



**DETERMINING PHYSICAL
CHARACTERISTICS WITH INFORMATION
LEAKAGE IN IEEE 802.11ac**

THESIS

Albert D. Taglieri, 1st Lieutenant, USAF
AFIT-ENG-MS-21-S-018

**DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY**

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views expressed in this document are those of the author and do not reflect the official policy or position of the United States Air Force, the United States Department of Defense or the United States Government. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

AFIT-ENG-MS-21-S-018

DETERMINING PHYSICAL CHARACTERISTICS WITH INFORMATION
LEAKAGE IN IEEE 802.11ac

THESIS

Presented to the Faculty
Department of Electrical and Computer Engineering
Graduate School of Engineering and Management
Air Force Institute of Technology
Air University
Air Education and Training Command
in Partial Fulfillment of the Requirements for the
Degree of Master of Science in Computer Science

Albert D. Taglieri, BCS, B. Math
1st Lieutenant, USAF

September 16, 2021

DISTRIBUTION STATEMENT A
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

AFIT-ENG-MS-21-S-018

DETERMINING PHYSICAL CHARACTERISTICS WITH INFORMATION
LEAKAGE IN IEEE 802.11ac

THESIS

Albert D. Taglieri, BCS, B. Math
1st Lieutenant, USAF

Committee Membership:

Barry E. Mullins, Ph.D., P.E.
Chair

Robert F. Mills, Ph.D.
Member

Timothy H. Lacey, Ph.D., CISSP
Member

Abstract

The nature of the wireless medium demands that advanced protocols utilize information about the physical environment in which they operate. The first update to the IEEE 802.11 protocol to explicitly detail physical information about the system is IEEE 802.11ac, with the introduction of VHT transmit beamforming. Previous research has demonstrated the dangers and potentials of using information leakage to determine characteristics of a system, such as through MAC addresses. This is the first study to examine similar information leakage through beamforming.

This thesis hypothesizes that beamforming in IEEE 802.11ac allows an eavesdropper to collect signals for the purpose of distinguishing and identifying device locations, and distinguishing the amount of motion in the network environment. This is done by devising a method to decode sniffed beamforming management frames in IEEE 802.11ac, as well as creates a scheme of statistical analysis and modeling for beamforming data. Through statistical tests, this research validates the ability to model beamforming data, then applies it in measuring the success and failure of the model to distinguish and identify device locations, as well as measuring the success and failure of the model to identify motion in the network environment.

The FPX is the process developed for extracting the encoded CSI from IEEE 802.11ac beamforming feedback. The FCaE tool provides the framework and utilities to analyze, correlate, and evaluate such data. After sniffing beamforming exchanges from a test network environment, with different device locations and amounts of motion, this work demonstrates the ability of the FCaE to use such information in distinguishing between locations with a 98.5% success rate. It further demonstrates a model which correctly identifies device location with up to 83.5% success. It also

demonstrates the distinguishing of different amounts of motion with a success rate of 75%. This work finally provides suggestions for future avenues of research, incorporating the characteristics of feedback matrix data, and the limitations of the tests developed in the FCaE.

In so doing, this research provides tools for expanding pattern analysis, and turning every IEEE 802.11ac device into a wireless sensor. It also highlights the security risks associated with information leakage, and suggests approaches to their exploitation.

Acknowledgements

I thank the members of the committee, Dr. Mullins, Dr. Mills, and Dr. Lacey for their guidance, patience, and assistance during the research. Especially, I thank my advisor, Dr. Mullins, for his careful reviewing and ensuring of the best possible research. Your questions and challenges made my work better. I thank also Dr. Taylor for his instruction in experimental design, which proved invaluable for the research. I never enjoyed statistics until your instruction. I thank my family and friends for their patience and assistance during the program. Especially, I thank Lt. Voltz for his feedback and assistance, and Lt. Lynch for his editing. Finally I thank my Christ for his mercy, grace, and love to me among other abundant gifts, which sustained me and without which I could not have succeeded. Surely you follow me, and I will dwell in your house forever.

Table of Contents

	Page
Abstract	iv
List of Figures	xi
List of Tables	xiii
Acronyms	xiv
I. Introduction	1
1.1 Problem Background	1
1.2 Problem Statement	2
1.3 Research Goals	2
1.4 Hypothesis	3
1.5 Approach	3
1.6 Assumptions and Limitations	4
1.7 Contributions	4
1.8 Thesis Overview	5
II. Background and Literature Review	6
2.1 Overview	6
2.2 The 802.11 Protocol	6
2.2.1 802.11 Operation	6
2.2.2 Improvements in 802.11ac	7
2.2.3 Beamforming in 802.11ac	8
2.3 Related Work	26
2.3.1 Information Leakage as a Security Concern	27
2.3.2 Pattern-of-Life Modeling with 802.11	27
2.3.3 Sensing for Location	28
2.3.4 Sensing for Device Identity	29
2.3.5 Sensing for System Characteristics	29
2.4 This Research in Relation to Previous Research	30
III. Design	32
3.1 Overview	32
3.2 System Components	32
3.2.1 ASUS AC1300 RT-ACRH13	32
3.2.2 Dell XPS 15 7590	33
3.2.3 Kali Virtual Machine	34
3.3 Feedback Packet Extractor (FPX)	38
3.3.1 Export	41

	Page
3.3.2 Decode	41
3.3.3 Decompress	45
3.4 Feedback Correlation and Evaluation Tool (FCaE)	46
3.4.1 Overview	46
3.4.2 FCaE Mathematics	46
3.4.3 FCaE Trial Resampling	48
3.4.4 FCaE Testing	49
3.4.5 FCaE Visualization	52
3.4.6 FCaE Process Summary	52
IV. Methodology	55
4.1 Overview	55
4.2 Problem	55
4.3 System Under Test	56
4.4 Assumptions	59
4.5 Response Variables	59
4.6 Controlled Variables	61
4.7 Uncontrolled Variables	63
4.8 Experimental Design	64
4.8.1 System Configuration Verification	64
4.8.2 Test Area Physical Layout	69
4.9 Data Collection	70
4.10 Statistical Analysis	73
4.10.1 Distinguishing Success Rate (DSR)	73
4.10.2 Location Correlation True Positive Rate (LCTPR)	73
4.10.3 Location Correlation False Positive Rate (LCFPR)	74
4.10.4 Location Correlation False Negative Rate (LCFNR)	74
4.10.5 Motion Correlation True Positive Rate (MCTPR)	75
4.10.6 Motion Correlation False Positive Rate (MCFPR)	75
4.10.7 Motion Correlation False Negative Rate (MCFNR)	75
4.10.8 Environmental Distinguishing Rate (EDR)	76
V. Results and Analysis	77
5.1 Overview	77
5.2 Feedback Matrix Characteristics	77
5.3 FCaE Training	82
5.3.1 Baseline Data Analysis	82
5.3.2 Distinguishing Success Rate (DSR):	86

	Page
5.4 FCaE Minimal Motion Evaluation	87
5.4.1 Location Correlation True Positive Rate (LCTPR)	87
5.4.2 Location Correlation False Positive Rate (LCFPR)	91
5.4.3 Location Correlation False Negative Rate (LCFNR)	91
5.5 FCaE Motion-Saturated Evaluation	93
5.5.1 Motion Correlation True Positive Rate (MCTPR)	93
5.5.2 Motion Correlation False Positive Rate (MCFPR)	94
5.5.3 Motion Correlation False Negative Rate (MCFNR)	94
5.6 FCaE Minimal Motion and Motion-Saturated Environment Analysis	96
5.7 Motion-Saturated Feedback as a Baseline	100
5.7.1 Baseline Accuracy	100
5.7.2 Minimal Motion Evaluation	102
VI. Conclusion	105
6.1 Overview	105
6.2 Research Conclusions	105
6.2.1 Device Location Correlation	105
6.2.2 Location Correlation During Motion	106
6.2.3 Distinguishing Environmental States	107
6.2.4 Conclusions from Feedback Analysis	108
6.3 Contributions	109
6.3.1 FPX	109
6.3.2 FCaE	110
6.3.3 Applications	110
6.4 Limitations	111
6.5 Future Work	112
Appendix A. Alternate Adapter Configuration Commands	114
Appendix B. FPX-AD Generated C Code Excerpt	115
Appendix C. FPX-AD Python Script	119
Appendix D. FPX-DCD Bash Script	123
Appendix E. FPX-DCP MATLAB Script	125
Appendix F. FPX Utility Functions	128

	Page
Appendix G. FCaE MATLAB Script	130
Appendix H. FCaE Utility Functions	153
Bibliography	163

List of Figures

Figure		Page
1.	Upconverter Diagram	11
2.	Upconversion Diagram	12
3.	Constellation Decoding Example	14
4.	Beamforming Constellation Comparison	16
5.	Beamforming Visualization	18
6.	IEEE 802.11 subcarrier alignment	21
7.	The MIMO control field's effect on the bit size of ϕ and ψ [1].	22
8.	IEEE 802.11 Feedback Angle Order	25
9.	Adapter Configuration	34
10.	Attack Platform	35
11.	VM Adapter Attachment	35
12.	Initial Adapter Configuration	36
13.	Adapter Monitor Mode Activation	37
14.	Adapter Frequency Correction	37
15.	Interface Activation	37
16.	Final Adapter Configuration	37
17.	FPX Process Block Diagram	40
18.	Feedback Decoding Layout	43
19.	Angle Bit Sizes	44
20.	FCaE Block Diagram	54
21.	System Under Test Diagram	58
22.	Adapter Configuration Verification	64

Figure		Page
23.	Router Configuration Confirmation	65
24.	802.11 Router Beacon Frame	67
25.	802.11 Adapter Probe Frame	68
26.	Experiment Location Layout	69
27.	Beamforming Feedback Wireshark Filter	72
28.	Baseline Feedback Comparison: West and East	80
29.	Environment Feedback Comparison: West.....	81
30.	Baseline Packet Distance Distribution: East	84
31.	Mean Packet Distance Distribution in Cardinal Directions	85
32.	Northeast Feedback Visualization	89
33.	North Feedback Comparison	99

List of Tables

Table		Page
1.	Research Comparison	31
2.	Randomized collection order for each test location. B is baseline collection, M is minimal-motion collection, and S is motion-saturated collection.	62
3.	DSR Baseline Results	87
4.	LCTPR Baseline Results	88
5.	Adjusted LCTPR Results	91
6.	LCFNR Baseline Results	92
7.	MCTPR Baseline Results	94
8.	MCFNR Baseline Results	95
9.	Motion Test Results for Alternate Evaluation Methods	95
10.	EDR Results: Motionless against Motion-Saturated	97
11.	EDR Results: Baseline against Tests	98
12.	DSR Motion-Saturated Baseline Results	101
13.	Motion-Saturated Baseline False Positives	101
14.	LCTPR Motion-Saturated Baseline Results	103
15.	LCFPR Motion-Saturated Baseline Results	103

Acronyms

AP IEEE 802.11 Access Point. 6, 7, 8, 19, 20, 23, 24, 56, 59, 67

BSS IEEE 802.11 Basic Service Set. 7

CSI Channel State Information. iv, 2, 4, 15, 18, 19, 27, 29, 30, 31, 51, 63

CSV Comma-Separated Values. 41

CUT Component Under Test. 32

DSFP Distinguishing Success False Positives. 59, 61

DSR Distinguishing Success Rate. 56, 73, 86, 87, 91, 92, 102

DSTP Distinguishing Success True Positives. 59, 61

EDC Environment Distinct Count. 60

EDR Environment Distinction Rate. 76, 96, 97

ESS IEEE 802.11 Extended Service Set. 7

FCaE Feedback Correlation and Evaluation Tool. iv, v, xi, 32, 33, 46, 54, 56, 58,
74, 75, 76, 77, 82, 83, 86, 88, 90, 91, 92, 93, 94, 96, 97, 100, 102, 104

FPX Feedback Packet Extractor Tool. iv, vii, 4, 32, 33, 38, 41, 45

FPX-DCP FPX Decompressor Component. 45

FPX-DCD FPX Decoder Component. 41, 45

GAO Government Accountability Office. 28

IEEE Institute of Electrical and Electronics Engineers. iv, v, xi, 1, 2, 4, 6, 7, 20, 21

IoT Internet of Things. 27, 28

ISM Industrial, Scientific, and Medical. 6, 7

JSON JavaScript Object Notation. 41

LCFN Location Correlation False Negatives. 60

LCFNR Location Correlation False Negative Rate. 74, 91, 92

LCFP Location Correlation False Positives. 60

LCFPR Location Correlation False Positive Rate. 74, 91, 92, 102

LCTP Location Correlation True Positives. 60, 87

LCTPR Location Correlation True Positive Rate. 74, 87, 90, 92, 102

LOS Line of Sight. 4, 17, 59, 63, 70

MAC Media Access Control. iv, 6, 28, 29

MCFN Motion Correlation False Negatives. 60

MCFNR Motion Correlation False Negative Rate. 75, 94

MCFP Motion Correlation False Positives. 60

MCFPR Motion Correlation False Positive Rate. 75, 94

MCS Modulation and Coding Set. 9, 15

MCTP Motion Correlation True Positives. 60

MCTPR Motion Correlation True Positive Rate. 75, 93, 94, 104

MIMO Multiple-Input Multiple-Output. xi, 8, 15, 20, 22

MU Multi-User. 4, 20

MU-MIMO Multi-User Multiple-Input Multiple-Output. 3, 8, 32

NDP Null-Data Packet. 19, 23

NIC Network Interface Card. 33, 34, 35, 36

PHY Physical Layer Specification. 6

QAM Quadrature Amplitude Modulation. 7, 8, 9, 13, 16

QPSK Quadrature Phase-Shift Keying. 8

SNR Signal to Noise Ratio. 18, 63

SoC System on a Chip. 32

STA IEEE 802.11 Station. 6, 7, 8, 19, 20, 26, 28

SU Single-User. 20, 21, 38, 62, 66

SUT System Under Test. 32, 55, 56, 58, 59

SVD Singular Value Decomposition. 17

VHT Very High Throughput. iv, 19, 20

VM Virtual Machine. 34

XML eXtensible Markup Language. 41

DETERMINING PHYSICAL CHARACTERISTICS WITH INFORMATION LEAKAGE IN IEEE 802.11ac

I. Introduction

1.1 Problem Background

Wi-Fi has become ubiquitous in modern society. At every coffee shop, library, school, home, and business, there is at least one Wi-Fi network, and often more. The number of devices, and number of wireless devices with it, are growing every day. More devices, with more capabilities, need the ability to connect and to transmit data. As devices become more sophisticated, the total quantity of data in need of transmission grows, leading to a demand for wireless systems and protocols to provide the capability. Wireless protocols, in satisfying the demands of data transmission needs, must deal with the constraints of the wireless medium and environments in which they operate. This means that protocols need more precise physical information about their environment to provide such services. The introduction of explicit beamforming mechanisms in IEEE 802.11ac (hereafter referred to as simply 802.11ac) [1] provides exactly this kind of physical information to wireless access point, allowing them to achieve a higher performance and channel capacity. Yet, the information which is sent over 802.11ac to accomplish this is sent without encryption or protection, as management frames for 802.11ac networks. This provides an opportunity for information leakage, which can reveal not only information about the devices and software on the system, but also physical information about the environment in which the wireless network resides, and the physical configuration of devices on the network

utilizing this beamforming capability.

1.2 Problem Statement

The purpose of this investigation is to discover whether IEEE 802.11ac beamforming feedback can provide insight into the physical environment of a network. The resolves into 2 specific questions:

1. Can 802.11ac beamforming feedback be correlated to physical position, such that two physical positions may be distinguished from one another by their beamforming feedback, and wireless device location identified by it?
2. Can the variance in 802.11ac beamforming feedback indicate information about the quantity of motion in the environment where the wireless network resides?

1.3 Research Goals

This work attempts to bring together and synthesize multiple strands of previous research for the purpose of providing new avenue of pattern analysis, which is difficult to defend against. The first strand of research is that of sensing, which several methods are used to sense locations of devices, movement or changes in the environment, or other information as a byproduct of monitoring the character of wireless signals. This research aims to build on this by using a new method of collecting information, utilizing 802.11ac devices transmitting beamforming feedback as the sensors, instead of installing dedicated sensors or reconfiguring and modifying hardware or software. The second strand is that of investigating the potentials of wireless protocols such as 802.11ac or 802.11ax, which incorporate several advanced features to increase their capability. Most previous research on Channel State Information (CSI) has been done with 802.11n, an older version of the protocol. This research aims to investigate a

particular feature of 802.11ac: the standardized explicit transmit beamforming, and analyze the information which it provides. The third strand is the use of information leakage to perform pattern analysis. This research attempts to demonstrate that the new beamforming capabilities of 802.11ac create a vulnerability for information leakage by directly including physical information about the system in the protocol, and do so in an unencrypted communication, which allows an eavesdropper to apply sensing analysis to beamforming data to be used in the service of pattern analysis, and to break previously tested pattern analysis defensive techniques.

1.4 Hypothesis

This research hypothesizes that the beamforming feedback data transmitted in 802.11ac allows an attacker to determine characteristics about the physical layout and environment in which the network resides. Specifically, the feedback data will have distinct characteristics depending on the device location, and the motion in the environment, and that these factors are detectable through it. Because of the complexities of multi-path optimization for Multi-User Multiple-Input-Multiple-Output (MU-MIMO) transmission, the reliability of detection will vary between environment states, being highest when the environment has minimal motion.

1.5 Approach

A test environment is created to consist of a client communicating with an 802.11ac access point, utilizing beamforming, and a third device to monitor and collect the traffic. To simulate a realistic environment, the test area is not cleared of objects, but resembles an apartment with various pieces of furniture, and other networks providing signal noise.

1.6 Assumptions and Limitations

The following assumptions are made when designing the test system:

- The access point is a realistic consumer router, representative of what might be found in a home or public area, while not intended to emulate higher-capability devices used for enterprise solutions.
- The beamforming is not performed between multiple disparately located devices simultaneously, but is performed with a single device at a time.
- There is not yet a Multiple User (MU) beamformee-capable adapter and driver which can operate in monitor mode.
- The most significant motion in the environment which affects CSI is that which disrupts Line of Sight (LOS).

1.7 Contributions

This document contributes to the field of IT security, and the field of sensing capability. These contributions are:

1. **Vulnerability Analysis.** This work demonstrates how 802.11ac and similar protocols can be monitored by an attacker for the purpose of characterizing the physical environment in which the signals occur.
2. **Feedback Packet Extractor (FPX).** This tool transforms the 802.11ac beamforming feedback report into mathematical form, through a series of scripts. The beamforming feedback report is normally encoded in a compressed form, with details contained in the IEEE 802.11ac standard, and requires decoding and decompression through complex formulas in order to provide the numerical form of the data.

3. **Feedback Correlation and Evaluation (FCaE).** This tool provides the ability describe and compare sets of beamforming feedback data, and to train a model of network behavior based on the collected data. From this, it provides methods for evaluating further beamforming feedback data against the model, which disclose information about locations of devices and the motion of objects in the network’s environment. It also provides a suite of analysis functions which allow basic trends in 802.11ac beamforming feedback to be visualized, described, and compared.
4. **Symbiosis.** This research demonstrates how different areas of research can be combined to further each one, specifically the areas of sensing and pattern analysis.

1.8 Thesis Overview

This document is arranged in six chapters. Chapter 2 provides a background on the mathematical and physical concepts necessary to understand the 802.11ac features which are leveraged in this work, as well as a survey and summary of the relevant research and influences upon which this thesis builds. Chapter 3 provides the design and configuration details of the system which is tested and evaluated. Chapter 4 discusses the methodology used in the experiments, and Chapter 5 provides a discussion of the results and analyzes their significance. Finally, Chapter 6 discusses the significance of the research as a whole in relation to the various strands from which it was derived, and suggests directions for future work that may build on it.

II. Background and Literature Review

2.1 Overview

This chapter describes the background information necessary for this research. First, the 802.11 protocol is discussed in Section 2.2. Section 2.3 surveys the state of related research in the areas of information leakage (Section 2.3.1), pattern analysis (Section 2.3.2), and wireless sensing (Section 2.3.3, Section 2.3.4, and Section 2.3.5), to provide the motivation for and basis upon which this research builds. Section 2.4 concludes by describing the relationship of this research to the previous work, and how it contributes to each surveyed field.

2.2 The 802.11 Protocol

IEEE 802.11 is the protocol specification used for wireless Ethernet. It simulates the same networking capabilities as Ethernet, but over a wireless medium instead of a wire [2]. Over time, 802.11 has improved and additional specifications for the Physical (PHY) and Media Access Control (MAC) layers were added to improve and extend functionality. The most recent revision is 802.11ax, approved in 2021 [3]. First Section 2.2.1 discusses the operation of the 802.11 protocol, and Section 2.2.2 discusses the improvements provided in the 802.11ac amendment to the protocol. Section 2.2.3 describes the beamforming mechanism in 802.11ac in detail.

2.2.1 802.11 Operation

IEEE 802.11 uses the 2.4 GHz industrial, scientific, and medical (ISM) band for radio communication. The makeup of an 802.11 network consists of stations (STA), a distribution system, and access points (AP). In contrast to Ethernet, 802.11 must use an extended Media Access Control MAC frame, due to the complexity and

diversity of physical communication methods which 802.11 can employ, in addition to the physical distribution of the system across multiple APs that is not present in Ethernet networks. In wireless networks, locality has a significance that is absent from Ethernet networks. With the basic units of APs and STAs, the network builds Basic Service Sets (BSS), which is the network services provided by access points to the area they are in. Extended Service Sets (ESS) are made up of BSSs and are the highest level of abstraction for an 802.11 network. In one sense, ESSs are networks, while BSSs are individual components of it, limited physically by the location of an AP. A consequence of the distinction between the BSS and ESS is that 802.11 allows a STA to move between different locations, served by different APs and BSSs, and to even so remain in the same network. IEEE 802.11 accounts for BSS changes, by tracking which BSS a device is part of in the distribution system, while maintaining ESS identity in a STA as it moves. Using this information, the distribution system may then determine which AP data must be sent to in order for it to arrive at the proper STA. The distribution system ordinarily resides on the router for a network [2].

2.2.2 Improvements in 802.11ac

Published in 2013, 802.11ac is an amendment to 802.11, with increased capabilities and new options for leveraging the wireless environment. The purpose of the ac amendment is to allow Wi-Fi to achieve gigabit speeds [4] [5]. It was formulated alongside 802.11ad, which provides similar speeds and options over different frequencies, and at different ranges. In order to accommodate these speeds, the 802.11ac protocol uses not only the 2.4 GHz band, but also the 5 GHz ISM band. It introduces wider channels, up to 160 MHz, and coding capabilities up to 256 QAM. Most importantly for this research, it introduces a new physical awareness of the

wireless environment through standardized transmit beamforming and multiple user MIMO (MU-MIMO) capabilities. This distinguishes it from its predecessor, 802.11n, which was both more complex and less capable, and relied on proprietary and diverse methods of beamforming [4]. 802.11ac has a standardized, simpler, and more capable scheme of beamforming. This new method is incidentally the basis of 802.11ac's MU-MIMO feature [4] [5].

2.2.3 Beamforming in 802.11ac

Beamforming is a method of steering a wireless signal [4]. Through this steering, multiple spatial streams can be directed to multiple devices in differing locations, thus performing MU-MIMO. In 802.11ac, beamforming comes in two varieties: implicit and explicit. Implicit beamforming is designed to accommodate older devices, which are not 802.11ac capable, and requires no interaction between the STA and the AP, with the AP autonomously determining the optimal steering. Explicit beamforming on the other hand, performs an exchange between the AP and the STA in order to focus the transmissions between them [4]. Section 2.2.3.1 discusses the mathematical basis for beamforming, Section 2.2.3.2 describes the standardized process by which 802.11ac performs beamforming, and Section 2.2.3.3 describes the form in which beamforming data is represented in the 802.11ac management packets.

2.2.3.1 Mathematical Theory of Beamforming

Beamforming is based in mathematical models of information transmission. The basic mathematical unit of information is a bit. Bits are grouped into symbols, which are represented using complex numbers [6]. The manner by which symbols become complex numbers is coding, and follows coding schemes such as quadrature phase-shift keying (QPSK), quadrature amplitude modulation (QAM), and others. 802.11ac

provides a diverse selection of coding schemes, called Modulation and Coding Sets (MCS) in the 802.11 standard [2]. QAM is defined by a square number, where it describes the number of different symbols that can be represented. The arrangement of the symbols on the complex plane is a constellation, which appears like a square grid in QAM schemes, and each constellation can be described in $\log_2(M)$ bits, where M is the number of symbols in the constellation [7]. In 802.11ac, 256 QAM allows 8 bits of information to be represented at once [4].

In radio transmissions, the coded symbol can be represented in its two components. The in-phase (real) component and the quadrature (imaginary) component must be transmitted in the same signal. This is done through using two orthogonal signals, one cosine and one sine. The communications channel over a wireless medium has a carrier frequency, f_c , and a bandwidth, B , where $B \ll f_c$. The transmitted radio signal can be represented by the equation

$$v_x(t) = A(t) * \cos(2\pi f_c t + \phi(t)) \quad (1)$$

where v_x represents the voltage which goes to the antenna, $A(t)$ is the amplitude function, and $\phi(t)$ is a phase function [6]. By using trigonometric identities, (1) becomes

$$v_x(t) = A(t) * \cos(\phi(t)) * \cos(2\pi f_c t) - A(t) * \sin(\phi(t)) * \sin(2\pi f_c t) \quad (2)$$

At this point, there are two orthogonal signals, which are represented together in one signal. The two orthogonal signals, represented by the phase function, allow both in-phase and quadrature components of the complex symbol x to be present and encoded in the same radio signal. The in-phase component $x_i(t)$ modulates the cosine part of the carrier, and the quadrature component $x_q(t)$ modulates the sine

part of the carrier. Because they are orthogonal in phase, they can coincide without interfering [6]. Since these numbers are complex, the representation of the complete symbol x is

$$x(t) = x_i(t) + ix_q(t) \quad (3)$$

Modifying (2) to be represented using Euler's formula for complex numbers, $e^{i\theta} = \cos(\theta) + i * \sin(\theta)$, provides the final form of the voltage to the antenna as

$$v_x(t) = \sqrt{2}(x(t) * e^{i2\pi f_c t}) \quad (4)$$

where the $\sqrt{2}$ factor is derived from splitting the components of the signal [7]. This represents the final voltage function for transmitting an encoded complex number through a carrier frequency.

This process is how two signals, in-phase and quadrature, can overlap in time and frequency, but remain orthogonal in phase, and thus be carried on the same signal. If the individual in-phase and quadrature signals are of a bandwidth $B/2$, the process of converting them into the components which are added together in the final signal, preserves the final bandwidth of B in the carrier. Figure 1 demonstrates a block diagram of an upconverter which performs this combination. The components of the signal are multiplied by the orthogonally phased carrier signal, and added together to produce the final output voltage. Figure 2 shows the signals in frequency domain through the process, demonstrating how the two components are encoded with the carrier, and combined into the final signal. The faded portions of the graphs represent the mathematically present negative frequencies, but which do not have any physical reality in the system [6] [7].

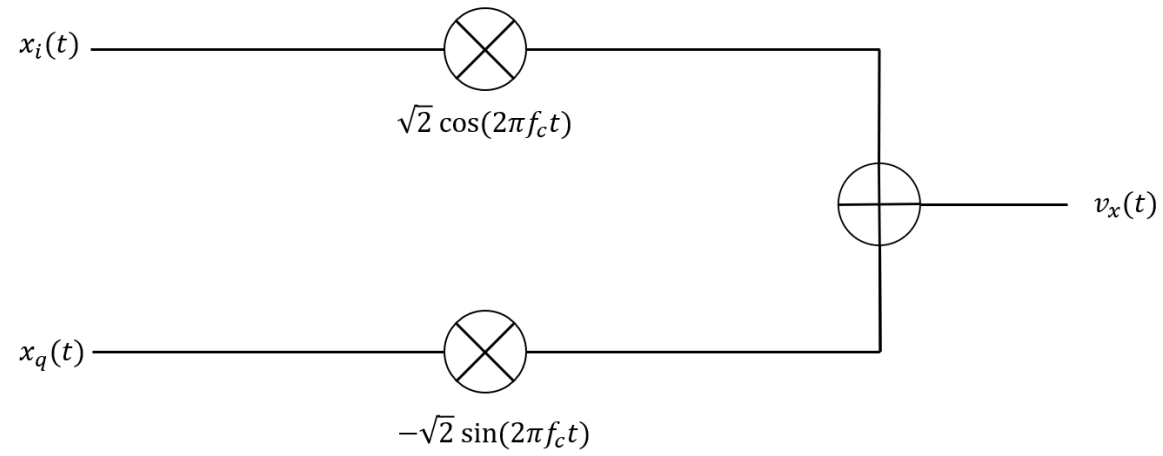


Figure 1: An upconverter block diagram for combining in-phase and quadrature components into one carrier signal [6] [7].

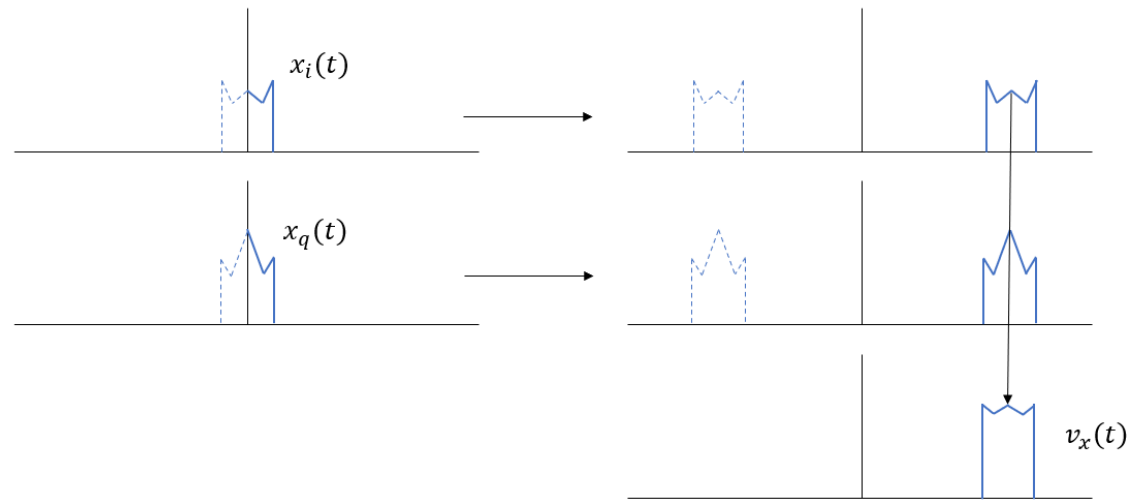


Figure 2: The in-phase and quadrature components in the frequency domain, converted and combined into a final signal at the carrier frequency, as performed by the upconverter in Figure 1 [7].

The Nyquist-Shannon Sampling Theorem allows that so long as the sampling is more than twice the frequency of the carrier signal, the signal may be abstracted from a continuous function to a discrete one, and a discretely sampled function can recover the original continuous function. This allows the symbols transmitted to be represented as a series of n discrete complex numbers, $x[0]...x[n - 1]$, instead of as functions. This sequence of symbols represents the data transmitted over time [7]. Because of the error caused by imprecise transmission in the wireless medium, when the receiver receives a signal and locates it on the complex plane, it decodes it to the symbol which is closest in the constellation to what was received. An example constellation using 16-QAM coding, and is provided in Figure 3. The figure shows how each possible symbol is encoded, and a received value is plotted and decoded. Because the received symbol is dependent on its proximity to the specified signals in the constellation, it may be mathematically represented as a random variable [8]. The effect of transmission on this random variable is described by the equation

$$y = Hx + n \tag{5}$$

where y is the received symbol (as a complex number), x is the transmitted symbol (as a complex number), and the effects of the wireless medium are modeled as H , with noise modeled as n .

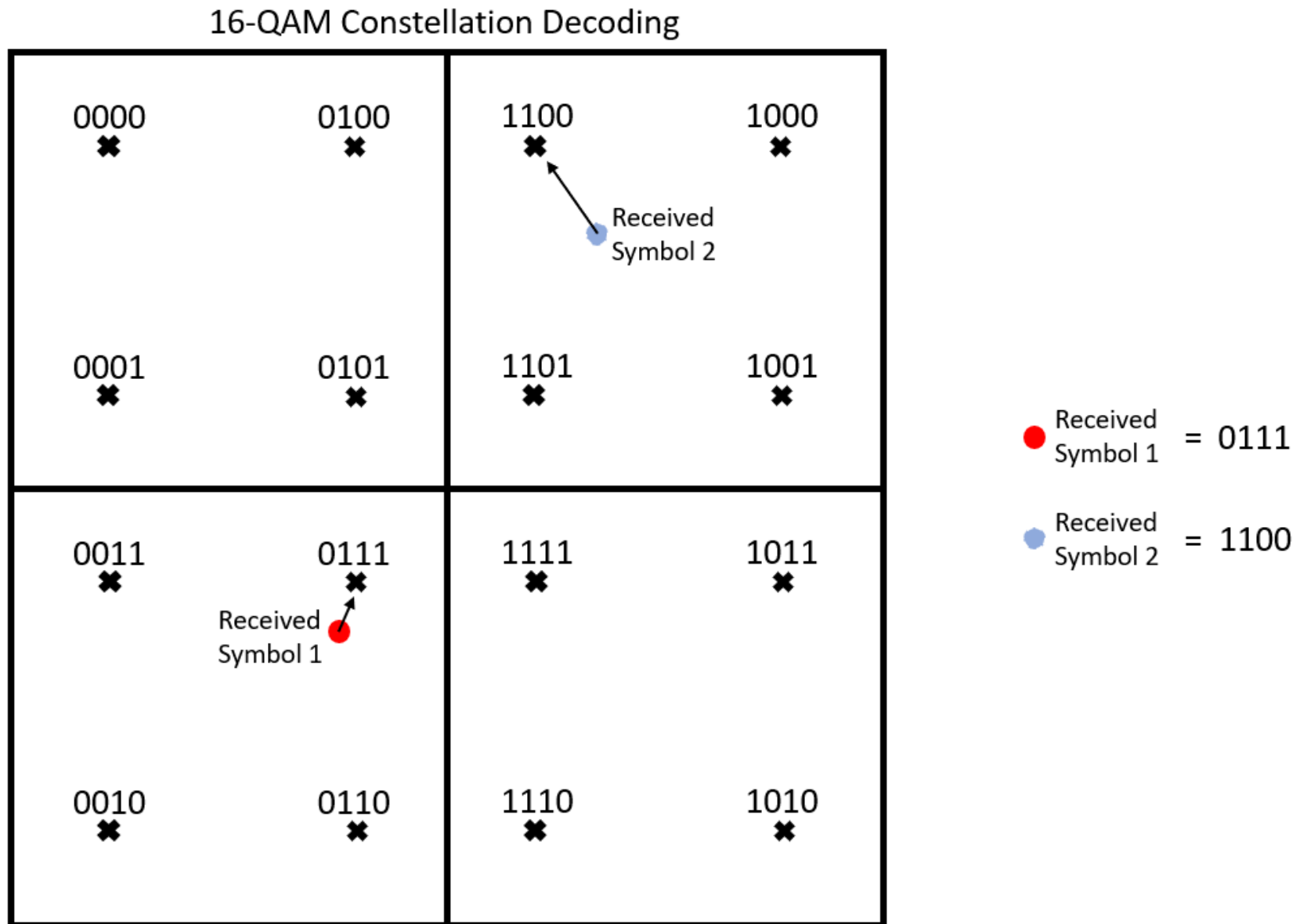


Figure 3: A 16-QAM Constellation is displayed, with two received symbols plotted and decoded. Each received symbol is decoded as the symbol on the constellation to which it is closest.

This model is adapted to MIMO communications, with multiple antennae, by turning the symbols into vectors to represent individual signals at each antenna. This means that the vector has size $N_{antennae}$, the number of antennae in the MIMO. H becomes a matrix, known as the channel matrix. The channel matrix describes the state of the wireless environment, and how it affects the signal between transmission and reception. It is also known as the Channel State Information (CSI) [7].

In 802.11ac, the function of beamforming is to provide a modifier to the signal that mitigates the effects of H on the transmitted signal, and allows y to be closer in value to x . The consequential reduction of error allows more precise data to be transmitted, and therefore allows the use of higher density coding schemes. A higher density coding scheme allows for more data to be transmitted at once by transmitting more bits per symbol. Figure 4 demonstrates the comparison between non-beamformed and beamformed constellations, simulated by MATLAB's example code for 802.11ac beamforming [9]. The result of beamforming is that the error rate is significantly reduced, because the received symbols are more closely clustered around the locations for each in the constellation. This has a twofold effect: the reduced error rate increases data rate, and the closer clustering allows a denser MCS to be used, which also increases the data rate.

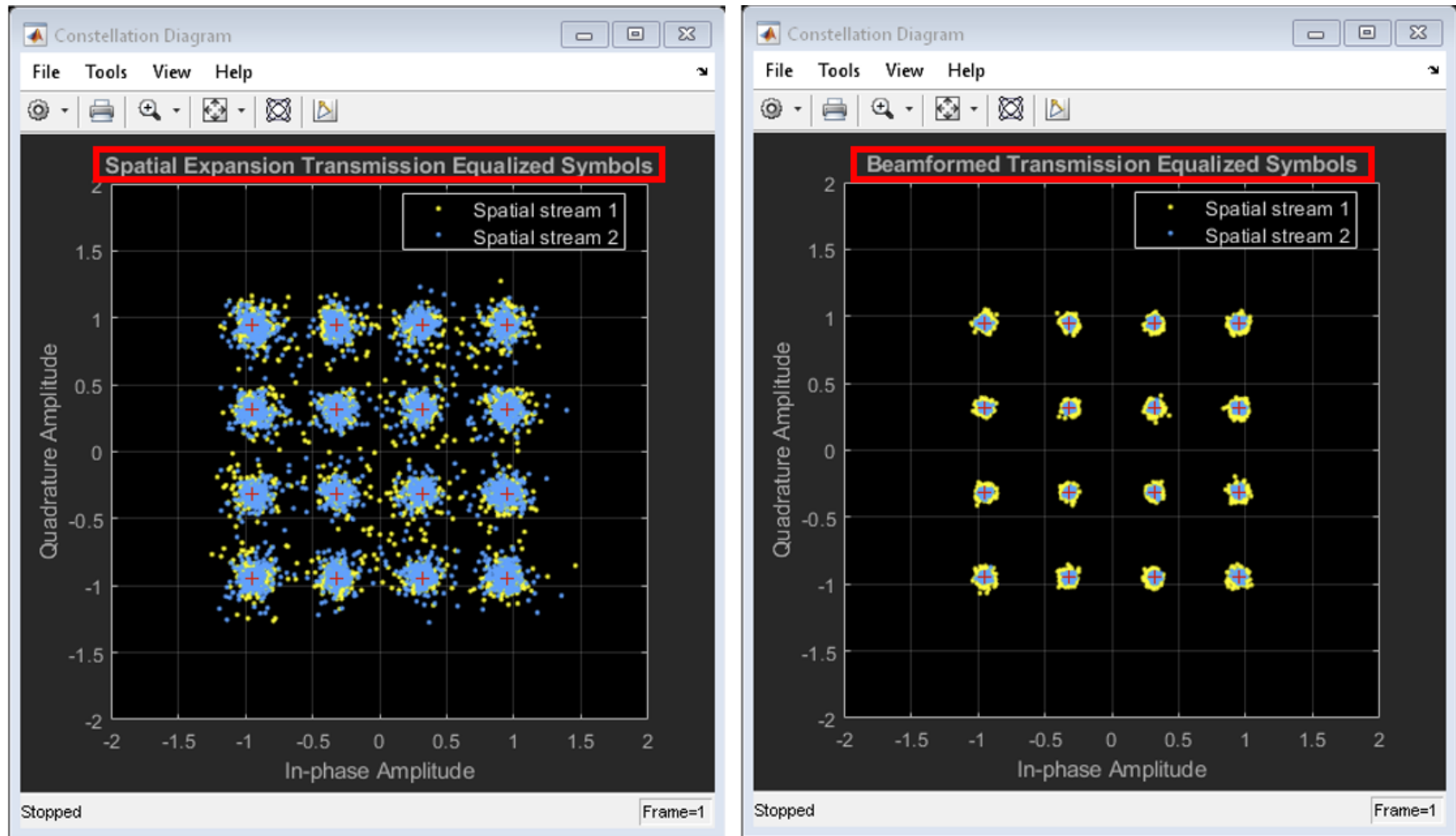


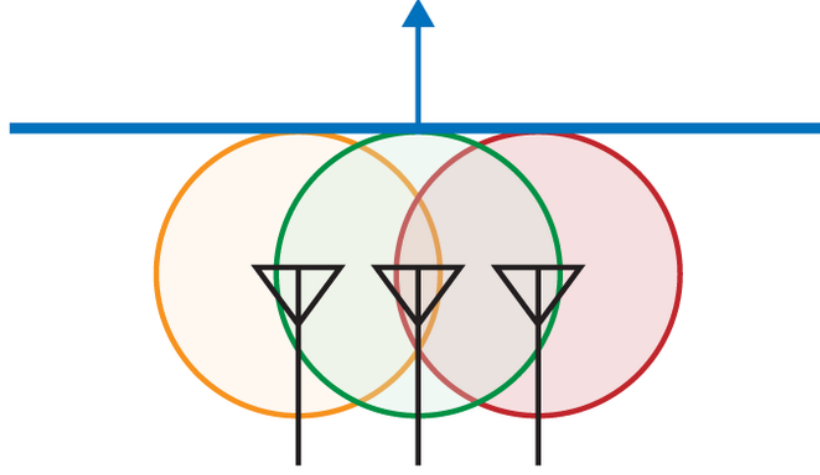
Figure 4: The MATLAB simulation suite for 802.11ac beamforming displays the differences in error by plotting the constellation for a non-beamformed, and then a beamformed transmission. The left plot demonstrates the non-beamformed transmission, and the right plot demonstrates the beamformed transmission. The clustering of received symbols in the non-beamformed transmission is much further apart than the beamformed one. The closer clustering of received symbols in the beamformed transmission will have a significantly lower error rate, and opens the potential to adopt a more dense coding scheme like 64 QAM, allowing it to transmit more data [9].

Figure 5 demonstrates how the physical mechanism for beamforming is to induce signal delays between antennae. These delays allow the angle of the transmission to be focused towards a specific target. However, the mathematical method by which this is accomplished is not through directly calculating or performing signal transmission delays. Instead, the signals that the antennae themselves transmit are modified prior to transmission with minor phase changes, in order to accomplish the same effect. The input voltage functions for each antenna are independently changed before transmission. This modification is done through what is known as a steering matrix, and also called the precoding matrix. Precoding is a process where the effects of the channel matrix H are countered by the interpolation of another matrix, F , to multiply the transmitted signal x by before it is transmitted. The new equation for modeling a beamformed system is

$$y = HFx + n \tag{6}$$

F may be subjected to singular value decomposition (SVD) and becomes three independent component matrices: the mixing matrix, the power allocation matrix, and the steering matrix. The final one, the steering matrix, is most directly responsible for beamforming, while the others are responsible for other modifications of the signal, such as antenna transmit power. When beamforming is done with line of sight (LOS) between linear arrays of antennae, the elements of the steering matrix are directly mathematically related to the angles between the transmitting and receiving arrays, individually describing each transmitting antenna. This means that the steering matrix in LOS situations will contain direct information about the physical layout of the system it is modifying. When there is not LOS, the specific multipath solutions which are used to accomplish the same task are still physically significant and meaningful in the system despite being too complex to represent in the same

(a) No steering matrix applied



(b) Steering matrix

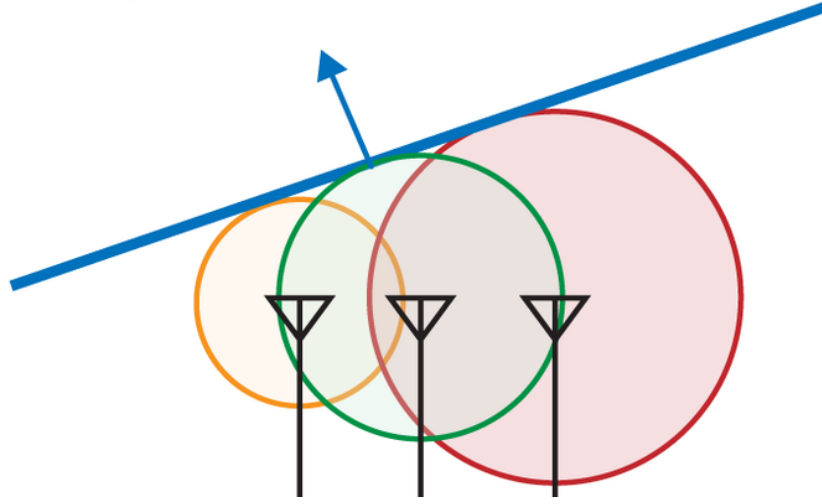


Figure 5: This image displays the physical effects and method of beamforming, by performing transmission delays that allow the wave to be directed in a particular way, instead of omnidirectionally. In the example shown, the leftmost antenna has the most delay, and the rightmost antenna transmits earliest. This lets the signals from each antenna overlap due to constructive interference, which improves the received SNR for a client which resides in the specified direction [4].

direct manner. In either case, the calculation of a suitable steering matrix depends on some level of knowledge of the CSI for the receiver. Therefore, the CSI must

be either implicitly estimated by the transmitter, or it must be explicitly calculated and given to the transmitter by the receiver, so that it may be used to calculate the steering matrix.

The consequence of this relation and calculation, is that while the physical arrangement of the systems is not directly described in the feedback matrices, and often not in the steering matrices, there is in fact a way in which both correlate to and reflect the physical arrangement of the systems. That is, the steering and feedback matrices both represent physical information about the systems.

2.2.3.2 802.11ac Process of Beamforming

The process which is specified in 802.11ac to perform beamforming is specified in Chapter 10 of the standard with exact details specified in Chapter 21, and is specified as the Very-High-Throughput (VHT) sounding protocol. The first step of the process is the AP sending out a VHT Null Data Packet (NDP) with the intention of channel sounding. This sounding process uses a known, pre-shared signal, sent to all STAs it is communicating with. It provides a measure for each STA to compare the transmitted x , as the pre-shared value, with the received y , and from that to determine an estimate of the CSI between them [1].

This CSI estimate is compressed into the feedback matrix by the STA. It is then transmitted from the STA back to the AP. Once the AP has an estimate of the CSI for the wireless channel, H in (5), it can use it to calculate the proper steering matrix and modify its transmitted signals accordingly. All subsequent data is beamformed with that steering matrix until the AP sounds the channel again to determine a new steering matrix [4].

2.2.3.3 802.11ac Representation of Beamforming

IEEE 802.11 allows for a variety of channel bandwidths, and the possible values for VHT MIMO management packets are described in Chapters 9, 19, and 21 of the standard. When an AP requests feedback from a STA, it requests an individual feedback matrix for each subcarrier that can be used. Subcarriers represent sidebands within the carrier signal, negative labels referring to sidebands below the carrier frequency, and positive labels referring to sidebands above the carrier frequency, both within the bandwidth. The label refers to the index of the subcarrier relative to the original carrier frequency. Figure 6 shows how a variety of different bandwidths and standards within IEEE 802.11 align subcarriers, as well as denoting which subcarriers are pilot subcarriers by the downward dips. At 80 MHz bandwidths, the subcarriers labeled -122 to -2, and 2 to 122 are evaluated and sent back. The total amount of subcarriers represented is 234, with those labeled 11, 39, 75, and 103 excluded in both the positive and negative sidebands, because in 80 MHz channels, these subcarriers are pilot subcarriers and do not transmit data. The beamforming feedback matrix for a subcarrier is labeled V_k in the standard, where k is the number of the subcarrier being described. The number of rows in each matrix is the number of spatial streams used in the signal. The elements of the feedback matrix are compressed using an angle form. The full feedback matrix is then reconstructed according to decoding and decompression equations, once it reaches the AP [1] [4].

The two angles, ϕ and ψ , are given unique bit sizes depending on if the feedback is for MU beamforming, or for SU beamforming, and depending on which codebook is selected. The codebook is the format in which the data is represented, and IEEE 802.11 provides 2 options: codebook 0 or codebook 1. This provides 4 different options for angle bit sizes in the standard. Both of these pieces of information are given in the MIMO Control Field section of the feedback frame, and Figure 7 shows

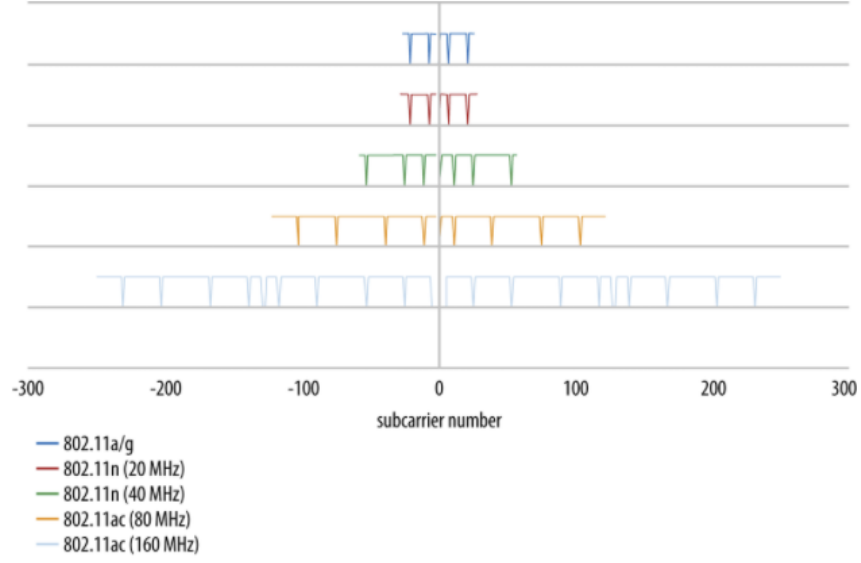


Figure 6: This image displays the alignment of subcarrier numbers, and pilot carriers, in different IEEE 802.11 bandwidth sizes and standards. The downward dips denote pilot carriers. [4].

their effect on the size of the angles. In the situations investigated by this work, SU beamforming and codebook 1 are used, causing bit sizes of 4 for any ψ angles, and 6 for any ϕ angles [1].

Table 9-66—Subfields of the VHT MIMO Control field *(continued)*

Subfield	Description
Codebook Information	Indicates the size of codebook entries: If Feedback Type is SU: Set to 0 for 2 bits for ψ, 4 bits for ϕ Set to 1 for 4 bits for ψ, 6 bits for ϕ If Feedback Type is MU: Set to 0 for 5 bits for ψ, 7 bits for ϕ Set to 1 for 7 bits for ψ, 9 bits for ϕ
Feedback Type	Indicates the feedback type: Set to 0 for SU Set to 1 for MU
Remaining Feedback Segments	Indicates the number of remaining feedback segments for the associated VHT Compressed Beamforming frame: Set to 0 for the last feedback segment of a segmented report or the only feedback segment of an unsegmented report. Set to a value between 1 and 6 for a feedback segment that is neither the first nor the last of a segmented report. Set to a value between 1 and 7 for a feedback segment that is not the last feedback segment of a segmented report. In a retransmitted feedback segment, the field is set to the same value associated with the feedback segment in the original transmission.
First Feedback Segment	Set to 1 for the first feedback segment of a segmented report or the only feedback segment of an unsegmented report; set to 0 if it is not the first feedback segment or if the VHT Compressed Beamforming Report field and MU Exclusive Beamforming Report field are not present in the frame. In a retransmitted feedback segment, the field is set to the same value associated with the feedback segment in the original transmission.
Sounding Dialog Token Number	The sounding dialog token from the VHT NDP Announcement frame soliciting feedback

Figure 7: The MIMO control field's effect on the bit size of ϕ and ψ [1].

The quantization of ψ angles is decoded by the equation

$$\psi = \frac{k\pi}{2^{b_\psi+1}} + \frac{\pi}{2^{b_\psi+2}} \quad (7)$$

where $k = 0, 1, \dots, 2^{b_\psi} - 1$ and b_ψ is the number of bits used to quantize ψ .

The equation for decoding the quantized form of ϕ angles is similar, and is

$$\phi = \frac{k\pi}{2^{b_\phi-1}} + \frac{\pi}{2^{b_\phi}} \quad (8)$$

Both decoded quantities are in radians, as described in Table-9-68 from the standard [1]. The numbers of bits are used in determining the size of the compressed beamforming feedback matrix for each subcarrier, according to the formula

$$N_a * (b_\psi + b_\phi) / 2 \quad (9)$$

where N_a is the number of angles used in each feedback matrix. Figure 8 displays Table 9-67 from the standard, which determines the number of angles in each feedback matrix, as determined by the number of rows and columns [1]. The number of rows in the matrix is the same as the number of spatial streams, N_{STS} , which is indicated by the NDP sent from the AP for channel sounding, and the number of columns is less than or equal to the minimum between N_{STS} and the number of receiving antennae. For the test cases analyzed in this work, $N_r = 2$ for two spatial streams used, and $N_c = 2$ for the two antennae used. In this case, the number of angles in each compressed feedback matrix is 2, and they are ϕ_{11} and ψ_{21} . In the compressed format, all that is provided is the two angles, not a full 2x2 matrix. The decompression, done

by the AP, follows the equation

$$V = \left[\prod_{i=1}^{\min(N_c, N_r-1)} [D_i(1_{i-1}, e^{i\phi_{i,i}}, \dots, e^{i\phi_{N_r-1,i}}, 1) \prod_{l=i+1}^{N_r} G_{li}^t(\psi_{li})] \right] I_{N_r \times N_c} \quad (10)$$

where the matrix D is a diagonal matrix, where the first element 1_{i-1} representing a sequence of 1s, such as

$$D_i = \begin{bmatrix} I_{i-1} & 0 & 0 & 0 & 0 \\ 0 & e^{i\phi_{i,i}} & 0 & 0 & 0 \\ 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 0 & e^{i\phi_{N_r-1,i}} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (11)$$

. The matrix G is an $N_r \times N_r$ Givens rotation matrix which has the form

$$G_{li}(\psi) = \begin{bmatrix} I_{i-1} & 0 & 0 & 0 & 0 \\ 0 & \cos(\psi) & 0 & \sin(\psi) & 0 \\ 0 & 0 & I_{l-i-1} & 0 & 0 \\ 0 & -\sin(\psi) & 0 & \cos(\psi) & 0 \\ 0 & 0 & 0 & 0 & I_{N_r-l} \end{bmatrix} \quad (12)$$

which are described in the standard as Equations 19-79, 19-80, and 19-81 [1].

Table 9-67—Order of angles in the Compressed Beamforming Feedback Matrix subfield

Size of V ($N_r \times N_c$)	Number of angles (N_a)	The order of angles in the Compressed Beamforming Feedback Matrix subfield
2×1	2	ϕ_{11}, ψ_{21}
2×2	2	ϕ_{11}, ψ_{21}
3×1	4	$\phi_{11}, \phi_{21}, \psi_{21}, \psi_{31}$
3×2	6	$\phi_{11}, \phi_{21}, \psi_{21}, \psi_{31}, \phi_{22}, \psi_{32}$
3×3	6	$\phi_{11}, \phi_{21}, \psi_{21}, \psi_{31}, \phi_{22}, \psi_{32}$
4×1	6	$\phi_{11}, \phi_{21}, \phi_{31}, \psi_{21}, \psi_{31}, \psi_{41}$
4×2	10	$\phi_{11}, \phi_{21}, \phi_{31}, \psi_{21}, \psi_{31}, \psi_{41}, \phi_{22}, \phi_{32}, \psi_{32}, \psi_{42}$
4×3	12	$\phi_{11}, \phi_{21}, \phi_{31}, \psi_{21}, \psi_{31}, \psi_{41}, \phi_{22}, \phi_{32}, \psi_{32}, \psi_{42}, \phi_{33}, \psi_{43}$
4×4	12	$\phi_{11}, \phi_{21}, \phi_{31}, \psi_{21}, \psi_{31}, \psi_{41}, \phi_{22}, \phi_{32}, \psi_{32}, \psi_{42}, \phi_{33}, \psi_{43}$
5×1	8	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}$
5×2	14	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}, \phi_{22}, \phi_{32}, \phi_{42}, \psi_{32}, \psi_{42}, \psi_{52}$
5×3	18	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}, \phi_{22}, \phi_{32}, \phi_{42}, \psi_{32}, \psi_{42}, \psi_{52}, \phi_{33}, \phi_{43}, \psi_{43}, \psi_{53}$
5×4	20	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}, \phi_{22}, \phi_{32}, \phi_{42}, \psi_{32}, \psi_{42}, \psi_{52}, \phi_{33}, \phi_{43}, \psi_{43}, \psi_{53}, \phi_{44}, \psi_{54}$
5×5	20	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}, \phi_{22}, \phi_{32}, \phi_{42}, \psi_{32}, \psi_{42}, \psi_{52}, \phi_{33}, \phi_{43}, \psi_{43}, \psi_{53}, \phi_{44}, \psi_{54}$
6×1	10	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \phi_{51}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}, \psi_{61}$
6×2	18	$\phi_{11}, \phi_{21}, \phi_{31}, \phi_{41}, \phi_{51}, \psi_{21}, \psi_{31}, \psi_{41}, \psi_{51}, \psi_{61}, \phi_{22}, \phi_{32}, \phi_{42}, \phi_{52}, \psi_{32}, \psi_{42}, \psi_{52}, \psi_{62}$

Figure 8: These are the first entries of Table 9-67 from the 802.11 standard, which describes the angles which are stored in various sizes of feedback matrices. The feedback matrix is given in a compressed form that consists of only the angles in the order given, while the actual feedback matrix size is shown in the size column. This work uses a constant size of 2x2, and thus has the smallest feedback matrix available in the standard [1] [4].

In the cases investigated in this work, the decompression of the matrices is simplified to the formula

$$V = D_1 * G_{21}(\psi_{21}) * I_{2 \times 2} \quad (13)$$

which is:

$$V = \begin{bmatrix} e^{i\phi_{11}} & 0 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} \cos(\psi_{21}) & \sin(\psi_{21}) \\ -\sin(\psi_{21}) & \cos(\psi_{21}) \end{bmatrix} * \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (14)$$

These are the same two angles which are provided in the compressed feedback matrix format for each subcarrier in the report sent back from the STA. Once obtained, decompressed matrix is

$$V = \begin{bmatrix} e^{i\phi_{11}} * \cos(\psi) & e^{i\phi_{11}} * \sin(\psi) \\ -\sin(\psi) & \cos(\psi) \end{bmatrix} \quad (15)$$

The final step of the decompression is substituting the values of the angles according to the decoding formulae specified in (7) and (8). The number from the compressed form for each angle is substituted as the value of k in the decoding formulae, and this completes the decompression of the feedback matrix [1].

2.3 Related Work

This section surveys related work which provides the context and basis of this research. Section 2.3.1 examines recent work in exploiting information leakage, and demonstrating its danger as a security threat. Next, Section 2.3.2 examines the application of information leakage to target the use of Wi-Fi networks and perform pattern-of-life modeling. Finally, Section 2.3.3, Section 2.3.4, and Section 2.3.5 discuss recent methods and applications of sensing based on wireless signals.

2.3.1 Information Leakage as a Security Concern

Information leakage is a serious threat to security. A variety of attacks utilize the principle of information leakage to violate security, including the recent SPECTRE [10] and MELTDOWN [11] attacks on CPU architecture. These use artifacts of information storage in memory access speeds to leak secrets from otherwise inaccessible memory. The method of doing this allows the security permissions of different areas of memory to be bypassed. Other recent work addresses the potential of information leakage through the use of cloud computing platforms and storage systems [12] [13] [14]. Because of sharing the same hardware, there are artifacts left from operations and information that can allow patterns to be observed and exploited for revealing otherwise secret and protected information. The principle of information leakage is that information leaves behind effects and artifacts which allow it to be deduced.

Information leakage may also be a consequence of traveling. [15] demonstrates that through open networks and unsecured information such as Domain Name System (DNS) requests, 68% of a target's communications could be leaked. [16] uses a side channel through Wi-Fi CSI to determine phone passwords.

Information leakage is not only a vulnerability, but a designed and intentional feature in modern mobile phones, as demonstrated in [17]. A recent survey [18] discusses the ecosystem of marketed use of intentional information leakage for the purpose of targeted advertisement, which has come under criticism for its potentially harmful effects [19].

2.3.2 Pattern-of-Life Modeling with 802.11

Wi-Fi is a peculiarly rich setting for exploiting information leakage. Smart home architecture, and increasingly capable Internet of Things (IoT) devices are growing in use, and with this growth is corresponding growth in attention to the risk involved

in these systems [20]. Especially concerning is the lack of attention to security in designing many of these systems, which is highlighted in the August 2020 Government Accountability Office (GAO) report on IoT use by federal agencies [21].

[22] has recently succeeded in using observation and classification of Wi-Fi events to train a system to perform pattern-of-life analysis, and demonstrated the ability to break into a home by utilizing the pattern analysis to determine when a resident is away. The method proposed relies on classifying devices and events, and identifies the devices and patterns with MAC addresses, which are visible and unencrypted even in secured 802.11 networks. A mitigation system was also designed to prevent pattern-of-life analysis, which consists in spoofing false devices and deceiving the analysis tools into classifying false events, and failing to classify true ones. This research was expanded more recently by [23] in through utilizing artificial intelligence methods to enhance the analysis techniques.

2.3.3 Sensing for Location

Sensing research has a long-standing interest in determining device location through analyzing wireless signals. One method is using directional antennae to locate access points and devices [24] [25] [26]. These projects have run into the challenge of multi-path signals, which cause a lack of precision when attempting to locate indoor targets. There has been some success in using directional antennae to locate rouge wireless STAs [27], but as the wireless environment becomes more congested, these successes fade. More recently, a method has been devised to use 3-antenna routers to determine device locations based on identifying angles of arrival for wireless signals [28], and it promises to overcome some of the complexities in indoor environments.

2.3.4 Sensing for Device Identity

Another method of wireless sensing identifies unique transmitters, by training systems to compare subtle differences in signal characteristics between different devices. This forms the area of device fingerprinting. [29] provides a method called "RF-DNA" to detect small variations in devices, and identify them. The technique has been extended to the Zigbee protocol, [30] [31], and has potential to identify unique devices for other wireless protocols, by using the physical characteristics of the signals, instead of information encoded in the data-link layer. [22] used the unencrypted MAC addresses in the data-link layer of 802.11 to detect and identify distinct devices, but by using only data-link layer data, a spoofed and real device could not be distinguished.

2.3.5 Sensing for System Characteristics

An application of sensing for information about features independent of the wireless environment has focused on how other entities and events affect the wireless environment. Therefore, by investigating these affects, sensing the wireless environment can disclose information about the non-wireless environment. A recent survey, [32], discusses diverse ways in which CSI may be used to perform device detection, motion analysis, and human behavior analysis. It also discusses various approaches in sensing for determining what the CSI is. [33] additionally develops a method of patching some Broadcom wireless drivers to provide a way of measuring CSI information on ordinary commercially available hardware, such as Raspberry Pis and smartphones. Additionally, work has been done by [34] to investigate the use of dedicated sensor configurations in detecting specific motions such as walking, sitting, and falling by training and analyzing the character of Wi-Fi CSI over time.

2.4 This Research in Relation to Previous Research

This research connects the areas of sensing and information leakage, by taking advantage of the CSI information already calculated and transmitted in the clear by 802.11ac’s beamforming capabilities. It uses this CSI information to determine unique device locations on the network, and to determine if the environment motionless or motion-saturated, by observing the effects of environmental factors on the captured CSI. This provides an easy method of accessing Wi-Fi CSI from multiple sources, without dedicated detectors, complex algorithms, or modified hardware and firmware. It also provides a method to correlate device location on a Wi-Fi network as a possible solution to defensive spoofing methods which prevent higher-level pattern-of-life analysis based on device classification. Table 1 demonstrates a comparison between previous research, and how this research brings together different topics and techniques to unify the different areas. This research uses the beamforming feedback reports encoded in the data-link layer, which contain the CSI for devices on an 802.11ac network, to perform device and event sensing. The encoding of this information, which is intended to describe the physical layer characteristics of the network, allows for physical information to be captured and identified without performing direct physical sensing, or using dedicated sensors. Through the leaked CSI, pattern detection is performed by training on the CSI characteristics in different scenarios based on device location, and motion in the environment.

Table 1: A comparison of previous research contributions.

Research	Device Sensing	Event Sensing	Information Leaks	Pattern Detection	CSI	802.11ac	Beamforming
Cheng (2013) [15]			X	X		X	
Kotaru (2015) [29]	X					X	
Li (2016) [16]		X	X		X	X	
Lewis (2017) [29]	X						
Law (2018) [24]	X	X				X	
Shculz (2018) [33]		X			X	X	
Beyer (2018) [22]		X	X	X		X	
Aragon (2019) [23]		X	X	X		X	
Ma (2019) [32]	X	X	X	X	X		
Cruz (2019) [30]	X						
Park (2019) [20]			X			X	
Matsui (2020) [31]	X						
Ullah (2020) [18]			X	X			
Taglieri (2021)	X	X	X	X	X	X	X

III. Design

3.1 Overview

This chapter describes the design of the System Under Test (SUT). First, Section 3.2 describes the devices and their configuration which compose the test network environment. Section 3.3 describes the design and components of the FPX tool, which is used to transform a packet capture of feedback data into mathematical form. Finally, Section 3.4 describes the design and components of the FCaE, which contains the Components Under Test (CUTs) in this research, and which is designed to explore and analyze feedback data.

3.2 System Components

3.2.1 ASUS AC1300 RT-ACRH13

The ASUS RT-ACRH13 router was chosen, because it contains the first 802.11ac wave 2 system on a chip (SoC) commercially available, the Qualcomm IPQ4019 chip [35] [36]. In possessing wave 2 capabilities, it is able to perform transmit beamforming and MU-MIMO, which are required capabilities for wave 2 802.11ac devices. Therefore it has all of the necessary features to allow observation of 802.11ac beamforming in its network [4].

The router is configured to run a 5 GHz Wi-Fi network (under the name “LexRex5”), with WPA2 security designed to use a pre-shared key. This is the highest level of security commonly available in Wi-Fi systems. The purpose of this security setting is to certify the presence of unencrypted information leakage through the beamforming feedback mechanism, despite a high level of security protecting and encrypting the actual network traffic itself.

Further configuration includes enabling explicit transmit beamforming on the router, and disabling implicit transmit beamforming. Without these choices, the router is not guaranteed to make use of the capability that it has. With this configuration, it attempts to use channel sounding to perform beamforming with beamformee-capable clients. To standardize the experiments, the channel width is limited to 80 MHz channels. As a consequence of this limit, the subcarriers evaluated in beamforming feedback are specifically in order from index labels -122 to -2, and 2 to 122, a total of 234 subcarriers are given feedback in every feedback report, each subcarrier having its own feedback matrix, as discussed in Section 2.2.3.3. While this is not necessary for beamforming itself to function, it allows for uniform values that can be correlated and compared for each trial, and enables the FPX and FCaE to utilize simpler mathematical analysis.

3.2.2 Dell XPS 15 7590

The Dell Laptop is used as the client for all test scenarios. It has attached to it two USB-connected Alfa AWUS036ACH adapters, designed as external wireless network interface cards (NIC). This NIC is 802.11ac capable, and may act as a beamformee. The adapter uses a Realtek chipset and driver. The reason for selecting an external adapter was to achieve finer control over the physical positioning of the client device. The only component that needs to be moved or maintained in position is the adapter, and the exact location of it is clear and visible, and therefore precisely measurable, unlike in the the case of an internal wireless adapter.

Due to the presence of an internal adapter on the laptop, this test adapter is designated on Windows as “Wi-Fi 2.” The test adapter has the MAC address 00:C0:CA:AD:09:87. It is used to connect the PC to the LexRex5 Wi-Fi network, so that it can transmit beamforming feedback reports to the router when the router

performs channel sounding.

3.2.3 Kali Virtual Machine

The attack platform is a Virtual Machine (VM) running the Kali operating system through Virtual Box. The reasons to select Kali as the attack platform are for its convenience and prevalence. It also contains the required drivers in its repositories, which allow it to use the Alfa NIC in monitor mode and listen to wireless communications. It uses the same model adapter as the Dell PC, to guarantee that it is able to receive and decode the traffic which passes between the Dell PC and the router. Because the adapter and driver are equipped for single user beamforming (SUBf), but not multi user beamforming (MUBf), and no driver equipped for MUBf is also equipped with monitor mode, the use of the same model adapter allows the monitoring of SUBf communications and feedback between the router and client. The MAC address of the monitor adapter is 00:C0:CA:AD:09:84.



Figure 9: The configuration of the test and monitor adapters together

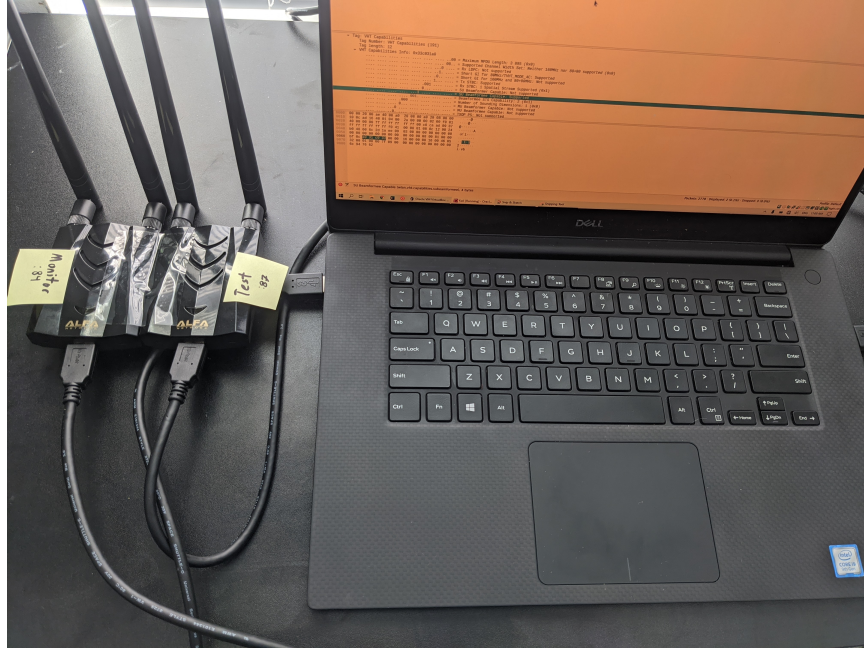


Figure 10: The final attack platform

To acquire the best quality data available during the experiment, the adapters are co-located, as demonstrated in Figure 9. This allows the almost certain guarantee that signals picked up and sent by the test adapter are also picked up by the monitor adapter. The close proximity of the adapters also increases the SNR of the feedback report's transmission for the monitor adapter, which minimizes the potential error in data collection resulting from the wireless environment.

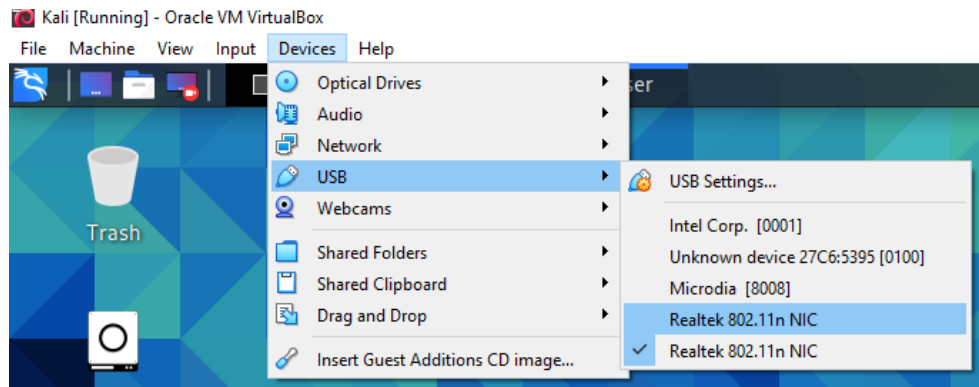


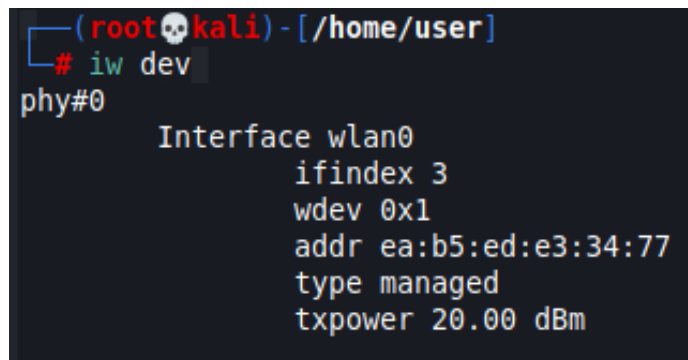
Figure 11: The second Alfa NIC is attached to the VM

The specific steps include first attaching the monitor adapter to the VM through the Virtual Box interface for attaching USB devices. This is shown in Figure 11. The adapter itself is labeled by Windows as an 802.11n NIC, and not an 802.11ac NIC, but this mislabelling does not affect the actual capabilities or operation of the NIC in using 802.11ac. The mislabelling of the NIC by the system is proven by its operation at 5 GHz frequencies. Because 802.11n does not support 5 GHz, and the NIC operates at 5 GHz, it follows that the NIC is not an 802.11n adapter, and is mislabelled by Windows in the figure.

The sequence of commands to set the adapter in monitor mode are shown from Figure 12 through Figure 16, the last of which displays the adapter in the configuration necessary for the experiment.

First, in Figure 12, the initial configuration is displayed which detail that the default state of the adapter is managed mode, which it would use to act as a STA in an 802.11 network. Next, in Figure 13 the mode of the adapter is set to monitor, which disables it from being a STA, but enables it to listen to all 802.11 signals and display them to the system, regardless of being connected to a network or not.

In Figure 14, the frequency of the adapter's operation is set to be within the 5 GHz band. The particular frequency of 5.805 GHz is chosen to match the observed



```
(root@kali) - [/home/user]
# iw dev
phy#0
    Interface wlan0
        ifindex 3
        wdev 0x1
        addr ea:b5:ed:e3:34:77
        type managed
        txpower 20.00 dBm
```

Figure 12: The initial configuration details of the adapter, showing that it is in managed mode by default

```
(root@kali) - [/home/user]
# iw wlan0 set monitor none
```

Figure 13: Setting the mode to monitor

frequency of the test network. Finally, the interface is made available by activating it in Figure 15.

The final configuration is confirmed, and displayed in Figure 16. An alternative procedure, in the event of this procedure's failure, is detailed in Appendix A, and may be necessary due to varying circumstances of virtual machine and hardware configurations.

```
(root@kali) - [/home/user]
# iw wlan0 set freq 5805
```

Figure 14: Setting the frequency to 5.805 GHz, which is the observed frequency of the LexRex5 test network

```
(root@kali) - [/home/user]
# ip link set wlan0 up
```

Figure 15: Activating the interface so that it becomes available for use by applications

```
(root@kali) - [/home/user]
# iwconfig wlan0
lo flags: no wireless extensions.
eth0 flags: no wireless extensions.
wlan0    unassociated  ESSID:""  Nickname:"<WIFI@REALTEK>"
         Mode:Monitor  Frequency=5.805 GHz  Access Point: Not-Associated
         Sensitivity:0/0
         Retry:off   RTS thr:off   Fragment thr:off
         Encryption key:off
         Power Management:off
         Link Quality=0/100  Signal level=0 dBm  Noise level=0 dBm
         Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
         Tx excessive retries:0  Invalid misc:0  Missed beacon:0
```

Figure 16: Confirming the final configuration of the adapter in monitor mode, in the 5 GHz band

3.3 Feedback Packet Extractor (FPX)

The Feedback Packet Extractor (FPX) is a multi-step system, composed of several scripts, which when executed in sequence allow a capture of beamforming feedback reports to be transformed and decompressed into a format usable by MATLAB R2020a for analysis. It is specialized to transform beamforming feedback matching the configuration used in this research of 80 MHz channel SU beamforming.

The requirements for the process are to transform the encoded feedback matrices in .pcap files from Wireshark sniffing into mathematical form in .mat files, which can be imported as data by MATLAB. This follows a series of individual transformations, as well as a series of scripts which perform the individual transformation step on multiple data sets which are collected by this research. The subsequent sections will clarify each component script's role in the process, as well as if it is an individual transformation or a collective one specific to this research.

The entire process may be visualized in the block diagram in Figure 17. It follows the steps:

1. **Export:** Wireshark is used to export beamforming feedback reports in C format. This is signified by the first block in Figure 17, and described in Section 3.3.1.
2. **Decode:** A combination of scripts decodes the beamforming feedback reports from their binary format, and produces the set of angles used in compressed feedback matrices (as described in Section 2.2.3.3), which are in MATLAB usable form. This is signified by the second block in Figure 17, and described in Section 3.3.2.
3. **Decompress:** A combination of MATLAB scripts decompresses the feedback matrices from angle form to numerical form, and produces a final matrix repre-

sentation of all feedback matrices from the original packet capture in MATLAB data format. This is signified by the final block in Figure 17, and described in Section 3.3.3.

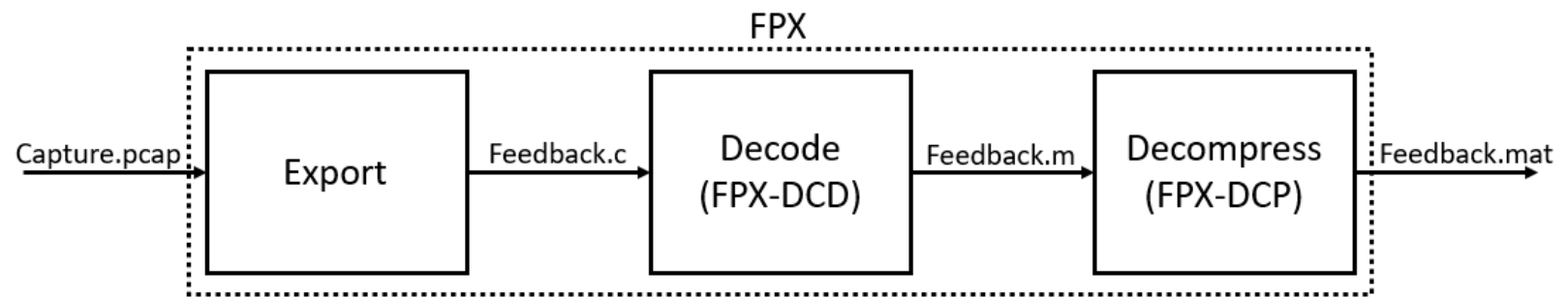


Figure 17: FPX Process Block Diagram

3.3.1 Export

No unique scripts are required for the exportation step. All the capabilities already exist in Wireshark’s exportation tools. It is assumed that the .pcap file contains only the beamforming feedback reports, and no other wireless traffic. The process by which this research gathers the specific .pcap file is described in Section 4.9. The possible options Wireshark offers for exportation are text, comma-separated-value (CSV), C, XML, and JSON formats. The choice of using the C format is to achieve direct access to the binary representation of feedback reports at the bit level, since the angle sizes used in this research do not fit byte boundaries.

Upon exportation, the C file produced is named for the dataset which it contains, and is then imported for use in the next step of the extraction process. Every packet is provided as a C array, named “pkt#” where its index in the capture is replaces the “#” sign. The content of each array is the raw binary representation of each byte in the packet.

3.3.2 Decode

The FPX Decoder (FPX-DCD) is the component of the FPX which transforms the encoded feedback matrix data within a feedback report packet into a set of compressed beamforming feedback angles. There are three individual scripts which are used in this step of the process. A Python script is utilized to produce a customized C program for each individual transformation, and a bash script is utilized to collectively run the Python script for all data collected by this research.

3.3.2.1 C Decoding Program

The C Decoding Program is what transforms the data from a .c file into a .m file. It consists of the same procedure applied to each individual feedback report

packet. Sample code for one program is included in Appendix B. The data from the Wireshark-exported .c file is captured using an “#include” directive, which makes the variables containing the packet data accessible to the program. This program is an individual transformation which must be applied to each collected data file.

The layout of the feedback matrices in the packet is visualized in Figure 18, which lines up the arrangement of bytes (top of figure) to each subcarrier and angle (bottom of figure). Each subcarrier has a total bit length of 10 bits, composed of ϕ_{11} and ψ_{11} , which are 6 and 4 bits respectively (as seen in Figure 19). Because a byte is 8 bits long, this means that the pattern repeats itself every 5 bytes, or every 4 subcarriers. Therefore, in order to successfully decode the angles from the packet data, the offset must be kept track of to determine the proper bits to decode. The C Decoding Program does this with a variable that is incremented and reset every 4 subcarriers, as it loops through every subcarrier in the report (indexed from 0 to 233). The feedback matrices start at the index 0x3F in the packet, and because the same channel configuration is kept for every scenario, this is constant across every packet collected in this research.

Upon properly decoding each angle, the program writes it to a .m file, formatted as entries of a MATLAB matrix. The format of the matrix is that every column represents an angle, and every row represents a subcarrier. Multiple packets are represented together in the same matrix by adding additional columns so that the final matrix ends with the number of rows equal to the number of subcarriers (234), and the number of columns equal to the angles multiplied by the amount of packets ($2*N_{packets}$). The generated .m file is concluded with the command to save the matrix to a .mat data file once executed in MATLAB, so that the data may be imported into other MATLAB scripts.

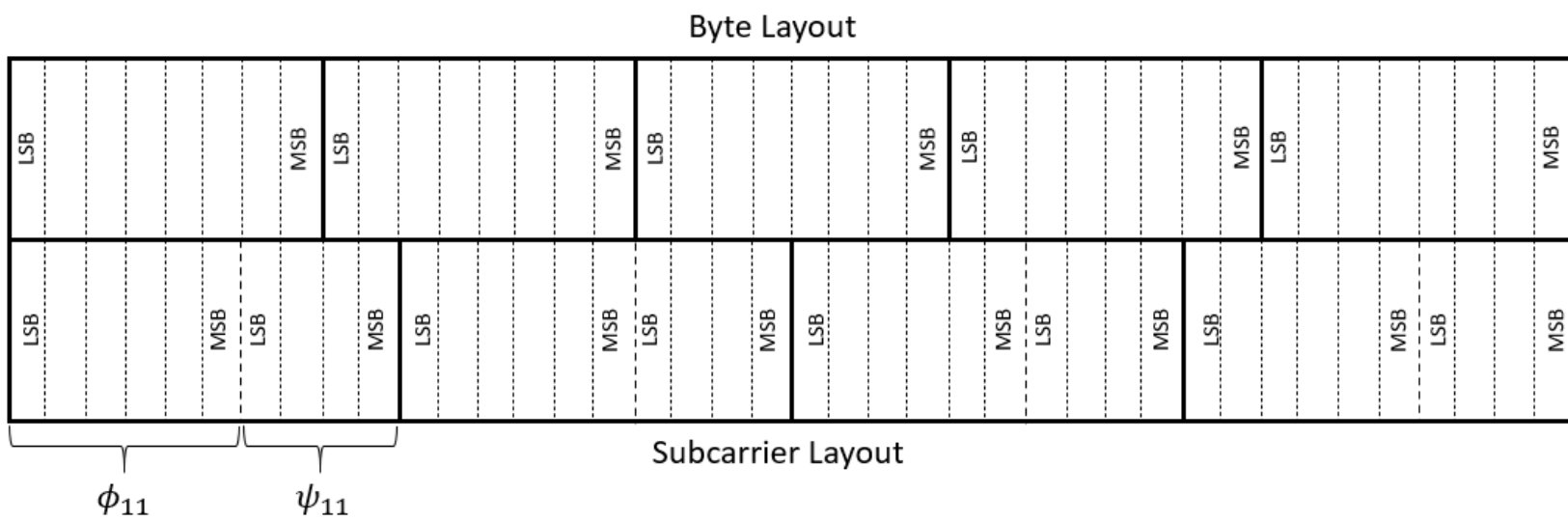


Figure 18: Feedback Decoding Layout

```

▼ VHT MIMO Control: 0x288489, Nc Index: 2 Columns, Nr Index: 2 Rows, Chann
    .... .001 = Nc Index: 2 Columns (0x1)
    .... ..00 1... = Nr Index: 2 Rows (0x1)
    .... 10.. .... = Channel Width: 80 MHz (0x2)
    .... 00 ..... = Grouping (Ng): 1 (No Grouping) (0x0)
    .... .1.. .... = Codebook Information: 0x1
    .... 0... .... = Feedback Type: SU (0x0)
    .... .000 .... = Remaining Feedback Segments: 0x0
    .... 1... .... = First Feedback Segments: 0x1
    .... ..00 .... = Reserved: 0x0
    0010 10.. .... = Sounding Dialog Token Number: 0x0a
▼ VHT Compressed Beamforming Report: 4547bffa7afbfef9ebaf7ff
  > Average Signal to Noise Ratio
  ▼ PHI and PSI Angle Decode
    PHI(6 bits):  PHI11: 47,
    PSI(4 bits):  PSI21: 15,

```

Figure 19: Angle Bit Sizes

3.3.2.2 Python Code Generator

The Python Code Generator is provided in Appendix C. This script is used to generate the customized version of the C Decoding Program described in Section 3.3.2.1, which is tuned to the specific Wireshark-exported .c file it is decoding. The generator takes two arguments: the name of the .c data file, and the number of packets contained in it. From this, it constructs the C Decoding Program by including the correct .c data file, and then generating the decoding procedure for each data packet variable individually. The name of the output .m file is configured to match the name of the input .c data file. The final command of the generator is the bash command which compiles and then runs the C Decoding Program. This script is individually applied to each data file.

3.3.2.3 Bash Script

The bash script in Appendix D is a collective program designed to run the Python Code Generator (Section 3.3.2.2) and C Decoding Program (Section 3.3.2.1) on each data file exported from Wireshark in this research. It decodes the first 100 packets from each capture, and performs the entire FPX-DCD portion of the process.

3.3.3 Decompress

The decompress section of the FPX process is performed by the FPX-DCP MATLAB script in Appendix E, with utility functions defined in Appendix F. This script performs the same decompression process as found in (14). It is designed to perform the decompression operations on every data set used in this research. The specific sequence for each individual data set involves iterating through each subcarrier in each packet, and applying the quantization formulas to the angles, then the decompression formula to achieve the resulting matrix. The matrix is stored using 4 columns

for each packet, representing the four matrix elements in order of top-left, top-right, bottom-left, bottom-right.

3.4 Feedback Correlation and Evaluation Tool (FCaE)

3.4.1 Overview

The Feedback Correlation and Evaluation Tool (FCaE) is a set of utilities in MATLAB, which allow feedback matrix data to be visualized and analyzed. The feedback data consists of 936 complex numbers in every packet (4 four each of the 234 subcarriers), and the purpose of the utilities in the FCaE is to reduce this quantity of numbers to characteristics which describe the shape of the data with less dimensions, which may be calculated and trained, and which allow for comparison and testing of data.

The FCaE does this by defining three quantities to measure data sets: mean feedback, packet distance, and packet distance variance. These measures allow the FCaE to train on and evaluate data sets of feedback. The process of the FCaE at a high level establishes a baseline and conducts trials with data to compare against it. The trials are put through statistical tests. The definitions and mathematics are described in Section 3.4.2. The resampling trial process is described in Section 3.4.3. The comparison tests are described in Section 3.4.4. The data visualization utility is described in Section 3.4.5. Finally, Section 3.4.6 takes a detailed look at how the parts fit together to produce 3 output variables for how the defined measures allow insights from feedback matrix data.

3.4.2 FCaE Mathematics

Every subcarrier has its own feedback matrix, of four elements. Since each subcarrier operates independently, in comparing them the FCaE analysis compares them

independently. The method used to do this is by establishing the euclidean distance between the two feedback matrices. Given two subcarrier feedback matrices X_s and Y_s on subcarrier s , the subcarrier distance d_s is defined as:

$$d_{s:X,Y} = \frac{\sum_{i=1}^4 \sqrt{(\text{real}(X_{s_i}) - \text{real}(Y_{s_i}))^2 + (\text{imag}(X_{s_i}) - \text{imag}(Y_{s_i}))^2}}{4} \quad (16)$$

S is the set of subcarriers which provide feedback. This implies that there are $n_s = |S|$ subcarriers total in an entire packet of beamforming feedback. In this research, it is 234. The packet distance is defined as the mean of all the subcarrier distances between two packets. Between packets X and Y , the packet distance $d_{p:X,Y}$ is defined as:

$$d_{p:X,Y} = \frac{\sum_{s \in S} d_{s:X,Y}}{n_s} \quad (17)$$

A capture, $C_{E,L}$, is a sequence of feedback packets in an environment E , and transmitted from a device at location L . The possible environments are baseline, denoted B , minimal-motion, denoted M , and motion-saturated, denoted S . The locations in this research are the cardinal and ordinal directions, labeled from 1-8, starting at the north and moving counterclockwise. This allows the shorthand for the capture $C_{B,2}$ of B_2 (which is the baseline environment capture at the northwest location). Captures are denoted as sequences instead of sets in order to acknowledge the time constrained nature of the feedback data. The number of packets in a capture is denoted by n_p . A specific packet is denoted by its index in the capture, with subscripts denoting the subcarrier and feedback matrix element (1-4), so that $4_{-5,2}$ represents the value of the top-right feedback matrix element for subcarrier -5, in the fourth packet of the capture.

Every capture has a mean and standard deviation matrix associated with it: M_C and D_C respectively. These matrices have the same dimensions as a matrix representing a packet: $n_s \times 4$, where each column represents an element of a feedback matrix. The mean matrix M_C is defined as the mean of each element of feedback for each subcarrier across every packet in a capture. The standard deviation matrix is defined as the standard deviation (split into real and complex components) of the same values across a captures. Each element $x_{s,i}$ of the matrix M_C is defined as:

$$x_{s,i} = \frac{\sum_{p=1}^{n_p} p_{s,i}}{n_p} \quad (18)$$

The deviation matrix, instead of being the mean of the set given for each element, is the standard deviation of each element's set, so that each element $x_{s,i}$ is defined as:

$$x_{s,i} = std(\{p_{s,i} : p = 1 \dots n_p\}) \quad (19)$$

The standard deviation in (19) is calculated by individually determining the standard deviation in the real and imaginary components, and then adding them together to create a complex standard deviation.

3.4.3 FCaE Trial Resampling

The closeness of different feedback matrices to each other may be determined by utilizing the mean packet matrix, and the packet distance. First, the mean, M_C , of the capture, C , is calculated, and the packet distances of each packet in C are calculated against the mean. This shows a distribution of how "far" each packet is from the average of the entire capture. However, this distribution results in a problem: it is left-skewed.

The left-skew of the distribution is due to the packet distance being calculated as a

Euclidean distance, which is only non-negative. Any variation in the same magnitude, but in opposite directions from the mean is in fact treated as the same under this distribution since packet distance only accounts for magnitude. In order to utilize the packet distance statistically, the FCaE transforms it into a normal distribution through resampling trial means.

Due to the Central Limit Theorem, distributions of means may be treated as normal distributions. Therefore, a distribution of mean packet distance may be treated as normal, where a packet distance distribution is not.

A trial may be defined as a proper subset of a capture, which is randomly selected with replacement. This allows each packet within the trial to be evaluated for packet distance (against some value), and then the mean packet distance across the trial to be calculated. The purpose of the FCaE Resampler component is to allow for the creation and evaluation of trials from a capture. A series of trials allows the distribution of trial means to be calculated, which is treated as a normal distribution due to the Central Limit Theorem.

When the FCaE trains a baseline, it does so by creating such a distribution from a number of trials equivalent to the size of the capture it trains on. The baseline consists of the mean feedback matrix for the capture, and the distribution of trial mean packet distances, with the distance evaluated against the mean for the capture.

3.4.4 FCaE Testing

The FCaE has two tests which it applies to the feedback data. The first is a two-sample t-test to assess location, and the second is a two-sample F-test to assess environment.

3.4.4.1 Location Testing

The location testing is performed with two inputs: a baseline, and a capture to be tested against it. The baseline is established as described in Section 3.4.3. Multiple trials are derived from the capture being evaluated, with the packet distance being evaluated against the mean in the baseline (which is not necessarily the mean of the capture being tested). These distances from multiple trials form a distribution, labeled a “group,” which is tested against the distribution in the baseline through a two-sample t-test.

Since the t-test fails to reject the hypothesis when there is not statistically significant difference between the distributions, in order to account for variation in the data and to ensure accuracy in the goal of distinguishing locations, the alpha value for the t-test is set to 0.0001. This requires there to be only a 0.01% probability that the distributions are the same for the test to fail distinguishing them.

The FCaE evaluates multiple groups when comparing the capture and baseline in location testing. This allows the results to be measured in success and failure rates. When the t-test rejects the null hypothesis, this is called a distinction. When the t-test fails to reject the null hypothesis, this is called a correlation. The FCaE’s location testing is evaluated by determining the success at performing distinctions by comparing the baselines for each test location to the baseline captures which trained them. It is also evaluated by determining the success at performing correlations by comparing the baselines for each test location to non-baseline captures taken from the same locations.

The purpose behind capture selection for testing is to best test each function. Testing the baselines against the baseline captures for distinction is designed to show that in the evaluation for distinction, that the FCaE may successfully distinguish between different locations, and train a model based off of them, that has inter-

nal consistency and use. This validates the training model, and proves the utility of beamforming feedback matrices in distinguishing locations, by concretely pairing certain data characteristics with certain locations. It would not be useful to utilize the baseline captures in evaluating the correlation capability however, as the testing procedure which produces groups is the same as that which generated the baseline. New data must be used to test the validity of location correlation, which validates the training model’s applicability, and demonstrates that beyond simply distinguishing locations, the FCaE may identify them as well using beamforming feedback data.

3.4.4.2 Environment Testing

The environment testing is performed between two capture inputs. The only purpose is to tell if the amount of motion in the environment near the devices which are beamforming is the same, or different, between the two captures, but not to identify which one has more or less motion. If a difference is detected, previous knowledge of one capture (for example, a training baseline), or the magnitude of packet distance variance, may be utilized in determining this information.

The test rests upon the basis that disruptions and changes in the physical environment change the path which beamforming utilizes, thus changing the CSI which is recorded in beamforming feedback. This change varies in magnitude in different sub-carriers. Thus, in an environment which remains constant, little variation in feedback is expected, while in an environment which contains significant motion to disrupt signal paths, much variation in feedback is expected. Thus, a two-sample F-test may be utilized to test the variance between samples taken from the two captures.

The test, similar to the location test is performed once per group. Each group is made up of a distribution of mean packet distances from a number of trials. Each trial evaluates packet distance against the mean of the packets within that trial.

This ensures that instead of being compared to each other initially, each capture’s own internal characteristics are first determined, and then compared. A rejection of the null hypothesis is called a distinction. This then forms a success and failure rate, based on the groups tested, by which the FCaE’s environment testing may be evaluated.

3.4.5 FCaE Visualization

The FCaE also provides a visualization tool, which allows the nature of a feedback matrix element across a capture to be intuited, and compared to other captures. Each element of the feedback matrices is complex, which means that it may be plotted on a complex plane. The mean and deviation matrices provide additional characteristics of the capture. The visualization tool then plots the mean across the capture, surrounded by three ellipses representing 1, 2, and 3 standard deviations out from it, as well as the value of the element from each packet in the capture. This shows the distribution and location on the complex plane, which characterizes one feedback matrix element for an entire capture.

The tool is not utilized in any mathematical analysis, but only for qualitative analysis and comparison between captures. It assists in perceiving the similarity and difference between different captures in mean, deviation, and distribution.

3.4.6 FCaE Process Summary

The FCaE in total is designed to provide a method of characterizing feedback data, training a model, and performing two tests of captures against that model. It has three functions: to distinguish locations, to correlate locations, and to distinguish environments. It starts with captures, establishing the data characteristics. It then uses a trial resampling process to both train baseline models, and to create test

distributions to evaluate against the models. The models and test distributions are evaluated by the location and environment testing. When under test, this research selects specific models and captures to test in order to evaluate the FCaE's success at performing its three functions. This is summarized in Figure 20, which describes the captures being processed into training and trials, and the trials and models being used to perform the two tests, which produce three outputs: the location distinction successes and failures, the location correlation successes and failures, and the environment distinction successes and failures. The whole process together is composed in a series of commands in Appendix G, which utilizes the functions in Appendix H to accomplish the tasks described in this section.

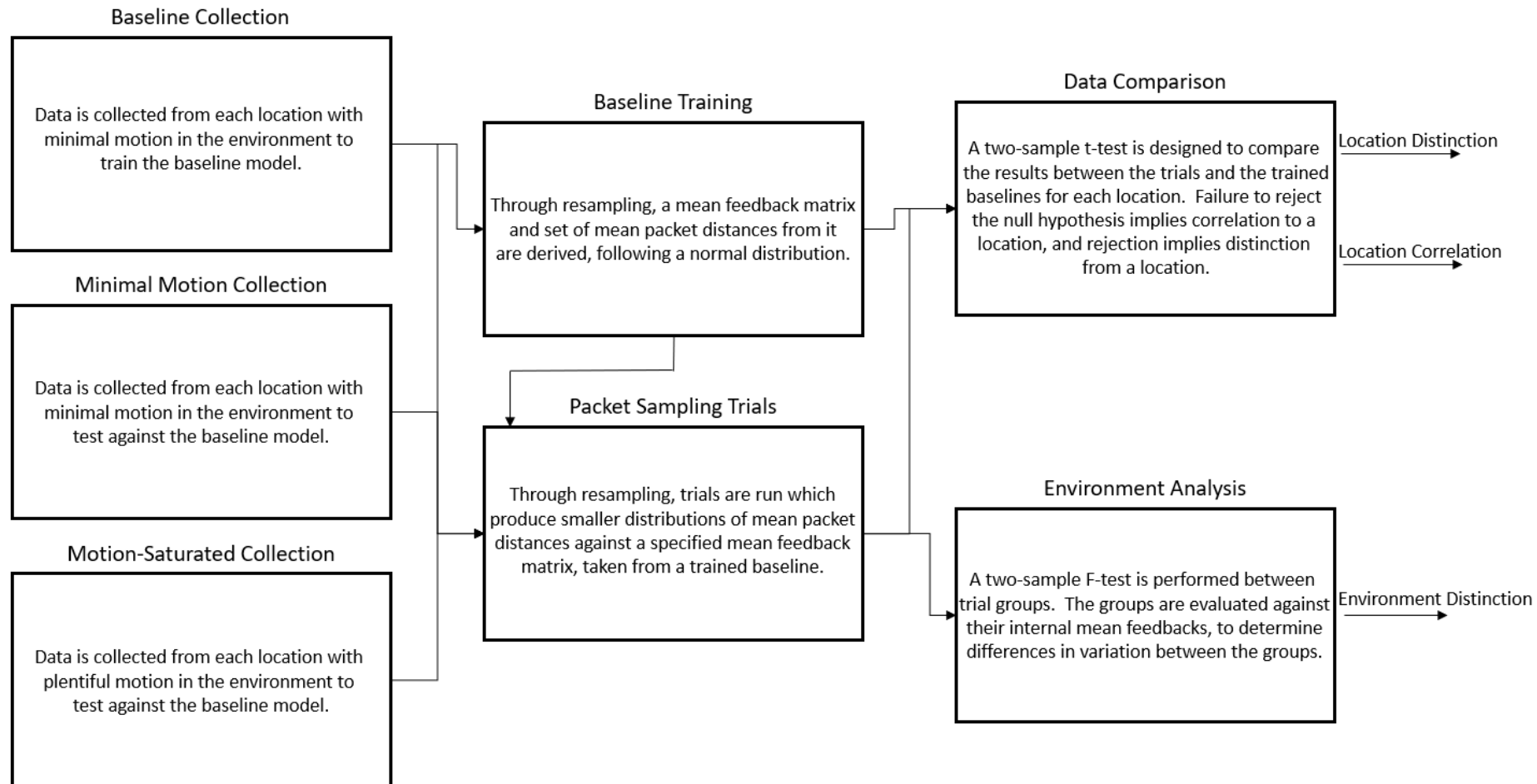


Figure 20: FCaE Block Diagram

IV. Methodology

4.1 Overview

This chapter discusses the methodology used by this research in conducting experiments and analyzing the results. Section 4.2 discusses the problem which this research investigates. Section 4.3 presents the entire System Under Test (SUT), with the factors and metrics investigated in this research. Section 4.4 discusses the assumptions made for the experiments. Section 4.5 presents the response variables analyzed, and sets them in the context of the metrics in the SUT. Section 4.6 discusses the variables which are controlled in the experiment, and Section 4.7 discusses ones left uncontrolled, and the reasons for doing so. Section 4.8 presents the design of the experiment area and system, while Section 4.9 presents the precise method used to collect the data when conducting the experiment. Finally, Section 4.10 discusses the analysis used to produce rates for evaluating the metrics from the response variables.

4.2 Problem

The problem which this research aims to solve pertains to the utility of information that is leaked by 802.11ac in its beamforming feedback reports. It seeks to show that an attacker can monitor the wireless network, and from that construct a baseline from which the feedback reports can be evaluated to distinguish the physical location of devices, and to detect motion in the environment. The objectives can be listed as:

1. Demonstrate that beamforming feedback reports may be observed, extracted, and displayed, without the observer connecting to or breaking into a secured 802.11ac network.

2. Collect baseline data from different locations and demonstrate the ability of the FCaE to distinguish the locations.
3. Collect additional data from the previous locations, and demonstrate the ability of the FCaE to correlate it with the baseline training.
4. Collect additional data from the previous locations when there is motion in the target environment, and demonstrate the ability of the FCaE to distinguish the state of the environment between captures taken at the same test location.

These methods of evaluation demonstrate that 802.11ac networks reveal physical information about the environment in which they operate, and about the locations of devices which make up the network.

4.3 System Under Test

The System Under Test (SUT) is displayed in Figure 21. The input factors being tested are the location of devices relative to the AP, and the motion in the environment. Section 4.6 discusses the system parameters which are kept controlled and constant throughout the experiment. The three components of the FCaE which are tested are those for training and distinguishing baselines from different locations, the comparison test which evaluates against baselines, and the environment analysis which detects differences in the amount of motion in the environment. The baseline training is evaluated by the location distinction metric, as measured by the DSR. The comparison test is evaluated through the location correlation and motion correlation results, which compare the minimal motion and motion-saturated captures against the baselines. The environment analysis component is evaluated by the environment distinction test. Each metric is described in terms of true positives, false positives, and false negatives, with the final results being described in terms of rates. The in-

dividual response variables are described in Section 4.5, and the resulting rates are described in Section 4.10.

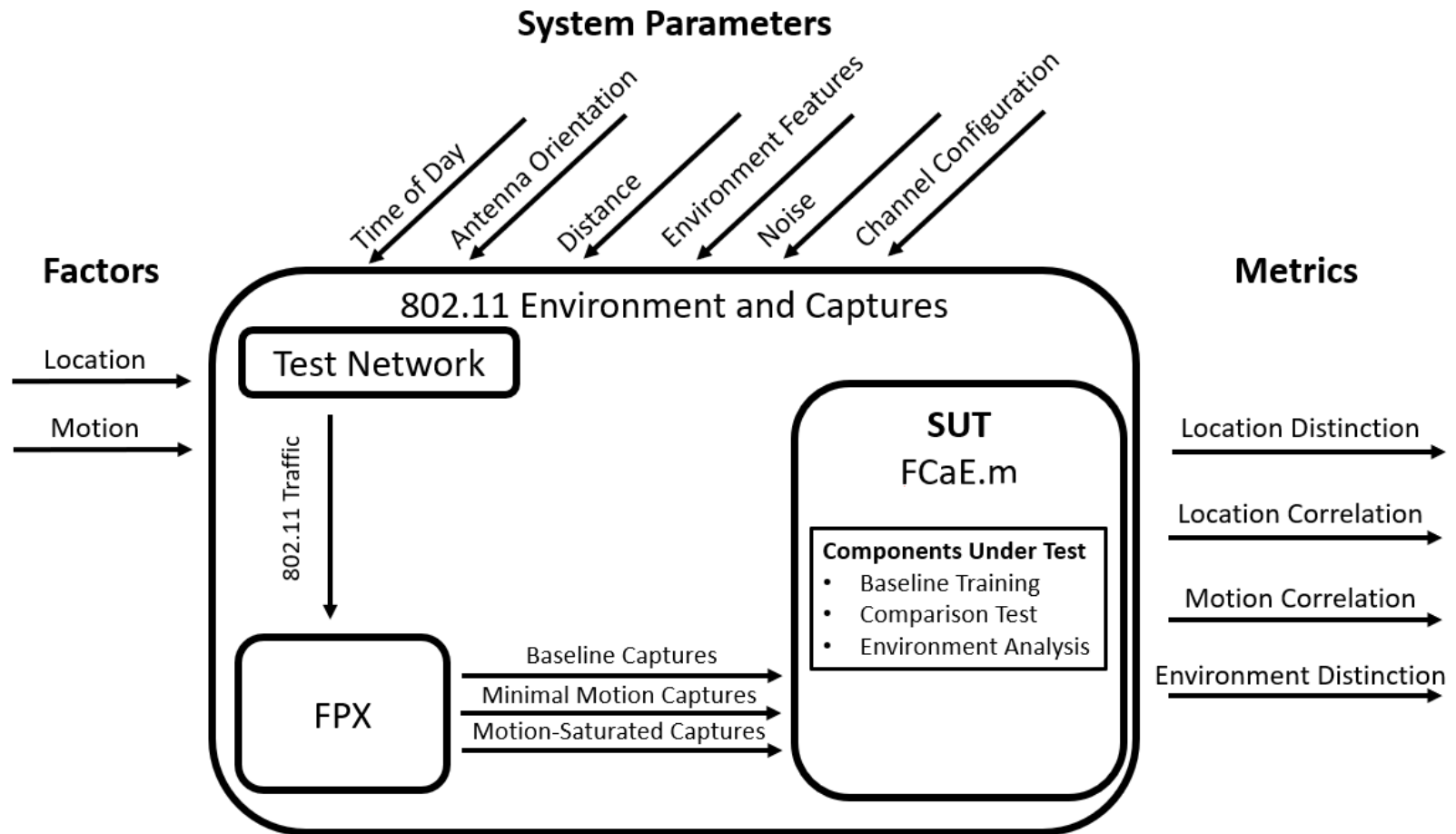


Figure 21: The SUT diagram demonstrates the controlled parameters, varied factors, and output metrics of this research. The FCaE has 3 components under test, which are evaluated by the success of different output metrics

4.4 Assumptions

The following assumptions are made for designing and conducting experiments on the SUT.

1. LOS between the AP and test adapter is not obstructed in the baseline state.
2. The multipath effect of the indoor environment is stable over time.
3. The router and client are on the same horizontal plane.
4. Noise from other 802.11 systems has a stable effect over time.
5. The router uses 80 MHz channels with 2 spatial streams.

4.5 Response Variables

The objectives of this experiment inform the response variables which are measured. The response variable changes with which experiment is being run, and what component is being tested. Success and failure rates can be directly observed. The variables below are organized under the metric categories from the SUT diagram, Figure 21.

- **Metric 1:** Perform Location Distinction.

Distinguishing Success True Positives (DSTP): The DSTP represents the number of trials in which the baseline training data for a location was successfully distinguished from all other baselines, and not distinguished from its own baseline.

Distinguishing Success False Positives (DSFP): The DSFP represents the number of trials in which the baseline training data for a location failed to be distinguished from another baseline.

- **Metric 2:** Perform Location Correlation.

Location Correlation True Positives (LCTP): The number of successful identifications with the correct baseline group of new minimal motion data.

Location Correlation False Positives (LCFP): The number of identifications which are with a baseline group to which a packet group of new minimal motion data did not belong.

Location Correlation False Negatives (LCFN): The number of rejections from a baseline group to which a packet group of new minimal motion data did belong.

- **Metric 3:** Perform Motion Correlation.

Motion Correlation True Positives (MCTP): The number of successful identifications with the baseline group from data taken during motion in the target environment.

Motion Correlation False Positives (MCFP): The number of incorrect identifications with a baseline group from data taken during motion in the target environment.

Motion Correlation False Negatives (MCFN): The number of incorrect rejections with a baseline group from data taken during motion in the target environment.

- **Metric 4:** Perform Environment Distinction.

Environmental Distinctions Count (EDC): The number of null-hypothesis rejections in comparing two captures with the environment analysis test.

This research does not examine the true negatives for metrics two and three. The reason for this is in the goal of what the metrics measure. True negatives verify

the distinction of locations, not the identification or correlation of locations against a model. Since the first metric, using the DSTP and DSFP is designed to test the capability to distinguish location, the true negatives are not of concern for the other metrics, which depend on a model validated by the first metric.

4.6 Controlled Variables

There are several factors in the system that could vary and be analyzed, but are being held constant for simplicity, and establishing a baseline of knowledge where no previous investigation or analysis has been performed. The following variables are intentionally kept constant:

1. **Position:** For all trials run at a particular position, the router and test adapter are not moved. This means that the baseline, minimal motion, and motion-saturated data are all collected before moving the test adapter to the next position. This is first, to ensure the accuracy of the data by guaranteeing that the position is identical. Second, to counteract the possible and uncontrollable effects of differences over time in the environment such as noise from other 802.11 networks. Each position roughly 1 meter between the center of the test adapter, and the center of the router regardless of position, in order to ensure that no distance-related affects would modify the results. This 1 meter distance was chosen to accommodate the test area's limitations, as well as to provide evidence of distinction at locations which were a short distance from each other. Eight test locations are chosen, and labeled by their direction relative to the router as North, Northwest, West, Southwest, South, Southeast, East, and Northeast.
2. **Time of day:** Trials ran during the day, between the hours of 10:00 AM and 2:00 PM. Running all the trials within the same brief timespan ensures that

changes and differences in the wireless environment over time are minimized, but since this time is during the day, it emulates a realistic wireless environment with other active network signals providing noise.

3. **Order of collection:** To ensure that the sequence of collection between the baseline, minimal motion, and motion-saturated captures did not influence results, the order is randomized. The randomized order is presented in Table 2.
4. **Channel Specifications:** The channel bandwidth is limited to 80 MHz on the router to ensure consistency of the feedback data's size and structure. The adapter is only SU beamforming capable, to further ensure that the type of feedback was consistent. The number of antennae was held constant to provide the final piece of constant data to standardize the format of the beamforming feedback. This was done with the smallest possible situation of 2x2 MIMO, to provide only one option for the beamforming feedback matrix size, as seen in Section 2.2.3.3.
5. **Antenna Orientation:** To prevent additional complexity, the orientation of the antennae relative to the router was maintained at every test location. Devices which are non-mobile are not expected to move or have changing antennae orientations.

Table 2: Randomized collection order for each test location. B is baseline collection, M is minimal-motion collection, and S is motion-saturated collection.

	N	NW	W	SW	S	SE	E	NE
First	M	M	S	M	B	S	S	M
Second	S	B	B	B	S	M	M	S
Third	M	S	M	S	M	B	B	B

4.7 Uncontrolled Variables

One of the consequences of modeling with a realistic wireless environment is the interference of other signals, and the potential of multipath communication. There are some variables in this situation which cannot be controlled, but are not expected to significantly impact the experiment.

1. **Noise:** The wireless noise level is not controllable, but is expected to remain stable with minimal impact over the timespan of data collection. The modeling equation $y = Hx + n$ for wireless channels also demonstrates that noise is modeled as a different effect than the CSI which the feedback matrix is intended to represent. With the close proximity of the test adapter and the router, and the lack of other transmission devices in the same room in which the experiments were conducted, noise is expected to have a minimal effect due to the higher signal-to-noise ratio (SNR), and to not affect the outcome of experiments.
2. **Multipath Communication:** The potential for multipath communication is high, especially for indoor environments, with walls and other objects providing reflective surfaces, like the test environment. However, multipath communication potential does not eliminate the connection between the CSI and the location. The signal paths are not being investigated by this research, but only the locations. The location correlation is not concerned with how the signal arrives at the location, only that it does. This is also less significant for the baseline training, where LOS is maintained, and minimal changes in the environment occur. During motion-saturated testing, multipath communication can play a more significant role, but the role it plays is used to distinguish an environment with motion from one without, and providing variation in the data.
3. **Airflow:** Within the test area are air vents, which are unable to be controlled.

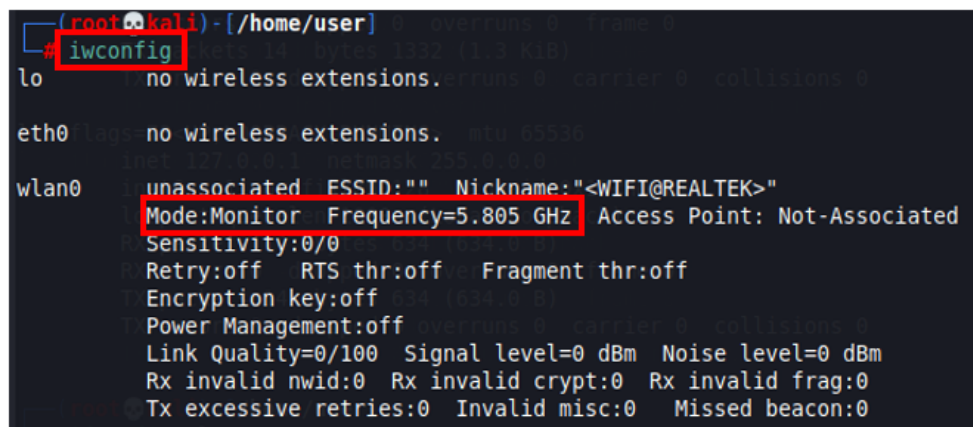
These vents cause irregularly timed air motion that may disrupt some tests. The vents themselves are near the Northeast test location. It is unknown if air motion is sufficient to cause disruption in testing, and this will be investigated and discussed in the results.

4.8 Experimental Design

4.8.1 System Configuration Verification

The system must be verified in its configuration before testing can proceed. First, the settings on the monitor adapter must be verified to assure it is in monitor mode, and it is tuned to the correct frequency range for observing the 5 GHz network. This is done with the command “iwconfig.” Figure 22 demonstrates the confirmation of settings.

The settings on the router must also be verified, that it is active on the 5 GHz band, protected by WPA2, and is limited to 80 MHz channels, with explicit beamforming capabilities activated. This can be done through the router’s web interface, as seen in Figure 23.

A terminal window with a dark background. The prompt is `(root@kali) - [/home/user]`. The command `# iwconfig` has been entered. The output shows details for `lo`, `eth0`, and `wlan0`. The `wlan0` section is highlighted with a red box, showing: `wlan0 unassociated ESSID:"" Nickname:"<WIFI@REALTEK>" Mode:Monitor Frequency=5.805 GHz Access Point: Not-Associated Sensitivity:0/0 Retry:off RTS thr:off Fragment thr:off Encryption key:off Power Management:off Link Quality=0/100 Signal level=0 dBm Noise level=0 dBm Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0 Tx excessive retries:0 Invalid misc:0 Missed beacon:0`.

```
(root@kali) - [/home/user]
# iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

wlan0     unassociated  ESSID:""  Nickname:"<WIFI@REALTEK>"
          Mode:Monitor  Frequency=5.805 GHz  Access Point: Not-Associated
          Sensitivity:0/0
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality=0/100  Signal level=0 dBm  Noise level=0 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0  Missed beacon:0
```

Figure 22: Confirming the configuration of the adapter in monitor mode, in the 5 GHz band for use in the experiment

Wireless - General

Set up the wireless related information below.

Band	5GHz ▾
Network Name (SSID)	LexRex5
Hide SSID	☐ Yes ☒ No
Wireless Mode	Auto ▾ ☐ Optimized
Channel bandwidth	80 MHz ▾
Control Channel	Auto ▾ Current control channel
Extension Channel	Auto ▾
Authentication Method	WPA2-Personal ▾
WPA Encryption	AES ▾

Wireless - Professional

Wireless Professional Setting allows you to set up additional parameters.

* Reminder: The system time has not been synchronized with an NTP server.

* Reminder: The System time zone is different from your locale setting.

Band	5GHz ▾
Enable Radio	☒ Yes ☐ No
Multi-User MIMO	Enable ▾
Explicit Beamforming	Enable ▾
Implicit beamforming	Disable ▾

Figure 23: Confirming the configuration settings of the 5 GHz band of the router

Further configuration and capabilities verification includes confirming that the adapter and the router share the same SU beamforming capability. This is verified through a brief Wireshark capture from the monitor adapter, which observes a beacon from the router, and a probe from the adapter. Figure 24 shows confirmation of the router’s capabilities, while Figure 25 shows confirmation of the adapter’s capabilities.

2417	32.939981943	ASUSTekC f3:a3:04	Alfa ad:09:84	802.11	416	Probe Response, SN=1349, FN=0, Flags=.....C,
2418	32.949537276	ASUSTekC f3:a3:04	Broadcast	802.11	311	Beacon frame, SN=1766, FN=0, Flags=.....C, E
2419	33.051782819	ASUSTekC f3:a3:04	Broadcast	802.11	311	Beacon frame, SN=1767, FN=0, Flags=.....C, E
2420	33.256362699	ASUSTekC f3:a3:04	Broadcast	802.11	311	Beacon frame, SN=1770, FN=0, Flags=.....C, E

▶	Frame 2418: 311 bytes on wire (2488 bits), 311 bytes captured (2488 bits) on interface wlan0, id 0
▶	Radiotap Header v0, Length 32
▶	802.11 radio information
▶	IEEE 802.11 Beacon frame, Flags: C
▼	IEEE 802.11 Wireless Management
▶	Fixed parameters (12 bytes)
▼	Tagged parameters (239 bytes)
▶	Tag: SSID parameter set: LexRex5
▶	Tag: Supported Rates 6(B), 9, 12(B), 18, 24(B), 36, 48, 54, [Mbit/sec]
▶	Tag: DS Parameter set: Current Channel: 161
▶	Tag: Traffic Indication Map (TIM): DTIM 1 of 1 bitmap
▶	Tag: Country Information: Country Code US, Environment Any
▶	Tag: HT Capabilities (802.11n D1.10)
▶	Tag: HT Information (802.11n D1.10)
▶	Tag: Extended Capabilities (8 octets)
▼	Tag: VHT Capabilities
▶	Tag Number: VHT Capabilities (191)
▶	Tag length: 12
▼	VHT Capabilities Info: 0x338959b2
.....	..10 = Maximum MPDU Length: 11 454 (0x2)
.....	00.. = Supported Channel Width Set: Neither 160MHz nor 80+80 supported (0x0)
.....	...1 = Rx LDPC: Supported
.....	...1. = Short GI for 80MHz/TVHT_MODE_4C: Supported
.....	...0.. = Short GI for 160MHz and 80+80MHz: Not supported
.....	...1... = Tx STBC: Supported
.....	...001 = Rx STBC: 1 Spatial Stream Supported (0x1)
.....	...1... = SU Beamformer Capable: Supported
.....	...1 = SU Beamformee Capable: Supported
.....	...010. = Beamformee STS Capability: 3 (0x2)
.....	...001 = Number of Sounding Dimensions: 2 (0x1)

Figure 24: This capture displays a beacon frame from the router. The source MAC address confirms that it is the test router, and the VHT capabilities field confirms that SU beamforming is a capability. An additional feature also displayed in the details is that the router is capable of being both beamformer and beamformee, although 802.11ac does not provide the opportunity for an AP to use this capability.

No.	Time	Source	Destination	Protocol	Length	Info
2420	33.256362699	ASUSTekC_f3:a3:04	Broadcast	802.11	311	Beacon frame, SN=1770, FN=0, Flags=.....C, E
2421	36.598558383	Alfa_ad:00:87	Broadcast	802.11	132	Probe Request, SN=1054, FN=0, Flags=.....C, E
2422	36.598559884	Alfa_ad:00:87	Broadcast	802.11	132	Probe Request, SN=1055, FN=0, Flags=.....C, E
2423	36.598561479	ASUSTekC_f3:a3:04	Broadcast	802.11	46	Acknowledgement, Flags=.....C


```

Supported Rates: 36 (0x48)
Supported Rates: 48 (0x60)
Supported Rates: 54 (0x6c)
  Tag: HT Capabilities (802.11n D1.10)
    Tag Number: HT Capabilities (802.11n D1.10) (45)
    Tag length: 26
    HT Capabilities Info: 0x09ee
    A-MPDU Parameters: 0x03
    Rx Supported Modulation and Coding Scheme Set: MCS Set
    HT Extended Capabilities: 0x0000
    Transmit Beam Forming (TxBF) Capabilities: 0x00810008
    Antenna Selection (ASEL) Capabilities: 0x00
  Tag: VHT Capabilities
    Tag Number: VHT Capabilities (191)
    Tag length: 12
    VHT Capabilities Info: 0x33c031a0
      .... = Maximum MPDU Length: 3 895 (0x0)
      .... = Supported Channel Width Set: Neither 160MHz nor 80+80 supported (0x0)
      .... = Rx LDPC: Not supported
      .... = Short GI for 80MHz/TVHT_MODE_4C: Supported
      .... = Short GI for 160MHz and 80+80MHz: Not supported
      .... = Tx STBC: Supported
      .... = Rx STBC: 1 Spatial Stream Supported (0x1)
      .... = SU Beamformer Capable: Not supported
      .... = SU Beamformee Capable: Supported
      .... = Beamformee SIS Capability: 2 (0x1)
      .... = Number of Sounding Dimensions: 1 (0x0)

```

Figure 25: This capture displays a probe frame from the test adapter. The source MAC address confirms that it is from the test adapter, and the VHT capabilities field confirms that it is a SU beamforming beamformee, although cannot be a beamformer.

4.8.2 Test Area Physical Layout

The physical layout of the test locations and devices is presented in Figure 26. The layout is designed to allow clear and comprehensive analysis of beamforming feedback from different directions between the router and client device.

A tape measure is used to perform measurements of 1 meter from the center of the router, and directed at the proper angle, so that the end of the tape is at the proper location for the client device's test adapter. Once the test adapter is placed in this location, and the measurement verified, the tape measure is removed. This step is performed to place the test adapter at each test location prior to collecting data.

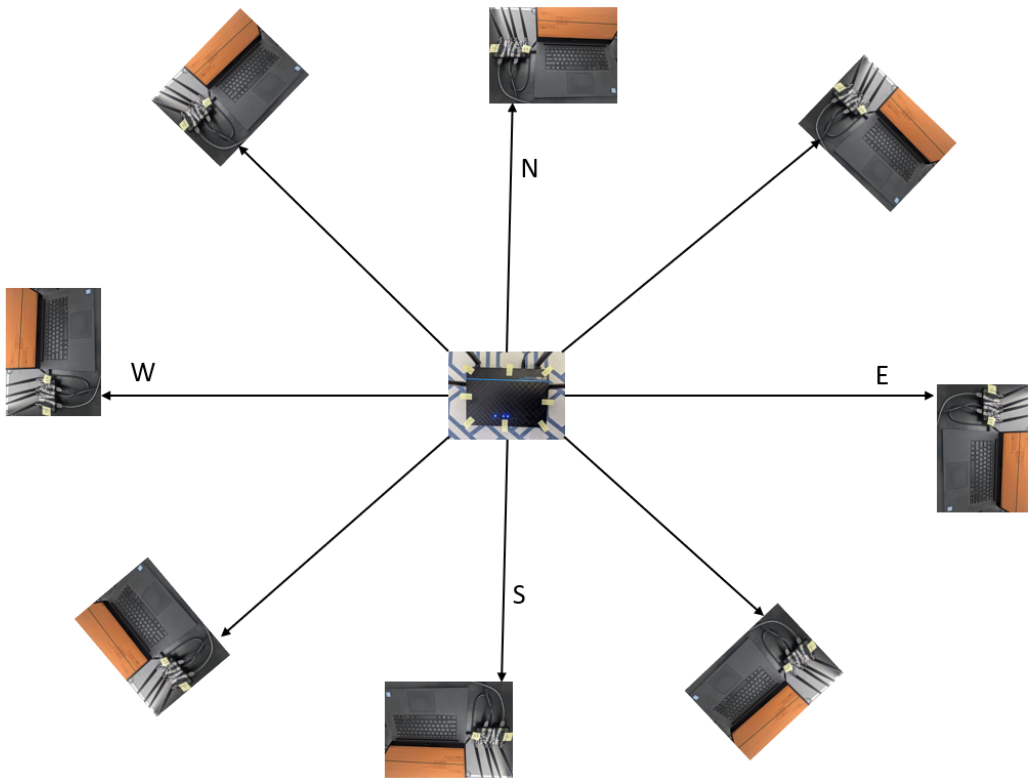


Figure 26: The layout is shown, with N, S, E, and W directions labeled. Each distance is approximately 1 meter. The router contains labels to denote each direction.

4.9 Data Collection

The process of data collection follows these steps:

1. **Verify Position:** The position is measured with a tape measure as described in the Section 4.8.2, and verified that it is correct. If it is incorrect, it is adjusted.
2. **Start Collection:** The Kali VM starts collecting 802.11 packets through the monitor adapter using Wireshark. The display is filtered to include only those packets that are transmitted from the test adapter and are VHT management packets, with the filter “`wlan.ta == 00:C0:CA:AD:09:87 && wlan.vht.action`”. This is shown in Figure 27.
3. **Connect to Network:** The PC host connects to the wireless test network, “LexRex5” with the test adapter.
4. **Wait for Collection:** As beamforming occurs, the amount of displayed packets in the Wireshark capture grows. This capture is monitored until the amount reaches approximately 100 for each captures, or ten minutes have passed, whichever comes sooner. During the baseline and minimal motion captures, there is minimal motion within the target environment. During the motion-saturated capture, a person walks around the target environment, and between the test adapter and the router, periodically disrupting LOS between them.
5. **Finish Collection:** Once the specified amount of packets or time limit has been exceeded, the capture is stopped. The filtered packets are exported as a collection of C arrays. This produces a final .c file with the C array representation of the filtered capture.
6. **Disconnect from Network:** The test adapter is disconnected from the network in preparation for the next trial. If data collection remains for the current

test location, this process is repeated. If not, the next test location is moved to, and this process is repeated.

wlan.ta == 00:C0:CA:AD:09:87 && wlan.vht.action						
No.	Time	Source	Destination	Protocol	Length	Info
22396	241.473087772	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3773, FN=0, Flags=...
22429	242.174187589	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3775, FN=0, Flags=...
22748	248.203113424	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3781, FN=0, Flags=...
22955	253.471709708	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3784, FN=0, Flags=...
23116	257.317061928	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3787, FN=0, Flags=...
23134	257.477448101	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3788, FN=0, Flags=...
23225	260.486748199	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3790, FN=0, Flags=...
23354	263.311593536	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3792, FN=0, Flags=...
23653	266.355539590	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3794, FN=0, Flags=...
23684	266.691787098	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3795, FN=0, Flags=...
24239	273.492058876	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3799, FN=0, Flags=...
24333	275.450338841	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3800, FN=0, Flags=...
24402	276.500231454	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3801, FN=0, Flags=...
24439	277.487834730	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3802, FN=0, Flags=...
24809	284.328963632	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3806, FN=0, Flags=...
25066	287.435480582	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3809, FN=0, Flags=...
25375	290.600964641	Alfa_ad:09:87	ASUSTekC_f3:a3:04	802.11	360	Action No Ack, SN=3812, FN=0, Flags=...

▶ Frame 28: 360 bytes on wire (2880 bits), 360 bytes captured (2880 bits) on interface wlan0, id 0
▶ Radiotap Header v0, Length 32
▶ 802.11 radio information
▶ IEEE 802.11 Action No Ack, Flags: C
▼ IEEE 802.11 Wireless Management
▼ Fixed parameters
Category code: VHT (21)
VHT Action: VHT Compressed Beamforming (0)
▶ VHT MIMO Control: 0xb48489, Nc Index: 2 Columns, Nr Index: 2 Rows, Channel Width: 80 MHz, Grouping (Ng): 1 (N
▼ VHT Compressed Beamforming Report: 18291249149144124924914412492491441249249144124924d1441249249144124d3491...
▶ Average Signal to Noise Ratio
▶ PH1 and PSI Angle Decode
▼ Beamforming Feedback Matrix
Compressed Beamforming Feedback Matrix for subcarrier -122
Compressed Beamforming Feedback Matrix for subcarrier -121
Compressed Beamforming Feedback Matrix for subcarrier -120
Compressed Beamforming Feedback Matrix for subcarrier -119
Compressed Beamforming Feedback Matrix for subcarrier -118
Compressed Beamforming Feedback Matrix for subcarrier -117

Figure 27: The filter displayed, “wlan.ta == 00:C0:CA:AD:09:87 && wlan.vht.action” correctly filters and displays only the beamforming feedback packets from the test adapter, which contain the compressed feedback matrix. This display also demonstrates that the size of each of these frames is 360 bytes, based on the channel and system controlled variables for the test.

4.10 Statistical Analysis

The statistical analysis is performed after collecting and formatting all of the baseline data as mean matrices and packet distances. Then, each packet from every situation is analyzed against every baseline to determine if it belongs to the same location as the baseline. This produces the following rates for analysis.

4.10.1 Distinguishing Success Rate (DSR)

The DSR is based on the total amount of packets in the baseline groups, which are successfully identified as their own group, and successfully rejected from every other group. The method is performed by taking groups of 10 packets at a time, to perform a t-test ($\alpha=0.001$) against each baseline's set of distances to that baseline's mean matrix. That is, two sets of mean packet distances are being compared: the baseline packet distances, and the test packet distances. The distances are calculated from the same mean matrix, which is the one for the baseline. The t-test is then performed by comparing the mean packet distances. If the test statistic is below the threshold, then the null hypothesis is rejected. The alpha level is made low in order to provide for variation, and to increase the likelihood of genuine distinction, as explained in Section 3.4.4.

For every packet group where the null hypothesis is rejected to baselines it didn't belong to and not rejected for the baseline it did belong to, a success is counted. Any other condition is a failure. Then, the DSR is the number of successes divided by the number of groups.

4.10.2 Location Correlation True Positive Rate (LCTPR)

The data used for this test is differently collected data from each location at rest. Each group of 10 packets is analyzed in the same format t-test as with the DSR.

The LCTPR is the amount of true positives (the FCaE correctly decides if the group belongs to a baseline). The final rate is:

$$LCTPR = \frac{N_{LCTP}}{N_{groups}} \quad (20)$$

where N_{LCTP} is the number of true positives, and N_{groups} is the number of test groups evaluated.

4.10.3 Location Correlation False Positive Rate (LCFPR)

The LCFPR is the measure of how often the FCaE mistakenly assigns a test group to a baseline to which it does not belong. It is described as:

$$LCFPR = \frac{N_{LCFP}}{N_{positives}} \quad (21)$$

where N_{LCFP} is the number of false positives, and $N_{positives}$ is the total number of positives evaluated.

4.10.4 Location Correlation False Negative Rate (LCFNR)

The LCFNR is the measure of how often the FCaE's negatives are false, and a distinction made is a false distinction. It is described as:

$$LCFNR = \frac{N_{LCFN}}{N_{negatives}} \quad (22)$$

where N_{LCFN} is the number of false negatives, and $N_{negatives}$ is the total number of negative results against the model.

4.10.5 Motion Correlation True Positive Rate (MCTPR)

The MCTPR is the measure of how the FCaE performs in correlating packets in a dynamic environment to their minimal motion baseline group. It is described as:

$$MCTPR = \frac{N_{MCTP}}{N_{groups}} \quad (23)$$

where N_{MCTP} is the number of correct identifications of a test group with its baseline, and N_{groups} is the number of test groups evaluated. This is a distinct number from the location correlation analysis, since it uses different data.

4.10.6 Motion Correlation False Positive Rate (MCFPR)

The MCFPR is the measure of how often the FCaE mistakenly identifies packets in a dynamic environment to a minimal motion baseline group not their own. It is described as:

$$MCFPR = \frac{N_{MCFP}}{N_{positives}} \quad (24)$$

where N_{MCFP} is the number of incorrect identifications of a test group with a baseline, and $N_{positives}$ is the number of total positives evaluated. This is also distinct number from the location correlation analysis.

4.10.7 Motion Correlation False Negative Rate (MCFNR)

The MCFNR is the measure of how often the FCaE's distinctions are false when testing a motion-saturated environment against a model. It is described as:

$$MCFNR = \frac{N_{MCFN}}{N_{negatives}} \quad (25)$$

where N_MCFN is the number of non-identifications of a test group with its baseline, and $N_{negatives}$ is the total number of negative results against the model.

4.10.8 Environmental Distinguishing Rate (EDR)

The EDR measures how often the FCaE is able to successfully distinguish between environments with minimal motion, and those which are motion-saturated. It is described as:

$$EDR = \frac{EDC}{N_{Trials}} \quad (26)$$

It is calculated individually for each location, and collated into a final rate.

V. Results and Analysis

5.1 Overview

This chapter discusses the results of the experiments. First, the general characteristics of the feedback matrix data are explored and discussed in Section 5.2. In Section 5.3 the baseline data and the correlation between location and feedback data is explored, evaluating the location distinction metric which tests the FCaE's ability to distinguish different locations. Following this Section 5.4 analyzes the minimal motion environment tests against the model to evaluate the location correlation metric. Section 5.5 investigates the motion-saturated data when tested against the model to evaluate the motion correlation metric. Section 5.6 investigates the environment distinction metric to determine the ability of the FCaE to distinguish between a minimal-motion and motion-saturated environment. Finally, the feasibility of using a motion-saturated environment for a baseline is investigated by re-evaluating the location distinction and location correlation metrics under this scenario in Section 5.7.

5.2 Feedback Matrix Characteristics

The feedback matrices are made of 4 complex numbers. The format is calculated from the compressed angle form following (14). There are 234 of each in a packet, one for each subcarrier from -122 to -2, and from 2 to 122 (minus the specific subcarriers discussed in Section 2.2.3.3). An example feedback matrix from subcarrier -122, in the west baseline capture is:

$$\begin{bmatrix} -0.3063 + 0.4129i & -0.5109 + 0.6889i \\ -0.8577 & 0.5141 \end{bmatrix} \quad (27)$$

Each element of the feedback matrix can be plotted on the complex plane, and

the standard deviation between different samples shown. A sample plot, showing the arrangement of data for the same element in the feedback matrix, on the same subcarrier, between the east and west baseline captures is shown in Figure 28. The purpose of this plot is to demonstrate two characteristics of the feedback matrix data. First, it demonstrates centrality, which is how despite variation, each element of feedback data is not randomly or haphazardly distributed around the unit circle, but rather is clustered in a distribution about a center: the mean of the capture in which it occurs. Centrality shows the stability of feedback data over time, and validates the use of statistical modeling. The second characteristic it demonstrates is location. The mean value is not at the same place for different data sets on the same subcarrier and element. Rather, each data set, corresponding to the device location, has its own location on the plot distinct from other data sets. This validates the use of feedback data in distinguishing and identifying device locations.

Another sample plot, showing the arrangement of data for the same element in the feedback matrix, on the same subcarrier, between the west minimal motion and motion captures is shown in Figure 29. This plot shows the same characteristics of centrality and location. However, it shows an additional characteristic, since the same location is observed within different environments. This additional characteristic is spread, which describes how closely or distantly the feedback data clusters around the mean. In the minimal motion plot, all the data points are clustered within three standard deviations, and most are within one or two. In the motion-saturated plot for the same feedback element, there are plentiful individual packets which contain data points spread outside the third standard deviation, and a higher variety of unique data points. These are different spreads of the data, which can be understood in a sense analogous to variance. The difference in spread, with the similarity of centrality and location, validates both an attempt to correlate and distinguish in a

motion-saturated environment, and the attempt to distinguish different environments by measuring spread through variance.

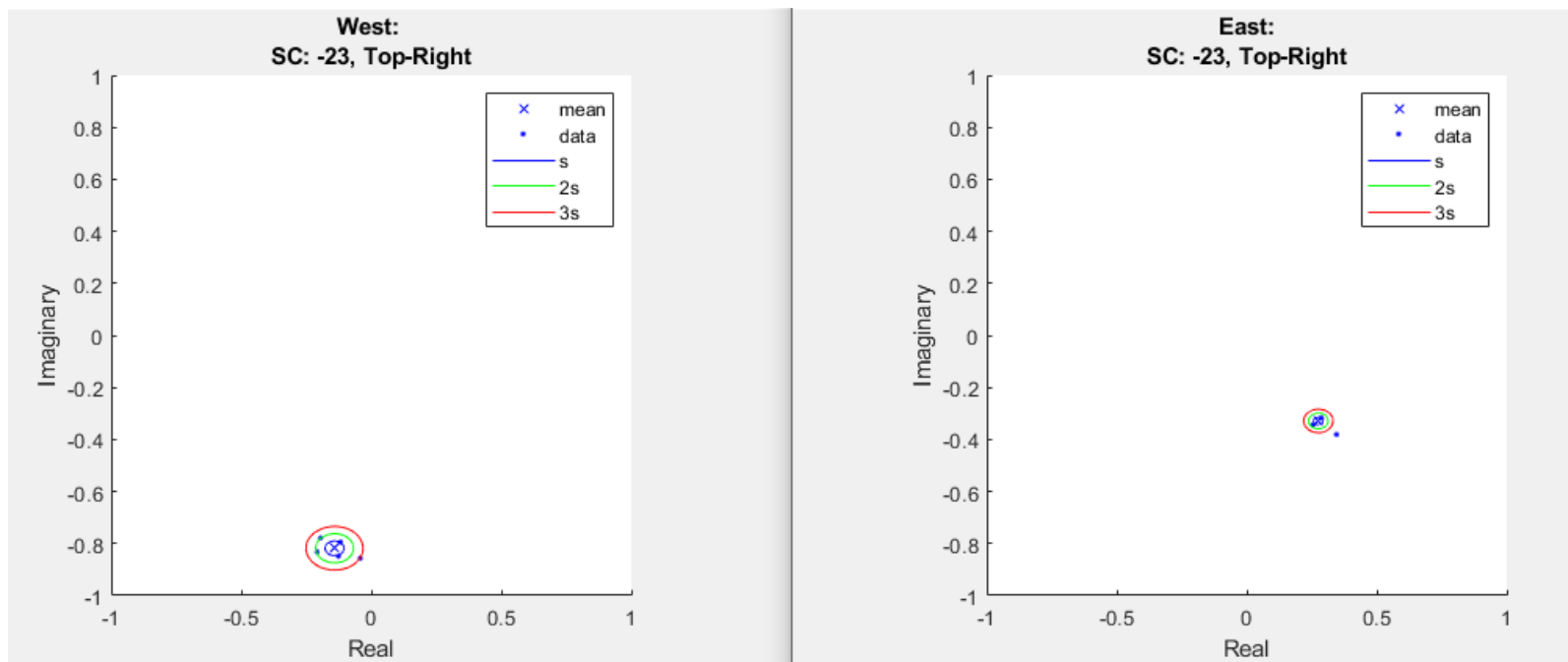


Figure 28: The West and East baselines are compared in plotting the top-right element of the feedback matrices on subcarrier -23.

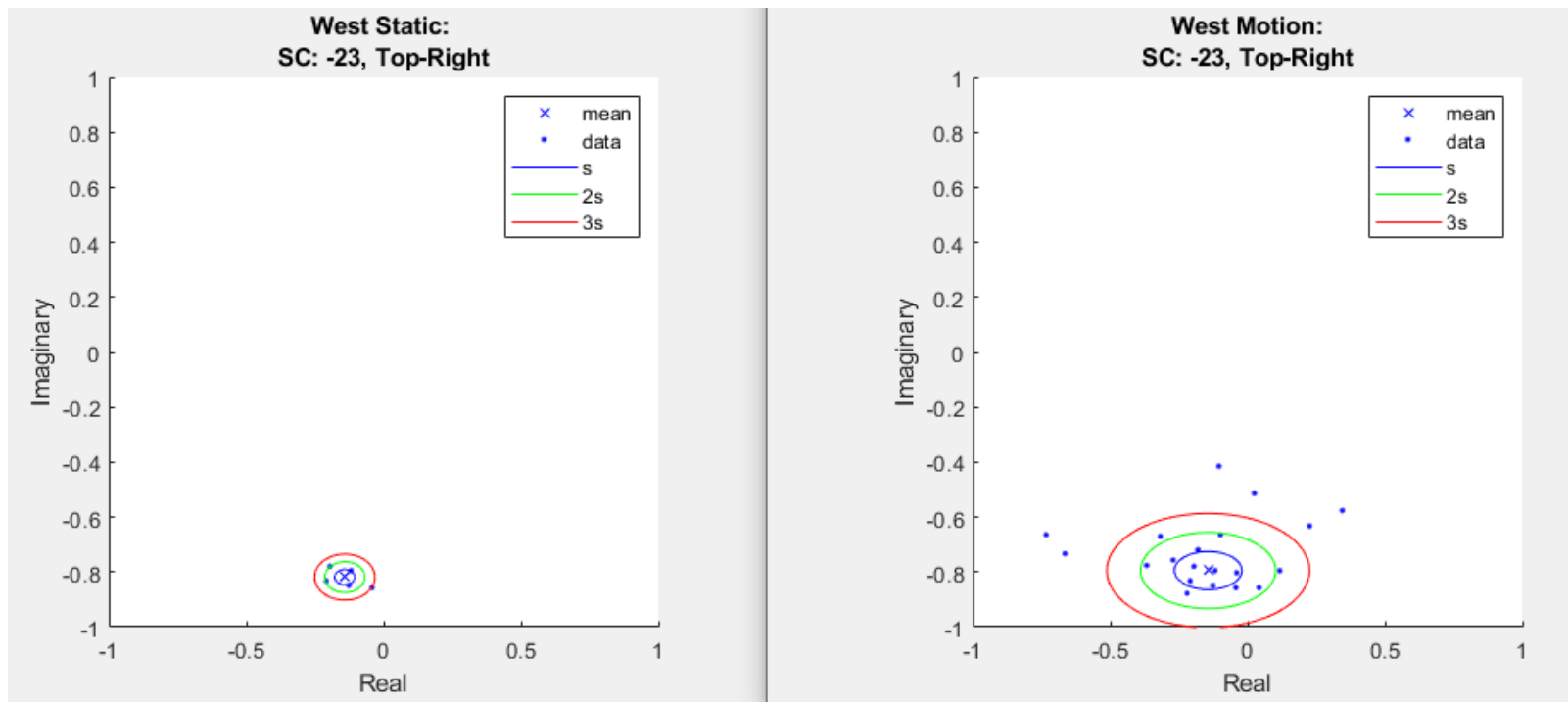


Figure 29: The West baseline and motion captures are compared in plotting the top-right element of the feedback matrices on subcarrier -23.

At a more mathematically precise level, which is contained within the visualization, the baseline mean for the West test location is $-0.1414 - 0.8181i$, and the motion-saturated mean marked for the same location in Figure 29 is $-0.1431 - 0.7941i$ in this particular feedback element. The motion-saturated mean is in fact within the first standard deviation of the baseline plot, which demonstrates mathematically the characteristics of centrality and location. This shows that the location of a device remains influential even when the environment contains motion. It also suggests that location correlation may be better served by testing not only mean packet distances to the mean feedback matrix, but also by incorporating the sample mean feedback matrix into the tests themselves.

This suggests that an observer, in collecting and displaying the results of beam-forming feedback over time, may be able to distinguish visually between devices at different locations, as well as between different states of the environment in which those devices reside, and that further analysis of data visualization, and more comprehensive visualization tools may support this hypothesis.

5.3 FCaE Training

5.3.1 Baseline Data Analysis

The FCaE is trained using the baseline captures for each location. The output of the training is a distribution of mean packet distances, for 100 4-packet trials, from each location’s mean feedback matrices, and the mean feedback matrix for each. Since the expected data for the actual packet distances is non-normal, as seen in the plot of Figure 30, the FCaE training uses bootstrapping to resample the data and provide means, which are approximately normal according to the Central Limit Theorem, as discussed in Section 3.4.3. The original data confirms the expected left-skew of packet distance due to the non-negative nature of Euclidean distances.

The approximately normal distributions of packet distance means for all 4 cardinal directions are displayed in Figure 31. These plots are normal probability plots, which arrange the data in a set along a line which models what a normal distribution would be expected to look like. By calculating the mean and standard deviation of the data set, the normal probability plot creates a line which shows the probability under the corresponding normal curve for different data values. The plot then orders all the data elements of the set, and assumes their probability value based on their index in the set. The least numerical value is given the left-most probability, and the greatest numerical value is given the right-most probability. This becomes the y-axis value of the data point. Its x-axis value is its numerical value. Should the data set be normal, the data points will be plotted closely along the line, neither trending above nor below it for extended amounts of time. Should the data be non-normal, it will be evident in the plot that it does not follow the normal line, but deviates from it consistently.

The plots show that the original data deviates following a left-skew. After re-sampling to produce a distribution of mean packet distances, the adjusted normal probability plot shows that the new distribution is approximately normal. This normality of the adjusted data validates the use of t-tests and F-tests in the FCaE's testing capabilities.

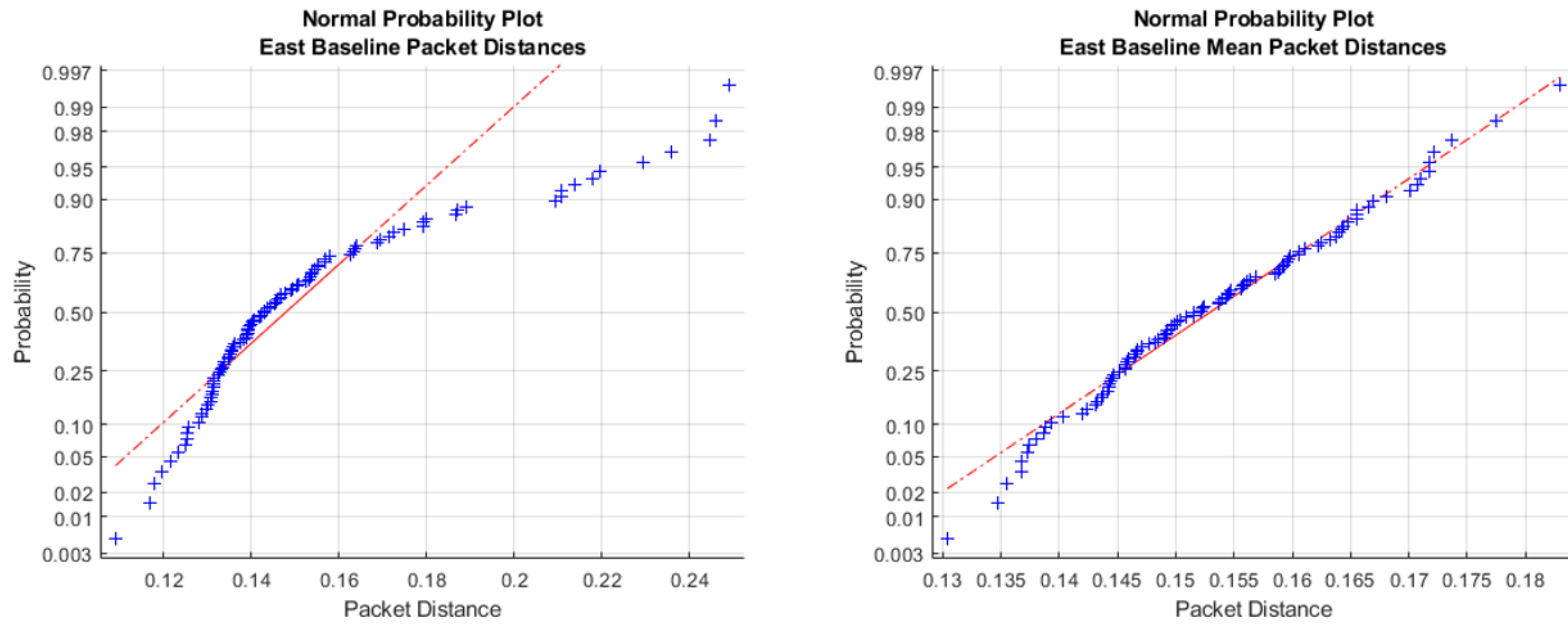


Figure 30: The normal probability plot of the baseline individual packet distances for the East location, compared to the normal probability plot of the mean packet distances. The individual packet data, as expected, is non-normal and skewed left. However, after bootstrapping and using means, approximate normality holds due to the central limit theorem.

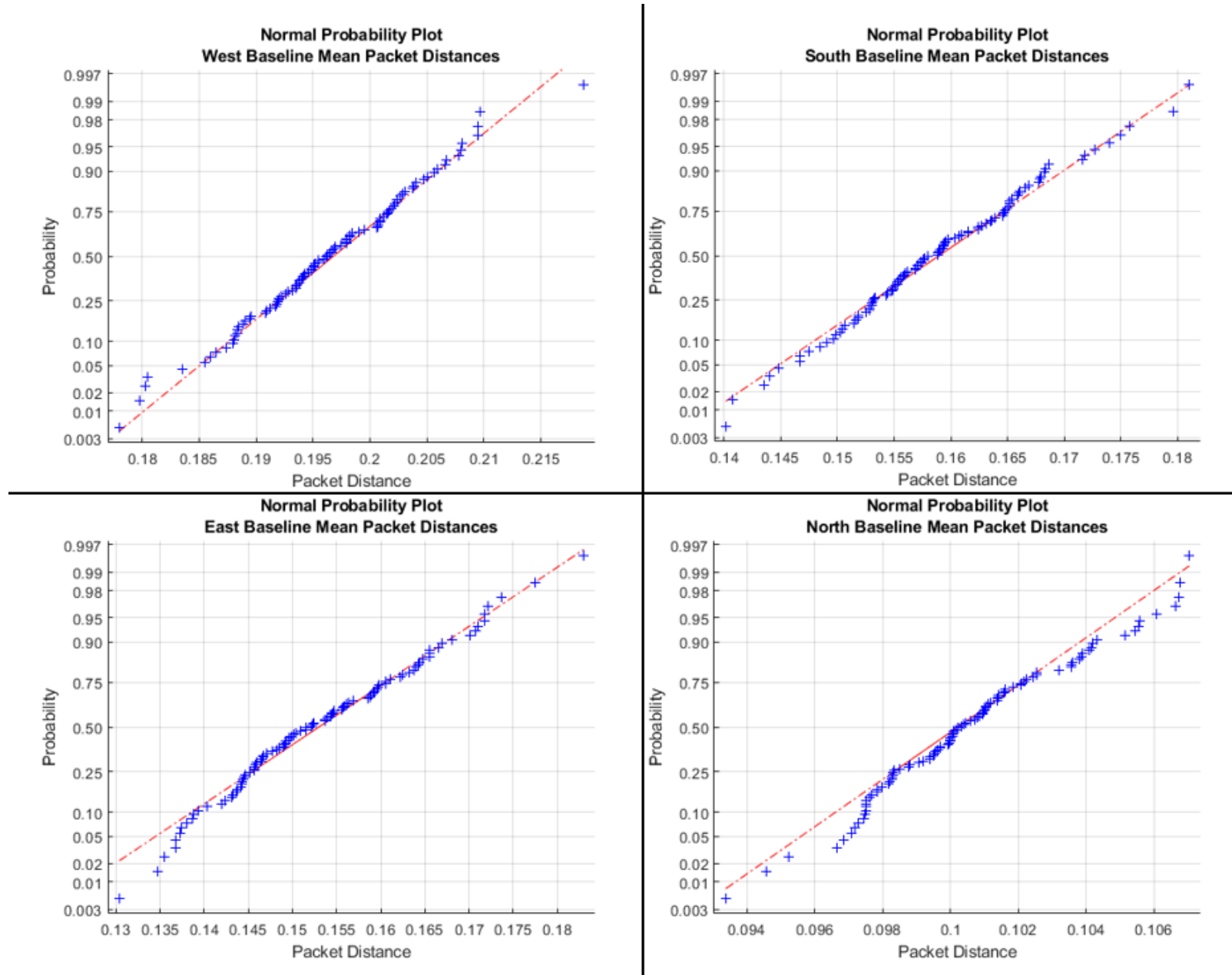


Figure 31: The normal probability plots of the baseline mean packet distances in the 4 cardinal directions. These demonstrate the approximate normality of the mean packet distance data.

These packet distances, paired with the mean feedback matrices for each location, provide the mechanism that the FCaE uses to evaluate further test data, and correlate location.

5.3.2 Distinguishing Success Rate (DSR):

The DSR is evaluated by testing groups of trial means from within the baselines themselves. The expectation is that most test groups will be found within their baseline location, and will be found outside of other locations. The threshold for the t-test used is 0.0001, to ensure definitive rejection of the null hypothesis from a baseline group, and account for potential variation in the feedback data, as described in Section 3.4.4. Table 3 shows the number of test groups from each baseline that were successfully identified as within the baseline, and outside of every other baseline. There were 25 total test groups for each baseline. This leads to a DSR of 98.5%. This means that the FCaE should succeed approximately 98.5% of the time at both rejecting a test group from a location it does not belong to, as well as failing to reject it from the location it does belong to.

The high success rate of the FCaE on the baseline data strongly supports the first hypothesis of this research, which is that device locations are distinguishable from one another through the feedback reporting mechanism of 802.11ac's beamforming feature. The strict requirements of the rejection test for alternative baselines, plus the previously shown minor deviation in the feedback (Figure 28) help reinforce this conclusion. Beamforming feedback data provides a logical, and leaked, representation of physical information about the network, and distinct positions can be successfully identified.

Table 3: Number of successful distinguishing between locations in 25 test groups from each baseline group.

Location	Trials	Successes	DSR
N	25	25	100%
NW	25	23	92%
W	25	25	100%
SW	25	25	100%
S	25	25	100%
SE	25	24	96%
E	25	25	100%
NE	25	25	100%

5.4 FCaE Minimal Motion Evaluation

5.4.1 Location Correlation True Positive Rate (LCTPR)

The LCTPR is established the same way as the DSR, by performing the same test on new data, taken from the minimal motion capture. The new data under investigation was collected while the client device remained in the same location, and the motion in the environment was minimal, as during the baseline. Table 4 shows the LCTP counts for each location from the test data out of 25 trials drawn from each location.

The LCTPR calculated from this is 50% overall. There is significant variation in the LCTPR between different locations. In particular, the NW test resulted in 100% ability to confirm the location against the baseline, while the NE test resulted in 0% positive reports. The explanations for these variations are location-dependent, and can be better explained by considering the feedback data itself. In the NE location, the distinction between the three captures: baseline, minimal motion, and motion-saturated, is presented in Figure 32. While the baseline capture has an extremely small variance, and the motion-saturated caption has a large variance, the minimal motion test capture has a small variance that is still visibly larger than the baseline,

Table 4: Number of true positives from each test group in the minimal motion environment captures from each location.

Location	Trials	True Positives	LCTPR
N	25	8	32%
NW	25	25	100%
W	25	15	60%
SW	25	7	28%
S	25	24	96%
SE	25	15	60%
E	25	6	24%
NE	25	0	0%

and thus each unique data point in it is not near enough to the baseline mean to be considered as valid in the test against it. This is confirmed when evaluating the mean packet distance of the baseline to the minimal motion test group, which are 0.1301 and 0.2616 respectively. Part of the reason for this may be due to the location itself, which has an uncontrollable air current that causes minor motion during some periods. This also suggests an answer to the previously unknown effect of air motion on beamforming feedback. It is minor, but visible and statistically significant to the FCaE’s test.

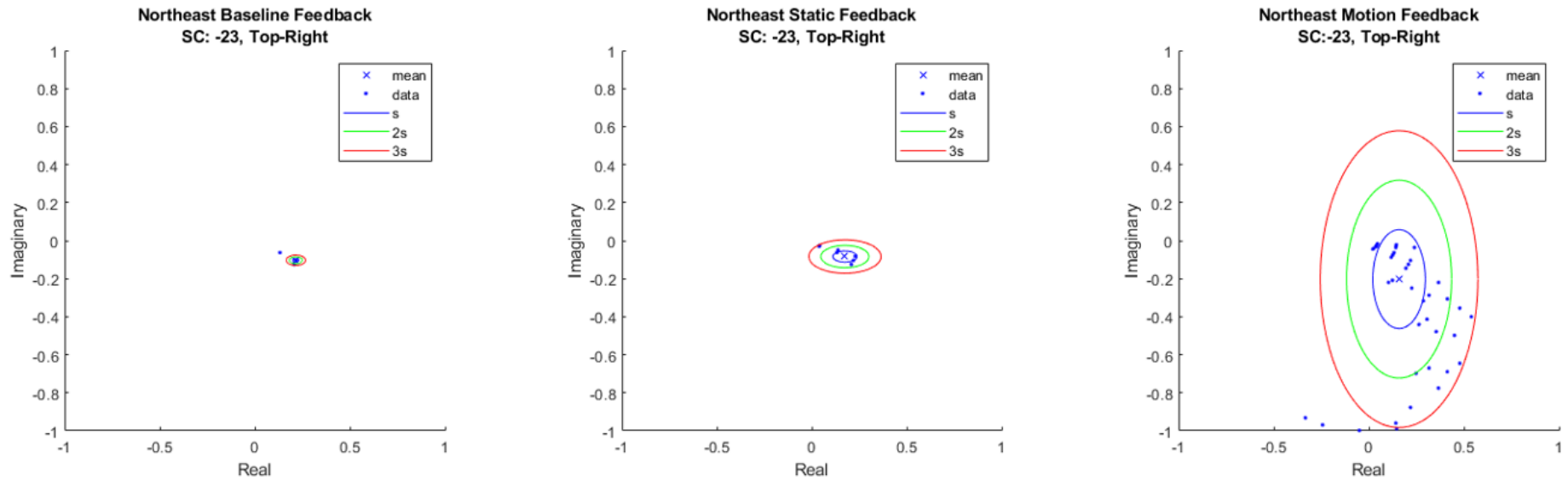


Figure 32: The comparison of feedback between the baseline, minimal motion test, and motion test captures at the NE test location. As seen, the minimal motion test capture has a slightly larger variance than the baseline test capture, but much smaller than the motion capture.

The reliability of the data is further evidenced in the fact that the means and zones of the feedback between the minimal motion test and baseline are still in close proximity to each other, and that some of the feedback from the minimal motion test falls within the baseline’s area on the plot. The conclusion of this is that while the FCaE provides a generally reliable method for distinguishing positions, its applicability is limited when the baseline is “too good” and constant, suggesting that training with a baseline under some small motion in the environment is a better method for being able to provide a reliable measure. A brief comparison of treating the baseline capture as the test group and the minimal motion evaluation capture as a training baseline confirms this, with the results in described in Table 5 for each of the locations that had under 50% true positives in the original test.

The result of this change is to bring the total LCTPR up to 83.5%. An additional suggestion is that the oversight of similar data, with a similar mean in the feedback, is the result of the final test being reduced to a one-dimensional result, instead of taking into account all 968 different feedback numbers in one report, and that one result being based on distance to a mean, instead of incorporating the mean itself at some level. A suggestion for future tests is then to implement either a more complicated test, which will be more accurate in handling all the dimensions of feedback data, or to implement a test where instead of a distances are used to form a mean, the test group’s mean matrix is calculated, and it’s distance to the baseline mean matrix is given, which would help account for situations where the variance difference is significant enough to be detected by the existing test, but should not be significant enough to warrant rejection, such as in Figure 32.

Table 5: The results for N, SW, E and NE when using the baseline capture as the test, and the minimal motion evaluation capture as the baseline.

Location	Trials	True Positives	LCTPR
N	25	19	76%
SW	25	25	100%
E	25	20	80%
NE	25	24	96%

5.4.2 Location Correlation False Positive Rate (LCFPR)

The false positive rate is describing the amount of test groups in which the incorrect location baseline is not rejected. There is a potential for up to 7 false positives per test group in this work’s experiment, due to there being 8 baselines which are tested against, and only one of them is correct. Throughout the experiment with minimal motion test groups compared to the baseline, there were no false positives. This yielded an LCFPR of 0%. The implication of this is that depending on the quality of baseline data, the FCaE may do a better job or not of considering test data within the location it belongs, it does a perfect job of rejecting test data from membership in a location to which it does not belong. This also provides further evidence for the unique nature of feedback to a physical location, and demonstrates the soundness of the hypothesis in correlating beamforming feedback to location. Since the threshold for the t-test used in comparing groups is also lower than typical (0.0001), it follows that the uniqueness is sufficient enough and distant enough between positions that false positives are not a risk due to the low threshold.

5.4.3 Location Correlation False Negative Rate (LCFNR)

The false negative rate describes the frequency of errors in distinction. Since the errors in distinction is similar to the converse of the DSR, which was shown to be 98.5% in Section 5.3.2, the LCFNR should be expected to be close to 1.5%. This is

because the converse of the DSR is the rate at which distinction fails, and includes when false distinctions of location are made. The rates for each location are shown in Table 6, and lead to a final LCFNR of 6.5%. If the same adjustments are made as in Section 5.4.1, when accounting for the low LCTPR in some areas, the final LCFNR goes down to 2.2% which is extremely close to the predicted 1.5% based on the DSR.

The false negatives in some data is to be expected due to minor variations present even in the baseline capture, as seen in the less than 100% DSR calculated in Section 5.3.2. However, the adjusted LCFNR of 2.2% suggests that the FCaE under proper baselines, is useful at identifying the location of new device data, and if sufficient data is collected, the LCFNR shows that there should still remain the ability to correlate location. Combined with the LCFPR of 0% in an environment without significant motion, such identifications by the model should also be considered reliable when they occur. The overall conclusion of the LCTPR, LCFPR, and LCFNR, is that the FCaE can successfully correlate and evaluate the position of individual devices on an 802.11ac network through the analysis of beamforming feedback information, when there is minimal motion in a wireless environment.

Table 6: The false negative rates for the test groups against the baselines at each location.

Location	Total Negatives	False Negatives	LCFNR
N	192	17	8.9%
NW	175	0	0.0%
W	185	10	5.4%
SW	193	18	9.3%
S	176	1	0.5%
SE	185	10	5.4%
E	194	19	9.8%
NE	200	25	12.5%

5.5 FCaE Motion-Saturated Evaluation

5.5.1 Motion Correlation True Positive Rate (MCTPR)

The MCTPR describes how often the FCaE fails to reject the correct location for a capture when there is significant motion in the environment. The results are shown in Table 7. The overall MCTPR is 0.5%. This demonstrates that the FCaE’s scheme for correlating location is not effective when there is significant motion in the environment. As discussed when addressing the problems with certain locations in Section 5.4.1, the deficiency in the FCaE’s analysis that makes it ineffective for this situation is the absence of accounting for the mean feedback, and only accounting for the distance from the mean. This makes the FCaE effective at correlating feedback data when the variance in the data is similar, but ineffective when the variance is not. Two test packets which lie in opposite directions of the mean according to the complex feedback plot, at 4 standard deviations each away, could in fact indicate that they belong to the same location as that mean. However, the mean distance does not account for this balancing, and remains outside of the acceptable zone for the FCaE’s analysis. The situation is considered exactly the same by the FCaE as that in which the two test packets are at the same distance from the mean, but right next to each other.

The result of this is that the FCaE is sufficient to account for the location, provided the variance in feedback remains similar to that of the baseline, and that it should be able to distinguish between different states of the environment between captures, if the location is unchanged, as discussed in Section 5.6. However, since the variance from the mean is fundamental to the FCaE’s analysis, the FCaE, while being able to distinguish states of the environment, is poor at accounting for such distinctions and maintaining an accurate analysis which is not covered by the baseline. Section 5.7 discusses some implications of using a different baseline further.

Table 7: The MCTPR describing the success of the FCaE at correlating feedback captured in an environment with motion, to a baseline captured in an environment without motion.

Location	Trials	True Positives	MCTPR
N	25	0	0%
NW	25	0	0%
W	25	1	4%
SW	25	0	0%
S	25	0	0%
SE	25	0	0%
E	25	0	0%
NE	25	0	0%

5.5.2 Motion Correlation False Positive Rate (MCFPR)

The MCFPR describes how often the FCaE incorrectly identifies a test group from an environment with motion as belonging to a location other than the one it is at. In every test, there were no false positives. While the FCaE analysis is poor at correctly identifying the locations of test groups in an environment with motion, due to the small variance from mean feedback observed in the baseline capture, this also makes it efficient at rejecting test groups from positions to which they are not at.

5.5.3 Motion Correlation False Negative Rate (MCFNR)

The MCFNR describes how often the FCaE rejects falsely instead of truly. The results are in Table 8, and result in a total MCFNR of 12.5%. This reinforces the discussion in Section 5.5.1.

A possible avenue of approach is to sample the test group, and take the mean feedback of the sample, and test the distance of that mean feedback to the baseline mean, instead of the individual test group packets. This produces a distance of the mean, instead of a mean of the distance. Using this alternate test produces the results in Table 9. This shows some improvement, but not enough to enable the

Table 8: The false negatives in testing the motion capture against the baseline.

Location	Trials	False Negatives	MCFNR
N	200	25	12.5%
NW	200	25	12.5%
W	199	24	12.1%
SW	200	25	12.5%
S	200	25	12.5%
SE	200	25	12.5%
E	200	25	12.5%
NE	200	25	12.5%

Table 9: The resulting rates evaluating the motion capture against the baseline from the alternate test using the packet distance of the mean, instead of the mean of the packet distance.

Location	Trials	MCTPR	MCFPR	MCFNR
N	25	0%	0%	12.5%
NW	25	12%	0%	11.2%
W	25	32%	0%	8.9%
SW	25	0%	0%	12.5%
S	25	0%	0%	12.5%
SE	25	0%	0%	12.5%
E	25	0%	0%	12.5%
NE	25	0%	0%	12.5%

FCaE to reliably predict position for an environment with motion, when the baseline was captured without motion. The conclusion of the alternative test means that the suggestion of incorporating the mean feedback value, instead of only distances to it, is a useful way to address the limitations of the FCaE’s analysis, but requires further investigation and development in determining quality tests to apply.

5.6 FCaE Minimal Motion and Motion-Saturated Environment Analysis

Due to the variations in mean packet distances between the minimal motion and motion environments, and the severe change in the FCaE’s reliability between them, the FCaE also compares the variance in test captures to determine if the environment contains motion or not. This uses a 2-sample F-test between trials of test groups sampled from the minimal motion capture and the motion capture. The trials use their own capture as a baseline to determine the mean packet distance, as if they used a common point it would not show any information about the variance within the captures themselves. The results of the test, at a threshold of 0.01 are shown in Table 10. The rate at which the FCaE is able to successfully distinguish the motion and minimal motion environment captures through the packet distance variance is the Environment Distinction Rate (EDR).

The overall EDR is 75%. This shows that the FCaE’s ability to distinguish between environments in motion and environments at rest is moderately reliable, but could use improvement. Every location tested had at least one success, which indicates that with enough sample data, even for difficult to distinguish environments, the FCaE can successfully show a difference. The lowest two locations for the EDR between minimal motion test and motion test captures are an interesting correlation to the failure of the FCaE to identify the location in the minimal motion test for those locations in Section 5.4.1. The low rate of distinguishing between the minimal motion

Table 10: The FCaE’s success at distinguishing between the minimal motion and motion test captures for each location, based on the variance of the mean packet distance.

Location	Trials	Successful Distinctions	EDR
N	25	25	100%
NW	25	25	100%
W	25	19	76%
SW	25	17	68%
S	25	25	100%
SE	25	25	100%
E	25	11	44%
NE	25	3	12%

and motion test environment captures here is consistent with the observed spread of the feedback in Figure 32. To confirm the relationship, the baseline is compared to the test captures the results of the EDR test are shown in Table 11.

The high distinction rate between the baseline and minimal motion test captures at the SW, E, and NE test locations is consistent with the observed behavior in Section 5.4.1, where the failure of the FCaE to achieve a high correlation for these locations was based on the higher variance in the feedback from the test environment than in the baseline.

The differences at the North test location require deeper investigation and explanation. The variances in the feedback between the minimal motion test and baseline captures are less distinguishable, which implies that the failure of the original correlation test must be due to another factor. Both of the captures have a very low variance among themselves, which makes the differences in means more significant. For some feedback elements, there was a variance of 0, due to the feedback remaining constant throughout the entire test. This allows minor variations to appear more significant. An example of the differing means is shown in Figure 33. As illustrated, the baseline has a small variance, as does the minimal motion test. However, the variation shown in the minimal motion test suggests that the outlier on the chart occurs somewhat

Table 11: The environment distinction test applied between the baseline capture and the test captures. The baseline compared to the minimal motion test is in the middle two columns, while the baseline compared to the motion test is in the right two columns.

Location	Trials	Distinctions _{minimalmotion}	EDR	Distinctions _{motion}	EDR
N	25	4	16%	25	100%
NW	25	16	64%	17	68%
W	25	0	0%	25	100%
SW	25	15	60%	25	100%
S	25	0	0%	25	100%
SE	25	2	8%	25	100%
E	25	22	88%	10	40%
NE	25	25	100%	25	100%

commonly, thus allowing test samples from the capture to often have a test group mean which departs from the local space of the baseline capture’s mean.

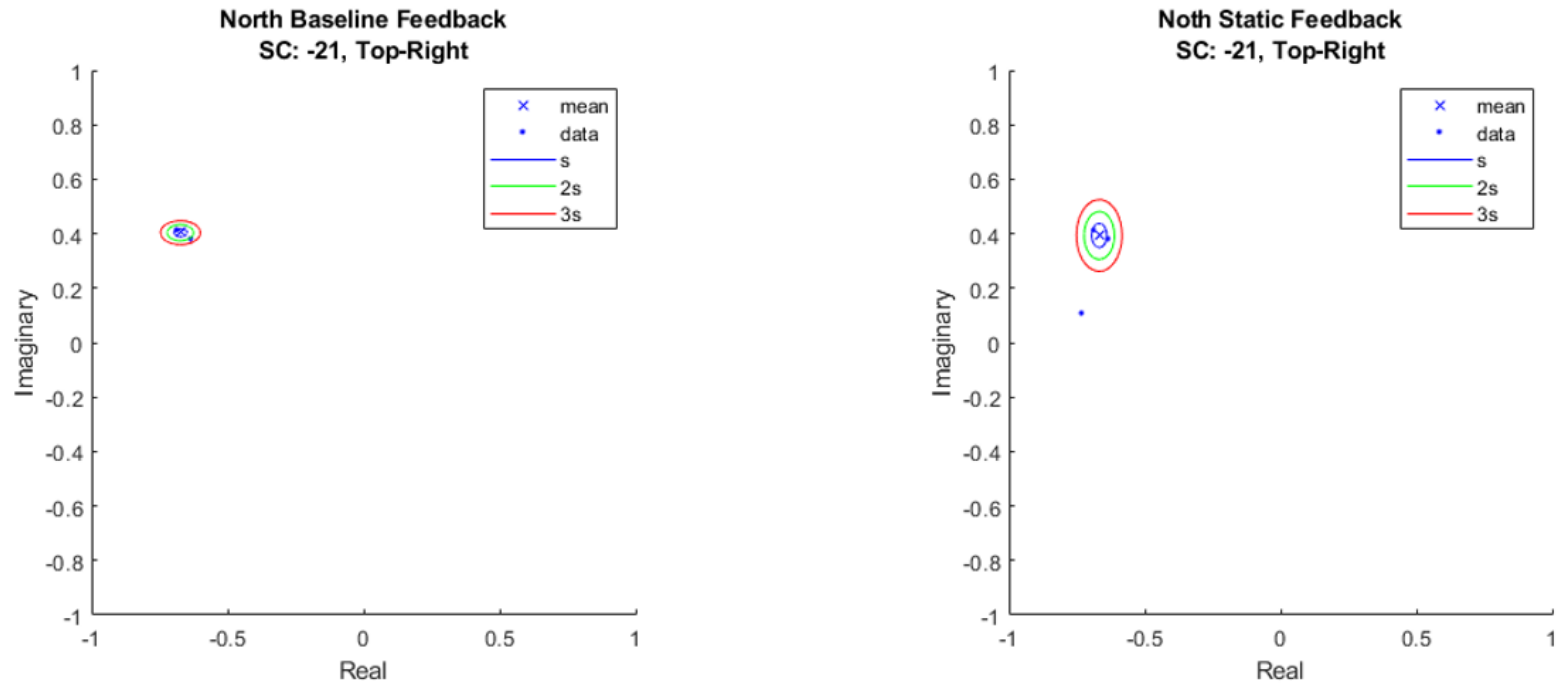


Figure 33: The baseline and minimal motion test captures have feedback with similar variance, but outliers with significant enough frequency in the minimal motion test capture allow a sample mean which falls outside of the baseline's parameters.

The conclusion of this test is that the FCaE is particularly helpful at distinguishing different states in the environment, and in doing so sheds greater light on previous results correlating locations. The low rate of correlation between the baseline and the minimal motion test in the NE location is paired with an almost certain distinction between the baseline and minimal motion test environments through the variance test, and a low rate of distinction between the minimal motion test and motion test environments. Possible avenues for revising the model may look at the magnitude of variance, and performing a more sophisticated test with larger data sets.

5.7 Motion-Saturated Feedback as a Baseline

The final exploration of this research consists in analyzing the behavior of the FCaE, when a motion-saturated feedback capture is used for training the baseline.

5.7.1 Baseline Accuracy

The DSR test is performed by evaluating the motion test capture against its own baseline. The results are displayed in Table 12. Recalling that the DSR test only counts a success as a true positive without any false positives ($N_{success} = |\{TP : \text{thesetoftestswheretherewasatruepositive}\} - \{FP : \text{thesetoftestswheretherewasafalsepositive}\}|$), there are less successes than only true positives. Looking more closely at the tests shows that the every trial resulted in a true positive. The overall DSR is 53%. This demonstrates the combined effect of what was observed at the baseline DSR, a distinct correlation between location and feedback, as well as the effect of the environment's motion which causes a wider variance in feedback. Thus, more data was accepted than in the few trials during the baseline test which were rejected.

However, due to the same variation, there are now more overlap between different locations, which causes a high false positive rate. Table 13 shows which trial loca-

Table 12: The DSR for using the motion test capture as a baseline.

Location	Trials	TP	TP with FP $\neq 0$	Successes	DSR
N	25	25	24	1	4%
NW	25	25	0	25	100%
W	25	25	0	25	100%
SW	25	25	21	4	16%
S	25	25	0	25	100%
SE	25	25	24	1	4%
E	25	25	25	0	0%
NE	25	25	0	25	100%

Table 13: The locations which provided false positives for trials under the motion baseline.

Location	False Positive Locations
N	SW
SW	N
SE	NE
E	NE

tions were correlated with which baseline locations. The explanation of this is not exclusively in the higher variance, but also in the use of multipath routing of signals when faced with obstacles to LOS. In particular, the NE location appears as if it was particularly favored in choosing new paths to the E and SE trial locations. The reason for this could be either location dependent, or it could speak to the nature of the data collected during the motion test itself at the NE position. However, the fact that the NE location is not universally a false positive seems to indicate that it was instead the best path chosen for the signals to travel to the other locations.

The conclusion is that the FCaE may still distinguish positions, when the baseline is taken with regular motion in the environment, but it is much less effective than when the baseline is taken without motion in the environment. The effectiveness of distinguishing locations dropped almost to half of what it was in the original baseline. A revised test may use a higher threshold that makes rejecting a test group of packets from the motion baseline easier.

5.7.2 Minimal Motion Evaluation

The LCTPR and LCFPR under the new baseline are shown in Table 14 and Table 15 respectively. As expected from the DSR test, while there several true positives, there are also many false positives. The final LCTPR is 37.5%, and the final LCFPR is 64.6%. Remembering that the LCFPR is out of the total amount of positives, and the LCTPR is out of the total trials conducted, this shows that the use of the motion test capture as the baseline is ineffective for training the FCaE. An interesting pattern is that as with the DSR test, a significant amount of false positives were matched to the Northeast trial location.

Since the variance in feedback is wider in the motion capture, there must be a reason why the minimal motion test capture did not produce more true positives.

Table 14: The true positives for the minimal motion test capture tested against a baseline trained on the motion capture.

Location	Trials	True Positives	LCTPR
N	25	0	0%
NW	25	0	0%
W	25	0	0%
SW	25	25	100%
S	25	25	100%
SE	25	0	0%
E	25	0	0%
NE	25	25	100%

Table 15: The false positives for the minimal motion test capture tested against a baseline trained on the motion capture.

Location	Total Positives	False Positives	LCFPR
N	25	25	100%
NW	0	0	0%
W	0	0	0%
SW	62	37	59.7%
S	25	0	0%
SE	25	25	100%
E	25	25	100%
NE	50	25	50%

The reason is precisely that variance, and the same reason for why the original test did not produce a high MCTPR: the number which is measured and tested is mean packet distance from the baseline, not distance from the baseline to the mean of the packets.

The conclusion from using the motion capture as a baseline is that it inherently increases the risk of false positives due to the greater spread of its feedback, which is witnessed in the preceding data, and also inherently decreases the chance of true positives, which is demonstrated in the preceding analysis. Because the existing analysis mechanisms of the FCaE consider distance to the mean feedback, multiple points which are within a small area may be falsely identified with a different baseline, simply because they are the same distance away from its mean as its data. They also will not be truly identified with the baseline to which they belong, even if they coincide with the true baseline's mean itself, because they have a much lower distance to that mean than any of the baseline's data.

VI. Conclusion

6.1 Overview

This chapter summarizes the experiment of this research and its results. Section 6.2 discusses the conclusions drawn from the statistical analysis of the data, and provides a final synthesis of the results. Section 6.3 describes the applications and import of the research contained in this work. Section 6.4 describes the limitations of the tools and results provided by this work, and explores possible avenues for transcending them. Section 6.5 discusses future research that can be performed using the FCaE and its tools.

6.2 Research Conclusions

This research began with three hypotheses about the nature of information leakage contained in 802.11ac beamforming feedback. This section reviews each hypothesis in turn, and synthesizes the results.

6.2.1 Device Location Correlation

The first hypothesis was that a significant correlation could be detected between the feedback matrices in 802.11ac beamforming, and the physical location of a device on the 802.11ac network. Using the analysis in the FCaE, this hypothesis was sustained by this work. The feedback matrices had a discernible pattern based on their location, and the DSR of the baseline model was 98.5%. The FCaE successfully determined the pattern, and showed that even with an alpha value as low as 0.0001, it could definitively reject the null hypothesis for a location that the device under investigation did not belong to. The FCaE also had difficulty in accounting for outliers though, which resulted several test samples not being successfully identified.

The hypothesis was further sustained by the testing of the trained FCaE against new data collected from devices in the same test locations. The variation in the new data provided a challenge which led to only a 50% LCTPR against the trained baseline. However, when a new baseline was provided with a slightly higher variance than the original baseline, the revised LCTPR was 83.5%.

This indicated the quality of baseline data that is required for the FCaE to work optimally: it must have non-minimal variation, as minimal variation does not allow for an effective means by which data can be identified with the correct location. However, in every trial performed there were no false positives, and for every location except one, there was at least one true positive, providing further evidence of the hypothesis, and demonstrating the utility of the FCaE in evaluating data.

6.2.2 Location Correlation During Motion

The second hypothesis was that a correlation of feedback matrices to location in a motion-saturated environment would not be successful. Specifically, that different locations would not be successfully distinguished from one another. Out of all the trials performed using the original analysis portion of the FCaE, only one true positive was identified. However, no false positives were identified. This shows that while the FCaE trained on a baseline in a motionless environment cannot reliably determine location in a motion-saturated environment, when it does identify the location, it does not do so incorrectly.

The failure of the FCaE's analysis in the motion-saturated environment was due to the reduction of dimensions in the feedback data to a single metric of packet distance to the mean of the feedback from the baseline training. The relatively high packet distances to the mean in a motion-saturated environment disguises when the mean of the test data is identical to that of the training data. Therefore, an alternate

test was devised to reduce the magnitude of the test data's packet distances, by determining a trial mean feedback, and using the distance of means, instead of the mean of distances. This alternate evaluation method improved the FCaE's ability to handle data from a motion-saturated environment from a 0.5% MCTPR, to a 5.5% MCTPR. There also remained no false positives in any trial. This shows that while an improvement, alternative evaluation methods still require investigation and the FCaE cannot be treated as an effective identifier of device location in a motion-saturated environment, when it was trained on a baseline from a motionless environment.

6.2.3 Distinguishing Environmental States

The third hypothesis was that the feedback collected from a motionless environment would differ significantly from that collected in a motion-saturated environment, and that the FCaE could perform a test to successfully distinguish these two environments based on feedback captures from each. The FCaE performed well, and had a 75% EDR, demonstrating that the differing environment states could be measured by the variance in a device's feedback. With more motion comes more variance, and with less motion comes less variance.

The test also provided insight into previous data analysis, and a measure by which qualitative analysis on test data evaluated against a baseline could be done. An inverse correlation was observed between the reliability of the FCaE's evaluation tests and the amount of difference detected between the environmental states of the baseline and test data. The more similar to the baseline that the test data was in terms of environmentally influenced variation, the more reliable the ability of the FCaE to correctly distinguish the location which the test data corresponded to.

6.2.4 Conclusions from Feedback Analysis

The analysis and evaluation tools and methods of the FCaE were valuable in proving the correlation between beamforming feedback and device location. Even more valuable was the absence of false positives from the use of the training baseline, even with a very low alpha value. There were several characteristics of feedback data that were highlighted throughout the experiment and results analysis.

1. **Repetition:** The feedback for a location, even in the motion-saturated environment, showed a penchant for repetition, including returning to previous values on some subcarriers after being disturbed. This shows that the feedback is relatively stable for a location. This further confirms that the feedback's correlation with physical location is a reliable source of information.
2. **Variance:** The feedback's variance for a location had a proportional relationship to the amount of motion in the environment. The less motion, the more consistent the feedback was, but the more motion, the more spread it spread out when plotted. This confirms that the feedback not only correlates to the location, but also the state of the environment, and the activity in it over time.
3. **Distance:** The feedback's distance to a mean was a useful measure for determining consistency and evaluating test location. However, since the packet distance to a mean is a reduction of 1936 dimensions into 1, it has limited utility or accuracy, and thus limits the situations where it can be reliably used. The distance also has the further limitation of being a left-skewed non-normal distribution. Adjusted to describe the mean distance of samples allows a useful approximation of normality. The variance in the distance is related to the variance in the feedback, and variance in the feedback often produces variance in packet distance. There is a better correlation between the variance in the

feedback and the mean packet distance instead, as variance in the feedback produces different packet distances, but not necessarily different packet distance variances. The mean packet distance in a motion-saturated environment is higher than that in a motionless environment.

4. **Mean:** The feedback mean is a useful measure for determining the uniqueness of a location. It provides a method of maintaining all 1936 dimensions of feedback data, as a standard against which to compare. It also remains within a small zone for the same location, irrespective of the motion in the environment. Accounting for the mean within the evaluation method improves the accuracy of the FCaE a small amount when investigating motion-saturated feedback data.

6.3 Contributions

This work contributes tools and methods for extracting and analyzing 802.11ac beamforming feedback. The FPX, and FCaE, as well as suggestions for applying them to related research.

6.3.1 FPX

The FPX is a dedicated set of tools for extracting and decoding beamforming feedback. It is specifically coded to a common situation of 80 MHz, 2x2 MIMO, and SU beamforming. However, it is easy to adapt to other configurations, or generalize to automatically adapt to them. This tool opens access to a new class of data, that was previously unanalyzed. In particular, collecting Wi-Fi CSI required specifically designed equipment or modified firmware. This work makes the already existing and broadcast CSI from 802.11ac beamforming feedback readily available for analysis.

6.3.2 FCaE

The FCaE provides a method for training, and then evaluating 802.11ac feedback. It also provides a method for comparing feedback captures, and determining the differences in environment motion between them. The framework contains helpful functions and tools for visualizing and comparing feedback groups, and is easily expandable with additional tests. It is also the first tool to use beamforming feedback to determine physical configuration information about a network.

6.3.3 Applications

The analysis in the FCaE can be applied for purposes in sensing and pattern analysis.

- **Sensing:** Determining differences and characteristics in an environment over time, through only passively observing 802.11ac signals. In particular, the FCaE provides a method for analyzing and visualizing feedback from different events and times. Also, the FPX allows sensing to be done without dedicated sensors in designed locations.
- **Pattern Analysis:** Previous work with pattern analysis through 802.11 has shown that spoofing signals from nonexistent devices can effectively fool the analysis. The FCaE provides a method of identifying unique devices through beamforming feedback, because of the relation between feedback and location. This allows spoofing devices to be discovered and eliminated from consideration, and pattern analysis at higher levels to continue unhindered. Further, the use of the FCaE opens up new data to incorporate into pattern analysis.

6.4 Limitations

The FPX and FCaE, while useful and having a generally reliable analysis, have several limitations which may be improved on in future work.

- **Training Baseline:** The training baseline data limits the accuracy and flexibility of the model. While a motionless environment is more ideal than a motion-saturated environment for collecting the baseline, to prevent false positives, as discussed in Section 5.7, there must also be some amount of variance in the baseline. A minimally variant baseline is unable to produce true positives at the rate it should, as discussed in Section 5.4.1. The baseline must be “good” but not “too good” to be of use in evaluation.
- **Motion-Saturated Environments:** The FCaE did not have good results in the motion-saturated environment. While the true positives were entirely correct, there was originally only 1 before the revised evaluation, and only 11 afterwards, out of 200 trials.
- **Packet Distance:** The current evaluation methods of the FCaE examine and test the packet distance to a mean feedback value. This metric is effective when the environments between the baseline and the sample are similar, and the baseline has a low variance. However, it is limited by its definition to discount the sample mean, or the relationship between sample feedback. Feedback that is spread out, or clustered together, can look identical under this test, resulting especially in a high false positive rate when the baseline has significant variance, as explained in Section 5.7.
- **Channel Character:** The FPX and FCaE were designed to specifically investigate the 80 MHz, 2x2 MIMO, SU beamformee channel. As currently composed,

they do not handle other channels, although the techniques and strategies for doing so are analogous to those used.

6.5 Future Work

The research in this work may be extended in several ways. These include:

1. **Generalization:** The FPX and FCaE are currently limited to one channel characteristic. The tools may be revised and extended such that 1) .pcap files may be directly handled, 2) multiple phases of the FPX are unnecessary, 3) more channel characteristics are defined.
2. **Multi-User MIMO:** The experiments in this research were conducted with SU beamforming, and further work may be done to examine the area of MU beamforming in MU-MIMO configurations, especially where a network is saturated with more devices requesting beamforming than it is able to handle simultaneously.
3. **Extension:** The FCaE is the beginning of analyzing beamforming feedback data, and provides beginning tools of analyzing and visualizing it. More methods of analysis exist, and can be added, and more methods of visualization may be developed. A specific suggestion is a visualization method for displaying the change of mean feedback over time.
4. **Evaluation Methods:** The existing evaluation in the FCaE is useful, but limited. Future work may investigate further methods of evaluating data, or a multi-part test which examines several individual characteristics of data to make a conclusion. Machine learning may also provide additional tools in the analysis and evaluation of beamforming feedback. Exploration of methods to visualize

feedback data, especially over time and over changing location, may also provide insights into the utility of capturing information from this information leakage.

5. **Devices in Motion:** As the beamforming feedback is related to physical location, is device tracking of mobile devices possible? Future work may investigate and answer this question. If it is, an investigation could also be made into possible patterns in the correlation of feedback to location. Future work may also address the issue of how feedback is affected through walls.
6. **Sensing Uses:** The differences between motion-saturated and motionless environments also display a use of the FCaE in sensing, and possible applications to already existing traditional problems such as using CSI to characterize types of motion. Future work may investigate and compare the quality of CSI sensing using beamforming feedback to previously developed methods of CSI sensing.
7. **Pattern-of-Life Analysis:** The FCaE can provide additional data and patterns to use in a pattern-of-life model, and in particular may be used to defeat proposed countermeasures to previous Wi-Fi-based analysis utilizing spoofed devices. Future work may incorporate these analysis tools into a new pattern detection model.

Appendix A. Alternate Adapter Configuration Commands

Due to variations in virtual machine and hardware setups, this alternative command sequence may be required for devices in which the method used in this research fails to properly configure the monitor mode adapter. These commands must be run as root, or with root privileges.

```
1 ifconfig wlan0 down
2 iwconfig wlan0 mode monitor
3 iw dev wlan0 set freq 5805
4 ifconfig wlan0 up
```

Appendix B. FPX-AD Generated C Code Excerpt

```
1 /* tmp.c
2  * This is an auto-generated C program utilized in the FPX-AD
3  * to input and output the binary data from wireshark's packet
4  * exporter into a MATLAB code file.
5  *
6  * The program is set for "NEC", or the "North Evaluation
7  * Capture" which is the motionless environment test data not
8  * collected for the baseline, at the North test location
9  */
10 #include <stdio.h>
11 #include <stdlib.h>
12
13
14 // This imports the raw data from Wireshark
15 #include "nec.c"
16
17 int main()
18 {
19
20     FILE *f = fopen("nec_export.m", "wb");
21     fprintf(f,"nec = [");
22
23     unsigned char phi = 0;
24     unsigned char psi = 0;
25     unsigned int ph = 0;
26     unsigned int ps = 0;
27     int sc = 0;
28     int tracker = 0;
29     unsigned char *tmp = NULL;
30
```

```

31  // The feedback matrixes start at 0x3F into the packet
32  // and the data is named "pkt#" automatically
33  tmp = pkt1 + 0x3F;
34
35  // Tracks feedback matrices, since each is 10 bits, and
36  // determines which bits to read
37  tracker = 0;
38  sc = 0;
39  fprintf(f, "[" );
40  while(sc != 234)
41  {
42      // Read the correct bits for the angles of this subcarrier
43      switch(tracker)
44      {
45          case 0:
46              phi = (tmp[0]>>2);
47              psi = ((tmp[0]%4)<<2) + (tmp[1]>>6);
48              break;
49          case 1:
50              phi = (tmp[1]%64);
51              psi = (tmp[2]>>4);
52              break;
53          case 2:
54              phi = ((tmp[2]%16)<<2) + (tmp[3]>>6);
55              psi = (tmp[3]>>2)%16;
56              break;
57          case 3:
58              phi = ((tmp[3]%4)<<2) + (tmp[4]>>4);
59              psi = (tmp[4]%16);
60      }
61
62      // Print the decoded angles (bit form, not radians) into MATLAB

```

```

63     fprintf(f, "%u, %u", phi, psi);
64
65     sc += 1;
66
67     // If only a subcarrier, print as a new row, if a new packet,
end the matrix
68     if(sc != 234) { fprintf(f, "; "); }
69     else { fprintf(f, "] "); }
70     tracker += 1;
71     tracker %= 4;
72
73     // Increment the address if needed (4 matrices = 40 bits = 5
bytes)
74     if(tracker == 0) { tmp += 5; }
75 }
76 printf("Exported Packet %d\n", 1);
77
78 // Proceed to the next packet
79 tmp = pkt2 + 0x3F;
80 tracker = 0;
81 sc = 0;
82 //...
83 //...
84 //...
85 printf("Exported Packet %d\n", 100);
86
87
88 // Print closing data to MATLAB script, including saving the
capture data
89 // to a MATLAB .mat data file
90 fprintf(f, "];\n\n");
91 fprintf(f, "save('nec.mat', 'nec')");

```

```
92  fclose(f);
93
94  // Now the generated MATLAB .m code is run, and the MATLAB
    readable
95  // data file is produced for use
96 }
```

Appendix C. FPX-AD Python Script

```
1 # Feedback Transform Script Generator
2 # This program generates a .c program to read, transform, and
   reprint feedback
3 # matrices from the export as C arrays option in Wireshark.
4
5 import sys
6 import os
7
8 # usage: python FPX-AD.py <wireshark export filename> <number of
   packets>
9 # filename is given without ".c" ending
10
11 Target_File = sys.argv[1]
12 N_Packets = sys.argv[2]
13
14 # Create temporary C program to import, analyze, and export data
   from binary form.
15
16 Program = """
17 #include <stdio.h>
18 #include <stdlib.h>
19
20 """
21
22 Program += "#include \"" + Target_File + ".c\"\\n\\n"
23
24 Program += """int main()
25 {
26
27     FILE *f = fopen("" + "\"" + Target_File + "_export.m\"" + "\", "
```

```

        wb");
28  fprintf(f, "" + "\"" + Target_File + " = [\");" + ""
29
30  unsigned char phi = 0;
31  unsigned char psi = 0;
32  unsigned int ph = 0;
33  unsigned int ps = 0;
34  int sc = 0;
35  int tracker = 0;
36  unsigned char *tmp = NULL;
37
38  ""
39
40  # Add a section for each packet in the data file (each named with
        format "pkt#")
41
42  for i in range(int(N_Packets)):
43      Program += ""    tmp = pkt"" + str(i + 1) + "" + 0x3F;
44      tracker = 0;
45      sc = 0;
46      fprintf(f, "[");
47      while(sc != 234)
48      {
49          switch(tracker)
50          {
51              case 0:
52                  phi = (tmp[0]>>2);
53                  psi = ((tmp[0]%4)<<2) + (tmp[1]>>6);
54                  break;
55              case 1:
56                  phi = (tmp[1]%64);
57                  psi = (tmp[2]>>4);

```

```

58     break;
59     case 2:
60         phi = ((tmp[2]%16)<<2) + (tmp[3]>>6);
61         psi = (tmp[3]>>2)%16;
62         break;
63     case 3:
64         phi = ((tmp[3]%4)<<2) + (tmp[4]>>4);
65         psi = (tmp[4]%16);
66     }
67
68     fprintf(f, "%u, %u", phi, psi);
69
70     sc += 1;
71     if(sc != 234) { fprintf(f, "; "); }
72     else { fprintf(f, "] "); }
73     tracker += 1;
74     tracker %= 4;
75     if(tracker == 0) { tmp += 5; }
76 }
77 printf"" + "("Exported Packet %d\\n\\n", " + str(i+1) + ")" + "";
78
79 ""
80
81 Program += ""
82
83 fprintf(f, "];\\n\\n");
84 fprintf(f, "save('"" + Target_File + """.mat', '"" + Target_File
    + ""')");
85 fclose(f);
86 }
87 ""
88

```

```
89 # write program
90
91 f = open("tmp.c", "w")
92 f.write(Program)
93 f.close()
94
95 # compile and execute program to decode the binary data into MATLAB
   usable form.
96
97 os.system("gcc -o tmp tmp.c -Wno-discarded-qualifiers && ./tmp && rm
   tmp");
```

Appendix D. FPX-DCD Bash Script

```
1 #!/bin/bash
2
3 python FXP-AD.py nbc 100
4 python FXP-AD.py nec 100
5 python FXP-AD.py nmc 100
6
7 python FXP-AD.py sbc 100
8 python FXP-AD.py sec 100
9 python FXP-AD.py smc 100
10
11 python FXP-AD.py ebc 100
12 python FXP-AD.py eec 100
13 python FXP-AD.py emc 100
14
15 python FXP-AD.py wbc 100
16 python FXP-AD.py wec 100
17 python FXP-AD.py wmc 100
18
19 python FXP-AD.py nebc 100
20 python FXP-AD.py neec 100
21 python FXP-AD.py nemc 100
22
23 python FXP-AD.py nwbc 100
24 python FXP-AD.py nwec 100
25 python FXP-AD.py nwmc 100
26
27 python FXP-AD.py sebc 100
28 python FXP-AD.py seec 100
29 python FXP-AD.py semc 100
30
```

```
31 python FXP-AD.py swbc 100
```

```
32 python FXP-AD.py swec 100
```

```
33 python FXP-AD.py swmc 100
```

Appendix E. FPX-DCP MATLAB Script

```
1 %% Feedback Packet Extractor - Adaptive decompressor (FPX-AP)
2 % Input is a filename formatted using feedback formatter python
   script,
3 % which outputs the feedback in a form for MATLAB to access.
4
5 %% Load data from export
6 load('wbc.mat');
7 load('wec.mat');
8 load('wmc.mat');
9 load('nwbc.mat');
10 load('nwec.mat');
11 load('nwmc.mat');
12 load('swbc.mat');
13 load('swec.mat');
14 load('swmc.mat');
15 load('ebc.mat');
16 load('eec.mat');
17 load('emc.mat');
18 load('nebc.mat');
19 load('neec.mat');
20 load('nemc.mat');
21 load('sebc.mat');
22 load('seec.mat');
23 load('semc.mat');
24 load('sbc.mat');
25 load('sec.mat');
26 load('smc.mat');
27 load('nbc.mat');
28 load('nec.mat');
29 load('nmc.mat');
```

```

30
31
32 %% Decompress matrices
33 west_bcap = capture_decode(wbc);
34 west_ecap = capture_decode(wec);
35 west_mcap = capture_decode(wmc);
36
37 northwest_bcap = capture_decode(nwbc);
38 northwest_ecap = capture_decode(nwec);
39 northwest_mcap = capture_decode(nwmc);
40
41 north_bcap = capture_decode(nbc);
42 north_ecap = capture_decode(nec);
43 north_mcap = capture_decode(nmc);
44
45 northeast_bcap = capture_decode(nebc);
46 northeast_ecap = capture_decode(neec);
47 northeast_mcap = capture_decode(nemc);
48
49 east_bcap = capture_decode(ebc);
50 east_ecap = capture_decode(eec);
51 east_mcap = capture_decode(emc);
52
53 southeast_bcap = capture_decode(sebc);
54 southeast_ecap = capture_decode(seec);
55 southeast_mcap = capture_decode(semc);
56
57 south_bcap = capture_decode(sbc);
58 south_ecap = capture_decode(sec);
59 south_mcap = capture_decode(sm);
60
61 southwest_bcap = capture_decode(swbc);

```

```

62 southwest_ecap = capture_decode(swec);
63 southwest_mcap = capture_decode(swmc);
64
65
66 %% save decompressed matrices
67 save('decompressed_feedback.mat', 'west_bcap', 'west_ecap', '
    west_mcap', 'northwest_bcap', 'northwest_ecap', 'northwest_mcap',
    'north_bcap', 'north_ecap', 'north_mcap', 'northeast_bcap', '
    northeast_ecap', 'northeast_mcap', 'east_bcap', 'east_ecap', '
    east_mcap', 'southeast_bcap', 'southeast_ecap', 'southeast_mcap',
    'south_bcap', 'south_ecap', 'south_mcap', 'southwest_bcap', '
    southwest_ecap', 'southwest_mcap');

```

Appendix F. FPX Utility Functions

```
1 function [psi] = psi_decode(k,b)
2 %psi_decode decodes the angle psi into radians
3 %   Follows the formula in 802.11 to decode the angle from its bit
   form
4 %   into radians
5 psi = k*pi/(2^(b + 1)) + pi/(2^(b+2));
6 end

1 function [phi] = phi_decode(k,b)
2 %phi_decode decodes the angle phi into radians
3 %   Follows the formula in 802.11 to decode the angle from its bit
   form
4 %   into radians
5 phi = k*pi/(2^(b - 1)) + pi/(2^b);
6 end

1 function [feedback] = feedback_decode(phi, psi)
2 %feedback_decode Decompresses a single subcarrier's feedback matrix
   from
3 %angle form into numerical form.
4
5 feedback = [exp(1i*phi)*cos(psi);
6   exp(1i*phi)*sin(psi);
7   -sin(psi);
8   cos(psi)]';
9 end

1 function [packet] = packet_decode(packetin)
2 %packet_decode Decodes each subcarrier's feedback matrix in a packet
3
4
```

```

5 n_sc = size(packetin, 1)
6 packet = zeros(n_sc, 4);
7 for i = 1:n_sc
8     hq = packetin(i, 1);
9     sq = packetin(i, 2);
10    packet(i, 1:4) = feedback_decode(phi_decode(hq, 6), psi_decode(
        sq, 4));
11 end
12
13 end

1 function [capture] = capture_decode(capturein)
2 %capture_decode This decodes an entire capture of packets
3 % Captures come in 234 x (2 * n_pkt) collections of angles. To
4 % decompress them, every packet gets decompressed and put into 234
   % x 4
5 % values, instead of 234 x 2, and therefore a matrix of size 234 x
   % (4 *
6 % n_pkt) is outputted as the decompressed capture.
7 n_pkt = size(capturein, 2)/2;
8 capture = zeros(234, 4 * n_pkt);
9 for i = 1:n_pkt
10    tmp_pkt = capturein(1:234, 2*i-1:2*i);
11    capture(1:234, 4*i-3:4*i)=packet_decode(tmp_pkt);
12 end
13 end

```

Appendix G. FCaE MATLAB Script

```
1 %% FCaE: Feedback Correlation and Evaluation
2
3 %% Load the data
4 load('decompressed_feedback.mat');
5
6 %% TRAINING: Analyze the baseline data
7
8 [nb_dist, nb_dev, nb_mm, nb_dm] = analyze(north_bcap);
9 [nwb_dist, nwb_dev, nwb_mm, nwb_dm] = analyze(northwest_bcap);
10 [neb_dist, neb_dev, neb_mm, neb_dm] = analyze(northeast_bcap);
11 [sb_dist, sb_dev, sb_mm, sb_dm] = analyze(south_bcap);
12 [swb_dist, swb_dev, swb_mm, swb_dm] = analyze(southwest_bcap);
13 [seb_dist, seb_dev, seb_mm, seb_dm] = analyze(southeast_bcap);
14 [wb_dist, wb_dev, wb_mm, wb_dm] = analyze(west_bcap);
15 [eb_dist, eb_dev, eb_mm, eb_dm] = analyze(east_bcap);
16
17 %% DSR Test: analyze packet groups from baseline
18 % The baseline capture is tested against the baseline training model
    % from each location, and the number of identifications with that
    % model is displayed.
19 dsr_north = sum([
20     group_test(north_bcap, nb_mm, nb_dist);
21     group_test(north_bcap, nwb_mm, nwb_dist);
22     group_test(north_bcap, wb_mm, wb_dist);
23     group_test(north_bcap, swb_mm, swb_dist);
24     group_test(north_bcap, sb_mm, sb_dist);
25     group_test(north_bcap, seb_mm, seb_dist);
26     group_test(north_bcap, eb_mm, eb_dist);
27     group_test(north_bcap, neb_mm, neb_dist);
28 ], 2)
```

29

```
30 dsr_northwest = sum([
31     group_test(northwest_bcap, nb_mm, nb_dist);
32     group_test(northwest_bcap, nwb_mm, nwb_dist);
33     group_test(northwest_bcap, wb_mm, wb_dist);
34     group_test(northwest_bcap, swb_mm, swb_dist);
35     group_test(northwest_bcap, sb_mm, sb_dist);
36     group_test(northwest_bcap, seb_mm, seb_dist);
37     group_test(northwest_bcap, eb_mm, eb_dist);
38     group_test(northwest_bcap, neb_mm, neb_dist);
39 ], 2)
```

40

```
41 dsr_west = sum([
42     group_test(west_bcap, nb_mm, nb_dist);
43     group_test(west_bcap, nwb_mm, nwb_dist);
44     group_test(west_bcap, wb_mm, wb_dist);
45     group_test(west_bcap, swb_mm, swb_dist);
46     group_test(west_bcap, sb_mm, sb_dist);
47     group_test(west_bcap, seb_mm, seb_dist);
48     group_test(west_bcap, eb_mm, eb_dist);
49     group_test(west_bcap, neb_mm, neb_dist);
50 ], 2)
```

51

```
52 dsr_southwest = sum([
53     group_test(southwest_bcap, nb_mm, nb_dist);
54     group_test(southwest_bcap, nwb_mm, nwb_dist);
55     group_test(southwest_bcap, wb_mm, wb_dist);
56     group_test(southwest_bcap, swb_mm, swb_dist);
57     group_test(southwest_bcap, sb_mm, sb_dist);
58     group_test(southwest_bcap, seb_mm, seb_dist);
59     group_test(southwest_bcap, eb_mm, eb_dist);
60     group_test(southwest_bcap, neb_mm, neb_dist);
```

```

61     ], 2)
62
63 dsr_south = sum([
64     group_test(south_bcap, nb_mm, nb_dist);
65     group_test(south_bcap, nwb_mm, nwb_dist);
66     group_test(south_bcap, wb_mm, wb_dist);
67     group_test(south_bcap, swb_mm, swb_dist);
68     group_test(south_bcap, sb_mm, sb_dist);
69     group_test(south_bcap, seb_mm, seb_dist);
70     group_test(south_bcap, eb_mm, eb_dist);
71     group_test(south_bcap, neb_mm, neb_dist);
72     ], 2)
73
74 dsr_southeast = sum([
75     group_test(southeast_bcap, nb_mm, nb_dist);
76     group_test(southeast_bcap, nwb_mm, nwb_dist);
77     group_test(southeast_bcap, wb_mm, wb_dist);
78     group_test(southeast_bcap, swb_mm, swb_dist);
79     group_test(southeast_bcap, sb_mm, sb_dist);
80     group_test(southeast_bcap, seb_mm, seb_dist);
81     group_test(southeast_bcap, eb_mm, eb_dist);
82     group_test(southeast_bcap, neb_mm, neb_dist);
83     ], 2)
84
85 dsr_east = sum([
86     group_test(east_bcap, nb_mm, nb_dist);
87     group_test(east_bcap, nwb_mm, nwb_dist);
88     group_test(east_bcap, wb_mm, wb_dist);
89     group_test(east_bcap, swb_mm, swb_dist);
90     group_test(east_bcap, sb_mm, sb_dist);
91     group_test(east_bcap, seb_mm, seb_dist);
92     group_test(east_bcap, eb_mm, eb_dist);

```

```

93     group_test(east_bcap, neb_mm, neb_dist);
94 ], 2)
95
96 dsr_northeast = sum([
97     group_test(northeast_bcap, nb_mm, nb_dist);
98     group_test(northeast_bcap, nwb_mm, nwb_dist);
99     group_test(northeast_bcap, wb_mm, wb_dist);
100    group_test(northeast_bcap, swb_mm, swb_dist);
101    group_test(northeast_bcap, sb_mm, sb_dist);
102    group_test(northeast_bcap, seb_mm, seb_dist);
103    group_test(northeast_bcap, eb_mm, eb_dist);
104    group_test(northeast_bcap, neb_mm, neb_dist);
105 ], 2)
106
107
108 %% Location Correlation Test
109 % This determines the LCTPR, LCFPR, and LCFNR. Every test capture
    is evaluated against the trained baseline model. The display is
    the amount of identifications made against each baseline model.
110
111 lct_north = sum([
112     group_test(north_ecap, nb_mm, nb_dist);
113     group_test(north_ecap, nwb_mm, nwb_dist);
114     group_test(north_ecap, wb_mm, wb_dist);
115     group_test(north_ecap, swb_mm, swb_dist);
116     group_test(north_ecap, sb_mm, sb_dist);
117     group_test(north_ecap, seb_mm, seb_dist);
118     group_test(north_ecap, eb_mm, eb_dist);
119     group_test(north_ecap, neb_mm, neb_dist);
120 ], 2)
121
122 lct_northwest = sum([

```

```

123     group_test(northwest_ecap, nb_mm, nb_dist);
124     group_test(northwest_ecap, nwb_mm, nwb_dist);
125     group_test(northwest_ecap, wb_mm, wb_dist);
126     group_test(northwest_ecap, swb_mm, swb_dist);
127     group_test(northwest_ecap, sb_mm, sb_dist);
128     group_test(northwest_ecap, seb_mm, seb_dist);
129     group_test(northwest_ecap, eb_mm, eb_dist);
130     group_test(northwest_ecap, neb_mm, neb_dist);
131 ], 2)
132
133 lct_west = sum([
134     group_test(west_ecap, nb_mm, nb_dist);
135     group_test(west_ecap, nwb_mm, nwb_dist);
136     group_test(west_ecap, wb_mm, wb_dist);
137     group_test(west_ecap, swb_mm, swb_dist);
138     group_test(west_ecap, sb_mm, sb_dist);
139     group_test(west_ecap, seb_mm, seb_dist);
140     group_test(west_ecap, eb_mm, eb_dist);
141     group_test(west_ecap, neb_mm, neb_dist);
142 ], 2)
143
144 lct_southwest = sum([
145     group_test(southwest_ecap, nb_mm, nb_dist);
146     group_test(southwest_ecap, nwb_mm, nwb_dist);
147     group_test(southwest_ecap, wb_mm, wb_dist);
148     group_test(southwest_ecap, swb_mm, swb_dist);
149     group_test(southwest_ecap, sb_mm, sb_dist);
150     group_test(southwest_ecap, seb_mm, seb_dist);
151     group_test(southwest_ecap, eb_mm, eb_dist);
152     group_test(southwest_ecap, neb_mm, neb_dist);
153 ], 2)
154

```

```

155 lct_south = sum([
156     group_test(south_ecap, nb_mm, nb_dist);
157     group_test(south_ecap, nwb_mm, nwb_dist);
158     group_test(south_ecap, wb_mm, wb_dist);
159     group_test(south_ecap, swb_mm, swb_dist);
160     group_test(south_ecap, sb_mm, sb_dist);
161     group_test(south_ecap, seb_mm, seb_dist);
162     group_test(south_ecap, eb_mm, eb_dist);
163     group_test(south_ecap, neb_mm, neb_dist);
164 ], 2)
165
166 lct_southeast = sum([
167     group_test(southeast_ecap, nb_mm, nb_dist);
168     group_test(southeast_ecap, nwb_mm, nwb_dist);
169     group_test(southeast_ecap, wb_mm, wb_dist);
170     group_test(southeast_ecap, swb_mm, swb_dist);
171     group_test(southeast_ecap, sb_mm, sb_dist);
172     group_test(southeast_ecap, seb_mm, seb_dist);
173     group_test(southeast_ecap, eb_mm, eb_dist);
174     group_test(southeast_ecap, neb_mm, neb_dist);
175 ], 2)
176
177 lct_east = sum([
178     group_test(east_ecap, nb_mm, nb_dist);
179     group_test(east_ecap, nwb_mm, nwb_dist);
180     group_test(east_ecap, wb_mm, wb_dist);
181     group_test(east_ecap, swb_mm, swb_dist);
182     group_test(east_ecap, sb_mm, sb_dist);
183     group_test(east_ecap, seb_mm, seb_dist);
184     group_test(east_ecap, eb_mm, eb_dist);
185     group_test(east_ecap, neb_mm, neb_dist);
186 ], 2)

```

```

187
188 lct_northeast = sum([
189     group_test(northeast_ecap, nb_mm, nb_dist);
190     group_test(northeast_ecap, nwb_mm, nwb_dist);
191     group_test(northeast_ecap, wb_mm, wb_dist);
192     group_test(northeast_ecap, swb_mm, swb_dist);
193     group_test(northeast_ecap, sb_mm, sb_dist);
194     group_test(northeast_ecap, seb_mm, seb_dist);
195     group_test(northeast_ecap, eb_mm, eb_dist);
196     group_test(northeast_ecap, neb_mm, neb_dist);
197 ], 2)
198
199 %% Motion Test
200 % The same procedure as above, but with the motion-saturated
    environment test capture instead of the motionless test capture.
201
202 mct_north = sum([
203     group_test(north_mcap, nb_mm, nb_dist);
204     group_test(north_mcap, nwb_mm, nwb_dist);
205     group_test(north_mcap, wb_mm, wb_dist);
206     group_test(north_mcap, swb_mm, swb_dist);
207     group_test(north_mcap, sb_mm, sb_dist);
208     group_test(north_mcap, seb_mm, seb_dist);
209     group_test(north_mcap, eb_mm, eb_dist);
210     group_test(north_mcap, neb_mm, neb_dist);
211 ], 2)
212
213 mct_northwest = sum([
214     group_test(northwest_mcap, nb_mm, nb_dist);
215     group_test(northwest_mcap, nwb_mm, nwb_dist);
216     group_test(northwest_mcap, wb_mm, wb_dist);
217     group_test(northwest_mcap, swb_mm, swb_dist);

```

```

218     group_test(northwest_mcap, sb_mm, sb_dist);
219     group_test(northwest_mcap, seb_mm, seb_dist);
220     group_test(northwest_mcap, eb_mm, eb_dist);
221     group_test(northwest_mcap, neb_mm, neb_dist);
222     ], 2)
223
224 mct_west = sum([
225     group_test(west_mcap, nb_mm, nb_dist);
226     group_test(west_mcap, nwb_mm, nwb_dist);
227     group_test(west_mcap, wb_mm, wb_dist);
228     group_test(west_mcap, swb_mm, swb_dist);
229     group_test(west_mcap, sb_mm, sb_dist);
230     group_test(west_mcap, seb_mm, seb_dist);
231     group_test(west_mcap, eb_mm, eb_dist);
232     group_test(west_mcap, neb_mm, neb_dist);
233     ], 2)
234
235 mct_southwest = sum([
236     group_test(southwest_mcap, nb_mm, nb_dist);
237     group_test(southwest_mcap, nwb_mm, nwb_dist);
238     group_test(southwest_mcap, wb_mm, wb_dist);
239     group_test(southwest_mcap, swb_mm, swb_dist);
240     group_test(southwest_mcap, sb_mm, sb_dist);
241     group_test(southwest_mcap, seb_mm, seb_dist);
242     group_test(southwest_mcap, eb_mm, eb_dist);
243     group_test(southwest_mcap, neb_mm, neb_dist);
244     ], 2)
245
246 mct_south = sum([
247     group_test(south_mcap, nb_mm, nb_dist);
248     group_test(south_mcap, nwb_mm, nwb_dist);
249     group_test(south_mcap, wb_mm, wb_dist);

```

```

250     group_test(south_mcap, swb_mm, swb_dist);
251     group_test(south_mcap, sb_mm, sb_dist);
252     group_test(south_mcap, seb_mm, seb_dist);
253     group_test(south_mcap, eb_mm, eb_dist);
254     group_test(south_mcap, neb_mm, neb_dist);
255 ], 2)
256
257 mct_southeast = sum([
258     group_test(southeast_mcap, nb_mm, nb_dist);
259     group_test(southeast_mcap, nwb_mm, nwb_dist);
260     group_test(southeast_mcap, wb_mm, wb_dist);
261     group_test(southeast_mcap, swb_mm, swb_dist);
262     group_test(southeast_mcap, sb_mm, sb_dist);
263     group_test(southeast_mcap, seb_mm, seb_dist);
264     group_test(southeast_mcap, eb_mm, eb_dist);
265     group_test(southeast_mcap, neb_mm, neb_dist);
266 ], 2)
267
268 mct_east = sum([
269     group_test(east_mcap, nb_mm, nb_dist);
270     group_test(east_mcap, nwb_mm, nwb_dist);
271     group_test(east_mcap, wb_mm, wb_dist);
272     group_test(east_mcap, swb_mm, swb_dist);
273     group_test(east_mcap, sb_mm, sb_dist);
274     group_test(east_mcap, seb_mm, seb_dist);
275     group_test(east_mcap, eb_mm, eb_dist);
276     group_test(east_mcap, neb_mm, neb_dist);
277 ], 2)
278
279 mct_northeast = sum([
280     group_test(northeast_mcap, nb_mm, nb_dist);
281     group_test(northeast_mcap, nwb_mm, nwb_dist);

```

```

282     group_test(northeast_mcap, wb_mm, wb_dist);
283     group_test(northeast_mcap, swb_mm, swb_dist);
284     group_test(northeast_mcap, sb_mm, sb_dist);
285     group_test(northeast_mcap, seb_mm, seb_dist);
286     group_test(northeast_mcap, eb_mm, eb_dist);
287     group_test(northeast_mcap, neb_mm, neb_dist);
288     ], 2)
289
290
291 %% Alternate MCT test
292 % This is the same as the previous, but uses the alternate metric
   adapted to account for the greater packet distance from motion in
   the environment. Positive results are counted and displayed.
293
294 mcta_north = sum([
295     alternate_group_test(north_mcap, nb_mm, nb_dist);
296     alternate_group_test(north_mcap, nwb_mm, nwb_dist);
297     alternate_group_test(north_mcap, wb_mm, wb_dist);
298     alternate_group_test(north_mcap, swb_mm, swb_dist);
299     alternate_group_test(north_mcap, sb_mm, sb_dist);
300     alternate_group_test(north_mcap, seb_mm, seb_dist);
301     alternate_group_test(north_mcap, eb_mm, eb_dist);
302     alternate_group_test(north_mcap, neb_mm, neb_dist);
303     ], 2)
304
305 mcta_northwest = sum([
306     alternate_group_test(northwest_mcap, nb_mm, nb_dist);
307     alternate_group_test(northwest_mcap, nwb_mm, nwb_dist);
308     alternate_group_test(northwest_mcap, wb_mm, wb_dist);
309     alternate_group_test(northwest_mcap, swb_mm, swb_dist);
310     alternate_group_test(northwest_mcap, sb_mm, sb_dist);
311     alternate_group_test(northwest_mcap, seb_mm, seb_dist);

```

```

312     alternate_group_test(northwest_mcap, eb_mm, eb_dist);
313     alternate_group_test(northwest_mcap, neb_mm, neb_dist);
314 ], 2)
315
316 mcta_west = sum([
317     alternate_group_test(west_mcap, nb_mm, nb_dist);
318     alternate_group_test(west_mcap, nwb_mm, nwb_dist);
319     alternate_group_test(west_mcap, wb_mm, wb_dist);
320     alternate_group_test(west_mcap, swb_mm, swb_dist);
321     alternate_group_test(west_mcap, sb_mm, sb_dist);
322     alternate_group_test(west_mcap, seb_mm, seb_dist);
323     alternate_group_test(west_mcap, eb_mm, eb_dist);
324     alternate_group_test(west_mcap, neb_mm, neb_dist);
325 ], 2)
326
327 mcta_southwest = sum([
328     alternate_group_test(southwest_mcap, nb_mm, nb_dist);
329     alternate_group_test(southwest_mcap, nwb_mm, nwb_dist);
330     alternate_group_test(southwest_mcap, wb_mm, wb_dist);
331     alternate_group_test(southwest_mcap, swb_mm, swb_dist);
332     alternate_group_test(southwest_mcap, sb_mm, sb_dist);
333     alternate_group_test(southwest_mcap, seb_mm, seb_dist);
334     alternate_group_test(southwest_mcap, eb_mm, eb_dist);
335     alternate_group_test(southwest_mcap, neb_mm, neb_dist);
336 ], 2)
337
338 mcta_south = sum([
339     alternate_group_test(south_mcap, nb_mm, nb_dist);
340     alternate_group_test(south_mcap, nwb_mm, nwb_dist);
341     alternate_group_test(south_mcap, wb_mm, wb_dist);
342     alternate_group_test(south_mcap, swb_mm, swb_dist);
343     alternate_group_test(south_mcap, sb_mm, sb_dist);

```

```

344     alternate_group_test(south_mcap, seb_mm, seb_dist);
345     alternate_group_test(south_mcap, eb_mm, eb_dist);
346     alternate_group_test(south_mcap, neb_mm, neb_dist);
347 ], 2)
348
349 mcta_southeast = sum([
350     alternate_group_test(southeast_mcap, nb_mm, nb_dist);
351     alternate_group_test(southeast_mcap, nwb_mm, nwb_dist);
352     alternate_group_test(southeast_mcap, wb_mm, wb_dist);
353     alternate_group_test(southeast_mcap, swb_mm, swb_dist);
354     alternate_group_test(southeast_mcap, sb_mm, sb_dist);
355     alternate_group_test(southeast_mcap, seb_mm, seb_dist);
356     alternate_group_test(southeast_mcap, eb_mm, eb_dist);
357     alternate_group_test(southeast_mcap, neb_mm, neb_dist);
358 ], 2)
359
360 mcta_east = sum([
361     alternate_group_test(east_mcap, nb_mm, nb_dist);
362     alternate_group_test(east_mcap, nwb_mm, nwb_dist);
363     alternate_group_test(east_mcap, wb_mm, wb_dist);
364     alternate_group_test(east_mcap, swb_mm, swb_dist);
365     alternate_group_test(east_mcap, sb_mm, sb_dist);
366     alternate_group_test(east_mcap, seb_mm, seb_dist);
367     alternate_group_test(east_mcap, eb_mm, eb_dist);
368     alternate_group_test(east_mcap, neb_mm, neb_dist);
369 ], 2)
370
371 mcta_northeast = sum([
372     alternate_group_test(northeast_mcap, nb_mm, nb_dist);
373     alternate_group_test(northeast_mcap, nwb_mm, nwb_dist);
374     alternate_group_test(northeast_mcap, wb_mm, wb_dist);
375     alternate_group_test(northeast_mcap, swb_mm, swb_dist);

```

```

376     alternate_group_test(northeast_mcap, sb_mm, sb_dist);
377     alternate_group_test(northeast_mcap, seb_mm, seb_dist);
378     alternate_group_test(northeast_mcap, eb_mm, eb_dist);
379     alternate_group_test(northeast_mcap, neb_mm, neb_dist);
380 ], 2)
381
382
383
384 %% Environment Distinguishing:
385 % This runs the test for environment distinguishing between
      different captures. The test captures are compared first, then
      the motion capture against the baseline, and the baseline against
      the motionless test. The number of positive distinctions are
      counted and displayed for each location.
386
387 em_north = sum(environment_detect(north_ecap, north_mcap))
388 em_northwest = sum(environment_detect(northwest_ecap, northwest_mcap
      ))
389 em_west = sum(environment_detect(west_ecap, west_mcap))
390 em_southwest = sum(environment_detect(southwest_ecap, southwest_mcap
      ))
391 em_south = sum(environment_detect(south_ecap, south_mcap))
392 em_southeast = sum(environment_detect(southeast_ecap, southeast_mcap
      ))
393 em_east = sum(environment_detect(east_ecap, east_mcap))
394 em_northeast = sum(environment_detect(northeast_ecap, northeast_mcap
      ))
395
396 bm_north = sum(environment_detect(north_bcap, north_mcap))
397 bm_northwest = sum(environment_detect(northwest_bcap, northwest_mcap
      ))
398 bm_west = sum(environment_detect(west_bcap, west_mcap))

```

```

399 bm_southwest = sum(environment_detect(southwest_bcap, southwest_mcap
    ))
400 bm_south = sum(environment_detect(south_bcap, south_mcap))
401 bm_southeast = sum(environment_detect(southeast_bcap, southeast_mcap
    ))
402 bm_east = sum(environment_detect(east_bcap, east_mcap))
403 bm_northeast = sum(environment_detect(northeast_bcap, northeast_mcap
    ))
404
405 be_north = sum(environment_detect(north_ecap, north_bcap))
406 be_northwest = sum(environment_detect(northwest_ecap, northwest_bcap
    ))
407 be_west = sum(environment_detect(west_ecap, west_bcap))
408 be_southwest = sum(environment_detect(southwest_ecap, southwest_bcap
    ))
409 be_south = sum(environment_detect(south_ecap, south_bcap))
410 be_southeast = sum(environment_detect(southeast_ecap, southeast_bcap
    ))
411 be_east = sum(environment_detect(east_ecap, east_bcap))
412 be_northeast = sum(environment_detect(northeast_ecap, northeast_bcap
    ))
413
414
415
416
417 %% RE-TRAINING: using the motion as a baseline
418
419 [n_mdlist, ~, n_mmm] = analyze(north_mcap);
420 [s_mdlist, ~, s_mmm] = analyze(south_mcap);
421 [e_mdlist, ~, e_mmm] = analyze(east_mcap);
422 [w_mdlist, ~, w_mmm] = analyze(west_mcap);
423 [nw_mdlist, ~, nw_mmm] = analyze(northwest_mcap);

```

```

424 [ne_mdist, ~, ne_mmm] = analyze(northeast_mcap);
425 [sw_mdist, ~, sw_mmm] = analyze(southwest_mcap);
426 [se_mdist, ~, se_mmm] = analyze(southeast_mcap);
427
428 %% DSR Test
429 % The same test as the original DSR is run, with the new baseline
430
431 mct_north = sum([
432     group_test(north_mcap, n_mmm, n_mdist);
433     group_test(north_mcap, nw_mmm, nw_mdist);
434     group_test(north_mcap, w_mmm, w_mdist);
435     group_test(north_mcap, sw_mmm, sw_mdist);
436     group_test(north_mcap, s_mmm, s_mdist);
437     group_test(north_mcap, se_mmm, se_mdist);
438     group_test(north_mcap, e_mmm, e_mdist);
439     group_test(north_mcap, ne_mmm, ne_mdist);
440 ], 2)
441
442 mct_northwest = sum([
443     group_test(northwest_mcap, n_mmm, n_mdist);
444     group_test(northwest_mcap, nw_mmm, nw_mdist);
445     group_test(northwest_mcap, w_mmm, w_mdist);
446     group_test(northwest_mcap, sw_mmm, sw_mdist);
447     group_test(northwest_mcap, s_mmm, s_mdist);
448     group_test(northwest_mcap, se_mmm, se_mdist);
449     group_test(northwest_mcap, e_mmm, e_mdist);
450     group_test(northwest_mcap, ne_mmm, ne_mdist);
451 ], 2)
452
453 mct_west = sum([
454     group_test(west_mcap, n_mmm, n_mdist);
455     group_test(west_mcap, nw_mmm, nw_mdist);

```

```

456     group_test(west_mcap, w_mmm, w_mdistr);
457     group_test(west_mcap, sw_mmm, sw_mdistr);
458     group_test(west_mcap, s_mmm, s_mdistr);
459     group_test(west_mcap, se_mmm, se_mdistr);
460     group_test(west_mcap, e_mmm, e_mdistr);
461     group_test(west_mcap, ne_mmm, ne_mdistr);
462 ], 2)
463
464 mct_southwest = sum([
465     group_test(southwest_mcap, n_mmm, n_mdistr);
466     group_test(southwest_mcap, nw_mmm, nw_mdistr);
467     group_test(southwest_mcap, w_mmm, w_mdistr);
468     group_test(southwest_mcap, sw_mmm, sw_mdistr);
469     group_test(southwest_mcap, s_mmm, s_mdistr);
470     group_test(southwest_mcap, se_mmm, se_mdistr);
471     group_test(southwest_mcap, e_mmm, e_mdistr);
472     group_test(southwest_mcap, ne_mmm, ne_mdistr);
473 ], 2)
474
475 mct_south = sum([
476     group_test(south_mcap, n_mmm, n_mdistr);
477     group_test(south_mcap, nw_mmm, nw_mdistr);
478     group_test(south_mcap, w_mmm, w_mdistr);
479     group_test(south_mcap, sw_mmm, sw_mdistr);
480     group_test(south_mcap, s_mmm, s_mdistr);
481     group_test(south_mcap, se_mmm, se_mdistr);
482     group_test(south_mcap, e_mmm, e_mdistr);
483     group_test(south_mcap, ne_mmm, ne_mdistr);
484 ], 2)
485
486 mct_southeast = sum([
487     group_test(southeast_mcap, n_mmm, n_mdistr);

```

```

488     group_test(southeast_mcap, nw_mmm, nw_mdist);
489     group_test(southeast_mcap, w_mmm, w_mdist);
490     group_test(southeast_mcap, sw_mmm, sw_mdist);
491     group_test(southeast_mcap, s_mmm, s_mdist);
492     group_test(southeast_mcap, se_mmm, se_mdist);
493     group_test(southeast_mcap, e_mmm, e_mdist);
494     group_test(southeast_mcap, ne_mmm, ne_mdist);
495     ], 2)
496
497 mct_east = sum([
498     group_test(east_mcap, n_mmm, n_mdist);
499     group_test(east_mcap, nw_mmm, nw_mdist);
500     group_test(east_mcap, w_mmm, w_mdist);
501     group_test(east_mcap, sw_mmm, sw_mdist);
502     group_test(east_mcap, s_mmm, s_mdist);
503     group_test(east_mcap, se_mmm, se_mdist);
504     group_test(east_mcap, e_mmm, e_mdist);
505     group_test(east_mcap, ne_mmm, ne_mdist);
506     ], 2)
507
508 mct_northeast = sum([
509     group_test(northeast_mcap, n_mmm, n_mdist);
510     group_test(northeast_mcap, nw_mmm, nw_mdist);
511     group_test(northeast_mcap, w_mmm, w_mdist);
512     group_test(northeast_mcap, sw_mmm, sw_mdist);
513     group_test(northeast_mcap, s_mmm, s_mdist);
514     group_test(northeast_mcap, se_mmm, se_mdist);
515     group_test(northeast_mcap, e_mmm, e_mdist);
516     group_test(northeast_mcap, ne_mmm, ne_mdist);
517     ], 2)
518
519

```

```

520 %% New Location Correlation Test
521 % The same evaluation metrics as before are run against the new
    baseline. This produces a new LCTPR, LCFPR, and LCFNR. Positive
    results are indicated.
522
523
524 lctm_north = sum([
525     group_test(north_ecap, n_mmm, n_mdistr);
526     group_test(north_ecap, nw_mmm, nw_mdistr);
527     group_test(north_ecap, w_mmm, w_mdistr);
528     group_test(north_ecap, sw_mmm, sw_mdistr);
529     group_test(north_ecap, s_mmm, s_mdistr);
530     group_test(north_ecap, se_mmm, se_mdistr);
531     group_test(north_ecap, e_mmm, e_mdistr);
532     group_test(north_ecap, ne_mmm, ne_mdistr);
533 ], 2)
534
535 lctm_northwest = sum([
536     group_test(northwest_ecap, n_mmm, n_mdistr);
537     group_test(northwest_ecap, nw_mmm, nw_mdistr);
538     group_test(northwest_ecap, w_mmm, w_mdistr);
539     group_test(northwest_ecap, sw_mmm, sw_mdistr);
540     group_test(northwest_ecap, s_mmm, s_mdistr);
541     group_test(northwest_ecap, se_mmm, se_mdistr);
542     group_test(northwest_ecap, e_mmm, e_mdistr);
543     group_test(northwest_ecap, ne_mmm, ne_mdistr);
544 ], 2)
545
546 lctm_west = sum([
547     group_test(west_ecap, n_mmm, n_mdistr);
548     group_test(west_ecap, nw_mmm, nw_mdistr);
549     group_test(west_ecap, w_mmm, w_mdistr);

```

```

550     group_test(west_ecap, sw_mmm, sw_mdists);
551     group_test(west_ecap, s_mmm, s_mdists);
552     group_test(west_ecap, se_mmm, se_mdists);
553     group_test(west_ecap, e_mmm, e_mdists);
554     group_test(west_ecap, ne_mmm, ne_mdists);
555 ], 2)
556
557 lctm_southwest = sum([
558     group_test(southwest_ecap, n_mmm, n_mdists);
559     group_test(southwest_ecap, nw_mmm, nw_mdists);
560     group_test(southwest_ecap, w_mmm, w_mdists);
561     group_test(southwest_ecap, sw_mmm, sw_mdists);
562     group_test(southwest_ecap, s_mmm, s_mdists);
563     group_test(southwest_ecap, se_mmm, se_mdists);
564     group_test(southwest_ecap, e_mmm, e_mdists);
565     group_test(southwest_ecap, ne_mmm, ne_mdists);
566 ], 2)
567
568 lctm_south = sum([
569     group_test(south_ecap, n_mmm, n_mdists);
570     group_test(south_ecap, nw_mmm, nw_mdists);
571     group_test(south_ecap, w_mmm, w_mdists);
572     group_test(south_ecap, sw_mmm, sw_mdists);
573     group_test(south_ecap, s_mmm, s_mdists);
574     group_test(south_ecap, se_mmm, se_mdists);
575     group_test(south_ecap, e_mmm, e_mdists);
576     group_test(south_ecap, ne_mmm, ne_mdists);
577 ], 2)
578
579 lctm_southeast = sum([
580     group_test(southeast_ecap, n_mmm, n_mdists);
581     group_test(southeast_ecap, nw_mmm, nw_mdists);

```

```

582     group_test(southeast_ecap, w_mmm, w_mdists);
583     group_test(southeast_ecap, sw_mmm, sw_mdists);
584     group_test(southeast_ecap, s_mmm, s_mdists);
585     group_test(southeast_ecap, se_mmm, se_mdists);
586     group_test(southeast_ecap, e_mmm, e_mdists);
587     group_test(southeast_ecap, ne_mmm, ne_mdists);
588     ], 2)
589
590 lctm_east = sum([
591     group_test(east_ecap, n_mmm, n_mdists);
592     group_test(east_ecap, nw_mmm, nw_mdists);
593     group_test(east_ecap, w_mmm, w_mdists);
594     group_test(east_ecap, sw_mmm, sw_mdists);
595     group_test(east_ecap, s_mmm, s_mdists);
596     group_test(east_ecap, se_mmm, se_mdists);
597     group_test(east_ecap, e_mmm, e_mdists);
598     group_test(east_ecap, ne_mmm, ne_mdists);
599     ], 2)
600
601 lctm_northeast = sum([
602     group_test(northeast_ecap, n_mmm, n_mdists);
603     group_test(northeast_ecap, nw_mmm, nw_mdists);
604     group_test(northeast_ecap, w_mmm, w_mdists);
605     group_test(northeast_ecap, sw_mmm, sw_mdists);
606     group_test(northeast_ecap, s_mmm, s_mdists);
607     group_test(northeast_ecap, se_mmm, se_mdists);
608     group_test(northeast_ecap, e_mmm, e_mdists);
609     group_test(northeast_ecap, ne_mmm, ne_mdists);
610     ], 2)
611
612 %% Alternate test under motion baseline
613 % This test determines the effect of the alternate evaluation

```

*technique, used for accounting for the mean feedback when
evaluating motion captures, on motionless captures with a motion-
saturated baseline*

614

615 lctm_north = sum([

616 alternate_group_test(north_ecap, n_mmm, n_mdlist);

617 alternate_group_test(north_ecap, nw_mmm, nw_mdlist);

618 alternate_group_test(north_ecap, w_mmm, w_mdlist);

619 alternate_group_test(north_ecap, sw_mmm, sw_mdlist);

620 alternate_group_test(north_ecap, s_mmm, s_mdlist);

621 alternate_group_test(north_ecap, se_mmm, se_mdlist);

622 alternate_group_test(north_ecap, e_mmm, e_mdlist);

623 alternate_group_test(north_ecap, ne_mmm, ne_mdlist);

624], 2)

625

626 lctm_northwest = sum([

627 alternate_group_test(northwest_ecap, n_mmm, n_mdlist);

628 alternate_group_test(northwest_ecap, nw_mmm, nw_mdlist);

629 alternate_group_test(northwest_ecap, w_mmm, w_mdlist);

630 alternate_group_test(northwest_ecap, sw_mmm, sw_mdlist);

631 alternate_group_test(northwest_ecap, s_mmm, s_mdlist);

632 alternate_group_test(northwest_ecap, se_mmm, se_mdlist);

633 alternate_group_test(northwest_ecap, e_mmm, e_mdlist);

634 alternate_group_test(northwest_ecap, ne_mmm, ne_mdlist);

635], 2)

636

637 lctm_west = sum([

638 alternate_group_test(west_ecap, n_mmm, n_mdlist);

639 alternate_group_test(west_ecap, nw_mmm, nw_mdlist);

640 alternate_group_test(west_ecap, w_mmm, w_mdlist);

641 alternate_group_test(west_ecap, sw_mmm, sw_mdlist);

642 alternate_group_test(west_ecap, s_mmm, s_mdlist);

```

643     alternate_group_test(west_ecap, se_mmm, se_mdist);
644     alternate_group_test(west_ecap, e_mmm, e_mdist);
645     alternate_group_test(west_ecap, ne_mmm, ne_mdist);
646     ], 2)
647
648 lctm_southwest = sum([
649     alternate_group_test(southwest_ecap, n_mmm, n_mdist);
650     alternate_group_test(southwest_ecap, nw_mmm, nw_mdist);
651     alternate_group_test(southwest_ecap, w_mmm, w_mdist);
652     alternate_group_test(southwest_ecap, sw_mmm, sw_mdist);
653     alternate_group_test(southwest_ecap, s_mmm, s_mdist);
654     alternate_group_test(southwest_ecap, se_mmm, se_mdist);
655     alternate_group_test(southwest_ecap, e_mmm, e_mdist);
656     alternate_group_test(southwest_ecap, ne_mmm, ne_mdist);
657     ], 2)
658
659 lctm_south = sum([
660     alternate_group_test(south_ecap, n_mmm, n_mdist);
661     alternate_group_test(south_ecap, nw_mmm, nw_mdist);
662     alternate_group_test(south_ecap, w_mmm, w_mdist);
663     alternate_group_test(south_ecap, sw_mmm, sw_mdist);
664     alternate_group_test(south_ecap, s_mmm, s_mdist);
665     alternate_group_test(south_ecap, se_mmm, se_mdist);
666     alternate_group_test(south_ecap, e_mmm, e_mdist);
667     alternate_group_test(south_ecap, ne_mmm, ne_mdist);
668     ], 2)
669
670 lctm_southeast = sum([
671     alternate_group_test(southeast_ecap, n_mmm, n_mdist);
672     alternate_group_test(southeast_ecap, nw_mmm, nw_mdist);
673     alternate_group_test(southeast_ecap, w_mmm, w_mdist);
674     alternate_group_test(southeast_ecap, sw_mmm, sw_mdist);

```

```

675     alternate_group_test(southeast_ecap, s_mmm, s_mdists);
676     alternate_group_test(southeast_ecap, se_mmm, se_mdists);
677     alternate_group_test(southeast_ecap, e_mmm, e_mdists);
678     alternate_group_test(southeast_ecap, ne_mmm, ne_mdists);
679 ], 2)
680
681 lctm_east = sum([
682     alternate_group_test(east_ecap, n_mmm, n_mdists);
683     alternate_group_test(east_ecap, nw_mmm, nw_mdists);
684     alternate_group_test(east_ecap, w_mmm, w_mdists);
685     alternate_group_test(east_ecap, sw_mmm, sw_mdists);
686     alternate_group_test(east_ecap, s_mmm, s_mdists);
687     alternate_group_test(east_ecap, se_mmm, se_mdists);
688     alternate_group_test(east_ecap, e_mmm, e_mdists);
689     alternate_group_test(east_ecap, ne_mmm, ne_mdists);
690 ], 2)
691
692 lctm_northeast = sum([
693     alternate_group_test(northeast_ecap, n_mmm, n_mdists);
694     alternate_group_test(northeast_ecap, nw_mmm, nw_mdists);
695     alternate_group_test(northeast_ecap, w_mmm, w_mdists);
696     alternate_group_test(northeast_ecap, sw_mmm, sw_mdists);
697     alternate_group_test(northeast_ecap, s_mmm, s_mdists);
698     alternate_group_test(northeast_ecap, se_mmm, se_mdists);
699     alternate_group_test(northeast_ecap, e_mmm, e_mdists);
700     alternate_group_test(northeast_ecap, ne_mmm, ne_mdists);
701 ], 2)

```

Appendix H. FCaE Utility Functions

```
1 function [lst] = get_elem(capture, sc, el)
2 %get_elem Retrieves a list of a particular feedback element from
   every
3 %packet in a capture.
4
5 n_pkt = size(capture, 2)/4;
6 lst = zeros(1, n_pkt);
7
8 for i = 1:n_pkt
9     lst(1, i) = capture(sc, 4*i - 4 + el);
10 end
11
12 end

1 function [meanmatrix, devmatrix] = mean_dev_capture(capture)
2 %mean_dev_capture Provides mean matrix and deviation matrix from a
   capture
3 % The capture contains n_sc x n_el complex numbers for each packet
   . This
4 % calculates the real and imaginary mean of each number across the
5 % packets, and the real and imaginary standard deviation for each
   number,
6 % and stores both in matrices the size of a packet.
7
8 n_sc = 234;
9 n_el = 4;
10 n_pkt = size(capture, 2)/n_el;
11
12 meanmatrix = zeros(n_sc, n_el);
13 devmatrix = zeros(n_sc, n_el);
14
```

```

15 for i = 1:n_sc
16     for j = 1:n_el
17         tmp = 0;
18         lst_i = zeros(1, n_pkt);
19         lst_q = zeros(1, n_pkt);
20         for k = 1:n_pkt
21             tmp = tmp + capture(i, n_el*k + j - n_el);
22             lst_i(1, k) = real(capture(i, n_el*k + j - n_el));
23             lst_q(1, k) = imag(capture(i, n_el*k + j - n_el));
24         end
25         tmp = tmp / n_pkt;
26         meanmatrix(i, j) = tmp;
27         dev_i = std(lst_i);
28         dev_q = std(lst_q);
29         devmatrix(i, j) = dev_i + dev_q*sqrt(-1);
30     end
31 end
32
33 end

1 function [d] = matrix_distance(m,n)
2 %matrix_distance Calculates the distance between two matrices
3
4 d = 0;
5 for i = 1:size(m, 1)
6     for j = 1:size(m, 2)
7         tmp = sqrt((real(m(i, j)) - real(n(i, j)))^2 + (imag(m(i, j))
            ) - imag(n(i, j)))^2);
8         d = d + tmp;
9     end
10 end
11 d = d / (size(m, 1) * size(m, 2));

```

```

12 end

1 function [d] = packet_distance(fullpacket,meanmatrix)
2 %packet_distance determines the distance between a packet and a mean
3 %feedback matrix
4
5 n_sc = size(fullpacket, 1);
6
7 d = 0;
8
9 for i = 1:n_sc
10     tmp_d = matrix_distance(fullpacket(i), meanmatrix(i));
11     d = d + tmp_d;
12 end
13
14 d = d / n_sc;
15
16 end

1 function [pkt] = pkt_at(capture, idx)
2 %pkt_at Retrieves the packet at an index, in a capture
3
4 pkt = capture(1:234, (idx-1)*4+1:idx*4);
5 end

1 function plot_elem(sc, elem, capture)
2 %plot_elem Plots an element of feedback from a capture
3 %   There are 234 subcarriers, each with 4 elements. This provides
4 %   a
5 %   method for quickly visualizing one of those elements on a graph,
6 %   by
7 %   plotting it, plotting the mean, and plotting three concentric
8 %   ellipses

```

```

6 %   at 1, 2, and 3 standard deviations out fro the mean.
7 hold on;
8 [mm, dm] = mean_dev_capture(capture);
9
10 data = get_elem(capture, sc, elem);
11
12 m = mm(sc, elem);
13 xC = real(m);
14 yC = imag(m);
15
16 dev = dm(sc, elem);
17 rX = real(dev);
18 rY = imag(dev);
19
20 th = 0:0.01:2*pi;
21 x1 = xC + rX*cos(th);
22 x2 = xC + 2*rX*cos(th);
23 x3 = xC + 3*rX*cos(th);
24
25 y1 = yC + rY*sin(th);
26 y2 = yC + 2*rY*sin(th);
27 y3 = yC + 3*rY*sin(th);
28
29 plot(m, 'bx')
30 plot(data, 'b. ');
31 plot(x1, y1, 'b');
32 plot(x2, y2, 'g');
33 plot(x3, y3, 'r');
34
35 legend('mean', 'data', 's', '2s', '3s');
36
37 axis square;

```

```

38 xlim([-1 1]);
39 ylim([-1 1]);
40
41 hold off;
42
43 end

1 function [distances, dev, mm, dm] = analyze(capture)
2 %analyze trains on a baseline capture
3 %   Produces the mean feedback matrix, deviation feedback matrix,
   and the
4 %   model for the mean packet distance normal distribution.
5
6 % Produces n_pkt means, with n_p packets drawn for each trial
7 n_p = 10;
8 [mm, dm] = mean_dev_capture(capture);
9 n_pkt = size(capture, 2)/4;
10 distances = zeros(1, n_pkt);
11
12 % Randomize packet sampling order
13 idx = mod(randperm(n_pkt*n_p), n_pkt) + 1;
14
15 for i = 1:(n_pkt)
16     trial_dist = 0;
17
18     for j=1:n_p
19         tmppkt = pkt_at(capture, idx(1, (j-1)*n_pkt + i));
20         trial_dist = trial_dist + packet_distance(tmppkt, mm);
21     end
22
23     dist = trial_dist / n_p;
24

```

```

25     distances(1, i) = dist;
26 end
27
28 dev = std(distances);
29
30 end

1 function [membership, grpd] = group_test(testcapture, baseline_mm,
    baseline_dist)
2 %group_test splits a capture into test groups, and compares against
    a baseline
3 % membership output is 0 for test group if negative, 1 if positive
4
5 % n_grp groups tested, n_tr trials per group, n_p packets per trial
6 n_p = 5;
7 n_grp = 25;
8 n_pkt = size(testcapture, 2)/4;
9 n_tr = 5;
10 membership = zeros(1, n_grp);
11
12 idx = mod(randperm(n_grp*n_p*n_tr), n_pkt) + 1;
13 grpd = zeros(n_grp, n_tr);
14
15 % iterate and test each group
16 for g = 1:n_grp
17
18     % get test packet distances from baseline mean
19     grp_dist = zeros(1, n_tr);
20     for tr = 1:n_tr
21         tmpdist = 0;
22         for pkt = 1:n_p
23             tmp_pkt = pkt_at(testcapture, idx(pkt + (tr-1)*n_p + (g

```

```

-1)*n_p*n_tr));
24         tmpdist = tmpdist + packet_distance(tmp_pkt, baseline_mm
);
25     end
26     tmpdist = tmpdist / n_p;
27     grp_dist(1, tr) = tmpdist;
28 end
29 grp_d(g, 1:n_tr) = grp_dist;
30 [h, p] = ttest2(baseline_dist, grp_dist, 'alpha', .0001);
31 membership(1, g) = 1-h;
32 end
33
34
35 end

```

```

1 function [membership, grp_d] = alternate_group_test(testcapture,
baseline_mm, baseline_dist)
2 %alternate_group_test The alternate evaluation mechanism
3 % Instead of taking a mean of packet distances to compare against
a
4 % baseline, this evaluation compares the distances of mean packets
5 % against the baseline. The purpose is to account for the mean
feedback
6 % more than the individual packet.
7
8 % n_grp groups tested, n_tr trials per group, n_p packets per trial
9 n_p = 10;
10 n_grp = 25;
11 n_tpkt = size(testcapture, 2)/4;
12 n_tr = 10;
13 membership = zeros(1, n_grp);
14

```

```

15 % randomly order packets to be sampled for trials
16 idx = mod(randperm(n_grp*n_p*n_tr), n_tpkt) + 1;
17 grpd = zeros(n_grp, n_tr);
18
19 % iterate and test each group
20 for g = 1:n_grp
21
22     % get test packet distances from baseline mean
23     grp_dist = zeros(1, n_tr);
24     for tr = 1:n_tr
25         tpkts = zeros(234, 4*n_p);
26         for pkt = 1:n_p
27             tpkts(1:234, (pkt - 1)*4 + 1:pkt*4) = pkt_at(testcapture
, idx(idx + (g-1)*n_p));
28         end
29         % Calculate mean feedback for trial
30         tmp_mm = mean_dev_capture(tpkts);
31         % Calculate distance between mean packet and baseline
32         grp_dist(1, tr) = packet_distance(tmp_mm, baseline_mm);
33     end
34     grpd(g, 1:n_tr) = grp_dist;
35     % designed to account for motion with higher variation, so
decrease
36     % alpha by 100-fold
37     [h, p] = ttest2(baseline_dist, grp_dist, 'alpha', .000001);
38     membership(1, g) = 1-h;
39 end
40
41 end

1 function [different] = environment_detect(capture1, capture2)
2 %environment_detect This detects differences in the environment

```

```

        between two
3 %captures
4 %   The differences in environment between captures results in more
5 %   variation in a motion-saturated environment, and less in a
        motionless
6 %   environment. This test samples the variance to determine if the
7 %   variance is different between two captures, and deduce if the
8 %   environment is distinguishable between them.
9
10 % n_grp tests, from n_tr trials each, with n_p packets per trial
11 n_p = 10;
12 n_grp = 25;
13 n_tr = 10;
14 n_pkt1 = size(capture1, 2)/4;
15 n_pkt2 = size(capture2, 2)/4;
16
17 different = zeros(1, n_grp);
18
19 % randomize sampling order
20 idx1 = mod(randperm(n_p*n_grp*n_tr), n_pkt1)+1;
21 idx2 = mod(randperm(n_p*n_grp*n_tr), n_pkt2)+1;
22
23 % determine each capture's mean feedback
24 [~, ~, mm1] = analyze(capture1);
25 [~, ~, mm2] = analyze(capture2);
26
27 for g=1:n_grp
28     grp_d1 = zeros(1, n_tr);
29     grp_d2 = zeros(1, n_tr);
30     for tr=1:n_tr
31         d1 = zeros(1, n_p);
32         d2 = zeros(1, n_p);

```

```

33         for p=1:n_p
34             pkt1 = pkt_at(capture1, idx1(1, p + (g-1)*n_p*n_tr + (tr
-1)*n_p));
35             pkt2 = pkt_at(capture2, idx2(1, p + (g-1)*n_p*n_tr + (tr
-1)*n_p));
36
37             d1(1, p) = packet_distance(pkt1, mm1);
38             d2(1, p) = packet_distance(pkt2, mm2);
39         end
40
41         % each trial results in a mean packet distance
42         grp_d1(1, tr) = mean(d1);
43         grp_d2(1, tr) = mean(d2);
44     end
45
46     % each test group compares the variance in mean packet distance
47     different(1, g) = vartest2(grp_d1, grp_d2, 'alpha', .01);
48 end
49
50 end

```

Bibliography

1. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2016, 2016, accessed September 24, 2019. [Online]. Available: https://standards.ieee.org/standard/802_11-2016.html
2. M. S. Gast, *802.11 Wireless Networks: The Definitive Guide*. O'Reilly Media, Inc., 2005.
3. *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 1: Enhancements for High Efficiency WLAN*, IEEE Standard 802.11ax-2021, 2021, accessed February 20, 2020. [Online]. Available: https://standards.ieee.org/standard/802_11ax-2021.html
4. M. S. Gast, *802.11ac: A Survival Guide: Wi-Fi at Gigabit and Beyond*. O'Reilly Media, Inc., 2013.
5. J.-P. Le Rouzic, *802.11ac: An Analysis of the Standard*. CreateSpace Independent Publishing Platform, 2013.
6. L. Couch, *Digital and Analog Communication Systems*. Prentice Hall, 2001.
7. R. Heath and A. Lozano, *Foundations of MIMO Communication*. Cambridge University Press, 2018.
8. C. Shannon and W. Weaver, *The Mathematical Theory of Communication*. University of Illinois Press, 1998.
9. "802.11ac transmit beamforming," MATLAB Documentation Website, accessed August 11, 2020. [Online]. Available: <https://www.mathworks.com/help/wlan/ug/802-11ac-transmit-beamforming.html>
10. P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher *et al.*, "Spectre attacks: Exploiting speculative execution," in *2019 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2019, pp. 1–19, accessed August 18, 2020. [Online]. Available: <https://arxiv.org/pdf/1801.01203.pdf>
11. M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Haas, A. Fogh, J. Horn, S. Mangard, P. Kocher, D. Genkin *et al.*, "Meltdown: Reading kernel memory from user space," in *27th {USENIX} Security Symposium ({USENIX} Security 18)*, 2018, pp. 973–990, accessed August 18, 2020. [Online]. Available: <https://www.usenix.org/system/files/conference/usenixsecurity18/sec18-lipp.pdf>
12. S.-T. Park, E.-M. Park, J.-H. Seo, and G. Li, "Factors affecting the continuous use of cloud service: focused on security risks," *Cluster computing*, vol. 19, no. 1, pp. 485–495, 2016, accessed September 15, 2020. [Online]. Available: <https://link.springer.com/content/pdf/10.1007/s10586-015-0516-y.pdf>

13. T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, “Hey, you, get off of my cloud: exploring information leakage in third-party compute clouds,” in *Proceedings of the 16th ACM conference on Computer and communications security*, 2009, pp. 199–212, accessed September 10, 2020. [Online]. Available: https://ecee.colorado.edu/~ekeller/classes/fall2017_advsec/papers/cloudsec-ccs09.pdf
14. H. Ritzdorf, G. Karame, C. Soriente, and S. Čapkun, “On information leakage in deduplicated storage systems,” in *Proceedings of the 2016 ACM on Cloud Computing Security Workshop*, 2016, pp. 61–72, accessed September 14, 2020. [Online]. Available: <http://library.usc.edu.ph/ACM/SIGSAC%202017/ccsw/p61.pdf>
15. N. Cheng, X. O. Wang, W. Cheng, P. Mohapatra, and A. Seneviratne, “Characterizing privacy leakage of public wifi networks for users on travel,” in *2013 Proceedings IEEE INFOCOM*. IEEE, 2013, pp. 2769–2777, accessed August 19, 2020. [Online]. Available: https://www.researchgate.net/profile/Aruna-Seneviratne/publication/261060472_Characterizing_privacy_leakage_of_public_WiFi_networks_for_users_on_travel/links/560e32d008ae2aa0be4a8265/Characterizing-privacy-leakage-of-public-WiFi-networks-for-users-on-travel.pdf
16. M. Li, Y. Meng, J. Liu, H. Zhu, X. Liang, Y. Liu, and N. Ruan, “When csi meets public wifi: inferring your mobile phone password via wifi signals,” in *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, 2016, pp. 1068–1079, accessed October 21, 2020. [Online]. Available: <http://www.cs.sjtu.edu.cn/~naruan/publications/When%20CSI%20Meets%20Public%20WiFi%20Inferring%20Your%20Mobile%20Phone%20Password%20via%20WiFi%20Signals.pdf>
17. T. Chen, I. Ullah, M. A. Kaafar, and R. Boreli, “Information leakage through mobile analytics services,” in *Proceedings of the 15th Workshop on Mobile Computing Systems and Applications*, 2014, pp. 1–6, accessed August 10, 2020. [Online]. Available: <https://www.cs.odu.edu/~cs441/Papers/sec-010.pdf>
18. I. Ullah, R. Boreli, and S. S. Kanhere, “Privacy in targeted advertising: A survey,” *arXiv preprint arXiv:2009.06861*, 2020, accessed December 15, 2020. [Online]. Available: <https://arxiv.org/pdf/2009.06861.pdf>
19. T. Speicher, M. Ali, G. Venkatadri, F. N. Ribeiro, G. Arvanitakis, F. Benevenuto, K. P. Gummadi, P. Loiseau, and A. Mislove, “Potential for discrimination in online targeted advertising,” in *Conference on Fairness, Accountability and Transparency*. PMLR, 2018, pp. 5–19, accessed December 16, 2020. [Online]. Available: <http://proceedings.mlr.press/v81/speicher18a/speicher18a.pdf>

20. M. Park, H. Oh, and K. Lee, "Security risk measurement for information leakage in iot-based smart homes from a situational awareness perspective," *Sensors*, vol. 19, no. 9, p. 2148, 2019, accessed June 16, 2020. [Online]. Available: <https://www.mdpi.com/1424-8220/19/9/2148/pdf>
21. G. A. Office, "Internet of things: Information on use by federal agencies," online, 2020, accessed February 17, 2020. [Online]. Available: <https://www.gao.gov/assets/gao-20-577.pdf>
22. S. M. Beyer, "Pattern-of-life modeling using data leakage in smart homes," Master's thesis, Air Force Institute of Technology, 2018, accessed April 6, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=2793&context=etd>
23. A. Aragon Jr, "Evaluating machine learning techniques for smart home device classification," Master's thesis, Air Force Institute of Technology, 2019, accessed April 9, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=3243&context=etd>
24. B. E. Law, "Passive radiolocation of iee 802.11 emitters using directional antennae," Master's thesis, Air Force Institute of Technology, 2018, accessed April 14, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=2812&context=etd>
25. C. M. Bramlette, "Cyber attack drone payload development and geolocation via directional antennae," Master's thesis, Air Force Institute of Technology, 2019, accessed April 16, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=3248&context=etd>
26. N. V. Barker, "Development of a drone-mounted wireless attack platform," Master's thesis, Air Force Institute of Technology, 2020, accessed April 23, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=4225&context=etd>
27. F. Adelstein, P. Alla, R. Joyce, and G. G. Richard, "Physically locating wireless intruders," in *International Conference on Information Technology: Coding and Computing, 2004. Proceedings. ITCC 2004.*, vol. 1. IEEE, 2004, pp. 482–489, accessed April 24, 2020. [Online]. Available: <https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.106.5369&rep=rep1&type=pdf>
28. M. Kotaru, K. Joshi, D. Bharadia, and S. Katti, "Spotfi: Decimeter level localization using wifi," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 269–282, accessed April 23, 2020. [Online]. Available: <https://www.academia.edu/download/48484483/p269.pdf>

29. T. A. L. Lewis Sr, “Biologically inspired network (bionet) authentication using logical and pathological rf-dna credential pairs,” Master’s thesis, Air Force Institute of Technology, 2017, accessed April 28, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=1767&context=etd>
30. F. A. Cruz, “Near real-time rf-dna fingerprinting for zigbee devices using software defined radios,” Master’s thesis, Air Force Institute of Technology, 2019, accessed April 31, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=3254&context=etd>
31. Y. Z. Matsui, “Near real-time zigbee device discrimination using cb-dna features,” Master’s thesis, Air Force Institute of Technology, 2020, accessed April 30, 2020. [Online]. Available: <https://scholar.afit.edu/cgi/viewcontent.cgi?article=4183&context=etd>
32. Y. Ma, G. Zhou, and S. Wang, “Wifi sensing with channel state information: A survey,” *ACM Computing Surveys (CSUR)*, vol. 52, no. 3, pp. 1–36, 2019, accessed September 8, 2020. [Online]. Available: http://gzhou.blogs.wm.edu/files/2017/12/WiFiSenseSurvey_CSUR19.pdf
33. M. Schulz, D. Wegemer, and M. Hollick, “The nexmon firmware analysis and modification framework: Empowering researchers to enhance wi-fi devices,” *Computer Communications*, vol. 129, pp. 269–285, 2018, accessed July 6, 2020. [Online]. Available: <https://www.sciencedirect.com/science/article/abs/pii/S014036641731294X>
34. N. Damodaran, E. Haruni, M. Kokhkhharova, and J. Schäfer, “Device free human activity and fall recognition using wifi channel state information (csi),” *CCF Transactions on Pervasive Computing and Interaction*, vol. 2, no. 1, pp. 1–17, 2020, accessed September 13, 2021. [Online]. Available: <https://link.springer.com/content/pdf/10.1007/s42486-020-00027-1.pdf>
35. Qualcomm ipq4019 product overview. Accessed March 19, 2020. [Online]. Available: <https://www.qualcomm.com/products/ipq4019>
36. Qualcomm ipq40x8/ipq40x9 product brief. Accessed March 19, 2020. [Online]. Available: <https://www.qualcomm.com/media/documents/files/ipq40x8-ipq40x9-product-brief.pdf>

REPORT DOCUMENTATION PAGE					<i>Form Approved</i> OMB No. 0704-0188	
The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.						
1. REPORT DATE (DD-MM-YYYY) 16-09-2021		2. REPORT TYPE Master's Thesis		3. DATES COVERED (From — To) Sept 2019 — Sept 2021		
4. TITLE AND SUBTITLE DETERMINING PHYSICAL CHARACTERISTICS WITH INFORMATION LEAKAGE IN 802.11AC				5a. CONTRACT NUMBER		
				5b. GRANT NUMBER		
				5c. PROGRAM ELEMENT NUMBER		
6. AUTHOR(S) 1Lt Albert D. Taglieri				5d. PROJECT NUMBER 21G532A		
				5e. TASK NUMBER		
				5f. WORK UNIT NUMBER		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Air Force Institute of Technology Graduate School of Engineering and Management (AFIT/EN) 2950 Hobson Way WPAFB OH 45433-7765				8. PERFORMING ORGANIZATION REPORT NUMBER AFIT-ENG-MS-21-S-018		
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) AFRL/RYAA 2241 Avionic Cir WPAFB OH 45433-7765 COMM 937-713-8573 Email: Eric.Lam.3@us.af.mil				10. SPONSOR/MONITOR'S ACRONYM(S) AFRL/RYAA		
				11. SPONSOR/MONITOR'S REPORT NUMBER(S)		
12. DISTRIBUTION / AVAILABILITY STATEMENT DISTRIBUTION STATEMENT A: APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.						
13. SUPPLEMENTARY NOTES						
14. ABSTRACT The risk of information leakage in 802.11ac allows an eavesdropper to monitor wireless traffic and correlate physical locations between devices, as well as environment changes such as the motion of a person. Previous pattern-analysis mitigation methods, which used nonexistent devices to fool an eavesdropper, are not effective in an 802.11ac network, because devices on the network can be correlated to their physical location, which a nonexistent device does not have. Further, additional information about motion in the target environment can be observed and analyzed, providing a new potential for pattern analysis and sensing. 802.11ac makes it possible to plug in a wireless adapter in monitor mode, listen, and analyze, without specialized equipment. The present work provides tools and methods for basic analysis of 802.11ac beamforming feedback, demonstrates their utility and limitations, discusses the qualitative aspects of the analysis, suggests avenues of future development to refine and improve such analysis, and discusses the applications of the existing analysis.						
15. SUBJECT TERMS subject terms here						
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON	
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Barry E. Mullins, AFIT/ENG	
U	U	U	UU	184	19b. TELEPHONE NUMBER (include area code) (937) 255-3636, x7979; barry.mullins@afit.edu	