the corresponding bins of other waveforms is always larger. This is because the impact of noise is more dominant in smaller time windows compared to larger time windows. We used R=10 in our implementation of WiKey.

PCA can lead to different ordering of principal components in waveforms of different keystrokes of the same key, because the ordering of waveforms done by PCA is based solely on the value of their

variance, which can change even if a key is pressed in a slightly different way. This is problematic because to recognize the keystrokes, we need to compare the projections of an unseen key with the corresponding projections of the keys in the training data. In order to minimize the possibility of reordering, we order the projected keystroke waveforms in descending order of their peak to peak values before using the waveforms for feature extraction and classifier training.

2.6 Feature Extraction

To differentiate between keystrokes, we need to extract features that can uniquely represent those keystrokes. As different keys on a keyboard are closely placed, standard features such as maximum peak power, mean amplitude, root mean square deviation of signal amplitude, second/third central moments, rate of change, signal energy or entropy, and number of zero crossings cannot be used because adjacent keys give almost the same values for these features. Tables 2.1, 2.2, 2.3 and 2.4 show means and variances of some of these features calculated for 2^{nd} waveform in the extracted CSI-waveforms for keystrokes of alphabetic keys pressed by a users. It can be observed that the values of these features for different keys (for example 'c' and 'd') come out to be very similar. Looking at the means calculated for features like energy and number of zero crossings in Tables 2.1 and 2.2, it seems that they have different values for different keys. But as we observe from Tables 2.3 and 2.4, the variance of those features is high. Due to the reasons above, it becomes infeasible to use these features for keystroke classification. Frequency analysis is also not feasible because the frequency components in keystrokes of many different keys are similar. Another reason behind inapplicability of frequency domain analysis is that they lead to complete loss of time domain information.

From our data set, we have observed that although the frequency components in most keys are similar, they occur at different time instants for different keys. Therefore, we use shapes of the extracted keystroke waveforms as their features because the shapes retain both time and frequency domain information of the waveforms and are thus more suited for use in classification. We observed experimentally that the shapes of different keystroke waveforms were quite different from each other,

Table 2.1: Average values of features extracted from keystrokes of keys collected from user 10

Features	a	b	c	d	e	f	g	h	i	j	k	l	m
Mean amplitude	-0	-0.04	0.0124	-0.03	0.045	-0.043	-0.076	-0.06	0.014	-0.03	0.03	-0.01	-0
Second central moment	0.08	0.133	0.0801	0.083	0.156	0.1818	0.6523	0.263	0.12	0.231	0.33	0.11	0.1
Third central moment	0.02	-0.03	0.0036	-0.01	0.029	-0.06	-0.919	-0.05	-0.01	-0.1	0.05	0.02	-0
RMS deviation	0.27	0.359	0.2782	0.285	0.385	0.4244	0.7899	0.506	0.332	0.472	0.57	0.32	0.3
Energy	71.5	116.6	69.788	73.34	137.5	159.43	570.8	232.1	104.8	201.4	288	95.2	83.7
Entropy	9.76	9.762	9.7616	9.762	9.762	9.7616	9.7616	9.762	9.762	9.762	9.76	9.76	9.76
Zero Crossings	11.8	6.913	12.363	6.225	6.4	4.375	4.075	3.4	12.08	9.088	6.05	13.7	10

Table 2.2: Average values of features extracted from keystrokes of keys collected from user 10

Features	n	o	p	q	r	s	t	u	v	w	x	y	z
Mean amplitude	0.032	0.02	0.03	-0.012	0.008	0.054	7E-04	-0.013	-0.02	-0	-0.1	-0.02	0.06
Second central moment	0.108	0.09	0.19	0.1022	0.051	0.245	0.192	0.062	0.12	0.097	0.26	0.09	0.21
Third central moment	0.01	0.01	0.04	-0.006	0.003	0.098	-0.101	-0.01	0.029	0.023	-0	-0.02	0.04
RMS deviation	0.323	0.29	0.43	0.3137	0.222	0.472	0.434	0.242	0.335	0.306	0.5	0.3	0.45
Energy	94.98	75.6	167	88.928	44.22	215.5	167.1	54.56	104.4	84.48	227	81.5	182
Entropy	9.762	9.76	9.76	9.7616	9.762	9.762	9.762	9.762	9.762	9.762	9.76	9.76	9.76
Zero Crossings	9.063	13.8	12.9	11.85	15.41	6.35	12.85	16.75	11.88	14.3	6.48	10.1	7.55

Table 2.3: Variance of different features extracted from keystrokes of keys collected from user 10

Features	a	b	c	d	e	f	g	h	i	j	k	l	m
Mean amplitude	0.00029	4E-04	0.0003	1E-04	4E-04	0.0002	0.0003	8E-04	5E-04	5E-04	5E-04	2E-04	5E-04
Second central moment	0.00513	0.003	0.0011	0.001	0.007	0.0028	0.1008	0.012	0.005	0.009	0.016	0.006	0.002
Third central moment	0.00155	9E-04	0.0001	2E-04	0.002	0.0033	0.7021	0.002	0.001	0.007	0.009	0.017	5E-04
RMS deviation	0.0108	0.006	0.0031	0.004	0.011	0.0038	0.0348	0.011	0.011	0.01	0.012	0.009	0.006
Energy	3874.59	2283	816.91	912.4	5204	2160	76863	9315	3925	6846	12094	4883	1679
Entropy	0	0	0	0	0	0	0	0	0	0	0	0	0
Zero Crossings	26.3859	12.36	33.196	12.94	9.433	6.2627	3.9943	3.585	44.91	13.14	12.58	31.14	28.51

as shown by Figure 2.5(a) and 2.5(b).

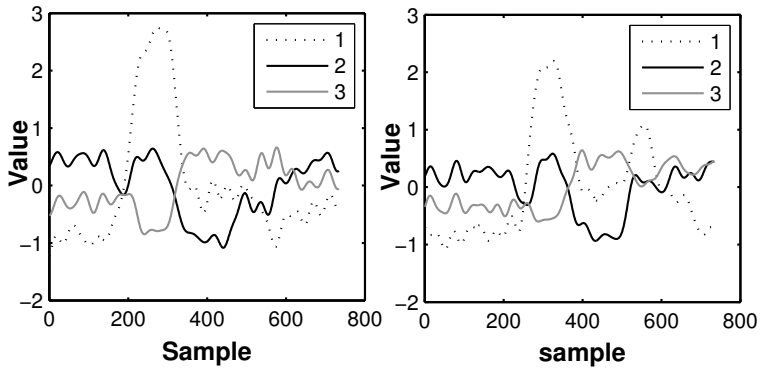
Directly using the extracted keystroke waveforms as keystroke features leads to high computational costs in the classification process because waveforms contain hundreds of data points per keystroke. Therefore, we apply Discrete Wavelet Transform (DWT) to compress the extracted keystroke waveforms while preserving most of the time and frequency domain information.

The DWT of a discrete signal $y[n]$ can be written in terms of wavelet basis functions as:

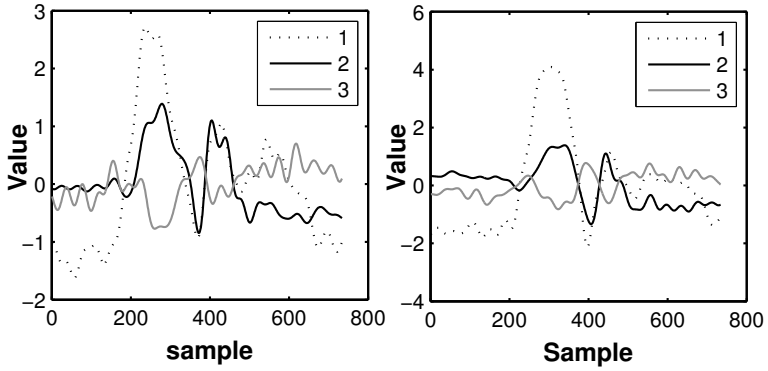
$$y[n] = \frac{1}{\sqrt{L}} \sum_{k} \lambda(j_0, k)\varphi_{j_0,k}(n) + \frac{1}{\sqrt{L}} \sum_{j=j_0}^{\infty} \sum_{k} \gamma(j, k)\psi_{j,k}(n)$$

where L represents the length of signal $y[n]$. The functions $\varphi_{j,k}(n)$ are called *scaling functions*, where as the corresponding coefficients $\lambda(j, k)$ are known as *scaling* or *approximation coefficients*. Similarly, the functions $\psi_{j,k}(n)$ are known as *wavelet functions* and the corresponding coefficients $\gamma(j, k)$ are known as *wavelet* or *detail coefficients*. To calculate approximation and detail coefficients, the scaling and wavelet functions are chosen such that they are orthonormal to each other. Thus, the following condition holds.

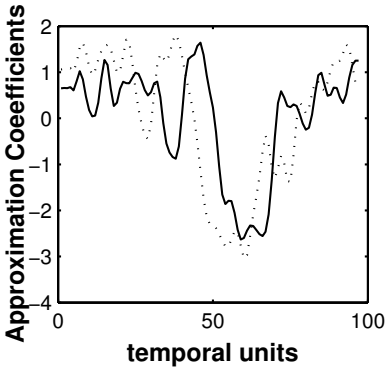
$$\langle \psi_{j,k}(n), \varphi_{j_0,m}(n) \rangle = \delta_{j,j_0}\delta_{k,m}$$



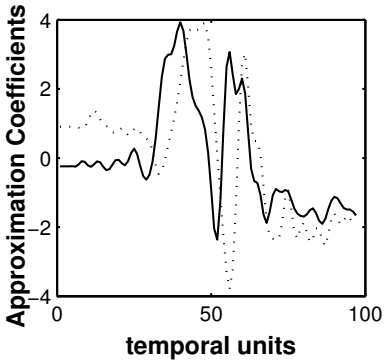
(a) Keystroke waveforms for key **i**



(b) Keystroke waveforms for key **o**



(c) DWT features of **i**



(d) DWT features of **o**

Figure 2.5: Feature extraction from 2^{nd} keystroke waveforms extracted from TX-1, RX-1 for I & O

Based on the condition above, the approximation and detail coefficients calculated for j-th scale can then be written as:

$$\lambda(j,k) = \langle y[n], \varphi_{j+1,k}(n) \rangle = \frac{1}{\sqrt{L}} \sum_{n} y[n] \varphi_{j+1,k}(n)$$

Table 2.4: Variance of different features extracted from keystrokes of keys collected from user 10

Features	n	o	p	q	r	s	t	u	v	w	x	y	z
Mean amplitude	3E-04	3E-04	5E-04	0.0003	0.00018	6E-04	4E-04	3E-04	1E-04	3E-04	4E-04	6E-04	4E-04
Second central moment	0.003	0.002	0.007	0.0017	0.00041	0.03	0.003	0.001	0.006	0.002	0.007	0.003	0.005
Third central moment	3E-04	5E-04	0.003	0.0003	7.70E-05	0.024	0.003	1E-04	0.015	9E-04	0.001	6E-04	0.003
RMS deviation	0.005	0.004	0.008	0.0042	0.00196	0.026	0.004	0.004	0.008	0.004	0.007	0.007	0.007
Energy	2153	1150	5048	1296.2	308.95	23201	2181	886.9	4714	1403	5166	2100	4147
Entropy	0	0	0	0	0	0	0	0	0	0	0	0	0
Zero Crossings	15.25	29.24	24.09	21.673	17.3847	12.21	17.7	36.85	21.63	27.71	13.67	31.49	6.529

$$\gamma(j,k) = \langle y[n], \psi_{j+1,k}(n) \rangle = \frac{1}{\sqrt{L}} \sum_{n} y[n] \psi_{j+1,k}(n)$$

To achieve desired compression using DWT, we need to select appropriate wavelet and scaling filters. We tested the accuracy of our classifier using two different wavelet filters: Daubechies and Symlets. We choose Daubechies D4 (four coefficients per filter) wavelet and scaling filters because the models trained with the DWT features extracted using these filters achieved higher classification accuracy. For each keystroke, we perform DWT 3 times on each one of its $(q-1) = 3$ waveforms, which is achieved by applying DWT on the approximation coefficients obtained from the previous steps. We choose to apply DWT 3 times because this preserves enough details of those waveforms required for successful classification while achieving maximum compression. WiKey uses only the approximation coefficients as keystroke features and discards the detail coefficients because approximation coefficients alone result in good classification accuracy. Therefore, we have $3 \times M_T \times M_R$ keystroke shapes for every keystroke, *i.e.* the approximation coefficients of all 3 waveforms extracted from the CSI time series in each TX-RX antenna pair, where M_T is the number of transmitting antennas and M_R is the number of receiving antennas. Figures 2.5(a) through 2.5(d) show feature extraction procedure performed on the 2^{nd} keystroke waveforms for keys 'i' and 'o', extracted from TX-1, RX-1 antenna pair.

2.7 Classification

After obtaining DWT based shape features of keystrokes, WiKey builds training models for classification using them. As WiKey needs to compare shape features of different keystrokes, we need a comparison metric that provides an effective measure of similarity between shape features of two keystrokes. WiKey uses a well-known method called *dynamic time warping* (DTW) that calculates the distance between waveforms by performing optimal alignment between them.

Using DTW distance as the comparison metric between keystroke shape features, WiKey trains an ensemble of *k-nearest neighbour* (kNN) classifiers using those features from all TX-RX antenna pairs. WiKey obtains decisions from each classifier in the ensemble and uses majority voting to obtain final result. Next, we first explain how we apply DTW on the keystroke shape features and then explain how we train the ensemble of classifiers.

2.7.1 Dynamic Time Warping

DTW is a dynamic programming based solution for obtaining minimum distance alignment between any two waveforms. DTW can handle waveforms of different lengths and allows a non-linear mapping of one waveform to another by minimizing the distance between the two. In contrast to Euclidean distance, DTW gives us intuitive distance between two waveforms by determining minimum distance warping path between them even if they are distorted or shifted versions of each other. *DTW distance* is the Euclidean distance of the optimal warping path between two waveforms calculated under boundary conditions and local path constraints [92]. In our experiments, DTW distance proves to be very effective metric for comparing two shape features of different keystrokes. WiKey uses the open source implementation of DTW in the Machine Learning Toolbox (MLT) by Jang [59]. WiKey uses local path constraints of 27, 45, and 63 degrees while determining minimum cost warping path between two waveforms. For the features extracted for keys ‘i’ and ‘o’ shown in figures 2.5(a) and 2.5(b), the DTW distance among features of key ‘i’ was 18.79 and the DTW distance among features of key ‘o’ was 19.44. However, the average DTW distance between features of these keys was 44.2.

2.7.2 Classifier Training

To maximize the advantage of having multiple shape features per keystroke obtained from multiple transmit-receive antenna pairs, we build separate classifiers for each of those shape features. We build an ensemble of $3 \times M_T \times M_R$ classifiers using kNN classification scheme. WiKey requires the user to provide training data for the keystrokes to be recognized and each classifier is trained

using the corresponding features extracted from CSI time series from all TX-RX antenna pairs. To classify a detected keystroke, WiKey feeds the shape features of that keystroke to their corresponding kNN classifiers and obtains a decision from each classifier in the ensemble. Each kNN classifier searches for the majority class label among k nearest neighbors of the corresponding shape feature using DTW distance metric. WiKey calculates the final result through majority voting on the decisions of all kNN classifiers in the ensemble.

2.7.3 Behavioral Clustering of User Data

In this section, we explain how we use the relative consistency of multiple training samples to reduce the computational complexity of WiKey's classification process in real-world experimental scenarios, where users type sentences, consisting of multiple different types of keystrokes. We observed from our experiments that every user has several distinct styles of pressing each key. We also observed that these distinct styles of pressing the keys remained consistent over time for all users. To utilize this consistency during classification, WiKey applies hierarchical clustering with Ward's linkage [149] on the training samples. WiKey uses DTW distance as comparison metric during this behavioral clustering step. It also provides the expected number of clusters C to the hierarchical clustering algorithm. After dividing the training samples into C clusters, WiKey randomly picks $\mathcal{P}$ percent of samples to train the KNN classifiers. Although behavioral clustering increases the classification speed, it slightly reduces the overall accuracies. We discuss this trade-off between speed and accuracy in Section 2.8.5 when evaluating WiKey's performance on recognizing sentences in real world typing scenario.

2.8 Implementation & Evaluation

2.8.1 Hardware Setup

We implemented our scheme using off-the-shelf hardware devices. Specifically, we use a Lenovo X200 laptop with Intel Link 5300 WiFi NIC as the receiver that connects to the three antennas of the X200 laptop. The X200 laptop has 2.26GHz Intel Core 2 Duo processor with 4GB of RAM and

Ubuntu 14.04 as its operating system. We used TP-Link TL-WR1043ND WiFi router as transmitter operating in 802.11n AP mode at 2.4GHz. We collect the CSI values from the Intel 5300 NIC using a modified driver developed by Halperin *et al.* [47]. The transmitter has 2 antennas and the receiver has 3 antennas, *i.e.*, $M_T = 2$ and $M_R = 3$. This gives $3 \times 2 \times 3 = 18$ classification models for each key in our evaluations.

We place the X200 laptop at a distance of 30 cm from the keyboard such that the back side of its screen faces the keyboard on which the users type and its screen is within the line-of-sight (LOS) of the WiFi router it is connected to. The distance of WiFi router from the target keyboard is 4 meters. The CSI values are measured on ICMP *ping* packets sent from the WiFi router, *i.e.*, the TP-Link TL-WR1043ND, to the laptop at high data rate of about 2500 packets/s. Setting a higher *ping* frequency leads to higher sampling rate of CSI, which ensures that the time resolution of the CSI values is high enough for capturing maximum details of different type of keystrokes.

2.8.2 Data Collection

To evaluate the accuracy of WiKey, we collected training and testing dataset from 10 users. These 10 users were general university students who volunteered for the experiments and only 2 out of them had some know how of wireless communication. Users 1–9 first provided 30 samples for each of the 37 keys (26 alphabets, 10 digits and 1 space bar) by pressing that key multiple times. After this, these users typed the sentence S_1 = "the quick brown fox jumped over the lazy dog" two times, without spaces.

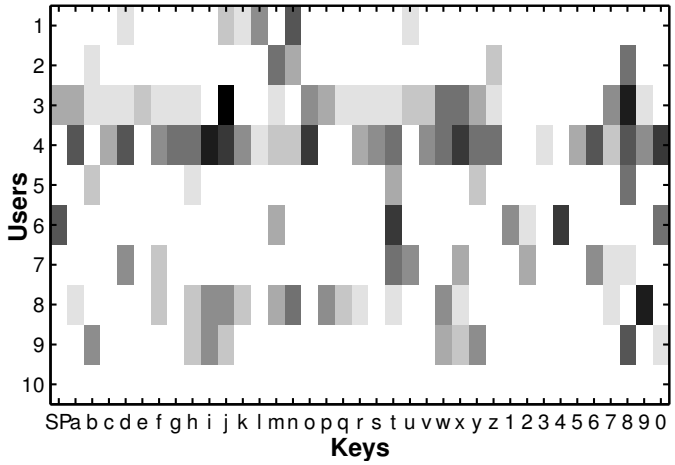
To evaluate how the number of training samples impact the accuracy, we collected 80 samples for each of the 37 keys from User 10. Afterwards, this user typed each of the following sentences 5 times, without spaces: S_1 ="the quick brown fox jumps over the lazy dog", S_2 = "nobody knew why the candles blew out", S_3 = "the autumn leaves look like golden snow", S_4 = "nothing is as profound as the imagination" and S_5 = "my small pet mouse escaped from his cage". We asked users to type naturally with multiple fingers but only press one key at a time while keeping the average keystroke inter-arrival time at 1 second. After recording the CSI time series for each of the

above experiments, we first applied our keystroke extraction algorithm on those recorded CSI time series to extract the CSI waveforms for individual keys and then extracted the DWT based shape features from each of the extracted keystroke waveforms.

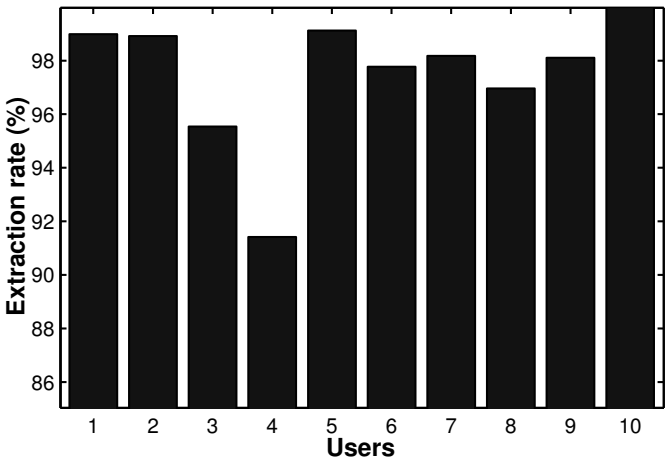
2.8.3 Keystroke Extraction Accuracy

We evaluate the accuracy of our keystroke extraction algorithm in terms of the detection ratio, which is defined as the total number of correctly detected keystrokes in a CSI time series divided by the total number of actual keystrokes. *The detection ratio of our proposed algorithm is more than 97.5%.* Figure 2.6(a) shows the color map showing the percentage of the missed keystrokes of all 37 keys for all 10 users.

The darker areas represent higher rate of missed keystrokes. We can observe from this figure that the number of missed keystrokes vary for different individuals depending upon their typing behaviors. For example we observed that the keystrokes of user 4 were missed in higher percentage with average detection ratio of 91.8% whereas the keystrokes of user 10 were not missed at all with average detection ratio of 100% calculated over all 37 keys. The lower extraction accuracy for user 4 shows that more keystrokes were missed, which is due to the significant difference in his typing behavior compared to other users. The accuracy of our scheme for such a user can be increased significantly by tuning the parameters of our algorithm for the given user. We also observe from this figure that the keystrokes that are missed are usually those for which fingers move very little when typing. For example, in pressing keys ‘a’, ‘d’, ‘f’, ‘i’, ‘j’ and ‘x’ the hands and fingers move very little, and thus the variations in the CSI values sometimes go undetected. Figure 2.6(b) shows the keystroke extraction rate for each user averaged over all 37 keys. The experimental results show that our keystroke extraction algorithm is robust because it consistently achieves high performance over different users without requiring any user specific tuning of system parameters.



(a) Colormap for missed keys



(b) Keystroke extraction rates per (averaged over all keys)

Figure 2.6: Keystroke extraction results

2.8.4 Classification Accuracy

We evaluate the classification accuracy of WiKey through two sets of experiments. In the first set of experiments, we build classifiers for each of the 10 users using 30 samples and measure the 10-fold cross validation accuracy of those classifiers. In the second set of experiments, we build classifier for user 10 while increasing the number of samples from 30 to 80 in order to observe the impact of increase in the number of training samples on the classification accuracy. Cross validation automatically picks a part of data for training and remaining for testing and does not use any data in testing that was used in training. Recall that the WiKey uses kNN classifiers for recognizing keys. In all our experiments, we set $k = 5$.

2.8.4.1 Accuracy with 30 Samples per Key

We evaluate the classification accuracy of WiKey in terms of average accuracy per key and average accuracy on all keys of any given user. We also present confusion matrices resulting from our experiments. A confusion matrix tells us which key was recognized by WiKey as which key with what percentage. We calculate the average accuracy per key by taking the average of confusion matrices obtained from all users and average accuracy on all keys of any given user by averaging the accuracy on all keys within the confusion matrix of that user. For each user, we trained each classifier using features from 30 samples of each key. We conducted experiments on all 37 keys as well as on only 26 alphabet keys and performed 10-fold cross validation to obtain the confusion matrices.

WiKey achieves an overall keystroke recognition accuracy of 82.87% in case of 37 keys and 83.46% in case of 26 alphabetic keys when averaged over all keys and users. Figure 2.7 shows the recognition accuracy for each key across all users for the 26 alphabetic keys. Similarly, Figure 2.8 shows the recognition accuracy for each key across all users for all 37 keys. Figure 2.9 shows the average recognition accuracy achieved by each user for both 26 keys and 37 keys. We observe that the recognition accuracy for 26 alphabetic keys is on average greater than the recognition accuracy for the all 37 keys. This is because the keystroke waveforms of the digit keys (0-9) often show similarity with keystroke waveforms of alphabet keys in the keyboard row staring with QWE, which leads to slightly greater number of misclassifications.

2.8.4.2 Accuracy vs. the Size for Training Set

To determine the impact of the number for training samples on the accuracy of WiKey, we again perform two sets of experiments: one for 26 alphabetic keys and other for all 37 keys.

The accuracy of WiKey increases when the number of training samples per key are increased from 30 to 80. Figure 2.10 shows the results from 10-fold cross validation for the 26 alphabetic keys when 80 training samples are used per key. We observe from this figure that the recognition accuracy increased from 88.3% (as seen in Figure 2.9) to 96.4% when the number of training

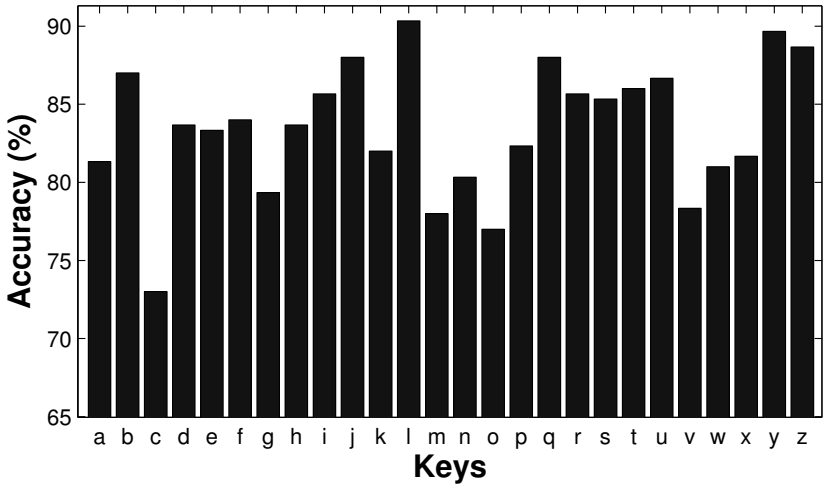


Figure 2.7: Mean accuracy for keys A-Z (Users 1-10)

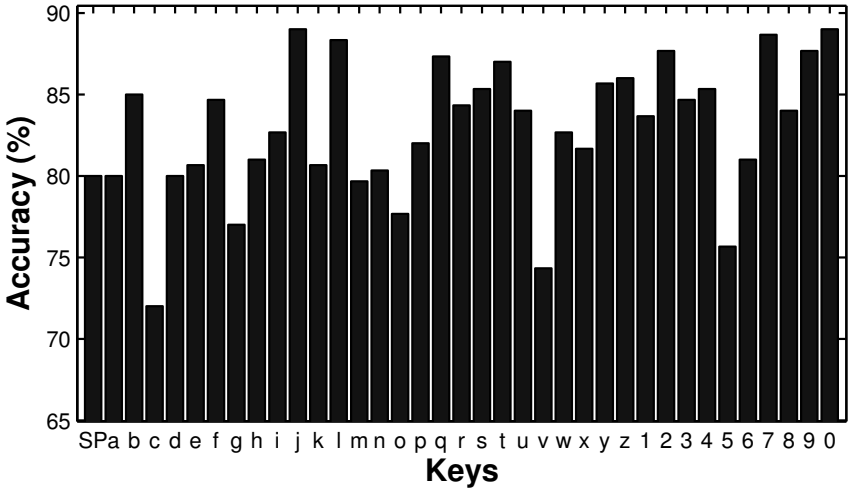


Figure 2.8: Mean accuracy for all 37 keys (Users 1-10)

samples are increased from 30 to 80. Figure 2.11 shows the results from 10-fold cross validation for all 37 keys when 80 training samples are used per key. We again observe that the recognition accuracy increased from 85.95% (as seen in Figure 2.9) to 89.7% when the number of training samples are increased from 30 to 80. The gray-scale maps of the confusion matrix obtained after 10-fold cross-validation on 80 training samples of User 10 is shown in Figure 2.12.

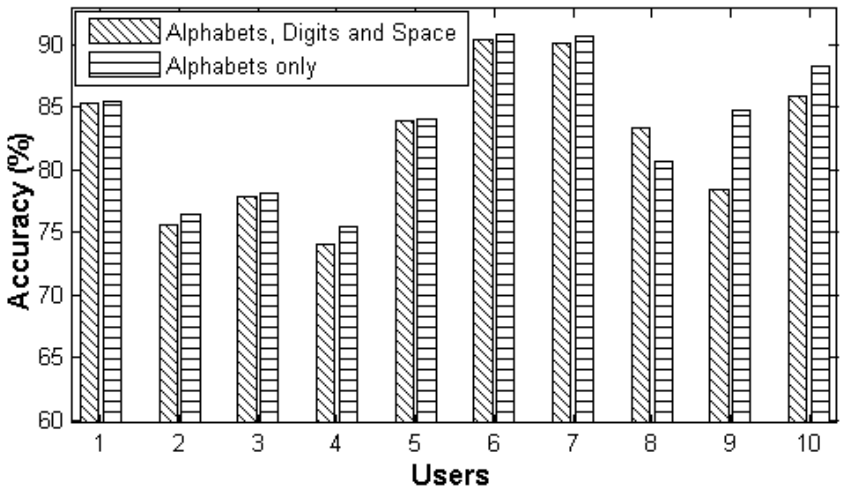


Figure 2.9: Per user average classifier accuracies

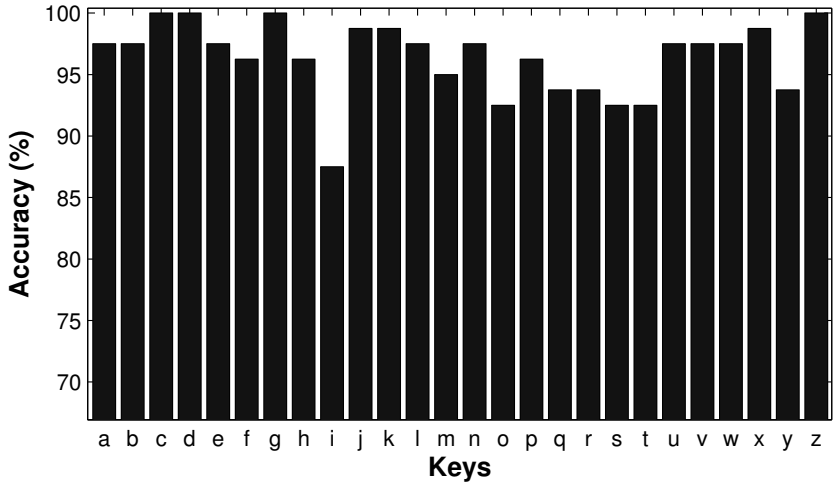


Figure 2.10: Accuracy for keys A-Z from user 10

2.8.4.3 Effects of CSI Sampling Rate and Training Samples

In previous experiments, we used high CSI sampling rate of 2500 samples/s. Furthermore, the 10-fold cross validation automatically chose 10% of the data for testing and remaining 90% for training. Next, we evaluate the effect of changing the CSI sampling rate and the percentage of data used for training on accuracy. To extract keystrokes, we halved the values used for W, D_u, I_u, B_{left}, and B_{right}. We performed $X-$fold cross validation ($2 \leq X \leq 10$) on the data obtained for alphabetic keys from user 10. Figure 2.13 plots the accuracies for number of folds varying from

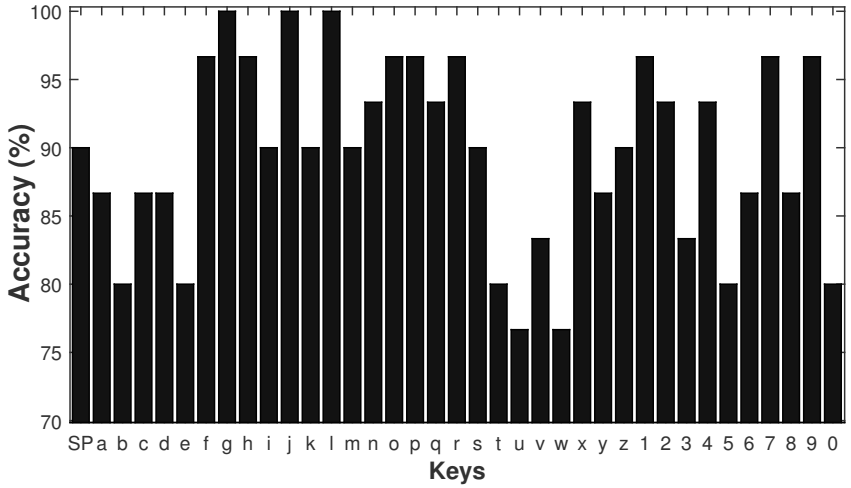


Figure 2.11: Accuracy for all 37 keys from user 10

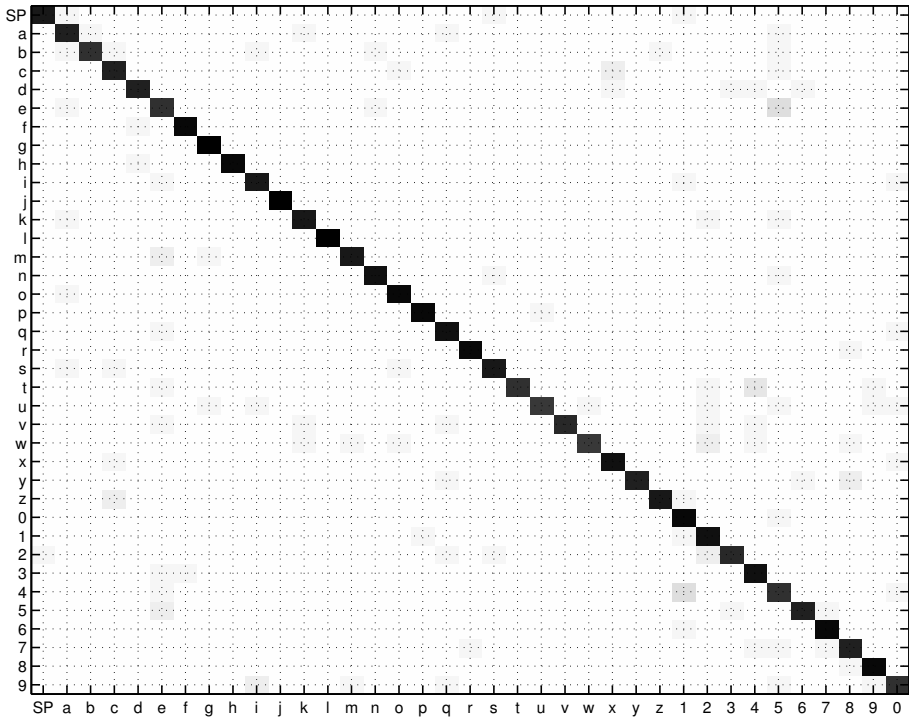


Figure 2.12: Color map of user 10's confusion matrix

2 to 10, where each plotted value if the average over all alphabet keys. We observe from Figure 2.13 that the accuracies dropped compared to previously achieved accuracy because of the drop in resolution of keystroke shapes due to reduced sampling rate. We also observe that recognition accuracies of the keys for which hands and fingers move little were affected the most. When 50% of data was used for training, *i.e.*, for 2-fold cross validation, the accuracies for keys 'j','x','v' and 'p' dropped below 60%. However, the average accuracy remained approximately 80% for all folds.

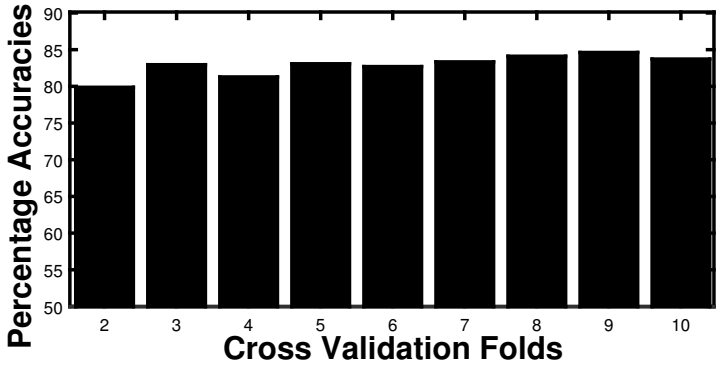


Figure 2.13: Multifold cross-validated average accuracies for user 10's lower resolution keystroke waveforms

2.8.5 Real-world Evaluation on Sentences

To evaluate WiKey in real world scenarios, we collected CSI data for different sentences typed by users 1 through 10, as mentioned in Section 2.8.2. To train the classifiers to recognize keystrokes in sentences, we used the same dataset of individual keystrokes that we used in the evaluations presented above. For the test samples, we used the the keystrokes extracted from datasets obtained from typing the sentences.

2.8.5.1 Accuracy

WiKey achieves an average keystroke recognition accuracy of 77.43% for typed sentences when 30 training samples per key were used. For each user, we trained classifiers using 30 samples for each of the 26 alphabetic keys. We then applied our keystroke extraction algorithm to first extract waveforms of individual keys, applied PCA on them to denoise the waveforms and then extracted the shape features for each extracted key and feeded them to the classifiers to recognize the keystrokes in the sentence. Figure 2.14 shows the keystroke recognition accuracy for the sentences typed by each user.

WiKey achieves an average keystroke recognition accuracy is 93.47% in continuously typed sentences with 80 training samples per key. We first trained classifiers using 80 samples for each of the 26 alphabetic keys and then fed them with keystrokes from typed sentences. Figure 2.15 shows the keystroke recognition accuracy for all the sentences (S_1 to S_5) typed by user 10. The average

keystroke recognition accuracy rate for user 10 in previous experiment, which used 30 samples for training classifiers was just 80%. Thus, we can conclude that increasing the number of training samples increases the accuracy of WiKey.

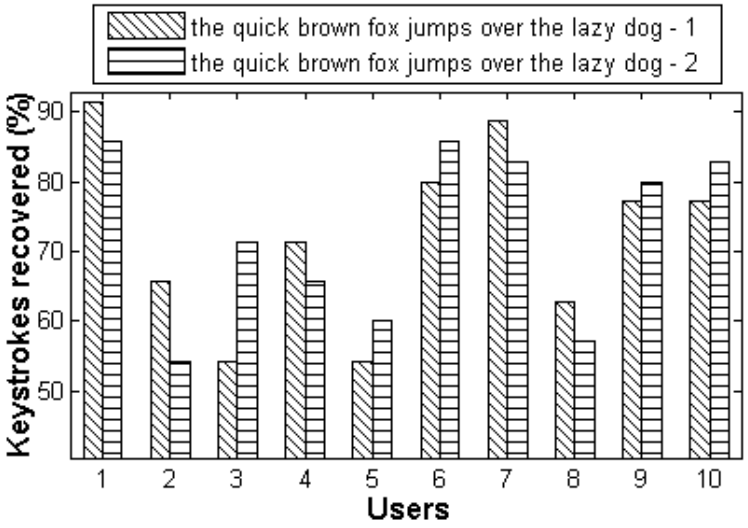


Figure 2.14: Keystroke recognition for sentences collected from all users using 30 samples per key

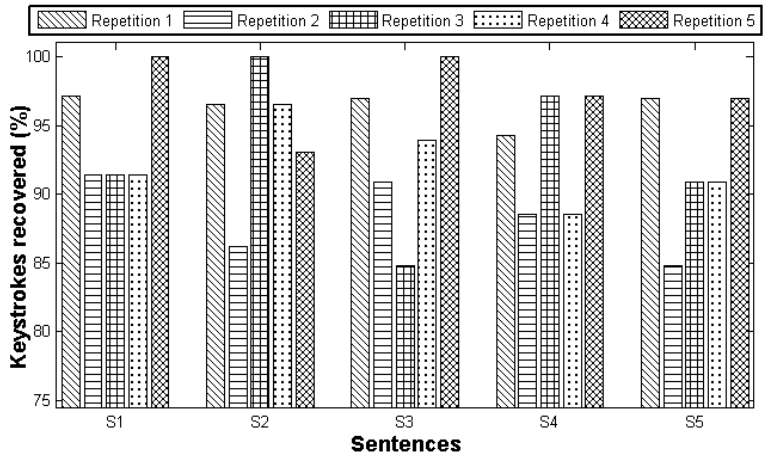


Figure 2.15: Keystroke recognition for sentences collected from user 10 using 80 samples per key

2.8.5.2 Effects of Behavioral Clustering

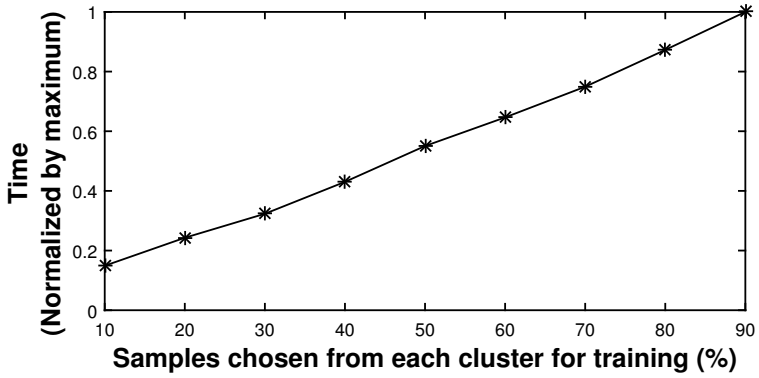
In this subsection, we show how behavioral clustering affects keystroke recognition time and accuracy of WiKey. As discussed in Section 2.7.3, every user has a set of distinct styles of pressing each key that all occur and stay consistent over time. WiKey leverages these consistent behaviors to

reduce computational complexity of KNN classifiers by reducing the number of training samples required to achieve high keystroke recognition accuracies.

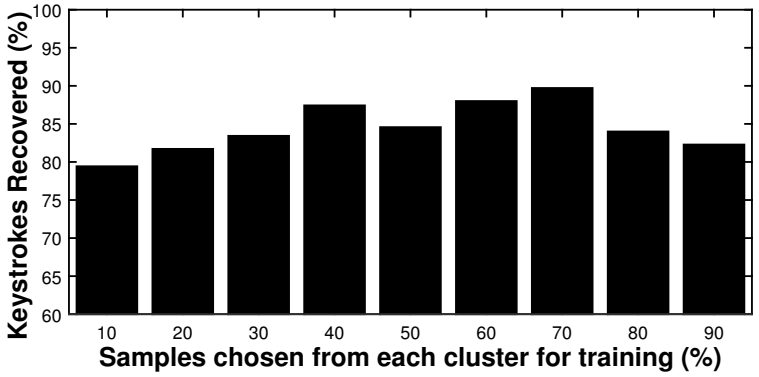
Figures 2.16(a) and 2.16(b) show the keystroke recognition accuracy and time, respectively, as we increase the percentage of samples ($\mathcal{P}$) that WiKey uses for training from each behavioral cluster. We obtained the results in these two figures on the keystrokes from the sentence S1 collected from user 10. Each value in these two figures is an average from the five repetitions of S1. We observe from Figure 2.16(a) that average recognition time increases consistently as the number of samples increase, which is intuitive because complexity of KNN classification (which relies on exhaustive search) increases with the increase in the number of training samples. At the same time, we also observe from Figure 2.16(b) that the keystroke recognition accuracy of WiKey suffers when using fewer training samples. Nonetheless, it still stays above 79%. Another interesting observation from Figure 2.16(b) is that the accuracy of WiKey increases when $\mathcal{P}$ increases from 10% up to 70%, which is intuitive, but it decreases beyond that. The reason behind this decrease is the inclusion of noisy and/or inconsistent samples into the training dataset when a large percentage of training samples is used for training. This observation implies that the behavioral clustering of training samples not only helps in decreasing the training time but also helps in eliminating noisy and inconsistent samples from the training set, which helps the overall accuracy of WiKey.

2.8.5.3 Auto-Correction and Word Recognition

Next, we apply dictionary based auto-correction on the recognized keystrokes and study whether it improves WiKey's accuracy in terms of correctly recognizing entire words instead of individual keystrokes. Auto-correction automatically replaces a word, which is not part of the dictionary, with its best match. For word recognition experiments, we chose the number of behavioral clusters to be $C = 6$. Furthermore, we use $\mathcal{P} = 70\%$ and $\mathcal{P} = 10\%$ samples, respectively, from each cluster for training. We average the recognition results for each sentence over its five repetitions and report the final word recognition accuracies. During auto-correction, we observed cases (not shown here) where words containing keystroke(s) that WiKey recognized incorrectly were actually valid words



(a) Recognition time versus samples used

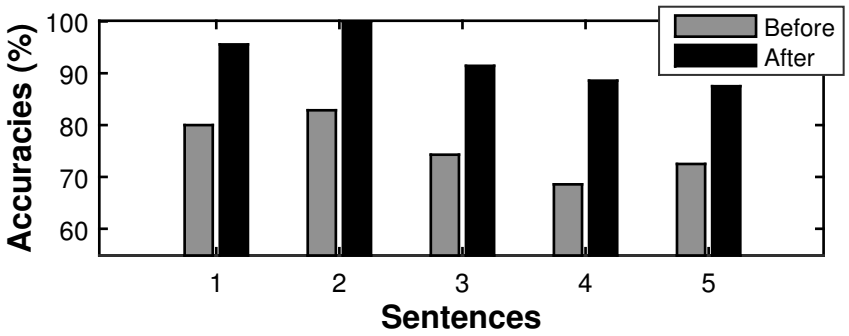


(b) Keystrokes recognized versus samples used

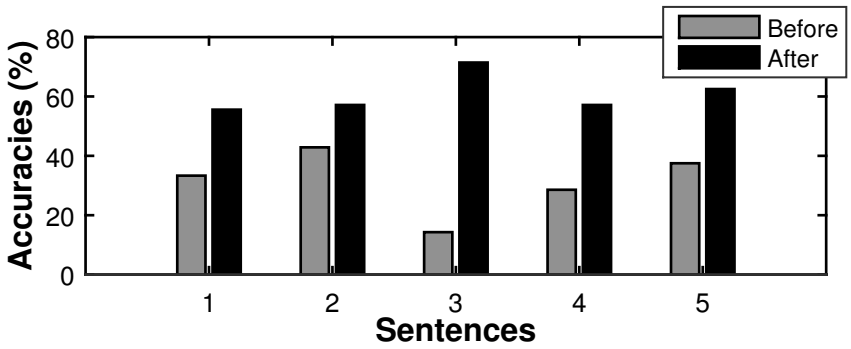
Figure 2.16: Keystrokes recognition for sentence S1 collected from user 10 after behavioral clustering

in the dictionary. For such cases, auto-correction has no effect on these words and thus on the recognition accuracy of WiKey. We treat such words as incorrect while calculating WiKey's word recognition accuracy.

Figures 2.17(a) and 2.17(b) show WiKey's word recognition accuracies, before and after auto-correction, for the five sentences collected from user 10. We observe from these figures that auto-correction improves word recognition accuracy in all cases. Furthermore, in some cases (such as for sentence 2 in Figure 2.17(a)) the word recognition accuracy reaches up to 100%. Comparing these two figures, we also observe that WiKey achieves higher word recognition accuracies when $\mathcal{P} = 70\%$ compared to when $\mathcal{P} = 10\%$. This happens due to the higher individual keystroke recognition accuracies for $\mathcal{P} = 70\%$ compared to $\mathcal{P} = 10\%$.



(a) Word recognition accuracies using 70% training samples



(b) Word recognition accuracies using 10% training samples

Figure 2.17: Comparison of word recognition accuracies before and after auto-correction

2.9 Limitations

Currently, WiKey works well under relatively stable and controlled environments. The accuracy of our current scheme will be affected by variations in the environment such as human motion in surrounding areas, changes in orientation and distance of transceivers [103] [104], typing speeds, and keyboard layout and size. During our experiments, we assumed that the major motion is due to keystrokes of the target user only and no other major motion such as walking occurs in the room where CSI data is collected. WiKey may be extended to allow small movement in the environment e.g. having multiple persons walking in a library, however this would require training WiKey with the profiles of those activities and adding the capability to subtract the waveforms of those activities to extract the waveforms of keystrokes. Furthermore, most of the parameters used in our keystroke extraction algorithm are scenario dependent and need to be changed if CSI sampling rates or physical parameters of the environment (e.g. transceiver positions) change.

During data collection, we instructed the users not to move their heads or other body parts

significantly while typing. However, we allowed natural motions which occur commonly when a person types, such as eye winking and movements in the arm, shoulder and fingers on the side of the hand being used for typing. We also instructed the users to type one key at a time while keeping the inter-arrival time of keystrokes between 0.5 to 1 second to facilitate correct identification of start and end times of keystrokes. However, we did allow users to use multiple fingers for typing so that they use whichever finger they naturally use to press any given key.

2.10 Conclusion

In this work, we make the following key contributions. First, we propose the first WiFi based keystroke recognition approach, which exploits the variations in CSI values caused by the micro-movements of hands and fingers in typing. The key intuition is that while typing a certain key, the hands and fingers of a user move in a unique formation and direction and thus generate a unique pattern in the time-series of CSI values for that key. Second, we propose a keystroke extraction algorithm that automatically detects and segments the recorded CSI time series to extract the waveforms for individual keystrokes. Third, we implemented and evaluated the WiKey system using a TP-Link TL-WR1043ND WiFi router and a Lenovo X200 laptop. Our experimental results show that WiKey achieves more than 97.5% detection rate for detecting the keystroke and 96.4% recognition accuracy for classifying single keys. In real-world experiments, WiKey can recognize keystrokes in a continuously typed sentence with an accuracy of 93.5%. The key scientific value of this work is in demonstrating the possibility of recognizing micro-gestures such as keystrokes using everyday COTS WiFi devices. We have shown that our technique works in controlled environments, and we hope that this work will spark interest of other researchers in this area to address the problem of mitigating the effects of more harsh wireless environments by building on our proposed micro-gesture extraction and recognition techniques.

CHAPTER 3

UNDERSTANDING AND MODELING WIFI SIGNALS BASED SLEEP MONITORING

3.1 Introduction

3.1.1 Motivation

Long-term sleep monitoring is crucial for the patients with sleep disorders, as well as, for the general population so that people can keep track of their sleep quality and improve their sleeping habits. Moreover, continual sleep monitoring can help with early identification of sleep disorders and related illnesses which would otherwise go undiagnosed. Sleep disorders such as insomnia, apnea, and narcolepsy are common in general population, and associated with greater risk of cardiovascular, neurological, psychiatric, and other disorders [68, 119]. Moreover, studies have shown that a majority of us remain unaware of our overall sleep quality and longer-term patterns in our sleep duration and consistency [138].

Recently, WiFi CSI signals based methods have emerged as an effective approach to low-cost and easily adoptable sleep monitoring for in-home environments. The idea is to track breathing and other body/limb activity, which are closely related to sleep quality in humans [33, 82, 117], by leveraging the changes caused by those bodily motions in WiFi CSI signals. CSI based methods are by far some of the least intrusive methods for monitoring sleep, both in terms of privacy and convenience of use. The "gold standard" procedure for scoring sleep and assessing sleep disorders is polysomnography (PSG) [70]. However, PSG is highly obtrusive mainly due to its prohibitively high costs and low patient comfort-level because it requires subjects to sleep in an unfamiliar and motion-constrained environment. Several other less obtrusive methods have been proposed for tracking vital signs (body motion, cardiac and/or respiratory activity) during sleep, for example Actigraphy [12, 40, 83, 99, 117, 118, 158] and EEG [31] based techniques. However, these methods still require body contact which is something people are often not comfortable with [27]. Moreover, some people (*e.g.* the elderly) can forget to wear such devices before going to sleep. For

these reasons, contact-less sleep monitoring has attracted significant interest, which mainly includes Audio [35,50,106], Video [52,107], Bed sensors (*e.g.* Ballistocardiography (BCG), pressure and/or motion sensors based techniques) [24, 28, 67, 87, 102, 153] and RF sensing based techniques (*e.g.* mmWave, Frequency Modulated Continuous Wave (FMCW) radar, Pulse-Doppler radar, RFIDs and WiFi based techniques) [9, 77, 80, 100, 101, 111, 160, 162]. Audio and video based techniques are privacy intrusive, and therefore, are often avoided. Bed sensors based techniques involve considerable deployment effort because the sensors have to be installed at specific locations on the bed or inside the pillows. Most existing radar-based techniques—whether they are mmWave or RF—can monitor breathing and other movements during sleep fairly well. However, their operation requires LOS which leads to significant directivity issues. Moreover, all of them require users to buy dedicated devices, which often tends to be expensive —*e.g.* Xethru costs $400[1] per module. This prevents their large-scale and long-term deployment.

3.1.2 Limitations of Prior Art

Multiple WiFi CSI based schemes have been proposed for tracking vital signs during sleep [53, 77, 80, 146, 164]. The key limitation of these schemes is the lack of a model that can correlate the changes introduced in WiFi CSI with the bodily activity (*e.g.* breathing and body/limb motion) during sleep. Due to the lack of such a model, they rely on trial-and-error based positioning of WiFi transceivers and signal processing techniques to track vital signs, which leads to high dependency on multiple, environment-dependent and difficult to tune parameters. Consequently, their techniques lack robustness to different individuals and environments, and they only work well in controlled lab experiments performed on the same user, where they require the user to lie down in between and/or very close to both transmitter (TX) and the receiver (RX) to ensure line-of-sight (LOS) scenarios. As all the previous WiFi based sleep monitoring scheme have been evaluated with short-duration mock sleep experiments in very controlled settings, the applicability of their techniques and findings in practical sleep monitoring scenarios is limited. Their methods may be

[1] at the time of writing [155]

suitable for controlled short-duration sleep experiments; however, they cannot be generalized to different individuals, environments, positioning of WiFi transceivers, LOS/NLOS situations, and to natural in-home full-night sleeping scenarios.

3.1.3 Proposed Approach

In this work, we propose Serene, a WiFi CSI based sleep quality monitoring scheme which can robustly track breathing and body/limb activity related vital signs during sleep throughout a night in an individual and environment independent manner. We propose two models based on which we develop Serene's signal processing pipeline: a *breath-multipath* model, and a *breath-subspace* model. Our breath-multipath model quantifies the effect of small breathing movements on the CSI signals, and allows Serene to robustly extract breathing waveforms. Our breath-subspace model quantifies how breathing affects the signal subspace formed by WiFi subcarriers very differently compared to other bodily motions, and allows Serene to robustly differentiate between breathing and body/limb activity during sleep. These two models combined correlate the changes in CSI values with the user's vital signs during sleep, and therefore, form the foundation of Serene's signal processing pipeline to robustly track those vital signs. On the hardware side, Serene consists of two commodity off-the-shelf (COTS) WiFi devices: a transmitter (*e.g.* a router) for continuously sending signals, and a receiver (*e.g.* a laptop or a small embedded device placed close to the sleeping user) for continuously receiving those signals to sample CSI. When a user is sleeping near the receiver, Serene continuously tracks their breathing and other body/limb activity related vitals signs. Using these vital signs, Serene also measures user's per-night sleep quality based on a well-known light-weight sleep scoring technique proposed in [150].

Our proposed breath-multipath and breath-subspace models advance the state-of-the-art on WiFi signal based sleep monitoring from two fronts. First, they provide us the theoretical basis to understand the relationship between changes in CSI values and a user's vital signs during sleep, and the relationship between the vital signs and the signal subspace formed by different subcarriers of the WiFi signals. Regarding the relationship between CSI value dynamics and the vital signs,

our model shows that if we differentiate (*i.e.* by taking *first order difference*) the CSI signals from each WiFi subcarrier, and then *max-min normalize* the CSI signal projection corresponding to variations due to breathing, we can robustly extract the waveform corresponding to user's breathing motion in an environment and individual independent manner, as long as the user sleeps close to the WiFi receiver. Such a requirement is easy to achieve in real-life in-home sleep scenarios. Regarding the relationship between the vital signs and the signal subspace formed by different subcarriers of the WiFi signals, our model shows that when there is no body/limb motion, there is only one dominant time-varying component in the subspace, which corresponds to breathing. However, more components along these dimensions evolve (*i.e.* show significant variations) during other body movements, *e.g.* during roll overs or arm/leg movement. Based on this model, we are able to distinguish breathing from body/limb motion events during sleep, without requiring any environment-dependent calibrations. To the best of our knowledge, our breath-subspace model is the first of its kind to utilize the formal concept of signal subspace from wireless literature [11] in a concrete real-world application. Our breath-multipath and breath-subspace models combined help Serene avoid the inconvenience of requiring every user to provide calibration for multiple different environments and possible positioning of the transceivers, and therefore, allow us to build a robust scheme which can track vital signs using design-time training data obtained from only few configurations and users. *Therefore, our modeling lowers the deployment and usability barriers of low-cost, in-home sleep monitoring using COTS WiFi devices.*

3.1.4 Technical Challenges and Solutions

The first technical challenge is to extract the CSI values that are resilient to static changes in the environment (*e.g.* changes in arrangement of room furniture). To address this challenge, based on our proposed *breath-multipath* model, we differentiate the CSI streams coming from WiFi NIC by taking their first order difference. This procedure not only removes the static changes in CSI values, but also brings CSI timeseries into a form using which Serene's breath tracking algorithm can robustly measure breathing rate independent of different individuals and deployment scenarios.

The second technical challenge is to isolate the CSI variations due to bodily activity during sleep from the noisy CSI values in real-time. The COTS WiFi NICs report noisy CSI values, both due to hardware limitations (*e.g.* low resolution Analog to Digital Converters (ADCs), etc.) and due to changing transmission power and rates. To remove such high frequency noise from the CSI streams, we use a combination of median, exponential moving average (EMA), and Butterworth low-pass filters. This combination of filters effectively removes higher frequency noise in CSI values, while also retaining information about the breathing and body/limb motion during sleep which is needed for Serene to robustly track those vitals signs.

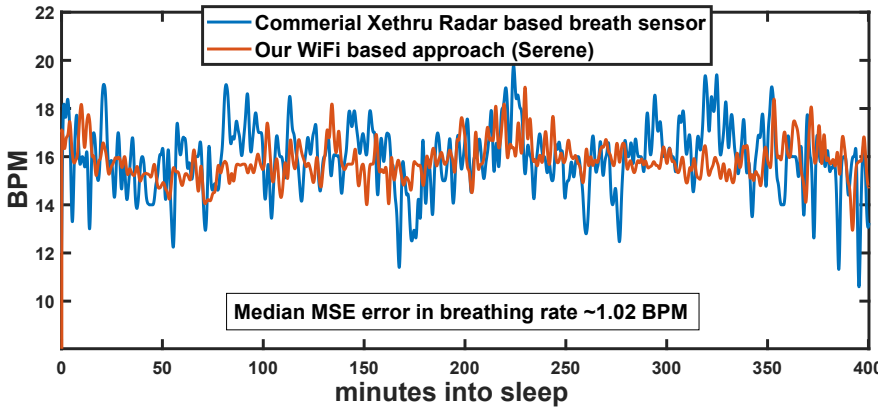
The third technical challenge is to robustly and accurately distinguish between breathing and other body movements. To achieve this, we follow our proposed *breath-subspace* model. Today's MIMO and OFDM based WiFi devices use many frequency subcarriers and multiple transmit-receive (Tx-Rx) antennas for data communication. The MIMO system consisting of the OFDM subcarriers and the Tx-Rx antennas, forms a multidimensional array which can be represented by a signal subspace along these frequency and spatial dimensions [11,32,134,151]. The key intuition is that while a user is sleeping, the signal subspace along these dimensions is affected by both breathing and body/limb motion. When there is no body/limb motion, there is only one dominant time-varying component in the subspace, which corresponds to breathing. However, as attested by our experimental observations, more components along these dimensions evolve (*i.e.* show considerable variations) during other body/limb activity *e.g.* during roll overs or arm/leg movement. Based on this principle, we can isolate breathing from limb motion without requiring any environment-dependent calibrations. To achieve this in Serene, we first extract a batch of top dominant components from the multi-dimensional CSI signal using Principal Component Analysis (PCA). In the absence of other body movements, Serene tracks breathing using the top-most PCA projection, which automatically captures the variations due to breathing. However, during body/limb movements, breathing cannot be tracked as these movements cause signification variations in multiple top PCA projections which makes breathing almost impossible to extract. Therefore, to accurately detect and then discard the CSI values corresponding to such body/limb movement events, we track variations in the lower PCA

components by means of a multi-dimensional clustering technique. During body movement, these lower PCA components show considerable variations, which are robustly and accurately detected by our clustering approach.

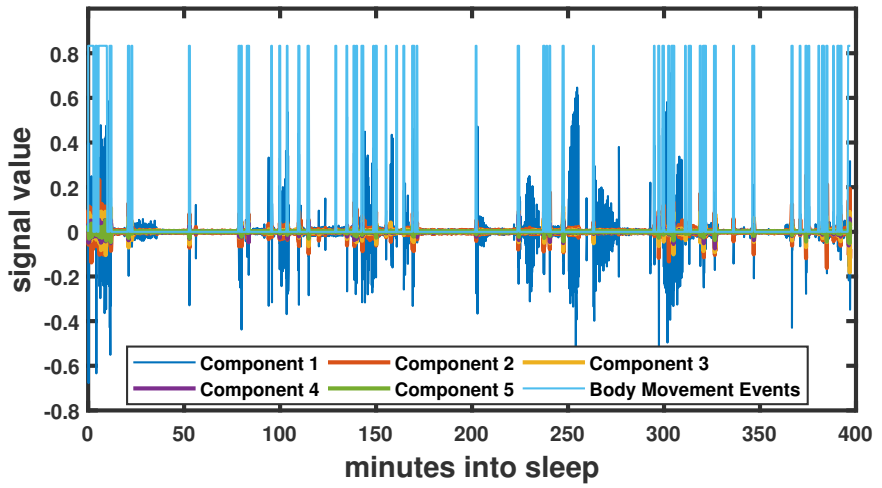
The fourth challenge is to accurately estimate the breathing rate using the top PCA projection of CSI streams. To estimate breathing rate, we use a *peak detection* based approach. First, following our breath-multipath model, we max-min normalize the signal which makes parametrization of our peak detection algorithm easily generalizable to different users. Second, to achieve accurate tracking of breathing rate, we perform parameterization of Serene's breath estimation algorithm using ground truths obtained from a contact-less COTS Xethru X4M200 Breath/Motion sensor. The Xethru radar-based sensor has been shown to have 96% breath tracking accuracy compared to PSG, *i.e.* the gold standard for sleep monitoring [154]. We perform this parametrization only during the design of our system and do not require any end-user calibration effort during real-world deployments.

3.1.5 Summary of Experimental Results

We developed HummingBoard (HMB) [127] based easy to deploy sleep monitoring devices, which were installed with Intel 5300 NIC for extracting CSI information [47]. We used Linksys AC1200+ routers in our deployments. Moreover, we developed client (in C) and server (in Python) programs capable of real-time CSI extraction and processing throughout the night. We tested Serene on 5 different individuals, where we collected >550 hours (80 nights) of CSI data at their apartments. 55% of our dataset corresponds to NLOS deployment scenarios, and 45% to LOS. Our results demonstrate that Serene can track breathing throughout a night with an average error of <0.59 BPM breaths per minute (BPM) for controlled sleep experiments and an average error of <1.19 BPM for real-world in-home sleep experiments, respectively. Our system experiences an average nightly outage (during which it cannot track the vitals) of less than 6.38 minutes. Figures 3.1(a) and 3.1(b) show how our proposed system tracks breathing and body movement vital signs during a full night's sleep of a subject, where the receiver was placed on a table close to the subject's bed and router



(a) Respiration tracking (full night's sleep)



(b) Body movement tracking (full night's sleep)

Figure 3.1: Example showing our system tracking breathing and body movements throughout full night's sleep of subject. Xethru radar (X4M200 [155]) ground truth is approximately synchronized with CSI data

was placed outside their bedroom in their TV lounge. We also assess per-night sleep efficiency (i.e. the percentage of night spent in sleep stage) results of our subjects based on a classic light weight actigraphy based sleep scoring algorithm [150], which tracks each night's sleep progress in terms of sleep and awake stages based on the body movement information provided by Serene. The sleep scoring algorithm we use has been shown to agree with EEG based sleep monitoring 94.46% of the time [150]). We also discuss the possibility of advanced sleep-stage classification based on the vital signs obtained from our system in section §2.9, where we compare some results of our current light weight sleep-stage classification approach with a commercial sleep monitoring device, ResMed S+ [113].

3.2 Related Work

3.2.1 Respiration, Body Movements and Sleep

Previous works have shown that breathing and body movements during sleep are closely related to sleep quality in humans [33, 82, 117]. These studies show that respiratory dynamics vary over sleep stages, which means that respiratory activity can be used to separate sleep stages [82]. For example, Dafna *et al.* evaluated whole night sleep based on sleep-awake classification using audio recordings of breathing sounds [33]. They captured and quantified variations in breathing features such as periodicity and consistency, and showed that these features contribute to distinguishing between sleep and wake epochs. Our work is motivated by such studies, where our goal is to develop a robust and generic scheme to extract breathing and limb/body activity related vital signs using CSI signals obtained from COTS WiFi devices. Our scheme can be used by sleep researchers to develop low-cost and easily deployable/scalable sleep-stage monitors which can track different stages (e.g. *light*, *deep*, *rapid-eye-movement* (REM), and *awake*) of sleep.

3.2.2 Sleep Monitoring Technologies

Several sleep monitoring techniques have been proposed in the past which use different sensing modalities, such as in-ear [97], inertial sensors (Actigraphy) [12, 40, 83, 99, 117, 118, 131, 158], EEG [31], Audio [35, 50, 106], Video [52, 107], Bed sensors (*e.g.* Ballistocardiography (BCG), pressure and/or motion sensors based techniques) [24, 28, 67, 87, 102, 153] and RF sensing based techniques (*e.g.* mmWave, Frequency Modulated Continuous Wave (FMCW) radar, Pulse-Doppler radar, RFID and WiFi based techniques) [9, 77, 80, 100, 101, 111, 160, 162]. For brevity, we will only discuss some of the closely related recent works on contact-less sleep monitoring, which include some sound, radar and WiFi CSI based techniques.

Lullaby [63] tracks various environmental factors, sound, light, temperature, and motion that help users assess the quality of their sleep environments. iSleep [50] uses the built-in microphone of the smartphone to detect the events that are closely related to sleep quality, including body movement, couch and snore, and infers quantitative measures of sleep quality. Sleep Hunter [45]

uses actigraphy and acoustic events to predict sleep stage transitions by smartphone. Toss-N-Turn [88] uses features such as sound amplitude, acceleration, light intensity, screen proximity, battery and screen states, etc. to track a subject's sleep quality. However, Audio based techniques are privacy invasive, and therefore, often avoided as sleep is a private activity.

RF sensing based techniques are by far the least intrusive methods for monitoring sleep, both in terms of privacy and convenience of use. DoppleSleep [111] is another unobtrusive sleep sensing system which uses short-range doppler radar to perform sleep stage classification (Sleep vs. Wake and REM vs. Non-REM). Vital-radio [9] develop an FMCW based system which is shown to accurately track a person's breathing and heart rate without body contact, from distances up to 8 meters. Based on the same system, [165] proposes a deep learning architecture to perform 4-stage sleep stage classification. More recently, authors of [162] proposed algorithms to achieve multi-person identification and breath monitoring based on the same FMCW hardware. Although the aforementioned radar based techniques do fairly well in terms of monitoring vital signs during sleep. However, they require dedicated hardware and spectrum, adding cost and/or RF regulation hurdles. These factors prevent their large-scale and long-term deployment. WiFi signals based sensing provides an effective approach towards low-cost and easily adoptable long-term sleep monitoring, as the widespread use of WiFi capable devices (*e.g.* smart-home assistants, smart-phones, etc.) has made WiFi signals the most ubiquitous form of sensing in homes requiring no additional hardware costs. Multiple WiFi CSI based schemes have been proposed for tracking vital signs during sleep [77, 80, 146, 164]. The key limitation of these schemes is the lack of a model that can correlate the changes introduced in WiFi CSI with the bodily activity (*e.g.* breathing and body/limb motion) during sleep. Due to lack of such a model, they rely on trial-and-error based positioning of WiFi transceivers and signal processing techniques to track vital signs, which leads to high dependency on multiple, environment-dependent and difficult to tune parameters. Consequently, their techniques lack robustness to different individuals and environments, and they only work well in controlled lab experiments performed on the same user, where they require the user to lie down in between and/or very close to both transmitter (TX) and the receiver (RX) to

ensure line-of-sight (LOS) scenarios. As all the previous WiFi based sleep monitoring scheme have been evaluated with short-duration mock sleep experiments in very controlled settings, the applicability of their techniques and findings in practical sleep monitoring scenarios is limited. Their methods may be suitable for controlled short-duration sleep experiments; however, they cannot be generalized to different individuals, environments, positioning of WiFi transceivers, LOS/NLOS situations, and to natural in-home full-night sleeping scenarios.

3.3 Modeling of Vital Signs and WiFi CSI

3.3.1 Overview of WiFi CSI

WiFi devices measure the Channel State Information (CSI), which characterises the surrounding wireless channel across bandwidth and multiple antennae. The Orthogonal Frequency Division Multiplexing (OFDM) communication scheme used in IEEE 802.11a/n/ac divides the wireless channel's bandwidth into multiple modulated subcarriers. To correct for channel frequency-selectivity (or equivalently the *delay spread* in time-domain) and maximise the link's capacity, WiFi devices continuously track changes over these subcarriers in terms of CSI values, which are then used to adapt transmission power and rates in real time. CSI values are the *Channel Frequency Response* (CFR) at per subcarrier granularity between each transmit-receive (Tx-Rx) antenna pair. When a user is sleeping, the chest and body movements change the constructive and destructive interference patterns of the WiFi signals. The CSI values are sensitive enough to measure these breathing movements, as CSI measurements can be obtained at high sampling rates and from multiple different OFDM subcarriers of each TX-RX stream. For example, the driver of the Intel 5300 WiFi NIC, which we use to implement our scheme, reports CSI values on 30 OFDM subcarriers [47] for each TX-RX antenna pair for every CSI measurement. This leads to 30 matrices with dimensions $M_t \times M_r$ per CSI sample, where M_t and M_r denote the number of transmit and receive antennas respectively. Such high dimensional data allows us to recover detailed information about the sleeping behavior even if the breathing movements only incur small changes in the CSI.

3.3.2 Breath-Multipath Model

Next, we develop our breath-multipath model to understand the effect of small breathing movements on the CSI signals. Based on this model, we design Serene's signal processing pipeline to robustly extract breathing waveforms in an individual and environment independent manner. Our model shows that if we differentiate (*i.e.* by taking *first order difference*) the CSI signals from each WiFi subcarrier, and then *max-min normalize* the CSI signal projection corresponding to variations due to breathing, we can robustly extract the waveform corresponding to user's breathing motion in an environment and individual independent manner, as long as the user sleeps close to the WiFi receiver. Such proximity requirement is easy to satisfy during real-life in-home sleep scenarios by either mounting receiver on the headboard of a bed frame or placing it on a table nearby. The basis of our model is formed by a closed form expression, which we derive using time-varying Channel Frequency Response (CFR) of WiFi channel. The time-varying CFR corresponding to a Tx-Rx antenna pair for a subcarrier with wavelength λ can be quantified as [42]:

$$H(f,t) = H_s(f) + \underbrace{\sum_{i=1}^{N} \frac{K}{D_i(t)^2} \times e^{\frac{j2\pi D_i(t)}{\lambda}}}_{H_d(f,t)} \tag{3.1}$$

In the equation above, N is the number of multipath reflections of the transmitted signal at the Rx end, D_i represents the distance traveled by i^{th} multipath reflection, and K is an environment dependent proportionality constant. $H_s(f)$ is the *static* component of CFR corresponding to all non-user multipath reflections, while the second term on the right hand side corresponds to the *dynamic* component of CFR, represented as $H_d(f,t)$, while the user is breathing and/or moving during sleeping. Now, let us assume that user is sleeping at a distance $D_{0,i}$ from the router, and $d_i(t)$ is the change in distance traveled by i^{th} reflected path due to breathing. To make our scheme resistant to static changes in the environment, we first eliminate $H_s(f)$ by differentiating the above equation with respect to t, and substitute $D_i(t) = D_{0,i} + d_i(t)$ to get:

$$H'(f,t) = \frac{d}{dt}\left[\sum_{i=1}^{N} \frac{k}{D_{0,i}^2}\left(1 + \frac{d_i(t)}{D_{0,i}}\right)^{-2} e^{\frac{j2\pi(D_{0,i}+d_i(t))}{\lambda}}\right] \tag{3.2}$$

As $d_i(t)$ caused by motion due to breathing is in the order of a few centimeters, whereas $D_{0,i}$ is usually in the order of meters (i.e. $d_i(t) \ll D_{0,i}$), we can expand the negative polynomial $(1 + \frac{d_i(t)}{D_{0,i}})^{-2}$ via binomial series expansion. After performing binomial expansion, discarding the $\frac{d_i(t)^m}{(D_{0,i})^n}$ terms with n = 4 or higher, and doing some algebraic manipulations, we get the following expression for $H'(f,t)$:

$$H' \approx k e^{\frac{j2\pi D_{0,i}}{\lambda}} \sum_{i=1}^{N} d_i'(t) \Bigg(-\frac{2}{D_{0,i}^3} + j\Bigg[\frac{2\pi}{\lambda D_{0,i}^2} - \frac{4\pi d_i(t)}{\lambda D_{0,i}^3}\Bigg]\Bigg) e^{\frac{j2\pi d_i(t)}{\lambda}}$$

After converting the term inside summation into polar coordinates, and discarding the $\frac{d_i(t)^m}{(D_{0,i})^n}$ terms with n = 4 or higher, we get the following simplified expression for $H'(f,t)$:

$$H' \approx \Bigg[\frac{2\pi k}{\lambda D_{0,i}^2}\sqrt{1 + \left(\frac{\lambda}{\pi D_{0,i}}\right)^2} \cdot e^{\frac{j2\pi D_{0,i}}{\lambda}}\Bigg] \cdot \Bigg[\sum_{i=1}^{N} d_i'(t) e^{\frac{j2\pi d_i(t)}{\lambda} + jA_i}\Bigg]$$

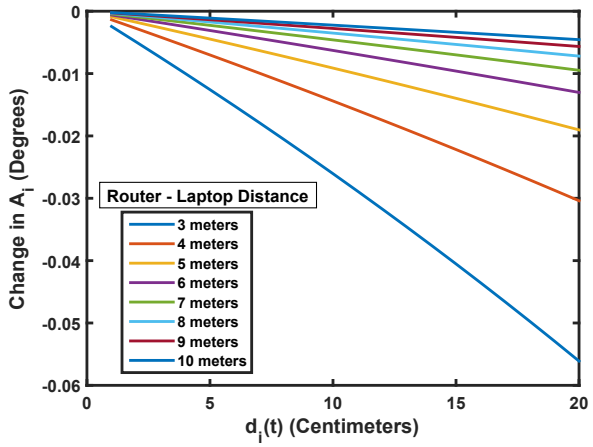
Here, $A_i = tan^{-1}\Big[\frac{\pi D_{0,i}}{\lambda}\Big(1 - \frac{2 \cdot d_i(t)}{D_{0,i}}\Big)\Big]$. Figure 3.2(a) shows variation of A_i with $d_i(t)$, as $d_i(t)$ varies from 1cm to 20cm (typical range for motion due to human breathing is 1-5cm [82]), for different router-receiver distances $D_{0,i}$ ranging from 3m - 10m (*i.e.* which is typical for regular home use cases).

We observe that changes in $d_i(t)$ do not significantly affect the value of A_i. Moreover, the impact of $d_i(t)$ on A_i decreases even further as the distance between receiver and the router it is connected to increases. Therefore, we can safely approximate $A_i \approx tan^{-1}\Big[\frac{\pi D_{0,i}}{\lambda}\Big] = A_{0,i}$ and write $H'(f,t)$ as:

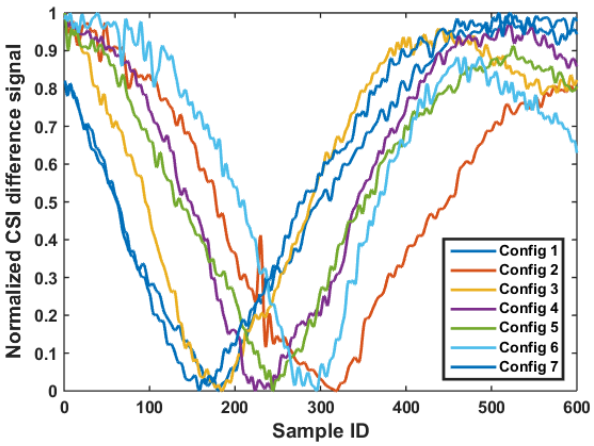
$$H' \approx \Bigg[\frac{2\pi k}{\lambda D_{0,i}^2}\sqrt{1 + \left(\frac{\lambda}{\pi D_{0,i}}\right)^2} e^{j\left(\frac{2\pi D_{0,i}}{\lambda} + A_{0,i}\right)}\Bigg] \cdot \sum_{i=1}^{N} d_i'(t) e^{\frac{j2\pi d_i(t)}{\lambda}}$$

The first term on right hand side of the equation above stays constant when receiver is placed on some surface, *e.g.* a desk/table, and is not moving. We write amplitude of CFR *i.e.* $|H'(f,t)|$ as:

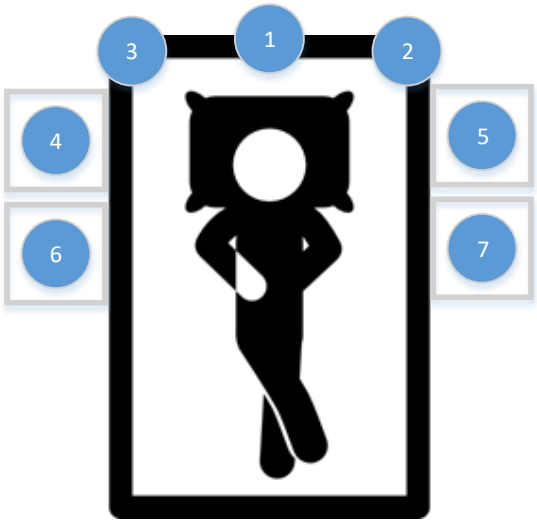
$$|H'(f,t)| \approx C_{0,i} \cdot \Big| \sum_{i=1}^{N} d_i'(t) e^{\frac{j2\pi d_i(t)}{\lambda}} \Big| \tag{3.3}$$



(a) Variation of A_i with $d_i(t)$ for $D_{0,i}$ 3 to 10



(b) Single breath samples for 7 configurations



(c) Configurations of receiver

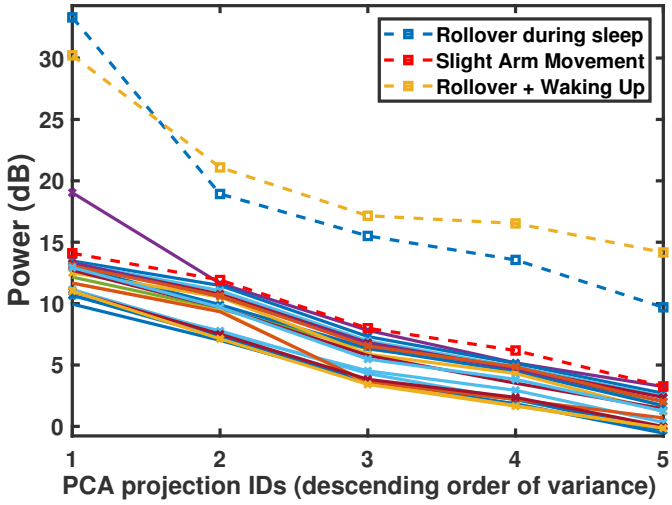
Figure 3.2: (a) Variation of A_i with $d_i(t)$ for different $D_{0,i}$; (b) Single breath samples for different configurations shown in (C)

The waveform $\left|\sum_{i=1}^{N} d_i'(t) e^{\frac{j2\pi d_i(t)}{\lambda}}\right|$ corresponds to the variations due to breathing. The proportionality term $C_{0,i}$ in breathing samples extracted from $|H'(f,t)|$ corresponding to different placement of receiver can be easily eliminated via *max-min normalization*. Figure 3.2(b) shows extracted and processed single breath samples from a user for seven slightly different receiver placement configurations close to the user, while the router was in subject's TV-lounge (router-receiver distance >10 meters).

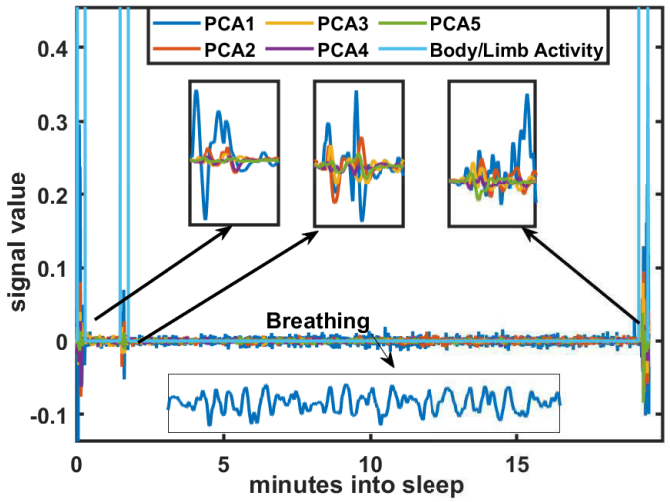
3.3.3 Breath-Subspace Model

Next, we present our breath-subspace model to understand how breathing affects the signal subspace formed by WiFi subcarriers compared to other bodily movements. Today's MIMO and OFDM based WiFi devices use many frequency subcarriers and multiple transmit-receive (Tx-Rx) antennas for data communication. The MIMO system between the OFDM subcarriers and the Tx-Rx antennas, forms a multidimensional array which effectively represents a high-dimensional mathematical space. Contained in this space is the signal subspace along frequency and spatial dimensions. The key intuition behind our model is that while a user is sleeping, the signal subspace along these dimensions is affected by both breathing and body/limb motion. When there is no body/limb motion, there is only one dominant time-varying component in the subspace, which corresponds to breathing. However, more components along these dimensions evolve (*i.e.* show considerable variations) during other body/limb activity *e.g.* during roll overs or arm/leg movement. Based on this principle, Serene isolates breathing from limb motion without requiring any environment-dependent calibrations.

To model this in Serene, we track the top dominant components in the CSI signal subspace using Principal Component Analysis (PCA). Figure 3.3(a) shows power values in top 5 PCA projections of the CSI signals corresponding to multiple sleep epochs during a sleep experiment, where the dotted lines correspond to epochs with motion events—highlighted in Figure 3.3(b). We observe that in the absence of any body/limb activity, the top-most PCA projection is enough to track breathing as it is the only major motion occurring in the environment. However, during body/limb movements, multiple lower PCA projections also show significant variations. Based on this phenomena, we accurately detect and then discard the CSI values corresponding to any body/limb activity by tracking variations in the lower PCA components (*e.g.* 3, 4 and 5) using a multi-dimensional clustering technique, which we discuss in §2.6.



(a) Power values in top PCA projections



(b) CSI timeseries for Fig. 3.3(a)

Figure 3.3: Impact of bodily activity during sleep on WiFi subspace

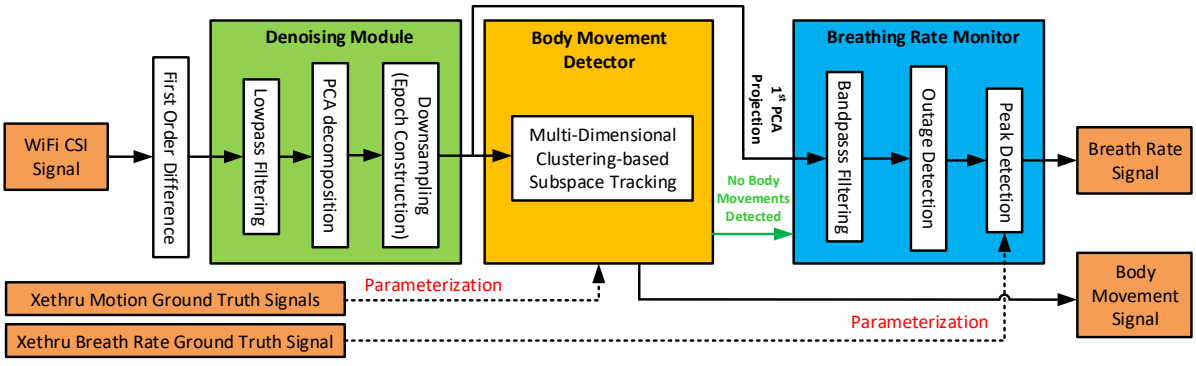


Figure 3.4: Our WiFi CSI signal processing architecture for extracting vital signs

3.4 CSI Signal Processing Architecture

To obtain CSI data in real-time during sleep, we develop a client-server based mechanism to communicate the CSI values extracted from WiFi NIC to a Python based CSI processing server. Based on our discussion in §5.3, we take first order difference of the incoming CSI data and then take its amplitude *i.e.* $|H'(f,t)|$ for further processing. From now onward, we use the term "CSI" to denote $|H'(f,t)|$. CSI data is collected in 30 second epochs, which is typically the partitioning convention followed by most sleep monitoring systems. Next, we first perform basic low-pass filtering for removing bursty noise due to hardware noise and isolate the signal of interest i.e. to extract human motion related frequencies only. Second, we perform PCA on the low-pass filtered CSI streams for dimensionality reduction and automatic distinguishing of CSI variations due to body movements from those of breathing in different subcarriers based on our discussion in §3.3.3. This avoids the need for complex trial-and-error based subcarrier selection procedures used in previous works [77, 80]. Third, we harmonise the filtered CSI data corresponding to each 30s sleep epoch into uniformly sampled and consistent measurements via down-sampling. Fourth, we robustly detect body movements by tracking lower PCA projections of CSI signals using a clustering-based event detection technique. Finally, we first detect the presence of breathing using a power threshold, and then perform further band-pass filtering to extract the signal corresponding to breathing. Figure 3.4 shows our system architecture. Next, we briefly discuss Serene's noise removal process.

3.4.1 Noise Removal

Commodity Wi-Fi NICs report noisy CSI values, both due to hardware limitations (such as low resolution Analog to Digital Converters (ADCs)) and due to changing transmission power and rates. We use a combination of median filter and an exponential moving average filter to get rid of such bursty noise and spikes in CSI time series. After this basic filtering step, we further remove any high frequency variations in CSI signals using *Butterworth* low-pass filter. Variations due to movement during sleep cause low frequency variations, typically under 5 Hz [82]. We use Butterworth low-pass filter for separating these variations from higher frequency noise in CSI values. Due to maximally

flat amplitude response of Butterworth filter, its application on CSI time series does not distort the shape of CSI variations due to body motion. Our scheme samples CSI values at a nominal frequency $F_s = 800$. With this in mind, we use cut-off frequency $\omega_c = \frac{2\pi * f}{F_s} = 0.0125\pi$ rad/s for Butterworth filtering. We apply the same filter on CSI timesseries of all the subcarriers, making sure that every CSI stream experiences the same phase distortion and group delay introduced by the filter.

Based on our experimental results, we observed that filtered CSI waveforms still retain some noisy variations which are not related to activity/breathing. We avoid any further low pass filtering on CSI streams as it can lead to loss in details of variations due to activity/breathing behavior. To remove such noise, we utilize the fact that the CSI variations in CSI streams of multiple subcarriers in each Tx-Rx antenna pair are correlated. We apply PCA on CSI obtained from all subcarriers and all Tx-Rx pairs, and retain only the waveforms that represent the most common variations in all the subcarriers, i.e., the variations due to breathing and/or body movements during sleep. That is, signal subspace-based filtering enables our scheme to automatically obtain the signals that are representative of the monitored vital signs only. PCA also reduces the dimensionality of data by discarding the principal components unrelated to the vital signs i.e. the noise subspace. Finally, we rearrange the multi-dimensional filtered CSI data corresponding to each 30s sleep epoch into consistent length samples (600 in our current implementation) via down-sampling, performing zero-padding where necessary. Although we have partitioned the incoming CSI data into 30s epochs, we concatenate data from consecutive epochs for real-time tracking of vital signs (*e.g.* breathing) which we discuss later in this section.

3.4.2 Tracking Body Movements

As discussed earlier, today's MIMO and OFDM based WiFi devices use multiple frequency subcarriers and Tx-Rx antennas for data communication. The MIMO system between the Tx-Rx antennas, and the OFDM subcarriers, form a multidimensional tensor along space-frequency axes. Contained in such tensor is the signal subspace we wish to track for breathing and body motion.

We observed that when there is no body/limb motion, there is only one dominant, time-varying component in the signal subspace, which corresponds to breathing. PCA sorts different principal components in descending order of their variation. During sleep, the signal subspace is rather quiet and breath is captured in the top PCA projection of the CSI time series. However, we observed that during episodes of other body movements—*e.g.* during roll overs or arm/leg movement—more signal subspace components evolve, since body movements cause more pronounced variations in the spatial-frequency subspace compared to faint breathing movements.

3.4.2.1 Body Movements Detection Approach

To robustly distinguish body/limb motion from breathing, we propose to use a multi-dimensional *hyper-ellipsoidal* clustering on the lower PCA projections of CSI data. At a high-level, we can think of this clustering method as a high-dimensional generalization to a Gaussian outlier rejection technique whereby measurements few sigma's away from the mean are deemed erroneous. Specifically, let $R_k = \{r_1, r_2, \cdots r_k\}$ be the first k samples of CSI vectors containing values from the selected signal subspace—*we use PCA projections 3, 4 and 5 in our current implementation*. Each sample r_i is a $d \times 1$ vector in $\mathbb{R}^d$, where d is the number of signal subspace components. This *hyper-ellipsoidal* technique clusters the normal data points (i.e. when there is no body movement in the environment), and any points lying outside the cluster are declared as outliers. The boundary of the cluster (a "hyper-ellipsoid' in this case) is related to a distance metric which is a function of mean $m_{R,k}$ and covariance S_k of the incoming signal subspace components R_k. We use the *Mahalanobis* distance metric, D_i, for which the cluster is arrived at according to the following [91]:

$$e_k(m_R, S_k^{-1}, t) = \{r_i \epsilon \mathbb{R}^d | \underbrace{\sqrt{(r_i - m_{R,k})^T {S_k}^{-1} (r_i - m_{R,k})}}_{D_i = Mahalanobis\ \ distance\ \ of\ \ r_i} \leq t\} \tag{3.4}$$

where e_k is the set of normal data points whose Mahalanobis distance, $D_i < t$ and t is the *effective* radius of the hyper-ellipsoid. The choice of t depends on the distribution of the normal data points. If the normal data follows a *chi-squared* distribution, it has been shown that up to 98% of the incoming normal data can be enclosed by the boundary of an hyper-ellipsoid, if the

effective radius t is chosen such that $t^2 = (\chi_d{}^2)^{-1}_{0.98}$ [91]. Data samples r_i for which $D_i > t$, are therefore, identified as outliers. As it is often not practical to store all the samples of a streaming data, therefore a recursive algorithm is required to update e_k. Let r_{k+1} be the most recent CSI sample. $m_{R,k+1} = \frac{km_{R,k}+r_{k+1}}{k+1}$ and $m_{R^2,k+1} = \frac{km_{R^2,k}+r_{k+1}r_{k+1}^T}{k+1}$ can be updated recursively from the previous means. By substituting covariance matrix $S_k = m_{R^2,k} - (m_{R,k}\, m_{R,k}^T)$ into Eq. (3.4) we can represent e_k entirely in terms of means. The resulting equation updates the cluster boundary and classifies the incoming data samples as normal readings or outliers. Our scheme uses above equations for initial estimation of mean and covariance. Afterwards, the mean $m_{R,k}$ is recursively calculated using an exponential moving average technique, where mean $m_{R,k+1}$ is updated as $m_{R,k+1} = \alpha\, m_{R,k} + (1-\alpha) r_{k+1}$, where $\alpha = 0.9995$ in our implementation. Moreover, after initial estimation of covariance, our scheme recursively updates the covariance inverse S_k^{-1} by using the following equation [91], which avoids extra computations required for calculating the inverse of matrix S:

$$S_{k+1}^{-1} = \frac{kS_k^{-1}}{\alpha(k-1)} \times \left[I - \frac{(r_{k+1} - m_{R,k})(r_{k+1} - m_{R,k})^T S_k^{-1}}{\frac{(k-1)}{\alpha} + (r_{k+1} - m_{R,k})^T S_k^{-1} (r_{k+1} - m_{R,k})} \right] \tag{3.5}$$

To determine the start and end of activity waveforms, we use both cardinality and temporal proximity of the detected outliers. If the number of consecutive outliers increases a threshold E_1, we declare a micro-event. Multiple micro-events constitute a activity event. All the data points including the points constituting the micro-events as well as the points in between the consecutive micro-events are recorded as part of activity waveform (*merging*). We only merge the micro-events which are within E_2 data points of each other. Both E1 and E2 are design time, easy to tune thresholds. Figure 3.5 shows some body movements detected by our algorithm in a portion of processed CSI timeseries corresponding to an in-home full-night sleep monitoring experiment.

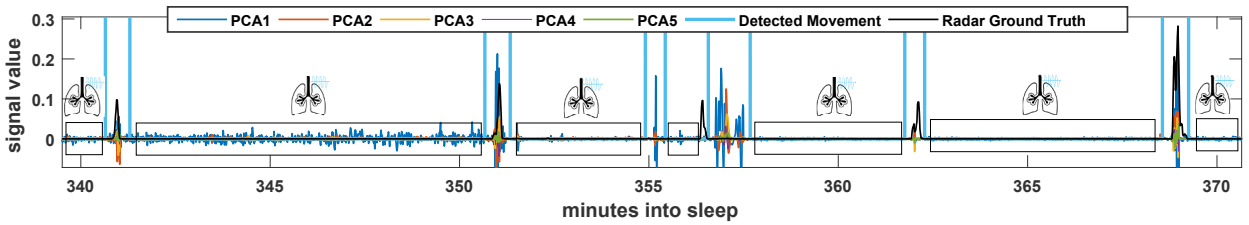


Figure 3.5: Example showing performance of our body movement detection algorithm, compared to Xethru radar ground truth. Boxes show the areas where breathing is usually present. Ground truth is approximately synchronized with CSI data

3.4.3 Tracking Breath

Human breath involves motion of chest and lungs during inspiration (when air enters the lungs) and expiration (when air is blown out from the lungs) [82]. These motions are often periodic (e.g. in case of healthy subjects [82]), and therefore, cause periodic variations in WiFi channel which we can extract using CSI data. In the absence other body movements, the first PCA projection of CSI data would be able to capture these variations due to breathing. However, as these minute variations are often embedded in noise, and because human subject might not be in proximity of the RX device, we can not always assume that breathing signal exists in a particular sleep epoch. Therefore, to robustly track breathing, we propose the following signal processing pipeline.

3.4.3.1 Bandpass Filtering

To extract the periodic variations in CSI due to breathing, we apply a Butterworth bandpass filter on the first PCA projection. We choose the filtering parameters of this filter according to the fact that breathing rate of humans (including adults as well as newly born babies) ranges between 10 - 40 breaths per minute (BPM) [82]. This step removes any non-breathing related noise present in the signal.

3.4.3.2 Detecting Outage in Breathing Signal

We define the *outage* of our system as the event when variations due to breathing are not present in the CSI signal. To detect outage, our system determines the noise floor of the environment using the first PCA projection. During real-time tracking, our system compares the average power of the

signal, calculated over 7.5s windows of a 30s sleep epoch (i.e. $1/4^{th}$ of an epoch's duration), with the determined noise floor. Our breathing rate measurement algorithm only runs if average power of the sleep epoch is greater than the noise floor.

3.4.3.3 Measuring Breathing Rate

We design our system to measure breathing rate in terms of breaths per minute (BPM). We measure the rate over a window of two sleep epochs in length, which moves over concatenated data from multiple consecutive sleep epochs. To report BPM every second, we move this window over the concatenated data every second (i.e. 20 samples a time).

To measure breathing rate, we employ a *peak detection* based approach. First, we max-min normalize the signal so that parametrization of our peak detection algorithm can be easily generalized to different users. Second, to robustly detect the number of peaks, we use three parameters, namely *minimum peak prominence* (MINPRO), *minimum peak distance* (MINDIST), and *minimum peak strength* (MINSTR). The prominence of a peak measures how much the peak stands out, due to its height and location, relative to other peaks around it. We tune MINPRO such that we only detect those peaks which have a relative importance of at least MINPRO. We tune MINDIST according to the fact that human breathing rate ranges between 10-40 BPM [82], so that redundant peaks are discarded. To further sift out redundant peaks, we only choose peaks of value greater than MINSTR times the median peak value. In our current implementation, we chose MINPRO = 0.025, MINDIST = 1.5 seconds and MINSTR = 0.6 which generalize well for different sleeping scenarios. To achieve accurate tracking of breathing rate, we perform parameterization of Serene's breath estimation algorithm using ground truths obtained from a contact-less COTS Xethru X4M200 Breath/Motion sensor [155]. We perform this parametrization only during the design of our system and do not require any end-user calibration effort in the real-world deployments.

3.5 Sleep Scoring

In this work, we take an *actigraphy* based approach towards sleep monitoring, where we classify the stage of each minute as *sleep* or *awake* period. Our approach is inspired by the classic actigraphy

based method proposed in [150], which determines sleep-awake stage of a minute by taking into account body movement related information corresponding to the surrounding minutes. The activity sleep-awake scores determined by their technique have been shown to agree with EEG based sleep monitoring 94.46% of the time [150]. In our implementation, we adopt the following model from their work, which takes 4 previous minutes and 2 following minutes into account to classify stage of the current minute:

$$s_m = \rho \times (w_{-4}a_{-4} + w_{-3}a_{-3} + w_{-2}a_{-2} + w_{-1}a_{-1} + w_0a_0 + w_{+1}a_{+1} + w_{+2}a_{+2}) \tag{3.6}$$

where s_m is the average sleep-awake score for the current minute, ρ is a scaling factor, a_{-i}, a_0, a_{+i} are activity scores (normalized number of body movement events in each minute) for previous, current and following minutes, and w_{-i}, w_0, w_{+i} represent weights for the previous minutes, current minute and following minutes. If $s_m \leq 1$, the current minute's stage is classified as *sleep*, and if $s_m > 1$, the current minute's stage is classified as *awake*. In our implementation, we chose $\rho = 0.125, w_{-4} = 0.15, w_{-3} = 0.15, w_{-2} = 0.15, w_{-1} = 0.08, w_0 = 0.21, w_1 = 0.12, w_2 = 0.13$, as suggested by the authors of [150] for best results in their real-world deployments.

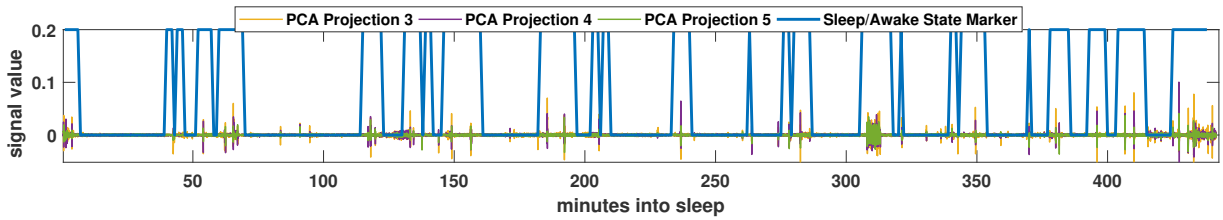


Figure 3.6: Example showing Sleep/Awake classification for full night's sleep of a subject. Sleep efficiency was 62.1%

We also calculate *sleep efficiency* (SE) for each night, which is defined as the ratio of actual time spent in sleep stages to total time spent in bed (i.e. $T_{sleep}/(T_{sleep} + T_{awake})$). Figure 3.6 showing Sleep/Awake classification performance for full night's sleep of a subject, where sleep efficiency was determined to be 62.1%.

3.6 Implementation and Evaluation

In this section, we present the performance evaluation of our system. Our results show that Serene can robustly and accurately track vital signs during sleep across different users and environments. Next, we first discuss our hardware implementation and experimental settings.

Figure 3.7: The real-world deployment scenarios used for evaluation of our sleep monitoring scheme "Serene"

3.6.1 Hardware Implementation

We developed compact HummingBoard (HMB)-based small-sized nodes as sleep monitoring devices [127] which makes Serene easy to deploy. HMB nodes were equipped with the Intel 5300 NICs with modified drivers for extracting CSI information [47]. We used Linksys AC1200+ routers as transmitters in our deployments. Moreover, we developed a client-server software architecture—in C and Python respectively—capable of the real-time extraction and processing of CSI data throughout the night. For body movement and breathing rate ground truths, we deployed state-of-the-art *pulse-doppler* radar-based Xethru X4M200 Breath/Motion sensors [155]. In terms of breathing rate accuracy, the X4M200 devices have been shown to perform very closely to a medical-grade, gold standard equipment (X4M200 has been shown to track breathing with up to 96% accuracy when compared to PSG) [154]. We utilize these ground truths in our system for: (1) the robust parametrization of our signal processing pipelines (e.g. breath tracking) and (2) measuring breath tracking accuracies.

3.6.2 Experimental Settings

We deployed our system in 5 apartments, where we collectively recorded more than 80 nights (>550 hours) of data from 5 different participants. The participants were male graduate students aged between 23 to 32 years. The duration of data collection for each participant varied from 5 to 31 days. Figure 3.7 shows the real-world deployment scenarios for our sleep experiments. The numbers in circles specify user/environment IDs. Data collected from environments 2 and 3 corresponds to NLOS deployment scenario, and constitute 55% of our dataset. Data collected from environments 1, 4, and 5 corresponds to LOS deployment scenario, and constitute 45% of our dataset. To evaluate Serene's vital signs tracking performance, we collected Xethru ground truth alongside CSI data for the environments 1, 2, 3 and 5. We evaluate Serene's performance in terms of three key metrics: (1) breath tracking accuracy, (2) breathing signal outage (during which Serene cannot track breathing), and (3) naturally occurring motion false positives due to activity of other house occupants. To determine long-term sleep quality metrics (*i.e.* sleep efficiency based on sleep-awake classification discussed in §3.5), we use data collected from the environments 1-4. Based on these metrics, we present insights on sleep efficiency of different users and how it varies in successive nights. Next, we first evaluate the breath tracking accuracy of our scheme.

3.6.3 Breath Tracking Accuracy

We evaluate the accuracy of our WiFi-based breathing rate estimation in terms of *BPM error*. We define BPM error as the average mean squared error (MSE) between per second BPM values reported by Serene and the corresponding ground truth BPM values reported by X4M200 over a specific time window. We perform this evaluation on data collected from environments 1,2,3 and 5. We evaluate both long-term (*i.e.* whole night) and short-term (*i.e.* specific short duration sleep windows during the night).

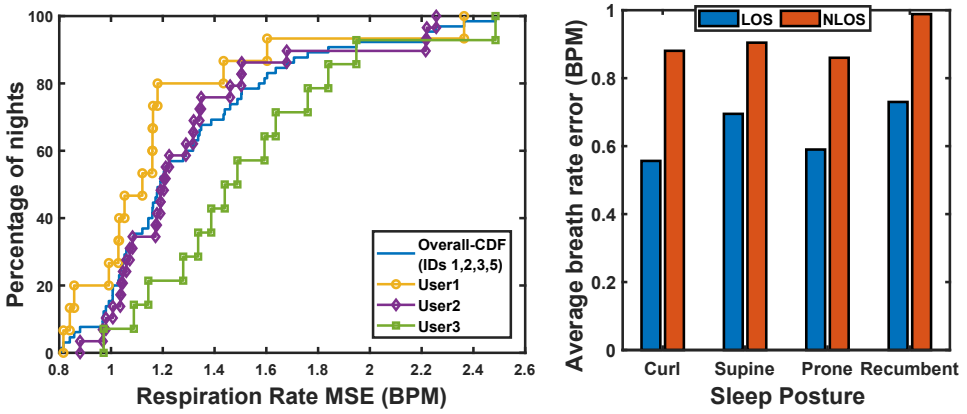
3.6.3.1 Long-term Accuracy

Serene achieves a median error of less than 1.19 BPM for real-world in-home full night sleep experiments. To evaluate Serene's long-term breath tracking accuracy, we compute the mean squared error (MSE) of instantaneous (per second) breathing rate estimate across an entire night of sleep. The overall cumulative distribution function (CDF) of the MSE error in breath per minute (BPM) is depicted in figure 3.8(a). Inspecting the blue curve, we see that the median accuracy of Serene's breathing rate estimate is 1.19 BPM, while its 95th percentile confidence is under 1.9 BPM. We skip User 5's CDF from the graph as we were only able to collect 5 nights of data from that user. The average, minimum and maximum BPM errors observed for User 5 were 1.1, 0.811, and 1.14, respectively. Figure 3.8(c) shows how Serene tracks breathing rate throughout the night in for 4 different users, where we have plotted X4M200 ground truth side by side. We can observe that even if BPM accuracies slightly drop during some parts of a night, Serene can still track the overall pattern in breath fairly well when compared to Xethru's ground truth.

Figure 3.8(a) also shows how BPM accuracy varies across subjects. For instance, the median accuracy was better than 1.12 BPM and 1.2 BPM for users 1 and 2, respectively. However, user 3's median accuracy was a bit higher (*i.e.* 1.488), while the 95th percentile confidence was as large as 1.95 BPM. These slight variations across different users and environments occur due to differences in physiques, respiratory system morphologies and environmental deployment conditions. For example, environments 2 and 3 both correspond to NLOS scenarios, which leads to relatively lower BPM accuracies. However, the level of robustness and accuracy Serene achieves is comparable to other commercial contact-less sleep monitoring products, and is enough for daily in-home use for gaining insights into sleep breathing patterns with minimal overhead using existing networking infrastructure.

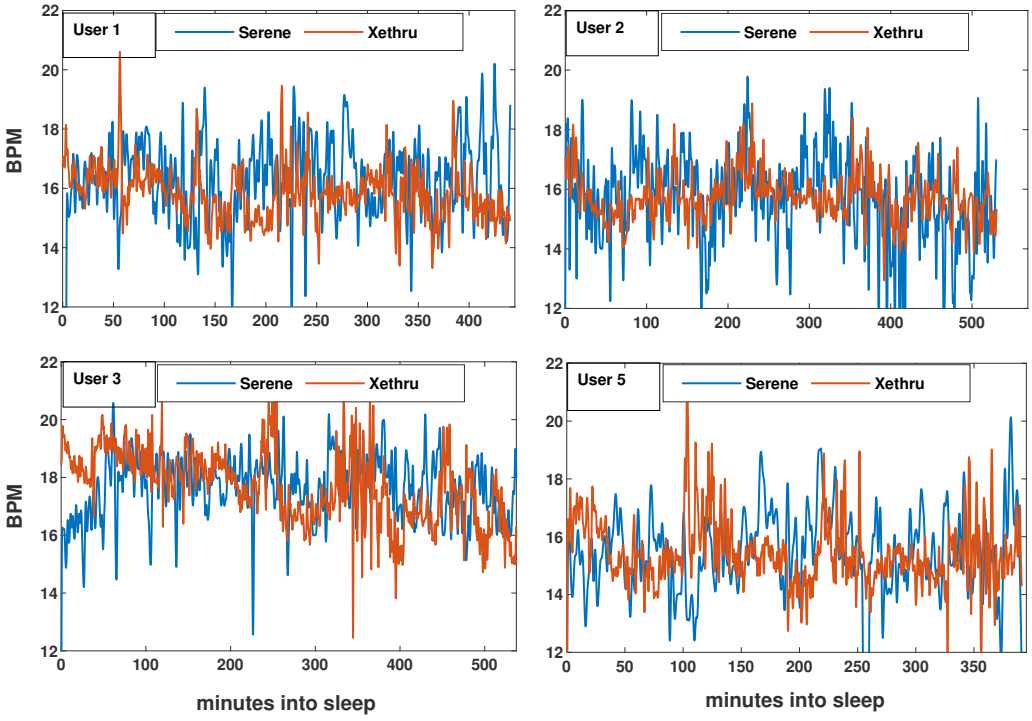
3.6.3.2 Short-term Accuracy

Serene can achieve an error of as low as 0.34 BPM during certain parts of a full-night sleep. In Fig. 3.8(c), we notice that there are certain time windows during the night where Serene matches



(a) CDF of overall and per-user BPM error.

(b) Sleep posture experiments.

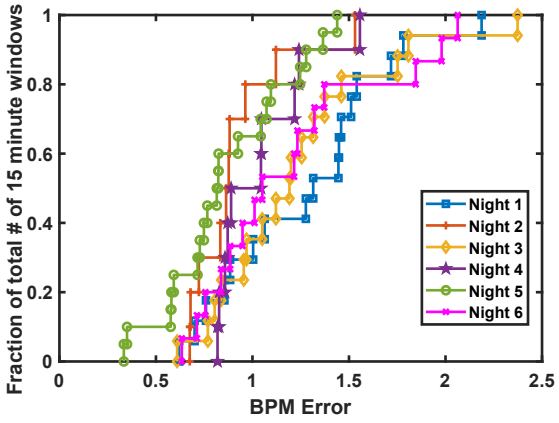


(c) Full night breath tracking.

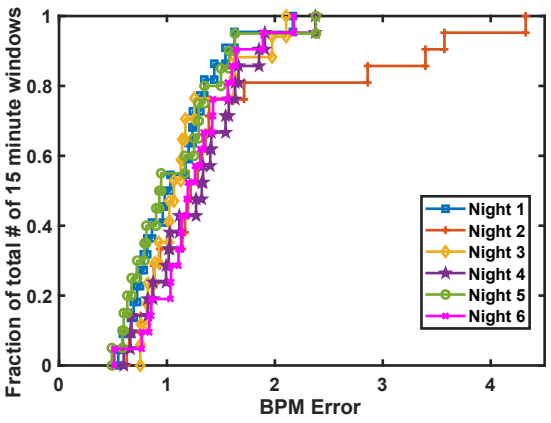
Figure 3.8: CDF of overall and per-user breathing rate MSE compared to a Xethru X4M200 ground truth; Serene's full-night breath tracking performance; and average BPM errors for short duration sleep experiments in different sleep postures

Xethru's performance very closely. To know how many times such time windows occur during different nights in our dataset, we divide each night into small 15 minute time windows and compute the MSE of per second BPM estimate in those windows. Figure 3.9 show the CDF plots for 6 different full-night sleep experiments corresponding to users 1, 2 and 3. From Fig. 3.9(a), we observe that in the case of User 1, Serene experienced a breathing rate error of only 0.34 BPM in

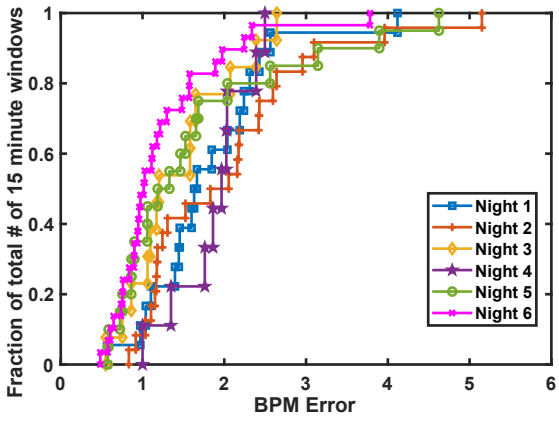
one 15 minute window during Night 6. Moreover, error in more than 50% of the time windows remained below 0.84 BPM during Night 6. Similarly, for other users, we observe that in multiple time windows during a full-night sleep, BPM error stays under 1 BPM. This shows that Serene does fairly well when compared to controlled short-duration sleep experiments performed in recent CSI based sleep monitoring studies. Also, figure 3.8(b) shows average BPM errors for controlled 10 minute sleep experiments in different sleep postures. We observe that Serene achieve an error of less than 1 BPM for most sleep postures even in NLOS scenarios. The errors were as low 0.55 BPM in LOS scenarios.



(a) User 1, 6 nights.



(b) User 2, 6 nights.

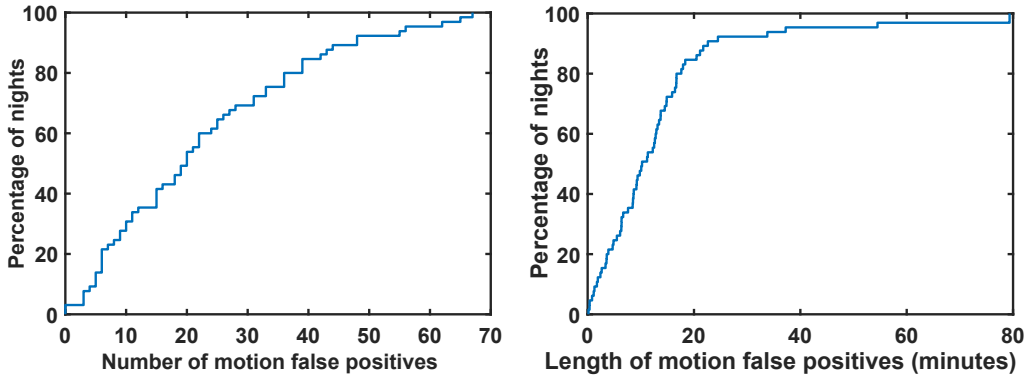


(c) User 3, 6 nights.

Figure 3.9: CDFs of BPM errors calculated over 15 minute windows for 6 different nights (Users 1, 2 and 3)

3.6.4 Naturally Occurring Motion False Positives

Serene experienced a median of 20 false positive limb/body motion events, which can be attributed to activity of other house residents while the user is sleeping. The total duration of such events stayed below 37 minutes more than 95% of the time. Radar and WiFi are both very sensitive to body motion. We observed from our experiments that whenever a user moves in his bed, both Serene and X4M200 successfully detect the motion event. However, we also observed scenarios where Serene detected body movements but the ground truth remained undisturbed (*i.e.* contained breathing signal only). We call such spurious movements detected by Serene as *motion false positives* (MFPs), which we attribute to other movements present in the environment (e.g. when one of the occupants wakes up to get water, etc.). To understand how significant such MFPs can be in real-world deployments of our WiFi CSI based sleep monitoring system, we evaluate two key metrics namely the *number* and *duration* of MFPs per night's sleep.



(a) motion false positives (b) duration of motion false positives

Figure 3.10: CDF's of numbers and total duration of motion false positives during a night when compared with X4M200 ground truths. Motion false positive naturally occur due to activity of other housemates

Figure 3.10 illustrates the CDFs of these two metrics evaluated over more than 65 nights in our database. We observe that our system detected less than 56 MFPs occurrences for 95% of tested nights 3.10(a). Moreover, when we observe that when MFPs occur, their collective duration remains bounded under 37 minutes for 95% of the time, as shown in figure 3.10(b)—which is a minuscule

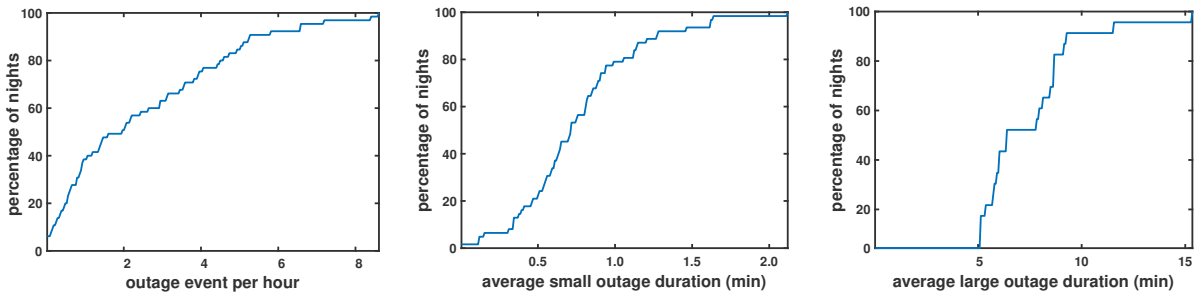
fraction of the entire 65+ night dataset.[2] Note that these MFPs do not signify any technical limitation of Serene, as such motions occur naturally in home environments. However, based on our results, we can conclude that the Serene will be able to successfully meet vital signs tracking accuracy requirements as expected from any other in-home COTS sleep monitor (e.g. X4M200, ResMed S+ [113], smart-watches, etc.), as the number and duration of naturally occurring interferences in WiFi signals due to activity of other residents is usually low during night time.

3.6.5 Breath Signal Outage

Serene experiences an average outage of less than 6.38 minutes, during which it cannot track any vital signs. As we discussed in 3.4.3, Serene identifies an *outage* event when power in the first PCA projection (*i.e.* the breath signal) corresponding to a sleep epoch goes below a threshold (*i.e.* noise floor). To assess our system's ability to continuously track vital signs (*i.e.* breathing and other limb/body activity) in the real-world, we measure Serene's per-night *outage* performance statistics. To achieve this, we follow the treatment of signal outage in wireless propagation literature. Specifically, we calculate two second-order statistics: level crossing rate (LCR), and average fade duration (AFD) [6]. LCR determines the rate at which outages occur during a full-night sleep, whereas AFD determines the duration of each outage. We analyze the LCR and AFD using the first PCA projection's power with respect to the noise floor. LCR and AFD have been extensively studied in body area network (BAN) literature owing to the complex and non-stationary way in which a human body interacts with the wireless channel [126]. Next, we present a summary of the aforementioned statistics derived from our entire dataset.

Figure 3.11(a) shows LCR or outage rate calculated per hour across our sleep dataset. We can observe that on average, the breathing rate estimation of participants experienced 2 outage events per hour. At 95 percentile confidence, outage amounted to less than 6.6 events per hour. However, in the context of sleep monitoring, a further piece of detail must be considered to fully understand outage events during sleep, *i.e.* the duration of such outages, which we characterize

[2] 65 nights with 7 hours average sleep equal 27300 minutes, making MFPs relative duration a mere 0.136%.



(a) Outage rate (b) small-scale outage duration (c) large-scale outage duration

Figure 3.11: Second-order statistics of breath estimation outage events. Outage rate and average outage duration mirror, respectively, their counterparts level crossing rate and average fade duration from wireless propagation literature

using AFD. Serene can experience two types of outage events: *small-scale* and/or *large-scale*. Small-scale outage may arise from users rolling over in bed to a different position or sleep posture while sleeping. This is because certain sleep postures may momentarily make it difficult for Serene to detect the breath signal due to weaker chest movements. In contrast, large-scale outage may occur when a user gets out of bed during the night, for example, to get water or go to restroom. To understand how such small- and large-scale outage events are distributed naturally in the real-world, we introduce a design threshold to separate the two types of outage events. From the analysis of our dataset, we set such design threshold to 5 minutes, where we consider outage events longer than 5 minutes as a large-scale outages and vice versa. Figure 3.11(b) elaborates on the statistical behavior of small-scale outage. On average, small-scale outage events lasted for 0.7 minutes, while the 95th percentile confidence outage duration is under 1.62 minutes. CDF of the duration of large-scale outage is shown in figure 3.11(c). Large-scale outage duration averaged around 6.38 minutes while its 95th percentile confidence is under 11.56 minutes, although durations in excess of 15 minutes can occur. We conclude our outage characterization by emphasizing that our current analysis is application and dataset specific. Many other factors such as participants' physiques (e.g. larger torso[3] area) and sleep patterns can be associated with both small- and large-scale outages. For example, a person suffering for *sleep apnea* may suffer multiple small outages thoughout his sleep. Our analysis provides a basis for sleep researchers to conduct larger and more representative studies

[3]It is beyond the scope of the article to delve into morphological factors contributing to breathing style variations within human populations.

on breath outage events using our proposed sleep monitoring scheme, as breath outages during sleep may indicate a possible disease in the user's respiratory tract, etc.

3.6.6 Sleep Insights

3.6.6.1 Sleep Quality

To motivate the merits of our accessible sleep monitoring scheme, we present a few interesting insights on sleep quality gained from our data collection campaign. Our findings motivate the usefulness of Serene in terms of tracking long term sleep quality and sleeping patterns. Next, we show how Serene can provide users with actionable feedback on a per-night basis towards the long-term tracking and management their sleeping habits. We perform these assessments using the CSI data corresponding to users 1 - 4.

Figure 3.12 shows three different metrics of sleep determined for 3 users over a period of more than 13 consecutive days, namely *sleep efficiency*, *aggregate motion* (in minutes) during sleep and *sleep length*. Sleep efficiency for each night of sleep was calculated using on our actigraphy based sleep scoring approach. As users manually started and ended each night's data collection using our software, the sleep lengths were easily determined according to those end points. We observe interesting insights for these long term sleep metrics. For instance, we can see that User 1 experienced a noticeably restless 9th night which resulted in poor sleep efficiency. User 4 only slept for 1.25 hours, but as he was awake for only 4.156 minutes during that time, his sleep efficiency reaches 95%.

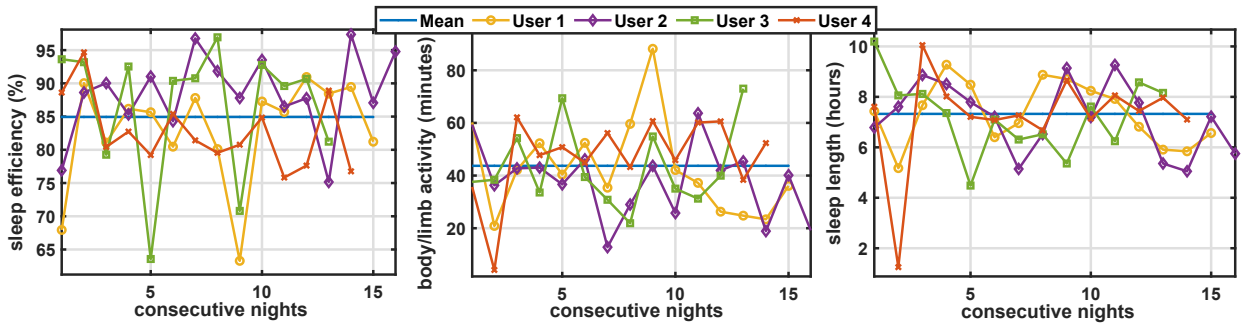
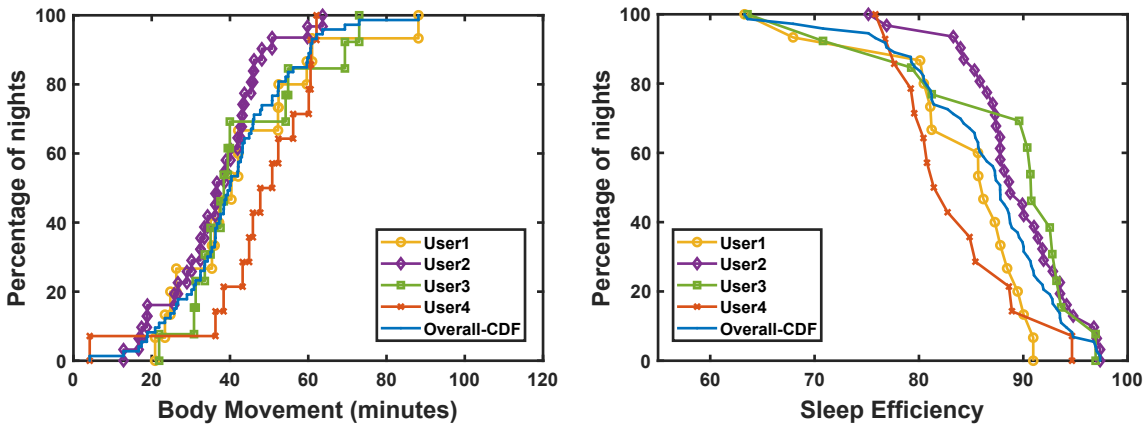


Figure 3.12: Sleep efficiency and body motion corresponding to 4 users and throughout 13+ consecutive nights



(a) Overall and per-user CDF for motion duration. (b) Overall and per-user CCDF for sleep efficiency

Figure 3.13: Overall and per-user CDF for motion duration and sleep efficiency

In terms of aggregate body motion statistics over nights and across subjects, we measured a median of 40 minutes with the 95th percentile being under 80 minutes as illustrated in the blue CDF in figure 3.13(a). On an individual basis, and considering user 2 and user 4 for instance, their median body movements were 36 minutes and 47 minutes, respectively. This insight is corroborated when inspecting the complementary CDF's depicted in figure 3.13(b). Specifically, while both users 2 and 3 have a comparable maximum sleep efficiency of 96%, User 3's sleep efficiency was lower than 80% on 3 different nights. Moreover, User 2 has a worst efficiency of 75%, whereas User 3 has worst efficiency of 63%. For the aggregate dataset, the median user population sleep efficiency was around 87%. The average sleep duration among these 3 users during this consecutive testing period was 7.32 hours. Note that the recommended sleep for ages 18-64 years is 7-9 hours [82].

3.6.7 Discussion

In this section, we provide commentary on the limitations of our work and discuss avenues of future research which we believe would be particularity exciting and fruitful.

Interpretable CSI dimensionality reduction. Our work is premised on the notion of subspace tracking and the ability to isolate instances of faint breathing from turbulent movements in the signal locale that would mask breathing. Incidentally, our sleep application requires a binary decision on subspace dynamics: quiet or turbulent. It would be interesting to perform the function of PCA

such that the resultant subspace can be interpreted in native ways that support atomic elements of inference i.e. require no training. Such putative subspace may facilitate more than binary decisions on the dynamics of the channel. As an example, it is shown in [11] that the CSI tensor can be "unfolded" as to expose the channel behaviour in space across antennae and in frequency across subcarriers. It is further shown that tracking the spatial subspace is indicative of multiuser activities. Returning to our sleep application, recasting the subspace in such terms may allow us to infer channel modulations from other users sharing a house and ultimately cancelling their masking contributions on the signal of interest.

Parameterization. Our proposed approach does not involve extensive environment- and/or participant-dependent parameterization. However, our aim is to develop techniques that can accurately and robustly monitor the primitives of sleep; namely, respiration and body movement vital signs. These techniques ought to perform reasonably well when compared to state-of-the-art, radar-based commercial sleep monitors. In order to achieve this objective, we have taken a cross-modal supervision approach, whereby we leverage the ground truth vital signs obtained from radar optimally parameterize our CSI processing pipeline. However, we stress that parametrization is only performed during the design phase and does not require calibration effort on the part of end-user during real-world deployments. We believe that it is possible to develop a machine learning (ML) architecture for CSI-based sleep tracking. With such an ML architecture, arriving at a transformation between the supervising modality (i.e. ground truth) and CSI is rendered automatic. However, questions pertaining to the requisite volume of data for such a task are open. A combination of (1) relatively benign urban channels, (2) an abundance of cellular basestation CSI data, and (3) superior MIMO resolutions, transpire in [130] as to afford the CSI channel sensing problem an intriguing and principled machine learning framework. Adapting such concepts within the context of the indoor CSI sensing channel and its peculiarities is an interesting future work direction.

Sleep stage classification. A more ambitious learning task pertaining to sleep monitoring is stage classification [18]. It has been shown in prior art using custom radar signalling that wireless sleep 4-stage classification can be achieved through a domain adaptation approach paired with ground

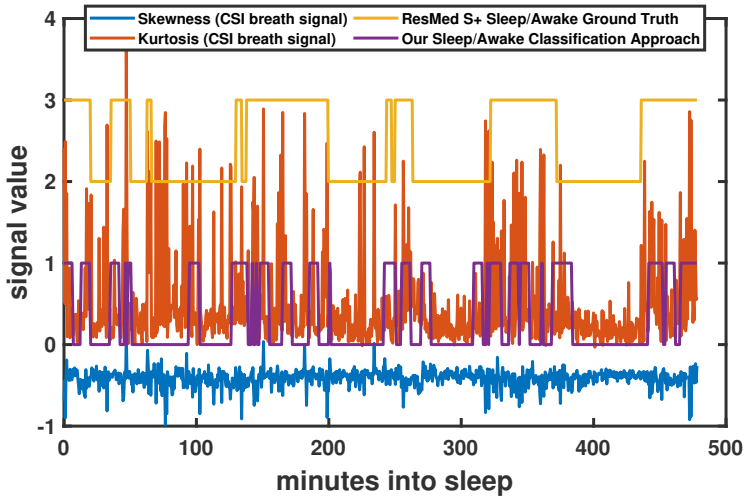


Figure 3.14: Two-stage (asleep versus awake) crude classification using simple feature engineering approach and as compared to classification from a commercial ResMed S+ device

truth from medical-grade devices [165]. The radar device we utilised in our sleep comparative study does not supply sleep stage classification ground truth. Nonetheless, we have demonstrated in this work the elements of sleep monitoring needed for such classification; namely, breathing and motion estimation. In order to investigate the potential extension of this work towards 4-stage sleep classification, we have conducted a limited pilot study using ground truth obtained from a commercial ResMed S+ device for sleep versus awake classification [113]. For demonstration purposed, we apply a simple feature engineering approach using statistical features derived from our respiration estimate. Figure 3.14 depicts a snapshot of such classification over an entire night. From close inspection, it is evident that this simple approach results in classification performance very close to that supplied by the ResMed S+ ground truth device. Therefore, advanced sleep-stage classification is possible by using both body movement and respiration based vital signs obtained from our system. We leave expanding this research strand for further work.

3.7 Conclusions

In this work, we design, implement, and evaluate Serene, a practical scheme for tracking respiration and body movement vital signs during sleep using WiFi CSI signals. Our approach is very easy to use and unobtrusive, as it is contact-less and uses exiting WiFi devices to sense

vital signs. We make three major contributions. First, we derive a closed-form expression which shows that robust breathing waveforms can be extracted using CSI signals independent of the router position and orientation as long as receiver is close to the subject. The insight gained by such expression enables us to develop our robust and generalizable CSI signal processing architecture to extract vital signs. Second, to accurately track variations due to breathing and body movements in the spatial-frequency subspace formed by the WiFi channel, we propose a novel multi-dimensional clustering subspace tracking technique. Our technique accurately detects and differentiates between waveforms corresponding to sleep vital signs. Third, we extensively evaluated our system in long-term experiments conducted in 5 different apartments, where we collected more than 550 hours (80 nights) of data from 5 individuals. Our system can track respiration throughout a night with an average accuracy of 1.19 breaths per minute (BPM) compared to state-of-the-art radar based sleep monitors, and experiences an average nightly outage of under 6.38 minutes only. Based on the interesting sleep insights we gained from our user study, we remark that our proposed system lowers the deployment barriers for in-home, long-term sleep quality monitoring. Such accessible sleep monitoring will not only help users improve their sleeping habits, but also potentially aid in the early identification of sleep-related disorders. In future, advanced machine learning algorithms can be developed to achieve multi-stage sleep classification using the body movement and respiration vitals signs obtained using Serene.

CHAPTER 4

MONITORING BROWSING BEHAVIOR OF CUSTOMERS VIA RFID IMAGING

4.1 Introduction

RFIDs based activity monitoring has recently emerged as a way to track customer activity in physical retail stores [41, 49, 79, 81, 122]. Acquiring customer activity information is important, as the amount of time that a customer spends on browsing an item is a key indicator of the amount of interest that the customer has towards the item. Manufacturers can use such information to improve the quality of their products, such as their visual attractiveness. Moreover, retailers can use such information for the strategic placement of retail items [19, 20, 61, 96]. However, existing RFID based systems for customer activity tracking in physical retail stores [41, 49, 79, 81, 122] have two key limitations. First, they require physical interactions with tagged display items for detecting human interest in places such as clothing stores [122]. Second, they do not work in multi-person environments, which are most common in reality. In contrast, we seek to leverage COTS RFID devices for monitoring *browsing activity* (*i.e.* when there is no physical interaction between customers and the display items) of customers in general retail stores. Such information is easy to obtain in online shopping environments by tracking customers' online browsing behavior, such as the amount of time spent on viewing a product or the number of clicks on a product, but it is difficult to obtain in physical shopping environments. Effective tracking of customer browsing activity in physical shopping environments will not only provide useful insights on customer behavior to product manufacturers and retail store managers, but can also help to shorten the gap between online shopping and physical shopping.

In this work, we propose *TagSee*, a multi-person activity tracking system based on RFID imaging. The hardware requirements of TagSee include a set of RFID tags and an RFID reader, both tags and the reader are COTS products. The tags are deployed on the boundaries of the shelves. The reader is deployed so that the customers are between the monitored shelves and the reader.

TagSee is based on the insight that when customers are browsing the items, as they stand between the tags and the reader, the multi-paths that the RFID signals travel along change, and therefore, both RSS and phase values of the RFID signals that the reader receives change as well. Based on these variations, TagSee constructs a coarse grained image of the customers and the tags using a model-driven deep learning framework. Afterwards, TagSee determines popularity of different item categories that are being browsed by the customers by analyzing the constructed images. Figure 4.1 shows an example of our RFID based customer monitoring system setup. The key novelty of this work is on achieving multi-person activity tracking in front of display items by constructing coarse grained images via robust, analytical model-driven deep learning based, RFID imaging using existing RFID devices and protocols. TagSee works for multi-person scenarios, works for scenarios where there is no physical interaction between customers and the display items, and is device-free (*i.e.* there is no need to attach anything to shelf items or customers).

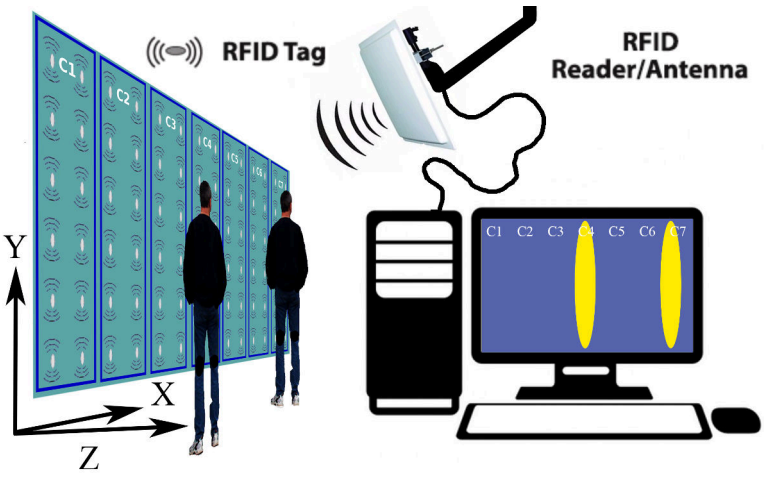


Figure 4.1: Example system setup

The first technical challenge is to robustly model the relationship between the signal attenuation caused by the human obstruction in RFID signals and the images of the human obstruction. This modeling is difficult for two major reasons. First, the interactions between human objects and RFID signals during a browsing activity are highly complex. Second, as we use *monostatic* RFID readers, which use a single antenna at a time for both transmitting and receiving RFID signals to and from the tags, modeling the impact of human obstructions on RFID signals becomes even more

difficult. This is because on any reader-tag-reader (TX-Tag-RX) path, the RFID signals experience two different attenuations due to an obstruction, once when signals are sent from the reader antenna to the tag, and the second when the tag backscatters those signals towards the same reader antenna. Employing geometrical and measurement models, such as the ones used in previous RF imaging techniques [95,133,152,166], will entail high dependency on multiple, environment dependent and difficult to tune parameters. Moreover, the imaging accuracy of such geometrical and measurement models based systems is not accurate and robust enough for imaging multiple customers showing interest in multiple different item categories simultaneously. Also, as interactions between human objects and RFID signals during a browsing activity are highly complex, an accurate mathematical is hard to derive. To address these challenges, we propose a model-driven *Deep Neural Networks* (DNNs) [43] based RF imaging approach. First, we mathematically formulate the problem of imaging human obstructions using monostatic RFID devices and derive an approximate analytical imaging model that correlates the variations caused by human obstructions in the RFID signals. Second, based on the derived imaging model, we develop a DNNs based deep learning framework to robustly image customers with high accuracy. Our key intuition is that by training our system with the images constructed based on RFID signals when humans are browsing items, our system can automatically learn the underlying relationship between those images and the observed RFID signal dynamics. Our DNN based approach is easy to realize in practice as it is environmental and hardware independent, and the thresholds and parameters are easy to tune. Moreover, our approach allows for robust imaging, even when customers are naturally moving to-and-fro or sideways while browsing or picking up/putting back items from a product category.

The second challenge is to enable multi-person imaging, but without changing the training requirements. That is, our system should only require the DNN to be trained for single-person scenarios, and it should not require the DNN to be trained for multi-person scenarios. This is because first, it is cumbersome to train the DNN with all possible multi-person scenarios, and second, it will lead to overfitting of the DNN. To address this challenge, we propose a spatial moving window based imaging technique to image multiple customers, who are browsing products

in different columns, simultaneously. The intuition is that, a single customer can significantly influence the RSS values of only a block of deployed tags (*i.e.*, the ones covered by the moving window), and that multiple customers maintain a distance between themselves while browsing any shelf. To achieve this, TagSee moves a window over the spatial distribution of tags, shifting it rightward, one column of tags at a time. For each instance of the moving window, TagSee replaces the observed RSS variations for the tags lying outside the moving window with random values, which are sampled from the Gaussian distributions defined by the mean and variance of RSS variations corresponding to those tags, observed during the calibration phase (*i.e.*, when there is no human obstruction around). Afterwards, TagSee constructs the images corresponding to each instance of the moving window, by using the modified RSS variation vectors, which consists of changes observed in the RSS values of every tag on the shelf, as the input to the DNN. Finally, it combines those images by applying the averaging filters to output the final image. Note that our approach does not require the exact locations of deployed tags and the reader antennas to be known in advance, which makes it easy to deploy in practice.

The third challenge in TagSee is to robustly quantify the variations introduced by human obstructions in the RSS values of the deployed tags. This is necessary because TagSee uses these RSS variations as inputs to its DNN for imaging obstructions. Anomalous variations in RSS values of different tags occur frequently during browsing behavior, either due to fading loss from constructive/destructive interference of RFID signals due to multipath effects, or due to the measurement noise of the RFID reader. To address this challenge, we leverage the *frequency hopping* (FH) capability of the multi-frequency UHF RFID hardware, which operates in the 902-928 MHz frequency range (divided into 50 closely spaced subcarriers). First, in scenarios where some part of RFID spectrum is under fade while a reader attempts to interrogate a tag, FH capability allows the reader to interogate that tag on stronger subcarriers, which helps TagSee gather enough measurements per tag per second, required for robust and accurate image reconstruction. Second, since the subcarriers are closely spaced in the frequency range 902-928 MHz, the impact of disturbances created by a human subject on the RSS values corresponding to a certain tag, is similar

across most subcarriers because transmitted power in all subcarriers are the same. Based on this intuition, TagSee robustly estimates the variations in RSS values of different deployed tags by taking the median of the RSS variations observed over multiple subcarriers. This reduces anomalies in the variations of RSS values, leading to significantly reduced distortions in the constructed images.

We implement TagSee system using a Impinj Speedway R420 reader and SMARTRAC Dog-Bone RFID tags. We attach RFID tags in a distributed and orderly fashion, just like a mesh, along the boundaries of a shelves, while covering all column-wise item categories. We call such tags which are attached to the shelves as *Static Tags*. In any monitoring scenario consisting of A number of RFID antennas, there are $A * K$ unique TX-Tag-RX links for K tags deployed along the boundaries of the shelf (so number of links $M = K$ in our case). We use Impinj Speedway R420 RFID readers, which are capable of reading upto $\sim 450\, tags/s$, which allows for gathering enough RSS and phase information from the deployed tags, required for smoother activity tracking. We create real-life scenarios to perform comprehensive experiments involving 10 different human subjects with IRB approval, and then evaluate the performance of TagSee on this data set. Our experimental results show that, on average, TagSee can achieve a true positive rates (TPR) of more than 90% and a false positive rates (FPR) of less than 10% using training data from just 2-3 users. Moreover, TagSee can achieve a TPR of more than 80% and a FPR of less than 15% in multi-person scenarios using training data from just 3-4 users.

The rest of the work proceeds as follows. In Section 4.2, we discuss related work. In Section 4.3, we give a brief overview of our TagSee system. In Section 4.4, we present the preprocessing techniques TagSee uses to prepare data for constructing images. In Section 4.5, we first mathematically formulate the problem of imaging humans using monostatic RFID devices and derive an approximate analytical imaging model that correlates the variations caused by human obstructions in the RFID signals. Based on the analytical model presented in Section 4.5, we develop our deep neural networks based RFID imaging framework in Section 4.6. In Section 4.7, we first extend our DNNs based imaging approach to image multiple persons simultaneously, and then present our customer activity tracking technique, which uses computer vision algorithms to determine popularity scores

of different item categories. Finally, we give concluding remarks in Section 6.9.

4.2 Related Work

4.2.1 Radio Tomographic Imaging (RTI)

Previously proposed RTI approaches, which are closest to our work, are either based on sensor networks [133, 152] or bistatic passive RFID (pRFID) systems [144]. In the sensor networks based approaches [133, 152], each node deployed around a monitored area is capable of both transmitting and receiving RF signals, independently, and for any TX-RX pair, there is exactly one communication link which gets affected by an obstruction. In contrast, the bistatic passive RFIDs based system proposed in [144] entails two communication links per tag read, i.e. a TX-Tag link from the TX antenna which interrogates the tag, and a Tag-RX link where the RX antenna receives the response back from the tag (we refer to RFID links as TX-Tag-RX in rest of the work). However, both of the aforementioned RTI scenarios are similar, because in each case, RF signals experience attenuation due to an obstruction only once on their way to the receiver side. Moreover, all previous RTI schemes have an inherent issue of high dependence on multiple, difficult to tune parameters, such as the parameters corresponding to different types of geometrical and measurement models, which are used to capture the effects of attenuation due to human obstructions [133, 152]. First, these parameters are often highly dependent on experimental scenarios and the hardware being used. Second, these parameters have to be tuned manually, which is time consuming, and often requires intensive calibration for each different deployment scenario. Third, inefficient tuning of these parameters leads to unstable and ineffective imaging results. Moreover, all the previous RF imaging approaches require the exact locations of all RF nodes to be known in advance. The aforementioned limitations make previously proposed RTI techniques practically difficult to realize. Compared to these previous RTI works, we develop a model-driven deep learning based imaging scheme for monostatic RFID systems, which gets rid of manual setting of difficult to tune RFID channel parameters, and enables accurate multi-person imaging using monostatic passive RFID hardware.

4.2.2 Customer Behavior Monitoring using RFIDs

RFID based techniques for human activity tracking in physical retail [41, 49, 79, 81, 122] utilize variations in received signal strength (RSS) and phase values of RFID signals to monitor customer behavior. However, most RFID based behavior monitoring techniques such as [49, 79, 122] only focus on clothing stores. Moreover, the current RFID based techniques are highly parameter dependent and fail to work well in multi-person scenarios. Moreover, these techniques require all retail items in a shelf to be tagged with RFID tags, a requirement which is often not satisfied in many retail stores. To the best of our knowledge, there is no prior work that can monitor customer browsing activity without using cameras or the requirement of physically touching retail items tagged with RFIDs.

4.2.3 Customer Behavior Monitoring using Cameras

Several camera based solutions to customer behavior analytics exist in literature [21, 30, 34, 66, 74, 78, 105, 108, 109]. A key advantage of RFID based solutions compared to dense deployment of cameras is that RFID based solutions are not privacy intrusive. In the past, multiple privacy concerns related to camera based solutions have been brought to light, for example with the 'Amazon Go' stores that use dense deployment of cameras to monitor customers [132, 137]. However, please note that even though our system has a privacy advantage, we do not seek to replace any existing camera based solutions. Instead, we envision that TagSee can be integrated with existing camera based techniques for improved customer behavior analytics and enhanced shopping experience solutions such as automatic check-out [57]. Such integration of RFID and camera based solutions will also help reduce privacy concerns by allowing the retail stores to enable smart solutions with less number of cameras. The goal of this paper is to advance the state-of-the-art in the emerging field of RFIDs based customer behavior analytics in retail stores by leveraging off-the-shelf RFID readers and tags for monitoring browsing activity of customers (i.e. when there is no physical interaction between customers and the display items). To the best of our knowledge, there is no prior work that can monitor customer browsing activity without using a camera or without the requirement of first

attaching tags on the retail items with RFIDs and then physically touching those items.

4.3 System Overview

In this section, we first give a brief overview of monostatic passive RFID (pRFID) technology. Second, we briefly discuss TagSee's imaging infrastructure. Third, we give an overview of TagSee's multi-person imaging and tracking scheme.

4.3.1 Monostatic Passive RFIDs

In a monostatic pRFID system, a *reader* transmits continuous wave signals to interrogate tags deployed in its proximity, and then receives backscattered signals from those tags which contain their unique IDs. In this work, use industrial standard, EPC Global Class 1 Generation 2 (C1G2) RFID [38] compatible, Ultra-High-Frequency (UHF) pRFID tags and Impinj Speedway R420 monostatic RFID readers in our proposed popularity tracking scheme. The RFID readers we use operate in the frequency range 902-928 MHz, through a *frequency hopping* (FH) mechanism, where the frequency range is divided into 50 subcarriers (*i.e.* 902.75 - 927.25 MHz with an interval of 0.25 MHz), which are randomly hopped between during each interrogation cycle. This FH capability reduces interference between nearby RFID readers, and leads to robust and reliable interrogation of tags in cases where some part of the spectrum is under fade. Also, these readers are equipped with multiple antennas. The query-response communication corresponding to each antenna is multiplexed in time, where each antenna interrogates the tags in an alternating manner. In each query-response communication, the EPC C1G2 compatible RFID tags respond to RFID signals from a reader, through a random access collision avoidance technique called *slotted* ALOHA [62]. The RFID readers we use are capable of reading upto $\sim$ 450 $tags/s$, which allows for gathering enough RSS and phase information from the deployed tags, required for smoother and near real-time popularity tracking.

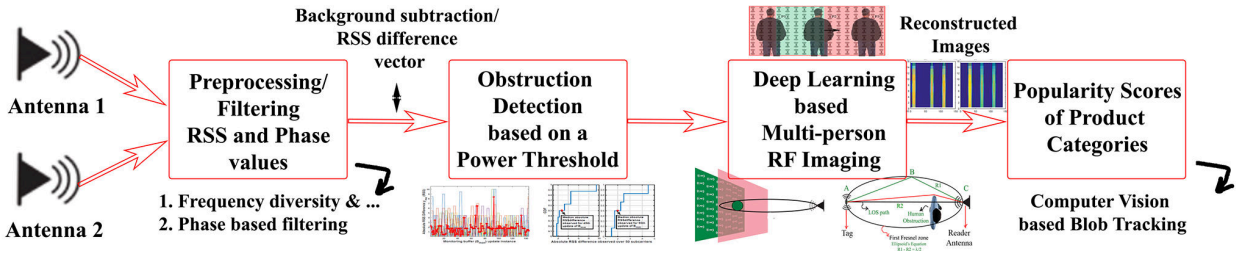


Figure 4.2: High level flow diagram of TagSee's monitoring mode

4.3.2 TagSee's Imaging Infrastructure

TagSee requires items to be placed in column-wise categories on shelves. TagSee also assumes that RFID tags are already attached in a distributed and orderly fashion, just like a mesh, along the boundaries of different shelves, while covering all item categories. We call such tags which are attached to the shelves as *Static Tags*. Although, TagSee does not require the items placed in the shelves to be tagged, yet, for the sake of completeness, we call such tags as *Dynamic Tags*, because their positions can change with time or they can even disappear from their shelves in cases where those items get purchased. Figure 4.1 shows the layout of RFID infrastructure used by TagSee. In any monitoring scenario consisting of A number of RFID antennas, there are $A * K$ unique TX-Tag-RX links for K tags deployed on a shelf (so number of links $M = K$ in our case). To track activity of customers in front of a shelf or a display item, an RFID *reader* transmits continuous wave signals to interrogate tags deployed in its proximity, and then receives backscattered signals from those tags which contain their unique IDs. Afterwards, the RSS and phase values corresponding to these tags are leveraged to image any customers standing between the tags and the reader. Our analytical model based monostatic RFID imaging approach requires the locations of all static tags and RFID reader antennas to be known in advance. However, our final deep learning based imaging approach does not have any such requirement. Next, we present a system level overview of TagSee's activity tracking scheme.

4.3.3 Overview of TagSee Imaging and Tracking Scheme

TagSee consists of two working modes, namely *calibration* mode and *monitoring* mode. In calibration mode, TagSee reads RSS and phase values from the deployed tags when there is no human obstruction in the monitored area. These calibrated RSS and phase values are used for background subtraction during the *monitoring* mode. In *monitoring* mode, TagSee constructs images after processing the RSS and phase values it reads from the deployed tags. Figure 4.2 shows a system level diagram of TagSee for *monitoring* mode. In the first step, the raw RSS and phase values obtained from the deployed tags are fed into a pre-processing module. The pre-processing module first applies a combination of moving average and moving median filters on streaming RSS and phase data, and then subtracts the calibrated RSS and phase values from the incoming filtered RSS and phase values, respectively. Afterwards, it filters out the anomalous variations in RSS values by applying *phase difference* and *frequency diversity* based filtering techniques. In the second step, the power of resultant RSS difference vectors obtained from pre-processing module is checked with a threshold, to determine the existence of obstruction in front of the deployed tags. If any obstructions are detected, the RSS difference vectors are then fed into a DNN based multi-person RFID imaging module, which then constructs coarse-grained images of the detected human obstructions. Finally, TagSee applies a computer vision based blob tracking technique on the constructed images, to track the browsing activity near different items, by determining the popularity scores of different item categories. Next, we discuss how TagSee processes the RSS and phase values during its two working modes.

4.4 Preprocessing RSS and Phase

In this section, we describe the pre-processing techniques TagSee uses during its two working modes, *i.e.*, *calibration* and *monitoring* modes, to prepare data for TagSee's imaging module. Next, we first introduce the RFID signal parameters which TagSee pre-processes.

Received Signal Strength (RSS): Monostatic RFID channel is a double fading channel, *i.e.* each fade is experienced twice, once in the forward link and once in the reverse link. A typical RFID use

case scenario involves indoor multipath environment, where each link consists of a line-of-sight (LOS) path and a few major reflections. For Mono-static RFID readers, for a given subcarrier of wavelength λ, the received power $P_r^{\{k,a\}}$ at the reader antenna a, of the backscattered signal from tag k located at distance d from the reader, can be approximated in terms of transmit power P_t for a free space scenario as:

$$P_r^{\{k,a\}} = P_t G_a^2 G_k^2 T_b^{\{k\}} \cdot (\frac{\lambda}{4\pi d})^4 \tag{4.1}$$

where G_a and G_k are a^{th} reader antenna and k^{th} tag antenna gains, respectively [98]. T_b^k is the backscatter or modulation loss of the k^{th} tag. Next, we assume that P_t, G_k, G_a and $T_b^{\{k\}}$ remain constant for a tag-antenna pair, and then re-write a simplified logarithmic relation for RSS at k^{th} tag as follows:

$$RSS^{\{k\}}\ (dBm) = A_0^{\{k\}} + 10 \cdot \beta^{\{k\}} \log[(\frac{\lambda}{4\pi d})] \tag{4.2}$$

where $A_0^{\{k\}}$ is assumed to be a constant for a specific environment, and the value of $\beta^{\{k\}} = 4$ in case of LOS free space path loss scenario depicted in Eq. (4.1). In reality, $\beta^{\{k\}}$ is dependent upon indoor multipath environment and shadowing effects due to obstructions. We use Eq. (4.2) while formulating our analytical RFID imaging approach.

Phase: For RFID propagation environment involving monostatic readers, the phase information of a received signal, received from k^{th} tag, provided by the reader can be written as:

$$\phi = mod(\phi_p + \phi_o + \phi_b^{\{k\}}, 2\pi) \tag{4.3}$$

where $\phi_p = 2\kappa d + \phi_m$ ($\kappa = \frac{2\pi f}{c}$, f = signal frequency and ϕ_m phase contribution due to constructive/destructive interference due to multipaths), ϕ_o is the phase offset which includes phases of the cables and other reader and antenna components, and $\phi_b^{\{k\}}$ is the backscatter phase of the k^{th} tag modulation. Next, we discuss the techniques we use in TagSee's *calibration* and *monitoring* working modes to pre-process RSS and phase data before feeding it to the imaging module.

4.4.1 Calibration Mode

In calibration mode, TagSee reads RSS and phase values from the deployed tags when there is no human obstruction in the monitored area. These calibrated RSS and phase values are used

for background subtraction during the *monitoring* mode. During this mode, TagSee keeps reading the RSS and phase values for $t_{cal} \approx 2\ minutes$, to ensure that it receives enough readings from each of the deployed tags and carrier frequencies used by the reader during frequency hopping. For F frequencies and K tags, TagSee records FK RSS and phase vectors, per RFID antenna. In the end, TagSee applies a combination of moving average and moving median filters (window size = 5) to all FK number of RSS and phase vectors, calculates the median of each of those FK filtered vectors, and records $F \times K$ dimensional RSS and phase calibration matrices (M_{rss}^{cal} and M_{phase}^{cal}), corresponding to each of the A antennas, for background subtraction during image construction in the monitoring mode.

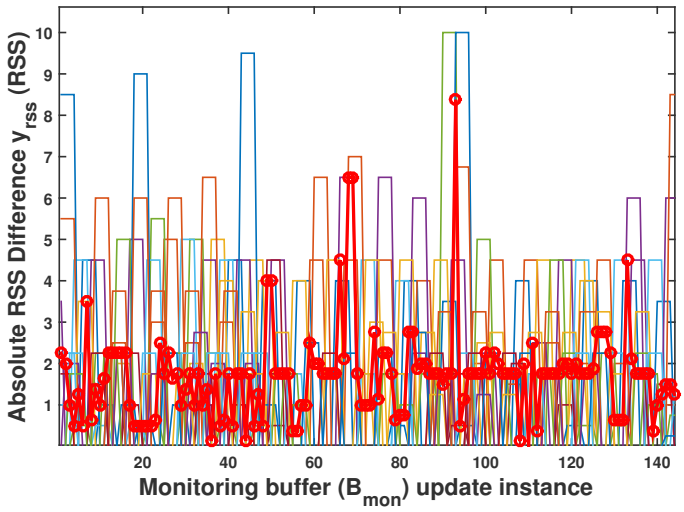
4.4.2 Monitoring Mode

In *monitoring* mode, TagSee constructs images after processing the RSS and phase values it reads from the deployed tags. During this mode, TagSee keeps reading the RSS and phase values, while maintaining the streaming values in a buffer B_{mon} for batch processing. In our experiments, we observed that every user takes at least $3-4\ secs$ while browsing a certain item category, which amounts to approximately $N_{mon} = 2000$ RSS and phase readings. Therefore, during monitoring mode, TagSee maintains latest $N_{mon} = 2000$ readings in buffer B_{mon}. B_{mon} is updated with new readings every $t_{mon} = 1\ secs$, which amounts to approximately 450-500 readings. As all tags contend for the medium through a random access protocol, TagSee might not receive readings for some tag-frequency pairs in a period of $3-4\ secs$. For utilizing frequency hopping capability of RFID readers efficiently during the frequency diversity based filtering process that we discuss later on, TagSee waits until it gets enough readings from the deployed tags for multiple different frequencies. We experimentally observe that the above values of N_{mon} and t_{mon} allow for recording enough readings for robustly constructing reasonable images. However, we also observed that during a browsing activity, it often happens that the data obtained during a certain time window does not contain data from all the deployed tags. Advanced *matrix completion* algorithms [65, 112] can be used to interpolate missing RSS data, however it will significantly increase the computational

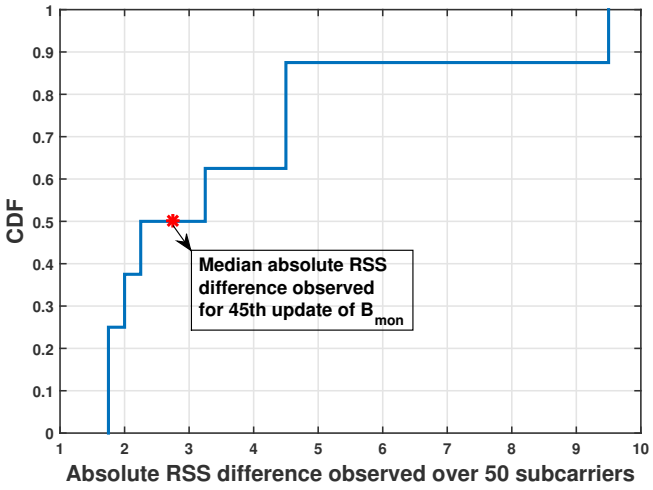
complexity. Therefore, in the case where TagSee does not find any reading in B_{mon} for a certain tag-frequency pair, it replicates the calibrated RSS and phase readings corresponding to that tag-frequency pair. Finally, TagSee applies a combination of moving average and moving median filters (window size = 5) to all FK number of RSS and phase vectors contained in B_{mon} corresponding to each antenna. Next, we describe how TagSee leverages the frequency diversity and phase difference to calculate a robust estimate of RSS difference vector $\mathbf{y_{rss}}$ for each RFID antenna.

Phase Difference based Filtering: To filter RSS based on phase difference, TagSee first calculates phase difference vector for each of the FK phase vectors contained in B_{mon} by subtracting the calibrated phase values in M^{cal}_{phase} corresponding to each of the FK frequency-tag pair, to obtain B'_{mon}. Next, TagSee leverages the concept of Fresnel zones [54] to filter out RSS values in B_{mon}. The first Fresnel zone determines the LOS path between two RF nodes, and encompasses most of the RF wavefronts which contribute significantly to RF propagation. Therefore, if the first Fresnel zone is clear of obstructions, we can assume LOS communication between a tag and its reader. For monostatic RFIDs, the phase difference between the direct LOS path, and any other RF propagation path lying within the first Fresnel zone, can be at max 2π, as show in 4.4(b), which corresponds to one wavelength. Assuming that the phase values obtained for each tag-frequency pair during calibration phase (M^{cal}_{phase}) correspond to the direct LOS paths between those tags and their corresponding antenna, TagSee discards all RSS values in B'_{mon}, for which the absolute difference between their phase readings and the corresponding calibrated values in M^{cal}_{phase}, is less than $\phi_{mon} = \pi/4$.

We chose $\phi_{mon} = \pi/4 = 0.125 \times 2\pi$ because we want to select those RSS values for imaging, which correspond to the scenarios where a human subject obstructs at least 6.25% of the Fresnel zone. As R420 reader only provides phase values in the range $0 - 2\pi$, phase wraps may occur, which can lead to improper phase difference based filtering. However, first of all, the phase difference between two RF propagation paths lying within the first Fresnel zone can be at max 2π. Second, neither the tags nor the RFID reader antennas move during the experiments. Therefore, TagSee assumes that phase difference between the phase values read during calibration and the ones read



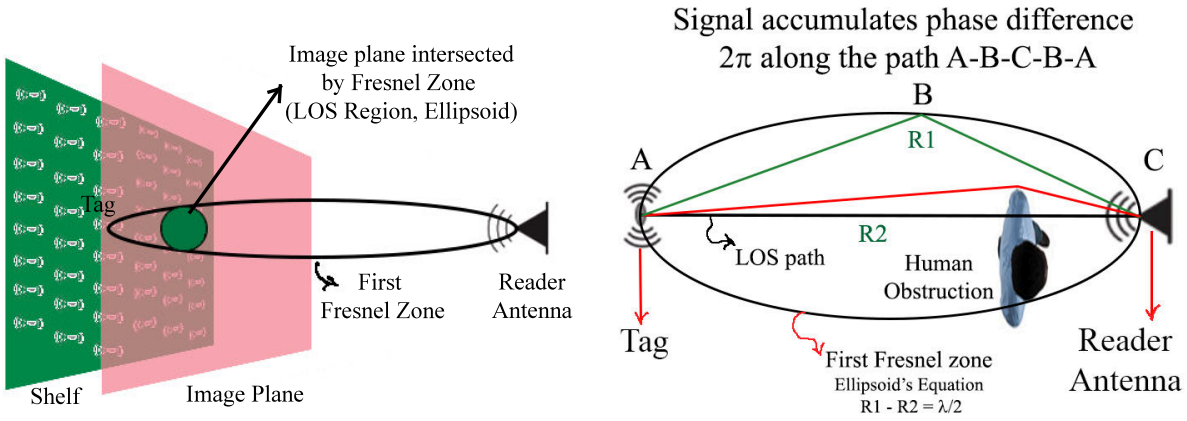
(a) y_{rss} filtered using phase values & frequency diversity (bold-red) is plotted over RSS differences obtained for 50 subcarriers



(b) CDF of absolute RSS difference values obtained from 46th tag for 50 subcarriers

Figure 4.3: Phase and frequency diversity based filtering for a tag obstructed by a human (head & arms allowed to move)

during monitoring mode remains below 2π, and does not take into account phase wraps of over 2π. Note that, a few longer paths, *i.e.* paths outside the first Fresnel zone, may exist at any point in time. However, we assume that the phase values corresponding to those paths are filtered out by aforementioned moving average and moving median filtering. After applying this filter on readings obtained from each antenna separately, TagSee calculates the median of each of those FK filtered vectors, and records $F \times K$ dimensional RSS matrix (M_{rss}^{mon}) corresponding to each of the A antennas.



(a) Image plane intersecting a Fresnel zone

(b) Impact of human obstruction on paths

Figure 4.4: Intuitions behind using the concept of First Fresnel zones [54] for phase based filtering & imaging

Frequency Diversity based Filtering. To calculate $\mathbf{y_{rss}}$, TagSee first calculates the absolute difference between calibrated and monitored RSS matrices, *i.e.* $Y_{rss} = |M_{rss}^{cal} - M_{rss}^{mon}|$, which gives an $F \times K$ dimensional RSS difference matrix. Next, TagSee leverages the frequency diversity of the RFID system we use, to reduce anomalous variations in Y_{rss}. Since the frequencies which RFID reader randomly hops between are closely spaced in the frequency range 902-928 MHz, the RSS difference observed for a tag over these multiple closely spaced frequencies, due to impact of human obstruction in a small time window of $3-4$ $secs$, is similar, given that the trasnmit power in each subcarrier is the same). Therefore, TagSee takes the median of RSS difference values in Y_{rss} over $F = 50$ different frequencies, to obtain the K dimensional RSS difference vector $\mathbf{y_{rss}}$, which is then fed into the imaging module, after every $t_{mon} = 1$ $secs$. Again, this filtering is performed for each antenna. Figure 4.3 shows the filtered RSS difference values, plotted in bold-red color over the RSS difference values corresponding to all 50 subcarriers, for a slightly blocked tag by a human standing at the same place, but allowed to move head and arms. We can observe that the filtered RSS difference values remain at a relatively stable level over time, as compared to the unfiltered RSS difference values showing the varying nature of individual subcarriers.

Power based filtering before RFID Imaging. Before feeding the filtered $\mathbf{y_{rss}}$ into imaging module, TagSee checks if the changes observed in RSS values of different tags deployed along a shelf

are significant enough to reveal existence of an obstruction. If there are no significant changes observed, then imaging should be skipped, which reduces unnecessary computations during image construction. To achieve this, TagSee compares the power contained in the $\mathbf{y_{rss}}$ vector with a simple to tune threshold ($\mathbf{P_{rss}} = 10$), and discards any $\mathbf{y_{rss}}$ vectors which do not meet this threshold. Note that, the aforementioned phase based filtering of RSS values, along with this power threshold based filtering of $\mathbf{y_{rss}}$ vectors, filter out the variations in RFID channel which do not correspond to a obstruction, which leads to more accurate imaging.

4.5 Analytical RFID Imaging Approach

In this subsection, we propose an analytical imaging approach for monostatic RFID systems, which forms the basis of our proposed *deep neural networks* (DNNs) based imaging technique in §4.6. We explain our model using Fig. 4.4(a). The green surface represents the surface of the shelf on which RFID tags are deployed. In a typical browsing scenario, customers will be standing closer to the shelf. Our aim is to image the customers while they stand close to the shelf for browsing the items displayed on that shelf. To develop our analytical model, we first erect an imaginary image plane at a point P_z along Z-axis, parallel to the X-Y-axes as shown in Fig. 4.4(a). Next, we divide the image plane into *voxels*, such that there are p_x *voxels* between each pair of tags along X-axis, p_y *voxels* between each pair of tags along Y-axis, and the coordinates of bottom leftmost *voxels* and top rightmost *voxels* are $(0,\ 0)$ and $(Max_x,\ Max_y)$, respectively ($p_x = p_y = 5$ in our current implementation, with inter-tag distance of 5 $inches$ along both axes). Our imaging problem can then be represented as a linear system of equations, where the changes in RSS of different links $\mathbf{y_{rss}}$ can be written in terms of the changing attenuations $\mathbf{x}$ and an $M \times N$ weight matrix $\mathbf{W}$ (where M = Number of links, N = Number of *voxels* in the image plane), specifying the contributions of each link towards the changes observed in attenuation at each different *voxel*, as $\mathbf{y_{rss}} = \mathbf{Wx} + \mathbf{n}$, where $\mathbf{n}$ corresponds to fading and measurement noise. The goal is to solve the system for each antenna separately, using the RSS difference vectors for each tag-antenna pair.

For the system represented by $\mathbf{y_{rss}} = \mathbf{Wx} + \mathbf{n}$, we first model the weight matrix $\mathbf{W}$ by employing

the concept of *Fresnel Zones* [54] between two nodes of a RF link, and use imaginary ellipsoids centered at the locations of different RF nodes in the network, to determine weights of different *voxels* in the image plane covering the monitored area. The intuition is, that these ellipsoids determine the LOS path (typically chosen to be the first Fresnel zone) of each respective link, and if a *voxel* is intersected by the LOS path of a link, it will be assigned more weight as compared to the *voxel* which does not fall in LOS. Figure 4.4(a) shows an example scenario. To construct image from the RSS difference values obtained for all the links (*i.e.* $\mathbf{y_{rss}}$), the most straightforward way is to calculate the least-squares solution as $\mathbf{x_{LS}} = \mathbf{P} \times \mathbf{y_{rss}}$, where $\mathbf{P} = (\mathbf{W}^T\mathbf{W})^{-1}\mathbf{W}^T$. However, the matrix $\mathbf{W}$ is not full rank in case of imaging systems, which makes imaging an *ill-posed inverse problem*. To handle ill-posedness, we use *Tikhonov Regularization* approach (such as the one proposed in [152] for minimizing objective function of the original problem.

Estimating Change in Path Loss: Monostatic RFID communications make the task of estimating weight matrix $\mathbf{W}$ more difficult, because signals experience attenuations multiple times before reaching back to the same receiver, *i.e.* once when signals are sent to the deployed tags, and second when those tags backscatter those signals back to the reader antenna. Assuming that both forward and reverse channels of a TX-Tag-RX link are symmetric, we assume an imaginary set of symmetric ellipsoids (approximate LOS regions) between each tag and reader antenna pair. Intuitively, the image plane cuts the Fresnel zones between each TX-Tag-RX link. Weights are assigned to each *voxel* of an image plane, based on whether they fall inside the imaginary ellipsoid of the TX-Tag-RX link or not. Given the value of free space path loss exponent for a TX-Tag-RX link corresponding to each tag k is 4, we can assume that during calibration phase, the initial value of $\beta_{init}^{\{k\}} = 4 + \phi_{init}^{\{k\}}$. Similarly, let us assume that for each new update of the buffer B_{mon} during the monitoring mode, the new $\beta_{new}^{\{k\}} = 4 + \phi_{new}^{\{k\}}$. Both $\beta_{init}^{\{k\}}$ and $\beta_{new}^{\{k\}}$ are unknown here, however, as will show next, only the difference $\beta_{new}^{\{k\}} - \beta_{init}^{\{k\}}$ is required. At any time instance, we can write this change in path loss exponent in terms of the change in RSS for tag k (i.e. $y_{rss,k}$) at the RFID reader antenna as follows:

$$\triangle \beta^{\{k\}} = \beta_{new}^{\{k\}} - \beta_{init}^{\{k\}} = \frac{y_{rss,k}\ (dBm)}{10 \cdot \log[(\frac{\lambda_{avg}}{4\pi d})]} \tag{4.4}$$

As the RFID reader uses multiple frequencies, we use average of those frequencies and choose $\lambda = \lambda_{avg}$ for simplicity. Given the fact that these frequencies are close to each other, the choice of this value of λ does not hurt the results of this scheme. After calculating $\triangle\beta$, TagSee assigns weights to each *voxel* on the image plane as:

$$w_{kj} = \begin{cases} \frac{1}{d^{(4-\triangle\beta)}}, & \text{if } d_{kj,1} + d_{kj,2} < d + \Theta \\ 0, & \text{otherwise} \end{cases}$$

Here d is distance between the reader and tag for a link k, $d_{kj,1}$ is the distance from center of pixel j to the tag and $d_{kj,2}$ and is the distance from center of pixel j to the reader antenna corresponding to link k. Θ is a parameter describing ellipsoid's width. Using the general equation for calculating the first Fresnel zone radius at any point in the middle of the link, we chose Θ to be as follows:

$$\Theta = \Theta_0 \times \sqrt{\frac{\lambda_{avg} \cdot d_{kj,1} \cdot d_{kj,2}}{d_{kj,1} + d_{kj,2}}}$$

where we chose Θ_0 is an environment and RFID infrastructure dependent parameter, which we tune to achieve reasonable imaging results. Finally, TagSee employs regularization techniques mentioned in [144, 152] to construct image by determining attenuation values per *voxel* (*i.e.* $\hat{\mathbf{x}}$) as $\hat{\mathbf{x}} = \mathbf{P} \times \mathbf{y_{rss}} = (\mathbf{W}^T\mathbf{W} + \sigma_N \mathbf{C_x}^{-1} + \alpha(\mathbf{D}_X^T\mathbf{D}_X + \mathbf{D}_Y^T\mathbf{D}_Y))^{-1}\mathbf{W}^T\mathbf{y_{rss}}$, where $\mathbf{D_x}$ and $\mathbf{D_y}$ are differential operators along X and Y directions respectively, catering for the "spread" of the impact of attenuations in RSS values along these axes. We tune $\alpha = 15$ in our current implementation, for reasonable imaging results . $\mathbf{C_x}^{-1}$ is another prior term, which controls the spatial correlation of the impact of RSS attenuations across neighboring pixels of the image. Although we do not know $\hat{\mathbf{x}}$ in advance, we approximate $\mathbf{C_x}$ based on a exponential spatial attenuation model [10] [114], as $\mathbf{C_x} = \frac{\sigma_x}{\delta} \exp^{-\mathbf{D_p}/\delta}$, where $\mathbf{D_p}$ is a square form distance matrix, containing distance between each pair of pixels along the imaginary image plane, and σ_x is prior pixel variance due to human motion. σ_N is the prior variance of noise in pixel values *i.e.* when there is no human motion. We approximate σ_N as function $c_n * \sigma_{\mathbf{y}}$ of variance in RSS values during the calibration phase. We approximate pixel variance σ_x as function $c_x * \sigma_{\mathbf{y}}$ of variance in RSS values during the monitoring phase. We tune the values of c_n, c_x and δ for the best results.

Imaging: For imaging, TagSee erects multiple image planes along Z-axis and then takes an average of the images corresponding to all planes. We tune the locations of these image planes for best results. Finally, TagSee combines the images obtained from all antennas by taking their average. *While evaluating TagSee's performance, we compare the imaging performance of this analytical measurement model based imaging approach with our DNN based approach.*

Next, we propose our DNN based imaging approach, which not only gets rid of manual tuning of parameters in our analytical imaging approach, but also enables multi-person imaging that is required for monitoring browsing behavior of multiple customers towards different item categories simultaneously.

4.6 Deep Learning based RFID Imaging

In our deep learning based RFID imaging approach, we aim to solve the linear regression problem posed in the equation $\mathbf{y_{rss}} = \mathbf{W}\mathbf{x} + \mathbf{n}$ by modeling it as a *deep regression* problem. Deep regression techniques have been shown to yield state-of-the-art results without having to resort to more complex and ad-hoc regression models [73]. To achieve this, we model $\mathbf{x} = \mathbf{P} \times \mathbf{y_{rss}}$ as a DNN, where $\mathbf{y_{rss}}$ corresponds to the input of first layer, $\mathbf{x}$ corresponds to the output of last layer and the image construction matrix $\mathbf{P}$ corresponds to the combination of all the layers in between input and output. The intution is that if we train our system with approximate images of how a human obstacle should look like while he browses a item category, it can automatically learn the underlying relationship between those images and the RFID channel dynamics observed during that browsing activity, which is otherwise difficult to model through geometrical or measurement models based approaches.

Choice of Layers in Neural Network. We chose the DNN layers such that the weights and biases learned at different layers of the network can mimic the impact of human obstructions on the RFID signals. Based on the solution to our analytical imaging approach, we can see that image construction matrix $\mathbf{P} = (\mathbf{W}^T\mathbf{W} + \sigma_N \mathbf{C_x}^{-1} + \alpha(\mathbf{D}_X^T\mathbf{D}_X + \mathbf{D}_Y^T\mathbf{D}_Y))^{-1}\mathbf{W}^T$ captures linearities as well as non-linearities related to the impact of human obstructions in terms of attenuations introduced in

the RFID channel. The non-linearities are introduced by terms in weight matrix $\mathbf{W}$, which models the exponential attenuation of RFID signals with distance, and the terms in correlation matrix $\mathbf{C_x}$, which models how the impact of attenuations due to human obstructions decays spatially along the 2D image plane. The linearities are introducted by the terms corresponding to linear differential operators $\mathbf{D_x}$ and $\mathbf{D_y}$, as well as, due to the inherent linear nature of our imaging problem, *i.e.* all different matrices are connected through basic linear operations such as multiplication, addition, and inverse. For DNN based imaging, we design the input RSS difference vectors $\mathbf{y_{rss}}$ and the output image vectors $\hat{\mathbf{x}}$ to be normalized vectors, containing values between 0 and 1. We normalize the input $\mathbf{y_{rss}}$ vectors by dividing with the maximum observed RSS difference, which we empirically estimate for the system. Moreover, the image construction matrix $\mathbf{P}$ can consist of both positive and negative values, due to the inverse operations involved in it.

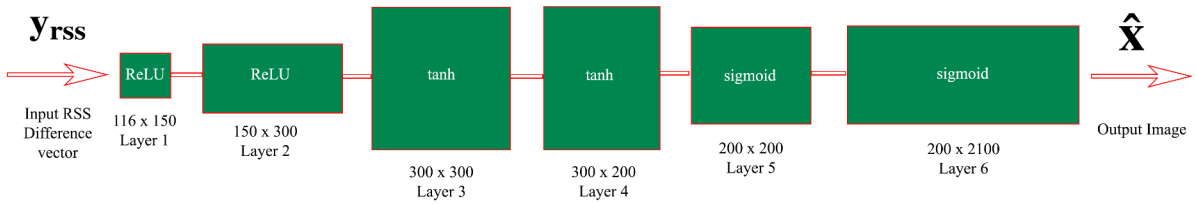


Figure 4.5: DNN architecture for $K = 116\ tags$, $k_x = 29\ tags$, $k_y = 4\ tags$ and inter-tag distance of 5 $inches$ along both axes

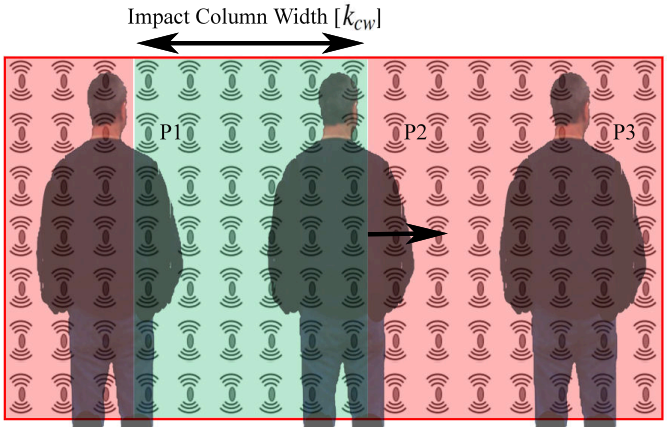
Based on the aforementioned nature of underlying parameters, we chose the first two layers of the DNN to be *rectified-linear* layers (ReLU), next two layers to be *tanh* layers and the last two layers to be *sigmoid* layers. The output of a ReLU layer is always non-negative real numbers, the output of a *tanh* layer remains between -1 and 1, and the output of a *sigmoid* layer always stays between 0 and 1. We propose to generalize the dimensions of aforementioned layers as $\{K \times \frac{3}{2}K\}$, $\{\frac{3}{2}K \times 3K\}$, $\{3K \times 3K\}$, $\{3K \times 2K\}$, $\{2K \times 2K\}$ and $\{2K \times p_x p_y (k_x - 1)(k_y - 1)\}$, respectively, where $K = 116$, $k_x = 29$, and $k_y = 4$ in our current implementation of TagSee. The number of DNN layers, as well as the dimension of each DNN layer, can be tuned empirically to achieve lowest *cross-validation* errors, and based on the intuition that the amount of information is limited by the number of RFID tags, *i.e.* any constructed image is basically an extrapolation of the impact of RSS variations observed for K tags. For example, in our proposed DNN architecture, we keep

the dimensions of all layers to be within 3 times the number of tags being used for imaging. Finally, to prevent DNN from over-fitting, we use L2 regularization in combination with *dropout* [129], where the dropout rate (*i.e.* the probability to retain a DNN unit during training) is 0.7. Figure 4.5 shows the visual representation of the DNN that we design for our current implementation.

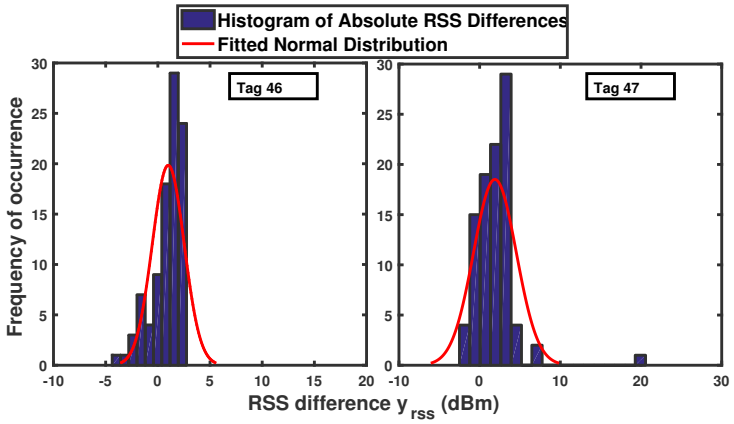
Training Requirements. We separately ask a couple of volunteers to browse each different item category by standing in front of them, for approximately $30 - 60$ $secs$. *We do not constrain natural human motion during this training phase,* i.e.*, the individuals browsing different item categories are allowed to move to-and-fro and sideways, and browse items in a natural manner.* TagSee uses the variations observed in RSS values of the deployed tags during this training phase as inputs to its DNN, after normalization. For each browsing activity during the training phase, we also generate approximate normalized images of human obstructions, which TagSee uses as the training "labels" or regression outputs to its DNN. In this work, we approximate the images of human subjects as ellipses, where the width and height of those ellipses is chosen as average width and height of the human subjects used for training TagSee's DNN. The height of the ellipses is limitted by the size and location of the deployed mesh of tags. For robustness, we design TagSee's DNN classification module as an ensemble of D single DNN classifiers. The final output is median of the outputs obtained from all the D single classifiers.

4.7 Multi-Person RFID Imaging

To develop our multi-person imaging technique, TagSee leverages the intuition that a human subject will only impact k_{cw} columns of tags along X-axis, for any deployment of a mesh of tags. Based on this intuition, TagSee creates multiple new vectors $\{\mathbf{y^i_{rss}}\}$ from each new RSS difference vector $\mathbf{y_{rss}}$ corresponding to an antenna, by moving a window of size k_{cw} columns over the spatial distribution of deployed tags, as shown in figure 4.6(a). For each window i, TagSee only copies those values from $\mathbf{y_{rss}}$ to $\mathbf{y^i_{rss}}$ which correspond to the tags contained inside that window, and replaces the values corresponding to remaining tags with two times the standard deviation values ($2 \cdot \sigma_{k,a,rss}$) of Gaussian distributions $\mathcal{N}(\mu_{k,a,rss}, \sigma^2_{k,a,rss})$, which model the RSS difference values observed for those tags, respectively, when there are no obstructions around.



(a) Spatial moving window of impact width $k_{cw} = 6$



(b) Distribution of $\mathbf{y_{rss}}$ during *calibration mode*

Figure 4.6: TagSee's spatial moving window based approach for multi-person imaging

This avoids spurious blobs during image construction. Figure 4.6(b) shows the distribution of the RSS difference values, obtained for two closely spaced tags, during calibration mode in one of our experiments, over a period of $\sim$ 2 $mins$. We can observe that the RSS difference values approximately follow a Gaussian distribution. The mean and variance values for the aforementioned distributions $\mathcal{N}(\mu_{k,a,rss}, \sigma^2_{k,a,rss})$ corresponding to each possible tag-antenna pair, are estimated during the calibration phase.

For k_x columns of tags, TagSee first generates $k_x - k_w + 1$ new vectors $\{\mathbf{y^i_{rss}}\}$. Afterwards, it constructs images corresponding to all vectors $\{\mathbf{y^i_{rss}}\}$ using the aforementioned DNN based imaging technique, and then merges them after passing through a 2D filter (median and averaging filters), to output the final image. TagSee applies this multi-person imaging technique for each antenna, separately, and finally combines the images obtained from all antennas through averaging. This

approach enables multi-person imaging without changing the training requirements of TagSee's imaging technique, *i.e.* our system does not require the DNN to be trained for multi-person scenarios. The only change required is to train DNN for all possible $\mathbf{y^i_{rss}}$ corresponding to every training sample. *Note that for any deployment, we train TagSee's DNN with data corresponding to 'no obstruction' scenarios as well to avoid detection of spurious blobs during image construction.* To train the DNN for 'no obstruction' scenarios, the input vectors are set to be the $2 \cdot \sigma_{k,a,rss}$ values corresponding to each deployed tag and the outputs are set to be zero vectors (i.e. blank images).

Monitoring Browsing Activity: TagSee monitors the customer browsing behavior towards different items in terms of popularity of those items. In monitoring mode, TagSee feeds the final constructed image frames to a *Blob Analysis* module [86], which determines the background using a few initial frames, and then outputs the coordinates of bounding boxes and centroids of any human images it detects in foreground of each consecutive frame. As the boundaries of item categories are known in advance, TagSee determines the *popularity* of each category by checking the proximity of the centroids corresponding to detected blobs in each frame to the centroid of that category. If the centroid of a blob is within η_2 *voxels* of the centroid of j^{th} category, TagSee increments the popularity $\mathcal{P}_j$. η_1 and η_2 is dependent upon the density of deployed tags, and can be easily tuned empirically for a certain deployment scenario. For robust popularity estimates, TagSee maintains a buffer consisting of 5 latest constructed images (which corresponds to a period of $\sim 3-4$ $secs$), takes the median of all those images and applies thresholding (on the scale from 0 to 1, pixel values below 0.1 are set to 0), before calculating $\mathcal{P}_j$'s.

4.8 Implementation & Evaluation

We implement TagSee using COTS UHF Impinj R420 pRFID reader [55] and SMARTRAC's Dogbone pRFID tags [125], which operates in frequency range 902.75 - 928.25 MHz and is compatible with EPC Global C1G2 [38] standard. We use two circular polarized antennas, which are connected to two of the four antenna ports of R420 reader. As reader interrogates the deployed *static* tags, the information containing IDs, time stamps, channel frequencies, reader antenna IDs, RSS values and phase values corresponding to the tags read in each cycle are sent through Ethernet

to a laptop running TagSee. We develop our RFID data collection module by bulding upon the JAVA based Impinj Octane SDK [56]. For any deployment scenario, TagSee first runs in *calibration mode*, for approximately 2 $mins$, to determine the background values of RSS and phase for all deployed tags. Afterwards, it turns on its *monitoring mode* to image customers and track popularity of different item categories being browsed by those customers.

4.8.1 Evaluation Methodology

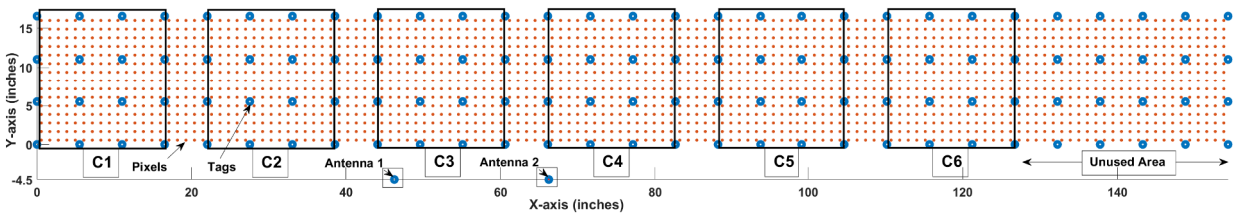


Figure 4.7: Detailed experimental setup

4.8.1.1 Experimental Setup

Fig. 4.7 shows detailed experimental setup of tags and reader antennas that we use to test TagSee. We deploy a total of $K = 116$ tags on a wooden shelf (each tag is first pasted on a sticky post-it note which is then attached to the shelf). We deploy $k_x = 29$ and $k_y = 4$ tags along X and Y axes, respectively, with an inter-tag distance of 5 $inches$ along both axes. The area between the tags is divided into *voxels* or image pixels, such that there are 5 *voxels* between each pair of tags deployed along both axes. The *voxels* are shown by red dots in Fig. 4.7. We place two reader antennas 13.78 ft away from the shelf, with an inter-antenna distance of 20 $inches$ as shown in Fig. 4.7. Tags are attached such that the distance of the first row of tags is 3.5 ft from the ground. Both antennas are placed parallel to the shelf, such that the distance of their centers is 3.5 ft from the ground. Moreover, as imaging is based on obstruction of LOS paths between tags and the antennas, we pointed both antennas towards the deployed tags. We mark 6 item categories on the shelf, where each category is covered by 4 separate columns of tags. Note that the exact dimensions of the setup are only required by our analytical imaging approach that derives our DNN based approach and

serves as its comparison metric. Our DNN based approach just requires that the locations of tags and antennas do not change after training as that will require retraining the DNN.

4.8.1.2 Data Collection

For data collection, we recruited 10 users who volunteered to provide data for our project. From 5 of those users, we were able to collect a total of 14617 samples (2500+ samples per person). We use this data to train TagSee's DNN model because we assume that it is big enough to capture the diversity of browsing movements of those users (who had different body widths and heights) reasonably well. However, due to time limitation, the remaining 5 users could only provide us with 2413 samples (< 500 samples per person). As the data obtained from these users is limited, we use their data for testing TagSee's performance. Hence, we test TagSee's performance using unseen data (i.e. data from the users who are not used for training TagSee's DNN), which makes our evaluation more robust. Note that the users in our study had different body widths and heights. However, an evaluation of the impact of such variations on TagSee's performance is out of the scope of this work and left as part of future work. In this work, we only focus on coarse-grained imaging of customers in front of the shelves.

4.8.1.3 Performance Metrics

Except for the scenarios where we compare TagSee's imaging performance, we evaluate TagSee's popularity tracking performance for any experiment using TPRs, FPRs and Miss Rates (MRs), which are calculated based on the correctness of item popularities $\mathcal{P}_j$ TagSee determines in different time windows. True positives correspond to the scenarios during which TagSee is able to detect interest in the categories being tested. However, TagSee may wrongly detect interest in categories (i.e. other than the ones being tested) as well, which correspond to false positive scenarios. TagSee *misses* when it is unable to detect interest in the tested categories during a time window (i.e. MR = 1 - TPR). Our goal is to achieve maximum TPRs and minimum FPRs.

4.8.2 Single Person Imaging Scenarios

TagSee can achieve TPRs of more than 90% and FPRs of less than 5% for single person monitoring scenarios. Figure 4.8 compares the imaging results of these two approaches, where TagSee constructs images of a user as he stands in front of each different item category, using 2 antennas. For this experiment, TagSee used a DNN trained for 3 volunteers, where the selected volunteers did not include the tested user. We can see that the images constructed by DNN based RFID imaging (bottom) are highly accurate as compared to the ones constructed using baseline approach (top). This is because, first, our DNN approach automatically tunes all values in the image construction matrix for minimum construction errors. Second, the DNN based approach is less vulnerable to natural human motions during browsing activity, which happens because it takes such variations due to motion into account during the training process. Next, we show how TagSee's performance in single person scenarios is affected by number of training users, impact width and number of antennas.

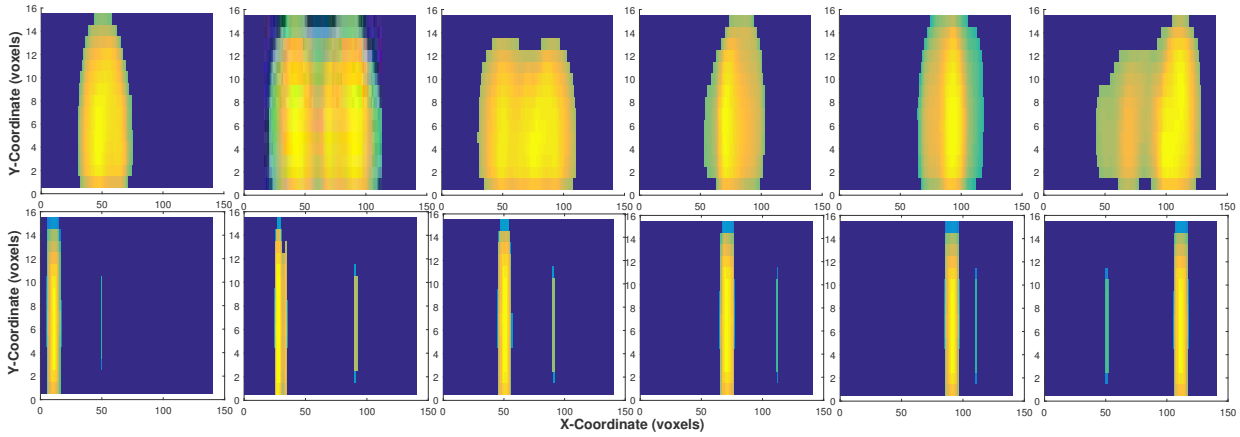
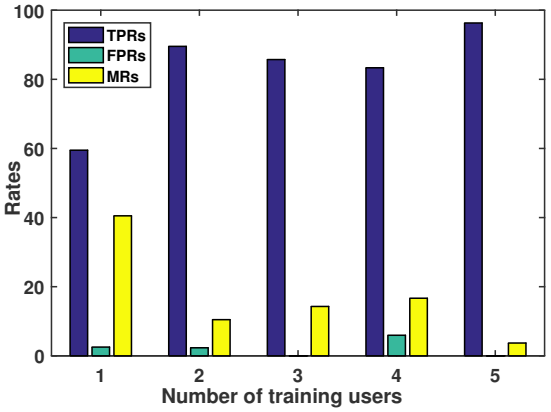


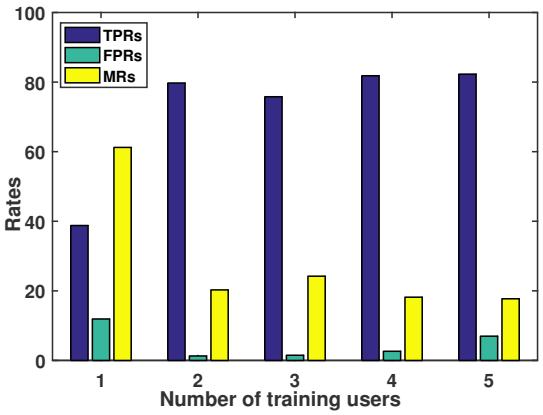
Figure 4.8: Comparison between TagSee's baseline (top) and DNN based (bottom) approaches for single person scenario

4.8.2.1 Effect of the number of training users

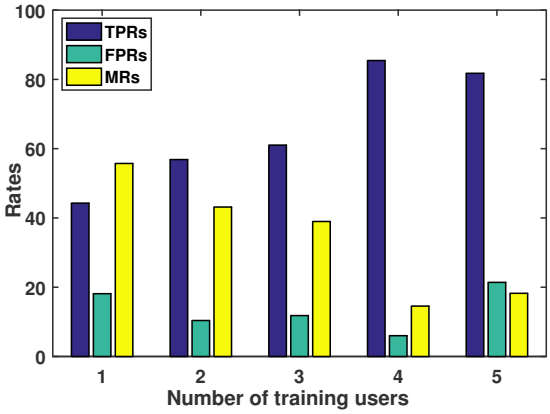
Figure 4.9 shows the impact of number of training users on TPRs, FPRs and MRs for 3 different experiments, performed for 3 of the 5 volunteers selected for testing. Impact width was set to $k_{cw} = 6$, and the error rates reported were averaged over 6 different item categories. Moreover,



(a) Average error rates for test user 1



(b) Average error rates for test user 2



(c) Average error rates for test user 3

Figure 4.9: Effect of number of training users on TagSee's performance (TPRs, FPRs and MRs) for $k_{cw} = 6$

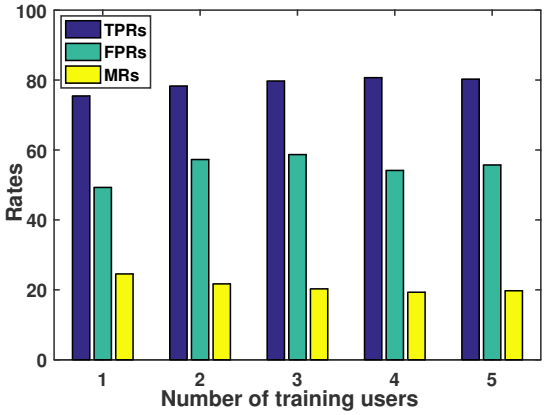


Figure 4.10: Performance in single person monitoring scenario using 1 reader antenna only, $k_{cw} = 8$

data from 2 RFID reader antennas was used for constructing images in these experiments. We can observe an increasing trend in TPRs for all three users 4.9(a)-4.9(b), which is intuitive. This

happens because as TagSee's DNN is trained with more scenarios, corresponding to different users of different shapes and sizes, its image construction becomes more accurate and robust, leading to higher detection rates. This is also the reason behind the decreasing trend in MRs, which we can observe for all three users 4.9(a)-4.9(b). However, we see that FPRs do not show an expected decreasing trend, which may seem counter-intuitive at first. This happens because when a customer is browsing an item category, our multi-person imaging algorithm sometimes wrongly detects and images human presence in front of nearby item categories as well, which leads to spurious popularity counts. In this scenario, TagSee can achieve TPRs of more than 85% and FPRs of less than 15%, averaged over all users and categories.

4.8.2.2 Effect of the number of reader antennas

In the aforementioned experiments, we were using 2 reader antennas. Figure 4.10 shows the error rates achieved using single reader antenna, for $k_{cw} = 8$ and different number of training users. We observe that the trend in TPRs and MRs remains similar to the ones corresponding to the 2 antenna scenarios (Fig. 4.9(a)). We observe that the average FPR drastically increases to more than 45%. This is because, excluding images from one of the antennas leads to ineffective filtering (as mentioned in § 4.7) of consecutive image frames, which may contain spurious popularity counts for untested categories.

4.8.3 Multi-Person Imaging Scenarios

TagSee can achieve more than 90% TPRs, and less than 10% FPRs for 2 person monitoring scenarios. Moreover, for 3 person monitoring scenarios, TagSee can achieve more than 80% TPRs, and less than 20% FPRs. Figure 4.11 shows some selected imaging results for these two approaches, which were constructed using 2 reader antennas, where test users performed browsing activities in front of item category sets {1,3}, {1,6}, {3,6}, {4,6}, {1,4,6} and {2,4,6} respectively. We can observe that TagSee's DNN based RFID imaging approach produces accurate images even for multi-person monitoring scenarios as well. We chose 3 of the 5 test users for TagSee's multi-person

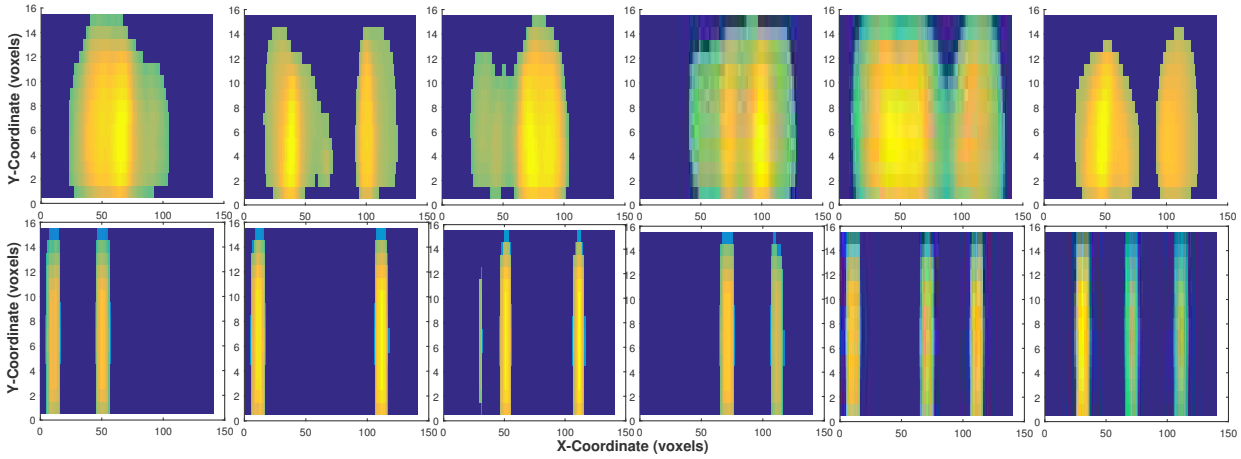
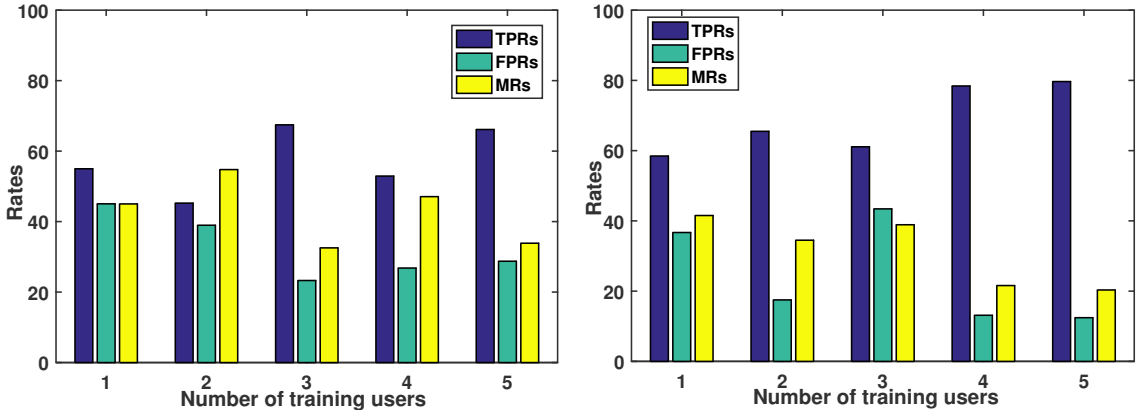
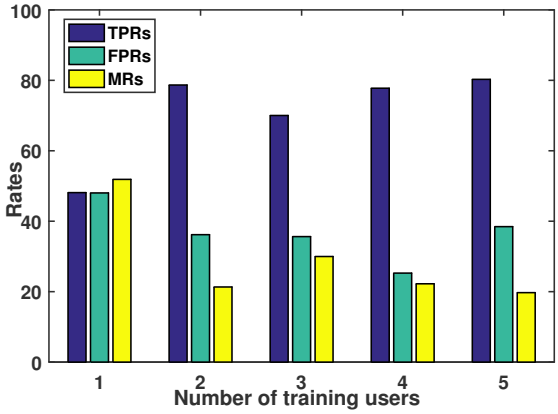


Figure 4.11: Comparison between TagSee's baseline (top) and DNN based (bottom) RFID imaging approaches for multi-person scenario. The leftmost 4 images correspond to 2-user scenarios, and the rightmost 2 images correspond to 3-user scanerios



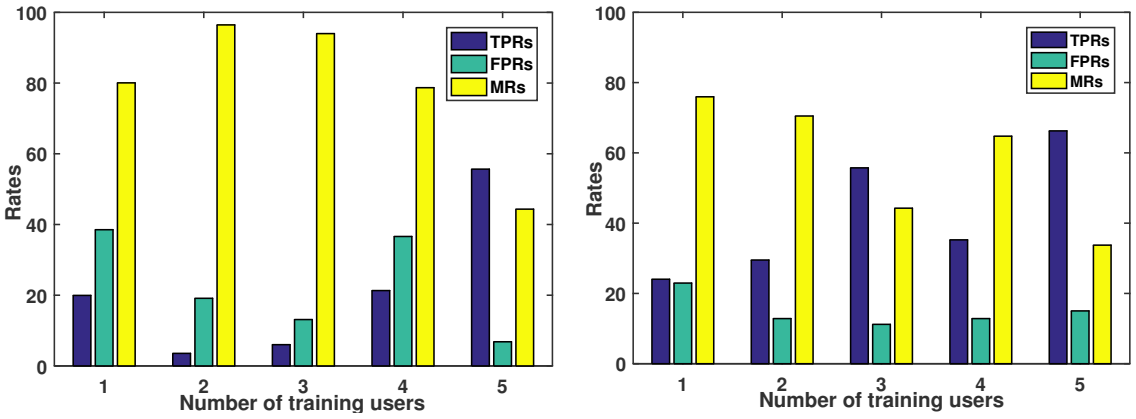
(a) Overall average error rates for $k_{CW} = 4$

(b) Overall average error rates for $k_{CW} = 6$



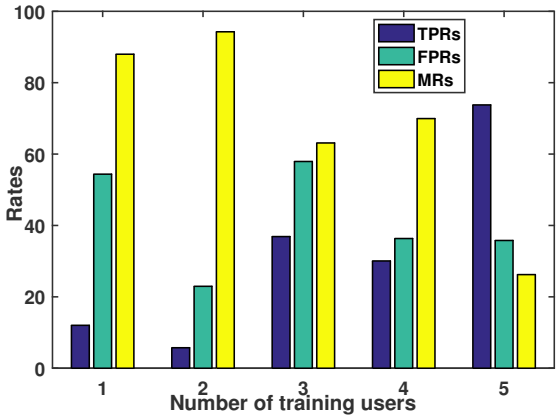
(c) Overall average error rates for $k_{CW} = 8$

Figure 4.12: Effect of impact width k_{CW} and number of training users on TagSee's performance in 2 person scenarios



(a) Per scenario error rates for $k_{CW} = 4$

(b) Per scenario error rates for $k_{CW} = 6$



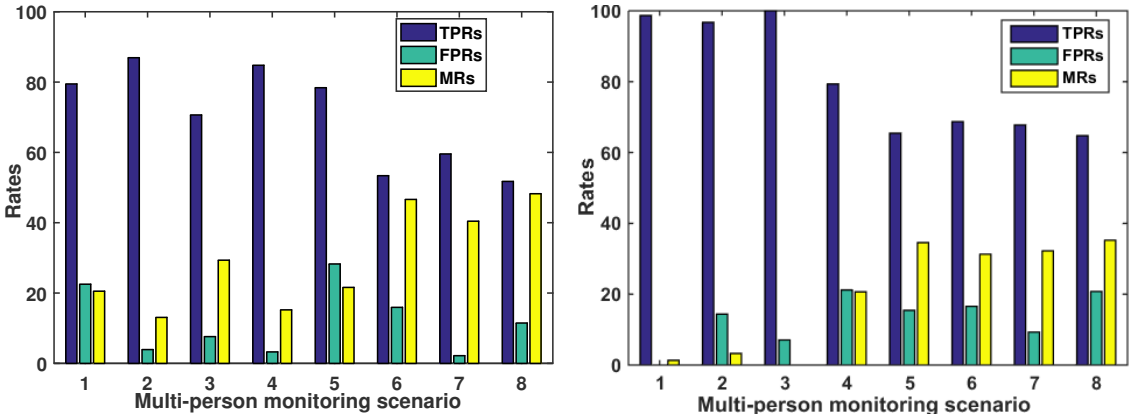
(c) Per scenario error rates for $k_{CW} = 8$

Figure 4.13: Effect of impact width k_{CW} and number of training users on TagSee's performance in 3 person scenarios

performance evaluation. Also, we kept the number of reader antennas $A = 2$ for robust imaging.

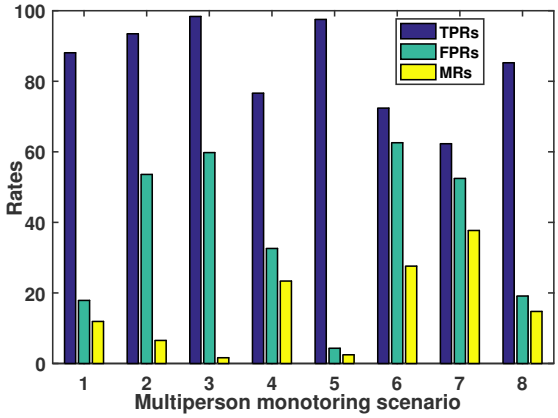
4.8.3.1 Effect of the number of training users

Here, first we discuss performance for 2 person monitoring scenarios i.e. corresponding to the category sets {1,3}, {1,4}, {1,5}, {1,6}, {3,6} and {4,6}. Figures 4.12(a)-4.12(c) show how TagSee's average performance changes with number of training users, for three different values of k_{CW}. We can observe an increasing trend in TPRs (decreasing MRs). We also see a decreasing trend in FPRs, but just like for single person monitoring scenarios, it is not consistent. For example, in case of $k_{CW} = 8$, FPRs decrease as training users increase from 1 to 4, but we see an increase in FPRs for the case corresponding to 5 training users. We attribute such unexpected changes in FPRs



(a) Per scenario error rates for $k_{cw} = 4$

(b) Per scenario error rates for $k_{cw} = 6$



(c) Per scenario error rates for $k_{cw} = 8$

Figure 4.14: Effect of impact width k_{cw} on TagSee's performance (TPRs, FPRs and MRs) for 8 different multi-person scenarios, i.e. item category sets {1,3}, {1,4}, {1,5}, {1,6}, {3,6}, {4,6}, {1,4,6}, {2,4,6}, 5 training users used

to spurious popularity counts. Second, we discuss performance for 3 person monitoring scenarios i.e. corresponding to the category sets {1,4,6} and {2,4,6}. Figures 4.13(a)-4.13(c) show TagSee's average performance for different number of training users and values of k_{cw}. Again, we observe an increasing trend in TPRs, however, the overall TPRs achieved are lower as compared to 2 person monitoring scenarios.

4.8.3.2 Effect of impact width k_{cw}

Figures 4.14(a)-4.14(c) show performance for the multi-person monitoring scenarios corresponding to all tested item category sets. By closely observing the figures 4.12(a)-4.12(c) and 4.14(a)-4.14(c), we can see that although, TPRs increase with k_{cw}, but FPRs also increase simulta-

neously. For both 2 person (1-6 in 4.14(a)-4.14(c))) and 3 person scenarios (7-8 in 4.14(a)-4.14(c))), we observe that TPRs increase with k_{cw} in almost each tested scenario, but FPRs also increase, which happens because in multi-person scenarios, the RSS values corresponding to the tags deployed around the item categories in between two nearby customers, are more aggressively affected, which can fool TagSee's multi-person imaging algorithm into wrongly incrementing the popularity counts of those categories. Although, TPRs achieved for $k_{cw} = 4$ and 6 are often lower than for $k_{cw} = 8$, but FPRs corresponding to those cases are considerably lower.

4.9 Discussions

TagSee is an early step towards monitoring customers' browsing behavior using COTS RFID devices. There is obviously room for continued research in various perspectives. In this section, we provide commentary on the limitations of our work and discuss avenues of future research.

DNN Architecture. The number of neurons per layer, number of layers and the types of layers are the primary hyper parameters of our DNN architecture. The problem of finding the correct hyper parameters for a neural network is a research problem in itself [17, 37, 84]. However, unlike standard practice in deep learning where many DNN architectures are randomly tried, the design of our DNN architecture is grounded on an analytical RFID imaging model that we derive in §4.5. From this model, we derive useful insights based on which we set those hyper parameters of our DNN architecture (§4.6). Our results show that our model driven architecture achieves reasonably good accuracies and generalizes well for unseen data. Other hyper parameters like dropout rate, learning rate, and regularization need to be estimated through trial and error, or a grid search. However, this search can be done before the model is deployed, so its cost does not affect the runtime of TagSee. Our DNN architecture can be generalized to tag matrices of different sizes and tag density (as discussed in §4.6) by changing the parameters k_x, k_y, p_x, p_y, and K. However, its evaluation is out of the scope of this paper.

Reading Rate. Figure 4.15 shows the impact of reading rate (normalized) on the Miss Rates and False Positive Rates in a 3-person imaging scenario. The MRs and FPRs were obtained for 50

different experiments that we ran for every plotted reading rate. We can observe that MRs and FPRs increase as reading rate decreases and vice versa. This is because imaging depends on variations in RSS signal from all the tags being affected by an obstruction. Receiving signal from only a few tags leads to imaging inaccuracies resulting in higher MRs and FPRs. However, note that our goal is to monitor the 'browsing' behavior of customers (i.e. when they stop to look at an item category without touching any items), not to continuously track the location of customers as they move about in the store. Browsing is a slow activity as it assumes that when a customer is browsing some item category, they usually spend a few seconds (e.g. 3-4 seconds) to browse the category. Because average tag read rate of our current system is approximately 475 reads/second, and there are only 116 tags in our current deployment, our system can obtain enough readings to construct reasonably accurate images of the users standing in front of different item categories. TagSee can work well in small stores (e.g. a small shoe store) with where the number of shelves is small and the number of deployed tags is on the order of reading rate. However, for a larger store, we can divide the store into multiple smaller regions and then deploy separate readers to monitor customer activity in each region to ensure that each tag is read frequently. In each region, the transmit power of antennas and the frequencies of operation can be set such that the inter-region interference is minimized.

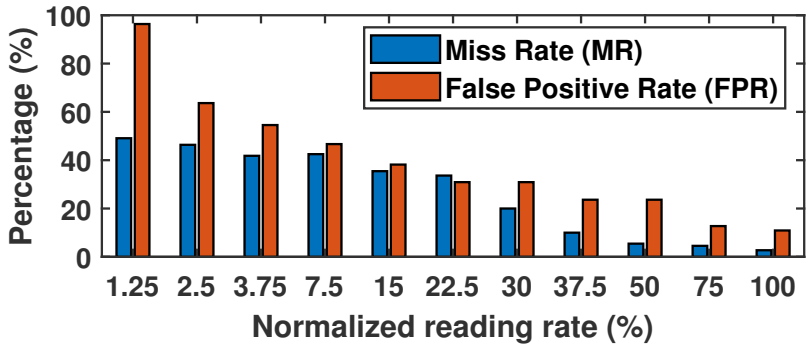


Figure 4.15: Impact of reading rate on FPRs and MPRs

Density of tags and image resolution: The parameters k_x, k_y, p_x, p_y, and K collectively control the size and density of a deployed mesh of tags. Imaging resolution will naturally be higher if tag density in a mesh is higher (i.e. inter-tag distance is small). For example, deploying tags more densely along X-axis can help better resolve two customers standing close to each other. We leave

the evaluation of such dynamics between tag density and imaging resolution as part of future work.

Collection of training data: Due to time and scheduling constraints, we were only able to collect data from a limited number of volunteers for testing. Although the data we collected using current setup is enough to test the basic working of TagSee, yet it is just the first step. In a real-life deployment scenario, for example in a shoe store, an automated camera triggered labeling system can be developed to calibrate TagSee over a period of few weeks. Such an automated calibration system will help generate a bigger and more diverse dataset that can be used to train a more robust DNN for TagSee.

Practical real-life deployments: TagSee's imaging scheme is based on the obstruction of LOS paths, or, more precisely, the Fresnel zones [54] between tags and reader antennas. In our current setup, the tags and antennas are placed in front of each other and at a certain height above the ground such that an approximate LOS is established between them. However, LOS can also be established in real-life in-store deployments, by hanging the antennas at an angle from the roof using ceiling mounts and attaching the tags to lower racks of the shelves as well as on the floor area near the shelves. In this way, when customers come near a shelf to browse an item category, they would obstruct the Fresnel zones between the tags and their respective reader antennas, based on which TagSee will try to determine the popularity of that item category.

4.10 Conclusions

In this work, we propose, implement, and evaluate TagSee, which is the first monostatic RFIDs based multi-person imaging scheme, which can be used to monitor browsing activity of customers near different display items in places such as physical retail stores. To achieve this, we propose a *deep neural networks* (DNNs) based RFID imaging approach, which robustly images the browsing activity of customers in front of the shelves with high accuracy. Our DNNs based imaging approach is driven by an analytical model, where we first mathematically formulate the problem of imaging humans using monostatic RFID devices and derive an approximate analytical imaging model that correlates the variations caused by human obstructions in the RFID signals. Afterwards, we

use that model to design our DNN's architecture. Finally, based on our proposed DNNs based imaging approach, we develop a technique which can track activity of multiple customers, showing interest in multiple different item categories, simultaneously. The key contribution of this work is in demonstrating the possibility of effective imaging of the browsing activity of multiple customers using existing RFID devices and protocols via robust, analytical model-driven deep learning based RFID imaging, which works even for scenarios where there is no interaction between customers and the display items. To the best of our knowledge, there is no prior work that can monitor customer browsing activity without using a camera or the requirement of physically touching retail items tagged with RFID tags. We believe our proposed RFIDs based multi-human activity tracking scheme for physical shopping environments will be useful for manufacturers and physical retail stores, and will help to shorten the gap between online and physical shopping.

CHAPTER 5

FINE-GRAINED VIBRATION BASED SENSING USING A SMARTPHONE

5.1 Introduction

5.1.1 Motivation

Vibration based sensing has been shown to be a low-cost and effective approach to recognizing different surfaces [26, 44, 69, 121]. A useful application of such sensing is symbolic localization/tagging, *e.g.* figuring out whether a user's device is in their hand, pocket, or at their bedroom table. The key intuition is that different surfaces respond to the same vibration differently because the surfaces may be made of different materials, and even if they are made of the same material, they may have different shapes and sizes. Even if two surfaces are made of the same material and have the same shape and size, they may have different objects placed on them, such that those surfaces still exhibit different vibration patterns because the objects placed on them may respond to the same vibration differently. Such symbolic tagging of locations can provide us with indirect information about user activities and intentions without any dedicated infrastructure, based on which we can enable useful services such as context aware notifications/alarms.

A robust and practical vibration based sensing scheme should satisfy three key requirements. First, it should work with commercial off-the-shelf (COTS) smartphones with different hardware, so that it can be easily deployed and widely adopted. Second, it should be able to extract fine-grained vibration signatures, so that it can accurately differentiate different surfaces. Third, it should be robust to environmental noise and hardware based irregularities, so that its accuracy stays consistent across different environments and devices.

5.1.2 Limitations of Prior Art

Several vibration based sensing schemes have been proposed in the past to realize different kinds of applications; however, none of them satisfies all the above three requirements. Existing vibration

based sensing schemes can be divided into two categories: custom hardware based and COTS smartphones based. The custom hardware based schemes use separate hardware including a micro-controller, a vibrator motor, and some piezoelectric (*e.g.* a microphone) or IMU sensors [69, 71, 75, 76, 121], which gives them fine-grained low level control over different physical layer parameters of their underlying hardware. However, these schemes are incompatible with COTS smartphones and are not easily generalizable to different hardware because most COTS smartphones have limited sensing capabilities and control over the hardware installed in them. Moreover, the custom hardware based schemes that use microphone to sense vibration are prone to short-term and constant background noises (*e.g.* intermittent talking, clapping, exhaust fan, etc.) because microphones not only capture the sounds created by vibration but also other interfering sounds present in the environment. The COTS smartphones based schemes rely on motion based features extracted from built-in IMU sensors [26, 44]. However, these features are very coarse-grained because the sampling frequencies of the IMU sensors are low, which naturally leads to low classification accuracies. Moreover, these schemes can only broadly differentiate between different types of surfaces (*e.g.* wood and plastic) and cannot differentiate between similar surfaces (*e.g.* two different wooden tables in Figs. 5.1(b) and 5.1(e)). Also, IMU readings get significantly affected by the smartphone's own motion in space (*e.g.* when a user moves his hand while holding the smartphone).

5.1.3 Proposed Approach

In this paper, we propose VibroTag, a vibration based sensing approach that can robustly recognize different surfaces based on their unique vibration signatures. Compared to previous work, VibroTag is robust and practical because it works with COTS smartphones, it can extract fine-grained features representative of different surfaces, and it is robust to hardware irregularities and background environmental noises. The key intuition is that as the vibrating mass inside a smartphone's vibrator motor repeatedly moves to and fro, the vibrating mass causes the whole smartphone structure and the hardware inside it to vibrate in a peculiar pattern, which depends upon the vibration response (or absorption properties) of the surface that smartphone is placed

on. These vibrations produce peculiar sound waves that VibroTag detects using the smartphone's microphone. Figure 5.1 shows the unique vibration signatures that VibroTag extracted for 6 different surfaces. We observe that vibration signatures of even two similar surfaces, *i.e.* Bed and Sofa, are quite different from each other.

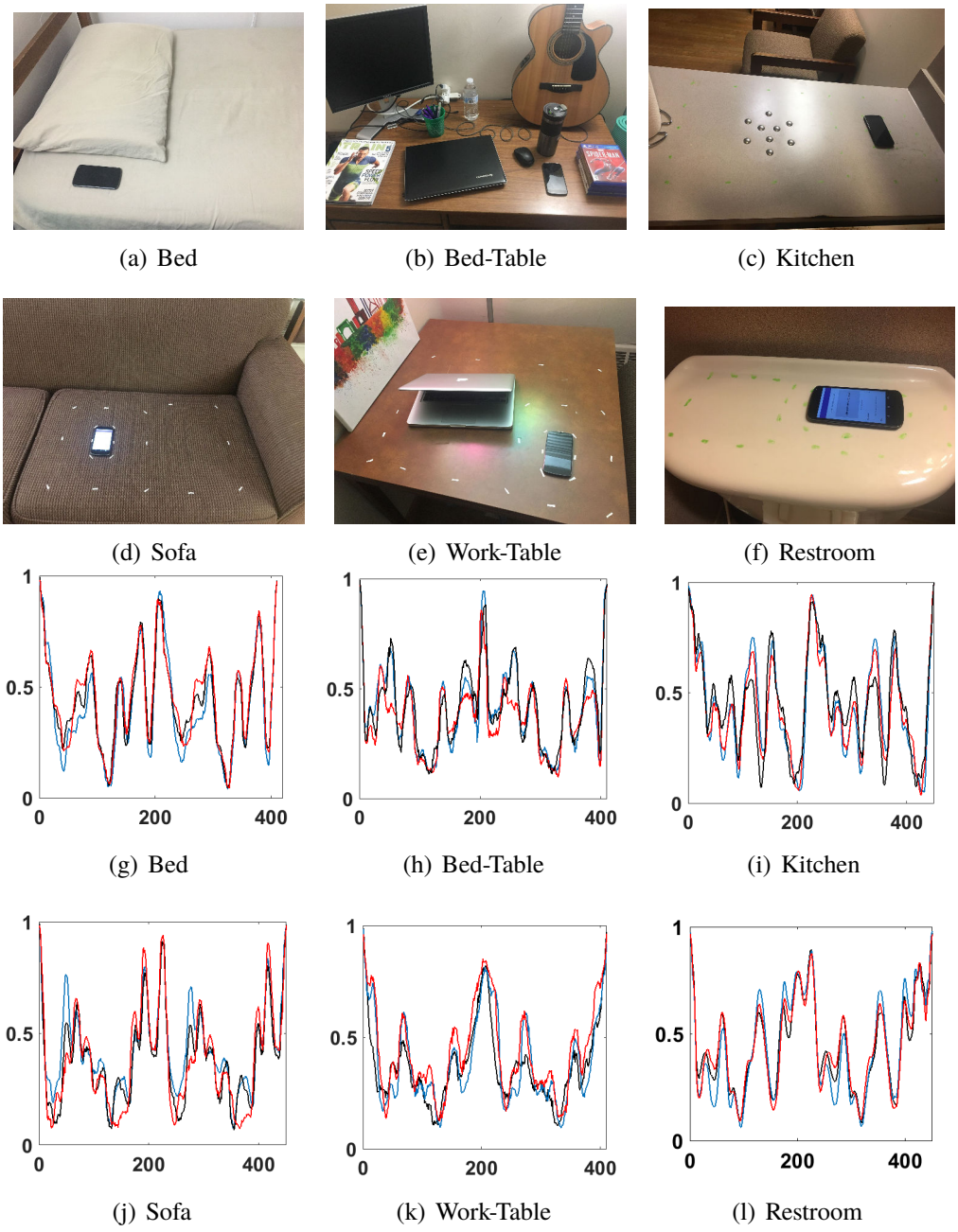


Figure 5.1: Experimental scenarios and their corresponding extracted acoustic time-series based vibration signatures

To make VibroTag easily scalable and compatible with COTS smartphones, we design VibroTag's signal processing pipeline such that it relies only on built-in vibration motors and microphone for sensing, and it is applicable to different phones with different hardware. To reliably extract fine-grained vibration signatures from the sound signals recorded during vibration, we propose a novel time-series based approach, which is robust to hardware irregularities and environmental noise. The key idea behind our vibration signature extraction approach is that even if there are irregularities in vibration frequencies due to hardware imperfections, the time-series patterns created during different vibration cycles are very similar. When a phone vibrates during a specific period of time, such as 3 seconds, multiple such patterns occur and get distributed all over the time-series of recorded sound signals. VibroTag finds multiple of these patterns in randomly selected intervals of the time-series, and then combines them into single time-series features that ensures consistency even if there are irregularities in the occurrence of those patterns and/or if the environment is slightly noisy. Afterwards, it uses these features to differentiate between surfaces.

5.1.4 Technical Challenges and Our Solutions

The first technical challenge is to reliably extract fine-grained vibration signatures. Based on our experiments on two different phones, we observed that the frequency response of a surface to vibrations introduced by vibration motors installed in COTS smartphones exhibit repeated irregularities, which makes extraction of reliable features a challenging task. This happens because the phone, its vibrator motor, and the rest of its hardware vibrate at irregular frequencies during every experiment, which occurs due to hardware imperfections. This behavior is random and uncontrollable, and therefore, is bound to create significant variations within features extracted at the same location, which will lead to classification inaccuracy. The existing techniques that use microphone to extract sound (sampled in the order of kHz) based straightforward frequency domain features (*e.g.* vibration sound spectrum [69]) are not only considerably susceptible to such hardware based irregularities, but also to short-term and constant environmental noises, where even intermittent talking or noise from a restroom's exhaust fan can significantly affect their performance.

To address this challenge, we take a time-series based vibration signature extraction approach. First, we differentiate the recorded sound signals and take their root mean square (RMS) envelope, which removes most of the unrelated constant and higher frequency background noise. Second, we develop a specialized peak-detection based algorithm to extract unique time-series patterns corresponding to vibrations from the RMS envelope, and then use them as *vibration signatures* to represent different surfaces. Our extraction algorithm is based on the observation that even if there are irregularities in vibration frequencies due to hardware imperfections, the time-series patterns created during different vibration cycle are very similar. When a phone vibrates during a specific period of time, multiple such patterns occur all over the time-series of recorded sound signals, which can be successfully extracted by our algorithm. To make the vibration signatures robust to environmental noise, VibroTag extracts numerous such vibration patterns across time during an experiment and combines them by taking their median.

The second technical challenge is to compare vibration signatures of any two surfaces. The midpoints of extracted vibration signatures of the same surface rarely align with each other because the start and end points determined by extraction algorithm are never perfectly aligned. Moreover, the lengths of different vibration signatures also differ slightly because the duration of vibration cycle can often be a little different due to hardware irregularities. Consequently, the midpoints and lengths of vibration signatures do not match either. Another issue is that the shape of different vibration signatures of the same surface are often distorted versions of each other, which occurs due to hardware based irregularities in the vibration mechanism. Therefore, two vibration signatures cannot be compared using standard measures like *correlation coefficient* or *Euclidean distance*. To address this challenge, we use the Dynamic Time Warping (DTW) to quantify the distance between any two vibration signatures. DTW can find the minimum distance alignment between two waveforms of different lengths. For classification, we employ a *Nearest-Neighbor* (NN) classifier with DTW distance as the comparison metric between different vibration signatures.

5.1.5 Key Novelty and Advantages

The key technical novelty of this paper is on proposing the first fine-grained vibration based sensing scheme that can recognize different surfaces using the vibration mechanism and microphone of a single COTS smartphone. Furthermore, we propose a novel signal processing technique to extract fine-grained vibration signatures that are robust to hardware irregularities and background environmental noises. The key insight is that even if there are irregularities in vibration frequencies due to hardware imperfections, the time-series patterns created during different vibration cycles are very similar. VibroTag finds many such patterns in the sound signals recorded during vibration, and combines them into single consistent vibration signatures. Compared to previous schemes, VibroTag works with COTS smartphones, it can extract fine-grained features representative of different surfaces, and it is robust to hardware irregularities and background environmental noises.

5.1.6 Summary of Experimental Results

We implemented VibroTag on two Android based smartphones, *i.e.* Nexus 4 and OnePlus 2, for which we developed an application for generating vibrations and to sample sound signals simultaneously. We tested our system for 4 different individuals, from whom we collected data for 5 - 20 days. We show that VibroTag achieves an average accuracy of 86.55% while recognizing 24 different surfaces, with as few as 15 training samples per surface. Moreover, VibroTag maintains an average accuracy of up to 85% without any re-training requirements after 3-4 days of training. We also implement the state-of-the-art IMUs based vibration sensing scheme for single COTS smartphones proposed in [26], and compare its surface recognition accuracy with VibroTag. We show that VibroTag achieves more than 37% higher accuracy when compared to the IMUs based scheme, while recognizing the 24 different surfaces.

5.2 Related Work

Existing work related to our work consists of some vibration based sensing schemes [26, 44, 69, 71, 75, 76, 121, 136] and sound based symbolic localization schemes [15, 135].

Vibration Based Sensing: Vibration based sensing schemes leverage the response of different surfaces to a specific vibration pattern to recognize those surfaces. Existing vibration based sensing schemes can be divided into two categories, *i.e.* custom hardware based, and COTS smartphones based. The custom hardware based schemes use separate customized hardware made using a set of micro-controller, vibrator motor, and piezoelectric (*e.g.* microphones) or IMU sensors [69, 71, 75, 76, 121], so that they have fine-grained low level control over different physical layer parameters of their hardware. ViBand uses variations introduced due to vibrations produced by different objects to identify those objects, *e.g.* electric tooth brush [71]. VibKeyboard [75] and VibSense [76] develop a virtual keyboard based on the idea that the impact of a touch on a surface such as a table or door causes a shockwave to be transmitted through the material that can be passively detected with accelerometers or more sensitive piezo-vibration sensors. Kunze *et al.* [69] develop customized hardware to recognize surfaces through active sampling of acceleration and sound signatures. However, the above schemes are incompatible with COTS smartphones and are not easily generalizable to different hardware because most COTS smartphones have limited sensing capabilities and control over the hardware installed in them. The COTS smartphones schemes rely on motion based features extracted from built-in IMU sensors [26, 44]. Cho *et al.* [26] and Shafer *et al.* [121] use built-in vibrator and accelerometer of a COTS smartphone to recognize surfaces. Griffin *et al.* use vibration detected by an acceleration signal to determine if a phone is in the user's hand [44]. However, the features used by these schemes are very coarse-grained because the sampling frequencies of IMU sensors are low, which naturally leads to low classification accuracies. Finally, all the previous schemes that use microphone based approaches to sense vibrations are prone to short-term and constant background environmental noises (*e.g.* intermittent talking, clapping, exhaust fan, dripping water, etc.). Compared to all the above schemes, VibroTag is robust and easily deployable because it works with COTS smartphones, it can extract fine-grained features representative of different surfaces/locations, and it is robust to hardware irregularities and background environmental noises.

Sound Based Symbolic Localization: Sound based symbolic localization systems leverage the

propagation of the sound generated using speakers of a device, such as a smartphone, to determine the symbolic location of that device (*e.g.* whether the device is in the user's kitchen or at his bedroom table). SurroundSense uses sensor data from a microphone, a light sensor, the wireless radio, and passive accelerometer data for localization [15]. However, their technique can only be used for very coarse-grained localization (*e.g.* room level) and not for finer-grained localization (*e.g.* whether the phone is on user's study table or his bedroom table). EchoTag generates ultrasound signals and then uses the reflections from the environment to achieve centimeter level tagging [135]. However, their work requires strict millimeter level marking of the tagged locations because the ultrasound signals based signatures that they use are highly location dependent, where even small variations in the phone's position leads to significant localization errors. This makes their scheme unsuitable for symbolic localization, and also puts significant calibration effort on the user end. In contrast to above schemes, VibroTag uses vibration instead of speaker generated sound signals for such symbolic localization. Moreover, VibroTag achieves finer-grained localization, and does not require strict marking of the tagged locations.

5.3 Understanding Vibrations

5.3.1 Vibrator Motors in Smartphones

Electric vibrator motors generate vibrations by periodically moving an unbalanced mass around a center position using the principles of electromagnetic induction. The vibrator motors used in today's smartphones are often known as *coin-type vibration motors* due to their coin-like shapes and sizes. There are two types of coin-type vibrator motors that are widely adopted in smartphones: (i) *Linear Resonant Actuator* (LRA) based (*e.g.* used in Nexus 4) and (ii) *Eccentric Rotating Mass* (ERM) based (*e.g.* used in OnePlus 2). Figure 5.2 shows the internals of ERM and LRA based vibration motors. ERM based motors use a DC motor to rotate an eccentric mass around an axis. As the mass is not symmetric with respect to its axis of rotation, it causes the device to vibrate during the motion. Both the amplitude and frequency of vibration depend on the rotational speed of the motor, which can in turn be controlled through an input DC voltage. With increasing input voltages,

both amplitude and frequency increase almost linearly and can be measured by an accelerometer. In LRA based motors, vibration is generated by the linear movement of a magnetic mass suspended near a coil, called the "voice coil". When an AC current is applied to the motor, the coil behaves like a magnet (due to the generated electromagnetic field) and causes the mass to be attracted or repelled, depending on the direction of the current. This generates vibration at the same frequency as the input AC signal, while the amplitude of vibration is dictated by the signal's peak-to-peak voltage. Thus, LRAs offer control on both the magnitude and frequency of vibration.

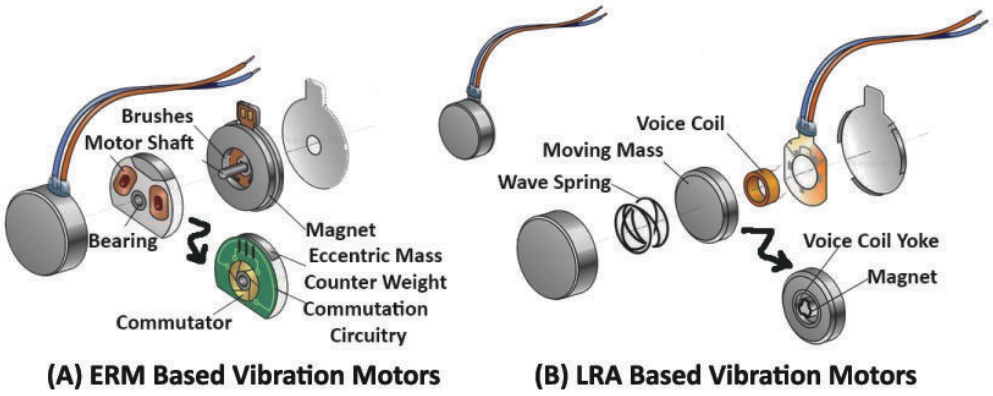


Figure 5.2: ERM and LRA based vibration motors [128]

5.3.2 Physics of Surface Response to Vibrations

Sound is essentially pressure waves created by vibrating matter. These waves are longitudinal, *i.e.* they oscillate along the axis of travel, where the oscillation is composed of compression and rarefaction of molecules in the medium (*e.g.* air). For example, human speech is based on vibrations created inside our vocal chords, and audio speakers generate sound by translating an electrical signal into physical vibrations via mechanical excitation of a diaphragm using an electromagnet.

VibroTag is based on the intuition that different surfaces exhibit different response to vibrations introduced by smartphone. When a smartphone vibrates, it mechanically excites not only it's own structure and hardware inside it, but also the surface on which it is placed. On one hand, some surfaces tend to absorb most of the vibration energy (*e.g.* Sofas), while on the other hand, some surfaces may exhibit a resonant response where they start vibrating in sync with the smartphone

(*e.g.* the smartphone's surface vibrates in sync with the vibrator motor inside). Moreover, the effect of these vibrations can reach different objects placed nearby, which may get mechanically excited as well (especially the lighter objects); therefore, leading to more peculiar sounds. As different surfaces respond to the vibrations differently (in terms of their absorption/dampening effect on smartphone's movements), and as different surfaces often have different objects placed on them, which also respond to those vibrations differently, pressure waves peculiar to those surfaces are created during the vibration, which we can sense using a piezoelectric device (*e.g.* a built-in microphone) and then leverage to differentiate those surfaces.

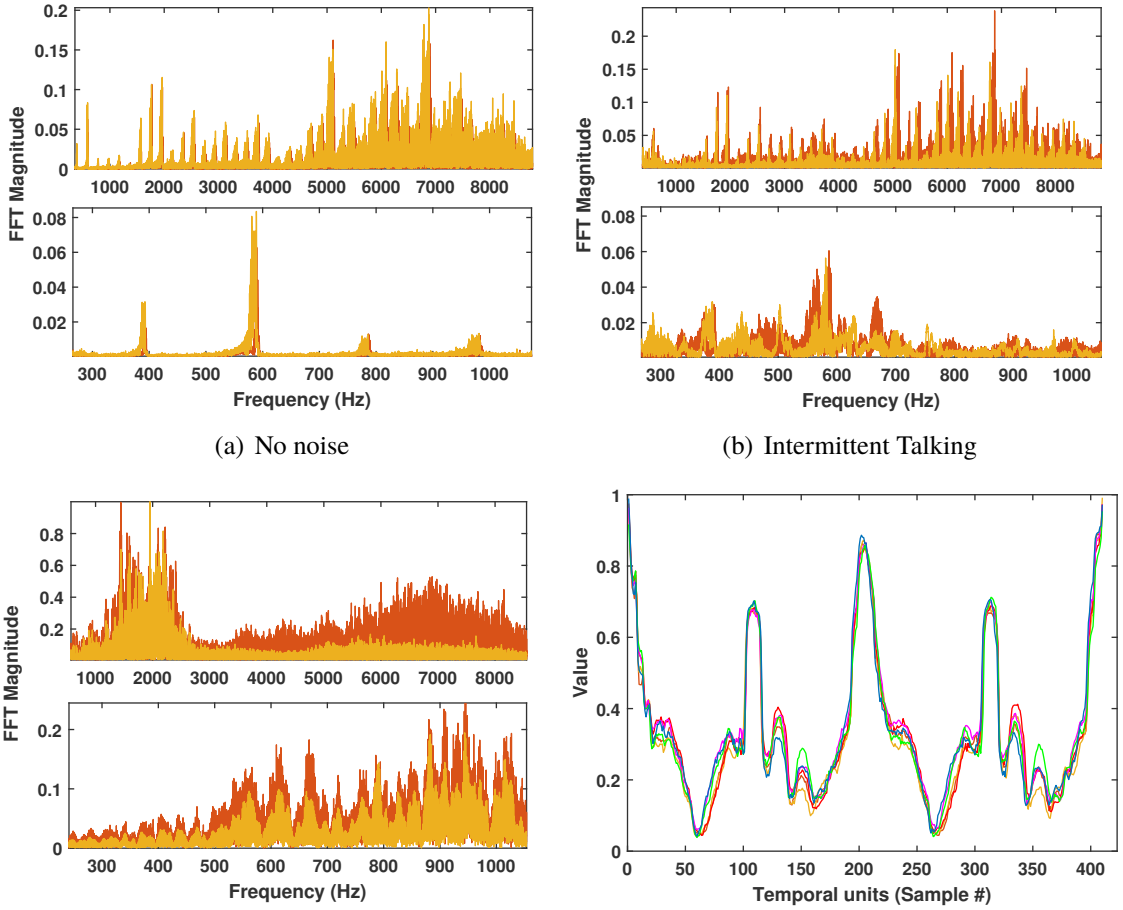
5.4 Feature Extraction

To differentiate between different locations, we need to extract features that can uniquely and consistently represent those locations. In VibroTag, a smartphone is vibrated for about three seconds while the surface response to the vibration is recorded simultaneously via the phone's built-in microphone. Sounds produced during vibration are sampled at fixed F_s = 44.1 kHz. The recorded sound is analyzed in both frequency and time domains to extract robust surface/location specific vibration signatures. There are two key challenges in feature extraction for VibroTag to be robust. The first challenge is on reducing impact of background noises (such as those created by fans and short-term human speech). The second challenge is on accommodating smartphone hardware imperfections (*i.e.* microphones and vibrator motors mainly) that degrades the quality of the signals collected when a smartphone vibrates.

5.4.1 Robustness to Background Noise

To understand the challenge posed by background noise, we use Fast Fourier Transform (FFT) based Power Spectral Density (PSD), which is one of the mainstream frequency based feature extraction techniques for acoustic sensing. Figure 5.3(a), 5.3(b), 5.3(c) show the FFT coefficients for both lower and higher frequency ranges corresponding to our experiments conducted at the same location on a wooden chair's cushion for three scenarios: (a) no noise, (b) intermittent human speech, and (c) clapping, respectively. We can observe that the FFT features are significantly affected

by the background noises because the frequencies produced by these noises directly interfere with the frequency bands for vibration based sensing. It also shows that mainstream techniques such as FFT or PSD are unsuitable for vibration based sensing on COTS smartphones when there are background noise sources present in the environment. In this work, we propose two schemes to reduce the impact of constant and intermittent short-term background noises, respectively.



(a) No noise

(b) Intermittent Talking

(c) Clapping

(d) VibroTag signatures (a)-(c)

Figure 5.3: Impact of background noises on features extracted by traditional techniques and VibroTag

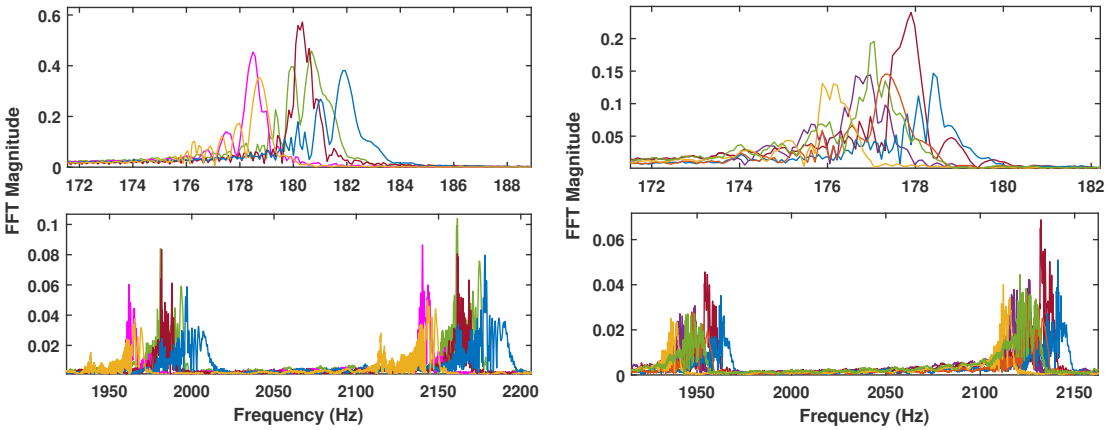
To reduce the impact of constant background noises in VibroTag, we take the *first order difference* of the recorded sound signals and then take their *root mean squared* (RMS) envelope. We choose RMS envelope for our analysis as it gives us a measure of the power of the vibration signals, while producing a waveform that is easy to analyze. Moreover, higher frequency noisy variations are averaged out in the envelope signal, while it still keeps most of the useful vibration

response related information intact. We take the RMS envelope of the signals over a sliding window of N samples, where N = 15 audio samples in our current implementation of VibroTag. In the rest of this work, when we mention sound signals, we mean the first order difference of the RMS envelope of those sound signals.

To reduce the impact of intermittent short-term noises (similar to Figures 5.3(b) and 5.3(c)), we vibrate the phone for at least 3 seconds, and extract multiple vibration patterns across time from the processed sound signals; then, we combine these virbration patterns to get a single consistent vibration signature. We will discuss how we extract such signatures in Section 5.4.3. Figure 5.3(d) shows the signatures extracted by VibroTag for the experiments corresponding to Figures 5.3(a), 5.3(b), 5.3(c), from which we observe that our signatures are consistent and almost identical even though there are intermittent short-term noises.

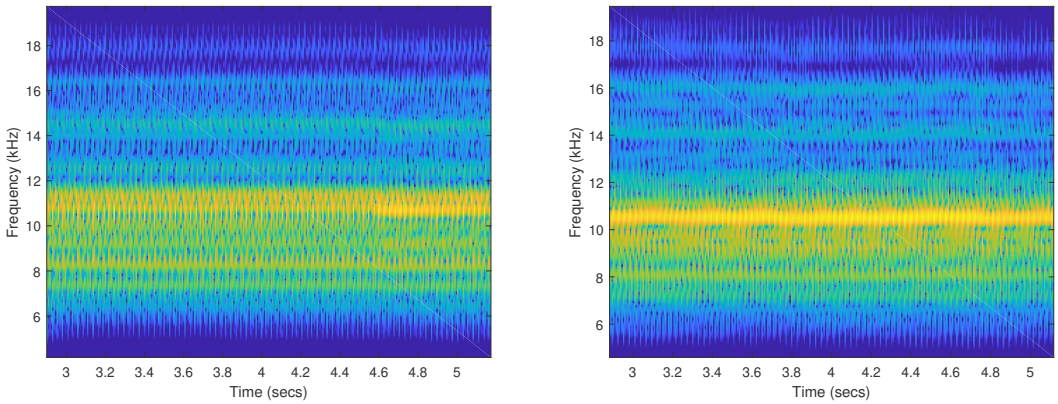
5.4.2 Robustness to Hardware Imperfections

To understand the challenge posed by smartphone hardware imperfections, we use PSD and Short Time Fourier Transform (STFT). Figures 5.4(a) and 5.4(b) show the PSD of the recorded unprocessed sound signals for multiple different experiments, which we performed by placing a smartphone at the same location on a sofa and a bed, respectively. Each figure shows the PSD coefficients over two different frequency ranges. We can observe that PSDs for both the sofa and the bed are not consistent for repetitive experiments, even when the smartphone's location and the environmental scenario while performing the experiments remained unchanged. This occurs because the smartphone, its vibrator motor, and the rest of its hardware vibrate at slightly different frequencies in each different experiment. This behavior is random and uncontrollable, and therefore, is bound to cause intra-class (*i.e.* within samples of the same class) variations, which will lead to classification errors. Moreover, due to this inconsistency, it often happens that the variations due to vibration on two different surfaces occur in similar set of frequencies (as shown by some samples in Figs. 5.4(a) and 5.4(b)), which further makes the use of such frequency domain features infeasible as they cannot uniquely represent different surfaces. Figs. 5.4(c) and 5.4(d) show the STFT of the



(a) Home Bed

(b) Home Living Room Sofa



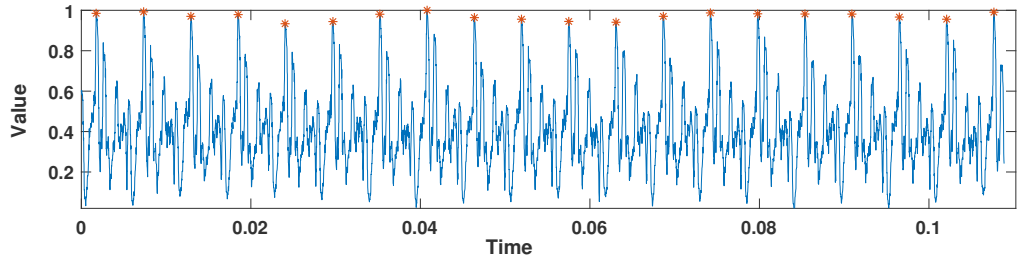
(c) Home Bed

(d) Home Living Room Sofa

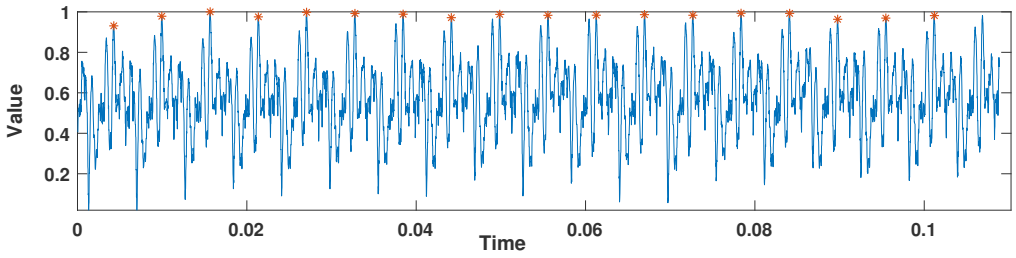
Figure 5.4: Impact of hardware imperfection on features extracted by traditional techniques PSD and STFT

unprocessed sound signals from one experiment corresponding to each of the two surfaces. Again, we observe that STFTs of both surfaces are very similar, and variations often occur in the similar frequency ranges that makes it harder to differentiate between those surfaces. Interestingly, we also observe that some patterns repeating along time; however, the time period of their repetition is not consistent.

We hypothesize that even if there are irregularities in the repetition frequency of such vibration patterns, the patterns themselves must be very similar. As we discussed in §5.3, smartphone vibrations are generated because of the to and fro motion of a mass inside its vibration motor. The vibration motor tries to move the smartphone (and the hardware inside) at its own vibration frequency (which is often irregular due hardware imperfections). However, the smartphone's motion



(a) Home Bed



(b) Home Living Room Sofa

Figure 5.5: Repetitive patterns appearing in the processed sound signals corresponding to vibration is constrained due to its own weight/structure and the absorption properties of the surface that it is placed on. This whole process during vibration gives rise to peculiar pressure waves, which can be sensed by the built-in microphone. Moreover, as the mass inside the motor repeatedly moves to and fro, it will give rise to similar pressure waves in every cycle of its "irregular" vibration period, which will reflect in time-series of the sound signals. These intuitions form the basis of our time-series based analysis of the surfaces' vibration response (§5.4.3).

In this work, we propose to address the issues due to smartphone hardware imperfections by extracting time-series based vibration signatures from the processed sound signals. Figures 5.5(a) and 5.5(b) show time-series of the sound signals (*i.e.* first order difference of the RMS envelope) corresponding to one experiment from each of the two different surfaces (*i.e.* Bed and Sofa), whose PSDs are shown in 5.4(a) and 5.4(b) and whose STFTs are shown in 5.4(c) and 5.4(d) , respectively. The two time-series correspond to a window of 4800 sound samples (*i.e.* ~0.1088 seconds for F_s = 44.1 kHz). We can easily observe distinguishing patterns repeating in both time-series, which repeat approximately with the frequency of the vibrating mass in the smartphone's vibration motor. We also observe that the patterns in both scenarios are consistent across time, and that the patterns in one scenario are different from the ones in other scenario.

5.4.3 Extraction of Vibration Signature

To robustly differentiate surfaces, we need to extract vibration patterns from time-series of the processed sound signals and then use those patterns to obtain consistent vibration signatures. However, we face multiple challenges. The first challenge is that intermittent short-term noises in real-life scenarios are uncontrollable, and therefore, can affect any part of the time-series. VibroTag needs to extract the vibration patterns that are representative of the whole time-series, *i.e.* the vibration patterns extracted from one segment of the time-series (*i.e.* a time window) should repeat in other segments, and therefore, truly represent the surface's vibration response. A naive approach is to extract all vibration patterns in the recorded signals and then combine them (*e.g.* by taking their average), which is computationally expensive. To address this challenge, we take a randomized approach, where we first divide the whole time-series of the sound signal into equally sized time windows, and then randomly select multiple of those time windows to extract vibration patterns from. Each window is of size S, where $S = 4800$ sound samples in our current implementation of VibroTag. Moreover, the windows are selected without replacement, *i.e.* once selected, they are not selected again. VibroTag keeps randomly selecting new time windows until M vibration patterns are extracted (M=100 in our implementation). *In real-life scenarios, the number of iterations required to extract M vibration patterns of a surface can be used to tell the user whether their environment is too noisy to extract a valid vibration signature or not.* For example, the average (taken over 20 different samples) number of iterations it took for convergence when loud music (*i.e.* high variable noise) was played on a laptop in the background was ∼1351, for medium noise/volume level the number was ∼495, whereas for no variable noise scenarios it was ∼136. If our algorithm cannot find enough vibration patterns and runs out of possible time windows to search for patterns, it will not converge. However, we did not experience any such scenarios during our testing.

The second challenge is to extract the vibration patterns from different randomly selected time windows by localizing their place of occurrence in those time windows. However, because of the inconsistencies in the vibration behavior of smartphone due to hardware imperfections, we cannot know the frequency of repetition of the vibration patterns, which makes it harder to

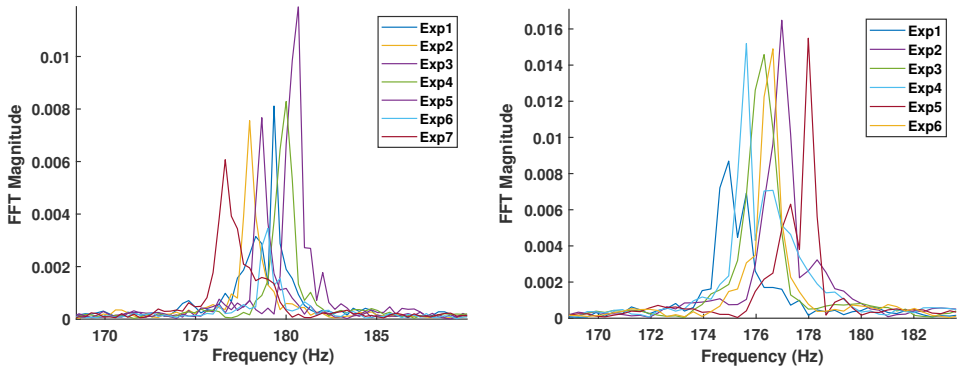
localize the place of occurrence of such patterns. To address this challenge, we develop a *peak detection* based algorithm to extract vibration patterns. Our algorithm is based on the observation that every vibration pattern has a peak value that occurs consistently at around the same part of every vibration cycle (as evident in Figs. 5.5(a) and 5.5(b)). Based on this algorithm, VibroTag determines the locations of multiple such peaks in each of the randomly selected time windows. Afterwards, VibroTag uses the consecutive peaks detected in each window to extract the vibration patterns between those peaks.

5.4.3.1 Extraction of Vibration Patterns

There are two key challenges in developing our peak detection based vibration pattern extraction algorithm. First, based on our experiments, we observe that the scale of variations due to vibration in the time-series of different randomly selected windows varies, even when the same smartphone is placed at the same location of the same surface, which happens due to hardware imperfections based inconsistencies in the vibration process. Moreover, different surfaces and different smartphones exhibit different scale of variations due to vibration. This makes the parametrization of our peak detection algorithm difficult to generalize. To address this challenge, VibroTag performs *max-min normalization* on the time-series corresponding to every selected window before feeding it to the peak detection algorithm. This step ensures that the parametrization of our algorithm can be easily generalized to different time windows and to different smartphones and surfaces.

The second challenge is to robustly determine the locations of peaks corresponding to different vibration patterns present in a time window. To address this, VibroTag's peak detection algorithm determines the location of such peaks based on three key parameters, namely *minimum peak prominence* (MINPRO), *minimum peak distance* (MINDIST), and *minimum peak strength* (MINSTR). The prominence of a peak measures how much the peak stands out, due to its height and location, relative to other peaks around it. We tune MINPRO such that we only detect those peaks which have a relative importance of at least MINPRO. We tune MINDIST according to the fact that maximum repetition rate of patterns is approximately $\hat{f_o} + \delta f$, such that the redundant peaks are

discarded, where $\hat{f_o}$ is an approximate number for the the frequency of repetition (f_o) of vibration patterns. As we discussed before, the frequency of repetition of vibration patterns in the processed sound signal is irregular. In order to determine $\hat{f_o}$, VibroTag calculates PSD of the time-series in the selected window. Figures 5.6(a) and 5.6(b) show example PSD's corresponding to one of the randomly selected time windows from seven different experiments performed on the Bed and the Sofa, respectively. VibroTag determines the peak frequency from the PSD, which corresponds to the approximate repetition frequency (*i.e.* $\hat{f_o}$) of the vibration patterns present in the window. The term δf represents the second standard deviation of the variation in vibration frequency around $\hat{f_o}$, which we estimate for every smartphone based on multiple different experiments. δf is required as some vibration patterns can repeat earlier than $1/\hat{f_o}$.



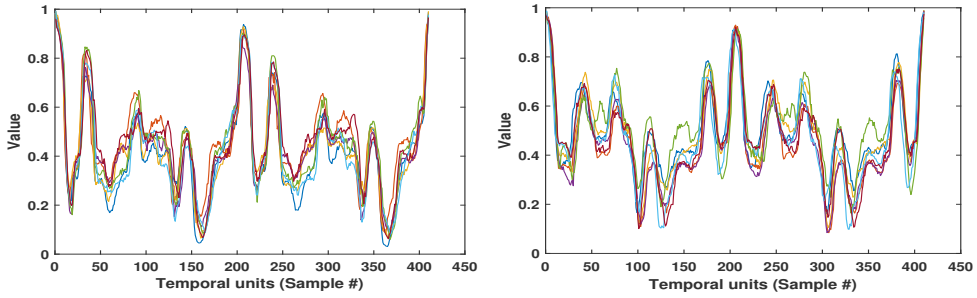
(a) Home Bed (b) Home Living Room Sofa

Figure 5.6: PSD of the sound signals in time windows corresponding to the scenarios in Figs. 5.5(a)-5.5(b)

To further sift out redundant peaks, we only choose peaks of value greater than MINSTR times the median value of the peaks detected in the window. In our current implementation, we chose MINPRO = 0.65, MINDIST = $\frac{1}{f_o+\delta f}$ seconds, δf = 6.5, and MINSTR = 0.5. We perform this parametrization only during the design time, which generalizes well for multiple different surfaces and smartphones (*i.e.* Nexus 4 and OnePlus 2). Our algorithm does not require any end-user calibration effort. VibroTag uses the consecutive peaks detected in each randomly selected window to extract multiple vibration patterns between those peaks.

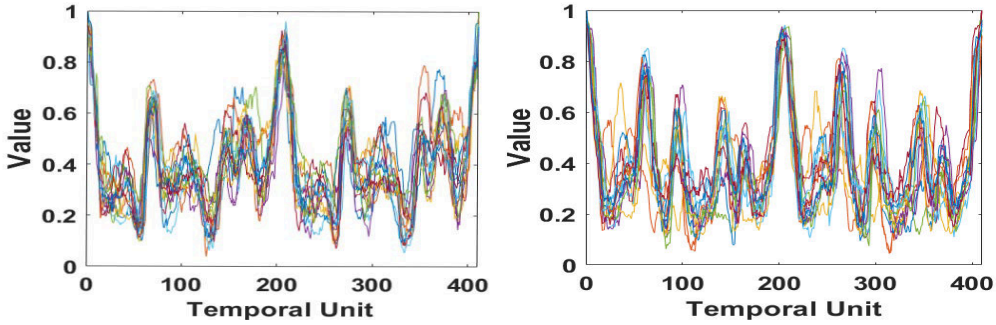
5.4.3.2 Construction of Vibration Signature

To construct a single consistent vibration signature, VibroTag first collects at least a total of $M = 100$ patterns extracted from the randomly selected time windows. Once an enough number of vibration patterns are extracted from different time windows, VibroTag combines all those patterns using median (loses anything lying outside 75% of the data) to get a single vibration signature. The median operation helps VibroTag remove short-term noisy variations in different vibration patterns, and therefore, allows it to extract a single robust vibration signature of the surface. Figures 5.7(a) and 5.7(b) show the example signatures extracted for the Bed and the Sofa related experiments, where we can observe that the vibration signatures of each surface are consistent and almost identical. Moreover, the extracted signatures uniquely represent their respective surfaces.



(a) Bed (b) Sofa

Figure 5.7: Extracted time-series features (Low Noise)



(a) Cafeteria Table 1 (b) Cafeteria Table 2

Figure 5.8: Extracted time-series features (High Noise)

Figure 5.8 shows the vibration signatures extracted for two similar tables during lunch time in a cafeteria on a university campus (*i.e.* a highly noisy environment). We can see that although

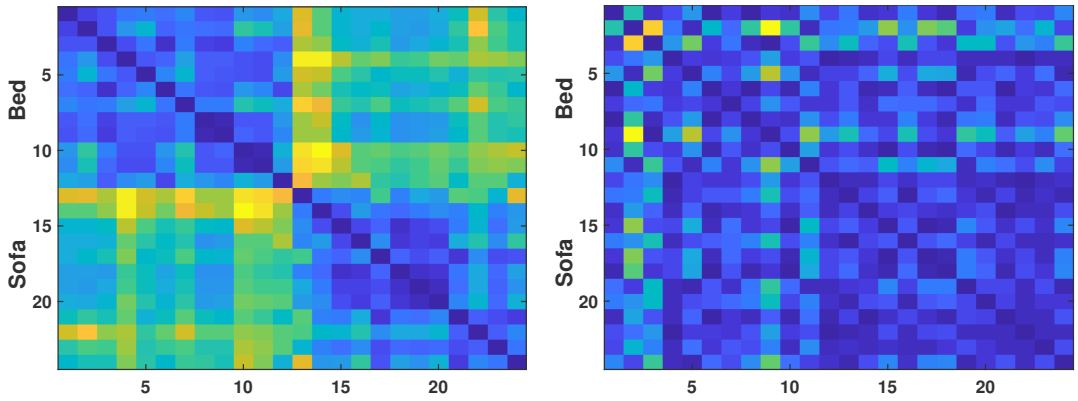
some vibration signatures that VibroTag extracted are inconsistent, yet it was able to extract several consistent vibration signatures even in such a highly noisy environment.

5.5 Classification & Recognition

We use the shapes of the extracted waveforms as features because the shapes retain both time and frequency domain information of the waveforms and are thus more suited for use in classification. After obtaining the time-series based vibration signatures, VibroTag uses them to build training models for classification. As VibroTag needs to compare vibration signatures obtained for different surfaces, we need a comparison metric that provides an effective measure of the similarity between vibration signatures of two surfaces. To achieve this, VibroTag uses the technique of Dynamic Time Warping (DTW) that calculates the distance between waveforms by performing optimal alignment between them. Using DTW distance as the comparison metric between vibration signatures, VibroTag trains a k-nearest neighbour (kNN) classifier using those signatures.

DTW is a dynamic programming based solution for obtaining the minimum distance alignment between any two waveforms. DTW can handle waveforms of different lengths and allows a non-linear mapping of one waveform to another by minimizing the distance between the two waveforms. In contrast to Euclidean distance, DTW gives us the intuitive distance between two waveforms by determining the minimum distance warping path between them even if they are distorted or shifted versions of each other. DTW distance is the Euclidean distance of the optimal warping path between two waveforms calculated under boundary conditions and local path constraints. In our experiments, DTW distance proves to be effective for comparing two vibration signatures of different surfaces. Figure 5.9(a) shows the colormap of DTW distance between the vibration signatures extracted by VibroTag from the experiments corresponding to the bed and the sofa (12 signatures each). The average DTW distance among signatures of the bed is ~2.3 and that for the sofa is ~3.1. However, the average DTW distance between the vibration signatures of the two surfaces was 16.59. Figure 5.9(b) shows the color map of Euclidean distance between features obtained using the IMUs based scheme proposed in [26]. We can see that IMU based features cannot successfully differentiate

between the two surfaces due to high similarity, whereas VibroTag's signatures are significantly better at differentiating the two seemingly similar surfaces.



(a) ViborTag (b) IMU features [26]

Figure 5.9: Colormaps of distance between features of (a) VibroTag (DTW) & (b) IMU scheme (Euclidean)

VibroTag requires training data for the surfaces to be recognized. Afterwards, it trains a kNN classifier using the vibration signatures corresponding to those surfaces. To recognize a surface, VibroTag feeds the detected vibration signature of that surface to the trained kNN classifier. The kNN classifier searches for the majority class label among k nearest neighbors of the corresponding vibration signature using the DTW distance metric. VibroTag declares the majority class label obtained from the kNN classifier as label of the tested surface. In the current implementation of VibroTag, we chose $k = 5$ so that the classification process averages more voters in each prediction, which makes our classifier more resilient to outliers.

5.6 Implementation & Evaluation

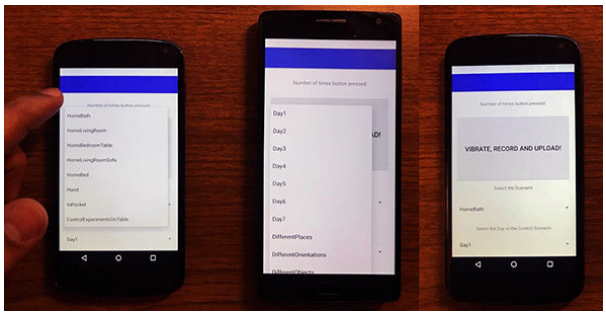
5.6.1 Implementation Details

We developed an Android application for generating vibrations and sampling sound signals simultaneously (Fig. 5.10(a) shows VibroTag's UI). Our application can record sound in a separate high priority asynchronous thread which helps minimize sampling related irregularities. We use a 16 bit PCM encoding on Mono channel with a sampling rate of 44,100Hz for sound recording. We also record the data from the smartphone's IMU sensors (*i.e.* accelerometer and gyroscope) in a separate

high priority thread. We use this data to implement the state-of-the-art IMUs based vibration sensing approach for single COTS smartphones proposed in [26], and then compare its surface recognition accuracy with VibroTag. Each data instance constitutes ~3 seconds of sound and IMU data, during which the vibration motor keeps vibrating. Our application controls a smartphone's vibrator motor only in terms of turning it ON or OFF, and therefore, does not change the amplitude or pattern of the vibrations. This allows the smartphone to vibrate at its default vibration settings, which helps keep data samples collected at the same surface/location consistent. Moreover, it makes VibroTag applicable to smartphones which do not provide any amplitude control over their vibration motors. We evaluated VibroTag using two smartphones, *i.e.* Google Nexus 4 and OnePlus 2.

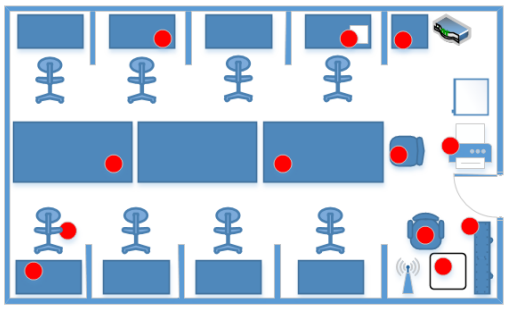
5.6.2 Evaluation Setup

We evaluated VibroTag's performance by conducting extensive experiments in two different type of environments, *i.e.* office and apartment. We selected these environments because they represent real-world use case scenarios where a user interacts with different objects and surfaces regularly. We collected data from 4 different volunteers, three with Nexus 4 and one with OnePlus 2, whom we name User-1 (Nexus), User-2 (Nexus), User-3 (OnePlus 2), User-4 (Nexus), respectively. All volunteers were university students who lived in different apartment complexes. No restrictions were imposed on the movement or work conditions of people residing/working in the apartments/office. For example, when we collected data in the office environment, other people in the office were working and chatting as they do on a normal working day. Similarly, data collection did not cause any interference in the daily activities (cooking, eating, watching TV, cleaning, etc.) in the volunteers' apartment mates. Therefore, our evaluation of VibroTag takes into account realistic environments where noise from human activities is present most of the time. We used metrics such as confusion matrices, True-Positive-Rates (TPRs) and False-Positive-Rates (FPRs) to evaluate VibroTag's classification performance. We also compare VibroTag's performance with the IMUs based approach proposed in [26].



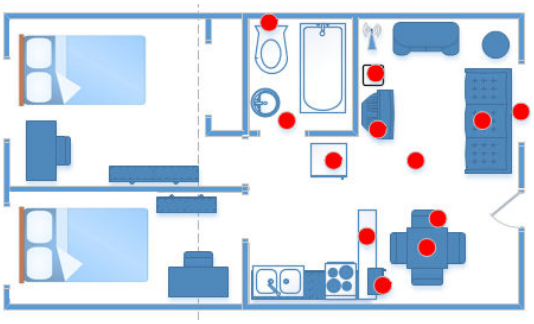
(a) ViborTag App interface

(b) Office



(c) Office locations

(d) Apartment



(e) Apartment locations

Figure 5.10: VibroTag Setup (a) VibroTag App (b) office environment (c) example of data collection locations in office (d) example of surfaces used for data collection (e) example of data collection locations in apartment

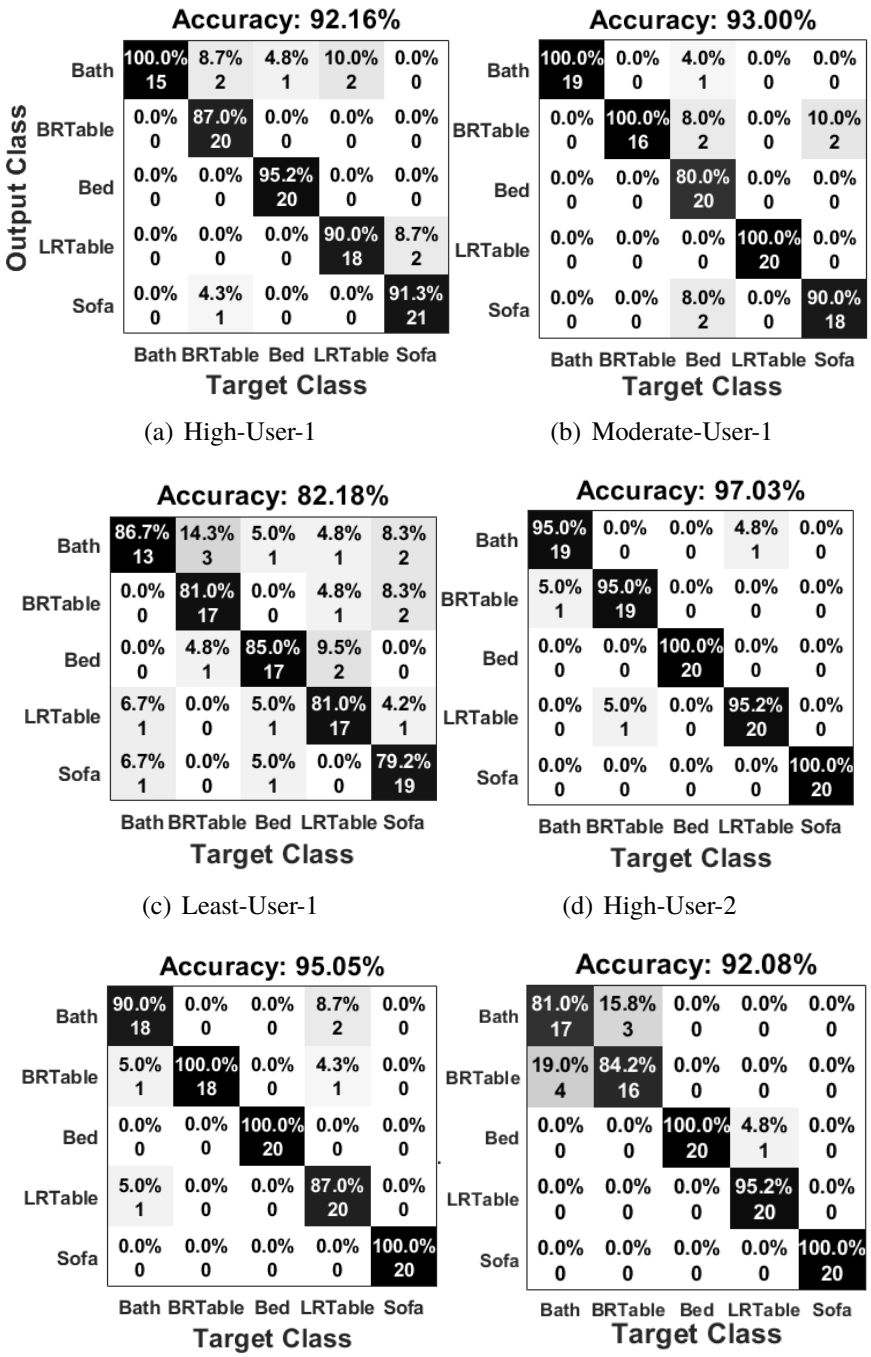
5.6.3 VibroTag's Sensitivity

We define VibroTag's sensitivity as its ability to differentiate between different positions and orientations of the smartphone placed on the same location/surface. For example, a user can place his smartphone on his office table in several possible positions and different orientations. VibroTag's sensitivity is an important metric since we claim that a user can place his smartphone on a surface with reasonable flexibility, without having to worry about centimeter level differences in

its position and orientation (unlike *e.g.* EchoTag [135]). This claim will not be satisfied if VibroTag is too sensitive.

To understand VibroTag's sensitivity, each volunteer collected data in his restroom (on the toilet tank), on his bed, bedroom table, living room table and living room sofa. Users collected 25 to 30 samples from each surface for three different smartphone placement scenarios *i.e.* (1) *least restricted*, (2) *moderately restricted* and (3) *highly restricted*. Each scenario corresponded to three rectangular regions of different sizes. We marked the highly restricted region to be approximately within a few inches of the same dimensions as that of the smartphone, the moderately restricted region to be ∼4 times larger than that of the highly restricted region, and the least restricted region to be about ∼3 times larger in size compared to moderately restricted region. for example, Figs. 5.1(d) and 5.1(e) show the marked regions for a sofa and a worktable, respectively, in an apartment. For the experiments related to the least restricted region, volunteers were allowed to place their smartphone even beyond the third zone, as long as their smartphone was placed on the same surface. Figure 5.11 shows confusion matrix plots for the tested 5 classes, (namely Bed, Living Room Table, Living Room Sofa, Restroom Ledge and Kitchen Counter), for both User-1 using Nexus 4 (Figs. 5.11(a)-5.11(c)) and User-2 using OnePlus 2 (Figs. 5.11(d)-5.11(f)). For each scenario, confusion matrices are plotted using results from 2-fold cross-validation. We observe that for both users, highly restricting the device placement results in highest average prediction accuracy, (i.e. 92.16% and 97.03% respectively) which gradually decreases as restriction on smartphone's position and orientation changes from high to least. From the confusion plots (Fig. 5.11), we observe that the accuracies corresponding to User-2 are higher that User-1's. This may be because either OnePlus 2 is able to extract better quality vibration signatures than Nexus 4, or because User-2's environment and the tested surfaces therein were different from User-1's (*e.g.* User-1's bedroom table might have some light objects (*e.g.* keys) placed close to the smartphone that created noise when responding to the vibration). We discuss such impact of surrounding objects on VibroTag in §5.6.4.

Our results show that average accuracy corresponding to User-1's moderately restricted scenario are higher than highly restricted scenario (Figs. 5.11(a)-5.11(c)), which may be attributed to more



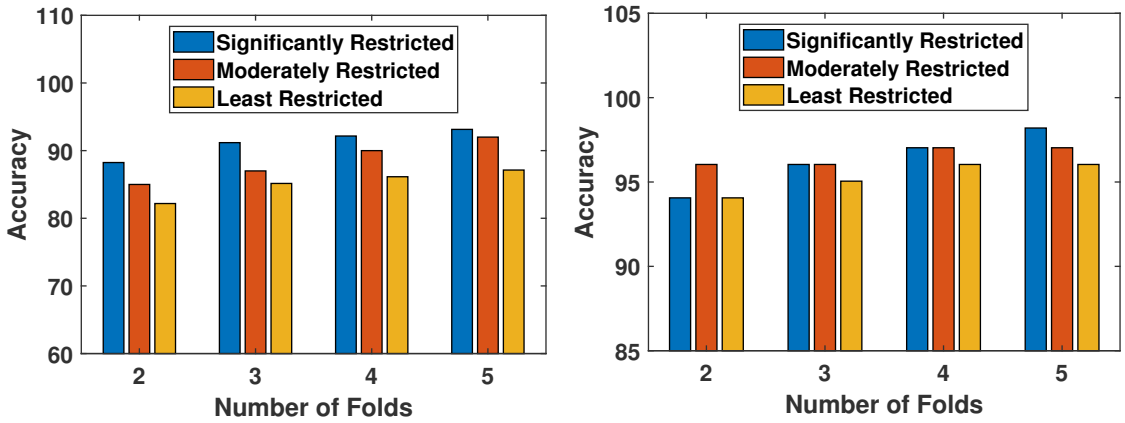
(a) High-User-1 (b) Moderate-User-1

(c) Least-User-1 (d) High-User-2

(e) Moderate-User-2 (f) Least-User-2

Figure 5.11: Confusion matrices for experiments performed by User-1 and User-2 to determine VibroTag's sensitivity

noisy samples obtained during highly restricted scenario. Figures 5.12(a) and 5.12(b) show average accuracy of all 5 classes for all 3 restriction scenarios obtained with 2-fold, 3-fold, 4-fold and 5-fold (*i.e.* increasing percentage data used for training from 50% to 80%) cross-validation classification



(a) k-folds on User-1 data

(b) k-folds on User-2 data

Figure 5.12: Average accuracy with increasing number of training samples (sensitivity experiments)

experiments. We observe that VibroTag performs well for all 3 restriction scenarios even when only 50% data is used for training and remaining for testing, and accuracies for even least restricted scenarios reach as high as 87% for User-1 and 95% for User-2 when percentage of training data reaches 80%.

5.6.4 VibroTag's Accuracy

5.6.4.1 Object and Location Recognition Accuracy

VibroTag achieves an average 4-fold accuracy of 86.55% when identifying different objects and locations, whereas the IMU based approach achieves only 49.25%. Table 5.1 shows average 2-fold and 4-fold classification accuracies obtained for 24 different objects (*e.g.* a box) and locations (*e.g.* kitchen counter) by User-1. For these experiment, User-1 collected 30-35 samples from each of those objects and locations in a moderately restricted manner. Our results show that VibroTag achieves an average (4-fold) recognition accuracy of 86.55%, which is 37% higher than the average accuracy achieved by the latest IMUs based approach [26], which achieves only 49.25%. Moreover, the lowest accuracy achieved by VibroTag is 70.33%, whereas the IMUs based method's accuracy goes as low as 16.38%. This shows that the features extracted using VibroTag can successfully differentiate between different objects, even when the objects are made of very similar material

(*e.g.* wood chair vs wood table, or metal drawer vs metal shelve).

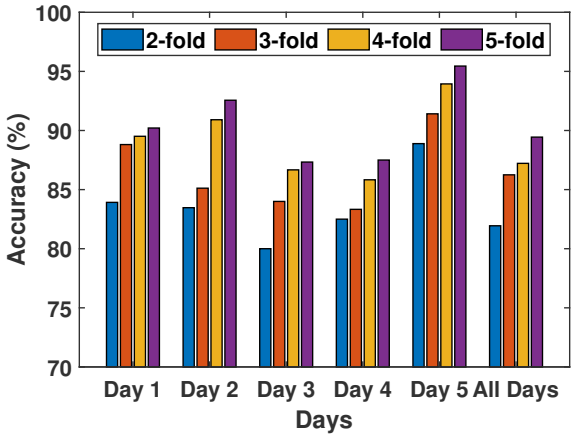
Table 5.1: Average accuracy of recognizing different surfaces in office and apartment scenarios

	Office environment											Apartment environment											
	Pages Bundle	Printer	Foam Chair	Metal Drawer	Carpet	Wooden Chair	Metal Shelve	Mouse Pad	Leather Chair	Center Desk (Wood)	Cardboard Box	XBox	Work Desk (Wood)	Window Ledge (Marble)	Bathtub Ledge	Living Table (Wood)	Glass Table	Kitchen Counter	Fridge	Wooden Floor	Bedroom Table (Wood)	TV Table (Wood)	Living Room Sofa	Microwave
	VibroTag's Accuracy																						
2-Fold	77.60	97.72	68.50	83.31	86.55	88.31	89.09	78.53	91.76	79.02	95.79	73.46	80.62	92.86	93.19	81.9	79.03	87.36	87.65	84.31	82.29	76.39	72.94	91.06
4-Fold	80.87	99.28	70.33	86.77	91.15	88.69	90.84	81.25	93.19	81.53	95.52	75.96	83.49	96.16	94.06	82.84	82.43	91.17	90.09	88.26	88.26	77.61	76.99	92.56
	State-of-the-Art IMUs Based Method's Accuracy [26]																						
2-Fold	68.64	87	30.35	68.3	35.13	94.94	89.96	64.05	75.45	27.07	34.29	19.55	49.41	65.08	38.78	25.18	16.89	19.42	56.60	32.98	51.10	30.07	29.31	64.98
4-Fold	75.63	90.39	31.23	70.02	37.48	95.91	91.52	66.31	78.23	29.69	40.21	21.56	52.92	66.95	40.26	25.92	17.04	20.23	63.40	33.98	52.35	32.53	32.27	65.27

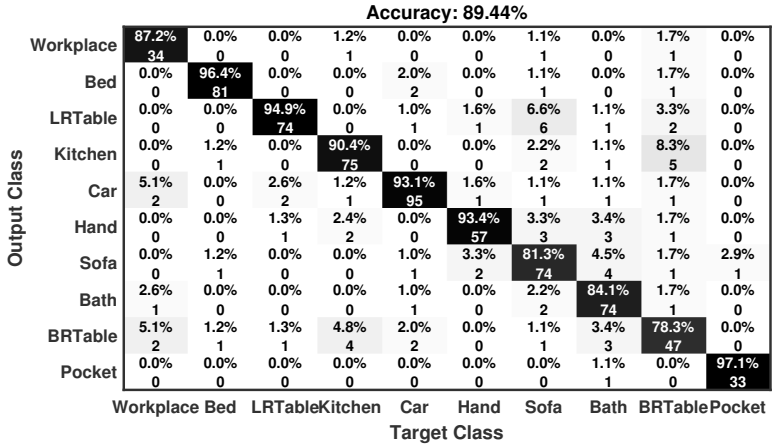
5.6.4.2 Location Recognition Accuracy over Days

VibroTag can maintain an average accuracy of up to 85% using training samples obtained for 3-4 days only. VibroTag's accuracy can change over days due to several different reasons, for example, changes in environmental noise and/or changes in position of other items placed on a surface (*e.g.* light objects such as keys, etc.). Next, we explore how VibroTag's accuracies change over days and how much training VibroTag requires to maintain high accuracies when testing on data from a new day.

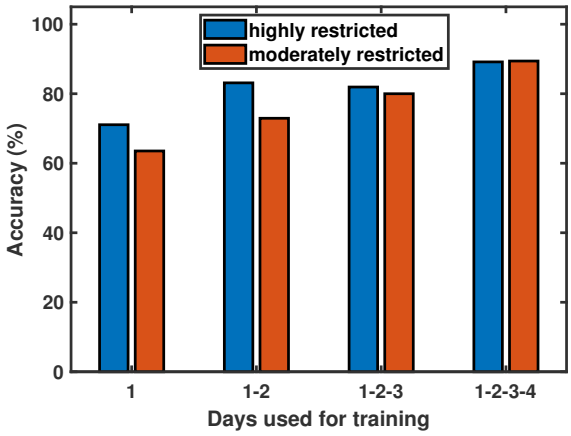
Figure 5.13(a) shows average cross-validation accuracy over all classes for data obtained from User-1 on 5 different days. The figure also shows cross-validation accuracy and confusion matrix obtained when data from all 5 days was combined. For these experiments, User-1 collected 6-20 samples from 10 different locations (*i.e.* Workplace (office table), Bed, Living Room Table, Kitchen (on marble counter), Car (small compartment in front of the gear stick), Hand, Living Room Sofa, Restroom (on toilet tank), Bedroom Table and Pocket) every day. For this set of experiments, data was collected for both highly and moderately restricted smartphone placement scenarios. Our results show that VibroTag achieves at least 79% (and at most 87%) accuracy everyday when using only 50% of a day's data for training. Figure 5.13(c) shows how combining data from previous days improves accuracy for data collected on the subsequent days from User-1. We observe that VibroTag can achieve accuracy of more than 80% on the unknown samples on day 5 for both restriction scenarios. Moreover, we observe that the accuracy of moderately restricted smartphone placement scenarios approaches highly restricted scenarios.



(a) Individual and all 5 days



(b) Confusion matrix for all 5 days data



(c) Consecutive days accuracy

Figure 5.13: (a) Average 4-fold cross-validation accuracies over all classes (User-1) (moderately restricted experiments), (b) Confusion matrix after cross-validation, (c) Training on data from previous days, testing on subsequent days

To understand how VibroTag's accuracy changes over days across multiple users, we collected 5 samples from Users 2, 3, and 4 for 4 different locations (*i.e.* Kitchen (on marble counter), Living Room Sofa, Restroom (on toilet tank), Bedroom Table and Living Room Table) for 20, 10, and 10 consecutive days, respectively. Fig. 5.14 shows how the classification accuracy changes for different users, where VibroTag is trained using the data from previous days and test on the subsequent days. We observe that the accuracy generally increases with days for User-2, however, there is a major dip for User-3 on day 5 and for User-4 between days 5-7, which can be attributed to major changes in the surrounding environment in terms of noise and/or addition/removal of different objects (such as keys or a pen) on the surface.

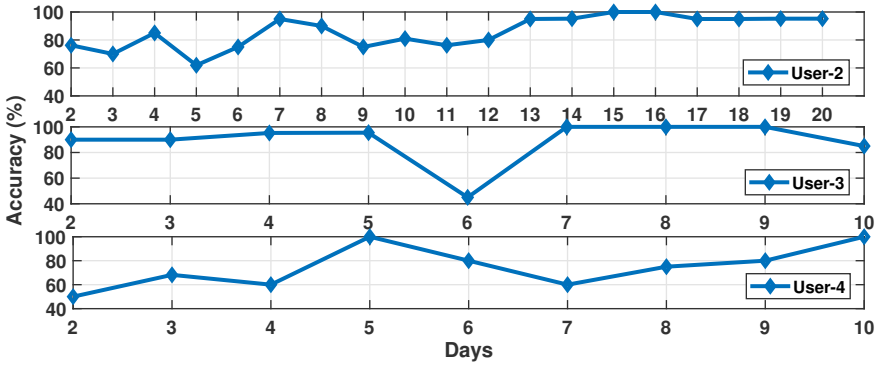
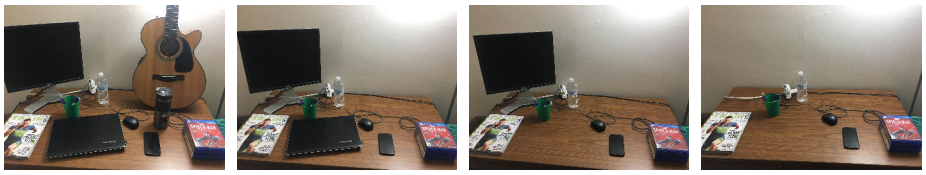


Figure 5.14: Accuracies on consecutive days

5.6.4.3 Impact of Surrounding Objects

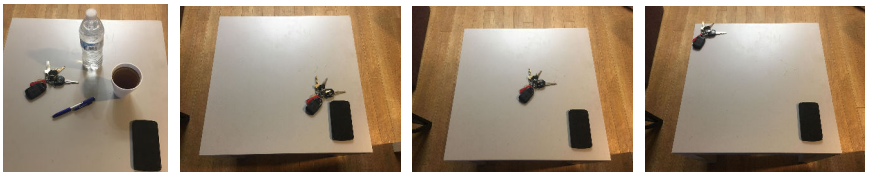
To understand the impact of surrounding objects on VibroTag's accuracy, we performed two different sets of experiments. In the first set of experiments, we collected data on a participant's bedroom table before and after removing 4 different heavier objects (*i.e.* a guitar, an LCD, a laptop and a mug) from the table one by one. Figure 5.15 shows the setup for these experiments, where 5.15(a) corresponds to the scenario where all objects were on the table, and 5.15(d) corresponds to the scenario where all three objects were removed. In the second set of experiments, we collected data on the participant's living room table by bringing 4 different lighter objects (*i.e.* a cup, a set of keys, a pen, and a water bottle) closer to the smartphone. Fig. 5.16 shows the setup, where 5.16(a) shows the different objects used in these experiments, and 5.16(b) - 5.16(d) shows a set of

keys being brought closer to the smartphone. Fig. 5.17 shows results for the aforementioned sets of experiments.



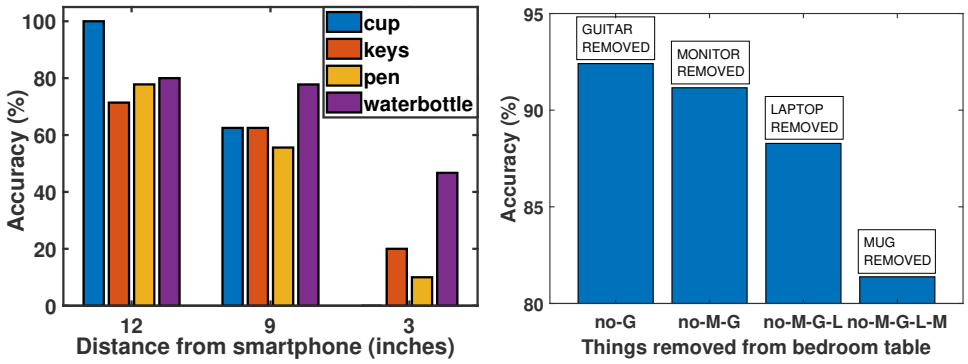
(a) setup (b) no guitar (c) no laptop (d) no LCD

Figure 5.15: Removing things from bedroom table



(a) setup (b) 3 inches (c) 9 inches (d) 12 inches

Figure 5.16: Bringing light objects closer to smartphone



(a) Living Room Table (b) Bedroom Table

Figure 5.17: Effect of (a) moving objects closer and of (b) removing objects on classification

We observe from Fig. 5.17(a) that as the lighter objects come closer to the smartphone (*i.e.* down from 12 inches to 3 inches closer), the classification accuracy of the table decreases significantly. For example, when the pen is within 3 inches of the phone, the accuracy goes as low as 9%. From Fig. 5.17(b), we observe that the impact of heavier objects on VibroTag's accuracy is not as significant as the lighter ones, which happens because the energy transfered by smartphone's vibration is not enough to make those objects vibrate significantly. However, we observe that the classification

accuracy still drops more than 10% as we slowly remove the objects that were previously placed on the table. This is because each of those objects has its own vibration response which was contributing to the overall vibration signature of the bedroom table. So, when those objects are removed one by one, their response is subsequently omitted in the new vibration signature of the surface, which leads to loss in the classification performance.

5.6.4.4 Impact of Upper Cut-Off Frequency

VibroTag achieves best accuracy when frequencies above 5500Hz are filtered out from the recorded sound signals. A smartphone's microphone can usually capture sounds in the frequency range of 20Hz - 20kHz. However, the smartphone's vibration usually causes variations in lower frequencies, and therefore, filtering out higher frequencies can reduce impact of background noise and any unwanted noisy variations. To understand which frequencies can be filtered out to achieve best accuracy in VibroTag, we employ a Butterworth band-pass filter, and determine the average accuracy for 5 different upper cut-off frequencies of the filter. Figure 5.18 shows how User-3's (OnePlus 2) multi-fold cross-validation accuracies vary as upper cut-off frequency increase from 1500 to 12000. We observe that VibroTag achieves best accuracy at cut-off frequency of 5500Hz. As the upper cut-off frequency increases, it allows higher frequency noisy variations in the vibration signatures, which leads to lower classification accuracies. We observed similar results for other users as well. Therefore, all accuracies reported in this work correspond to 5500Hz cut-off frequency. Note that other smartphones may exhibit better accuracies for cut-off frequencies which are slightly different from 5500Hz, however, this is an aspect which is out of the scope of this work.

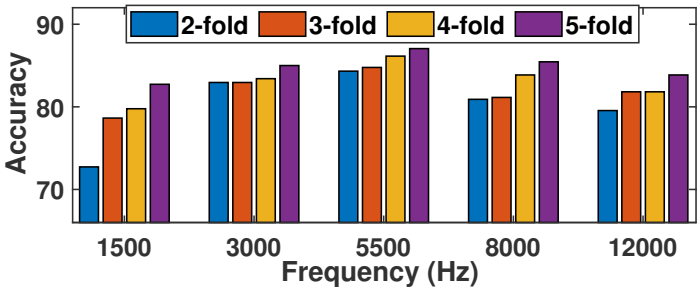


Figure 5.18: Cross-validation accuracies for different band-pass filter upper cut-off frequencies (User-3)

5.7 Usability Study

We carried out a usability study and asked 24 participants (20 male, 4 female), recruited at university, about the flexibility and usability of the VibroTag application in daily life. The participants comprised of students and university employees of ages 19 to 35. They were first briefed about the working of VibroTag and then its target applications such as symbolic localization. The volunteers were shown the VibroTag application interface as pictured in Figure 5.10(a) and given a demo of the acoustic trace collection. They were also briefed on how the smartphone can be placed on a surface with 3 different levels of restriction flexibility. At the end, they were given a set of usability questions given below and summarized in Figure 5.19: **Q1.** Are you comfortable using smartphone for location recognition? **Q2.** Can VibroTag help you save time by setting reminders? **Q3.** Are you comfortable with VibroTags' use of vibration? **Q4.** Is it easy to place smartphone on preferred locations for learning? **Q5.** Can VibroTag help you in setting smart notifications linked to locations? **Q6.** Is VibroTag useful in activating other smart applications? **Q7.** Do you find VibroTag application valuable and fun to use?

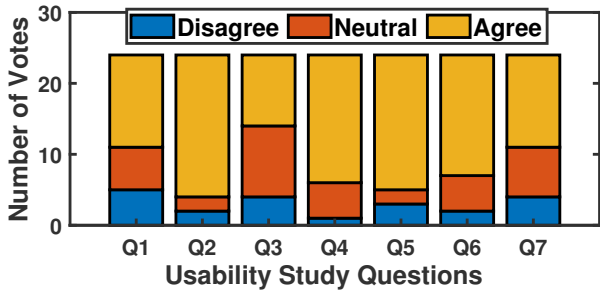


Figure 5.19: Vote distribution of 7 VibroTag's usability questions asked from 24 participants

Our study indicates high agreement on the usefulness of VibroTag based smart notifications and reminders. It also indicates some discomfort in the use of vibration.

5.8 Conclusion

In this work, we make the following contributions. First, we propose the first fine-grained vibration based sensing scheme, that can recognize different surfaces using the vibration mechanism and microphone of a single COTS smartphone. The intuition is that the smartphone's vibration

causes the whole smartphone structure and the hardware inside it to vibrate in a peculiar pattern, which depends upon the absorption properties of the surface that the smartphone is placed on. These vibrations produce peculiar sound waves that we detect using the smartphone's microphone. Second, we propose a novel signal processing technique to extract fine-grained vibration signatures that are robust to hardware irregularities and background environmental noises. We implemented VibroTag on two different Android phones and evaluated in multiple different environments. Our results show that VibroTag achieves an average surface recognition accuracy of 86.55%, which is 37% higher than the average accuracy of only 49.25% achieved by the state-of-the-art IMUs based schemes.

CHAPTER 6

DISTRIBUTED SPECTRUM SHARING FOR ENTERPRISE POWERLINE COMMUNICATION BASED IOT NETWORKS

6.1 Introduction

As powerline communication (PLC) technology does not require dedicated cabling and network setup, it can be used to easily connect multitude of Internet of Things (IoT) devices deployed in enterprise environments for sensing and control related applications. Thanks to the *plug-n-play* nature of PLC technology, a PLC enabled device just needs to be connected to a wall socket, and it will automatically form a mesh network with nearby PLC devices. IEEE has standardized the PLC protocol in IEEE 1901, also known as HomePlug AV (HPAV) [3, 5], which has been widely adopted in mainstream PLC devices.

A key weakness of HPAV protocol is that it does not support spectrum sharing. Currently, each link in an HPAV PLC network operates over the whole available spectrum, and only one link can operate at any time within a single collision domain. Figure 6.1 shows an example enterprise level IoT application scenario, where multiple PLC nodes (including multiple gateway nodes) are connected in the same MAC collision domain to a power distribution network. Currently, two *disjoint* PLC links (*e.g.* 5-8 and 12-11 in Fig. 6.1) cannot operate concurrently with existing HPAV MAC protocols. However, in real enterprise PLC deployments, we often encounter scenarios where a subset of subcarriers on some PLC links are highly underutilized as compared to other links, which implies that the low-modulated subcarriers of one PLC link can be utilized by one of the other links to improve the aggregated throughput. Moreover, if multiple PLC links, which may be competing for the same channel simultaneously, can operate in parallel via sharing spectrum, many costly collisions can be avoided.

In this work, through an extensive measurement study of HPAV PLCs in a real enterprise environment using commodity off-the-shelf (COTS) HPAV PLC devices, we discover that spectrum sharing can significantly benefit enterprise level PLC networks. Our first finding is that PLC

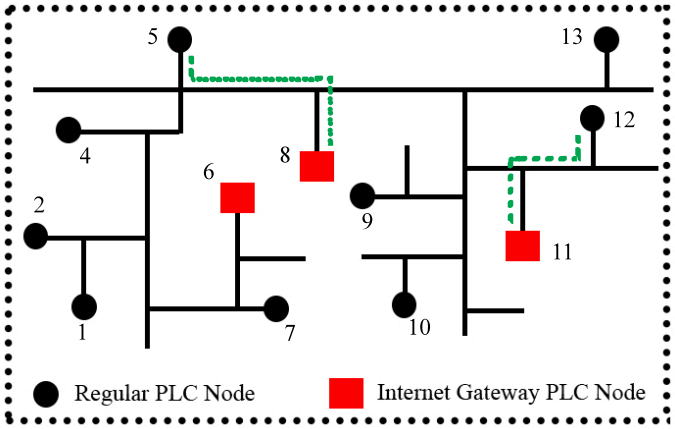


Figure 6.1: Example scenario: Links 5-8 and 12-11 in the same collision domain can share spectrum for concurrent operation

nodes connected under the same circuit breaker in a building's power distribution network can communicate at 6.5 times higher throughput than the PLC nodes connected under two different breakers, and 18-30 times higher throughput than the PLC nodes connected to two completely different power distribution/trunk lines. This implies that enterprise PLC networks must have at least one gateway node connected under every breaker, to provide best possible connectivity to the IoT devices connected under that breaker. As each power distribution line can contain tens to hundreds of breakers with multitude of IoT devices connected to a gateway under each breaker, the number of *disjoint links*, which consist of different source-destination pairs and may compete for the same channel simultaneously, becomes significant. Second, based on our subcarrier level spectral analysis, we observe that PLC channels of more than 50% of the PLC links are significantly different from each other due to highly location dependent multipath characteristics. As the performance of different frequency subcarriers varies among different PLC links, low-modulated subcarriers of one link can be utilized by other links, and vice versa. Third, most links in an enterprise PLC network are *pseudo-stationary*, *i.e.* the channel characteristics between any two PLC nodes have low temporal variability (standard deviation of throughput observed over 15 minute time windows is below 2.2Mbps for more than 80% of the links), and therefore, a spectrum sharing scheme can be achieved at low channel estimation related control overhead.

Multiple Frequency Division Multiplexing (FDM) based spectrum sharing techniques have

been proposed for PLCs [7, 51]. However, such spectrum sharing techniques have three major limitations. First, they are incompatible with the HPAV MAC, which makes them difficult to be adopted. Second, they are designed for WiFi like point-to-multipoint communications. This may be suitable for home PLC networks where a few IoT devices are connected to a single PLC gateway node. However, it is unsuitable for enterprise PLC environments, as enterprise PLC networks are mesh networks, with multitude of disjoint links between IoT devices and their respective gateway nodes. Third, they have prohibitively high computational and control overheads involved in their underlying subcarrier assignment and bit loading algorithms. This makes them impractical for real world deployment.

In this work, we aim to design a spectrum sharing scheme which is compatible with HPAV MAC, is suitable for enterprise level PLC mesh networks, and incurs minimal computational and control overheads. To this end, we propose an HPAV compatible, distributed, low overhead spectrum sharing approach for enterprise PLC networks. Currently, HPAV MAC protocol uses Carrier-Sense Multiple Access with Collision Avoidance (CSMA/CA) and Time-Division Multiple Access (TDMA) techniques for sharing medium access among PLC nodes. To make our scheme compatible with existing HPAV MAC, we design it such that any link which occupies the PLC channel following the regular HPAV CSMA/CA or TDMA protocol shares a part of its spectrum with another link to improve the aggregated throughput of both links. Moreover, we design our scheme such that it can be enabled in the current HPAV PLC devices while incurring minimum firmware level changes. We call the links which occupy the PLC channel following regular HPAV CSMA/CA or TDMA protocol as *primary links*, and the links with which the primary links share their spectrum as *secondary links*. To make our scheme suitable for enterprise level PLC mesh networks, we develop a distributed spectrum sharing strategy. To achieve this, we develop an optimal spectrum sharing algorithm which each node uses to locally compute a complete set of network-wide spectrum sharing rules for all possible *primary* links and their corresponding *secondary* links in the network. Our algorithm leverages subcarrier level channel information corresponding to all possible links in the network to compute those rules. Based on these rules,

any *primary* link can decide which of the possible *secondary* links should it share its spectrum with, and what part of the spectrum should it share, to achieve best possible spectrum sharing gains. When the source node of a *primary* PLC link gets channel access, it broadcasts the link's source-destination IDs to all the remaining nodes in its network. Next, it picks one of the possible *secondary* links to share its spectrum with, based on its locally computed network-wide spectrum sharing rules, and then continues its remaining transmission in the unshared region of spectrum. Meanwhile, the source and destination nodes of the chosen *secondary* link establish connection, and start operating in parallel with the *primary* link over the shared region of spectrum. This happens automatically, as both source and destination nodes of the chosen *secondary* link already know the source-destination IDs of the *primary* link and have the same set of network-wide spectrum sharing rules. Transmission of *secondary* link finishes as soon as the *primary* link finishes its transmission. To minimize the computational and control overhead of our scheme, we take the following design decisions: First, we design our spectrum sharing algorithm such that the basic optimization problem which it solves comes down to optimally sharing spectrum between just two links (*i.e.*, a *primary* and a *secondary*), which is a computationally simpler problem to solve than sharing spectrum with several links simultaneously. Second, we design our scheme to operate in a distributed manner, where each node locally computes network-wide spectrum sharing rules. This makes real-time spectrum sharing seamless, as it completely avoids any extra control related communications for coordinating spectrum sharing in the network. Third, our design takes advantage of the *pseudo-stationary* nature of enterprise PLC channels to reduce channel estimation related overhead. The computation of network-wide spectrum sharing rules at each node requires latest subcarrier level channel information of all possible links in the network. To achieve this, each node first gets channel information corresponding to all possible links it can form, and then shares that information with other nodes in the network, which can involve considerable communication overhead. However, as most PLC channels in an enterprise setting are *pseudo-stationary*, PLC nodes do not need to update their copy of network-wide channel information too frequently. Therefore, the frequency of channel probing is significantly reduced, which maintains the spectrum sharing gains.

We implement and evaluate our proposed spectrum sharing techniques on HPAV CSMA protocol only, as the integration of our spectrum sharing technique with HPAV TDMA protocol is relatively straightforward to achieve (we present a detailed discussion on this in §6.6). We perform trace driven simulations using channel response (*tonemap*) traces collected from seven different 4-node PLC deployments. We show that fine-grained distributed spectrum sharing can boost the aggregated and per-link throughput by more than 60% and 250% respectively.

6.2 Related work

Hayasaki *et al.* [51] and Achaichia *et al.* [7] have proposed FDM based multiple access techniques in the context of point-to-multipoint communication in PLC networks. Hayasaki *et al.* [51] proposed a theoretical bit-loading based OFDMA scheme for in-home PLCs, as an alternative to TDMA/CSMA based medium access. Their scheme consists of two iterative algorithms: a subcarrier assignment algorithm and a bit-loading algorithm. The subcarrier assignment algorithm assigns subcarriers to maximize the whole throughput, while satisfying the minimum throughput guarantees of each destination PLC node first. Afterwards, the bit-loading algorithm is utilized for loading bits into the assigned subcarriers, while optimizing both the bit quantity on each subcarrier as well as the whole code rate, subject to BER constraints on each subcarrier. Achaichia *et al.* [7] proposed a similar technique named *Tone Maps Splitting Algorithm* (TMSA) to orthogonalize spectrum assigned to multiple active links in a point-to-multipoint communication. However, the aforementioned techniques are designed for WiFi like point-to-multipoint communication in PLCs, and proposed as an alternative MAC protocol to existing HPAV TDMA/CSMA based MAC. Therefore, their techniques are incompatible with current HPAV MAC. Moreover, the aforementioned techniques involve high computational and control overheads corresponding to their underlying resource allocation schemes (i.e. subcarrier assignment and bit loading algorithms), which makes them impractical for real world deployment scenarios. In contrast, our goal is to design a distributed spectrum sharing technique for HPAV PLC networks while incurring minimal computational and control overheads. Moreover, our aim is to augment and integrate spectrum sharing on top of exist-

ing TDMA/CSMA MAC used in the mainstream PLC devices such as HPAV, etc. while incurring minimal firmware level modifications.

In [93, 142] authors compare HPAV with WiFi performance. They study temporal and spatial variations of the throughput of PLC links and make a case for hybrid PLC-WiFi networks [142]. The measurement study in [14] shows the multi-flow performance of PLC networks. It then presents BOLT, which seeks to manage traffic flows in PLC networks. The above studies ignore the spectral inefficiencies at MAC layer of HPAV networks. In contrast, we extensively study the behavior of PLCs in spatial, temporal and spectral dimensions, and propose novel spectrum sharing strategies to improve per-link and aggregated throughput of enterprise level PLC networks.

6.3 HomePlug AV Powerline Communications

6.3.1 PLC Channel Characteristics

Multipath is a key characteristic of PLC channels, which is attributed to unmatched electric loads or branch circuits connected to different sockets on the powerline. In a typical power distribution network of a large building, there are multiple branch circuits with different impedances, and therefore, PLC signals are reflected from multiple reflection points leading to multipath effects. On top of multipath attenuations, several different types of noise in PLC channels have been identified [22, 36]. Harmonics of AC mains and other low power noise sources in the power lines lead to colored background noise, which decreases with frequency. Periodic impulsive noise is created due to rectifiers, switching power supplies and AC/DC converters, which can be either synchronous or asynchronous with AC line cycle. Aperiodic impulsive noise also exists in PLC channels due to switching transients in power supplies, AC/DC converters, etc.

6.3.2 HomePlug AV standard

The most widely adopted family of PLC standards are HomePlug AV, AV2 and Green PHY standards [72]. HomePlug AV2, which is the latest of these standards, can support up to 1 Gbps PHY rates. Our study focuses on the HomePlug AV standard, which has been widely used in home

networks to improve coverage, and can support maximum PHY rates of up to 200 Mbps [3, 5]. However, our findings and solutions can also be generalized for PLC technologies other than HPAV, such as HPAV2.

HPAV PHY-layer: HPAV uses 1.8-30 MHz frequency band and employs OFDM with 917 subcarriers (for the USA devices), where each subcarrier can use any modulation scheme from BPSK to 1024-QAM depending on the channel conditions [72]. In order to update the modulation schemes for each subcarrier, two communicating HPAV PLC devices continuously exchange and maintain *tonemaps* between them. Tonemaps refer to the information about the modulation scheme used per subcarrier, i.e. the number of bits modulated per subcarrier. The tonemaps exchanged are estimated for multiple different sub-intervals of the AC mains cycle. Tonemaps are exchanged between PLC devices through a *sounding* process, where the transmitter sends sounding frames to the receiver using QPSK for all subcarriers, the destination estimates the channel quality and sends back the tonemaps corresponding to different sub-intervals of AC mains cycle back to the transmitter. The destination can communicate up to 7 tonemaps, i.e. 6 tonemaps for the different sub-intervals of the AC line cycle called slots and one default tonemap [72], depending on the condition of noise and attenuation observed in different parts of AC line cycle. Tonemaps are continuously updated by default after 30 seconds or when the error rate exceeds a threshold [72]. *Tonemaps provide us with the information about Channel Frequency Response (CFR) of the channel between two communicating PLC devices.*

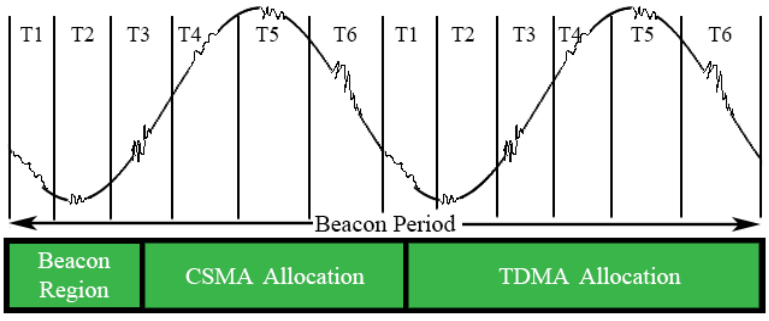


Figure 6.2: Basic Beacon Period structure in HPAV MAC

HPAV MAC-layer: MAC-layer of HPAV based PLCs works very differently from that of WiFi MAC. First, unlike WiFi, *channelization* is not allowed and not used in PLCs, which limits the

possibility of deploying non-interfering networks, such as WiFi networks on different channels. Second, there is no concept of a central *Access Point* (AP) in PLCs. There exists a dynamically chosen central authority to manage network, called the *Central Coordinator* or CCo, and large PLC networks can contain multiple CCo's managing their own collision domains. However, CCo's role is passive, mainly authentication and association of new nodes, monitoring the network, synchronizing it with the AC line cycle, and taking time-division access decisions in terms of allocating TDMA and CSMA/CA slots. In contrast, the WiFi AP-mode forces downlink/uplink traffic types and a star-like logical network. PLCs only form mesh networks, and every node can communicate with its peers, without relaying through the CCo. Both TDMA and CSMA/CA are supported by HPAV [72]. Tonemaps are optimized for the QoS required for the traffic in the TDMA allocations. HPAV uses a Beacon Period, managed by a CCo, for allocating CSMA and TDMA sessions (Fig. 6.2). The Beacon Period is synchronized with AC line cycle and is two AC line cycles in length. The schedules advertised in the Beacon are persistent and are not changed for a number of Beacon Periods. The CSMA protocol of HPAV devices (IEEE 1901) is different from CSMA/CA used by WiFi devices (IEEE 802.11). Both use a time-slotted random backoff with a *backoff counter* (BC) and a *contention window* (CW). 1901 includes two more counters in its backoff procedure, i.e., *backoff procedure counter* (BPC) and the *deferral counter* (DC). Using DC based backoff procedure, HPAV PLC nodes increase their contention windows not only after a collision, but also after sensing the medium to be busy, which reduces the probability of collision. Details of the 1901 backoff procedure can be found in [141]. Request to Send (RTS) and Clear to Send (CTS) delimiters can be enabled during CSMA slots to handle hidden nodes. HPAV frames are 512 byte aggregated physical blocks (PBs) of data. In order to reduce protocol overheads, HPAV employs two-level frame aggregation, where the First, the data is organized in 512 byte physical blocks (PB). PBs are then aggregated into HPAV frames. Reception of each PB of a frame is separately acknowledged, so that the transmitter retransmits only the corrupted PBs.

6.4 A Measurement Study of Enterprise PLCs

Experimental setup: Our study is based on measurements with commodity HomePlug AV hardware. We use Meconet HomePlug AV mini-PCI adapters with Intellon INT6300 chipsets, which can support 200 Mbps PHY rates. We connect the PLC adapters to ALIX 2D2 boards, which run OpenWrt operating system. We use open source PLC software tool named *open-plc-utils*, which is developed by Qualcomm, to extract PHY and MAC-layer feedback (such as tonemaps), directly from the Meconet HPAV adapters. Note that the measurements, analysis and solutions proposed in this work can also be applied to newer HomePlug AV2 devices with a few modifications.

Experimental methodology: For our experiments we place our PLC nodes in various locations of an enterprise building. We generate saturated *iperf* UDP traffic among the PLC nodes. Results we report are averaged over multiple runs.

Metrics: We analyze the performance of PLC networks by first collecting *iperf* throughput statistics. We further elaborate on the per-subcarrier PLC network performance by analyzing the tonemaps extracted by the *open-plc-utils* tool running on PLC nodes. For a given PLC communication link and for the k^{th} sub-interval of AC line cycle, the effective PHY rate can be estimated from tonemaps as $R_{phy}^{\{k\}} = \frac{[\sum_{j=1}^{N} T[j]^{\{k\}}] \cdot C^{\{k\}} \cdot (1 - B_{err}^{\{k\}})}{T_s}$ [4], where j is subcarrier number and N is total number of subcarriers. $T[j]$ is the modulation rate (i.e., bits per subcarrier) of the j^{th} subcarrier [4]. C is Forward Error Correction (FEC) code rate. HomePlug AV supports FEC code rates of 1/2 and 16/21. Finally, B_{err} is the bit error rate and T_s is the symbol interval of OFDM communication. T_s is approximately $\sim 46\mu s$ for HomePlug AV including all overheads [72]. The expected throughput, averaged over all the sub-intervals of the AC line cycle, can be written as $\mathcal{T} \approx (1 - F_o) \cdot \sum_{k=1}^{N_{AC}} R_{phy}^{\{k\}} / N_{AC}$. Here F_o accounts for HPAV protocol overheads and N_{AC} is the number of sub-intervals of AC line cycle. N_{AC} is 5 or 6 for USA frequencies and F_o is typically ~ 0.4 based on *iperf* throughput measurements. In all our experiments, we observed that the FEC code rate was always 16/21 for the communication among our HPAV devices. Therefore, we assume FEC code rate of 16/21 in rest of the work, unless explicitly mentioned otherwise.

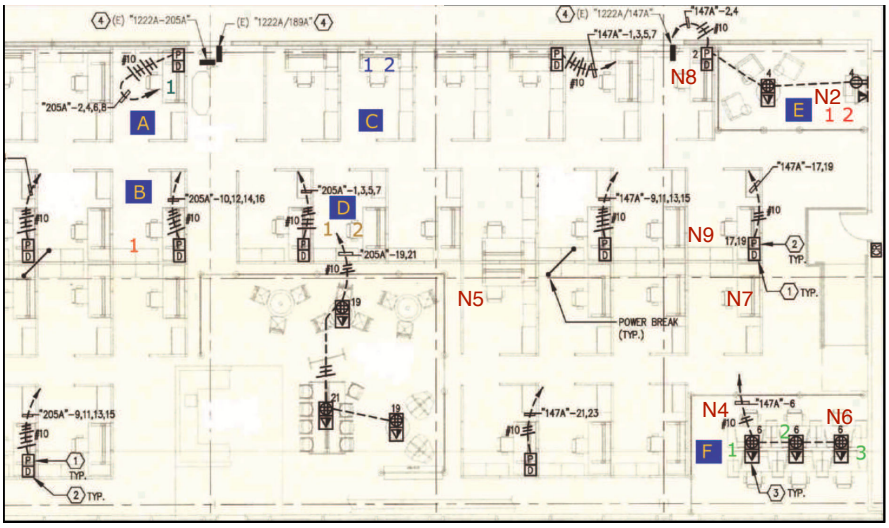
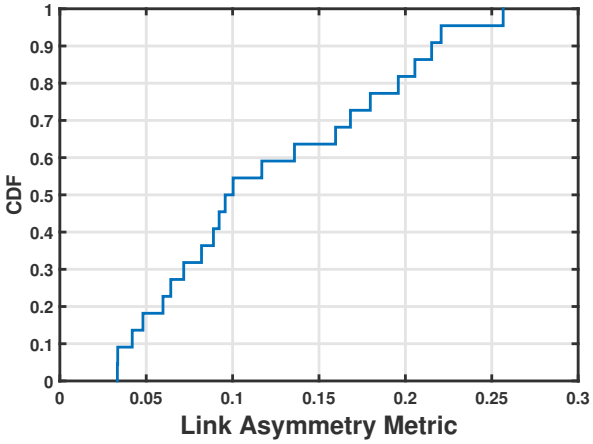


Figure 6.3: Building power distribution plan

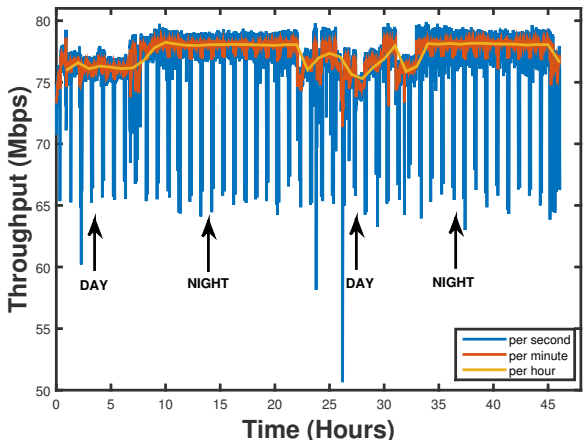
6.4.1 Many Disjoint PLC Links Compete for Channel Access

To understand how significant the number of *disjoint links* can become in a an enterprise level PLC network for IoT applications, we study the impact of different components of a power distribution network (*e.g.* phases, breakers and distribution/trunk lines) by measuring the throughput performance of more than 40 links (PLC transmitter-receiver pairs). Power distribution network floorplan of the enterprise building, where we conducted our experiments, is shown in Figure 6.3. The main switchboard of the enterprise, steps down the voltage from thousands to hundreds of Volts and the down converted electric power is then distributed towards different floors of different buildings in the enterprise, through multiple different distribution lines or *trunk lines* [161] (represented with hexagonal boxes with #4 written on them). The power from the trunk lines coming into the floor of a building is then further distributed into different parts of the floor, through a distribution board containing a set of circuit breakers which divide the electrical power feed into subsidiary circuits. Each trunk line consists of 3 cables corresponding to 3 different phases and each distribution board contains multiple breakers per phase. The letters (A-E) and numbers in the floorpan of Figure 6.3 represents some of the different locations where we placed our PLC nodes. Next, we elaborate on our experiments.

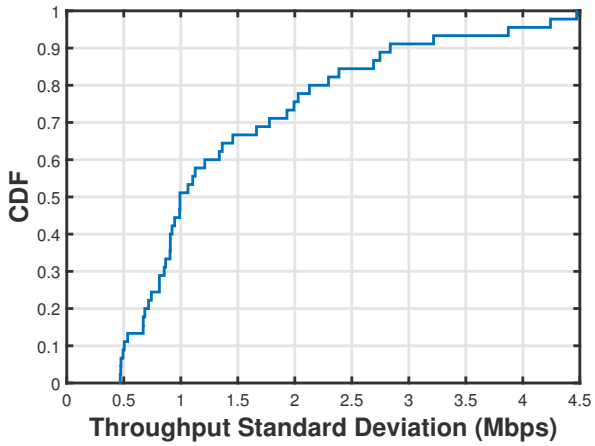
Case 1: We observed that the performance of a PLC link operating on same breaker and same



(a) Link asymmetry.



(b) PLC throughput timeseries (48 hours).



(c) Link stability.

Figure 6.4: (a) Link asymmetry, (b) Temporal variation in throughput over 2 days, (c) Link throughput stability CDF (45 links)

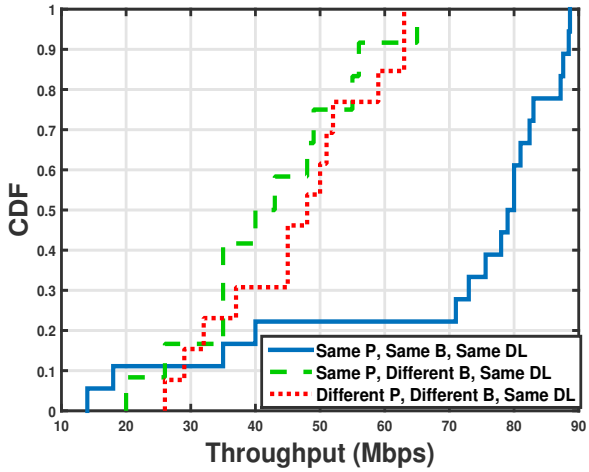


Figure 6.5: CDF of throughputs observed in different cases

distribution line is mainly affected by the location of PLC nodes with respect to the interfering electrical appliances. Highly attenuating device impedances or severe device interferences can lead to significant performance degradation (we observe ~6.5 fold decrease in throughput). Moreover, as shown by the CDF in Fig 6.5, throughputs of more than 70 Mbps were observed across the tested links approximately 75% of the time. Jitter was low, with the median being 0.2 ms and a maximum of 2.5 ms.

Case 2: PLC nodes connected to same (or different) phase but different breakers operate over lower throughputs as compared to same phase, same breaker case (~20-30% decrease in observed throughput)[1]. This is because signals experience higher attenuations while passing through the breaker circuitry located between the PLC nodes. We observed maximum throughput of 63 Mbps, which is 25.6 Mbps lower (29% decrease) than the previous case where nodes were connected under the same breaker. The median throughput observed was 51 Mbps, with minimum being 26 Mbps, which is higher than the minimum of same breaker case as we did not encounter any high interference from electric appliances this case.

Case 3: PLC performance significantly drops (~18-30 folds throughput decrease) when nodes are located at different distribution lines. Distribution lines can make PLC connectivity often impossible, due to transformers in between. The maximum throughput that we observed between any two pair of nodes was 3 Mbps and 5 Mbps for both directions, and the jitter varied between 2.03 ms and 5.7 ms.

Conclusions: PLC nodes connected under the same breaker in a building's power distribution network can communicate at 6.5 times higher throughput than the PLC nodes connected under two different breakers, and 18-30 times higher throughput than the PLC nodes connected to two completely different power distribution/trunk lines. This implies that enterprise PLC networks must have at least one gateway node connected under every breaker, to provide best possible connectivity to the IoT devices connected under that breaker. As each power distribution line can contain tens to hundreds of breakers with multitude of IoT devices connected to a gateway under each breaker, the

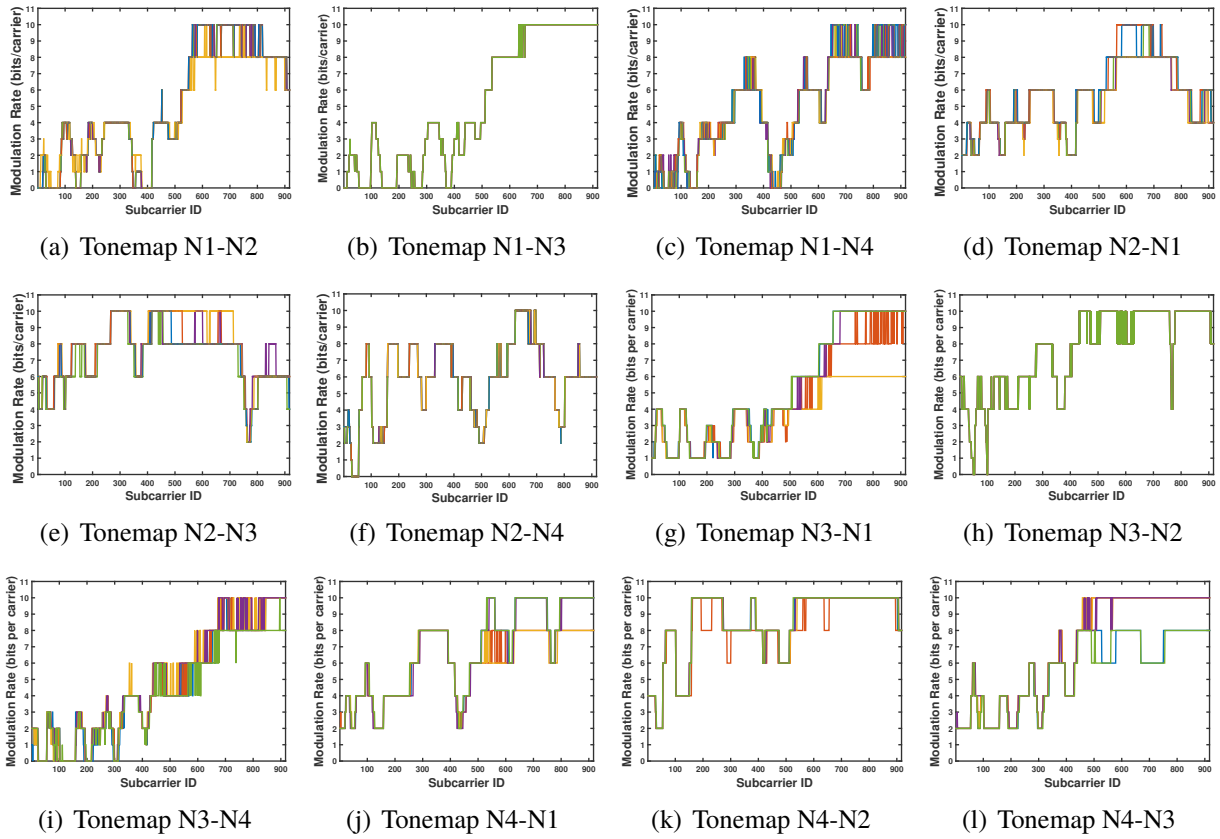
[1] We have excluded the cases of high interference from electric devices.

number of *disjoint links*, which consist of different source-destination pairs and may compete for the same channel simultaneously, becomes significant.

6.4.2 Enterprise PLC Channels are Highly Location Dependent

We measure the intensity of location dependence of PLC channels through a PLC *link asymmetry* metric $\mathcal{A}_{a,b}$. Asymmetry of a PLC link depends on channel frequency response or transfer function between PLC nodes communicating over that link, and it can be directly attributed to the different multipath characteristics of the powerline, which can vary depending on the location of PLC nodes compared to branch circuits or other connected electrical devices [171–173] (*i.e.* location dependent multipath characteristics). We quantify asymmetry of a PLC link $a - b$ as $[\mathcal{A}_{a,b} = \sum_{k=1}^{N_{AC}}[\sum_{j=1}^{N} |T_{a\rightarrow b}[j]^{\{k\}} - T_{b\rightarrow a}[j]^{\{k\}}|]]/N_{AC}$, where N is the number of subcarriers, T_j is the modulation rate of the j^{th} subcarrier and N_{AC} is the number of sub-intervals of AC line cycle. The above equation estimates asymmetry between two links as the distance between tonemaps of these links, averaged over all AC line cycle sub-intervals. The max and min values for $A_{a,b}$ are 9170 (917 subcarriers $\times$ 10 bits/carrier) and 0, respectively. In Figure 6.4(a) we present the distribution of our link asymmetry metric $A_{a,b}$ normalized by the maximum $A_{a,b}$ (which is 9170), from the tonemaps of 25 pair of nodes in the same neighborhood a, b. We observe that for more than 50% of the links, the normalized $A_{a,b}$ is greater than 0.1 (917 bits). The maximum throughput difference observed in asymmetric links is 15 Mbps.

Figure 6.6 shows snapshots of the tonemaps of 12 different links from a real world scenario, where we deployed a network of 4 PLC nodes in our test environment. We observe that the same subcarriers perform differently for different links. If we consider the last 200 subcarriers (717-917) for all the links of node N1, we observe the modulation is at least 6 bits per carrier (cf. Figures 6.6(a), 6.6(b), 6.6(c)). On the other hand, the last 200 subcarriers for all the links of node N2, show lower modulation, which can be as low as 2 bits per carrier (cf. Figures 6.6(d), 6.6(e), 6.6(f)). The modulations of N2's links, for the first 100 subcarriers (e.g. 1-100), are overall better compared to the corresponding modulations of N1's links. A spectrum sharing strategy could allow both N1



(a) Tonemap N1-N2 (b) Tonemap N1-N3 (c) Tonemap N1-N4 (d) Tonemap N2-N1

(e) Tonemap N2-N3 (f) Tonemap N2-N4 (g) Tonemap N3-N1 (h) Tonemap N3-N2

(i) Tonemap N3-N4 (j) Tonemap N4-N1 (k) Tonemap N4-N2 (l) Tonemap N4-N3

Figure 6.6: Tonemaps of 12 links among 4 PLC nodes in one of our PLC deployments, showing possibility of gains from SS

and N2 to transmit at the same time to their neighbors (e.g. N1-N3 and N2-N4) using only their high-performance subcarriers. Similar observations hold for other links (tonemaps not shown here), where certain subcarriers cannot carry data (0 modulation) and others can allow high modulations.

Conclusion: Per-subcarrier performance can vary significantly among different links in enterprise PLC networks. Therefore, the low-modulated subcarriers of one PLC link can be utilized by other PLC links, and vice versa.

6.4.3 Enterprise PLC Channels are Pseudo-Stationary

Performance of PLCs in enterprise settings can be dynamic either due to interference from already connected appliances, or due to a multitude of electrical devices being turned on/off on a regular basis. In order to study temporal dynamics, we measure performance of a PLC link for a long time periods. Figure 6.4(b) shows a representative scenario of a PLC link throughput

variation, for 2 days (48 hours) period. The throughput variations are averaged over one second, one minute and one hour time windows respectively. We observe that the throughput performance can vary from 52 Mbps to 80 Mbps. The link appears to be highly bursty, which shows that some intense performance dynamics happening at small time scales, which are attributed to interference created by nearby electrical devices. The throughput variations observed at coarser time scales (minutes or hours) are attributed to human activity (e.g. connection/disconnection of new devices, etc.). The analysis of tonemaps (not shown here) also verifies the link variations with time, as we observed that the tonemaps exchanged among PLC nodes during day were different from those during night. However, we observed that throughput between most PLC links remained quite stable. Figure 6.4(c) shows the CDF plot of standard deviation (averaged over 10 second intervals) of the real time throughput of 45 different links we tested in our building. Throughput for each link was collected over 15 minute time windows. It can be observed that more than 60% of the time, the standard deviation of throughput is below 1.5 Mbps, which shows that throughput performance of most PLC links remains consistent over time.

Conclusion: Most links in an enterprise PLC network are *pseudo-stationary*, i.e. the channel characteristics between any two PLC nodes have low temporal variability. Therefore, a spectrum sharing can be realized at low control overheads.

6.5 Distributed Spectrum Sharing for HPAV PLCs

In this section, we lay the theoretical foundations of our proposed spectrum sharing (SS) strategy. Our proposed techniques can be generalized for other PLC technologies, such as HPAV2, which use bit-loaded OFDM at PHY layer, as our technique shares spectrum at OFDM subcarrier level.

6.5.1 Preliminary Definitions

Primary & Secondary Links. We call the links which occupy the PLC channel through regular HPAV CSMA/CA or TDMA protocol as *primary* (P-Link or $p_{i\rightarrow j}$), and the links with which a primary link shares spectrum with, as *secondary* (S-Link or $s_{m\rightarrow n}$). Whenever a P-Link is established, only one S-Link can operate during that communication slot. For example, if we

assume that all links are saturated (i.e., each node always has traffic to send), the S-Link which gives maximum possible gain by sharing spectrum with an established P-Link will operate in parallel with that P-Link. Later on, we will present a ranking based strategy which each node in the network can follow locally to resolve contention for S-Link.

Tonemaps. Let $[T_{i\rightarrow j}]^{p}_{1\times N}$ and $[T_{m\rightarrow n}]^{s}_{1\times N}$ be the vector of tonemaps of a pair of P-Link ($i \rightarrow j$) and S-Link ($m \rightarrow n$), respectively. The difference between tonemap vectors of P-Link and S-Link can then be denoted by $[D_{i\rightarrow j,m\rightarrow n}]_{1\times N} = [T_{i\rightarrow j}]^{p}_{1\times N} - [T_{m\rightarrow n}]^{s}_{1\times N}$, and vice versa.

Minimum Throughput Requirement. Let us denote the number of PLC nodes in a network to be E. Moreover, let us denote the minimum throughput requirement of e-th node as $\mathcal{T}_e$. Then we can represent the minimum number of bits to be modulated across a given set of OFDM subcarriers (tonemap), required to meet $\mathcal{T}_e$ as $\tau_e = \frac{\mathcal{T}_e}{T_s}$. Note that for any P-Link ($i \rightarrow j$) and S-Link ($m \rightarrow n$) pair, our SS strategy needs to meet throughput requirement of the P-Link only.

Allowed Tonemaps & Link Ranks. Each node e in the network will locally calculate an SS matrix using the proposed SS algorithm. The entries of the SS matrix consist of two entities, namely *allowed tonemaps* - $[ST, PT]$ (where $[ST]$ is a set of subcarriers allowed to be modulated on an S-Link while a P-Link operates, and it is vice versa for $[PT]$), and *link ranks* - r (i.e. a rank proportional to the SS gain of an S-Link when sharing spectrum with a P-Link). For each node e, there are $2\times(E-1)(E-2)$ possible P-Links which can operate in its vicinity. Moreover, for each of those possible P-Links, there are $2\times(E-2)(E-3)$ possible S-Links which can operate in parallel. Therefore, a locally computed SS matrix any node would be of size $4\times(E-1)(E-2)\times(E-2)(E-3)$, where each $2\times(E-2)\times(E-3)$ will correspond to the allowed tonemaps and ranks of all possible S-Links corresponding to one of the $2\times(E-1)(E-2)$ possible P-Links.

Spectrum Sharing Gain. We represent the gain $G_{m\rightarrow n}$ obtained by allowing an S-Link to operate with a P-Link as:

$$G_{m\rightarrow n} = [\sum_{[ST]}[T_{m\rightarrow n}] + \sum_{[PT]}[T_{i\rightarrow j}]] - \sum_{[1,N]}[T_{i\rightarrow j}] \tag{6.1}$$

6.5.2 Spectrum Sharing (SS) Algorithm

We design our SS algorithm to meet two key requirements: **(a)** *It must take into account minimum throughput requirements of the destination node of each possible P-Link in the network*, and **(b)** *It should involve minimal channel probing and control overhead.* As our SS approach is designed to work on top of existing user scheduling provided by current HPAV/AV2 CSMA/CA or TDMA procedures, therefore, SS is performed only when a P-Link is established and is already operating. Our SS algorithm runs locally at each node of the network, and therefore assumes that each node in the network has complete tonemap information about all other possible links in the network. Later in section 6.6, we explain how this can be achieved using current HPAV protocols, while incurring minimal channel probing and control overhead. Moreover, for simplicity of our discussion, we assume that all nodes in the network are in the same collision domain, and that there are no hidden nodes in the network (Request to Send (RTS) and Clear to Send (CTS) delimiters can be used by regular HPAV MAC to handle hidden nodes during CSMA/CA).

6.5.2.1 Optimal SS approach:

Our optimal SS approach is described in Algorithm 2. The algorithm runs on each node of the network separately, and computes $4 \times (E-1)(E-2) \times (E-2)(E-3)$ *allowed tonemaps* - $[ST]$'s for all S-Links, corresponding to all possible P-Links. To describe in words, for each P-Link and S-Link pair, the above algorithm first sorts the difference vector $D_{i\to j,m\to n}$, and starts assigning the subcarriers to P-Link corresponding to descending order of the entries $\widetilde{D}_{i\to j,m\to n}$, until its minimum throughput requirement is met. The remaining subcarriers are then assigned to the S-Link. Although, we did not observe such cases in our deployments, but in practice, a PLC network can may contain some extremely bad P-Links (modulation of all subcarriers is very low). Following the aforementioned algorithm, *bad P-Links would only share their spectrum once their own throughput requirements are met, and therefore, will not starve.* Next, we discuss how this SS approach can be optimized for overall network throughput and fairness in spectrum, respectively.

Optimizing overall network throughput. *Once minimum throughput requirement of a P-Link is*

Algorithm 2: Optimal algorithm for distributed spectrum sharing in HPAV PLC-Nets

1: /*Takes in a set of tonemaps for all possible links*/
2: **procedure** GETALLOWEDTONEMAPS_SS($[T]^E$)
3: $P \leftarrow all\ possible\ P-links$
4: $S \leftarrow all\ possible\ S-links$
5: **for** each $(i \rightarrow j) \in P$ **do**
6: **for** each $(m \rightarrow n) \in S$ **do**
7: $[ST] \leftarrow [1, N]$ ▷ Allowed indices for S-Link
8: $[PT] \leftarrow \emptyset$ ▷ Allowed indices for P-Link
9: $t_n \leftarrow 0$
10: $D_{i\rightarrow j,m\rightarrow n} \leftarrow [T_{i\rightarrow j}]^p_{1\times N} - [T_{m\rightarrow n}]^s_{1\times N}$
11: $\hat{I}, \hat{D}_{i\rightarrow j,m\rightarrow n} \leftarrow sort(D_{i\rightarrow j,m\rightarrow n},\ descend)$
12: ▷ $\hat{I}$ = indices corresponding to sorted entries
13: **while** $t_n < \tau_n$ **do**
14: $t_n \leftarrow t_n + T_{i\rightarrow j}(\hat{I}(1))$
15: $[ST] \leftarrow [ST] - \{\hat{I}(1)\}$ ▷ remove from set
16: $[PT] \leftarrow [PT] + \{\hat{I}(1)\}$ ▷ add to set
17: $\hat{I} \leftarrow \hat{I} - \{\hat{I}(1)\}$ ▷ remove from index set
18: **end while**
19: **end for**
20: **end for**
21: **end procedure**

met, the remaining subcarriers are assigned to both P-Link and S-Link such that the total number of modulated bits is maximized i.e. $\max(G_{m\rightarrow n})$. This requires a slight modification to Algorithm 2 (between steps 13-18), such that it will keep assigning subcarriers to P-Link in descending order of the entries in $D_{i\rightarrow j,m\rightarrow n}$, as long as the total number of bits modulated on both links increases.

Optimizing for overall spectrum fairness. *Once minimum throughput requirement of a P-Link is met, the remaining subcarriers are assigned to both P-Link and S-Link such that the ratio of number bits modulated along both links approaches* 1.0*, i.e.* $\frac{\sum_{[ST]}[T_{m\rightarrow n}]}{\sum_{[PT]}[T_{i\rightarrow j}]} \approx 1.0$. This also requires slight modifications to Algorithm 2 (between steps 13-18).

Complexity. The aforementioned SS approaches will require approximately $4\times(E-1)(E-2)(E-2)(E-3)(N*log(N)+N)$ computations at each PLC node. The overall complexity of the algorithm can be written as $O(E^4 * N * log(N))$.

6.5.2.2 Ranking of S-Links

While computing allowed tonemaps for each S-Link $m \rightarrow n$, Algorithm 2 also assigns a rank r_{mn} to that S-Link proportional to its SS gain ($r_{mn} = k \cdot G_{m \rightarrow n}$, where $k = 1$ in our work). We will show how we use these ranks while in the design of our proposed SS protocol for HPAV devices, later in Section 6.6.

6.6 Enabling Spectrum Sharing for HPAV PLCs

In this section, we show how our proposed SS strategy can be enable enabled in the MAC layer of current HPAV PLC devices while incurring minimum firmware level changes.

Channel probing and control overheads. As mentioned before, our SS algorithm works in a distributed manner and runs locally at each node, such that the network level SS decisions are eventually known to each node in the PLC network. Therefore, PLC nodes will not have to distribute their SS decisions to other nodes in the network. However, our SS algorithm requires tonemap information about all possible links in the network, which will incur communication overhead. The communication overhead will be on the order of $O(E^2)$, as each node in a PLC network will broadcast tonemap information for its $(E - 1)$ possible links with other nodes in the network. However, due to the pseudo-stationary nature of PLC links (6.4.3), this probing overhead will be minimal and will not interfere with regular data transmissions.

A channel probing interval t_{probe} for SS can be set by the CCo of a PLC network. The CCo can then periodically command all nodes the network to log tonemaps of all possible links and formulate their SS decisions. CCo can use control-related messaging schemes already built into HPAV MAC (e.g. Management Messages (MMEs)) for this purpose [3, 5], and t_{probe} can be chosen such that the exchange of control messages incurs minimal overhead and interference to data transmissions. The probing frequency (i.e. $1/\tau_{probe}$) must be kept within a certain threshold in case CCo observes some very dynamic PLC links in its network, because otherwise spectrum sharing may lead to loss of overall network throughput due to high channel probing overhead. CCo can also completely stop spectrum sharing throughout its network and fall back to default HPAV MAC if the channel

conditions of PLC links in its network are not conducive to SS. *Note that SS will not be performed during the exchange of control messages.*

Periodic Re-evaluation of Full Spectrum: All nodes in the network will periodically disable SS and transmit across full spectrum following default HPAV MAC. No S-Link will operate in this case. The frequency of this periodic behavior can be chosen by CCo of the network, based on temporal dynamics 6.4.3 of PLC links its network. Such periodic use of the whole spectrum will allow each node to automatically update its full spectrum tonemaps towards other nodes in the network, during regular data transmissions. The network CCo will then re-evaluate the SS decisions in its network by accessing these tonemaps as described before.

Medium Access during SS: Current HPAV MAC is centrally controlled through Beacon signals from CCo. The Beacon signals broadcast by CCo to establish Beacon Periods (BPs) with TDMA and CSMA slots are robust and reliable (Beacons and several other control-related messages operate over ROBust mOdulation (ROBO) modes [72]). Next, we explain how the medium access will work during TDMA and CSMA slots in an SS enabled HPAV MAC.

TDMA: Whenever a P-Link is scheduled to send traffic in a TDMA slot, the highest ranked S-Link (according to the SS algorithm) corresponding to that P-Link will be scheduled to operate in the same slot. In case some of the S-Links corresponding to that P-Link do not have any traffic to send, the highest ranked S-Link will only be chosen from among the S-Links which are waiting in line to send traffic. Therefore, the allowed tonemaps for both P-Link (i.e. $[PT]$) and the selected S-Link (i.e. $[ST]$) will be chosen accordingly.

CSMA/CA: In case of TDMA, the selection of allowed tonemaps $[PT]$ and $[ST]$ is straight forward, since the P-Link and S-Link connections can be specifically scheduled by the CCo to operate in the same slot. However, two major issues arise in case of CSMA: (a) *How will a P-Link know which of the possible S-Links have traffic to send, so it can select its* $[PT]$ *accordingly?*, and (b) *Assuming issue (a) is resolved, how will the S-Link know that a P-Link is established so that it can select its* $[ST]$ *according to* $[PT]$*?* In following steps, we discuss how medium access and the consequent link interactions will differ from the regular HPAV CSMA/CA.

(i) Before broadcasting Beacon signals, the CCo identifies all links with pending traffic, and then shares that information with each node in its network through HPAV control-related messaging. *This resolves issue (a).*

(ii) Once a P-Link gets medium access, the remaining nodes go into their backoff stages, following the regular CSMA/CA procedure. Afterwards, the source node of the P-Link enables the *Multicast Flag* (MCF) in the *Start-of-Frame Control* (SOF) field of its *MAC Protocol Data Unit* (MPDU) [72] while establishing its connection with the destination node, so that all remaining nodes in the network can extract the source and destination IDs of the P-Link from this SOF delimiter field. *This resolves issue (b)* later in step(iv).

(iii) Both source and destination nodes of the P-Link select [PT] corresponding to the highest ranked S-Link among the S-Links with pending traffic (given the information received in step (i)), and use the unshared subcarriers for transmission/reception, while disabling the shared ones.

(iv) After knowing the P-Link information from SOF delimiter in P-Link's broadcast MPDU frame, the source and destination nodes of the highest ranked S-Link with pending traffic enable [ST] and disable [PT]. The S-Link then operates in parallel with the P-link.

(v) Nodes belonging to any active P-Link come out of their SS state (i.e. re-enable all subcarriers and enter again into contention for whole spectrum) when the transmission between them is finished. As soon as P-Link's transmission ends, the S-Link finishes its transmission as well, and comes out of spectrum sharing state (as the transmission frame ends).

Disabling modulation of subcarriers: Thankfully, it is easy to disable the subcarriers in HPAV devices. In current HPAV PHY, each subcarrier is independently modulated based on channel characteristics between transmitter and receiver (i.e. *bit-loading)*. HPAV PHY allows dynamic *notching* of specific subcarriers by turning them off, which can be achieved by making soft changes to device's tone mask (enabled subcarriers) [3, 72]. However, currently, this functionality needs proprietary access to firmware supplied by vendors. Up to 30 dB deep notches are possible in HPAV, and typically 4 additional subcarriers on each side of a notch can be turned off to achieve a 30 dB notch depth, resulting in about 200 KHz of guard-band overhead for each notch.

6.7 Implementation and Evaluation

We evaluate our proposed SS strategy through trace-driven simulations using traces we obtained from multiple PLC network deployments in our enterprise. We implement and evaluate our proposed SS strategies on top of HPAV CSMA protocol only. We perform trace driven simulations using tonemap traces collected from seven different 4-node PLC deployments (Figure 6.6 represents Deployment#1). Our simulations do not take into account frame aggregation procedures, bit loading of ethernet frames inside PLC frames, management messages and channel errors, since these parameters are proprietary vendor-specific implementation information. In our simulations, we choose collision duration $\tau_c = 2920.64\mu s$, duration of successful transmission $\tau_s = 2542.64\mu s$ and frame length $F_l = 2050$ [140, 141]. The contention window (CW) and deferral counter (DC) values used for each HPAV CSMA/CA back off stage are [8, 16, 32, 64] and [0, 1, 3, 15], respectively. We assume that there are no hidden terminals, and transmission failures are only due to collisions.

6.7.1 Evaluation Metrics

We use following metrics to evaluate the performance of our proposed SS approaches:

Throughput: We calculate the normalized throughput Thr for each link $m \rightarrow n$ in our simulation as follows:

$$Thr = 100 \cdot \frac{[\sum_{i=1}^{[\#SuccessTransmissions]} S_{F_i}] \cdot [Frame\ length]}{Total\ simulation\ time}$$

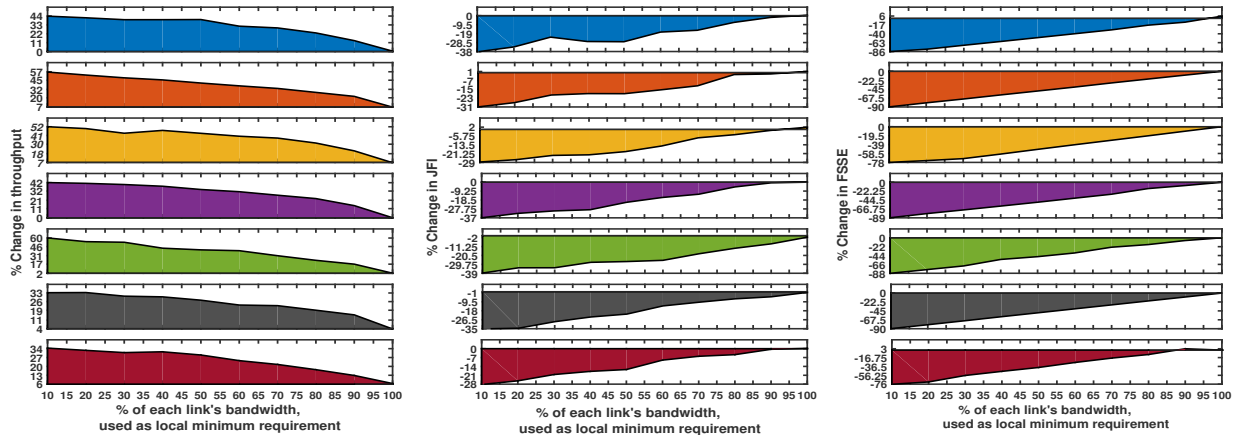
S_{F_i} represents the fraction of spectrum utilized at i-th transmission. $S_{F_i} = \sum_{j=1}^{N} [T_{m \rightarrow n}]/9170$, such that $\max(S_{F_i}) = 1$ and $\min(S_{F_i}) = 0$.

Fairness: We evaluate the fairness of different SS strategies by calculating *Jain's fairness index* (JFI) [58] and *Fairly Shared Spectrum Efficiency* (FSSE) [39]. An allocation strategy is maximally fair if all nodes in a PLC-Net allocate the same throughput, in which case JFI = 1. On the other hand, FSSE of a PLC-Net gives the *spectrum efficiency* (SE) of the PLC node with minimum throughput in the network. In case of maximum spectrum fairness, FSSE is equal to the SE of the

whole network. For a PLC network, we define its SE to be its average throughput, and its FSSE to be its minimum throughput.

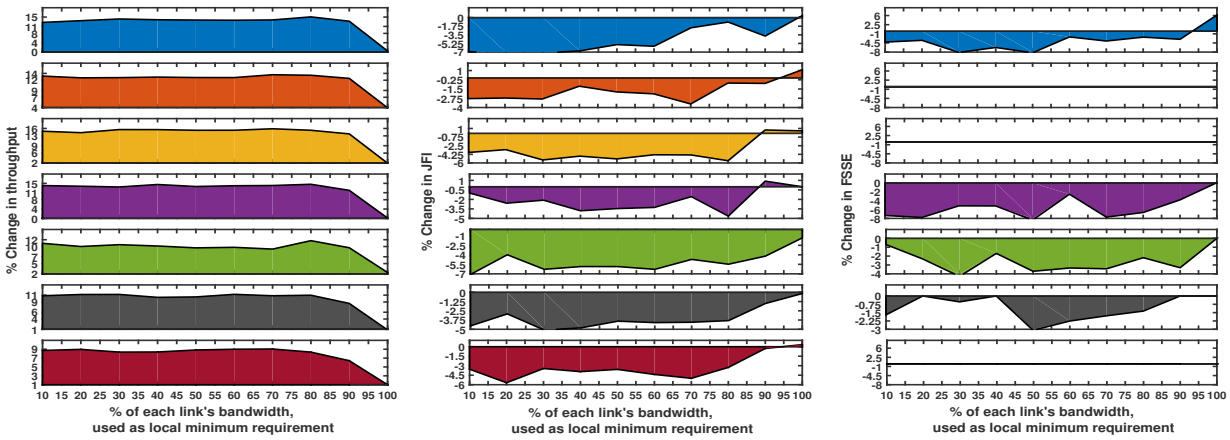
Next, we show how our optimal SS strategies can achieve higher overall network throughput (when optimized for maximum throughput), and maintain higher fairness (when optimized for fairness), while meeting per-link minimum bandwidth requirements. We test following two scenarios:

Per-link (Local) Minimum $\mathcal{T}_e$: In this scenario, each P-Link uses a percentage of it's available bandwidth as it's minimum $\mathcal{T}_e$. Figures 6.7(a)-6.7(c) show how net throughput, JFI and FSSE of the seven deployments change as $\mathcal{T}_e$ increases, when SS is optimized for maximum net throughput. The X-axis starts from 10%, i.e. when $\mathcal{T}_e$ is 10% of the available bandwidth. We can observe net throughput gains of up to 60%, and per-link throughput gains as high as 250% (Fig. 6.9(a)). However, it comes at the expense of large decrements in overall fairness. Figures 6.8(a)-6.8(c) show results for scenario when SS is optimized to maintain fairness in the network. In this case, we observe net throughput gains of up to 14% and per-link throughput gains as high as 110% (Fig. 6.9(b)), while incurring much lower decrease in overall fairness, leading to 30% and 87% better JFI and FSSE values (JFI and FSSE of some deployments improve up to 1% and 6%, respectively, for some values of $\mathcal{T}_e$'s)).



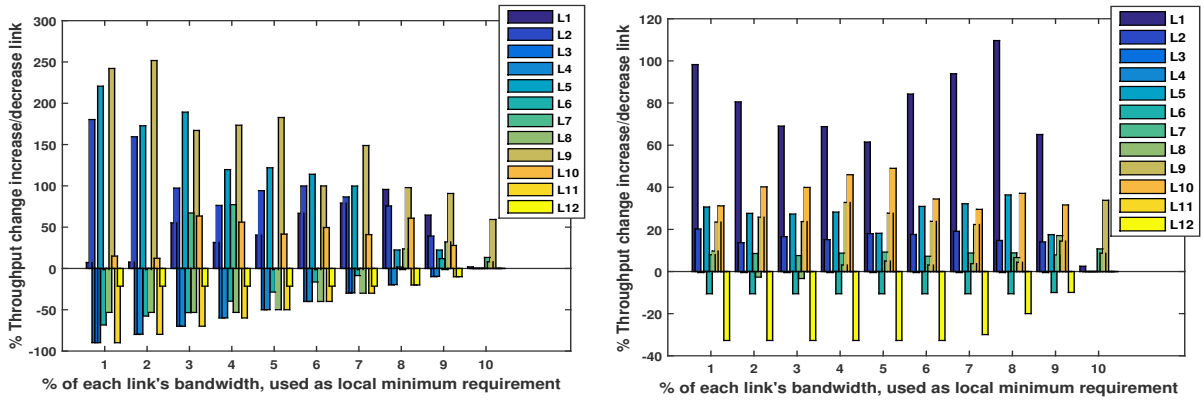
(a) Network throughput variation. (b) Jain's Fairness Index variation. (c) FSSE variation.

Figure 6.7: Testing scenario with per-link (local) minimum $\mathcal{T}_e$, optimizing for net throughput (#1-#7, top-bottom)



(a) Network throughput variation. (b) Jain's Fairness Index variation. (c) FSSE variation.

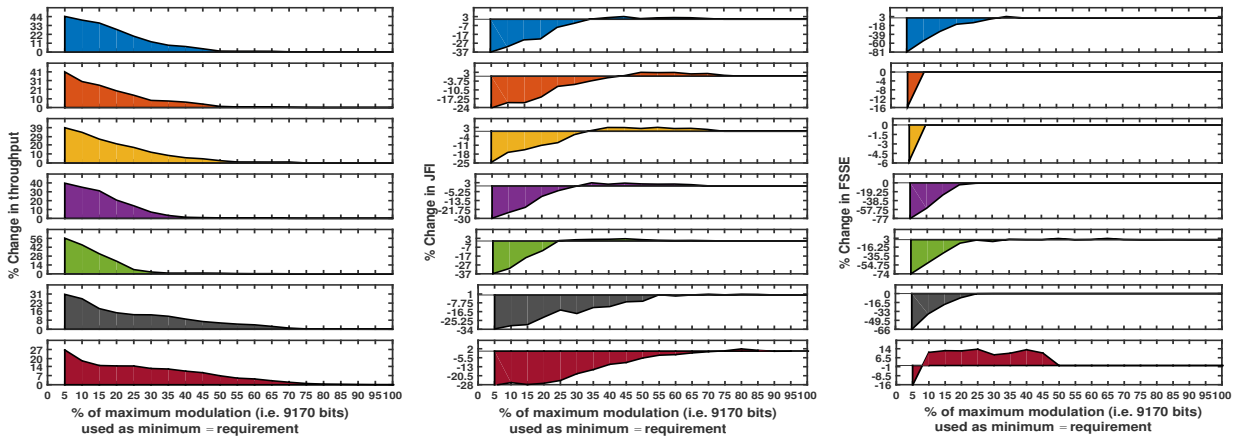
Figure 6.8: Testing scenario with per-link (local) minimum $\mathcal{T}_e$, optimizing overall fairness (#1-#7, top-bottom)



(a) Overall network throughput optimization policy. (b) Fairness optimization policy.

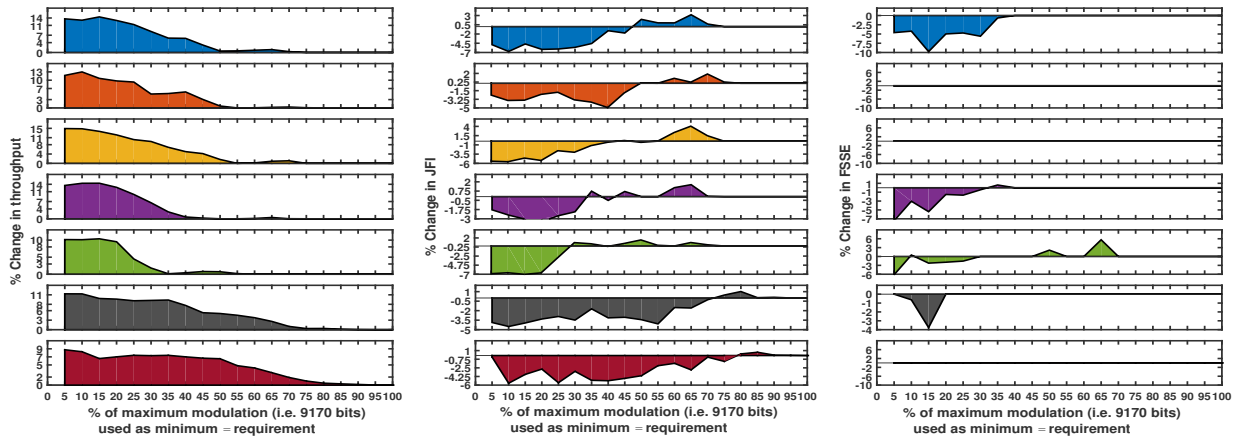
Figure 6.9: Per-link throughput changes for Deployment#1 (testing scenario with per-link (local) minimum $\mathcal{T}_e$ requirement)

Network-wide Minimum $\mathcal{T}_e$: In this scenario, a percentage of maximum number of bits which can be modulated over any link (i.e. $10 \cdot N = 9170\ bits$) is used as minimum $\mathcal{T}_e$ requirement for each P-Link. Such a scenario can arise when a PLC network is required to meet bandwidth requirements of a certain type of application. Figures 6.10(a)-6.10(c) show how net throughput, JFI and FSSE of the seven deployments change as $\mathcal{T}_e$ increases, when SS is optimized for maximum net throughput. We can observe net throughput gains of up to 56%, and per-link throughput gains as high as 180% (Fig. 6.12(a)), which in most cases comes at the expense of large decrement in fairness performance (except for Deployment#7 (Fig. 6.10(c) whose FSSE improves up to 14% for some $\mathcal{T}_e$'s)). Figures



(a) Network throughput variation. (b) Jain's Fairness Index variation. (c) FSSE variation.

Figure 6.10: Testing scenario with network-wide minimum $\mathcal{T}_e$ requirement, optimizing for net throughput (#1-#7, top-bottom)

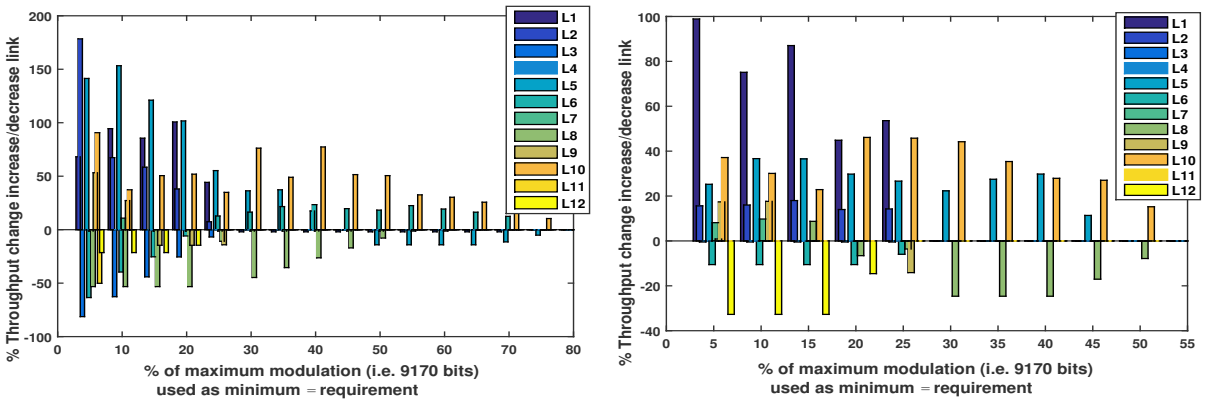


(a) Network throughput variation. (b) Jain's Fairness Index variation. (c) FSSE variation.

Figure 6.11: Testing scenario with network-wide minimum $\mathcal{T}_e$ requirement, optimizing for overall fairness (#1-#7, top-bottom)

6.11(a)-6.11(c) show results for scenario when SS is optimized to maintain fairness in the network. In this case, we observe net throughput gains of up to 15% and per-link throughput gains as high as 100% (Fig. 6.12(b)), while incurring much lower decrease in fairness performance, leading to 25% and 60% better JFI and FSSE values (JFI and FSSE of some deployments improve up to 4% for some values of $\mathcal{T}_e$'s)).

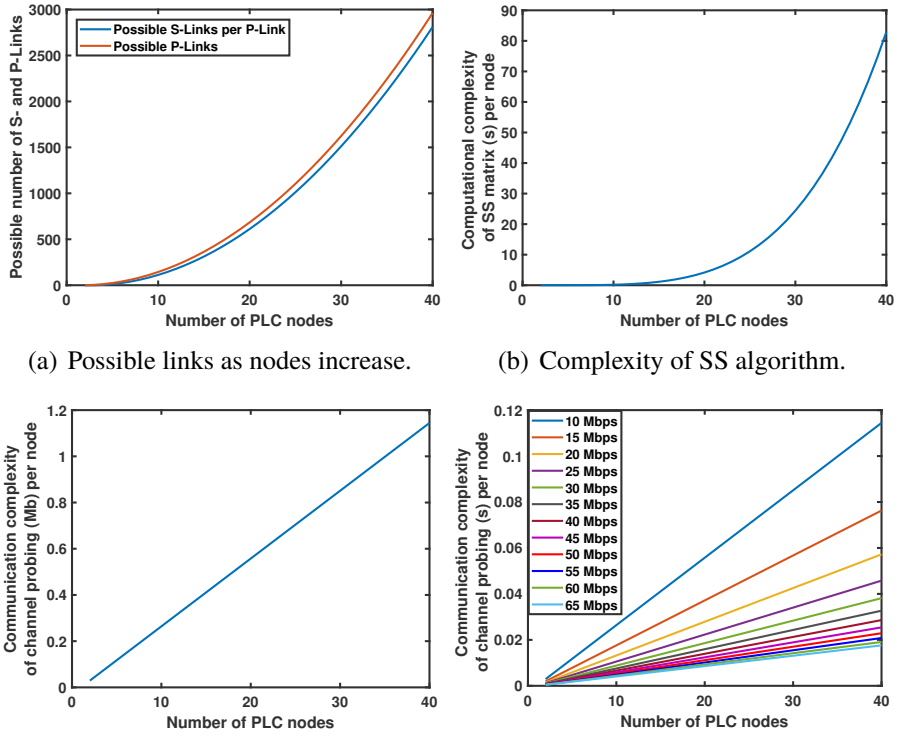
Effect of the changes in PLC channels: In real world scenarios, PLC channels can change due to interference caused by electrical appliances or power distribution components, as discussed in



(a) Overall network throughput optimization policy.

(b) Fairness optimization policy.

Figure 6.12: Per-link throughput changes for Deployment#1 (testing scenario with network-wide minimum $\mathcal{T}_e$ requirement)



(a) Possible links as nodes increase.

(b) Complexity of SS algorithm.

(c) Communication complexity (Mbs).

(d) Communication complexity (secs).

Figure 6.13: Computational and communication complexity of our spectrum sharing approach as number of PLC nodes increase

§6.4. Therefore, two communicating HPAV PLC devices must continuously exchange and update the tonemaps between them to update the modulation schemes for each subcarrier. This is the default behavior of all HPAV based PLC devices, where each device follows a tonemaps updating

protocol (that runs over one of the ROBust mOdulation (ROBO) modes [72]) to update tonemaps while communicating with other nodes in a network. Without regular tonemap updates, PLC links risk high packet loss, specially when the channel between two nodes changes significantly.

Similarly, when SS is enabled, nodes of any S- or P-Link update their tonemaps by simply following the regular HPAV protocol. However, to fully utilize SS, all PLC nodes in the network must share their latest tonemaps with other nodes in the network and then recompute the local SS matrices. However, this incurs both computational and communication related overheads, as discussed in §6.5 and §6.6. Figs. 6.13(a) - 6.13(d) show how both overheads change as the number of PLC nodes in a network increase. We observe that both overheads increase as the number of nodes increase, where the computational complexity of SS algorithm specifically increases significantly. Therefore, spectrum sharing will only be beneficial when most PLC channels in a network are pseudo-stationary, as this will significantly reduce the frequency of channel probing and recalculation of the local SS matrices.

To understand how frequent changes in PLC channels in a network can impact the spectrum sharing gains, we run simulations where we manually add noise to real tonemap data obtained from 3 different deployments. We introduce and evaluate the impact of two parameters in our simulations: (1) *probability* (range 0.05 through 1) that the channel will change during a specific communication instance between two nodes, and (2) *standard deviation* (STD) (range 1 through 3) of the variations introduced in subcarriers (this values stays same for all subcarriers during a specific simulation) whenever the channel changes. In these simulations, we implemented the "net throughput optimization" SS strategy mentioned earlier in this section, where we used a network-wide minimum throughput requirement of T_e = 25% of the maximum possible modulation (*i.e.* 9170/4 = 2292.5 bits). In our simulations, we measure the normalized throughput and the percentage change in throughput correspond to 4 different cases: (1) channels do not change at all, (2) channels change, tonemaps updated between communicating nodes, tonemaps shared between all nodes and subcarriers optimally reassigned based on SS algorithm, (3) channels change, tonemaps updated between communicating nodes, but the subcarriers are not reassigned (*i.e.* assignment stays intact),

and (4) channels change but tonemaps are not updated between the communicating nodes. Figs. 6.14 - 6.20 show the results corresponding to all of the 3 real-world deployments. We observe that throughput deteriorates significantly in case 4, as not updating tonemaps leads to significant number of unsuccessful transmissions. We discuss case 4 for the sake of completeness, as it never

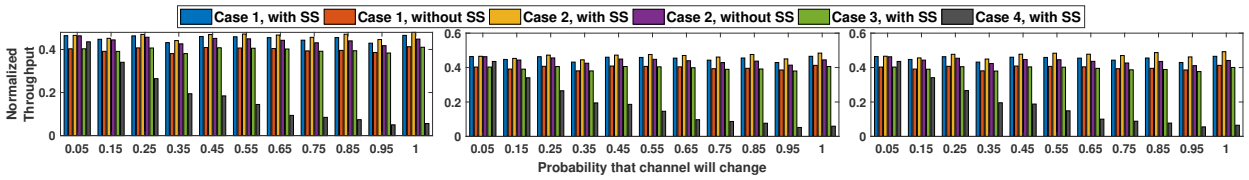


Figure 6.14: Normalized throughputs observed for different cases in deployment #1 (STD of noise 1, 2 and 3 from left-right)

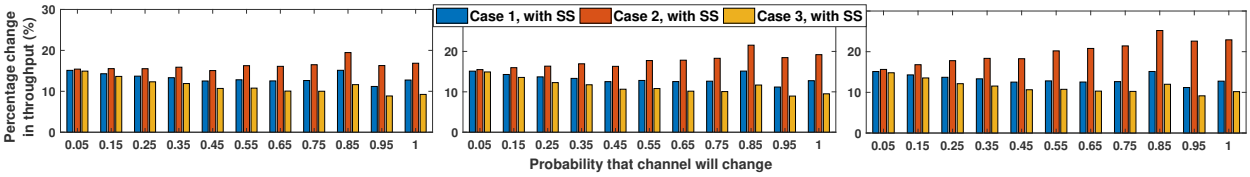


Figure 6.15: Percentage change in throughput for different cases in deployment #1 (STD of noise 1, 2 and 3 from left-right)

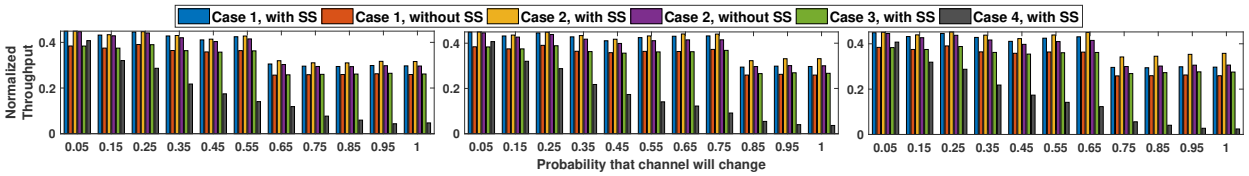


Figure 6.16: Normalized throughputs observed for different cases in deployment #2 (STD of noise 1, 2 and 3 from left-right)

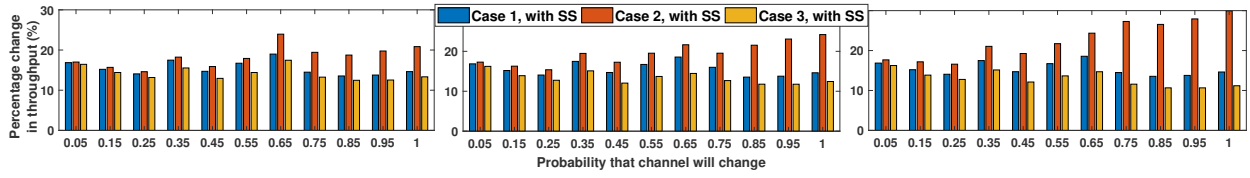


Figure 6.17: Percentage change in throughput for different cases in deployment #2 (STD of noise 1, 2 and 3 from left-right)

occurs either in regular or SS enabled HPAV protocol. Interestingly, we observe that case 3 exhibits significant SS gains even though the latest tonemaps were not shared between all nodes and the subcarriers were not optimally reassigned. However, the gains in case 3 are less than the optimal

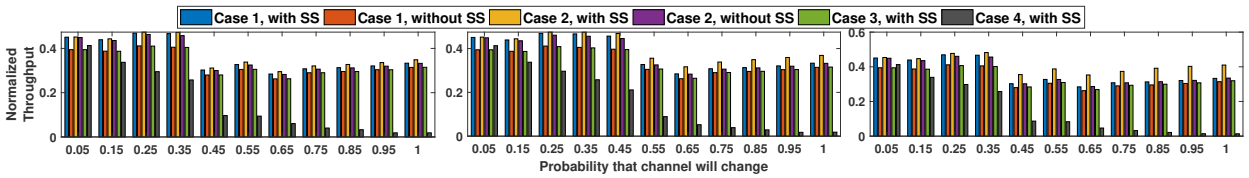


Figure 6.18: Normalized throughputs observed for different cases in deployment #3 (STD of noise 1, 2 and 3 from left-right)

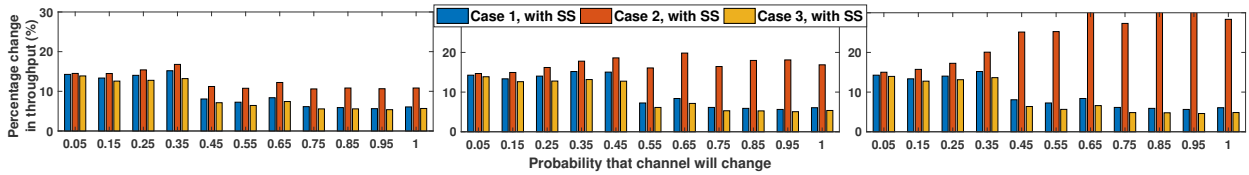
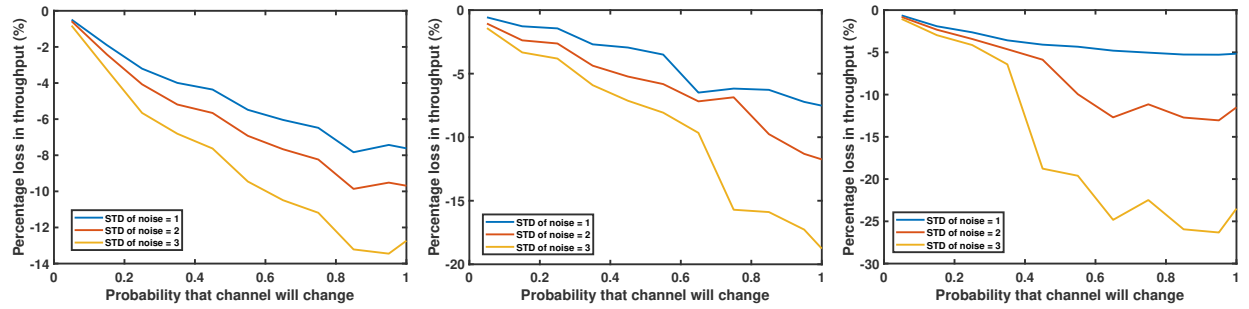


Figure 6.19: Percentage change in throughput for different cases in deployment #3 (STD of noise 1, 2 and 3 from left-right)



(a) Deployment #1. (b) Deployment #2. (c) Deployment #3.

Figure 6.20: Percentage loss in throughput of case 3 compared to case 2 as the probability of change in PLC channels increases

gains in case 2, and they decrease as the probability of channel change and/or STD of variations increases, as shown by results in Figs 6.20(a) - 6.20(c).

6.8 Advantages of Multi-Hop Routing in PLCs

HomePlug PLC devices currently do not support multi-hop communication [3, 5]. However, direct link communication in PLC networks can often be impossible or show very low throughput, either due to highly location dependent multipath attenuations and/or interference from appliances. Next, we explore if multi-hop routing can improve throughput and connectivity in large building settings, such as enterprises, through real world experiments.

For our evaluation, we use the optimized link state routing protocol (OLSR) [29], which is a table-driven proactive link-state routing protocol and has been widely used in 802.11 wireless

mesh networks. For our testbed experiments, we first port the open-source OLSR and ETT implementations [1, 2] in our OpenWrt boards. Then, we deploy 9 PLC nodes in various topologies in our floorplan (Fig. 6.3) and then evaluate routing performance of the PLC-Net. Our results show that routing can significantly improve PLC-Net performance in scenarios where certain PLC links perform very poorly. We identified such a scenario during the communication between PLC nodes N9 and N6, which were located at different breakers but in the same distribution line (cf. Fig. 6.3). Figure 6.21, shows the UDP throughput performance between PLC node N9 and N6 for one minute window, while OLSR is enabled and disabled. When OLSR is turned on, UDP throughput between N9-N6 and N6-N9 is 5.6 and 4.5 times higher, respectively, as compared to the case when OLSR is off. We observe that such communication is affected by electrical devices (lamps, phone chargers, monitors) between N9 and N6, which interfere with the PLC network. When OLSR is enabled, N9 and N6 communicate through node N7 or N8, avoiding such interferences. The throughput temporal variations shown in Fig. 6.21 are attributed to the interference dynamics, which make OLSR to change routes periodically. We make the same observations for TCP traffic (Table 6.1). When OLSR is on, TCP throughput is up to 3.6× higher compared to the case when OLSR is off.

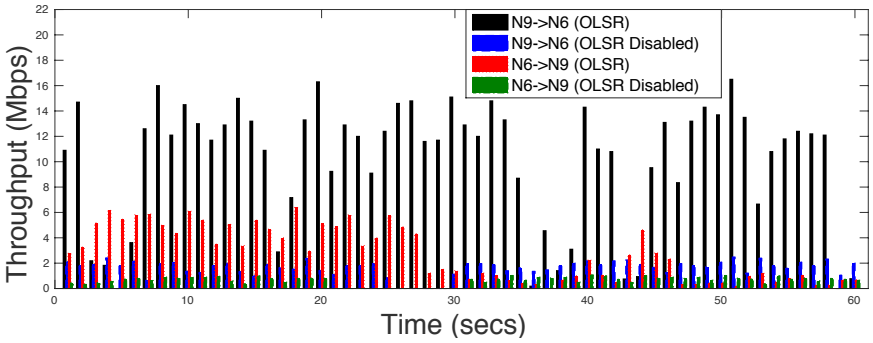


Figure 6.21: Throughput with OLSR on/off for 60 secs

Conclusion: *Mesh routing can significantly boost PLC-Net performance in scenarios where direct PLC links perform very poorly and multi-hop communication is required.* We believe that combining multi-hop routing with fine grained spectrum sharing can potentially improve PLC network performance even further, especially in scenarios where direct PLC links perform poorly. We will pursue this direction in future.

Table 6.1: UDP and single-flow TCP throughput and jitter with OLSR on/off (jitter is reported by iperf only for UDP traffic)

Traffic	Flow	Thr(Mbps)	Jitter(ms)
UDP	N9→N6 (olsr on)	9.5	5.9
	N9→N6 (olsr off)	1.7	17.8
	N6→N9 (olsr on)	2.7	11.6
	N6→N9 (olsr off)	0.6	18.7
TCP	N9→N6 (olsr on)	4.2	-
	N9→N6 (olsr off)	1.4	-
	N6→N9 (olsr on)	1.8	-
	N6→N9 (olsr off)	0.5	-

6.9 Conclusions

PLC technology has the potential to improve connectivity and allow for large-scale sensing, control and automation applications at low cost, without any need for dedicated network cabling. In this work, we make following contributions. First, we conduct an extensive measurement study of PLCs in a real enterprise environment using COTS HPAV PLC devices, based on which we conclude that spectrum sharing (not supported by existing PLC standards) can significantly benefit enterprise level PLC mesh networks. Second, we propose, implement and evaluate a spectrum sharing scheme, and show that fine-grained distributed spectrum sharing can significantly boost the aggregated and per-link throughput performance by up to 60% and 250% respectively, by allowing multiple PLC links to communicate concurrently, while requiring a few modifications to the existing HPAV devices and protocols. We believe that combining multi-hop routing with fine grained spectrum sharing can potentially improve PLC network performance even further, especially in scenarios where direct PLC links perform poorly. This is can be a possible future research direction.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

In this chapter, I provide an overview of some future research directions I have been exploring during my PhD and conclude my dissertation.

7.1 Future Work

7.1.1 WiFi Signals Based Typing Biometrics

7.1.1.1 Motivation

Typing behavior based biometrics provide an extra level of security on top of traditional passphrase and PINs based authentication schemes, which are often vulnerable to shoulder surfing, keylogging malwares and video based attacks [89, 90, 115, 116, 163, 167]. All such schemes are based on the hypothesis that every user has a unique and consistent typing behavior, and that it is difficult for an adversary to reproduce that behavior. Existing typing biometric schemes have used modalities such as time delays between key presses or video of hands as the user types, to capture the uniqueness in typing behavior. This work concerns the problem of developing a new method for getting typing biometrics, which captures the uniqueness of a user's typing behavior by leveraging the changes caused in CSI signals due to motion of the user's hands and fingers during typing.

7.1.1.2 Challenges

The key technical challenge is to extract CSI based typing behavior features such that they are robust to changes in position and orientation of a user's laptop, as well as, resilient to static changes in the environment (e.g. changes in arrangement of room furniture). In our preliminary study, we collected over 6000 samples corresponding to 8 different words from 10 volunteers. For user identification, our current scheme is able to achieve accuracies of up to 90% with passwords of length as small as 9 letters. For user authentication, our current scheme can achieve Equal Error Rates (EER) of less than 15%. However, these accuracies are not enough for authentication

and identification purposes. We have observed that longer and complex passwords lead to higher authentication and identification accuracies. The increased accuracy in case of longer and complex words can be attributed to the fact that those words are relatively difficult and longer to type than simple to type words, which makes each user's typing behavior more distinguishable from the other users. We believe that typing longer passwords or sentences can significantly improve the uniqueness of CSI based biometrics. Results from our preliminary study have shown that identification accuracies of up to 97.5% and EERs of less than 5% can be achieved for identification and authentication purposes, respectively, where users typed a known sequence of 8 words in form of a sentence.

7.1.2 Effective Fusion of Orthogonal Components in WiFi Signals Subspace For Improved Activity Recognition

7.1.2.1 Motivation

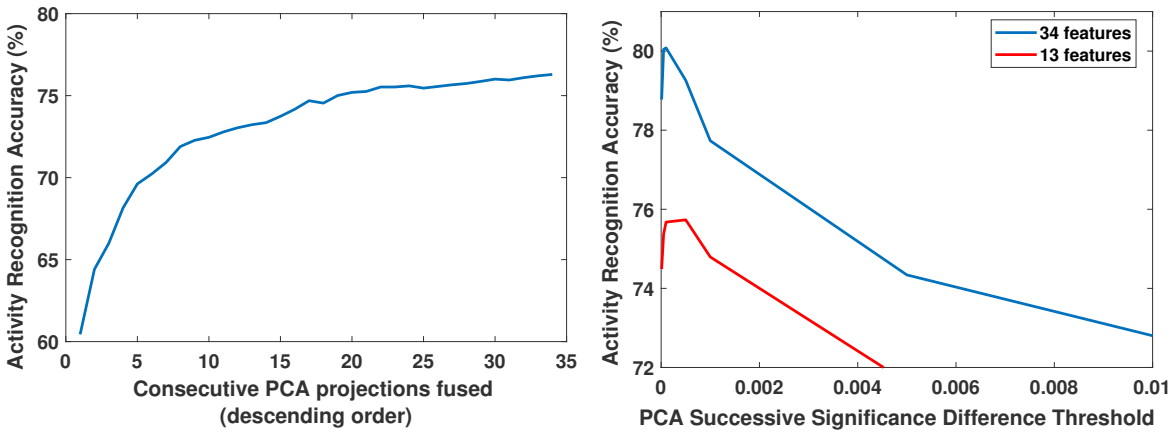
Many WiFi signals based human activity recognition schemes have been proposed in the last 5-6 years. However, a major limitation of all existing schemes is that they are highly environment (*e.g.* the position of WiFi transceivers, location of the individual, etc.) and individual dependent, and therefore, do not work in different environments and with different individuals. Some recent schemes have tried to address this issue [60, 139, 147]. Wang *et al.* have proposed a CSI-speed model, which quantifies the correlation between the CSI value dynamics and human movement speeds, and a CSI-activity model, which quantifies the correlation between the movement speeds of different human body parts and a specific human activity. Based on these two models, they quantitatively build the correlation between CSI value dynamics and a specific human activity. However, their models are only able to recognize macro-level daily human activities (*e.g.* walking, running and opening a door/fridge) and not finer-grained gestures (*e.g.* flick, push and pull). Virmani *et al.* propose a novel translation function, which automatically generates virtual samples for all gestures in all possible configurations using the training samples provided for all gestures in only one configuration. Using the virtual samples corresponding to each specific configuration, they create machine learning

models for every configuration. To recognize gestures of a user at runtime, as soon as the user performs a gesture, their scheme first automatically estimates the configuration of the user and then evaluates the gesture against the classification model corresponding to that estimated configuration. However, their technique requires fine-grained smartphone based calibration of every gesture in the learning configuration, which is highly dependent not only on the individual but also the position of WiFi transceivers, thus rendering their scheme difficult to realize. Jiang *et al.* have proposed a deep-learning based activity recognition framework that, according to their claim, can remove the environment and subject specific information contained in the activity data and can extract environment/subject-independent features shared by the data collected on different individuals under different environments. Their scheme converts the CSI data obtained from different TX-RX antenna streams to "images" and then directly (blindly) feeds that multi-dimensional time-series data into the deep-learning network. However, there is a major issue in their pipeline. We have empirically observed that the effect of the same human activity on a specific WiFi subcarrier and TX-RX antenna stream is highly location and individual dependent. Basically, even if the activity's effect on the overall WiFi subspace is similar, the effect on individual CSI streams can be very different. Talking in the terms of computer vision, the way an activity affects WiFi signals in different environments is like a cat which looks like a normal cat in one environment, but has its eyes replaced by its toes and vice versa. Most deep learning frameworks will not be able to identify such a cat. One major reason behind why Jiang *et al.* are able to achieve a reasonable (up to 75%) activity recognition accuracy is because they place the router and WiFi receiver in almost the same manner for every different location and individual, which leads to similar variations in each specific CSI stream/dimension. Our work concerns the problem of developing a significantly improved WiFi based human activity and gesture recognition scheme that can effectively fuse the activity related information present in the signal subspace formed by different WiFi TX-RX streams, to get more resilient activity specific features.

7.1.3 Challenges

The key technical challenge here is to extract CSI based features of different human activities that are environment and individual independent. Our hypothesis that an activity performed by different individuals in different locations affects the WiFi subspace similarly. If we can effective combine/fuse the information in different orthogonal components in the WiFi subspace, we can improve the activity recognition accuracies. To achieve this, we have tried different techniques to fuse the information from different orthogonal components present in the WiFi subspace formed the CSI signals corresponding to different activities/gestures. By orthogonal components of WiFi subspace formed the CSI signals corresponding to different activities, we mean the orthogonal components obtained after applying Principal Component Analysis (PCA) on a specific multi-dimensional CSI sample corresponding to that activity/gesture. In our previous work 2, we only kept top few PCA projections of the CSI data for building classifiers and discarded the rest, which we think is not a good idea when the aim is to develop a generic activity/gesture recognition scheme, as we believe that rest of the PCA projections also contain important activity/gesture related information. However, we cannot simply add those projections to our feature set either, as they may contain noise or unrelated variations. Moreover, as we discussed before, the PCA projections can change positions which depends on the location and individual performing the activity (referring to the cat's eyes and toes example in 7.1.2.1).

Recently, we have tried combining consecutive PCA projections by first *max-min normalizing* each projection and then taking the magnitude to obtain a single timeseries/waveform, just like we take magnitude of readings obtained from different dimensions of an accelerometer. Our hypothesis behind this step was that activity recognition accuracies should increase as we fuse information from successive PCA projections (in descending order of variance). To test this, we collected over 13000 samples corresponding to several different activities (namely 'NoPresence','Presence', 'Sitting', 'Standing', 'Walking', and 'Waving') from over 70 different users in 11 different environments to build a database of diverse samples. Using this data, we trained an SVM classifier using 13 different time-domain features (details not mentioned here). Figure 7.1(a) shows the 10-fold cross-validation



(a) Effect of fusing information from more PCA projections on recognition accuracy (10-fold cross-validation)

(b) Effect of Successive Significance Difference (SSD) threshold on accuracy (10-fold cross-validation)

Figure 7.1: Effect of fusing information from successive PCA projections on recognition accuracy (10-fold cross-validation)

accuracies obtained from our initial experiments. Based on this result, we claim that our hypothesis is true, and successive lower PCA components in the WiFi subspace also contain important activity related information. However, this fusion mechanism is ad-hoc and makes it hard to know after which PCA projection we should stop fusing information. Moreover, the number of such PCA projections which contain important activity related information can be different even for the same activity performed by the same individual at the same location. So there is a need for automatic selection of important PCA components in every data sample. To deal with this issue, we have come up with a *Successive Significance Difference* (SSD) threshold, based on which we discard all the successive unimportant PCA components. We compute SSD of the PCA projections of data sample by first *max-min normalizing* the PCA coefficients corresponding to those projections, and then taking their first order difference (*i.e.* subtracting normalized coefficients of each successive PCA component from its previous one). Any successive PCA projection after the first PCA projection that does not meet the SSD threshold requirement is automatically discarded. Figure 7.1(b) shows how the 10-fold cross-validation accuracies vary with SSD threshold. For this experiment, we trained an SVM classifier using 13 and 34 different time-domain features (details not mentioned here), respectively. We can observe that accuracy in both scenarios reaches an approximately optimal point, after which fusing information from any extra PCA projections leads to significantly lower

accuracies due to addition of noisy PCA projections. Note that some of the time-domain features (*e.g. mean*, which signifies the average magnitude of variations in the CSI signals) we use with SVM classifiers are obtained after directly taking magnitude of the selected PCA projections, without max-min normalizing the timeseries of the individual PCA projections. Currently, we working on further improving our fusion scheme and developing a deep learning based framework which maybe able to learn more representative features of different activities by taking both timeseries shape and frequency domain features (*e.g.* FFTs) of the waveforms obtained after fusion.

7.2 Conclusions

In my dissertation, I have shown that several mainstream commodity off-the-shelf (COTS) electronic devices of daily use can be leveraged to develop interesting new IoT sensing and connectivity solutions. In my research, I revisited the physical-layer of various everyday COTS electronic devices such as WiFi, RFID, Smartphone (vibration mechanism), and Powerline Communication (PLC) devices, either to leverage the signals obtained from their physical layers to develop novel sensing applications, or to improve/modify their protocols to enable more useful deployment scenarios and networking applications - which they are not originally designed for - by introducing mere software/firmware level changes and completely avoiding any hardware level changes. Adding such new usefulness and functionalities on top of existing everyday infrastructure and electronics has advantages both in terms of cost and convenience of use/deployment, as those devices (and their protocols) are already mainstream, easily available, and often already purchased and in use/deployed to serve their mainstream purpose of use. In my works on WiFi signals based sensing, I developed signal processing and machine learning approaches to enable fine-grained gesture recognition and sleep monitoring using COTS WiFi devices. In my work on RFID signals based sensing, I developed signal processing and machine learning approaches to effectively image customers' browsing behavior in front of display items in places such as retail stores using COTS RFID devices. In my work on smartphone's vibration based sensing, I developed a robust and practical vibration based sensing scheme that works with COTS smartphones with different hardware,

can extract fine-grained vibration signatures of different surfaces, and is robust to environmental noise and hardware based irregularities. This work finds its applications in symbolic localization based context aware services, for example, training a smartphone to turn off lights when a user puts it on their bed-side table. And finally, as communication and sensing go hand in hand in the world of IoTs, I worked on PLCs (*i.e.*, communication mechanisms that leverage existing power distribution network/infrastructure inside a building for communication), where I developed a distributed spectrum sharing scheme to make enterprise level PLCs based IoT networks faster. This work is a major step towards using existing COTS PLC devices to connect different types of IoT devices for sensing and control related applications in large campuses such as enterprises.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Olsr routing protocol. In *http://www.olsr.org*.

[2] Olsr with link cost extensions. In *http://sourceforge.net/projects/olsr-lc/*.

[3] Homeplug av whitepaper. *http://www.homeplug.org/tech-resources/resources/*, 2007.

[4] Ieee standard for broadband over power line networks: Medium access control and physical layer specifications. In *IEEE Std. 1901*, 2010.

[5] Homeplug av2 whitepaper. *http://www.homeplug.org/tech-resources/resources/*, 2011.

[6] Ali Abdi, Kyle Wills, H Allen Barger, M-S Alouini, and Mostafa Kaveh. Comparison of the level crossing rate and average fade duration of rayleigh, rice and nakagami fading models with mobile channel data. In *Vehicular Technology Conference , 2000. IEEE-VTS Fall VTC 2000. 52nd*, volume 4, pages 1850–1857. IEEE, 2000.

[7] Pierre Achaichia, Marie Le Bot, and Pierre Siohan. Point-to-multipoint communication in power line networks: A novel fdm access method. In *IEEE International Conference on Communications*. IEEE, 2012.

[8] Fadel Adib, Zach Kabelac, Dina Katabi, and Robert C Miller. 3d tracking via body radio reflections. In *Usenix NSDI*, 2013.

[9] Fadel Adib, Hongzi Mao, Zachary Kabelac, Dina Katabi, and Robert C Miller. Smart homes that monitor breathing and heart rate. In *Proceedings of ACM CHI*, 2015.

[10] Piyush Agrawal and Neal Patwari. Correlated link shadow fading in multi-hop wireless networks. *IEEE TWC*, 2009.

[11] M. Alloulah, A. Isopoussu, C. Min, and F. Kawsar. On Tracking the Physicality of Wi-Fi: A Subspace Approach. *IEEE Access*, pages 1–1, 2019.

[12] Sonia Ancoli-Israel, Roger Cole, Cathy Alessi, Mark Chambers, William Moorcroft, and Charles P Pollak. The role of actigraphy in the study of sleep and circadian rhythms. *Sleep*, 2003.

[13] Dmitri Asonov and Rakesh Agrawal. Keyboard acoustic emanations. In *2012 IEEE Symposium on Security and Privacy*. IEEE Computer Society, 2004.

[14] Ahmed Osama Fathy Atya et al. Bolt: Realizing high throughput power line communication networks. In *Proceedings of ACM CoNEXT*, 2015.

[15] Martin Azizyan, Ionut Constandache, and Romit Roy Choudhury. Surroundsense: mobile phone localization via ambience fingerprinting. In *ACM MOBICOM*, 2009.

[16] Davide Balzarotti, Marco Cova, and Giovanni Vigna. Clearshot: Eavesdropping on keyboard input from video. In *Security and Privacy, 2008. SP 2008. IEEE Symposium on*. IEEE, 2008.

[17] James S Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in neural information processing systems*.

[18] Richard B Berry, Rita Brooks, Charlene E Gamaldo, Susan M Harding, CL Marcus, BV Vaughn, et al. The aasm manual for the scoring of sleep and associated events. *Rules, Terminology and Technical Specifications, Darien, Illinois, American Academy of Sleep Medicine*, 2012.

[19] Peter H Bloch and Marsha L Richins. Shopping without purchase: An investigation of consumer browsing behavior. *NA-Advances in Consumer Research Volume 10*, 1983.

[20] Peter H Bloch, Nancy M Ridgway, and Daniel L Sherrell. Extending the concept of shopping: An investigation of browsing activity. *Journal of the Academy of Marketing Science*, 1989.

[21] Anthony Burke. System and method for retail customer tracking in surveillance camera network, September 28 2017. US Patent App. 15/076,708.

[22] Hasan Basri Çelebi. *Noise and multipath characteristics of power line communication channels*. PhD thesis, University of South Florida, 2010.

[23] Bo Chen, Vivek Yenamandra, and Kannan Srinivasan. Tracking keystrokes using wireless signals. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2015.

[24] Wenxi Chen, Xin Zhu, Tetsu Nemoto, Yumi Kanemitsu, Keiichiro Kitamura, and Ken-ichi Yamakoshi. Unconstrained detection of respiration rhythm and pulse rate with one under-pillow sensor during sleep. *Medical and Biological Engineering and Computing*, 2005.

[25] Donald G Childers, David P Skinner, and Robert C Kemerait. The cepstrum: A guide to processing. *Proceedings of the IEEE*, 1977.

[26] Jungchan Cho, Inhwan Hwang, and Songhwai Oh. Vibration-based surface recognition for smartphones. In *IEEE RTCSA*, 2012.

[27] Eun Kyoung Choe, Julie A Kientz, Sajanee Halko, Amanda Fonville, Dawn Sakaguchi, and Nathaniel F Watson. Opportunities for computing to support healthy sleep behavior. In *ACM CHI*, 2010.

[28] Byung Hun Choi, Gih Sung Chung, Jin-Seong Lee, Do-Un Jeong, and Kwang Suk Park. Slow-wave sleep estimation on a load-cell-installed bed: a non-constrained method. *Physiological measurement*, 2009.

[29] Clausen et al. Optimized link state routing protocol (olsr). In *RFC 3626*, 2003.

[30] Jonathan Connell, Quanfu Fan, Prasad Gabbur, Norman Haas, Sharath Pankanti, and Hoang Trinh. Retail video analytics: an overview and survey. In *Video Surveillance and Transportation Imaging Applications*. Int. Society for Optics and Photonics, 2013.

[31] Gibson Research Corporation. Zeo sleep manager pro. https://uk.pcmag.com/zeo-sleep-manager-pro/5064/review/zeo-sleep-manager-pro, December 2012.

[32] Nelson Costa and Simon Haykin. A novel wideband MIMO channel model and experimental validation. *IEEE Trans. Antennas Propag.*, 56(2):550–562, 2008.

[33] Eliran Dafna, Ariel Tarasiuk, and Yaniv Zigel. Sleep-wake evaluation from whole-night non-contact audio recordings of breathing sounds. *PloS one*, 2015.

[34] JP D'Amato, C Garcia Bauzaa, and E Rinaldib. Consumer buying metrics extraction using computer vision techniques.

[35] GR De Bruijne, PCW Sommen, and RM Aarts. Detection of epileptic seizures through audio classification. In *4th European conference of the International Federation for Medical and Biological Engineering*. Springer, 2009.

[36] Luca Di Bert, Peter Caldera, David Schwingshackl, and Andrea M Tonello. On noise modeling for power line communications. In *IEEE International Symposium on Power Line Communications and Its Applications*. IEEE, 2011.

[37] Tobias Domhan, Jost Tobias Springenberg, and Frank Hutter. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[38] EPC EPCglobal. Radio-frequency identity protocols class-1 generation-2 uhf rfid protocol for communications at 860 mhz - 960 mhz. EPCGlobal Inc., 1.2.0 edition, 2008.

[39] Magnus Eriksson. Dynamic single frequency networks. *Selected Areas in Communications, IEEE Journal on*, 2001.

[40] Fitbit. Fitbit. https://www.fitbit.com/, July 2018.

[41] Toshiki Fujino, Masaki Kitazawa, Takashi Yamada, Masakazu Takahashi, Gaku Yamamoto, Atsushi Yoshikawa, and Takao Terano. Analyzing in-store shopping paths from indirect observation with rfidtags communication data. *Journal on Innovation and Sustainability. RISUS ISSN 2179-3565*, 2014.

[42] Andrea Goldsmith. *Wireless communications*. Cambridge university press, 2005.

[43] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[44] Jason Griffin and Steven Fyke. User hand detection for wireless devices, 2008. US Patent 7,430,439.

[45] Weixi Gu, Zheng Yang, Longfei Shangguan, Wei Sun, Kun Jin, and Yunhao Liu. Intelligent sleep stage mining service with smartphones. In *Proceedings of ACM Ubicomp*, 2014.

[46] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. 802.11 with multiple antennas for dummies. *ACM SIGCOMM Computer Communication Review*, 2010.

[47] Daniel Halperin, Wenjun Hu, Anmol Sheth, and David Wetherall. Tool release: gathering 802.11 n traces with channel state information. *ACM SIGCOMM Computer Communication Review*, 2011.

[48] Chunmei Han, Kaishun Wu, Yuxi Wang, and Lionel M Ni. Wifall: Device-free fall detection by wireless networks. In *INFOCOM, 2014 Proceedings IEEE*. IEEE, 2014.

[49] Jinsong Han, Han Ding, Chen Qian, Wei Xi, Zhi Wang, Zhiping Jiang, Longfei Shangguan, and Jizhong Zhao. Cbid: A customer behavior identification system using passive tags. *IEEE/ACM Transactions on Networking*, 2016.

[50] Tian Hao, Guoliang Xing, and Gang Zhou. isleep: unobtrusive sleep quality monitoring using smartphones. In *Proceedings of ACM Sensys*, 2013.

[51] Taro Hayasaki, Daisuke Umehara, Satoshi Denno, and Masahiro Morikura. A bit-loaded ofdma for in-home power line communications. In *Power Line Communications and Its Applications, 2009. ISPLC 2009. IEEE International Symposium on*, pages 171–176. IEEE, 2009.

[52] Adrienne Heinrich, Frank van Heesch, Bhargava Puvvula, and Mukul Rocque. Video based actigraphy and breathing monitoring from the bedside table of shared beds. *Journal of Ambient Intelligence and Humanized Computing*, 2015.

[53] Peter Hillyard, Anh Luong, Alemayehu Solomon Abrar, Neal Patwari, Krishna Sundar, Robert Farney, Jason Burch, Christina Porucznik, and Sarah Hatch Pollard. Experience: Cross-technology radio respiratory monitoring performance study. In *ACM MOBICOM*, 2018.

[54] Hristo D Hristov. *Fresnal Zones in Wireless Links, Zone Plate Lenses and Antennas*. Artech House, Inc., 2000.

[55] Impinj. Impinj Speedway R420. http://www.impinj.com/products/readers/speedway-revolution/, 2017. [Online; accessed February 24, 2017].

[56] Impinj. Octane SDK. https://support.impinj.com/hc/en-us/articles/202755268-Octane-SDK, 2017. [Online; accessed March 25, 2019].

[57] Impinj. Enhanced Shopper Experience. https://www.impinj.com/solutions/retail/enhanced-shopper-experience/, 2019. [Online; accessed Oct 20, 2019].

[58] Rajendra K Jain, Dah-Ming W Chiu, and William R Hawe. *A quantitative measure of fairness and discrimination for resource allocation in shared computer system*. Eastern Research Laboratory, MA, 1984.

[59] Jyh-Shing Roger Jang. Machine learning toolbox, available at http://mirlab.org/jang/matlab/toolbox/machinelearning, accessed on december 23, 2014.

[60] Wenjun Jiang, Chenglin Miao, Fenglong Ma, Shuochao Yao, Yaqing Wang, Ye Yuan, Hongfei Xue, Chen Song, Xin Ma, Dimitrios Koutsonikolas, et al. Towards environment independent device free human activity recognition. In *ACM MOBICOM*, 2018.

[61] Rasoul Karimi, Alexandros Nanopoulos, and Lars Schmidt-Thieme. Rfid-enhanced museum for interactive experience. In *Multimedia for cultural heritage*, pages 192–205. Springer, 2012.

[62] Yuusuke Kawakita and Jin Mitsugi. Anti-collision performance of gen2 air protocol in random error communication link. In *Applications and the Internet Workshops, 2006. SAINT Workshops 2006. International Symposium on*. IEEE, 2006.

[63] Matthew Kay, Eun Kyoung Choe, Jesse Shepherd, Benjamin Greenstein, Nathaniel Watson, Sunny Consolvo, and Julie A Kientz. Lullaby: a capture & access system for understanding the sleep environment. In *Proceedings of ACM Ubicomp*, 2012.

[64] Bryce Kellogg, Vamsi Talla, and Shyamnath Gollakota. Bringing gesture recognition to all devices. In *Usenix NSDI*, 2014.

[65] Raghunandan H Keshavan, Andrea Montanari, and Sewoong Oh. Matrix completion from noisy entries. *JMLR*, 2010.

[66] Erick Christian Kobres and Lyle Sandler. Whole store scanner, October 18 2016. US Patent 9,473,747.

[67] Juha M Kortelainen, Martin O Mendez, Anna Maria Bianchi, Matteo Matteucci, and Sergio Cerutti. Sleep staging based on signals acquired through bed sensor. *IEEE Transactions on Information Technology in Biomedicine*, 2010.

[68] Jean Krieger, Nelly Maglasiu, Emilia Sforza, and Daniel Kurtz. Breathing during sleep in normal middle-aged subjects. *Sleep*, 1990.

[69] Kai Kunze and Paul Lukowicz. Symbolic object localization through active sampling of acceleration and sound signatures. In *Springer International Conference on Ubiquitous Computing*, 2007.

[70] Clete A Kushida, Michael R Littner, Timothy Morgenthaler, Cathy A Alessi, Dennis Bailey, Jack Coleman Jr, Leah Friedman, Max Hirshkowitz, Sheldon Kapen, Milton Kramer, et al. Practice parameters for the indications for polysomnography and related procedures: an update for 2005. *Sleep*, 2005.

[71] Gierad Laput, Robert Xiao, and Chris Harrison. Viband: High-fidelity bio-acoustic sensing using commodity smartwatch accelerometers. In *ACM UIST*, 2016.

[72] Haniph A Latchman, Srinivas Katar, Larry Yonge, and Sherman Gavette. *Homeplug AV and IEEE 1901: A Handbook for PLC Designers and Users*. John Wiley & Sons, 2013.

[73] Stéphane Lathuilière, Pablo Mesejo, Xavier Alameda-Pineda, and Radu Horaud. A comprehensive analysis of deep regression. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[74] Daniele Liciotti, Marco Contigiani, Emanuele Frontoni, Adriano Mancini, Primo Zingaretti, and Valerio Placidi. Shopper analytics: A customer activity recognition system using a distributed rgb-d camera network. In *International workshop on video analytics for audience measurement in retail and digital signage*, 2014.

[75] Jian Liu, Yingying Chen, and Marco Gruteser. Vibkeyboard: virtual keyboard leveraging physical vibration. In *ACM MOBICOM*, 2016.

[76] Jian Liu, Yingying Chen, Marco Gruteser, and Yan Wang. Vibsense: Sensing touches on ubiquitous surfaces through vibration. In *IEEE SECON*, 2017.

[77] Jian Liu, Yan Wang, Yingying Chen, Jie Yang, Xu Chen, and Jerry Cheng. Tracking vital signs during sleep leveraging off-the-shelf wifi. In *Proceedings of ACM MobiHoc*, 2015.

[78] Jingwen Liu, Yanlei Gu, and Shunsuke Kamijo. Customer behavior recognition in retail store from surveillance camera. In *2015 IEEE International Symposium on Multimedia (ISM)*, 2015.

[79] Tianci Liu, Lei Yang, Xiang-Yang Li, Huaiyi Huang, and Yunhao Liu. Tagbooth: Deep shopping data acquisition powered by rfid tags. In *Computer Communications (INFOCOM), 2015 IEEE Conference on*, pages 1670–1678. IEEE, 2015.

[80] Xuefeng Liu, Jiannong Cao, Shaojie Tang, and Jiaqi Wen. Wi-sleep: Contactless sleep monitoring via wifi signals. In *IEEE RTSS*, 2014.

[81] Yunhao Liu, Yiyang Zhao, Lei Chen, Jian Pei, and Jinsong Han. Mining frequent trajectory patterns for activity monitoring using radio frequency tag arrays. *IEEE TPDS*, 2012.

[82] Xi Long. On the analysis and classification of sleep stages from cardiorespiratory activity. *SLEEP-WAKE*, 2015.

[83] Xi Long, Pedro Fonseca, Jérôme Foussier, Reinder Haakma, and Ronald M Aarts. Sleep and wake classification with actigraphy and respiratory effort using dynamic warping. *IEEE journal of biomedical and health informatics*, 2014.

[84] Ilya Loshchilov and Frank Hutter. Cma-es for hyperparameter optimization of deep neural networks. *arXiv preprint arXiv:1604.07269*, 2016.

[85] Bastien Lyonnet, Cornel Ioana, and Moeness G Amin. Human gait classification using microdoppler time-frequency signal representations. In *Radar Conference , 2010 IEEE*. IEEE, 2010.

[86] MathWorks. Blob Analysis. https://www.mathworks.com/help/vision/ref/blobanalysis.html, 2017. [Online; accessed February 21, 2017].

[87] Matteo Migliorini, Anna M Bianchi, Domenico Nisticò, Juha Kortelainen, Edgar Arce-Santana, Sergio Cerutti, and Martin O Mendez. Automatic sleep staging based on ballisto-cardiographic signals recorded through bed sensors. In *IEEE EMBC*, 2010.

[88] Jun-Ki Min, Afsaneh Doryab, Jason Wiese, Shahriyar Amini, John Zimmerman, and Jason I Hong. Toss'n'turn: smartphone as sleep and sleep quality detector. In *Proceedings of ACM CHI*, 2014.

[89] Fabian Monrose, Michael K Reiter, and Susanne Wetzel. Password hardening based on keystroke dynamics. *International Journal of Information Security - Springer*, 2002.

[90] Fabian Monrose and Aviel Rubin. Authentication via keystroke dynamics. In *ACM CCS*, 1997.

[91] M. Moshtaghi and et. al. Incremental elliptical boundary estimation for anomaly detection in wireless sensor networks. In *IEEE ICDM*, 2011.

[92] Meinard Müller. Dynamic time warping. *Information retrieval for music and motion*, 2007.

[93] Rohan Murty et al. Characterizing the end-to-end performance of indoor powerline networks. *Harvard University Microsoft Research*, 2008.

[94] Rajalakshmi Nandakumar, Bryce Kellogg, and Shyamnath Gollakota. Wi-fi gesture recognition on existing devices. *arXiv preprint arXiv:1411.5394*, 2014.

[95] Santosh Nannuru, Yunpeng Li, Yan Zeng, Mark Coates, and Bo Yang. Radio-frequency tomography for passive indoor multitarget tracking. *IEEE Transactions on Mobile Computing*, 2013.

[96] Andrew J Newman and Gordon R Foxall. In-store customer behaviour in the fashion sector: some emerging methodological and theoretical directions. *International Journal of Retail & Distribution Management*, 2003.

[97] Anh Nguyen, Raghda Alqurashi, Zohreh Raghebi, Farnoush Banaei-Kashani, Ann C Halbower, and Tam Vu. Libs: A lightweight and inexpensive in-ear sensing system for automatic whole-night sleep stage monitoring. *GetMobile: Mobile Computing and Communications*, 2017.

[98] Pavel V Nikitin and KV Seshagiri Rao. Antennas and propagation in uhf rfid systems. In *IEEE international conference on RFID*, 2008.

[99] O. Oura ring. https://ouraring.com/, July 2018.

[100] Cecilia Occhiuzzi and Gaetano Marrocco. The rfid technology for neurosciences: feasibility of limbs' monitoring in sleep diseases. *IEEE Transactions on Information Technology in Biomedicine*, 2010.

[101] Cecilia Occhiuzzi, Carmen Vallese, Sara Amendola, Sabina Manzari, and Gaetano Marrocco. Night-care: A passive rfid system for remote monitoring and control of overnight living environment. *Elsevier Procedia Computer Science*, 2014.

[102] Joonas Paalasmaa, Mikko Waris, Hannu Toivonen, Lasse Leppäkorpi, and Markku Partinen. Unobtrusive online monitoring of sleep at home. In *IEEE EMBC*, 2012.

[103] Neal Patwari and Sneha K Kasera. Robust location distinction using temporal link signatures. In *Proceedings of the 13th annual ACM International Conference on Mobile computing and networking*. ACM, 2007.

[104] Neal Patwari and Sneha K Kasera. Temporal link signature measurements for location distinction. *Mobile Computing, IEEE Transactions on*, 2011.

[105] Eduardo Marques Pereira, Jaime S Cardoso, and Ricardo Morla. Motion flow tracking in unconstrained videos for retail scenario. In *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2013.

[106] Dirk Pevernagie, Ronald M Aarts, and Micheline De Meyer. The acoustics of snoring. *Sleep medicine reviews*, 2010.

[107] Ming-Zher Poh, Daniel J McDuff, and Rosalind W Picard. Advancements in noncontact, multiparameter physiological measurements using a webcam. *IEEE Transactions on Biomedical Engineering*, 2011.

[108] Mirela Popa, Alper Kemal Koc, Leon JM Rothkrantz, Caifeng Shan, and Pascal Wiggers. Kinect sensing of shopping related actions. In *International Joint Conference on Ambient Intelligence*. Springer, 2011.

[109] Mirela Popa, Leon Rothkrantz, Zhenke Yang, Pascal Wiggers, Ralph Braspenning, and Caifeng Shan. Analysis of shopping behavior based on surveillance system. In *IEEE SMC*, 2010.

[110] Qifan Pu, Sidhant Gupta, Shyamnath Gollakota, and Shwetak Patel. Whole-home gesture recognition using wireless signals. In *Proceedings of the 19th annual International Conference on Mobile computing & networking*. ACM, 2013.

[111] Tauhidur Rahman, Alexander T Adams, Ruth Vinisha Ravichandran, Mi Zhang, Shwetak N Patel, Julie A Kientz, and Tanzeem Choudhury. Dopplesleep: A contactless unobtrusive sleep sensing system using short-range doppler radar. In *Proceedings of ACM Ubicomp*, 2015.

[112] Benjamin Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 2011.

[113] RedMed. S+ sleep sensor. https://www.resmed.com/us/en/consumer/s-plus.html, 2018.

[114] Brian D Ripley. *Spatial statistics*. John Wiley & Sons, 2005.

[115] Joseph Roth, Xiaoming Liu, and Dimitris Metaxas. On continuous user authentication via typing behavior. *Image Processing, IEEE Transactions on*, 2014.

[116] Joseph Roth, Xiaoming Liu, Arun Ross, and Dimitris Metaxas. Investigating the discriminative power of keystroke sound. *Information Forensics and Security, IEEE Transactions on*, 2015.

[117] Avi Sadeh and Christine Acebo. The role of actigraphy in sleep medicine. *Sleep medicine reviews*, 2002.

[118] Avi Sadeh, Peter J Hauri, Daniel F Kripke, and Peretz Lavie. The role of actigraphy in the evaluation of sleep disorders. *Sleep*, 1995.

[119] Michael J Sateia. International classification of sleep disorders. *Elsevier Chest*, 2014.

[120] Souvik Sen, Jeongkeun Lee, Kyu-Han Kim, and Paul Congdon. Avoiding multipath to revive inbuilding wifi localization. In *Proceedings of the 11th annual International Conference on Mobile systems, applications, and services*. ACM, 2013.

[121] Ilari Shafer. Learning location from vibration. http://www.mrcaps.com/, 2013.

[122] Longfei Shangguan, Zimu Zhou, Xiaolong Zheng, Lei Yang, Yunhao Liu, and Jinsong Han. Shopminer: Mining customer shopping behavior in physical clothing stores with cots rfid devices. In *ACM Sensys*, 2015.

[123] Stephan Sigg, Markus Scholz, Shuyu Shi, Yusheng Ji, and Michael Beigl. Rf-sensing of activities from non-cooperative subjects in device-free recognition systems using ambient and local signals. *Mobile Computing, IEEE Transactions on*, 2014.

[124] Stephan Sigg, Shuyu Shi, Felix Buesching, Yusheng Ji, and Lars Wolf. Leveraging rf-channel fluctuation for activity recognition: Active and passive systems, continuous and rssi-based signal features. In *Proceedings of International Conference on Advances in Mobile Computing & Multimedia*. ACM, 2013.

[125] SMARTRAC. DogBone Impinj Monza 4D. https://www.smartrac-group.com/files/content/Products_Services/PDF/SMARTRAC_DOGBONE_IMPINJ_MONZA_4D.pdf, 2017. [Online; accessed March 25, 2019].

[126] David B Smith and Leif W Hanlen. Channel modeling for wireless body area networks. In *Ultra-Low-Power Short-Range Radios*, pages 25–55. Springer, 2015.

[127] SolidRun. Hummingboard. https://www.solid-run.com/nxp-family/hummingboard/, 2018.

[128] Sparkfun. Erm and lra motors. https://learn.sparkfun.com/tutorials/haptic-motor-driver-hook-up-guide/erm-and-lra-motors, 2018.

[129] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, January 2014.

[130] C. Studer, S. Medjkouh, E. Gönültaş, T. Goldstein, and O. Tirkkonen. Channel Charting: Locating Users within the Radio Environment using Channel State Information. *IEEE Access*, 6:47682–47698, 2018.

[131] Xiao Sun, Li Qiu, Yibo Wu, Yeming Tang, and Guohong Cao. Sleepmonitor: Monitoring respiratory rate and body position during sleep using smartwatch. *Proceedings of ACM IMWUT*, 2017.

[132] TechWorld. Amazon Go looks convenient, but raises huge questions over privacy. https://www.techworld.com/business/amazon-go-looks-amazing-but-at-what-cost-3651434/, 2018. [Online; accessed November 3, 2019].

[133] Frederic Thouin, Santosh Nannuru, and Mark Coates. Multi-target tracking for measurement models with additive contributions. In *Information Fusion (FUSION), 2011 Proceedings of the 14th International Conference on*. IEEE, 2011.

[134] Antonia M Tulino, Angel Lozano, and Sergio Verdú. Capacity-Achieving Input Covariance for Single-User Multi-Antenna Channels. *IEEE Trans. Wireless Commun.*, 5(3), 2006.

[135] Yu-Chih Tung and Kang G Shin. Echotag: Accurate infrastructure-free indoor location tagging with smartphones. In *ACM MOBICOM*, 2015.

[136] Yu-Chih Tung and Kang G Shin. Expansion of human-phone interface by sensing structure-borne sound propagation. In *ACM MOBISYS*, 2016.

[137] Twice. Are Amazon Go Stores Putting Consumer Data At Risk? https://www.twice.com/blog/are-amazon-go-stores-putting-consumer-data-at-risk, 2018. [Online; accessed November 3, 2019].

[138] Joris C Verster, Seithikurippu R Pandi-Perumal, and David L Streiner. *Sleep and quality of life in clinical medicine*. Springer, 2008.

[139] Aditya Virmani and Muhammad Shahzad. Position and orientation agnostic gesture recognition using wifi. In *ACM MOBISYS*, 2017.

[140] Christina Vlachou, Albert Banchs, Julien Herzen, and Patrick Thiran. Analyzing and boosting the performance of power-line communication networks. In *Proceedings of ACM International on Conference on emerging Networking Experiments and Technologies*. ACM, 2014.

[141] Christina Vlachou, Albert Banchs, Julien Herzen, and Patrick Thiran. On the mac for power-line communications: Modeling assumptions and performance tradeoffs. In *IEEE International Conference on Network Protocols (ICNP)*. IEEE, 2014.

[142] Christina Vlachou et al. Electri-fi your data: Measuring and combining power-line communications with wifi. In *Proceedings of ACM Internet Measurement Conference* , number EPFL-CONF-211905, 2015.

[143] Martin Vuagnoux and Sylvain Pasini. Compromising electromagnetic emanations of wired and wireless keyboards. In *USENIX Security Symposium* , 2009.

[144] Benjamin Wagner, Neal Patwari, and Dirk Timmermann. Passive rfid tomographic imaging for device-free user localization. In *IEEE Positioning Navigation and Communication*, 2012.

[145] Guanhua Wang, Yongpan Zou, Zimu Zhou, Kaishun Wu, and Lionel M Ni. We can hear you with wi-fi! In *Proceedings of the 20th annual International Conference on Mobile computing and networking*. ACM, 2014.

[146] Hao Wang, Daqing Zhang, Junyi Ma, Yasha Wang, Yuxiang Wang, Dan Wu, Tao Gu, and Bing Xie. Human respiration detection with commodity wifi devices: do user location and body orientation matter? In *Proceedings of ACM Ubicomp*, 2016.

[147] Wei Wang, Alex X Liu, Muhammad Shahzad, Kang Ling, and Sanglu Lu. Understanding and modeling of wifi signal based human activity recognition. In *ACM MOBICOM*, 2015.

[148] Yan Wang, Jian Liu, Yingying Chen, Marco Gruteser, Jie Yang, and Hongbo Liu. E-eyes: device-free location-oriented activity identification using fine-grained wifi signatures. In *Proceedings of the 20th annual International Conference on Mobile computing and networking*. ACM, 2014.

[149] Joe H Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 1963.

[150] John B Webster, Daniel F Kripke, Sam Messin, Daniel J Mullaney, and Grant Wyborney. An activity-based sleep monitor system for ambulatory use. *Sleep*, 1982.

[151] Werner Weichselberger, Markus Herdin, Huseyin Ozcelik, and Ernst Bonek. A stochastic MIMO channel model with joint correlation of both link ends. *IEEE Trans. Wireless Commun.*, 5(1):90–100, 2006.

[152] Joey Wilson and Neal Patwari. Radio tomographic imaging with wireless networks. *Mobile Computing, IEEE Transactions on*, 2010.

[153] Withings. Withings sleep tracking mat. https://www.withings.com/us/en/sleep, 2018.

[154] Xethru. Xethru vs. polysomnography (psg) comparative study. https://www.xethru.com/community/resources/categories/white-papers.6/.

[155] Xethru. Respiration sensor x4m200. https://www.xethru.com/x4m200-respiration-sensor.html, 2018.

[156] Wei Xi, Jizhong Zhao, Xiang-Yang Li, Kun Zhao, Shaojie Tang, Xue Liu, and Zhiping Jiang. Electronic frog eye: Counting crowd using wifi. In *INFOCOM, 2014 Proceedings IEEE*, 2014.

[157] Jiang Xiao, Kaishun Wu, Youwen Yi, and Lionel M Ni. Fifs: Fine-grained indoor fingerprinting system. In *Computer Communications and Networks (ICCCN), 2012 21st International Conference on*. IEEE, 2012.

[158] Xiaomi. Xiaomi mi band 3. https://www.mi.com/en/miband/, July 2018.

[159] Zheng Yang, Zimu Zhou, and Yunhao Liu. From rssi to csi: Indoor localization via channel response. *ACM Computing Surveys (CSUR)*, 2013.

[160] Zhicheng Yang, Parth H Pathak, Yunze Zeng, Xixi Liran, and Prasant Mohapatra. Monitoring vital signs using millimeter wave. In *Proceedings of ACM MobiHoc*, 2016.

[161] Vivek Yenamandra and Srinivasan Kannan. Vidyut: Exploiting power line infrastructure for enterprise wireless networks. In *Proceedings of ACM SIGCOMM*, 2014.

[162] Shichao Yue, Hao He, Hao Wang, Hariharan Rahul, and Dina Katabi. Extracting multi-person respiration from entangled rf signals. *Proceedings of ACM IMWUT*, 2018.

[163] Saira Zahid, Muhammad Shahzad, Syed Ali Khayam, and Muddassar Farooq. Keystroke-based user identification on smart phones. In *Recent Advances in Intrusion Detection - Springer*, 2009.

[164] Fusang Zhang, Daqing Zhang, Jie Xiong, Hao Wang, Kai Niu, Beihong Jin, and Yuxiang Wang. From fresnel diffraction model to fine-grained human respiration sensing with commodity wi-fi devices. *ACM IMWUT*, 2018.

[165] Mingmin Zhao, Shichao Yue, Dina Katabi, Tommi S Jaakkola, and Matt T Bianchi. Learning sleep stages from radio signals: a conditional adversarial architecture. In *IEEE ICML*, 2017.

[166] Yang Zhao, Neal Patwari, Jeff M Phillips, and Suresh Venkatasubramanian. Radio tomographic imaging and tracking of stationary and moving people via kernel distance. In *ACM IPSN 2013*.

[167] Yu Zhong, Yan Deng, and Anubhav K Jain. Keystroke dynamics for user authentication. In *IEEE CVPR Workshop*, 2012.

[168] Zimu Zhou, Zheng Yang, Chenshu Wu, Longfei Shangguan, and Yunhao Liu. Towards omnidirectional passive human detection. In *INFOCOM, 2013 Proceedings IEEE*. IEEE, 2013.

[169] Tong Zhu, Qiang Ma, Shanfeng Zhang, and Yunhao Liu. Context-free attacks using keyboard acoustic emanations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, 2014.

[170] Li Zhuang, Feng Zhou, and J Doug Tygar. Keyboard acoustic emanations revisited. *ACM Transactions on Information and System Security (TISSEC)*, 2009.

[171] Manfred Zimmermann and Dostert Klaus. An analysis of the broadband noise scenario in powerline networks. In *International Symposium on Powerline Communications and its Applications*, 2000.

[172] Manfred Zimmermann and Dostert Klaus. Analysis and modeling of impulsive noise in broad-band powerline communications. *IEEE Transactions on Electromagnetic Compatibility*, (1), 2002.

[173] Manfred Zimmermann and Dostert Klaus. A multipath model for the powerline channel. *IEEE Transactions on Communications*, (4), 2002.