Object Localization Using RF Based IEEE 802.11 Frame Transceiving with FER Based RSSI Controlled Calibration

by

Gorla Praveen

A thesis submitted in partial fulfillment of the requirements for the
degree of Master of Engineering in
Telecommunications

Examination Committee: Dr. Teerapat Sanguankotchakorn (Chairperson)
Dr. Poompat Saengudomlert
Dr. Attaphongse Taparugssanagorn

Nationality: Indian
Previous Degree: Bachelor of Technology in Electronics and Communications
Engineering
Jawaharlal Nehru Technological University, Hyderabad,
Telangana, India

Scholarship Donor: AIT Fellowship

Asian Institute of Technology
School of Engineering and Technology
Thailand
May 2019

ACKNOWLEDGEMENTS

I am grateful to many people those who have helped and supported me in gaining knowledge and experience for the completion of my graduation. First, I would like to express my profound gratitude to my esteemed advisor, Dr. Teerapat San guankotchakorn for his advice throughout my master's course and thesis. It is also a great pleasure for me to thank my co-advisor, Dr. Poompat Saengudomlert for providing hardware and timely guidance in the methodological development of the thesis. I also take this space to extend my sincere gratitude to my committee member, Dr. Attaphongse Taparugssanagorn for his comments on improving the thesis.

Secondly, I take this opportunity to convey many thanks to Ms. Premma Rao and Mr. Rajesh Kumar Dehury for their timely act on providing hardware and lab infrastructure in the Telecommunication Engineering department for doing experimentation. In addition, a grateful thanks are extended to the Asian Institute of Technology, Thailand for providing AIT fellowship to support my masters.

Finally, I remember Dr. L. Pratap Reddy, Professor, JNT University Hyderabad, India because of whom my focus towards open research is improved from while I being in my undergraduate.

— Praveen

ABSTRACT

This thesis presents an approach of integrating single board computers as an access point and as a short-range object localization system. This is achieved by using open hardware and implementing IEEE 802.11 a/g/p based transceiving on low-cost Adlam Pluto Software Defined Radio (SDR), which operates in the S-band frequency range with Frame Error Rate (FER) controlled calibration. Experiments are conducted to investigate the effect of FER on the Received Signal Strength Indicator (RSSI) of IEEE 802.11 a/g/p transceiving, there by drawing the connection between the RSSI and the FER. Object localization is done based on the referenced RSSI and the real-time RSSI with FER based controlled calibration of RSSI. About 80% of accuracy can be expected from the implemented object localization system.

TABLE OF CONTENTS

CHAPTER	TITLE	PAGE
	Title Page	i
	Acknowledgments	ii
	Abstract	iii
	Table of Contents	iv
	List of Figures	vi
	List of Tables	viii
	List of Abbreviations	ix
	List of Symbols	xi
1	INTRODUCTION	1
	1.1 Overview	1
	1.2 Problem Statement	1
	1.3 Objectives	2
	1.4 Limitations and Scope	2
	1.5 Organization of the report	2
2	LITERATURE REVIEW	3
	2.1 Bistatic Radar and Bistatic Range	3
	2.2 Kalman Filter	4
	2.3 OFDM Signal Structure	4
	2.4 Methods for object localization and detection using Raspberry Pi	5
	2.5 Methods for object localization and detection using RF signals	5
	2.6 Additional parameters for human body detection	9
	2.7 Current state of the research	10
	2.8 Discussion	12
3	METHODOLOGY	13
	3.1 Introduction	13
	3.2 Power Analysis of Wireless Signals	13
	3.3 Object Localization using Wireless Access Point	14
	3.4 IEEE 802.11 a/g/p based object Localization with RSSI Calibration	17
	3.5 Hardware	22
	3.6 Software	24
	3.7 Proposed Model	24
4	EXPERIMENTATION RESULTS	26
	4.1 Introduction	26

	4.2 Power Analysis	26
	4.3 Wireless Access Point Localization	30
	4.4 RSSI Based on IEEE 802.11 a/g/p implementation	36
5	CONCLUSION	50
	References	51
	Appendices	52

LIST OF FIGURES

FIGURE	TITLE	PAGE
2.1	Block diagram of bistatic radar (<i>Nick Young. NATS, 2017</i>)	3
2.2	Block diagram of bistatic range (<i>Willis, Nicholas. 2007. Bistatic Radar</i>)	3
2.3	OFDM Signal Structure	5
2.4	Ambiguity function diagram (<i>Kelvin Chetty et al., 2009</i>)	6
2.5	Range resolution (<i>Chetty et al., 2009</i>)	7
2.6	Doppler resolution (<i>Chetty et al., 2009</i>)	7
2.7	Clutter (static environment) response (<i>Chetty et al., 2009</i>)	8
2.8	Target response (<i>Chetty et al., 2009</i>)	8
2.9	FMCW (<i>Adib et al., 2013</i>)	9
3.1	Basic model using signal generator and spectrum analyzer for feasibility test (Tx and Rx antennas of 2.4 GHz are available in AIT TC Lab)	13
3.2	Algorithmic flowchart based on Wireless Access Point	16
3.3	Implementation of realtime RSSI receiving and calibration for IEEE 802.11 a/g/p localization	17
3.4	Localization Estimation	17
3.5	Basic Model of Raspberry pi (<i>Raspberry Pi foundation., 2016</i>)	22
3.6	ADALM-PLUTO (<i>Analog Devices., 2017</i>)	23
3.7	Proposed Hardware model	24
3.8	Connection diagram for proposed model	25
4.1	Power Analysis Experimentation Scenario	26
4.2	Power Analysis Experimentation 1	27
4.3	Power Analysis Experimentation 2	27
4.4	Power Analysis Experimentation 3	28
4.5	Experiment, when Mobile client is moving from Access Point	30
4.6	Experiment, When mobile client is moving from left and right of AP	30
4.7	RSSI reception by using bash shell scripting	31
4.8	Actual distance vs. estimated distance for every one-meter range from experiment 1	32
4.9	Actual distance vs. estimated distance for every one-meter range from experiment 2	32
4.10	Actual distance vs. estimated distance for every one-meter range to the left and right of the AP from experiment 1	33
4.11	Actual distance vs. estimated distance for every one-meter range to the left and right of the AP from experiment 2	33
4.12	Real-time values of WAP based localization on dashboard	34
4.13	Kalman filtered distance and real-time distance	34

4.14	Kalman filtered velocity and real-time velocity	35
4.15	Frame Error Rate Analysis	36
4.16	FER and Distance	38
4.17	FER vs. RSSI Variance	39
4.18	FER vs. RSSI	40
4.19	Actual distance vs. estimated distance with and without <i>FER</i> based correction	41
4.20	Actual distance and estimated error with and without <i>FER</i> based correction	42
4.21	Actual distance and estimated error percentage with and without <i>FER</i> based correction	42
4.22	Error percentage in distance estimation with and without <i>FER</i>	43
4.23	Experimentation setup at various stages	44
4.24	Tx moving away from Rx1 and Rx2	45
4.25	Distance estimated by Rx1 for Tx as shown in figure 4.24	46
4.26	Distance estimated by Rx2 for Tx as shown in figure 4.24	46
4.27	Tx is moved from Left to Right	47
4.28	Distance estimated by Rx1 when Tx is moved as shown in figure 4.27	47
4.29	Obstacle between Tx and Rx	48
4.30	Distance estimated by Rx1 for Tx moving as shown in figure 4.29	48
4.31	Distance estimated by Rx2 for Tx moving as shown in figure 4.29	49
A.1	Working Images	93
A.2	Working Images	94

LIST OF TABLES

TABLE	TITLE	PAGE
2.1	Kalman Table	4
2.2	ACS values of human body deduced from the literature	10
3.1	Key parameters	14
4.1	Probability of FER Occurrence	37
4.2	Probability of FER Occurrence	37
4.3	Δ RSSI Calibration Values	40
4.4	Error percentage in distance estimation with and without <i>FER</i>	43

LIST OF ABBREVIATIONS

AP	Access Point
ACR	Absorption Cross Section
CFR	Channel Frequency Response
CIR	Channel Impulse Response
CS	Compressive Sensing
CSI	Channel State Estimation
DBPSK	Differential Binary Phase Shift Keying
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
DQPSK	Differential Quadrature Phase Shift Keying
DSI	Direct Signal Interference
DSSS	Direct Sequence Spread Spectrum
FER	Frame Error Rate
FMCW	Frequency-Modulated Continuous-Wave
FPGA	Field-Programmable Gate Array
GNU	GNU's NOT Unix
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronics Engineers
LOS	Line of Sight
MAC	Media Access Controller
MHz	Megahertz
NLOS	Non Line of Sight
NLR	Nonlinear Regression
OFDM	Orthogonal Frequency Division Multiplexing
PHY	Physical Layer
PLCP	Physical Layer Convergence Protocol
PPDU	PLCP Protocol Data Unit
QAM	Quadrature Amplitude Modulation
RADAR	RAdio Direction And Ranging
RF	Radio Frequecny
RSSI	Received Signal Strength Indicator
SDR	Software Defined Radio
SAR	Specific Absorption Rate

SSH	Secure Shell
SSID	Service Set Identifier
TOF	Time of Flight
WAP	Wireless Access Point
WC-CWL	Weighted Centroid - Constrained Weighted Least Square
Wi-Fi	Wireless Fidelity
WPA	Wi-Fi Protected Access

LIST OF SYMBOLS

c	Velocity of light
D_{rx}	Distance between R_x and target
D_{tx}	Distance between T_x and target
f_d	Frequency shift
n	Path loss component
$p(x)$	Probability of x
Rx	Receiving antenna
Tx	Transmitting antenna
δf	Change in frequency
V	Velocity of target
χ	Basic ambiguity function
τ	Time delay
$\Delta RSSI$	Change in $RSSI$

CHAPTER 1

INTRODUCTION

1.1 Overview

Indoor localization and detection of objects using wireless signals (*Adib et al., 2014*) has been complex issue because of the closed architecture which does not allow customizable system design. Two models are considered in this thesis, one at the primary level of research and other at the secondary level of research. At the primary level, the focus is on detection of objects with efficient algorithms using wireless signals and digital signal processing . At the secondary level, the focus is on implementing the above said concept on an Open Router as an Access Point for object localization with the existing Open Hardware and with the implementation of IEEE 802.11 a/g/p based transceiving on low cost Adlam Pluto Software Defined Radio (SDR), which operates in the S-band frequency range, with Frame Error Rate (FER) controlled calibration . There are several models demonstrated by research scholars for target detection and indoor localization using wireless signals (*Guo et al., 2008*). This thesis demonstrates the use of single board computers which has routing capabilities as an Access Point and SDR for target detection using wireless signals.

1.2 Problem Statement

Indoor localization in this era of Internet of Things has been on the rise to detect objects with high accuracy. But it needs a large number of sensors and at least one networking device (router) to be employed for object sensing and data communications. Existing methods include the use of RSSI and Channel State Information (CSI) from the reflected signals where external WiFi access points (AP) act as a transmitters along with reference antennas and surveillance antennas for reception (*Chetty et al., 2009*). Other methods are based on calculating time of flight (ToF) using a chirp generator with transmitting and receiving antennas (*Adib et al., 2014*). However the use of RSSI still lacks calibration when there are occurrences of frame errors for IEEE 802.11 a/g/p devices. This proposal is a novel approach of integrating object detection in router acting as an Access Point with controlled calibration of IEEE 802.11 a/g/p RSSI based object localization. The implementation employs single board computers and software defined radio (SDR).

1.3 Objectives

- To design a system for indoor localization using wireless signals.
- To implement indoor localization in a router (Wireless Access Point) using single board computer(s).
- To implement indoor localization based on Adlam Pluto SDR for IEEE 802.11 a/g/p transceiving with FER controlled calibration.

1.4 Limitations and Scope

The developed system is used to detect and localize objects such as IEEE 802.11 a/g/p devices with FER based RSSI calibration at short-range distances. Due to low processing capabilities of low-cost Software Defined Radios and single board computers, the accuracy is moderate. The developed system can be extended to detect and localize different objects for automation in the Internet of Things (IoT) domain.

1.5 Organization of the report

Chapter 1: Introduction: Overview, problem statement, objective, scope and limitations and organization of the report.

Chapter 2: Literature review on the current state of research.

Chapter 3: Research methodology and the expected outcome.

Chapter 4: Experimentation results and analysis.

Chapter 5: Conclusion on current and future scope of the research.

CHAPTER 2

LITERATURE REVIEW

There has been numerous development in radar systems for localization from the past which has led us to improve more refined systems for indoor localization. Earlier radar systems are developed especially for military purpose but now research has expanded to develop tiny system at affordable cost with low range object detection for localization. The following represents some of the methods and the current state of the research.

2.1 Bistatic Radar and Bistatic Range

Bistatic radar as shown in Figure 2.1, is a type of radar system in which the distance between the transmitter and the receiver is almost correspondingly comparable to the approximated target range. In opposite to this there are other radar systems such as Pseudo-monostatic radar, Forward scatter radar, Multistatic radar and passive radar where these are different from Bistatic radar in a way of separate Tx & Rx, range of detection, principle of detection, type and number of antennas used.

Figure 2.1: Block diagram of bistatic radar (*Nick Young. NATS, 2017*)

Figure 2.2: Block diagram of bistatic range (*Willis, Nicholas. 2007. Bistatic Radar*)

Basic determination of range using separate transmitter and receiver by calculating time of arrival of the object is called as bistatic range. Bistatic range as shown in Figure 2.2 is equivalent to $D_{tx} + D_{rx} - L$ where D_{tx} , D_{rx} and L are distances between transmitter and target object, target object and receiver, and transmitter and receiver respectively (Willis, Nicholas. 2007).

2.2 Kalman Filter

Table 2.1: Kalman Table

<i>State Prediction</i>	$X_{predicted} = AX_{n-1} + BU_n$	Next output prediction
<i>Covariance Prediction</i>	$P_{predicted} = AP_{n-1}A^T + Q$	Next error prediction
<i>Innovation</i>	$\bar{Y} = Z_n - HX_{predicted}$	Difference between reality and prediction
<i>Innovation Covariance</i>	$S = HP_{predicted}H^T + R$	Difference between real error and predicted error
<i>Kalman Gain</i>	$K = P_{predicted}H^T S^{-1}$	Kalman Gain calculation
<i>State Update</i>	$X_n = X_{predicted} + K\bar{Y}$	Update estimate
<i>Covariance Update</i>	$P_n = (I - KH)P_{predicted}$	Update error

Kalman filter, as a linear quadratic equation that tracks the estimated state and variance of the system. There by considering state transition model an updated and more accurate value of estimate is resulted. Parameters such as X_{n-1} (previous state estimate) and P_{n-1} (previous state error) are taken in to consideration along with $\bar{Y}$ (Innovation) and K (Kalman Gain) to often result an accurate estimate.

Extended models such as *extend Kalman filter* and *unscented Kalman filter* to overcome the failure of Kalman filter in accurate estimation of a state in non-linear systems.

2.3 OFDM Signal Structure

Software Defined Radio based IEEE 802.11 a/g/p uses OFDM signal structure for frame transceiving. The following frame represented in figure 2.3 contains data which is OFDM encoded with variable length

with varying payload between 0 - 4095 bytes. OFDM uses guard intervals (GI) in between long preamble, Signal field, Service and payload.

Figure 2.3: OFDM Signal Structure

2.4 Methods for object localization and detection using Raspberry Pi

Raspberry Pi and digital camera

Raspberry Pi, a single board computer is a great tool to interface with digital cameras for image processing techniques. Raspberry Pi as a low cost medium performance microprocessor is capable of processing vision applications and is extensively used for surveillance as it finds easier interface with cameras. As described by *Abaya et al., 2014* it is possible to realize low cost smart security camera with night vision using Raspberry and Open CV along with infrared(IR) led.

Basic system includes use of IR for activating night mode vision of webcam when there is a stance of an object presence. Raspberry pi acts as a image processing unit and upon processing along with noise elimination by background subtraction. Here background subtraction helps in detecting moving objects by frame difference method. it imports data to web server where a person away from home can access this data anywhere. There are many applications of Raspberry Pi in detecting object using digital cameras such as in *Islam et al., 2014*, Raspberry Pi is used as an image processing tool in electronic voting machine for detecting face. It is also used in *Pereira et al., 2014* for object sorting by a robotic arm and placing the object in desired position and moreover in *Janard et al., 2015* it is used for real time detection of miscellaneous object and to follow them by using telepresence robots.

2.5 Methods for object localization and detection using RF signals

Ambiguity function

Ambiguity function analysis is mainly used in radar engineering to establish relation between transmitted signal and the received signal which describes extent of distortion between transmitted and received signal. But many definitions of ambiguity function exist such as describing propagation delay and Doppler shift in wideband signals. However basic definition is to describe the relation between two signals in time and frequency domain.

It is a two- dimensional functional analysis of time delay and Doppler shift in frequency of moving object. Where Equation 2.1 represents self ambiguity function of a signal, i.e the relation between a signal ($x_1(t)$) and its time delayed (τ) version with frequency shift (f_d) . Equation 2.2 represents cross ambiguity function between transmitted signal ($x_1(t)$) and received reflected signal ($x_2(t)$) from the moving target object with time delay (τ) and Doppler frequency shift (f_d).

$$|\chi_s(\tau, f_d)| = \left| \int_{-\infty}^{\infty} x_1(t) x_1^*(t - \tau) \exp(j2\pi f_d t) dt \right| \quad (\text{Equation 2.1})$$

$$|\chi_c(\tau, f_d)| = \left| \int_{-\infty}^{\infty} x_1(t) x_2^*(t - \tau) \exp(j2\pi f_d t) dt \right| \quad (\text{Equation 2.2})$$

Further this ambiguity function can be described for zero Doppler shift ($f_d = 0$), i.e when the target object is stationary, as a correlation between ($x_1(t)$) and ($x_2(t)$) as $\chi_c(\tau, 0)$.

Ambiguity function graph in figure 2.4 from *Kelvin Chetty et al., 2009* represents normalized amplitude $|\chi_c(\tau, f_d)|$ in z-axis, Doppler shift (f_d) in y-axis and delay (τ) expressed in terms of distance (m) from which target object is located from the system. So normalized amplitude $|\chi_c(\tau, f_d)|$ shows extent of distortion between transmitted signal and the received signal.

Figure 2.4: Ambiguity function diagram (*Kelvin Chetty et al., 2009*)

Data reception and data processing

1. When a signal is transmitted with certain frequency and the reception of signal is done through receiving antenna which configured to receive for some time. All the data is processed in MATLAB using cross ambiguity function.

2. For determining range resolution, taking for zero Doppler cut i.e $f_d = 0$. Graph between normalized amplitude and range is done as an example shown in figure 2.5 from *Kelvin Chetty et al., 2009* where range resolution is determined.

Figure 2.5: Range resolution (*Chetty et al., 2009*)

3. For determining Doppler resolution, considering zero range cuts. Graph between normalized amplitude and Doppler shift is obtained as an example shown in figure 2.6 from *Chetty et al., 2009* where Doppler shift resolution is determined.

Figure 2.6: Doppler resolution (*Chetty et al., 2009*)

4. Then Doppler-range graphs are obtained when the reception is done and when there is no presence of object. Here the received signal is prone to Direct Signal interference (DSI) and static surrounding environment.
5. Through DSI cancellation, i.e subtracting self ambiguity from the cross ambiguity and again plotting Doppler-range graph as a example shown in figure 2.7 which is an elimination of DSI but with the presence of static environment (Clutter).

Figure 2.7: Clutter (static environment) response (*Chetty et al., 2009*)

6. When the target is made to present and data reception is done for some time along with DSI cancellation. Doppler - range graph are obtained. By analyzing the Doppler-range graphs from (5) and (6) with the shift in Doppler and range makes us to distinguish absence and presence of object

Figure 2.8: Target response (*Chetty et al., 2009*)

From FMCW generator

Frequency Modulated Carrier Wave which is a periodic RF signal with linear increase of frequency in a time as shown in figure 2.9. The transmitted signal reflects after some time from the target which is present in the room, in which reflected is received by FMCW receiver. Here the time of flight is used for determination of l of object from the FMCW system.

Figure 2.9: FMCW (Adib et al., 2013)

Range resolution which is a minimum measurable change in location with minimum measurable shift in frequency δf_{min} is represented in Equation 2.3 where delay is the Time of Flight (TOF) with minimum TOF as TOF_{min} of a signal which is reflected back to the receiver upon transmitted from transmitter when there is a presence of target, δf is the instantaneous differences in frequency at transmitter and receiver, C is the speed of light and factor 2 accounts for the twice of distance between system and target where the signal travelled from transmitter to target and from target to receiver.

$$Resolution = C \times \frac{TOF_{min}}{2} = C \times \frac{\delta f_{min}}{2(slope)} \quad (\text{Equation 2.3})$$

1. In order to distinguish between presence and absence of object. Initially, we made to work transmission of FMCW and its reception of reflected wave when there is no presence of object and with static environment. Thus considering this as a background noise. The data such as TOF corresponding to each frequency is noted.
2. Now the target such as human body is made to present and the data such as TOF is recorded. Then after subtracting background noise, we could able to distinguish the presence of absence with a distance of D as represented in the Equation 2.4 from the system.

$$D = C \times TOF \quad (\text{Equation 2.4})$$

2.6 Additional parameters for human body detection

Human body responses at different levels at different frequencies. In order to study the reflection parameters and to deduce the parameters for detection we need certain parameters like absorption rate. One such parameter is Specific Absorption Rate SAR , which defines the amount of absorbed power (in Watts) of an EM signal per unit mass (of human body). Relation is shown in Equation 2.5

$$SAR = \frac{Power_{absorbed}(W)}{mass} \quad (\text{Equation 2.5})$$

The other most accurate form is to define Absorption Cross Section ACS , as amount of power absorbed

by the body per unit power density S_c of an incident signal. Relation is as shown in Equation 2.6

$$ACR(\sigma_a) = \frac{P_{\text{power absorbed}}(W)}{S_c(W/m^2)} \quad (\text{Equation 2.6})$$

The following parameters as shown in the Table 2.2 represents ACS values of the human body at different frequencies at different scenarios (Melia, 2013.)

Table 2.2: ACS values of human body deduced from the literature

Authors	$f(\text{GHz})$	Method	Subject	ACS(m^2)
Andersen et al.	3 - 8	Exp	Several	0.33
Bamba et al.	2.4 - 3.0	Exp.	Several avg 70kg 175cm	0.34, 0.36
Harima et al.	1 - 4	Exp	70.6 kg male	0.33 - 0.11
Hirata et al	2	FDTD sim.	Japanese adult child	0.4 - 0.5
Hurst et al.	2.1	Not stated	Unknown	0.4
Robinson et al.	0.91	TLM sim.	thin boundary models	0.25
Uusitupa et al	0.9 - 5	sim	VF Male (72.2kg)	0.4-0.25
Findaly et al.	1 - 2	FDTD sim.	NORMAN (73.0 kg)	0.35 - 0.3
Kientega et al.	2.4	FDTD sim.	Thelonius (1.17m, 19 kg)	0.06 - 0.14

2.7 Current state of the research

- Guo et al., 2008 - This is the first experiment to detect objects with Wi-Fi signals by using Beacon transmissions such as two parts of DBPSK and DQPSK with DSSS modulation used for transmission for wireless based passive radar. Self ambiguity function analysis and cross ambiguity analysis is done upon reception of signals for detection.
- Ahamed et al., 2008 - Tracking system for pervasive environment, linear approximation with RSSI is used by collecting samples by measuring distance and direction of Wi-Fi APs.
- Chetty et al., 2009 - Detection of target using Passive Bistatic Wi-Fi radar when system and target are in high densified echo environment. By using Wi-Fi Access point, reference receiver and

surveillance receiver with OFDM, 64 QAM transmission, ambiguity analysis is done for location approximation using bistatic radar concept with nullification of DSI (Direct Signal Interference). A similar work by the same author (Kevin Chetty et al., 2011) is through the wall sensing of personnel using passive bistatic Wi-Fi radar at standoff distances. In this targets are monitored from outside the wall so as to detect the target satisfying the concept of seeing through the walls.

- Falcone et al., 2012 - Tracking of non stationary objects is done with Wi-Fi based radar by using Global Positioning System (GPS) for accurate positioning.
- Rzewuski and kulpa, 2011 -Designing of system concept for localization using PPDU (PLCP Protocol data Unit) of MAC address to locate the destination.
- Zhou et al., 2015 - Explained the concept of using Channel State Information (CSI), a PHY feature for extracting Channel Frequency Response (CFR) and Channel Impulse Response (CIR) with amplitude and phase detection of the reflected signal. CSI can be obtained from the Wi-Fi network cards with a constraint of using CSI of only 40 MHz (802.11n) bandwidth.
- Wu et al., 2015 - Examined the use of CSI for tracking humans, phase and frequency diversity is used to map body with use of chest motion of human. Experiment is done for LOS and NLOS detection.
- Van Dijk et al., 2008 - Design of FMCW antenna Array for radar applications with independency in transmitter and receiver reception with Direct Digital Synthesis Technology.
- Adib et al., 2013 - See-Through walls with Wi-Fi, a wall gesture based communication using inverse sympathetic detection (using only one receiver). Algorithm based detection by calculating the correlation matrix and eigenvalues for separating the signal from the noise along with indication of direction.
- Adib et al., 2014 - Based on RF Reflections of the body by using static frequency modulated continuous wave (FMCW) where different parts of body respond for reflecting signals at different frequencies. Here FMCW is operated for some duration, so different body parts reflect at different frequencies and the shape forming of the body is done.
- Adib et al., 2014 - A similar work by using RF reflections for multi personnel localisation by using multi-shift FMCW.
- Adib et al., 2015 - Based on variations in reflected signal from the stationary human body monitoring of heart beat is done. Without loss in generality that these variations are due to heartbeat reflections of stationary human body at particular reflected frequency of heart with narrow bandwidth.
- Paolo Barsocchi et al., 2007 - This paper demonstrates the commonly used frame loss model for simulation over Wi-Fi channels. It estimates the variance of RSSI with FER.
- Zan Li et al., 2015 - about study of Passive Wi-Fi source Localization implemented on GNU Radio testbed

2.8 Discussion

Literature review on several existing methods has shown significant development in designing indoor positioning systems. Some of the methods discussed in section 2.7 use RSSI and CSI (Zhou *et al.*, 2015) of the wireless signals. Wireless beacon transmission, such as DSSS & DBPSK, DSSS & DQPSK modulated signal described in (Geo *et al.*, 2008), is the first method used for target detection. The transmitted beacon signal changes its characteristics upon reflection from the target, and these changed characteristics are used for signal analysis. In particular, with the use of the ambiguity function, changes in characteristics of the transmitted signal are analyzed and the results can lead to determine the characteristics of a target.

Latest research reveals that the use of a chirp generator described in (Adib *et al.*, 2014, Adib *et al.*, 2015) finds its applications not only in target detection but also in shape forming. When a chirp generator generates signals with varying frequencies from which different human body parts reflect, these reflections are captured by receiving antennas, to be followed by sample collection and signal processing techniques. This method also improves detection accuracy with moderate accuracy in shape forming.

All these systems are capable of detecting a target, but these methods are difficult to integrate with existing systems because of closed system properties and interoperability. However, these methods are the foundation for research in radar engineering, especially in indoor localization.

As the RSSI is prone to frame error (Paolo Barsocchi *et al.*, 2007), existing system lack in calibrating the RSSI based on the Frame Error Rate (FER). They also need more infrastructure. Some of the methods use external Wi-Fi APs as transmitters, which increases the system dependency on others and limits the detection accuracy because of unknown exact natures of APs. This thesis contributes to designing a system with the use of Single Board Computers which simultaneously act as a Wi-Fi AP and an object detecting system with the use of SDR calibrating the RSSI based on FER. This system is also relatively low-cost and small in size, with the use of minimal infrastructure and increasing independency.

CHAPTER 3

METHODOLOGY

3.1 Introduction

Experimentation methodology involves a basic **Power Analysis** of Transmitting & Receiving Signals using Signal Generator & Oscilloscope as discussed in section 3.2. Then, **Access Point** based localization is discussed in section 3.3 and finally, object localization using **IEEE 802.11 a/g/p Frame Transceiving** is implemented on **Adalam Pluto** SDR with controlled calibration of RSSI based on Frame Error Rate (FER) is discussed in section 3.4

3.2 Power Analysis of Wireless Signals

We examine the feasibility of the proposed model by transmitting, and receiving the reflected signal by using available transmitting(Tx) and receiving(Rx) antennas which works in the MHz up to GHz frequency ranges. For this we need to do the following to examine the feasibility of establishing consistency in object detection based on received signal levels.

1. Tx Antenna : A transmitting antenna which is readily available for experimentation and which generates from MHz to GHz frequencies is needed for physical experimentation.
2. Rx Antenna : An antenna capable of detecting the signal having frequencies from MHz to GHz frequencies.
3. Signal generator which is connected to Tx antenna and Oscilloscope connected to Rx antenna as shown in figure 3.1 is used to primarily transmit and receive the signal and later on they are replaced with microprocessor to generate and detect frequencies and process the information. Key parameters that are to be observed are shown in Table 3.1

Figure 3.1: Basic model using signal generator and spectrum analyzer for feasibility test (Tx and Rx antennas of 2.4 GHz are available in AIT TC Lab)

Table 3.1: Key parameters

Key parameters
Transmitted power (P_{tx})
Received power (P_{rx})

Further process of experimentation and its results are discussed in section 4.2 of chapter 4 are to test feasibility of consistency in object localization based on RSSI.

3.3 Object Localization using Wireless Access Point

The following steps involves in making the experiment:

- **Access Point Setup** : Raspberry Pi is booted with Linux Operating System called Raspbian OS. The following setup is required for Access Point creation.
 - Hostapd: It is a WAP software daemon which is used to Setup the Wi-Fi. .
 - Installing and configuring the Hostapd and assigning the channel, internal IP range, DHCP, session time, SSID and WPA2 authentication support for Access Point. It also involves configuring DNS and the routing tables for making the system to access the internet through the ethernet.
 - Client association with the Access Point
- **Client RSSI received reading in Access Point:**
 - Client mobile is connected to the Access Point.
 - Access Point is accessed through Secure Shell (SSH).
 - RSSI receiving: Networking commands are used for finding the RSSI of the particular client.
 - Bash scripts are written to automate the process of writing the RSSI values to a script files as shown in figure 4.7.
- **Filtering:**
 - Instantaneous value of RSSI is fetched as said above to calculate the localization distance.
 - Instantaneous time difference (Δt) between two consecutive values is determined.

Consider x = distance in *meters*, $\dot{x}$ = instantaneous velocity in *meters/second*.

$$\bar{x} = \dot{x}\Delta t + x \quad (\text{Equation 3.1})$$

$$\bar{\dot{x}} = \dot{x} \quad (\text{Equation 3.2})$$

$$\begin{bmatrix} \bar{x} \\ \dot{\bar{x}} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \quad (\text{Equation 3.3})$$

where $\bar{x}$ and $\dot{\bar{x}}$ are predicted values of object position and instantaneous velocity respectively. Equation 3.3 can be re-written as follows as shown in equation Equation 3.4

$$\begin{bmatrix} P_{predicted} \\ V_{velocity} \end{bmatrix} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \times \begin{bmatrix} P_{initial} \\ V_{initial} \end{bmatrix} \quad (\text{Equation 3.4})$$

In Equation 3.4 , $P_{initial}$ and $V_{initial}$ are the previous position and velocity of the object respectively where $P_{predicted}$ and $V_{predicted}$ next values after time stamp Δt . Default distance value of one *meter* and average velocity of 1.4 *meters/second* are used as kalman filter default value as describe in section 2.2 with 0.01 as default system noise. This parameters are used in Kalman filter for evaluating the results as shown in figure 4.13 and 4.14

3.3.1 Algorithmic Flowchart based on WAP

The following flow chart as shown in figure 3.2 describes RSSI calibration and distance estimation with Wireless Access Point.

Figure 3.2: Algorithmic flowchart based on Wireless Access Point

3.3.2 RSSI receiving and Calibration method

As shown in the figure 3.2, initial RSSI feed at one meter distance of client from the WAP is used for calibration. In this, large sample of RSSI feed is recorded for the purpose of calibration and to estimate the finite value of RSSI at one meter distance, is calculated based on Average and Standard deviation of the samples. Large samples of RSSI values are filtered by eliminating the lowest 5% and highest 5% of samples having least probability, $p(x) \leq 0.1$.

In real time mode, distance estimation of client is based on instantaneous RSSI and previous four values of RSSI, so that the error due to immediate large change of RSSI can be corrected.

$$RSSI(dBm) = -10n \log_{10}(d/d_0) \quad (\text{Equation 3.5})$$

Equation 3.5 represents relation between $RSSI(dBm)$ and $distance(meters)$, where $d_0(m)$ is known distance of object and with $RSSI(dBm)$ known at unknown distance $d(m)$ from the WAP system.

$$RSSI(dBm) = -10n \log_{10}(d) + A \quad (\text{Equation 3.6})$$

Equation 3.5 can be written as Equation 3.6 where, A is the $RSSI(dBm)$ known at a distance of object at one meter from the system. Further IEEE 802.11a/g/p methodological process based on calibrated RSSI is discussed in section 3.4 and its experimentation is discussed in section 4.4.

3.4 IEEE 802.11 a/g/p based object Localization with RSSI Calibration

Figure 3.3: Implementation of realtime RSSI receiving and calibration for IEEE 802.11 a/g/p localization

Figure 3.4: Localization Estimation

As shown in the figure 3.3, initial RSSI feed at one meter distance of client from the IEEE 802.11 a/g/p Transceiver is used for calibration. In this, large sample of RSSI feed is recorded for the purpose of calibration and to estimate the finite value of RSSI at one meter distance, is calculated based on Average and Standard deviation of the samples. Large samples of RSSI values are filtered by eliminating the lowest 5% and highest 5% of samples having least probability, $p(x) \leq 0.1$.

In realtime processing, as shown in figure 3.3, RSSI feed is instantaneously recorded and calibrated as discussed in further section 3.4.1 based on the instantaneous FER. Its algorithm part for FER based RSSI calibration is shown in section 3.4.1 and experimental result and analysis are discussed in section 4.4. As shown in figure 3.3 consideration of RSSI for $RSSI_{(updated)}$ at calibration and realtime is based on RSSI variance and FER. In calibration there are large number of samples for finding the best value, it is observed that large variance in RSSI (approximately greater than $1.5dBm$ variance) contributes to the change of approximated RSSI, similarly during realtime measurement approximation of instantaneous RSSI is done by considering instantaneous and last four RSSI values in order to reduce the approximate error, here since the samples are less it is observed that at least variance in RSSI of approx $1.25dBm$ has effect on change in RSSI for localization. Moreover the larger value of variance in $RSSI$ leads to FER which in turn increases the probability of change in $RSSI$. During experimentation it is evaluated as shown in table 4.1 that probability, $(p_{FER>0.52})$ of FER greater than 0.52 is 0.786 which changes the RSSI.

3.4.1 FER Calibration and $\Delta RSSI_{avg}$ Estimation Algorithm

$RSSI$ and $RSSI_{avg}$ is calculated for various values of FER, and the change in $RSSI$, i.e $\Delta RSSI$ is estimated for $RSSI$ Calibration and distance estimation.

Function $RSSI-FER_{d1}()$

```

Data:  $RSSI()$  and  $FER()$  data from generated SDR-RF data
/* Known values of  $RSSI$  and  $FER$  at  $d1$  meters
Result: Calculate  $\Delta RSSI$  for  $FER$ 
Initialization at  $d1$ ;
if  $FER = 0$  then
     $RSSI_{avg0} \leftarrow RSSI();$ 
     $RSSI_{std0} \leftarrow RSSI();$ 
end
else if  $0 < FER \leq 0.5$  then
     $RSSI_{avg0.5} \leftarrow RSSI();$ 
     $\Delta RSSI_{avg0.5} = \Delta (RSSI_{avg0}, RSSI_{avg0.5});$ 
end
else if  $0.5 < FER \leq 0.9999$  then
     $RSSI_{avg0.9999} \leftarrow RSSI();$ 
     $\Delta RSSI_{avg0.9999} = \Delta (RSSI_{avg0.5}, RSSI_{avg0.9999});$ 
end
else
     $exit;$ 
end
end

```

Function $RSSI-FER_{d2}()$

```

Data:  $RSSI - FER_{d1}()$ 
/* Calculate  $RSSI$  at  $d2$  meters based on  $RSSI_{avg0}$ 
Result: Calculate  $\Delta RSSI_{d2}$  for  $FER$ 
 $RSSI_{d2} = f_{xRSSI}(d1, RSSI_{avg0});$ 
if  $FER = 0$  then
     $RSSI_{d2avg0} \leftarrow RSSI();$ 
     $\Delta RSSI_{d2avg0} = \Delta (RSSI_{d2}, RSSI_{d2avg0});$ 
end
else if  $0 < FER \leq 0.5$  then
     $RSSI_{d2avg0.5} \leftarrow RSSI();$ 
     $\Delta RSSI_{d2avg0.5} = \Delta (RSSI_{d2}, RSSI_{d2avg0.5});$ 
end
else if  $0.5 < FER \leq 0.9999$  then
     $RSSI_{d2avg0.9999} \leftarrow RSSI();$ 
     $\Delta RSSI_{d2avg0.9999} = \Delta (RSSI_{d2}, RSSI_{d2avg0.9999});$ 
end
end

```

Algorithm as represented through the functions $RSSI - FER_{d1}()$ and $RSSI - FER_{d2}()$ are used to calculate the change in $RSSI$ (i.e $\Delta RSSI$) with change in FER during calibration and real-time measurements respectively. $RSSI()$ and $FER()$ represents the $RSSI$ and FER data generated through using Software Defined Radio.

In $RSSI - FER_{d1}()$:

- $RSSI()$ is the largest set of $RSSI$ samples generated by SDR at reference distance $d1$ (meters).
- $FER()$ is the largest set of FER values associated with respective to $RSSI()$ at reference distance $d1$ (meters).
- $RSSI_{avg0}$ is the average value of $RSSI()$ samples at reference distance ($d1$ meters) for $FER = 0$.
- $RSSI_{std0}$ is the standard deviation value of $RSSI()$ samples at the reference distance ($d1$ meters) for $FER = 0$.
- $RSSI_{avg0.5}$ is the average value of $RSSI()$ at reference distance ($d1$ meters) for $0 < FER \leq 0.5$.
- $\Delta RSSI_{avg0.5}$ is the difference value between $RSSI_{avg0}$ and $RSSI_{avg0.5}$.
- $RSSI_{avg0.9999}$ is the average value of $RSSI()$ at reference distance ($d1$ meters) for $0.5 < FER \leq 0.9999$.
- $\Delta RSSI_{avg0.9999}$ is the difference value between $RSSI_{avg0}$ and $RSSI_{avg0.9999}$.

At a calibration reference distance of d_1 as shown in function $RSSI - FER_{d1}()$:

- Firstly, $RSSI_{avg0}$ and $RSSI_{std0}$ is calculated from $RSSI()$ when $FER = 0$. This $RSSI_{avg0}$ and $RSSI_{std0}$ are used as a reference values at calibration distance having zero FER .
- Next, if $0 < FER \leq 0.5$ then, $RSSI_{avg0.5}$ is calculated from $RSSI()$.
 - Here, change in $RSSI$ is calculated when there is a change of FER from $FER = 0$ to $0 < FER \leq 0.5$. The calculated change is $\Delta RSSI_{avg0.5}$ which is used as a reference values at calibration distance having FER in range $0 < FER \leq 0.5$.
- Finally, if $0.5 < FER \leq 0.9999$, then $RSSI_{avg0.9999}$ is calculated from $RSSI()$.
 - Here, change in $RSSI$ is calculated when there is a change of FER from $FER = 0.5$ to $0.5 < FER \leq 0.9999$. The calculated change is $\Delta RSSI_{avg0.9999}$ which is used as a reference values at calibration distance having FER in range $0.5 < FER \leq 0.9999$.

In $RSSI - FER_{d2}()$:

- $RSSI_{d2}$ is the calculated $RSSI$ at $d2$ (meters) distance using $RSSI_{avg0}$ obtained at $d1$ reference distance.
- $RSSI()$ (when considered in $RSSI - FER_{d2}()$) is the largest set of $RSSI$ samples generated by SDR at real-time distance, $d2$ (meters).
- $\Delta RSSI_{d2avg0}$ is the average value of $RSSI$ samples generated at $d2$ meters for $FER = 0$.
- $\Delta RSSI_{d2avg0.5}$ is the average value of $RSSI$ samples generated at $d2$ meters for $0 < FER \leq 0.5$.
- $\Delta RSSI_{d2avg0.9999}$ is the average value of $RSSI$ samples generated at $d2$ meters for $0 < FER \leq 0.9999$.
- $\Delta RSSI_{d2avg0.5}$ is the difference value between $RSSI_{d2}$ and $RSSI_{d2avg}$ for $FER = 0$.
- $\Delta RSSI_{d2avg0.5}$ is the difference value between $RSSI_{d2}$ and $RSSI_{d2avg}$ for $0 < FER \leq 0.5$.
- $\Delta RSSI_{d2avg0.9999}$ is the difference value between $RSSI_{d2}$ and $RSSI_{d2avg}$ for $0.5 < FER \leq 0.9999$.

At the real-time measurement distance of d_2 as shown in algorithmic function $RSSI - FER_{d2}()$:
Calibration data from the function $RSSI - FER_{d1}$ and $RSSI()$ during real-time measurement is fetched.

- Firstly, $RSSI$ at $d2$ meters (*i.e* $RSSI_{d2}$) is calculated using the values $d2$, $d1$ and calibrated $RSSI_{avg0}$ at $d1$ meters. $fx_{RSSI}(d1, RSSI_{avg0})$ results the value of $RSSI_{d2}$.
- Next, if $FER = 0$ then, $\Delta RSSI_{d2avg0}$ is calculated.
 - Here, change in calculated $RSSI$ (*i.e* $\Delta RSSI_{d2}$) and measured $RSSI$ (*i.e*, $RSSI_{d2avg0}$) is calculated. The calculated change, $\Delta RSSI_{d2avg0}$ represents the change in $RSSI$ at zero FER .
- Then, if $0 < FER \leq 0.5$ then, $\Delta RSSI_{d2avg0.5}$ is calculated.
 - Here, change in calculated $RSSI$ (*i.e* $\Delta RSSI_{d2}$) and measured $RSSI$ (*i.e*, $RSSI_{d2avg0.5}$) is calculated. The calculated change, $\Delta RSSI_{d2avg0.5}$ represents the change in $RSSI$ for FER values in range $0 < FER \leq 0.5$.
- Finally, if $0.5 < FER \leq 0.9999$
 - Here, change in calculated $RSSI$ (*i.e* $\Delta RSSI_{d2}$) and measured $RSSI$ (*i.e*, $RSSI_{d2avg0.9999}$) is calculated. The calculated change, $\Delta RSSI_{d2avg0.9999}$ represents the change in $RSSI$ for FER values in range $0.5 < FER \leq 0.9999$.

Experimentation analysis using the results for section 3.4 are demonstrated and discussed in section 4.4.

3.5 Hardware

Single-Board Computer

Single Board computer such as Raspberry pi as shown in Figure 3.5 or BeagleBone Black are low cost microprocessors which has revolutionized the way of using tiny computers for Internet of Things applications especially for basic provision of doing research. Generally these computers have moderate performance and can be used for generating narrow band Rf signals .

The latest version of Raspberry Pi has the following features.

Raspberry Pi 3:

- Quad Core 1.2GHz Broadcom BCM 2837 64bit CPU
- 1GB RAM
- BCMBCM43438 wireless LAN and Bluetooth
- Low Energy (BLE) on board
- 40-pin extended GPIO
- 4 USB 2 ports
- 4 Pole stereo output and composite video port
- Full size HDMI
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- Micro SD port for loading your operating system and storing data
- Upgraded switched Micro USB power source up to 2.5A

Figure 3.5: Basic Model of Raspberry pi (*Raspberry Pi foundation., 2016*)

Software Defined Radio (SDR)

Software Defined Radio Peripheral is an inexpensive radio frequency transceivers which finds its application in low range radar engineering for chirp generation. ADALM-PLUTO as shown in Figure 3.6 is the basic inexpensive hardware for transceiving wide range of frequencies.

Figure 3.6: ADALM-PLUTO (*Analog Devices., 2017*)

The following are the features of the ADAM-PLUTO sdr:

- Based on Analog Devices AD9363–Highly Integrated RF Agile Transceiver and Xilinx® Zynq Z-7010 FPGA.
- Radio frequency coverage ranging from 325 MHz to 3.8 GHz.
- 20MHz instantaneous bandwidth.
- 12-bit ADC and DAC.
- Sampling rate upto 61.44 mega samples per second (MSPS)
- One transmitter and one receiver with half or full duplex
- MATLAB, Simulink and GNU Radio sink source block support.
- libio, a C, C++, C, and Python API.
- Linux, Windows and OS X operating system support.
- USB 2.0 powered interface.

3.6 Software

GNU Radio - SDR software

GNU Radio [13] is a free and open source software toolkit for signal processing for implementing software defined radios. It has features of simulation environment and also for interfacing hardware for generating and processing signals. GNU Radio along with SDR is used in Raspberry Pi for generation of RF signals.

BASH

Bash is a command line scripting language. It is a core component of Linux based systems. For users, this is mostly used for automating the system process through series of command execution written in bash. In this thesis, bash scripting is used for automating the system process while performing the experiment.

Python - Programming Language

Python is a general-purpose language. In this thesis, it is mainly used for performing numerical and scientific computations.

Operating System

Linux operating system such as Debian, Ubuntu or, Raspbian OS especially for raspberry pi is used . As these distributions are based on Linux kernel which Free and Open source Software giving flexibility of editing the source code and also adding addon's applications. Raspberry pi is mounted with Linux operating System and addon's such as Hostapd.

3.7 Proposed Model

Figure 3.7: Proposed Hardware model

Figure 3.8: Connection diagram for proposed model

Figure 3.7 is the physical realization of proposed model with Raspberry Pi, Pluto SDR and respective connection diagram is shown in figure 3.8.

CHAPTER 4

EXPERIMENTATION RESULTS

4.1 Introduction

In this chapter, experimentation results and analysis are done for Power Analysis in the section 4.2, for Wireless Access Point based localization in the section 4.3 and IEEE 802.11 a/g/p RSSI calibration based localization in section 4.4.

4.2 Power Analysis

The following are the experimentation results of Power Analysis methodology as discussed in section 3.2 of chapter 3.

Figure 4.1: Power Analysis Experimentation Scenario

Experimentation results are performed as describe in the following:

- Obstacle is introduced at different distance (LOS), moving in range from left of transmitter(Tx) to the receivers(Rx) right scaling from -15cm to 45 cm as shown in Figure 4.1 and observations are recorded.
- Experiment is repeated for three times (at different timings)with a stamp for every 3cm.
- For each time and for every stamp of 3cm, 6 values are observed and average are taken into consideration for every graph analysis.

Figure 4.2: Power Analysis Experimentation 1

Figure 4.3: Power Analysis Experimentation 2

Figure 4.4: Power Analysis Experimentation 3

Power Analysis: Graph Analysis: Graphs represented through figure 4.2, figure 4.3 and figure 4.4 is a graphical signal representation of experimentation scenario in figure 4.1. These three graphs (figures 4.2, 4.3 and 4.4) are generated using the same methodology for the scenario in figure 4.1, but the experimentation is performed at different time. Purpose of this to lay and analogy for using power analysis based experimentation for localization, which is discussed in the following sections.

For analysis we divide the scenario into four parts such as left of Tx (-15cm to 0cm), right of Tx (0cm to 15cm), left of Rx (15cm to 30cm) and right of Tx (30cm to 45cm).

- **Right of Tx (0 cm to 15 cm) :** $Rx\text{ dBm}$ increases till the mid value of right of Tx (0 cm to 15 cm), about 7.5 cm then the graph starts decreases from that midpoint to endpoint of right of Tx (0 cm to 15 cm), i.e mid point of Tx - Rx distance.

Here the effect of small sidelobes are reduced till the quarter distance of 30 cm due to which the graph increases and in the next quarter often sidelobes effect is not nullified and the signal reflection from obstacle can cause a decrease of signal level due to which graph falls down.

- **Left of Rx (15 cm to 30 cm) :** from mid point of Tx - Rx , graph starts increasing till the mid point of left of Rx (15 cm to 30 cm), about at 21 cm and then the graphs start decreasing until the end point of left of Rx (15 cm to 30 cm), i.e at 30 cm near the receiver. The value of received dBm is low at 30 cm point because of blockage of the signal from the transmitter(Tx) to the receiver(Rx) by the obstacle.
- **Left of Tx (-15 cm to 0 cm):** Similar effect, Transmitted power level is less when obstacle is at Tx (0 cm) and it increases as the obstacle moves from farther left of Tx and starts decreasing from

the quarter point (0 cm to -15cm) i.e the effect of side lobe is observed and it starts increasing from point the mid point when the sidelobe effect is reduced. And then the reflections from the main lobe due to the obstacle can cause the signal to interference and the signal attenuation is observed at the receiver.

- **Right of Rx(30cm to 45 cm) :** Similar effect of sidelobe and main lobe reflection by the obstacle when it moves from right to farther right of Rx can cause increase and decrease of signal power.

Ratio of Average received signal in dBm when there is no presence of obstacle to the received signal dBm when obstacle is introduced at instantaneous distance from Tx is approximately equal for all the experiments shown in graphs represented through figure 4.2, figure 4.3 and figure 4.4

$$PowerAnalysisExperiment(\frac{Average Rx dBm received when no obstacle}{Rx dBm received instantaneously when obstacle is present})$$

(Equation 4.1)

From the graphical analysis represented through figure 4.2, figure 4.3 and figure 4.4, an inference can be laid that the instantaneous value of the equation 4.1 is approximately equal for all the three experiments (as represented through figure 4.2, figure 4.3 and figure 4.4).

4.3 Wireless Access Point Localization

In general, the proposed system consist of Raspberry Pi (as Access Point) and a Software Defined Radio which is used for object localization. Here as Raspberry pi is used as an Access Point. This can be used as single Access Point localization along with the object localization with the Two receivers and transmitters of SDR's which can increase the accuracy of detection. The following section performs the experimentation in finding the distance of object from the Raspberry Pi Access point.

4.3.1 Experimentation for distance determination

After the client is associate with the Access Point. RSSI values of the client at the Access Point is obtained and the distance calculation is done. This experimentation tells that how accurate that the single Wi-Fi access point with single antenna is.

- The readings of the RSSI received at the AP is read by AP, by placing the Mobile client (connected to AP) at varies distances ranging from one meter to five meters with an interval of one meter. This experimentation is performed twice. Average RSSI values are calculated and the distance estimation is done in the later section s of the documents. The experimentation scenario is observed in the figure 4.5
- Again Now the AP is placed at the center and the Mobile client is made to move from left of AP and also to Right of AP as shown in figure 4.6

Figure 4.5: Experiment, when Mobile client is moving from Access Point

Figure 4.6: Experiment, When mobile client is moving from left and right of AP

```

11: signal: -60 [-60] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-740697703
11: signal: -60 [-60] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-764828441
11: signal: -60 [-60] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-788884335
11: signal: -60 [-60] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-812890750
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-836908519
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-861178215
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-885174630
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-909098129
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-933435220
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-957400333
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:56 PM +07-56-981573675
11: signal: -62 [-62] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-005694465
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-029806452
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-053784065
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-078076730
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-102134134
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-126045862
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-150103058
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-174133953
11: signal: -64 [-64] dBm
3:====Time stamp:====+ 2018-02-25-03:29:57 PM +07-57-197070620

```

Figure 4.7: RSSI reception by using bash shell scripting

We know that RSSI to the distance conversion is as shown in the Equation 4.2. Graphs in Figures 4.8, 4.9, 4.10 and 4.11 are plotted by showing how an observed(calculated) distance value is relative to the ground truth.

$$RSSI(dBm) = -10n \log_{10}(d/d_0) \quad (\text{Equation 4.2})$$

where $d_0(m)$ is known distance of object and with $RSSI$ known at unknown distance of d meters. Equation 4.2 can be written as Equation Equation 4.3 where, A is the $RSSI(dBm)$ known at a distance of object at 1m from the system.

$$RSSI(dBm) = -10n \log_{10}(d) + A \quad (\text{Equation 4.3})$$

Distance calculation is done using Equation 4.3 at varies values of path loss component (n) as as described in scenario figure 4.5 and figure 4.6.

Following graphs represented through figure 4.8, figure 4.9, figure 4.10 and figure 4.11 are the graphs for the varies values of path loss component n .

Figure 4.8: Actual distance vs. estimated distance for every one-meter range from experiment 1

Figure 4.9: Actual distance vs. estimated distance for every one-meter range from experiment 2

The following Figures 4.10 and 4.11 represents how the estimated and the actual ground truth distances

are varied from the mobile client to the Access Point from different directions.

Figure 4.10: Actual distance vs. estimated distance for every one-meter range to the left and right of the AP from experiment 1

Figure 4.11: Actual distance vs. estimated distance for every one-meter range to the left and right of the AP from experiment 2

Figure 4.12: Real-time values of WAP based localization on dashboard

Graphs shown in figure 4.13 and 4.14 represents the real time dynamic object localization and its Kalman filtered output for both distance(*meters*) and velocity(*meters/sec*). Figure 4.7 and figure 4.12 shows the real-time values of WAP based localization on dashboard, where the scripts are written in for the purpose, in Python and Linux Bash Shell for system automation.

Figure 4.13: Kalman filtered distance and real-time distance

Figure 4.14: Kalman filtered velocity and real-time velocity

The above results shows that distance estimation by using RSSI received at single Access Point. But the accuracy can be increased by using SDR for localizing IEEE 802.11 a/g/p based Transceiving devices.

As the proposed system contains Access Point and SDR can accurately detect the object type and can locate them accurately.

4.4 RSSI Based on IEEE 802.11 a/g/p implementation

RSSI based transceiving is implemented using IEEE 802.11 a/g/p implementation on Adlam Pluto SDR as demonstrated in the section 3.4 of chapter 3. In this, Instantaneous RSSI through frame receiving is done by the Software Defined Radio receiver and Frame Error Rate (FER) estimation is done. Then analysis of Frame Error Rate (FER) is done and conclusions are drawn for accurate estimation of RSSI for localization with MAC address of object.

4.4.1 Observed FER and its Analysis

For the purpose of FER analysis, a large number of RSSI samples and their associated FER values are recorded. This is done by using IEEE 802.11 a/g/p based receivers and transmitters at various indoor environments with the similar setups as showing the figure 4.23. This experimentation is performed at different time intervals to approximate the standardized FER for all possible indoor positioning environments.

Following graph shown in figure 4.15 is a density analysis of normalized count of RSSI samples (of about two thousand in number) for various Frame Error Rate (FER). Density analysis shows that most of the samples have FER range in between 0.5 and 0.9999.

Figure 4.15: Frame Error Rate Analysis

4.4.2 Analysis based on FER Probability

Table 4.1: Probability of FER Occurrence

FER Probability _(p)	Value
$p_{(0 <= FER < 0.02)}$	= 0.0360
$p_{(0.02 <= FER < 0.50)}$	= 0.0000
$p_{(0.50 <= FER < 0.52)}$	= 0.1780
$p_{(0.52 <= FER < 0.66)}$	= 0.0000
$p_{(0.66 <= FER < 0.68)}$	= 0.0295
$p_{(0.68 <= FER < 0.75)}$	= 0.0000
$p_{(0.75 <= FER < 0.89)}$	= 0.2745
$p_{(0.89 <= FER < 0.98)}$	= 0.4820

From the table 4.1, as shown, an inference can be laid down as shown in figure 4.2.

Table 4.2: Probability of FER Occurrence

$p_{(FER=0.00)} = 0.0360$
$p_{(FER<0.50)} = 0.0360$
$p_{(FER>=0.75)} = 0.7565$

Figure 4.2 shows that probability of occurrence of FER being greater than 0.75 is 0.7565 . In the subsequent steps of this section more analysis is done to lay conclusion on FER associated with RSSI.

Figure 4.16: FER and Distance

Figure 4.17: FER vs. RSSI Variance

Graph shown in figure 4.16 represents varying Variance of FER with Average FER and graph shown in figure 4.17 represents varying Variance of RSSI with respect to FER. As observed there is a huge density of RSSI variance for the FER in range 0.75 - 0.985

Figure 4.18: FER vs. RSSI

Table 4.3: Δ RSSI Calibration Values

<i>$\Delta RSSI$ Change Calculation</i>					
0	0.5	> 0.5	Δ RSSI (0, 0.5)	Δ RSSI (0, > 0.5)	Δ RSSI (0.5, > 0.5)
16.94134078	17.23733984	17.81143611	0.29599906	0.87009533	0.57409627

4.4.3 Distance Estimation based with and without FER based Correction

Figure 4.19: Actual distance vs. estimated distance with and without *FER* based correction

As described in algorithm in the section 3.4.1, graphs shown in figure 4.18 represents change of RSSI (consolidated representation of values at reference distance) with at various FER. From this an inference as shown in table 4.3 is drawn to calculate $\Delta RSSI$ for *RSSI* calibration with *RSSI* change of $0.870 \text{ dBm} \approx 1 \text{ dBm}$. Graph as shown in figure 4.19 represents evaluated distance *v/s* actual distance with *FER* based correction and without *FER* based correction.

Figure 4.20: Actual distance and estimated error with and without *FER* based correction

Figure 4.21: Actual distance and estimated error percentage with and without *FER* based correction

Graphs as shown in figure 4.20 and figure 4.21 represents the estimated error and estimated error percentage for with and without *FER* based correction respectively.

Table 4.4: Error percentage in distance estimation with and without *FER*

	Error Minimum (%)	Error Maximum (%)	Error Average (%)
With <i>FER</i>	0.2825	21.0487	4.3013
Without <i>FER</i>	2.0000	16.7395	7.8318

Figure 4.22: Error percentage in distance estimation with and without *FER*

4.4.4 Comparison with existing model

Estimated error values for actual distance vs. estimated distance while considering with *FER* and without *FER* based correction of *RSSI* for the graph scenario shown in figure 4.19 is represented in table 4.4 and in graph figure 4.22. In comparison with the existing model such as NLR (Zan Li et al., 2009) with WC-CWLS, average error of NLR model is $1.7m$ and the average error of our model without *FER* based correction is $16.319cm$ and with *FER* based correction is $6.242cm$. Our model outperforms the existing models thereby concluding the consideration of *FER* for object localization implemented using IEEE 802.11/a/g/p based devices is needed.

Figure 4.23 shows various implementation stages during the experiment. Images shown in this figure represents the experimentation setup (1), high clutter region (2, 6, 7), with obstacle (4), Line of Sight(LOS) (2) and Non-LOS diagonal (5) . Various implementation stages are discussed in the section 4.4.5.

Figure 4.23: Experimentation setup at various stages

4.4.5 Implementation

Different Implementation scenarios are discussed in figure 4.24, figure 4.27 and 4.29 with their high clutter environment in the Indoor. Their graphs with distance estimation is represented with calibrated RSSI based on respective FER. Graphs shown in figure 4.25 and figure 4.26 are for the implementation scenario represented in figure 4.24. Graphs shown in figure 4.28 is for the implementation scenario represented in figure 4.27. Graph shown in figure 4.30 and figure 4.31 are for the implementation scenario represented in figure 4.29

Figure 4.24: Tx moving away from Rx1 and Rx2

Figure 4.25: Distance estimated by Rx1 for Tx as shown in figure 4.24

Figure 4.26: Distance estimated by Rx2 for Tx as shown in figure 4.24

Figure 4.27: Tx is moved from Left to Right

Figure 4.28: Distance estimated by Rx1 when Tx is moved as shown in figure 4.27

Figure 4.29: Obstacle between Tx and Rx

Figure 4.30: Distance estimated by Rx1 for Tx moving as shown in figure 4.29

Figure 4.31: Distance estimated by Rx2 for Tx moving as shown in figure 4.29

On observation for the implementations as described in section 4.4.5, the mean error is less than 25cm, *i.e* for 5m, average observed distance is 4.75m. Mean error is 5% and mean Accuracy is 95%. The maximum error is almost 90cm for 5m distance, *i.e* at 5cm, observed distance is 4.1m. Maximum error is 18% and minimum Accuracy is 82% .

CHAPTER 5

CONCLUSION

Conducted experiments showed that the use of the FER for correcting the RSSI for distance estimation provides an improvement in localization of objects. In this experiment, the methodological process for approximating the accurate value of RSSI based on FER led to achieving a minimum accuracy of 82% with a mean accuracy of 95% for object localization. During the process, an effort has been put on for analysing the samples of RSSI with respect to FER at various environments. The object of this work is to provide a novel framework for RSSI estimation considering FER with implementation on a low-cost system which can be used for object localization.

Future Scope of the Research:

In future, Single Board Computers (SBCs) can be used for object localization with inbuilt processing of the estimated algorithm. Further, this existing work can be used for improving the localization system by applying deep learning based techniques at the physical layer in wireless communications. Compressive Sensing (CS) can be used for intelligent estimation of spatial and temporal correlation of the Channel State Information (CSI) .

These systems should have capability to sense the compressed transmitted signal. These compressed signals are less prone to frame errors when compared to normal transmitted signal, besides this data driven deep learning algorithms employed at receiver can use predicted CSI based on Bayesian Estimation models for accurate object sensing and localization.

REFERENCE

- Adib, Mao, Kabelac, Katabi and Miller. (2015) Smart Homes that Monitor Breathing and Heart Rate. ACM.
- Adib and Katabi. (2013). See Through Walls with Wi-Fi!. ACM.
- Adib, Kabelac, Katabi and Miller. (2014) 3D Tracking via Body Radio Reflections. MIT.
- Adib, Kabelac, Katabi and Miller. (2014) Multi-Person Localization via RF Body Reflections. MIT.
- Ahmed, Talukder and Monjur. (2008). WiFi Radar: Design and Implementation of an Infrastructure-less Location Tracking System for Passive Environment. IEEE.
- Andersen, Chee, M. Jacob, Pedersen, and Kurner. 2012. Reverberation and absorption in an aircraft cabin with the impact of passengers. IEEE Transactions.
- Bamba, Joseph, B. Andersen, Tanghe, Vermeeren, Plets, Nielsen, and Martens. 2012. Experimental assessment of specific absorption rate using room electro magnetics. IEEE Transactions.
- Chetty, Smith, Guo and Woodbridge. (2009). Target Detection in High Clutter using Passive Bistatic WiFi Radar. IEEE.
- Chetty, Smith and Woodbridge. (2011). Through-the-wall Sensing of Personnel Using Passive Bistatic WiFi Radar at Standoff Distances. IEEE.
- Falcone, Colone, Macera and Lombardo. (2012). Localisation and tracking of moving targets with WiFi-based passive radar. IEEE.
- Falcone, Colone, Bongioanni and Lombardo. (2010). Experimental Results for OFDM WiFi-Based Passive Bistatic Radar. IEEE.
- Findlay and Dimbylow. 2008. Calculated SAR distributions in a human voxel phantom due to the reflection of electromagnetic fields from a ground plane between 65 MHz and 2 GHz. Physics in Medicine and Biology.
- Guo, Woodbridge, and C. J. Baker. (2008). Evaluation of Wifi Beacon transmission for wireless based passive radar. IEEE.
- GNU Radio with Raspberry pi for SDR applications,
<https://www.rs-online.com/designspark/taking-the-raspberry-pi-2-for-a-test-drive-with-gnu-radio-2>
- GNU radio for Raspberry pi,
<https://github.com/scateu/GNURadioForRaspberryPiOnArchlinux>
- GNURadio, from
<https://www.gnuradio.org/>
- Harima. 2012. Estimation of power absorbed by human body using reverberation chamber. IEEE.

Hirata, Nagaya, Osamu, Nagaoka and Watanabe. 2007. Correlation between absorption cross section and body surface area of human for far-field exposure at GHz bands. IEEE. International Symposium on Electromagnetic Compatibility.

Hostpd for Raspberry pi,
<https://wireless.wiki.kernel.org/en/users/documentation/hostapd>

Hurst and Ellingson. 2008. Path loss from a transmitter inside an aircraft cabin to an exterior fuselage-mounted antenna.. IEEE Transactions.

Kientega, Conil, Hadjem, Richalot, Gati, Wong, Picon, and Wiart. 2011. A surrogate model to assess the whole body sar induced by multiple plane waves at 2.4 GHz. annals of telecommunications - annales des telecommunications.

Melia, G. (2013). Electromagnetic Absorption by the Human Body from 1 to 15 GHz. (Doctoral thesis, The University of York).

Mishra and Potnis. (2017). Wireless Transmission Detection and Monitoring System using GNU Radio and Multiple RTL-SDr Receivers. International Journal of Engineering Research and Application.

Nezirovic A., Liu M., Yarovoy A.G. 2007. Modelling of Reflection of UWB Pulses from Trapped Human Beings. In: Baum C.E., Stone A.P., Tyo J.S. (eds) Ultra-Wideband Short-Pulse Electromagnetics 8. Springer, New York, NY.

Pluto sdr,
<https://wiki.analog.com/university/tools/pluto>

Robinson, Clegg, and Marvin. Radio frequency electromagnetic fields in large conducting enclosures: effects of apertures and human bodies on propagation and field- statistics. IEEE Transactions on Electromagnetic Compatibility.

Rzewuski and Kulpa. (2011). System Concept of WIFI Based Passive radar. INTL JOURNAL OF ELECTRONICS AND TELECOMMUNICATIONS.

Raspberry pi 3,
<https://www.raspberrypi.org/products/raspberry-pi-3-model-b/>

APPENDIX A1

ACCESS POINT LOCALIZATION

Bash Shell Script

```
”—————Name this file as mainbash_1.sh—————”
#!/bin/bash
case "$1" in
—calibrate)
    echo 'In calibration mode'
    echo 'executing raspap_calibration.sh'
    #./raspap_calibration.sh !#
    Counter=0
    while [[ $Counter -le 1 ]]; do
        ./raspap_calibration.sh
    done
    ;;
—realtime)
    echo 'In realtime mode'
    Counter=0
    while [[ $Counter -le 1 ]]; do
        sed -i '/NONE/Id' kalman_distance_values.txt
        sed -i '/NONE/Id' kalman_velocity_values.txt
        ./raspap_realtime.sh
        python mainscript_1.py
        python kalman_filter_design.py
        python normal_velocity_update.py
        ./time_interval.sh

        echo '====='
    done
    ;;
*)
    echo ""It seems that you used a wrong way!. USe the following formates""
    echo
    echo './mainbash_1.sh —calibrate'
    echo
    echo './mainbash_1.sh —realtime'
    ;;
esac

”—————End of mainbash_1.sh—————”
```


APPENDIX A2

ACCESS POINT LOCALIZATION

Bash Shell Script

```
"-----Name this file as default.sh-----"
#!/bin/bash
case "$1" in
    --all)
        echo "Setting Every *.log and *.txt files to default ..."
        echo "Within 5 secs , Press CTRL+D to stop Here"
        sleep 5
        echo ". . ."

        echo " "

        echo " Setting Kalman_distance_values.txt to default..."
        bash -c "date >> Log_Files/kalman_distance_values.txt"
        bash -c "cat kalman_distance_values.txt >>
Log_Files/kalman_distance_values.txt"
        echo "Appended kalman_distance_values.txt values
to Log_Files folder"
        echo > kalman_distance_values.txt
        echo 1 >>kalman_distance_values.txt
        echo "Kalman_distance_values is
now in default with one initial
Kalman Position as value as 1m"
        echo ". . ."
        echo "Within 3 secs , Press CTRL+D to stop Here"
        sleep 3
        echo "Proceeding next"
        echo ". . ."
        echo " "

        echo " Setting kalman_velocity_values.txt to default..."
        bash -c "date >>Log_Files/kalman_velocity_values.txt "
        bash -c "cat kalman_velocity_values.txt >>
Log_Files/kalman_velocity_values.txt"
        echo "Appended kalman_velocity_values.txt values to
Log_Files folder"
        echo > kalman_velocity_values.txt
        echo 0 >>kalman_velocity_values.txt
        echo "kalman_velocity_values.txt is
now in default with one initial
Kalman Velocity value as 0 m/sec"
        echo ". . ."
        echo "Within 3 secs , Press CTRL+D to stop Here"
```

```

sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting time_distance.txt to default..."
bash -c "date >> Log_Files/time_distance.txt"
bash -c "cat time_distance.txt >> Log_Files/time_distance.txt"
echo "Appended time_distance.txt values to Log_Files folder"
echo > time_distance.txt
echo "time_distance.txt is now in default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting obj_timedif_distance_velocity.txt to default..."
bash -c "date >>
Log_Files/obj_timedif_distance_velocity.txt"
bash -c "cat obj_timedif_distance_velocity.txt >
Log_Files/obj_timedif_distance_velocity.txt"
echo "Appended obj_timedif_distance_velocity.txt
values to Log_Files folder"
echo > obj_timedif_distance_velocity.txt
echo "obj_timedif_distance_velocity.t
xts is now in default with null value"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting original_time.txt to default..."
bash -c "date >> Log_Files/normal_velocity.txt"
bash -c "cat original_time.txt >>
Log_Files/original_time.txt"
echo "Appended normal_velocity.txt
values to Log_Files folder"
echo > original_time.txt
echo "original_time.txt is now in
default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"

```

```

echo ". . ."
echo " "

echo " Setting time_interval.txt to default..."
bash -c "date >> Log_Files/time_interval.txt"
bash -c "cat normal_velocity.txt >>
Log_Files/time_interval.txt"
echo "Appended time_interval.txt values to Log_Files folder"
echo > time_interval.txt
echo "time_interval.txt is now in
default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to
stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

```

```

echo " Setting normal_distance.txt to
default..."
bash -c "date >>
Log_Files/normal_distance.txt"
bash -c "cat normal_distance.txt >>
Log_Files/normal_distance.txt"
echo "Appended normal_distance.txt

```

```

values to Log_Files folder"
echo > normal_distance.txt
echo "normal_distance.txt is now in
default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to
stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

```

```

echo " Setting normal_velocity.txt to
default..."
bash -c "date >> Log_Files/normal_velocity.txt"
bash -c "cat normal_velocity.txt >>
Log_Files/normal_velocity.txt"
echo "Appended normal_velocity.txt
values to Log_Files folder"

```

```

echo > normal_velocity.txt
echo "normal_velocity.txt is now in
default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to
stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting normal_velocity_update.txt to default..."
bash -c "date >>
Log_Files/normal_velocity_update.txt"
bash -c "cat
normal_velocity_update.txt >>
Log_Files/normal_velocity_update.txt"
echo "Appended normal_velocity_update.txt values to
Log_Files folder"
echo > normal_velocity_update.txt
echo "normal_velocity_update.txt is
now in default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting raspap_calibration.txt to default..."
bash -c "date >>
Log_Files/raspap_calibration.txt"
bash -c "cat raspap_calibration.txt
>> Log_Files/raspap_calibration.txt"
echo "Appended raspap_calibration.txt
values to Log_Files folder"
echo > raspap_calibration.txt
echo "raspap_calibration.txt is now
in default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting raspap_realtime.txt to default..."
bash -c "date >>

```

```

Log_Files/raspap_realtime.txt"
bash -c "cat raspap_realtime.txt >>
Log_Files/raspap_realtime.txt"
echo "Appended raspap_realtime.txt
values to Log_Files folder"
echo > raspap_realtime.txt
echo "raspap_realtime.txt is now in
default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to
stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo ". . . . . ."
echo "Restored these to defaults :
kalman_distance_values.txt , kalman_velocity.txt ,
time_distance.txt ,
obj_timedif_distance_velocity.txt ,
normal_distance.txt , normal_velocity.txt ,
raspap_calibration.txt and raspap_realtime.txt  "
;;
—help)
echo "run 'default.sh' for making
kalman_distance_values.txt ,
kalman_velocity.txt ,
time_distance.txt ,
obj_timedif_distance_velocity.txt ,
normal_distance.txt ,
normal_velocity.txt, to default
initial state leaving
'raspap_calibration.txt' &
'raspap_realtime.txt' in the current state "
echo "run 'default.sh —all' for making every *.log and *.txt
files to its default initial state..."
echo "run 'default.sh —help' for any
help"
;;
*)
echo "Setting *.log and *.txt files to default state except
'raspap_calibration.txt' &
'raspap_realtime.txt' ..."
echo "Within 5 secs , Press CTRL+D to stop Here"
sleep 5
echo ". . ."

```

```

echo " "

echo " Setting Kalman_distance_values.txt to default..."
bash -c "date >>
Log_Files/kalman_distance_values.txt"
bash -c "cat kalman_distance_values.txt >>
Log_Files/kalman_distance_values.txt"
echo "Appended kalman_distance_values.txt values to
Log_Files folder"
echo > kalman_distance_values.txt
echo 1 >>kalman_distance_values.txt
echo "Kalman_distance_values is now
in default with one initial Kalman Position as value as 1m"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting kalman_velocity_values.txt to default..."
bash -c "date >>Log_Files/kalman_velocity_values.txt "
bash -c "cat kalman_velocity_values.txt >>
Log_Files/kalman_velocity_values.txt"
echo "Appended kalman_velocity_values.txt values to
Log_Files folder"
echo > kalman_velocity_values.txt
echo 0 >>kalman_velocity_values.txt
echo echo "kalman_velocity_values.txt
is now in default with one initial
Kalman Velocity value as 0 m/sec"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting time_distance.txt to default..."
bash -c "date >> Log_Files/time_distance.txt"
bash -c "cat time_distance.txt >>
Log_Files/time_distance.txt"
echo "Appended time_distance.txt
values to Log_Files folder"
echo > time_distance.txt
echo "time_distance.txt is now in
default with null"
echo ". . ."

```

```

echo "Within 3 secs , Press CTRL+D to
stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting obj_timedif_distance_velocity.txt to
default..."
bash -c "date >> Log_Files/obj_timedif_distance_velocity.txt"
bash -c "cat obj_timedif_distance_velocity.txt >>
Log_Files/obj_timedif_distance_velocity.txt"
echo "Appended obj_timedif_distance_velocity.txt
values to Log_Files folder"
echo > obj_timedif_distance_velocity.txt
echo "obj_timedif_distance_velocity.txt
s is now in default with nul value"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting original_time.txt to default..."
bash -c "date >>
Log_Files/normal_velocity.txt"
bash -c "cat original_time.txt >>
Log_Files/original_time.txt"
echo "Appended normal_velocity.txt
values to Log_Files folder"
echo > original_time.txt
echo "original_time.txt is now in
default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to
stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting time_interval.txt to default..."
bash -c "date >>
Log_Files/time_interval.txt"
bash -c "cat normal_velocity.txt >>
Log_Files/time_interval.txt"
echo "Appended time_interval.txt

```

```

values to Log_Files folder"
echo > time_interval.txt
echo "time_interval.txt is now in default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting normal_distance.txt to default..."
bash -c "date >>
Log_Files/normal_distance.txt"
bash -c "cat normal_distance.txt >>
Log_Files/normal_distance.txt"
echo "Appended normal_distance.txt values to Log_Files folder"
echo > normal_distance.txt
echo "normal_distance.txt is now in default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo " Setting normal_velocity.txt to default..."
bash -c "date >> Log_Files/normal_velocity.txt"
bash -c "cat normal_velocity.txt >>
Log_Files/normal_velocity.txt"
echo "Appended normal_velocity.txt
values to Log_Files folder"
echo > normal_velocity.txt
echo "normal_velocity.txt is now in default with null"
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo ". . ."

echo " Setting normal_velocity_update.txt to default..."
bash -c "date >>
Log_Files/normal_velocity_update.txt"
bash -c "cat normal_velocity_update.txt >>
Log_Files/normal_velocity_update.txt"
echo "Appended normal_velocity_update.txt values to
Log_Files folder"
echo > normal_velocity_update.txt

```



```

echo "normal_velocity_update.txt is now in default with null"
echo ". . ."
echo "Within 3 secs , Press CTRL+D to stop Here"
sleep 3
echo "Proceeding next"
echo ". . ."
echo " "

echo ". . ."
echo ". . ."
echo " "
echo ". . . . . "
echo "_____ "
echo "Restored to Default : 'kalman_distance_values.txt ',
'kalman_velocity_values.txt ',
'time_distance.txt ',
'obj_distance_timediff.txt ',
'normal_distance.txt ',
'normal_velocity.txt '"
echo "Not Restored to default:
'raspap_calibration.txt ' &
'raspap_realtime.txt ' are kept same"
echo "_____ "
;;
esac

"_____End of default.sh_____ "

```

APPENDIX A3

ACCESS POINT LOCALIZATION

Bash Shell Script

```
"----- Name this script as raspap_realtime.sh -----"
#!/bin/bash
#echo '====RSSSI of connected devices at '"$(date '+%F%T.%N%Z')'"
>>raspap.log>>/var/www/html/raspap_gen.log
echo "$(iw dev wlan0 station dump)">>raspap.log
#echo "$(grep -Eo '[-+][0-9]|[-+][0-9][0-9]|[-+][0-9][0-9]
[0-9]|![-+]][0-9]|![-+]][0-9][0-9]|![-+]][0-9][0-9][0-9]'
raspap.log)">>/var/www/html/raspap_gen.log
echo "$(grep -Eo '[-+][0-9]|[-+][0-9][0-9]|[-+][0-9][0-9]
[0-9]|![-+]][0-9]|![-+]][0-9][0-9]|![-+]][0-9][0-9][0-9]'
raspap.log)">>raspap_realtime.txt
>raspap.log
"----- End of raspap_realtime.sh -----"
```

APPENDIX A4

ACCESS POINT LOCALIZATION

Bash Shell Script

```
"-----Name this as raspap_calibration.sh-----"
#!/bin/bash
#echo '====RSSSI of connected devices at '"$(date '+%F%T.%N%Z')'"
>>raspap.log>>/var/www/html/raspap_gen.log
echo "$(iw dev wlan0 station dump)">>raspap.log
#echo "$(grep -Eo '[-+][0-9]|[-+][0-9][0-9]|[-+][0-9][0-9]
[0-9]|![-+]][0-9]|![-+]][0-9][0-9]|![-+]][0-9][0-9][0-9]'
raspap.log)">>/var/www/html/raspap_gen.log
echo "$(grep -Eo '[-+][0-9]|[-+][0-9][0-9]|[-+][0-9][0-9]
[0-9]|![-+]][0-9]|![-+]][0-9][0-9]|![-+]][0-9][0-9][0-9]'
raspap.log)">>raspap_calibration.txt
>raspap.log
"-----End of raspap_calibration.sh-----"
```

APPENDIX A5

ACCESS POINT LOCALIZATION

Python Code

```
"""-----Name this script as mainscript_1.py-----"""
##Processing the RSSI values from the file to calculate the
standard deviation and for precision adjustments

import numpy as np
import math as mt
import time as tm
import subprocess as sp
RS_Calb_array=[]
RS_recv_latest=[]
RS_recv_lines_append=[]
#### Initial calibration using known samples of RSSI at distance 1m
#RS_Calb=open("/home/debian/Documents/test.txt","r")
RS_Calb=open("raspap_calibration.txt","r")
for lines in RS_Calb.readlines():
    for i in lines.split():
        RS_Calb_array.append(float(i))
StDev_Calb=np.std(RS_Calb_array)
RS_Calb_min_With_StD=np.mean(RS_Calb_array)-StDev_Calb
RS_Calb_max_With_StD=np.mean(RS_Calb_array)+StDev_Calb
#####Now Fetching Instantaneous RSSI
values and calculating Standard
deviations and the mean.
#RS_recv=open("/home/debian/Documents/test.txt","r")
RS_recv=open("raspap_realtime.txt","r")
for RS_recv_lines in RS_recv.readlines():
    for i in RS_recv_lines.split():
        RS_recv_lines_append.append(float(i))
        #RS_recv_lines_append.append(i)
for i in range(1,4):
    RS_recv_latest.append(RS_recv_lines_append
    [len(RS_recv_lines_append)-i])
StDev_recv=np.std(RS_recv_latest)
RS_recv_min_With_StD=np.mean(RS_recv_latest)-StDev_recv
RS_recv_max_With_StD=np.mean(RS_recv_latest)+StDev_recv

print "The Min RSSI at distance D meters is
=", RS_recv_min_With_StD
print "The Max RSSI at Distance D meters is
=", RS_recv_max_With_StD

n=2.5;
```

```

Time= tm.time()
original_time=open("original_time.txt", "a+")
original_time.write(str(Time)+"\n")
original_time.close()
Distance_recv=mt.pow(10,((RS_Calb_min-With_StD)–
(RS_recv_min-With_StD))/(10*n))
time_distance=open("time_distance.txt", "a+")
time_distance.write("\n"+str(Distance_recv)+"\n")
time_distance.write(str(Time))
time_distance.close()
print "Object is in the radius of ", Distance_recv, "Meters"
time_distance=open("time_distance.txt", "a+")
time_distance_list_lines=list(time_distance.readlines())
T2=time_distance_list_lines[len(time_distance_list_lines)–1]
D2=time_distance_list_lines[len(time_distance_list_lines)–2]
T1=time_distance_list_lines[len(time_distance_list_lines)–3]
D1=time_distance_list_lines[len(time_distance_list_lines)–4]
time_distance.close()
print T2, T1, D2, D1
obj_timedif_distance_velocity=open("obj_timedif_distance_velocity
.txt", "a+")
dt=float(T2)–float(T1)
obj_timedif_distance_velocity.write(str(dt)+'\n')
obj_timedif_distance_velocity.write(str(D2))
Vel=((float(D2))–(float(D1)))/((float(T2))–(float(T1)))
Vel =abs(Vel)
obj_timedif_distance_velocity.write(str(abs(Vel))+'\n')
print "Instantaneous Velocity = ", Vel
normal_distance=open("normal_distance.txt", "a+")
normal_distance.write(str(Distance_recv)+'\n')
normal_distance.close()
normal_velocity=open("normal_velocity.txt", "a+")
normal_velocity.write(str(Vel)+"\n")
normal_velocity.close()
"—————End of manuscript_1.py—————"

```

APPENDIX A6

ACCESS POINT LOCALIZATION

Kalman Filter Python Code

```
"""-----Name This File as kalman_filter_design.py-----"""
import numpy as np
import sys
class kalman_filter_design:
    def par_measure(self):
        n_types=2 #velocity and distance
        n_objects=1 ##For simplicity, can fetch from the file
        obj_timedif_distance_velocity=open("obj_timedif_distance_
        velocity.txt", "a+")
        obj_timedif_distance_velocity_lines=list(obj_timedif_distance_
        velocity.readlines())

        kalman_distance_values=open("kalman_distance_values.txt",
        "a+")
        kalman_distance_values_lines=list(kalman_distance_values
        .readlines())
        Xpprevious=kalman_distance_values_lines[len(kalman_distance_
        values_lines)-1]
        Xprealtime=obj_timedif_distance_velocity_lines[len(obj_
        timedif_distance_velocity_lines)-2]

        kalman_velocity_values=open("kalman_velocity_values.txt",
        "a+")
        kalman_velocity_values_lines=list(kalman_velocity_values.
        readlines())
        Xvprevious= kalman_velocity_values_lines[len(kalman_
        velocity_values_lines)-1]
        Xvrealtime=obj_timedif_distance_velocity_lines[len(obj_
        timedif_distance_velocity_lines)-1]

        print "Position Previous Kalman ==", Xpprevious
        print "Velocity Previous Kalman ==", Xvprevious
        print "Position previous realtime==", Xprealtime
        print "Velocity previous realtime==", Xvrealtime
        print("")
        A=np.array([[1]])
        Q=np.array([[0.01]]) #Amount of noise in the system
        R=np.array([[0.1]]) #Noise in sensors as covariance matrix
        H=np.array([[1,0]])
```

```

dt=obj_timedif_distance_velocity_lines[len(obj_timedif_
distance_velocity_lines)-3]

Xcrealtime=np.vstack((float(Xprealtime), float(Xvrealtime)))
D2=time_distance_list_lines[len(time_distance_list_lines)-2]
T1=time_distance_list_lines[len(time_distance_list_lines)-3]
D1=time_distance_list_lines[len(time_distance_list_lines)-4]
Xcprevious=np.vstack((float(Xpprevious),float(Xvprevious)))
Xcpredict=np.dot(np.array([[float(A),float(dt)],[float(0),
float(1)]]),np.array(Xcprevious))
Pprevious=np.array([[1,0],[0,1.96]])
Ppredict=np.multiply(A,np.multiply(Pprevious,np.transpose(A)))
+Q
Yinv=Xcrealtime-np.dot(H,Xcpredict)
Yinv_co=np.dot(H,np.dot(Ppredict,np.transpose(H))) + R
kf_gain=np.dot(Ppredict,np.dot(np.transpose(H),np.linalg.
inv(Yinv_co)))
Xupdate=Xcpredict+np.multiply(kf_gain,Yinv)
Pupdate=np.dot((np.identity(Ppredict.shape[0])-np.dot(kf_gain,H))
,Ppredict)
Xpupdate,Xvupdate=np.split(Xupdate,n_types)
print "Position update after Kalman =", np.savetxt(sys.stdout,
Xpupdate, fmt="%.7f")
np.savetxt(kalman_distance_values,Xpupdate, fmt="%.7f")
kalman_distance_values.close()
print "Velocity update after Kalman =", np.savetxt(sys.stdout,
abs(Xvupdate), fmt="%.7f")
np.savetxt(kalman_velocity_values,abs(Xvupdate), fmt="%.7f")
kalman_velocity_values.close()
return
kalman_filter_design=kalman_filter_design()
kalman_filter_design.par_measure()
sys.exit
"-----End of kalman_filter_design.py-----"

```

APPENDIX A7

ACCESS POINT LOCALIZATION

Normal Velocity Update Python Code

```
"""----- Name as 'normal_velocity_update.py'-----"""
#This script is used to Update the normal_velocity based on
kalman_distance_update and time Difference, although we have
kalma_velocity_update. Normal velocity is always comparable
to the existing works.
import numpy as np
import math as mt
kalman_distance=open("kalman_distance_values.txt", "a+")
kalman_distance_list_lines=list(kalman_distance.readlines())
KD2=kalman_distance_list_lines[len(kalman_distance_list_lines)-1]
KD1=kalman_distance_list_lines[len(kalman_distance_list_lines)-2]
obj_timedif_distance_velocity=open("obj_timedif_distance_velocity
.txt", "a+")
obj_timedif_distance_velocity_lines=list(obj_timedif_distance
_velocity.readlines())
dt=obj_timedif_distance
_velocity_lines[len(obj_timedif_distance_velocity_lines)-3]
normal_velocity_update_value=((float(KD2))-(float(KD1)))/(float(dt))
normal_velocity_update_value=abs(normal_velocity_update_value)
normal_velocity_update=open("normal_velocity_update.txt", "a+")
normal_velocity_update.write(str(normal_velocity_update_value)+"\n")
normal_velocity_update.close()
print "Normal Velocity Update = ", normal_velocity_update_value

"""-----End of 'normal_velocity_update.py'-----"""
```

Note: All the scripts in **Appendix A1** to **Appendix A7** are customized exclusively for this Thesis and are mostly interdependent on each other. So, it is advisable to have all the scripts when executing the other.

APPENDIX B1

INSTALLATION

Bash Shell Script

```
”—————Name as setup.sh—————”
#!/bin/bash
#./setup.sh is a free software , intend for the purpose
of automatic installation of RF Communication packages
based on GNURadio
#copyright (c) 2016 Author(s): Gorla Praven
<gorlapraveen@gmail.com>
#This program is free software: you can redistribute it and/or
modify
#it under the terms of the GNU General Public License as
published by
#the Free Software Foundation , either version 3 of the
License , or (at your option) any later version .
This program is distributed in the hope that it will be useful ,
#but WITHOUT ANY WARRANTY; without even the implied warranty of
#MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
#GNU General Public License for more details .
#You should have received a copy of the GNU General Public License along with th
If not , see <https://www.gnu.org/licenses/> .

#####

echo checking gnu radio installation and related dependencies...
if [ $(dpkg-query -W -f='${Status}' gnuradio 2>/dev/null
| grep -c "no packages" | wc -l ) -gt 0 ]; then
    echo GNURadio not found
    echo Installing it
    sudo apt -y install gnuradio;
else
    echo 'GNU already Installed.....'
fi
echo
if [ $(dpkg-query -W -f='${Status}' gnuradio-dev libxml2
libxml2-dev bison flex cmake git libaio-dev libboost-all-dev
swig 2>/dev/null | grep -c "no packages" | wc -l ) -gt 0 ]; then
    echo 'ADLAM PLuto SDR dependencies not found for GNU Radio... '
    echo 'Installing Adlam Pluto Dependencis for GNU radio.....'
    sudo apt -y install gnuradio-dev libxml2 libxml2-dev bison flex
    cmake git libaio-dev libboost-all-dev swig;
    sudo apt -y -f install;
else
    echo 'Adlam Pluto Dependencies for GNU already Installed.....'
```



```

fi
echo
if [ $(dpkg-query -W -f='${Status}' libaio-dev libusb-1.0-0-dev
libserialport-dev libxml2-dev libavahi-client-dev autoconf
2>/dev/nul
l | grep -c "no packages" | wc -l ) -gt 0 ]; then
    echo SerialCommunication packages...
    echo Installing them....
    sudo apt -y install libaio-dev libusb-1.0-0-dev
    libserialport-dev libxml2-dev libavahi-client-dev autoconf;
    sudo apt -y -f install;
else
    echo 'SerialCommunication packages Already Installed '
fi
echo #####
if [ $(dpkg-query -W -f='${Status}' sigrok* libserialport0
libserialport-dev 2>/dev/null | grep -c "no packages"
| wc -l ) -gt 0 ]; then
    echo Other SerialCommunication packages...
    echo Installing them....
    sudo apt -y install sigrok* libserialport0 libserialport-dev;
    sudo apt -y -f install;

else
    echo 'Required SerialCommunication packages Already Installed '
fi

sudo apt -y install gr-iio* libad9361* libgnuradio-iio0 libiio*;

echo
mkdir Adlam_pluto_required
bash -c "cd Adlam_pluto_required && git clone https://github.com/
analogdevicesinc/libiio.git gr-libiio"
bash -c "cd Adlam_pluto_required/gr-libiio/ && cmake ."
bash -c "cd Adlam_pluto_required/gr-libiio/ && make"
bash -c "cd Adlam_pluto_required/gr-libiio/ && sudo make install"

bash -c "cd Adlam_pluto_required && git clone https://github.com/
analogdevicesinc/libad9361-iio.git gr-libad9361-iio"

bash -c "cd Adlam_pluto_required/gr-libad9361-iio/ && cmake ."
bash -c "cd Adlam_pluto_required/gr-libad9361-iio/ && make"
bash -c "cd Adlam_pluto_required/gr-libad9361-iio/ && sudo
make install"

```

```

bash -c "cd Adlam_pluto_required && git clone https://github.com/
analogdevicesinc/gr-iio.git gr-iio"
bash -c "cd Adlam_pluto_required/gr-iio/ && cmake
-DCMAKE_INSTALL_PREFIX=/usr ."
bash -c "cd Adlam_pluto_required/gr-iio/ && make"
bash -c "cd Adlam_pluto_required/gr-iio/ && sudo make install"
bash -c "sudo ldconfig"

bash -c "cp -r /usr/local/lib/python2.7/dist-packages/gnuradio/
iio /usr/lib/python2.7/dist-packages/gnuradio/"

echo

echo
echo Installation of Adlam pltuio is completed
echo
echo
echo Checking Installtion of IEEE802.11 GNU radio based libraries.

echo Installing required packages

sudo apt-get install liblog4cpp5-dev
bash -c "cd Adlam_pluto_required &&
git clone https://github.com/bastibl/gr-foo.git gr-foo"
bash -c "cd Adlam_pluto_required/gr-foo && git checkout master"
bash -c "mkdir Adlam_pluto_required/gr-foo/build"
echo 'Installing the gr-foo, a meta package required for
GNU Radio IEEE802.11 Implementation'

bash -c "cd Adlam_pluto_required/gr-foo/build && cmake .."
bash -c "cd Adlam_pluto_required/gr-foo/build && make"
bash -c "cd Adlam_pluto_required/gr-foo/build && sudo make install"
bash -c "sudo ldconfig"

echo

bash -c "cd Adlam_pluto_required && git clone https://github.
com/bastibl/gr-ieee802-11.git gr-ieee802-11"
bash -c "mkdir Adlam_pluto_required/gr-ieee802-11/build"
echo 'Installing the GNU Radio IEEE802.11 Implementation'
bash -c "cd Adlam_pluto_required/gr-ieee802-11/build && cmake .."
bash -c "cd Adlam_pluto_required/gr-ieee802-11/build && make"
bash -c "cd Adlam_pluto_required/gr-ieee802-11/build && sudo make
install"
bash -c "sudo ldconfig"

```

```

echo
echo Adjusting the Maximum Shared Memory to 2147483648

sudo sysctl -w kernel.shmmax=2147483648

bash -c "cd ../ && python wifi_phy_hier.py"
sudo mkdir ~/.gnuradio
echo [grc]>>~/.gnuradio/config.conf
echo global_blocks_path = /opt/local/share/gnuradio/grc/blocks >>
~/.gnuradio/config.conf
echo local_blocks_path = /Users/basti/usr/share/gnuradio/grc
/blocks >>~/.gnuradio/config.conf
echo
echo Installing gfortran, lapack, itpp packages

sudo apt -y install gfortran
sudo apt -y install libatlas-base-dev libblas-dev liblapack-dev
bash -c "cd ../required/itpp-4.3.1/ && mkdir build"
bash -c "cd ../required/itpp-4.3.1/build && cmake .."

bash -c "cd ../required/itpp-4.3.1/build && make"
bash -c "cd ../required/itpp-4.3.1/build && sudo make install."

sudo apt -y install libatlas-base-dev libblas-dev liblapack-dev
sudo apt -f install

echo 'Installing RF-TAP, gr-rftap libraries for gnu radio'
bash -c "cd Adlam_pluto_required && git clone https://
github.com/rftap/gr-rftap gr-rftap"
bash -c "mkdir Adlam_pluto_required/gr-rftap/build"
echo 'Installing the gr-rftap, a meta package required for
GNU Radio IEEE802.11 Implementation'
bash -c "cd Adlam_pluto_required/gr-rftap/build && cmake .."
bash -c "cd Adlam_pluto_required/gr-rftap/build && make"
bash -c "cd Adlam_pluto_required/gr-rftap/build && sudo
make install"
bash -c "cd Adlam_pluto_required/ && sudo ldconfig"

echo Installing wireshark and tshark.....
sudo apt -y install wireshark*
sudo apt -y install tshark*
sudo apt -y -f install
echo
echo
echo _____

```

APPENDIX B2

IEEE802.11 a/g/p

Bash Shell Script

```
"-----Name as main_output_newfile.sh-----"
#!/bin/bash
#example when the main.sh(this script) is initiated with following
"./main.sh R1 1.5" means taking measurement of of receiver(R1)
and distance from transmitter from being at 1.5 metere
if [[ $1 == R1 ]]; then
echo "Executing in GNU Radio R1 Mode"
case "$2" in
0)
echo "executing R2 at generic mode"
echo "probably at 0 m"
echo "Find the results [instantaneous] at "
echo "For pcap : pcap/wifi_pluto_rx_general.pcap"
echo "For Results: results/wifi_pluot_rx_RSSI_general.log "
bash -c "gnuradio-companion wifi_pluto_rx_general.grc"
bash -c "tshark -r pcap/wifi_pluto_rx_general.pcap >
results/wifi_pluot_rx_RSSI_general.log"
;;
0.25)
echo "at 0.25m "
bash -c "gnuradio-companion wifi_pluto_rx1_0_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_0_25.pcap >
results/wifi_pluto_rx1_RSSI_0_25.log"
;;
0.5)
echo "at 0.5m"
bash -c "gnuradio-companion wifi_pluto_rx1_0_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_0_5.pcap >
results/wifi_pluto_rx1_RSSI_0_5.log"
;;
0.75)
echo "at 0.75m"
bash -c "gnuradio-companion wifi_pluto_rx1_0_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_0_75.pcap >
results/wifi_pluto_rx1_RSSI_0_75.log"
;;
1)
echo "at 1m"
bash -c "gnuradio-companion wifi_pluto_rx1_1_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_1_0.pcap >
results/wifi_pluto_rx1_RSSI_1_0.log"
;;
```

```

1.25)
echo "at 1.25m"
bash -c "gnuradio-companion wifi_pluto_rx1_1_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_1_25.pcap >
results/wifi_pluto_rx1-RSSI-1_25.log"
;;
1.5)
echo "at 1.5m"
bash -c "gnuradio-companion wifi_pluto_rx1_1_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_1_5.pcap >
results/wifi_pluto_rx1-RSSI-1_5.log"
;;
1.75)
echo "at 1.75m"
bash -c "gnuradio-companion wifi_pluto_rx1_1_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_1_75.pcap >
results/wifi_pluto_rx1-RSSI-1_75.log"
;;
2)
echo "at 2m"
bash -c "gnuradio-companion wifi_pluto_rx1_2_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_2_0.pcap >
results/wifi_pluto_rx1-RSSI-2_0.log"
;;
2.25)
echo "at 2.25m"
bash -c "gnuradio-companion wifi_pluto_rx1_2_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_2_25.pcap >
results/wifi_pluto_rx1-RSSI-2_25.log"
;;
2.5)
echo "at 2.5m"
bash -c "gnuradio-companion wifi_pluto_rx1_2_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_2_5.pcap >
results/wifi_pluto_rx1-RSSI-2_5.log"
;;
2.75)
echo "at 2.75m"
bash -c "gnuradio-companion wifi_pluto_rx1_2_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_2_75.pcap >
results/wifi_pluto_rx1-RSSI-2_75.log"
;;
3)
echo "at 3m"
bash -c "gnuradio-companion wifi_pluto_rx1_3_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_3_0.pcap >
results/wifi_pluto_rx1-RSSI-3_0.log"
;;

```

```

3.25)
bash "at 3.25m"
bash -c "gnuradio-companion wifi_pluto_rx1_3_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_3_25.pcap >
results/wifi_pluto_rx1-RSSI-3_25.log"
;;
3.5)
echo "at 3.5m"
bash -c "gnuradio-companion wifi_pluto_rx1_3_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_3_5.pcap >
results/wifi_pluto_rx1-RSSI-3_5.log"
;;
3.75)
echo "at 3.75m"
bash -c "gnuradio-companion wifi_pluto_rx1_3_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_3_75.pcap >
results/wifi_pluto_rx1-RSSI-3_75.log"
;;
4)
echo "at 4m"
bash -c "gnuradio-companion wifi_pluto_rx1_4_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_4_0.pcap >
results/wifi_pluto_rx1-RSSI-4_0.log"
;;
4.25)
echo "at 4.25m"
bash -c "gnuradio-companion wifi_pluto_rx1_4_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_4_25.pcap >
results/wifi_pluto_rx1-RSSI-4_25.log"
;;
4.5)
echo "at 4.5m"
bash -c "gnuradio-companion wifi_pluto_rx1_4_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_4_5.pcap >
results/wifi_pluto_rx1-RSSI-4_5.log"
;;
4.75)
bash "at 4.75m"
bash -c "gnuradio-companion wifi_pluto_rx1_4_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_4_75.pcap >
results/wifi_pluto_rx1-RSSI-4_75.log"
;;
5)
echo "at 5m"
bash -c "gnuradio-companion wifi_pluto_rx1_5_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_5_0.pcap >
results/wifi_pluto_rx1-RSSI-5_0.log"
;;

```

```

esac
elif [[ $1 == R2 ]]; then
echo "Executing in GNU Radio R2 mode"
case "$2" in
0)
echo "executing R2 at generic mode"
echo "probably at 0 m"
echo "Find the results [instantaneous] at "
echo "For pcap : pcap/wifi_pluto_rx_general.pcap"
echo "For Results: results/wifi_pluto_rx_RSSI_general.log "
bash -c "gnuradio-companion wifi_pluto_rx_general.grc"
bash -c "tshark -r pcap/wifi_pluto_rx_general.pcap >
results/wifi_pluto_rx_RSSI_general.log"
;;
0.25)
echo "at 0.25m"
bash -c "gnuradio-companion wifi_pluto_rx2_0_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_0_25.pcap >
results/wifi_pluto_rx2_RSSI_0_25.log"
;;
0.5)
echo "at 0.5m"
bash -c "gnuradio-companion wifi_pluto_rx2_0_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_0_5.pcap >
results/wifi_pluto_rx2_RSSI_0_5.log"
;;
0.75)
echo "at 0.75m"
bash -c "gnuradio-companion wifi_pluto_rx2_0_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_0_75.pcap >
results/wifi_pluto_rx2_RSSI_0_75.log"
;;
1)
echo "at 1m"
bash -c "gnuradio-companion wifi_pluto_rx2_1_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_1_0.pcap >
results/wifi_pluto_rx2_RSSI_1_0.log"
;;
1.25)
echo "at 1.25m"
bash -c "gnuradio-companion wifi_pluto_rx2_1_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_1_25.pcap >
results/wifi_pluto_rx1_RSSI_1_25.log"
;;
1.5)
echo "at 1.5m"
bash -c "gnuradio-companion wifi_pluto_rx2_1_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_1_5.pcap >

```

```

results/wifi_pluto_rx2_RSSI_1_5.log"
;;
1.75)
echo "at 1.75m"
bash -c "gnuradio-companion wifi_pluto_rx2_1_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx1_2_75.pcap >
results/wifi_pluto_rx2_RSSI_1_75.log"
;;
2)
echo "at 2m"
bash -c "gnuradio-companion wifi_pluto_rx2_2_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_2_0.pcap >
results/wifi_pluto_rx2_RSSI_2_0.log"
;;
2.25)
echo "at 2.25m"
bash -c "gnuradio-companion wifi_pluto_rx2_2_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_2_25.pcap >
results/wifi_pluto_rx2_RSSI_2_25.log"
;;
2.5)
echo "at 2.5m"
bash -c "gnuradio-companion wifi_pluto_rx2_2_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_2_5.pcap >
results/wifi_pluto_rx2_RSSI_2_5.log"
;;
2.75)
echo "at 2.75m"
bash -c "gnuradio-companion wifi_pluto_rx2_2_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_2_75.pcap >
results/wifi_pluto_rx2_RSSI_2_75.log"
;;
3)
echo "at 3m"
bash -c "gnuradio-companion wifi_pluto_rx2_3_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_3_0.pcap >
results/wifi_pluto_rx2_RSSI_3_0.log"
;;
3.25)
echo "at 3.25m"
bash -c "gnuradio-companion wifi_pluto_rx2_3_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_3_25.pcap >
results/wifi_pluto_rx2_RSSI_3_25.log"
;;
3.5)
echo "at 3.5 m"
bash -c "gnuradio-companion wifi_pluto_rx2_3_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_3_5.pcap >

```



```

results/wifi_pluto_rx2_RSSI_3_5.log"
;;
3.75)
echo "at 3.75m"
bash -c "gnuradio-companion wifi_pluto_rx2_3_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_3_75.pcap >
results/wifi_pluto_rx2_RSSI_3_75.log"
;;
4)
echo "at 4m"
bash -c "gnuradio-companion wifi_pluto_rx2_4_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_4_0.pcap >
results/wifi_pluto_rx2_RSSI_4_0.log"
;;
4.25)
echo "at 4.25 m"
bash -c "gnuradio-companion wifi_pluto_rx2_4_25.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_4_25.pcap >
results/wifi_pluto_rx2_RSSI_4_25.log"
;;
4.5)
echo "at 4.5m"
bash -c "gnuradio-companion wifi_pluto_rx2_4_5.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_4_5.pcap >
results/wifi_pluto_rx2_RSSI_4_5.log"
;;
4.75)
echo "at 4.75m"
bash -c "gnuradio-companion wifi_pluto_rx2_4_75.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_4_75.pcap >
results/wifi_pluto_rx2_RSSI_4_75.log"
;;
5)
echo "at 5m"
bash -c "gnuradio-companion wifi_pluto_rx2_5_0.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_5_0.pcap >
results/wifi_pluto_rx2_RSSI_5_0.log"
;;
esac
elif [[ $1 == AR1 ]]; then
echo "Executing in GNU Radio R1, Aggressive Mode, Without Obstracle"
case "$2" in
1)
echo "At 1m aggressive"
bash -c "gnuradio-companion wifi_pluto_rx_at_1m_aggressive_
LOS.grc"
bash -c "tshark -r pcap/wifi_pluto_rx_at_1m_
aggressive_LOS.pcap >

```

```

    results/wifi_pluto_rx_at_1m_
    aggressive_LOS.log"
    ;;
esac
elif [[ $1 == AR2 ]]; then
    echo "Executing in GNU Radio R2, Aggressive Mode, Obstracle"
    case "$2" in
        1)
            echo "At 1m aggressive"
            bash -c "gnuradio-companion wifi_pluto_rx_at_1m_aggressive
            _OBS.grc"
            bash -c "tshark -r pcap/wifi_pluto_rx_at_1m_aggressive
            _OBS.pcap >
            results/wifi_pluto_rx_at_1m_aggressive
            _OBS.log"
            ;;
        esac
    elif [[ $1 == ADR1 ]]; then
        echo "Execting in Dynamic Mode for R1 receiver"
        case "$2" in
            1)
                echo "Dynamic Mode Aggressive at 1m for R1 receiver"
                bash -c "gnuradio-companion wifi_pluto_rx1_at_1m_aggressive
                _dynamic.grc"
                bash -c "tshark -r pcap/wifi_pluto_rx1_at_1m_aggressive
                _dynamic.pcap >
                results/wifi_pluto_rx1_at_1m_aggressive
                _dynamic.log"
                ;;
            esac
        case "$2" in
            DM)
                echo "Dynamic Mode Aggressive at All distances for R1 receiver"
                bash -c "gnuradio-companion wifi_pluto_rx1_at_all_aggressiv
                e_dynamic.grc"
                bash -c "tshark -r pcap/wifi_pluto_rx1_at_all_aggressive
                _dynamic.pcap >
                results/wifi_pluto_rx1_at_all_aggressive
                _dynamic.log"
                ;;
            esac
        elif [[ $1 == ADR2 ]]; then
            echo "Execting in Dynamic Mode for R2 receiver"
            case "$2" in
                1)
                    echo "Dynamic Mode Aggressive at 2m for R2 receiver"
                    bash -c "gnuradio-companion wifi_pluto_rx2_at_1m_aggressive
                    _dynamic.grc"

```

```

bash -c "tshark -r pcap/wifi_pluto_rx2_at_1m_aggressive
_dynamic.pcap >
results/wifi_pluto_rx2_at_1m_aggressive
_dynamic.log"
;;
esac
case "$2" in
DM)
echo "Dynamic Mode Aggressive at All distances for R2 receiver"
bash -c "gnuradio-companion wifi_pluto_rx2_at_all_aggressive
_dynamic.grc"
bash -c "tshark -r pcap/wifi_pluto_rx2_at_all_aggressive
_dynamic.pcap >
results/wifi_pluto_rx2_at_all_aggressive
_dynamic.log"
;;
esac
else
echo "No input matched"
echo "executing R at generic mode probably at 0 m"
bash -c "gnuradio-companion wifi_pluto_rx_general.grc"
bash -c "tshark -r pcap/wifi_pluto_rx_general.pcap >
results/wifi_pluto_rx_RSSI_general.log"
fi
"-----End of main_output_newfiles.sh-----"

```

APPENDIX B3

IEEE802.11 a/g/p

Gnu Radio Python Script

```
"""-----Name as wifi_pluto_rx_general.py-----"""
#!/usr/bin/env python2
# -*- coding: utf-8 -*-
#####
# GNU Radio Python Flow Graph
# Title: Wifi Rx
# Generated: Sat Mar 16 17:09:46 2019
#####

if __name__ == '__main__':
    import ctypes
    import sys
    if sys.platform.startswith('linux'):
        try:
            x11 = ctypes.cdll.LoadLibrary('libX11.so')
            x11.XInitThreads()
        except:
            print "Warning: failed to XInitThreads()"

from PyQt4 import Qt
from PyQt4.QtCore import QObject, pyqtSlot
from gnuradio import blocks
from gnuradio import eng_notation
from gnuradio import fft
from gnuradio import gr
from gnuradio import iio
from gnuradio import qtgui
from gnuradio.eng_option import eng_option
from gnuradio.fft import window
from gnuradio.filter import firdec
from gnuradio.qtgui import Range, RangeWidget
from optparse import OptionParser
import foo
import ieee802_11
import sip
import sys

class wifi_rx(gr.top_block, Qt.QWidget):

    def __init__(self):
        gr.top_block.__init__(self, "Wifi Rx")
```

```

Qt.QWidget.__init__(self)
self.setWindowTitle("Wifi Rx")
try:
self.setWindowIcon(Qt.QIcon.fromTheme('gnuradio-grc'))
except:
    pass
self.top_scroll_layout = Qt.QVBoxLayout()
self.setLayout(self.top_scroll_layout)
self.top_scroll = Qt.QScrollArea()
self.top_scroll.setFrameStyle(Qt.QFrame.NoFrame)
self.top_scroll_layout.addWidget(self.top_scroll)
self.top_scroll.setWidgetResizable(True)
self.top_widget = Qt.QWidget()
self.top_scroll.setWidget(self.top_widget)
self.top_layout = Qt.QVBoxLayout(self.top_widget)
self.top_grid_layout = Qt.QGridLayout()
self.top_layout.addLayout(self.top_grid_layout)

self.settings = Qt.QSettings("GNU Radio", "wifi_rx")
self.restoreGeometry(self.settings.value("geometry").
toByteArray())

#####
# Variables
#####
self.window_size = window_size = 48
self.sync_length = sync_length = 320
self.samp_rate = samp_rate = 10e6
self.lo_offset = lo_offset = 0
self.gain = gain = .75
self.freq = freq = 3500000000
self.chan_est = chan_est = 0

#####
# Blocks
#####
self._samp_rate_options = [5e6, 10e6, 20e6]
self._samp_rate_labels = ["5 MHz", "10 MHz", "20 MHz"]
self._samp_rate_tool_bar = Qt.QToolBar(self)
self._samp_rate_tool_bar.addWidget(
Qt.QLabel("samp_rate"+"": ))
self._samp_rate_combo_box = Qt.QComboBox()
self._samp_rate_tool_bar.addWidget
(self._samp_rate_combo_box)
for label in self._samp_rate_labels:
self._samp_rate_combo_box.addItem(label)
self._samp_rate_callback = lambda i:
Qt.QMetaObject.invokeMethod(

```

```

self._samp_rate_combo_box, "setCurrentIndex", Qt.Q_ARG
("int", self._samp_rate_options.index(i)))
self._samp_rate_callback(self.samp_rate)
self._samp_rate_combo_box.currentIndexChanged.connect(
    lambda i: self.
set_samp_rate(self._samp_rate_options[i]))
self.top_layout.addWidget(self._samp_rate_tool_bar)
self._freq_options = [2412000000.0, 2417000000.0,
2422000000.0, 2427000000.0,
2432000000.0, 2437000000.0, 2442000000.0,
2447000000.0, 2452000000.0, 2457000000.0,
2462000000.0, 2467000000.0, 2472000000.0,
2484000000.0, 5170000000.0, 5180000000.0,
5190000000.0, 5200000000.0, 5210000000.0,
5220000000.0, 5230000000.0, 5240000000.0,
5250000000.0, 5260000000.0, 5270000000.0,
5280000000.0, 5290000000.0, 5300000000.0,
5310000000.0, 5320000000.0, 5500000000.0,
5510000000.0, 5520000000.0, 5530000000.0,
5540000000.0, 5550000000.0, 5560000000.0,
5570000000.0, 5580000000.0, 5590000000.0,
5600000000.0, 5610000000.0, 5620000000.0,
5630000000.0, 5640000000.0, 5660000000.0,
5670000000.0, 5680000000.0, 5690000000.0,
5700000000.0, 5710000000.0, 5720000000.0,
5745000000.0, 5755000000.0, 5765000000.0,
5775000000.0, 5785000000.0, 5795000000.0,
5805000000.0, 5825000000.0, 5860000000.0,
5870000000.0, 5880000000.0, 5890000000.0,
5900000000.0, 5910000000.0, 5920000000.0,
3500000000.0]
self._freq_labels = [' 1 | 2412.0 | 11g', ' 2 |
2417.0 | 11g', ' 3 | 2422.0 | 11g', ' 4 |
2427.0 | 11g', ' 5 | 2432.0 | 11g', ' 6 | 2437.0 | 11g', '
2442.0 | 11g', ' 8 | 2447.0 | 11g', ' 9 |
2452.0 | 11g', ' 10 | 2457.0 | 11g', ' 11 |
2462.0 | 11g', ' 12 | 2467.0 | 11g', ' 13 |
2472.0 | 11g', ' 14 | 2484.0 | 11g', ' 34 |
5170.0 | 11a', ' 36 | 5180.0 | 11a', ' 38 |
5190.0 | 11a', ' 40 | 5200.0 | 11a', ' 42 |
5210.0 | 11a', ' 44 | 5220.0 | 11a', ' 46 | 5230.0 | 11a', ' 48 |
5240.0 | 11a', ' 50 | 5250.0 | 11a', ' 52 |
5260.0 | 11a', ' 54 | 5270.0 | 11a', ' 56 |
5280.0 | 11a', ' 58 | 5290.0 | 11a', ' 60 | 5300.0 | 11a', ' 62 |
5310.0 | 11a', ' 64 | 5320.0 | 11a', '100 | 5500.0 | 11a', '102 |
5510.0 | 11a', '104 | 5520.0 | 11a', '106 |
5530.0 | 11a', '108 | 5540.0 | 11a', '110 |

```

5550.0		11a', '112		5560.0		11a', '114	
5570.0		11a', '116		5580.0		11a', '118	
5590.0		11a', '120					
5600.0		11a', '122		5610.0		11a', '124	
5620.0		11a', '126		5630.0		11a', '128	
5640.0		11a', '132		5660.0		11a', '134	
5670.0		11a', '136		5680.0		11a', '138	
5690.0		11a', '140		5700.0		11a', '142	
5710.0		11a', '144		5720.0		11a', '149	
5745.0		11a (SRD)', '151		5755.0			
		11a (SRD)', '153					
5765.0		11a (SRD)', '155		5775.0			
		11a (SRD)', '157					
5785.0		11a (SRD)', '159		5795.0			
		11a (SRD)', '161					
5805.0		11a (SRD)', '165		5825.0			
		11a (SRD)', '172					
5860.0		11p', '174		5870.0		11p', '176	
5880.0		11p', '178		5890.0		11p', '180	
5900.0		11p', '182		5910.0		11p', '184	
5920.0		11p', '185		3.5G'			

```

self._freq_tool_bar = Qt.QToolBar(self)
self._freq_tool_bar.addWidget(Qt.QLabel("freq"+"": ))
self._freq_combo_box = Qt.QComboBox()
self._freq_tool_bar
.add_widget(self._freq_combo_box)
for label in self._freq_labels: self._freq_combo_box
.add_item(label)
self._freq_callback = lambda i: Qt.QMetaObject.
invokeMethod(self._freq_combo_box, "setCurrentIndex",
Qt.Q_ARG("int", self._freq_options.index(i)))
self._freq_callback(self.freq)
self._freq_combo_box.currentIndexChanged.connect(
lambda i: self.set_freq(self._freq_options[i]))
self.top_layout.addWidget(self._freq_tool_bar)
self._chan_est_options = [ieee802_11.LS, ieee802_11.LMS,
ieee802_11.STA, ieee802_11.COMB]
self._chan_est_labels = ["LS", "LMS", "STA",
"Linear Comb"]
self._chan_est_group_box = Qt.QGroupBox("chan_est")
self._chan_est_box = Qt.QHBoxLayout()
class variable_chooser_button_group(Qt.QButtonGroup):
def __init__(self, parent=None):
Qt.QButtonGroup.__init__(self, parent)
@pyqtSlot(int)
def updateButtonChecked(self, button_id):

```

```

        self.button(button_id).setChecked(True)
self._chan_est_button_group = variable_chooser
_button_group()
self._chan_est_group_box.setLayout(self._chan_est_box)

for i, label in enumerate(self._chan_est_labels):
    radio_button = Qt.QRadioButton(label)
    self._chan_est_box.addWidget(radio_button)
    self._chan_est_button_group.addButton(radio_button, i)
self._chan_est_callback = lambda i: Qt.QMetaObject.
invokeMethod(self._chan_est_button_group, "
updateButtonChecked", Qt.Q_ARG("int",
self._chan_est_options.index(i)))
self._chan_est_callback(
self.chan_est)
self._chan_est_button_group.
buttonClicked[int].connect(
lambda i: self.set_chan_est(self.
_chan_est_options[i]))
self.top_layout.addWidget(self._chan_est_group_box)
self.qtgui_time_sink_x_0 = qtgui.time_sink_f(
    1024, #size
    samp_rate, #samp_rate
    "", #name
    1 #number of inputs
)
self.qtgui_time_sink_x_0.set_update_time(0.10)
self.qtgui_time_sink_x_0.set_y_axis(-1, 1)

self.qtgui_time_sink_x_0.set_y_label("Amplitude", "")

self.qtgui_time_sink_x_0.enable_tags(-1, True)
self.qtgui_time_sink_x_0.set_trigger_mode(qtgui.
TRIG_MODE_FREE, qtgui.TRIG_SLOPE_POS, 0.0, 0, 0, "")
self.qtgui_time_sink_x_0.enable_autoscale(True)
self.qtgui_time_sink_x_0.enable_grid(True)
self.qtgui_time_sink_x_0.enable_control_panel(False)

if not True:
    self.qtgui_time_sink_x_0.disable_legend()

labels = ["", "", "", "", "",
          "", "", "", "", "", ""]
widths = [1, 1, 1, 1, 1,
          1, 1, 1, 1, 1]
colors = ["blue", "red", "green", "black", "cyan",
          "magenta", "yellow",
          "dark red", "dark green", "blue"]

```



```

styles = [1, 1, 1, 1, 1,
          1, 1, 1, 1, 1]
markers = [-1, -1, -1, -1, -1,
           -1, -1, -1, -1, -1]
alphas = [1.0, 1.0, 1.0, 1.0, 1.0,
          1.0, 1.0, 1.0, 1.0, 1.0]

for i in xrange(1):
    if len(labels[i]) == 0:
        self.qtgui_time_sink_x_0.set_line_label(i,
            "Data {0}".format(i))
    else:
        self.
            qtgui_time_sink_x_0.set_line_label(i, labels[i])
        self.qtgui_time_sink_x_0.set_line_width(i, widths[i])
        self.qtgui_time_sink_x_0.set_line_color(i, colors[i])
        self.qtgui_time_sink_x_0.set_line_style(i, styles[i])
        self.qtgui_time_sink_x_0.set_line_marker(i, markers[i])
        self.qtgui_time_sink_x_0.set_line_alpha(i, alphas[i])

self._qtgui_time_sink_x_0_win = sip.wrapinstance(self.
qtgui_time_sink_x_0.pyqwidget(), Qt.QWidget)
self.top_layout.addWidget(self._qtgui_time_sink_x_0_win)
self.qtgui_const_sink_x_0 = qtgui.const_sink_c(
    48*10, #size
    "", #name
    1 #number of inputs
)
self.qtgui_const_sink_x_0.set_update_time(0.10)
self.qtgui_const_sink_x_0.set_y_axis(-2, 2)
self.qtgui_const_sink_x_0.set_x_axis(-2, 2)
self.qtgui_const_sink_x_0.set_trigger_mode(qtgui
.TRIG_MODE_FREE, qtgui.TRIG_SLOPE_POS, 0.0, 0, "")
self.qtgui_const_sink_x_0.enable_autoscale(True)
self.qtgui_const_sink_x_0.enable_grid(True)

if not True:
    self.qtgui_const_sink_x_0.disable_legend()

labels = ["", "", "", "", "",
          "", "", "", "", ""]
widths = [1, 1, 1, 1, 1,
          1, 1, 1, 1, 1]
colors = ["blue", "red", "red", "red", "red",
          "red", "red", "red", "red", "red"]
styles = [0, 0, 0, 0, 0,
          0, 0, 0, 0, 0]
markers = [1, 0, 0, 0, 0,

```

```

        0, 0, 0, 0, 0]
alphas = [1.0, 1.0, 1.0, 1.0, 1.0,
          1.0, 1.0, 1.0, 1.0, 1.0]
for i in xrange(1):
    if len(labels[i]) == 0:
self.qtdgui_const_sink_x_0.set_line_label(i,
    "Data {0}".format(i))
    else:
self.qtdgui_const_sink_x_0.set_line_label(i,
labels[i])
    self.qtdgui_const_sink_x_0.set_line_width(i
    , widths[i])
    self.qtdgui_const_sink_x_0.set_line_color(i
    , colors[i])
    self.qtdgui_const_sink_x_0.set_line_style(i
    , styles[i])
    self.qtdgui_const_sink_x_0.set_line_marker(i
    , markers[i])
    self.qtdgui_const_sink_x_0.set_line_alpha(i
    , alphas[i])

self._qtdgui_const_sink_x_0_win = sip.wrapinstance(self.
qtdgui_const_sink_x_0.pyqwidget(), Qt.QWidget)
self.top_layout.addWidget(self._qtdgui_const_sink_x_0_win)
self.pluto_source_0 =
iio.pluto_source("",
int(freq), int(10000000), int(20000000), 0x8000, True,
True,
True, "manual", 64.0, "", True)
self._lo_offset_options =
(0, 6e6, 11e6, )
self._lo_offset_labels =
(str(self._lo_offset_options[0]),
str(self._lo_offset_options[1]),

str(self._lo_offset_options[2]), )
self._lo_offset_tool_bar =

Qt.QToolBar(self)
self._lo_offset_tool_bar.addWidget(Qt.QLabel("
lo_offset"+"": ))
self.
_lo_offset_combo_box =

Qt.QComboBox()
self._lo_offset_tool_bar.addWidget(self.
_lo_offset_combo_box)
for label in self._lo_offset_labels:

```

```

self._lo_offset_combo_box.addItem(label)

self._lo_offset_callback =

lambda i: Qt
.QMetaObject.invokeMethod(self
._lo_offset_combo_box, "
setCurrentIndex", Qt.
Q_ARG("int", self._lo_offset_options.index(i)))

self._lo_offset_callback
(self.lo_offset)
self.
_lo_offset_combo_box
.
currentIndexChanged.connect(
    lambda i: self.set_lo_offset(self.

        _lo_offset_options[i]))
self.top_layout.addWidget(self.

_lo_offset_tool_bar)
self.ieee802_11_sync_short_0 =
ieee802_11.sync_short(0.56, 2, False, False)
self.ieee802_11_sync_long_0 =
ieee802_11.sync_long(sync_length, True, False)
self.ieee802_11_parse_mac_0
=
ieee802_11.parse_mac(True, True)
self.ieee802_11_moving_average_xx_1
=
ieee802_11.moving_average_ff(window_size + 16)
self.ieee802_11_moving_average_xx_0
=
ieee802_11.moving_average_cc(window_size)
self.ieee802_11_frame_equalizer_0 =
ieee802_11.frame_equalizer(chan_est,
freq, samp_rate, False, False)
self.ieee802_11_decode_mac_0 =
ieee802_11.decode_mac(True, False)
self._gain_range =
Range(0, 1, 0.01, .75, 200)
self._gain_win =
RangeWidget(self._gain_range, self.set_gain, "gain",
"counter_slider", float)
self.top_layout.addWidget(self._gain_win)
self.foo_wireshark_connector_0 =

```

```

foo.wireshark_connector(127, True)
self.fft_vxx_0 =
fft.fft_vcc(64, True, (window.rectangular(64)), True, 1)
self.blocks_stream_to_vector_0 =
blocks.stream_to_vector(gr.sizeof_gr_complex*1, 64)
self.blocks_pdu_to_tagged_stream_1 =
blocks.pdu_to_tagged_stream(blocks.complex_t, "packet_len")
self.blocks_multiply_xx_0 =
blocks.multiply_vcc(1)
self.blocks_file_sink_0
= blocks.file_sink(gr.sizeof_char*1,
"/home/praveen-ubuntu-radio/Documents/Gnu-radio/
Gnu-radio-programs/ieee802_11-Radio-Localization/Original/
pcap/wifi-pluto-rx2-at-all-aggressive-dynamic.pcap", True)
self.blocks_file_sink_0.set_unbuffered(True)
self.blocks_divide_xx_0
= blocks.divide_ff(1)
self.blocks_delay_0_0
= blocks.delay(gr.sizeof_gr_complex*1, 16)
self.blocks_delay_0
= blocks.delay(gr.sizeof_gr_complex*1, sync_length)
self.blocks_conjugate_cc_0
= blocks.conjugate_cc()
self.blocks_complex_to_mag_squared_0
= blocks.complex_to_mag_squared(1)
self.blocks_complex_to_mag_0
= blocks.complex_to_mag(1)

#####
# Connections
#####
self.msg_connect((self.ieee802_11_decode_mac_0, 'out'),
(self.foo_wireshark_connector_0, 'in'))
self.msg_connect((self.ieee802_11_decode_mac_0, 'out'),
(self.ieee802_11_parse_mac_0, 'in'))
self.msg_connect((self.ieee802_11_frame_equalizer_0, 'symbols'),
(self.blocks_pdu_to_tagged_stream_1, 'pdus'))
self.connect((self.blocks_complex_to_mag_0, 0),
(self.blocks_divide_xx_0, 0))
self.connect((self.blocks_complex_to_mag_squared_0, 0),
(self.ieee802_11_moving_average_xx_1, 0))
self.connect((self.blocks_conjugate_cc_0, 0),
(self.blocks_multiply_xx_0, 1))
self.connect((self.blocks_delay_0, 0),
(self.ieee802_11_sync_long_0, 1))
self.connect((self.blocks_delay_0_0, 0),
(self.blocks_conjugate_cc_0, 0))
self.connect((self.blocks_delay_0_0, 0),

```

```

(self.ieee802_11_sync_short_0 , 0))
self.connect((self.blocks_divide_xx_0 , 0),
(self.ieee802_11_sync_short_0 , 2))
self.connect((self.blocks_divide_xx_0 , 0),
(self.qtgui_time_sink_x_0 , 0))
self.connect((self.blocks_multiply_xx_0 , 0),
(self.ieee802_11_moving_average_xx_0 , 0))
self.connect((self.blocks_pdu_to_tagged_stream_1 , 0),
(self.qtgui_const_sink_x_0 , 0))
self.connect((self.blocks_stream_to_vector_0 , 0),
(self.fft_vxx_0 , 0))
self.connect((self.fft_vxx_0 , 0),
(self.ieee802_11_frame_equalizer_0 , 0))
self.connect((self.foo_wireshark_connector_0 , 0),
(self.blocks_file_sink_0 , 0))
self.connect((self.ieee802_11_frame_equalizer_0 , 0),
(self.ieee802_11_decode_mac_0 , 0))
self.connect((self.ieee802_11_moving_average_xx_0 , 0),
(self.blocks_complex_to_mag_0 , 0))
self.connect((self.ieee802_11_moving_average_xx_0 , 0),
(self.ieee802_11_sync_short_0 , 1))
self.connect((self.ieee802_11_moving_average_xx_1 , 0),
(self.blocks_divide_xx_0 , 1))
self.connect((self.ieee802_11_sync_long_0 , 0),
(self.blocks_stream_to_vector_0 , 0))
self.connect((self.ieee802_11_sync_short_0 , 0),
(self.blocks_delay_0 , 0))
self.connect((self.ieee802_11_sync_short_0 , 0),
(self.ieee802_11_sync_long_0 , 0))
self.connect((self.pluto_source_0 , 0),
(self.blocks_complex_to_mag_squared_0 , 0))
self.connect((self.pluto_source_0 , 0),
(self.blocks_delay_0_0 , 0))
self.connect((self.pluto_source_0 , 0),
(self.blocks_multiply_xx_0 , 0))

def closeEvent(self , event):
    self.settings = Qt.QSettings("GNU Radio", "wifi_rx")
    self.settings.setValue("geometry", self.saveGeometry())
    event.accept()

def get_window_size(self):
    return self.window_size

def set_window_size(self , window_size):
    self.window_size = window_size
    self.ieee802_11_moving_average_xx_0.set_length(

```

```

        self.window_size)
        self.ieee802_11_moving_average_xx_1.set_length(
            self.window_size + 16)

    def get_sync_length(self):
        return self.sync_length

    def set_sync_length(self, sync_length):
        self.sync_length = sync_length
        self.blocks_delay_0.set_dly(self.sync_length)

    def get_samp_rate(self):
        return self.samp_rate

    def set_samp_rate(self, samp_rate):
        self.samp_rate = samp_rate
        self._samp_rate_callback(self.samp_rate)
        self.ieee802_11_frame_equalizer_0.set_bandwidth(
            self.samp_rate)
        self.qtgui_time_sink_x_0.set_samp_rate(self.samp_rate)

    def get_lo_offset(self):
        return self.lo_offset

    def set_lo_offset(self, lo_offset):
        self.lo_offset = lo_offset
        self._lo_offset_callback(
            self.lo_offset)

    def get_gain(self):
        return self.gain

    def set_gain(self, gain):
        self.gain = gain

    def get_freq(self):
        return self.freq

    def set_freq(self, freq):
        self.freq = freq
        self._freq_callback(
            self.freq)
        self.ieee802_11_frame_equalizer_0.set_frequency(self
            .freq)
        self.pluto_source_0.set_params(int
            (self.freq),

```

```

        int(10000000), int(20000000), True, True, True,
        "manual", 64.0, "", True)

def get_chan_est(self):
    return self.
    chan_est

def set_chan_est(self, chan_est):
    self.chan_est = chan_est
    self._chan_est_callback(self.chan_est)
    self.ieee802_11_frame_equalizer_0.
    set_algorithm(self.chan_est)

def main(top_block_cls=wifi_rx, options=None):

    from distutils.version import StrictVersion
    if StrictVersion(Qt.qVersion()) >= StrictVersion("4.5.0"):
        style = gr.prefs().get_string('qtgui', 'style', 'raster')
        Qt.QApplication.setGraphicsSystem(style)
    qapp = Qt.QApplication(sys.argv)

    tb = top_block_cls()
    tb.start()
    tb.show()

    def quitting():
        tb.stop()
        tb.wait()
    qapp.connect(qapp, Qt.SIGNAL("aboutToQuit()"), quitting)
    qapp.exec_()

if __name__ == '__main__':
    main()
"----- End of wifi_pluto_rx_general.py-----"

```

Note: All the scripts in **Appendix B1** to **Appendix B3** are customized exclusively for this Thesis and are mostly interdependent on each other. So, it is advisable to have all the scripts when executing the other.

— End of Programming Scripts —

WORKING IMAGES

Figure A.1: Working Images

In Figure A.1: Figure A.1 shows working images at various stages of experimentation which are represented as images of 2X3 matrix.

- Figure A.1(1, 1): WAP based real-time RSSI and localization dashboard.
- Figure A.1(1, 2): IEEE 802.11 a/g/p based localization when obstacle is between Rx & Tx.
- Figure A.1(1, 3): IEEE 802.11 a/g/p based localization when Tx and Rx are in LOS.
- Figure A.1(2, 1): Experimentation set up with Raspberry Pi, SDR Rx and Laptop.
- Figure A.1(2, 2): IEEE 802.11 a/g/p based localization when Tx and Rx are in diagonal.
- Figure A.1(2, 3): IEEE 802.11 a/g/p based localization when Tx and Rx are in high clutter region.

Figure A.2: Working Images

In Figure A.2: Figure A.2 shows working images at various stages of experimentation which are represented as images of 2X3 matrix.

- Figure A.2(1, 1): IEEE 802.11 a/g/p based localization when obstacle is between Rx & Tx.
- Figure A.2(1, 2): IEEE 802.11 a/g/p localization when Tx and Rx are at near LOS.
- Figure A.2(1, 3): IEEE 802.11 a/g/p localization when Tx and Rx are at high clutter diagonal.
- Figure A.2(2, 1): IEEE 802.11 a/g/p localization shows Tx at high clutter region.
- Figure A.2(2, 2): IEEE 802.11 a/g/p localization when Tx and Rx in LOS.
- Figure A.2(2, 3): Screen Showing the software setup and RSSI reception from Tx.

APPENDIX D1

NOTABLE PEOPLE

Dr. Teerapat Sanguankotchakorn

(D.Eng, Tokyo Institute of Technology, Japan.)
Associate Professor & Head, Telecommunication Engineering
Asian Institute of Technology, Thailand
<https://set.ait.ac.th/faculty/teerapat-sanguankotchakorn/>

Dr. Poompat Saengudomlert

(Ph.D, Massachusetts Institute of Technology(MIT), USA)
Adjunct Professor, Telecommunication Engineering
Asian Institute of Technology, Thailand
<https://set.ait.ac.th/faculty/poompat-saengudomlert/>

Dr. Attaphongse Taparugssanagorn

(D.Tech, University of Oulu, Finland)
Associate Professor, Telecommunication Engineering
Asian Institute of Technology, Thailand
<https://set.ait.ac.th/faculty/attaphongse-taparugssanagorn/>

Ms. Premma Rao

Administrative Secretary, Telecommunication Engineering
Asian Institute of Technology, Thailand
<https://set.ait.ac.th/staff/>

Mr. Rajesh Kumar Dehury

Technical Lab Supervisor, Telecommunication Engineering
Asian Institute of Technology, Thailand
http://intranet.ait.ac.th/advsearch/Employee_search

Dr. L. Pratap Reddy

(Ph.D, Jawaharlal Nehru Technological University Hyderabad, India)
Professor, Electronics and Communication Engineering
JNT University Hyderabad (JNTUH) - College of Engineering Hyderabad, India
http://jntuhceh.ac.in/faculty_details/4/dept/337