A Dissertation

entitled

Cooperative Localization in Cellular Networks

by

Thabet Mismar

Submitted to the Graduate Faculty as partial fulfillment of the requirements for the
Doctor of Philosophy Degree in Engineering

Dr. Junghwan Kim, Committee Chair

Dr. Mansoor Alam, Committee Member

Dr. Mohsin Jamali, Committee Member

Dr. Ezzatollah Salari, Committee Member

Dr. Dong-Shik Kim, Committee Member

Dr. Patricia R. Komuniecki, Dean
College of Graduate Studies

The University of Toledo

May 2015
An Abstract of

Cooperative Localization in Cellular Networks

by

Thabet Mismar

Submitted to the Graduate Faculty As partial fulfillment of the requirements for the Doctor of Philosophy Degree in Engineering

The University of Toledo
May 2015

Since its introduction to the public in 1994, the Global Positioning System (GPS) has been used in every aspect of our lives. While GPS works just fine most of the times, such as navigating an unknown city using a portable GPS unit or soldiers using it in the battle field, scenarios arise where GPS fails as a reliable positioning and localization system; like inside buildings and tunnels.

Cooperative localization using cellular phones would solve this problem since there is a much higher number of information sources (other surrounding cellular users) that are much closer to the user, whereas the GPS system only has a limited number of information sources which are thousands of kilometers away.

Spoofing, although not popular in GPS, is a topic of a few research papers. With the introduction of cooperative localization, spoofing will be easier to implement and harder to defend against. Not much research has been done to study the effects of spoofing attacks on cooperative localization networks.
In this dissertation, it is shown that reliable positioning and localization is possible using cooperative localization through cellular networks. In addition, it proves that processing sufficient information from a number of information transmitting users can mitigate spoofing attacks on any network. This can be done by filtering out the extreme location estimations resulting from the spoofing attacks.

Since Kalman filter helps in predicting the movement of mobile users, using Kalman filter for mobile network users significantly helped reduce the error in estimating the user’s location.
For my parents, my wife, and my two boys. Thank you for your support and encouragement through the years working on this dissertation.
Acknowledgments

The guidance and mentoring of my advisor Prof. Kim made this dissertation possible. I would like to thank the committee members for the confidence and support from the start. A special appreciation to Dr. Roshdi Khalil for his guidance in the mathematical part of this work.
# Table of Contents

Abstract ............................................................................................................................... iii

Acknowledgments ............................................................................................................... vi

Table of Contents .............................................................................................................. vii

List of Tables ................................................................................................................... ix

List of Figures .................................................................................................................. xi

List of Abbreviations ...................................................................................................... xviii

1 Introduction .................................................................................................................... 1

1.1. Motivation for Research .......................................................................................... 3

1.2. Related and Previous Works ................................................................................... 4

1.3. Research Objective .................................................................................................. 8

1.4. Dissertation Organization ....................................................................................... 9

2 Theoretical Background ............................................................................................... 11

2.1. Mathematical Background ..................................................................................... 11

2.2. Signal Propagation Models ..................................................................................... 16

2.3. Mobile Users and Kalman Filtering ......................................................................... 18

2.4. Barometric Equations ............................................................................................. 21
2.5. Spoofing .......................................................................................................................... 23
2.6. Minimum Mean Square Error (MMSE) .............................................................................. 25

3 Environment and Modeling .................................................................................................... 28
3.1. User Environment .............................................................................................................. 28
3.1.1. Two Dimensional (2D) Localization ............................................................................. 29
3.1.2. Three Dimensional (3D) Localization .......................................................................... 30
3.2. User Mobility .................................................................................................................... 30
3.3. System Model .................................................................................................................... 31

4 Simulation and Results .......................................................................................................... 38
4.1. Case-by-case Results ........................................................................................................ 38
4.1.1. The Indoor and Outdoor LOS Case .............................................................................. 40
4.1.2. The Indoor NLOS Case ............................................................................................... 51
4.1.3. The Outdoor NLOS Case ............................................................................................ 60
4.2. General Results ................................................................................................................ 67

5 Conclusion and Future Work .................................................................................................. 72

References .................................................................................................................................. 74

A The Derivation of \( f_{\mathbf{e}_{\mathbf{x}_0}}(\mathbf{e}_{\mathbf{x}_0}) \) ................................................................................. 78

B The Matlab Program Code .................................................................................................... 96
## List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>The average percentage of processed NLOS signals received by the target user</td>
<td>19</td>
</tr>
<tr>
<td>3.1</td>
<td>Simulation Environment Parameters</td>
<td>28</td>
</tr>
<tr>
<td>4.1</td>
<td>The Simulation Cases</td>
<td>39</td>
</tr>
<tr>
<td>4.2</td>
<td>The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 2D LOS Environment</td>
<td>46</td>
</tr>
<tr>
<td>4.3</td>
<td>The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 3D LOS Environment</td>
<td>50</td>
</tr>
<tr>
<td>4.4</td>
<td>The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 2D Indoor NLOS Environment</td>
<td>54</td>
</tr>
<tr>
<td>4.5</td>
<td>The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 3D Indoor NLOS Environment</td>
<td>59</td>
</tr>
<tr>
<td>4.6</td>
<td>The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 2D Outdoor NLOS Environment</td>
<td>63</td>
</tr>
<tr>
<td>4.7</td>
<td>The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 3D Outdoor NLOS Environment</td>
<td>66</td>
</tr>
</tbody>
</table>
4.8 The percentage the localization method returns with no results..........................67

4.9 The average run-time for the cooperative localization method to return with a location estimation..........................................................68
List of Figures

1-1 The triangulation method uses the distance from three users to determine the location of the forth user \((x_0, y_0)\). .................................................................2

2-1 The probability density function of \(e_{x_0}\) ..................................................................................15

2-2 The Effect of the Spoofing attack on the Target User .................................................................25

2-3 The Main Algorithm of the MMSE Location Estimation Method .............................................26

3-1 Relationship between the input variance of the error in the users’ location and the output average location estimation error ..................................................................................29

3-2 The main algorithm used to estimate the target user’s true location ..................................32

3-3 Signal Simulation; where users’ locations are used to simulate the received signal at the target user ..................................................................................................................32

3-4 The process of seperating and filtering each received signal ..............................................33

3-5 The Costas Recovery Circuit ........................................................................................................34

3-6 Location estimation based on RSS .............................................................................................36

4-1 The Error in Location Estimation Resulting from Different User Densities in a 2D LOS environment .................................................................................................................40
4-2 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D LOS environment for a Density of 500 users/ km$^2$. .................................................................41

4-3 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D LOS Environment for a Density of 500 users/ km$^2$. ..........42

4-4 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D LOS Environment for a Density of 2000 users/ km$^2$. .........................................................................................42

4-5 The Error in Location Estimation Resulting from Different Attack Errors from Two Spoofing Users in a 2D LOS Environment for a Density of 2000 users/ km$^2$. 43

4-6 The Error in Location Estimation Resulting from Different User Densities in a 2D LOS Environment with a Kalman Filter. .................................................................44

4-7 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D LOS Environment with a Kalman Filter. .................................................................45

4-8 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D LOS Environment with a Kalman Filter. .........................45

4-9 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D LOS Environment. .........................47
4-10 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D LOS Environment. ..........................................................48

4-11 The Error in Location Estimation Resulting from Different User Densities in a 3D LOS Environment with a Kalman Filter. .................................................................49

4-12 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D LOS Environment with a Kalman Filter. .................................................................49

4-13 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D LOS Environment with Kalman Filter. ..............................50

4-14 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Indoor Multipath and NLOS Environment. .................................................................51

4-15 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D Indoor Multipath and NLOS Environment. .........................52

4-16 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Indoor Multipath and NLOS Environment with a Kalman Filter. .................................................................53

4-17 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D Indoor Multipath and NLOS Environment with a Kalman Filter. .................................................................................53
4-18 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 2000 users/ km². .................................................................55

4-19 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 2000 users/ km². ........................................................................................................55

4-20 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 8000 users/ km². ............................................................................56

4-21 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 8000 users/ km². ........................................................................................................56

4-22 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 2000 users/ km². ..................57

4-23 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 2000 users/ km². .................................................................58

4-24 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 8000 users/ km². ......................58
4-25 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 8000 users/km$^2$. .........................................................59

4-26 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Outdoor Multipath and NLOS Environment. .................................................................60

4-27 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D Outdoor Multipath and NLOS Environment. .........................61

4-28 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Outdoor Multipath and NLOS Environment with a Kalman Filter. .............................................................62

4-29 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D Outdoor Multipath and NLOS Environment with a Kalman Filter. .............................................................62

4-30 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Outdoor Multipath and NLOS Environment. .................................................................64

4-31 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Outdoor Multipath and NLOS Environment. ......................64
4-32 The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Outdoor Multipath and NLOS Environment with a Kalman Filter. .................................................................65

4-33 The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Outdoor Multipath and NLOS Environment with a Kalman Filter. .................................................................66

4-34 Comparing the Error in Location Estimation in a LOS Environment between the Proposed Method and the MMSE Method. .................................................................69

4-35 Comparing the Error in Location Estimation Resulting from 20 m Error Attack from Three Spoofing Users a LOS Environment between the Proposed Method and the MMSE Method. .................................................................69

4-36 Comparing the Error in Location Estimation Resulting from 60 m Error Attack from Two Spoofing Users in a LOS Environment between the Proposed Method and the MMSE Method. .................................................................70

4-37 Comparing the Error in Location Estimation in a 3D LOS Environment with mobile users between the Proposed Method and the MMSE Method with no spoofing attacks. .................................................................70

4-38 Comparing the Error in Location Estimation in a 3D Indoor NLOS Environment with mobile users between the Proposed Method and the MMSE Method with no spoofing attacks. .................................................................71
Comparing the Error in Location Estimation in a 3D Outdoor NLOS Environment with mobile users between the Proposed Method and the MMSE Method with no spoofing attacks.
List of Abbreviations

2D..........................2 Dimensional
3D..........................3 Dimensional
4G..........................4th Generation Cellular Technology

AOA..........................Angle of Arrival

BPF..........................Band Pass Filter
BSF..........................Band Stop Filter

FM..........................Frequency Modulation

GPS..........................Global Positioning System

IP..........................Internet Protocol

J..............................Joule

K..............................Kelvin
kg..........................kilogram
km..........................kilometer
kmol..........................kilo mole

LOS..........................Line of Sight
LPF..........................Low Pass Filter

m..........................meter
MSE..........................Mean Square Error
MMSE..........................Minimum Mean Square Error
mol..........................mole

NLOS..........................Non-Line of Sight

Pa..........................Pascal
pdf..........................probability density function

QPSK..........................Quadrature Phase Shift Keying
RSS .......................... Received Signal Strength
s .................................. second
TDOA ............................ Time Difference of Arrival
TOA ............................... Time of Arrival
UMTS ............................ Universal Mobile Telecommunications System
VCO ............................... Voltage Controlled Oscillator
W ................................. Watt
WSN ............................... Wireless Sensor Network
Chapter 1

Introduction

GPS (Global Positioning System), while very useful in our daily life, has its limitations in location estimation. The low power received signal from GPS satellites outdoors along with multipath and the geometry of the satellite positions cause errors in locating a device while making it almost impossible to use indoors.

To overcome the limitations of the GPS regarding the position of the user, research in the past few years has been focused on the use of the cooperative localization concept. Cooperative localization is based on the fact that users in a certain environment can help each other to determine their locations using information sent from several other users.

Research in cooperative localization is split amongst different applications, technology used, and location identification method. Some focused on helping robots navigate buildings [1], while some worked on cellular networks [2]. Some used the newly approved UWB devices in cooperative localization since these devices can deal with sharp pulses resulting in high accuracy localization [3], while others used WSN (Wireless Sensor Networks) [4] for cooperative localization. Different location identification methods were used, such as TOA (Time of Arrival) [3], RSS (Received Signal Strength),
TDOA (Time Difference of Arrival), AOA (Angle of Arrival) or a combination of two or more methods [5].

The conventional techniques in localization use three sources of information to identify the user’s location (Figure 1-1). This would work just fine for LOS (Line of Sight) signal propagation. Unfortunately, situations of NLOS (Non-LOS), multipath, and spoofing attacks would suffer great errors. Using multiple sources of information would certainly help the accuracy but would take more time to compute and drain resources. This research shows that processing information from at least seven transmitting users can considerably enhance the location estimation accuracy without suffering in computation time from the higher number of processed user information.

![Figure 1-1: The triangulation method uses the distance from three users to determine the location of the forth user \((x_0, y_0)\).](image)
Kalman filter [6] uses the movement pattern of a user to predict the location of the user by correcting and learning from previous movements. Using an error covariance matrix to correct location estimates, and the use of the laws of the movement pattern, Kalman filter effectively corrected the estimates provided from processing the information from the transmitting users.

1.1. Motivation for Research

GPS, while very useful for civil, commercial, and military uses, the low power received signal from GPS satellites, coupled with multipath and the geometry of the satellite positions cause errors in locating a device while making it very difficult to use indoors. Being at an altitude of 20,000 km [7], the GPS satellite signal arrives very weak to any GPS receiver [8], which makes it very difficult for any localization process to work indoors using GPS. Multipath and the geometry of the satellite positions add to the problems of GPS, where multipath causes errors in estimating the distance between the GPS receiver and the satellite, and the geometry of the satellite positions affects the search area where the GPS receiver needs to find its location.

After establishing that GPS has its flaws, a solution needs to be found. Cooperative localization solves these problems; where it works indoors and gets better accuracy. It solves the problem of low power by using transmitters much closer (1 km compared to 20,000 km). Problems associated with multipath and the geometry of the transmitters can be resolved by processing more information from transmitting users.
Unfortunately, the current research in cooperative localization uses technologies that will not work for the hardware of cellular networks, yields errors that surpass those of GPS, and are prone to spoofing attacks. The aim of this research is to use technologies that work with the cellular networks and users, have a better accuracy than the GPS, and are not affected by spoofing attacks. Cellular networks these days cover more area than before and the prospect of using all the cellular phones for those networks is enormous; where each cellular phone can be used as a transmitter (or a source) of location information compared to only 24 GPS satellites orbiting the earth.

1.2. Related and Previous Works

Research has been done on cooperative localization using various methods, technologies and field of use; from Ultra Wide Band (UWB) [3] to Wireless Sensor Networks (WSN) [4], from time of arrival to direction of arrival, and from robots [1] to cellular networks [2, 9].

Some have achieved reasonable results (errors of less than 1 m) [10] and many have achieved disappointing results (errors of more than 100 m) [11].

Cooperative localization using UWB has been extensively studied [3, 12, 13]. The main idea behind using UWB in localization is the use of large bandwidth to send sharp pulses, which allows them to get low location estimation errors.

UWB received wide interest after the U.S. Federal Communications Commission (FCC) allowed the use of unlicensed UWB communications [14]. The large bandwidth used by UWB improves reliability, as the signal contains different frequency
components, which increases the probability that at least some of them can go through or around obstacles. Furthermore, a large absolute bandwidth offers improved localization accuracy [12].

The general aspects of using UWB in cooperative localization have been discussed in [3, 12, 13], where [3] focused on Time of Arrival (TOA) in UWB localization, and [12] studied the use of UWB localization in WSNs. The research in [13] focused on error mitigation in UWB localization caused by NLOS.

Although good results were obtained using UWB in cooperative localization, the short range it works in [15], the large bandwidth required, and the need to develop new equipment for real applications makes it impractical these days. In these regards, the objective of this research is to build on more popular frequency bands, modulation methods, and multiple access schemes.

WSN [4, 12, 16, 17] and robots [1, 18] have been used in research for cooperative localization. Cooperative localization in WSN’s was focused on a small number of static users and robots used cooperative localization to navigate specific buildings.

Although they have good applications, these applications are limited and focused on small networks and specific uses, usually with the aid of GPS, which this research is trying to replace or at least not use.

There has been some research dealing with localization for cellular networks [5, 9, 11, 19], from UMTS (Universal Mobile Telecommunications System) [9] to 4G cellular networks [5, 11, 19]. The localization protocol model was set for higher layers in [9] but not the physical layer and without any simulation or results. The algorithms in [5, 11, 19] depend heavily upon base stations, where they used two base stations and a mobile user
or one base station and two mobile users. In addition, the results they obtained had a mean error of 15 m or more, where this research achieved, as a worst case scenario, around 10 m and around 1 m for mobile users moving in a pattern recognized by Kalman filter.

Unlike other researches [11], which mainly used the base station and only one or two users to provide information, it is shown that it is not very practical to use only three sources of location information, in terms of localization accuracy and countering spoofing attacks. Although the reliable use of base station was not used, the results obtained were encouraging enough not to need to use base stations as location information sources.

A few results have been published to address the effects of spoofing attacks on cooperative localization systems in WSNs [20-28], but none of them addressed the issue for cooperative localization in cellular networks. They all agree that any motivated attacker can easily bypass the authentication methods, and that the main countermeasure is to identify and/or remove the effect of the attacker on the user’s location estimation process [20]. One method that was used is limited to WSNs [21], since WSNs have a limited range due to the limited power of the sensor, and would not work for cellular networks. Others [22-24] assumed that the user is always in the middle of the triangle created by the transmitting users, which eliminates a lot of good location estimation possibilities. Some [25, 26] did not show the effect of the attacks and the methods used on the accuracy of the user’s location estimation.

One [27], on the other hand, used the same idea of processing information from multiple transmitting users. In [27], the method of MMSE (Minimum Mean Square
Estimation) was used twice, the first time for location estimation; by calculating the location estimation using a matrix equation incorporating the information from all the users. For the second time, MMSE was used to eliminate the effects of the attack by repeating the process of MMSE with the elimination of a certain number of users to see if the effect of the attack was eliminated. The proposed method in [27] was able to significantly reduce the effects of one attacker, but was less effective to do that for multiple attackers.

Another location estimation method was also described in [27], where the voting-based location estimation quantizes the target field into a grid of cells, and has each sensor node determine how likely it is in each cell based on each location reference [27]. While the results are very promising, the accuracy of this method heavily depends on the size of the target area. Applying this to a WSN works since the target area of any WSN is small enough for accurate measurements. However, the larger areas where cellular networks operate would make it difficult to implement this method for cooperative localization in cellular networks.

An expanded version of [27] was presented in [28], expanding the results of the methods in [27] and modifying other ones. Enhanced Greedy Algorithm for Attack Resistant MMSE (EARMMSE) [28] changed the conditions of acceptable number of location references; whereas [27] focused on the use of a minimum of three location references, EARMMSE in [28] changed the minimum to two location references. EARMMSE effectively reduces the effect of any attacks in the author’s controlled environment, but fails to do so in a random placement of beacons.
This research used general methods of modulation, multiple access schemes so that more than one platform can work using this cooperative localization method. Expanding the research to use other signals like WLAN, FM (Frequency Modulation) and satellite signals [29] would be impractical since they differ from cellular signals in more than one way, but using this cooperative localization method is still possible for each separate technology.

1.3. Research Objective

The main objective of this research is to focus on proving that cellular network based indoor localization is possible in a 2D and 3D setting, to prove that GPS can be replaced indoors and outdoors, and to mitigate any possible spoofing attack on the network of users.

Since GPS does not work indoors due to the low received power of the GPS signal, this research will show that through cooperative localization, it is possible to have reliable location information. This will be done using different scenarios; outdoor and indoor LOS, outdoor NLOS, and indoor NLOS, all in 2D environment and 3D environment.

The case of outdoor and indoor LOS will simulate users in an open area whether indoor or outdoor. Outdoor NLOS will simulate users in a city with buildings around. Indoor NLOS will simulate a building with various rooms (in 2D), and a building with various rooms and levels (in 3D).
Two different methods will assist in reducing the estimation errors resulting signal
degradation from the information transmitting users, and from any possible spoofing
attacker. The first method is increasing the number of information transmitting users
being processed and filtering out the extreme estimates resulting from high signal
degradation and spoofing attacks. The second method is Kalman filter which predicts the
movement of mobile users to reduce the estimation errors.

While in 3D a device, installed on many cellular phones these days, called a
barometric sensor would assist in the calculations of the height information for the user.
Using the pressure information provided from the sensor and the pressure information
sent from other users, the target user can easily calculate the height information required
for accurate location estimation.

1.4. Dissertation Organization

The rest of this dissertation is organized as follows. Chapter two briefly describes the
theoretical background for the methods used in this research, starting with the
mathematical background, the signal propagation models, the Kalman filtering for mobile
users, the barometric equations used for the height information in the 3D cases, and
ending with the background on spoofing attacks.

Chapter three lays down the setting for the simulation environment, and describes the
different components of the system model.
Chapter four shows the different results the research obtained through simulation and comments on those results. The results are split into case-by-case results that only apply to the specific case they represent and general results that apply to all the cases.
Chapter 2

Theoretical Background

2.1. Mathematical Background

To find the effect of the error in reporting the transmitting users’ location on the probability density function (pdf) of the error ($e_{x_0}$ and $e_{y_0}$) in estimating the target user’s location ($x_0$ and $y_0$), the following triangulation equations are used:

\[
\begin{align*}
(x_1 + e_{x_1} - (x_0 + e_{x_0}))^2 + (y_1 + e_{y_1} - (y_0 + e_{y_0}))^2 &= D_1^2 \\
(x_2 + e_{x_2} - (x_0 + e_{x_0}))^2 + (y_2 + e_{y_2} - (y_0 + e_{y_0}))^2 &= D_2^2 \\
(x_3 + e_{x_3} - (x_0 + e_{x_0}))^2 + (y_3 + e_{y_3} - (y_0 + e_{y_0}))^2 &= D_3^2
\end{align*}
\]  

(2 - 1)

where $e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}$, and $e_{y_3}$ are the errors in reporting the users’ location $x_1, y_1, x_2, y_2, x_3$, and $y_3$.

Let’s define four additional dummy variables to find $e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}$, and $e_{y_3}$ in terms of $e_{x_0}$ and $e_{y_0}$:

\[
\begin{align*}
w_0 &= e_{x_1} \\
\nu_0 &= e_{x_2} \\
n_0 &= e_{x_3}
\end{align*}
\]  

(2 - 2)
\[ m_0 = e_{y_1} \]

By substituting Equations (2-2) in Equations (2-1), the following equations are obtained:

\[
\begin{align*}
  e_{x_1} &= (x_0 + e_{x_0}) - x_1 \mp \sqrt{D_1^2 - (y_1 + m_0 - (y_0 + e_{y_0}))^2} \\
  e_{x_2} &= (x_0 + e_{x_0}) - x_2 \mp \sqrt{D_2^2 - (y_2 + e_{y_2} - (y_0 + e_{y_0}))^2} \\
  e_{x_3} &= (x_0 + e_{x_0}) - x_3 \mp \sqrt{D_3^2 - (y_3 + e_{y_3} - (y_0 + e_{y_0}))^2} \\
  e_{y_1} &= (y_0 + e_{y_0}) - y_1 \mp \sqrt{D_1^2 - (x_1 + w_0 - (x_0 + e_{x_0}))^2} \\
  e_{y_2} &= (y_0 + e_{y_0}) - y_2 \mp \sqrt{D_2^2 - (x_2 + v_0 - (x_0 + e_{x_0}))^2} \\
  e_{y_3} &= (y_0 + e_{y_0}) - y_3 \mp \sqrt{D_3^2 - (x_3 + n_0 - (x_0 + e_{x_0}))^2}
\end{align*}
\]  

(2 - 3)

The joint probability density function of \( e_{x_0}, e_{y_0}, w_0, v_0, n_0, \) and \( m_0 \) can be found using the known random variable transformation equation:

\[
\begin{align*}
  f_{e_{x_0}, e_{y_0}, w_0, v_0, n_0, m_0}(e_{x_0}, e_{y_0}, w_0, v_0, n_0, m_0) \\
  &= f_{e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}, e_{x_0}, e_{y_0}}(e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}) |J| \\
  \text{where the Jacobian matrix } (J) \text{ is defined as}
\end{align*}
\]  

(2 - 4)
and

$$J = \begin{bmatrix}
\frac{\partial e_{x_1}}{\partial x_0} & \frac{\partial e_{x_1}}{\partial y_0} & \frac{\partial e_{x_1}}{\partial n_0} & \frac{\partial e_{x_1}}{\partial m_0} \\
\frac{\partial e_{x_2}}{\partial x_0} & \frac{\partial e_{x_2}}{\partial y_0} & \frac{\partial e_{x_2}}{\partial n_0} & \frac{\partial e_{x_2}}{\partial m_0} \\
\frac{\partial e_{x_3}}{\partial x_0} & \frac{\partial e_{x_3}}{\partial y_0} & \frac{\partial e_{x_3}}{\partial n_0} & \frac{\partial e_{x_3}}{\partial m_0} \\
\frac{\partial e_{y_1}}{\partial x_0} & \frac{\partial e_{y_1}}{\partial y_0} & \frac{\partial e_{y_1}}{\partial n_0} & \frac{\partial e_{y_1}}{\partial m_0} \\
\frac{\partial e_{y_2}}{\partial x_0} & \frac{\partial e_{y_2}}{\partial y_0} & \frac{\partial e_{y_2}}{\partial n_0} & \frac{\partial e_{y_2}}{\partial m_0} \\
\frac{\partial e_{y_3}}{\partial x_0} & \frac{\partial e_{y_3}}{\partial y_0} & \frac{\partial e_{y_3}}{\partial n_0} & \frac{\partial e_{y_3}}{\partial m_0}
\end{bmatrix}$$  \hspace{1cm} (2 - 5)

$$f_{e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}}(e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}) = \frac{1}{(2\pi\sigma^2\sqrt{1 - \rho^2})^3} \exp\left(\frac{2\rho(e_{x_1}e_{y_1} + e_{x_2}e_{y_2} + e_{x_3}e_{y_3}) - e_{x_1}^2 - e_{y_1}^2 - e_{x_2}^2 - e_{y_2}^2 - e_{x_3}^2 - e_{y_3}^2}{2(1 - \rho^2)\sigma^2}\right)$$  \hspace{1cm} (2 - 6)

where \(\sigma\) is assumed to be an equal variance of \(e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}\), and \(\rho\) is the correlation coefficient between \(e_x\) and \(e_y\) and is assumed equal for each pair of \(e_x\) and \(e_y\).

The determinant of the Jacobian matrix is

$$|J| = \frac{J_n}{J_d}$$  \hspace{1cm} (2 - 7)

where

$$J_n = \left( x_1 + w_0 - (x_0 + e_{x_0}) \right) \left( x_2 + v_0 - (x_0 + e_{x_0}) \right) \left( x_3 + n_0 - (x_0 + e_{x_0}) \right) \left( y_1 + m_0 - (y_0 + e_{y_0}) \right)$$

$$J_d = \left( x_1 + w_0 - (x_0 + e_{x_0}) \right) \left( x_2 + v_0 - (x_0 + e_{x_0}) \right)$$
Finding the probability density function of the error \((e_{x_0} \text{ and } e_{y_0})\) in estimating the target user’s location \((x_0, y_0)\) can be done by integrating \(f_{e_{x_0}, e_{y_0}, w_0, n_0, m_0}(e_{x_0}, e_{y_0}, w_0, v_0, n_0, m_0)\) to obtain the marginal probability density function of:

\[
f_{e_{x_0}, e_{y_0}}(e_{x_0}, e_{y_0})
\]

\[
= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f_{e_{x_0}, e_{y_0}, w_0, v_0, n_0, m_0}(e_{x_0}, e_{y_0}, w_0, v_0, n_0, m_0) dw_0 dv_0 dn_0 dm_0 \quad (2 - 10)
\]

Substituting Equation (2-4) into Equation (2-10) results in the following equation:

\[
f_{e_{x_0}, e_{y_0}}(e_{x_0}, e_{y_0})
\]

\[
= \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f_{e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}}(e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}) \sqrt{w_0^2 + v_0^2 + n_0^2 + m_0^2} dw_0 dv_0 dn_0 dm_0 \quad (2 - 11)
\]
To facilitate the calculations of the quadruple integration, the exponential term in Equation (2-6) can be approximated using the first two terms of the Taylor series expansion of the exponential function:

\[
f_{e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}}(e_{x_1}, e_{y_1}, e_{x_2}, e_{y_2}, e_{x_3}, e_{y_3}) = \frac{1}{(2\pi\sigma^2\sqrt{1 - \rho^2})^3} \left( 1 + \frac{2\rho(e_{x_1}e_{y_1} + e_{x_2}e_{y_2} + e_{x_3}e_{y_3}) - e_{x_1}^2 - e_{x_2}^2 - e_{x_3}^2 - e_{y_1}^2 - e_{y_2}^2 - e_{y_3}^2)}{2(1 - \rho^2)\sigma^2} \right)
\] (2 - 12)

The detailed derivation of the equations for \(f_{e_{x_0}, e_{y_0}}(e_{x_0}, e_{y_0})\) and \(f_{e_{x_0}}(e_{x_0})\) are shown in Appendix A, and the plot of the probability density function of \(e_{x_0}\) is shown in Figure 2-1. Axis values are not shown since they depend on the values chosen for \(\sigma\) and \(\rho\). Although the variance of the error \(\sigma_{e_{x_0}}\) slightly varies depending on the values of \(\rho\), the small variation is centered around \(10\sigma\), but with the filtration and averaging processes, the
variance of the error was reduced to a value close to \( \sigma \). The value of the correlation (\( \rho \)) for the joint probability density function (\( f_{e_{x_0}e_{y_0}}(e_{x_0},e_{y_0}) \)) also varies depending on the input \( \rho \) between \((-0.03, 0.03)\). Taking the worst case of the most dependency between \( e_{x_0} \) and \( e_{y_0} \), \( \rho \) was set to 0.03. The Figure for the probability density function of \( e_{y_0}, f_{e_{y_0}}(e_{y_0}) \), is not shown because the same equations apply to it and would result in the same plot. Furthermore, Figure 2-1 shows that the probability density function \( f_{e_{x_0}}(e_{x_0}) \) has the characteristics of a Gaussian probability density function.

Using the fact that the probability density function has the characteristics of a Gaussian distribution, the Maximum Likelihood estimation is known to be the mean of all the samples [30]. By removing the extreme estimated values that result from extreme errors in the transmitting users’ location or as a result of a spoofing attack, the maximum likelihood estimate of a group of samples is the mean of those samples.

2.2. Signal Propagation Models

There are three main propagation paths for any signal to travel a transmitter to a receiver; the LOS (Line-of-Sight) path, the NLOS through the walls path, and the NLOS reflected path.

Signal loss and degradation in LOS cases are based on the free space propagation model; where the signal’s received power depends on the transmitted signal power (\( P_{Tx} \)), the carrier frequency [31] of the signal (\( f_c \)), and the distance between the transmitter and receiver (\( D \)).
\[ P_{Rx} = P_{Tx} \left( \frac{C}{4 \pi f D} \right)^2 \]  

(2 - 13)

where \( C \) is the speed of light.

The NLOS signal passing through the walls and the NLOS signal reflected paths can be modeled using the electrical field reflection equations [31]:

\[ \Gamma_\parallel = \frac{E_r}{E_i} = \frac{\eta_2 \sin \theta_t - \eta_1 \sin \theta_i}{\eta_2 \sin \theta_t + \eta_1 \sin \theta_i} \]  

(2 - 14)

\[ \Gamma_\perp = \frac{E_r}{E_i} = \frac{\eta_2 \sin \theta_i - \eta_1 \sin \theta_t}{\eta_2 \sin \theta_i + \eta_1 \sin \theta_t} \]  

(2 - 15)

where \( \Gamma_\parallel \) and \( \Gamma_\perp \) are the parallel and orthogonal reflection coefficients, \( E_r \) and \( E_i \) are the reflected and incident electrical field strengths, \( \theta_t \) and \( \theta_i \) are the incident and transmitted angles with the plane of incidence, and \( \eta_1 \) and \( \eta_2 \) are the impedances for the first and second materials, which are defined as

\[ \eta = \sqrt{\mu/\varepsilon} \]  

(2 - 16)

where \( \mu \) is the permeability of a medium and \( \varepsilon \) is the permittivity of the medium.

But since one of the mediums in the case of indoor propagation is free space and \( \mu_1 = \mu_2 \approx 1 \) [32], the reflection coefficients’ equations can be rewritten as:

\[ \Gamma_\parallel = \frac{-\varepsilon_r \sin \theta_i + \sqrt{\varepsilon_r - \cos^2 \theta_i}}{\varepsilon_r \sin \theta_i + \sqrt{\varepsilon_r - \cos^2 \theta_i}} \]  

(2 - 17)

and

\[ \Gamma_\perp = \frac{\sin \theta_i - \sqrt{\varepsilon_r - \cos^2 \theta_i}}{\sin \theta_i + \sqrt{\varepsilon_r - \cos^2 \theta_i}} \]  

(2 - 18)
and since
\[
P = \frac{E^2}{R}
\]  
\[(2 - 19)\]

where \(P\) is the power of the signal and \(R\) is the impedance of the free space. Substituting in
\[
E_r = \Gamma E_i
\]
\[(2 - 20)\]
\[
E_t = (1 + \Gamma)E_i
\]
\[(2 - 21)\]

where \(\Gamma\) can either be the parallel or orthogonal reflection coefficient depending on whether \(E\) is vertical or horizontal to the incidence plane.

Equations (2 - 20) and (2 - 21) become
\[
P_r = \Gamma^2 P_t
\]
\[(2 - 22)\]
\[
P_t = (1 + \Gamma^2)^2 P_t
\]
\[(2 - 23)\]

where \(P_r\) is the reflected signal power, \(P_t\) is the transmitted (through the walls) signal power and \(P_i\) is the incidence signal power. The second square is present in Equation (2 - 23) since the signal needs to be transmitted through two incidence planes before exiting the wall surface.

Although the discussion here is about the NLOS case, NLOS signals are not the only ones present, a significant percentage of LOS signals is also present (Table 2.1).

2.3. Mobile Users and Kalman Filtering

Using the fact that any single mobile user usually moves in a pattern, the Kalman filter can be used to predict this movement pattern and help identify the user’s location [6].
The x and y coordinates (the z coordinate was not used in Kalman filter since barometric equations (section 2.4) were used to find it) were separated when processed through the Kalman filter.

The state vector was defined as

\[ S(n) = \begin{bmatrix} x(n) \\ v(n) \end{bmatrix} \]  \hspace{1cm} (2 - 24)

where \( x(n) \) is the x-coordinate (or the y-coordinate) at the time \( n \), and \( v(n) \) is the velocity at the same time.

Defining the state equation through the laws of physics gives the following:

\[ S(n) = AS(n - 1) \]  \hspace{1cm} (2 - 25)

where \( A = \begin{bmatrix} I & \Delta t \\ 0 & I \end{bmatrix} \), and \( \Delta t \) is the time difference between measurements.

The location estimates through processing the transmitting users’ information were set as the observation values for \( S \)

<table>
<thead>
<tr>
<th>Number of processed information signals</th>
<th>Percentage of NLOS signals (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>43.28</td>
</tr>
<tr>
<td>4</td>
<td>46.33</td>
</tr>
<tr>
<td>5</td>
<td>49.00</td>
</tr>
<tr>
<td>6</td>
<td>55.50</td>
</tr>
<tr>
<td>7</td>
<td>60.42</td>
</tr>
<tr>
<td>8</td>
<td>61.00</td>
</tr>
<tr>
<td>9</td>
<td>68.22</td>
</tr>
<tr>
<td>10</td>
<td>70.20</td>
</tr>
<tr>
<td>11</td>
<td>73.81</td>
</tr>
<tr>
<td>12</td>
<td>72.41</td>
</tr>
</tbody>
</table>
\[ \hat{\mathbf{S}}(n) = C(n)\mathbf{S}(n) + \mathbf{w}(n) \quad (2-26) \]

where \( \mathbf{w} \) is the zero mean white Gaussian noise in estimating the user’s location, and 
\( C = [I \ 0] \) for this case.

Starting with initial values for
\[ \hat{\mathbf{S}}(\theta|\theta) = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad (2-27) \]

and the initial error covariance matrix was defined as
\[ \mathbf{P}(\theta|\theta) = \begin{bmatrix} L & 0 \\ 0 & L \end{bmatrix} \quad (2-28) \]

where \( L \) is a very large number, set here to be \( 10^5 \).

To estimate the user location through Kalman filter, the update phase uses the initial values of the x and y coordinate and the estimates obtained from processing the transmitting users’ information. Each coordinate is processed separately using the following matrix equations at each iteration (user location update and estimation) [6]
\[ \hat{\mathbf{S}}(n|n-1) = \mathbf{A}\hat{\mathbf{S}}(n-1|n-1) \quad (2-29) \]
\[ \mathbf{P}(n|n-1) = \mathbf{A}\mathbf{P}(n-1|n-1)\mathbf{A}^T \quad (2-30) \]
\[ \mathbf{K}(n) = \mathbf{P}(n|n-1)\mathbf{C}(n)\mathbf{P}(n|n-1)\mathbf{C}(n)^T + \mathbf{Q}_w(n) \mathbf{C}(n) \mathbf{P}(n|n-1)\mathbf{C}(n)^T + \mathbf{Q}_w(n) \mathbf{C}(n) \mathbf{P}(n|n-1)\mathbf{C}(n)^T + \mathbf{Q}_w(n) \mathbf{C}(n) \mathbf{P}(n|n-1)\mathbf{C}(n)^T + \mathbf{Q}_w(n) \quad (2-31) \]
\[ \hat{\mathbf{S}}(n|n) = \hat{\mathbf{S}}(n|n-1) + \mathbf{K}(n)[\hat{\mathbf{S}}(n) - \mathbf{C}(n)\hat{\mathbf{S}}(n|n-1)] \quad (2-32) \]
\[ \mathbf{P}(n|n) = [I - \mathbf{K}(n)\mathbf{C}(n)]\mathbf{P}(n|n-1) \quad (2-33) \]

where \( \mathbf{Q}_w = [\sigma_w^2] \), \( I \) is the identity matrix, and \( \mathbf{K} \) is the Kalman filter gain.
2.4. Barometric Equations

Using a barometer in the 3D localization process helps to more accurately estimate the user’s height. Users can determine their height using the pressure reading of the barometer given a certain temperature. Assuming the user has no knowledge of the surrounding temperature, the user can estimate the temperature using the pressure information other users transmit [33]:

\[
T_i = \frac{mg h_i}{R (\ln P_0 - \ln P_i)} \quad (2-34)
\]

where \(T_i\) is the calculated temperature at the location of user \(i\), \(m\) is the mean molecular weight of air (28.9644 kg.kmol\(^{-1}\)), \(g\) is the standard geopotential gravitational constant (9.80665 m\(^2\)s\(^{-1}\)), \(h_i\) is the height of the user \(i\) above sea level (in meters), \(R\) is the universal gas constant (8.31432 joules K\(^{-1}\) mol\(^{-1}\)), \(P_o\) is the sea level standard atmospheric pressure (101325 Pa), and \(P_i\) is the pressure measured at the location of user \(i\).

To estimate the temperature at the location of the target user, the calculated temperatures at the locations of the other users are averaged since all of the users are in close proximity of each other using the equation:

\[
T_u = \frac{1}{N} \sum_{i=1}^{N} T_i \quad (2-35)
\]

where \(T_u\) is the estimated temperature at the location of the target user, and \(N\) is the total number of processed information transmitting users.

Finally, to estimate the height of the target user above sea level, the target user utilizes the temperature estimated in Equation (2-34) with the pressure information
obtained from the barometer sensor onboard the target user’s device [34] using the following equation:

$$h_u = \frac{T_u R}{mg} (\ln P_0 - \ln P_u) \tag{2 - 36}$$

where $h_u$ is the estimated height of the target user, and $P_u$ is the pressure measured at the location of the target user.

Location estimation using on RSS in a 3D environment uses the same basic triangulation equations to calculate the distance between the sender and receiver (Equation (3-6)):

$$(x_1 - x_0)^2 + (y_1 - y_0)^2 + (z_1 - z_0)^2 = D_1^2$$

$$(x_2 - x_0)^2 + (y_2 - y_0)^2 + (z_2 - z_0)^2 = D_2^2 \tag{2 - 37}$$

$$(x_3 - x_0)^2 + (y_3 - y_0)^2 + (z_3 - z_0)^2 = D_3^2$$

$$(x_4 - x_0)^2 + (y_4 - y_0)^2 + (z_4 - z_0)^2 = D_4^2$$

where $x_1, y_1, z_1, x_2, y_2, z_2, x_3, y_3, z_3, x_4, y_4, z_4$ are the x, y and z coordinates of users 1, 2, 3, and 4. $x_0, y_0$ and $z_0$ are the x, y and z coordinates of the unknown user location. $D_1, D_2, D_3$, and $D_4$ are the distances between the target user and users 1, 2, 3, and 4.

The use of a barometric sensor, which helps in providing the users with altitude information, in the users’ devices, allows the reduction of the number of needed users for the location estimation in 3D to three users instead of four.

After reducing the number of location estimation users to three users and since the barometric sensor provided the information for $z_i$, it can be considered as a constant, Equations (2-37) can be written as
where $\delta_i = D_i^2 - (z_i - z_0)^2$

2.5. Spoofing

Spoofing has been the subject of many researches in disciplines like IP (Internet Protocol) spoofing and email spoofing, but only a few [35, 36] were about the effects of spoofing on GPS receivers. They were mainly focused on the effect, the detection, and the combat of spoofing attacks on GPS receivers.

With the introduction of cooperative localization to any market, spoofer would find cooperative localization an attractive methodology to attack the cellular network user due to the fact that the manipulation of transmitters would be easy. But very little research has been done to detect and combat the effects of spoofing attacks on cooperative localization network of users, since cooperative localization is relatively new and not commercialized yet.

In GPS spoofing, the attacker claims to be a GPS satellite, to gain access to a network to disrupt the services provided by the GPS satellites, and manipulate users to the attacker’s advantage. In cooperative localization, spoofing attacks are much easier, since the transmitters are easier to access and attacks are more effective.

Spoofing attackers in cooperative localization would manipulate the data sent in their transmitted signal to disrupt the services of cooperative localization; either by changing
the power of the transmitted signal, the time stamp provided in the message, or the transmitter’s location provided in the message. By doing so, any user trying to locate their device will be led to believe in a location they are not near (Figure 2-2).

One or two estimates of the user’s location might be enough for the cases where no spoofing occurs, but when at least one spoofing user enters the network, errors in the user location estimate start to duplicate.

Adding more information transmitting users, and subsequently location estimates, would help in reducing the effect of the spoofing user; by reducing the overall percentage to false location estimates.

As a worst case scenario (all spoofing attackers are in one location), the percentage to false estimates ($F$) to the total number estimates ($E$) is defined as

$$\text{percentage} = 100 \times \frac{F}{E} = 100 \times \frac{E - T}{E} \quad (2 - 39)$$

where $T$ is the number of true estimates. And since only three users are needed for each estimate, the number of estimates from $U$ number of information transmitting users of which the target user processes can be defined as:

![Figure 2-2: The Effect of the Spoofing attack on the Target User](image)
\[
E = \binom{U}{3} = \frac{U!}{3!(U-3)!} = \frac{1}{6} U(U - 1)(U - 2) \tag{2-40}
\]

which makes Equation (2-39)

\[
\text{percentage} = 100 \times \frac{(U)(U - 1)(U - 2) - (U - S)(U - S - 1)(U - S - 2)}{(U - S)(U - S - 1)(U - S - 2)}
= 100 \times \frac{S^3 + 3U^2S - 3US^2 + 3S^2 - 6US + 2S}{U^3 - S^3 - 3U^2S + 3US^2 - 3U^2 - 3S^2 + 6US + 2U - 2S} \tag{2-41}
\]

where \(S\) is the number of spoofing users.

Introducing mobile users to the network, since they can move away from the spoofing attackers, would considerably aid in reducing the effect of the spoofing attacks.

### 2.6. Minimum Mean Square Error (MMSE)

The 2D MMSE location estimation method detailed in [27, 28] uses the same basic principle as the proposed method of using information from more than three transmitting users to mitigate the effect of spoofing attacks, but differs from the proposed method on how this information is processed.

The MMSE location estimation method depicted in Figure 2-3 eliminates location estimates it thinks are affected by the spoofing attack, by calculating the MSE (Mean Square Error) using Equation (2-42) and eliminating any location estimate that falls above a certain threshold \((\tau)\), where the MSE is defined as

\[
m = \sum_{i=1}^{m} \left( D_i - \sqrt{(\bar{x}_0 - x_i)^2 + (\bar{y}_0 - y_i)^2} \right)^2 \tag{2-42}
\]
where $\gamma$ is the MSE, $x_i$ and $y_i$ are the x and y coordinates of the $i^{th}$ user, $\bar{x}_0$ and $\bar{y}_0$ are the estimated location of the target user, $D_i$ is the distance between user $i^{th}$ and the target user, and $m$ is the number of processed information transmitting users.

In each iteration, the MMSE method calculates $\gamma$ for each group of information transmitting users. The groups are created by removing one information transmitting user from each group, by doing so the MMSE method is trying to exclude the spoofing attackers. At the end of the iteration, the MMSE method keeps the group with the least $\gamma$, assuming that the high error comes from the spoofing attackers’ information. In the next
iterations, the MMSE method repeats the process until the $\gamma$ falls below the threshold ($\gamma < \tau$), which indicates that all the spoofing attackers have been eliminated, or until a certain number of information transmitting users are present, which indicates the failure to detect and eliminate the information from the spoofing attackers.

Although the literature for MMSE does not mention the study of any 3D cases, Equation (2-42) could be changed to accommodate any 3D environment being tested by adding the $z$-dimension as follows:

$$\gamma^2 = \sum_{i=1}^{m} \frac{(D_i - \sqrt{(\tilde{x}_0 - x_i)^2 + (\tilde{y}_0 - y_i)^2 + (\tilde{z}_0 - z_i)^2})^2}{m}$$  \hspace{1cm} (2-43)

where $z_i$ is the $z$ coordinate of the $i^{th}$ user, and and $\tilde{z}_0$ is the estimated $z$ coordinate of the target user. Equation (2-43) was utilized for the simulation of the 3D cases to be compared to the proposed method in chapter 4.
Chapter 3

Environment and Modeling

3.1. User Environment

To simulate the real world environment, transmitted user locations were sent with a normally distributed error of mean ($\mu$) zero and variance ($\sigma^2$) (Table 3.1), with a correlation ($\rho$) between the coordinates’ error of 0.03 (section 2.1).

The relationship between the variance of the assumed error in the transmitted user locations and the average of the error in the estimated location is shown in Figure 3-1. Although the input error variance was changed (from 1 m to 35 m) for each case in Figure 3-1, the resulting average error was not significantly changed (<5 m output change for a

<table>
<thead>
<tr>
<th>Simulation Environment Parameters</th>
<th>Parameter Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sigma$</td>
<td>1-10 m</td>
</tr>
<tr>
<td>$\mu$</td>
<td>0</td>
</tr>
<tr>
<td>$\rho$</td>
<td>0.03</td>
</tr>
<tr>
<td>$P_{Tx}$</td>
<td>5 W</td>
</tr>
</tbody>
</table>
Since the average output error for each case is different (chapter 4), different cases were tested to see the effect of the input error on the output error. The different lines in Figure 3-1 were not labeled since each one represents all the cases that have close output error values.

This research on cooperative localization assumed different areas depending on the case being studied, with different user densities:

3.1.1. Two Dimensional (2D) Localization

In 2D, the areas being simulated are flat with the only heights being considered are the user’s antenna heights. The antenna heights were assumed to be 1.5 m. Three cases were studied for the 2D RSS triangulation calculations:
a. LOS: All users are assumed to be in direct line of sight of each other, thus allowing signals to travel with no obstacles; free space propagation model is assumed.

b. Indoor NLOS: Users are assumed to be inside an office building with heavy blockage present between users resulting in weak signals due to signal reflection and refraction.

c. Outdoor NLOS: Light blockage, due to wide spread of buildings, between users is assumed with large areas of direct line of sight.

3.1.2. Three Dimensional (3D) Localization

In 3D, the areas being simulated have multiple heights in addition to the height of the user’s antenna. Three cases were studied for the 3D RSS triangulation calculations:

a. LOS: Users are assumed to be in direct line of sight of each other with different heights; free space propagation is assumed.

b. Indoor NLOS: Users are assumed to be in a multi-level office building with heavy blockage, and signals travelling in all directions.

c. Outdoor NLOS: Light blockage is assumed between users while at different heights.

3.2. User Mobility

For each case above, users’ location estimation will be evaluated for static and mobile users. Users were assumed to move according to the random way point model [37]. In random way point model, each user moves independently from each other, each will set a
final location and speed chosen from a minimum-maximum speed range and move toward that location at that speed. Once users reach their location, they stop for a random stop period and then set a new final location and a new speed.

For this research, users chose their speed at random (uniformly distributed) from the range (0.4-1.4 m/s) [38]. The maximum pause time for the users was set at 10 seconds, since it is not desirable for a user to be stopped for the entire simulation.

3.3. System Model

Cooperative localization enables users to find their location based on information sent to them from other users. By filtering and processing each transmitting user’s information, such as user location and signal power, users can easily estimate their location.

Figure 3-2 shows the main algorithm, where users’ locations are used to simulate a signal received by the target user (Figure 3-3), which are then filtered and separated to use in the RSS location estimations. Out of these estimates, one final location estimate is found.

The signal received by the target user is simulated first by calculating the LOS (Line of Sight) distance that the transmitted signal will travel before reaching the target user. Assuming a transmitted signal power of 5W, the received signal power is calculated using the equation:

\[
P_{Rx} = \left(\frac{C}{4\pi f D}\right)^2 P_{Tx}
\]  

(3 - 1)
where $e$ is the received signal power, $c$ is the speed of light, $f$ is the transmitted signal’s carrier frequency, $D$ is the distance between the transmitter and receiver, and $P_{Rx}$ is the transmitted signal power (set here to 5W).

Using the received power, a QPSK (Quadrature Phase Shift Keying) signal is generated for each transmitting user. These signals are then added to simulate one received signal at the target user.

To use RSS for location estimation, the received signal should be divided into separate signals, each coming from a different transmitting user. Filtration of received signals (Figure 3-4) starts by determining the frequency with the highest power. Then the exact carrier frequency is determined. The signal is then entered into two separate filters; a BPF (Band-pass Filter) to set that signal aside, and a BSF (Band-stop Filter) to continue.
processing the signals until all signals are separated. Both the BPF and the BSF used here are Butterworth filters centered around the signal’s carrier frequency and a 3dB bandwidth equal to the bit rate of the QPSK signal.

To determine the exact carrier frequency the signal is operating at, a Costas recovery circuit [39] is used (Figure 3-5). The input signal \( X \) is mixed twice with a sine and cosine signal at \( F_{VCO} \).

\[
X_I = X \ast \sin(2\pi f_{VCO} t) \quad (3 - 2)
\]

Figure 3-5: The Costas Recovery Circuit
Then \(X_Q = X \sin(2\pi f_{VCO} t)\)  

(3 - 3)

Then \(X_I\) and \(X_Q\) are passed through LPFs (Low-pass Filters) resulting in \(I\) and \(Q\) signals, respectively. These \(I\) and \(Q\) signals are then passed through a limiter to get \(\bar{I}\) and \(\bar{Q}\) signals. \(\bar{I}\) and \(\bar{Q}\) signals are then multiplied with the original version of \(I\) and \(Q\) signals to produce \(Y_I\) and \(Y_Q\), such that:

\[Y_I = I \times \bar{Q}\]  

(3 - 4)

\[Y_Q = Q \times \bar{I}\]  

(3 - 5)

\[Y = Y_I - Y_Q\]  

(3 - 6)

\(Y\) is then passed through a LPF to set a frequency for the VCO (Voltage Controlled Oscillator) to enter the loop again. The final detected frequency is \(f_{VCO}\) given that it does not change during the loop.

Location estimation based on RSS is based on the triangulation method, where using the information from three nodes (in two dimensions) or four nodes (in three dimensions) can determine the location of the fourth node. Location estimation based on RSS (Figure 3-6) uses two pieces of information sent from the other users to estimate the user’s location; the location of the sender and the signal power. Senders include their location to the main body of the message.

Location estimation based on RSS uses the basic triangulation equation to calculate the distance between the sender and the receiver:

\[
\frac{(x_1 - x_0)^2 + (y_1 - y_0)^2}{(x_2 - x_0)^2 + (y_2 - y_0)^2} = \frac{D_1^2}{D_2^2} = \left(\frac{A_2}{A_1}\right)^2
\]  

(3 - 6)

where \(x_1, y_1, x_2, y_2\) are the \(x\) and \(y\) coordinates of users 1 and 2, respectively. \(x_0\) and \(y_0\) are the \(x\) and \(y\) coordinates of the unknown user location (Figure 1-1). \(D_1\) and \(D_2\) are the
distances between the user and users 1 and 2. $A_1$ and $A_2$ are the magnitudes of the signals received by the user from users 1 and 2.

From Equation (3-6), two possible locations for the user are found. The information from the third user helps to eliminate one of these ambiguities.

Location estimation based on RSS uses the power of the received signals from the free space propagation model to determine the distance between the transmitter and receiver:
where $P_{Tx}$ is the power of the transmitted signal, $P_{Rx}$ is the power of the received signal, $C$ is the speed of light, and $f$ is the frequency of the signal.

Assuming that the transmitted signal has a power ($P_{Tx}$), the received power ($P_{Rx}$) can be calculated using the following equation:

$$P_{Rx} = \lim_{T \to \infty} \frac{1}{2T} \int_{-\infty}^{\infty} s^2(t) \, dt$$

(3 – 8)

where $s(t)$ is the received signal, $T$ is the period of the received signal.

But since the simulation was done on a discrete and finite signal, the equation used was:

$$P_{Rx} = \frac{1}{N} \sum_{n=1}^{N} s^2(n)$$

(3 – 9)

where $N$ is the length of the received signal stream $s(n)$.

For every combination of three information transmitting users, the target user uses to estimate its location, one estimate is found. As the target user uses information from more information transmitting users, the number of estimates is duplicated following the equation:

$$E = \frac{U!}{3! (U-3)!}$$

$$= \frac{1}{6} U(U-1)(U-2) = \frac{1}{6} U^3 - \frac{1}{2} U^2 + \frac{1}{3} U$$

(3 – 10)
where $E$ is the number of estimates, and $U$ is the number of users the target user processes their information.

To find the true estimate for the target user’s location, the extreme estimates found so far are filtered out since they represent the extreme signal degradation that happened to the signals received by the target user.
Chapter 4

Simulation and Results

Simulation results are split into two parts. First, the case-by-case results are presented in section 4.1, which are specific to the cases being studied. Second, general results are presented in section 4.2 to show the effects of some common parameters between different cases and to highlight the work done.

4.1. Case-by-case Results

All cases considered can be initially classified based on the propagation environment into indoor and outdoor LOS, indoor NLOS, and outdoor NLOS. Secondly, all of the cases can be divided into 2D and 3D cases. Thirdly, cases can also be divided based on the users’ movement into static and mobile users. Finally, by varying the number of information transmitting users processed, the density of the users, and the number of spoofing attackers, these results represent the specific case being studied (Table 4.1).

These results commonly show that by processing information from seven or more transmitting users, the proposed algorithm was able to reduce the errors in the location
estimation of the target user as well as minimizing the effect of any spoofing users. The following specifies the case-by-case results:

Table 4.1: The Simulation Cases

<table>
<thead>
<tr>
<th></th>
<th>Static Users</th>
<th>Mobile Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indoor and Outdoor LOS</td>
<td>• No spoofing attack (Figures 4.1-4.5)</td>
<td>• No spoofing attack (Figures 4.6-4.8)</td>
</tr>
<tr>
<td>2D</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figures 4.2 and 4.5)</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-7)</td>
</tr>
<tr>
<td></td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figures 4.3 and 4.5)</td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-8)</td>
</tr>
<tr>
<td>3D</td>
<td>• No spoofing attack (Figures 4.9 and 4.10)</td>
<td>• No spoofing attack (Figures 4.11-4.13)</td>
</tr>
<tr>
<td></td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-9)</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-12)</td>
</tr>
<tr>
<td></td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-10)</td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-13)</td>
</tr>
<tr>
<td>Indoor NLOS</td>
<td>• No spoofing attack (Figures 4.14 and 4.15)</td>
<td>• No spoofing attack (Figures 4.16 and 4.17)</td>
</tr>
<tr>
<td>2D</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-14)</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-16)</td>
</tr>
<tr>
<td></td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-15)</td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-17)</td>
</tr>
<tr>
<td>3D</td>
<td>• No spoofing attack (Figures 4.18-4.21)</td>
<td>• No spoofing attack (Figures 4.16 and 4.17)</td>
</tr>
<tr>
<td></td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figures 4.18 and 4.20)</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-16)</td>
</tr>
<tr>
<td></td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figures 4.19 and 4.21)</td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-17)</td>
</tr>
<tr>
<td>Outdoor NLOS</td>
<td>• No spoofing attack (Figures 4.26 and 4.27)</td>
<td>• No spoofing attack (Figures 4.28 and 4.29)</td>
</tr>
<tr>
<td>2D</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-26)</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-28)</td>
</tr>
<tr>
<td></td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-27)</td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-29)</td>
</tr>
<tr>
<td>3D</td>
<td>• No spoofing attack (Figures 4.30 and 4.31)</td>
<td>• No spoofing attack (Figures 4.32 and 4.33)</td>
</tr>
<tr>
<td></td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-30)</td>
<td>• 1, 2, and 3 spoofing users with 20 m spoofing error attack (Figure 4-32)</td>
</tr>
<tr>
<td></td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-31)</td>
<td>• 2 spoofing users with 20, 40, and 60 m spoofing error attack (Figure 4-33)</td>
</tr>
</tbody>
</table>
4.1.1. The Indoor and Outdoor LOS Case

In the indoor and outdoor LOS case, different conditions were varied to test the algorithm in various scenarios. The number of information transmitting users processed, the density of the total number of users, the number of spoofing attackers, and the severity of the spoofing attack (the distance between the real location and the false reported location of the spoofing user shown in Figure 2-2) were all varied:

4.1.1.1. 2D Case in a LOS Environment

a. 2D Case for Static Users in a LOS Environment

Since different locations can have different densities of users, the cooperative localization process would produce different accuracies. Testing for different densities resulted in Figure 4-1, which shows improvement in accuracy when increasing the user density per km$^2$; since the added density would mean a decrease in the distance between

![Figure 4-1: The Error in Location Estimation Resulting from Different User Densities in a 2D LOS environment.](image-url)
users and ultimately increase the quality of the received signal.

Under spoofing attack, the cooperative localization process performs very well for six or more processed information users (Figures 4-2 - 4-5). Figures 4-2 and 4-3 show the error in the location estimation for a density of 500 users/km$^2$ while the error produced from a density of 2000 users/km$^2$ was shown in Figures 4-4 and 4-5. These results commonly show that regardless of the number of spoofing users and the considered user density, processing information from more transmitting users can reduce the location estimation error.

![Figure 4-2: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D LOS environment for a Density of 500 users/ km$^2$.](image-url)
Figure 4-3: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D LOS Environment for a Density of 500 users/km².

Figure 4-4: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D LOS Environment for a Density of 2000 users/km².
b. 2D Case for Mobile Users in a LOS Environment

Testing for different densities for mobile users (moving according to section 3.2) with Kalman filter shows that the change in densities affects the location estimation error in a minimal way (Figure 4-6). Both Figures 4-1 and 4-6 show an average error of around 5 m with a slight advantage to the mobile users’ case, since the use of the Kalman filter aided in the reduction of the location estimation error.

Figure 4-5: The Error in Location Estimation Resulting from Different Attack Errors from Two Spoofing Users in a 2D LOS Environment for a Density of 2000 users/ km².
For mobile users in a 2D LOS environment using Kalman filter, processing information from 6 or more transmitting users minimized the effect of any attack (Figures 4-7 and 4-8). The high location estimation errors in Figures 4-7 and 4-8 while processing information from five or less transmitting users are due to the low overall number of estimates produced, which lead to a high percentage of estimates being affected by the spoofing attack thus producing a high location estimation error.

The use of the Kalman filter reduced the error in the location estimation (Table 4.2) at an average of 0.5m for the most part compared to the case where the Kalman filter was not used. Since the Kalman filter depends heavily on the observation values, processing information from a low number of transmitting users would lead to a bad observation which would in turn lead to a bad Kalman filter estimate. These bad Kalman estimates would actually be worse than the observation values which would result in the negative values in Table 4.2.

Figure 4-6: The Error in Location Estimation Resulting from Different User Densities in a 2D LOS Environment with a Kalman Filter.
Figure 4-7: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D LOS Environment with a Kalman Filter.

Figure 4-8: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D LOS Environment with a Kalman Filter.
4.1.1.2. 3D Case in a LOS Environment

a. 3D Case for Static Users in a LOS Environment

Although testing in 3D adds the element of altitude, the results (Figures 4-9 and 4-10) show the error in the barometer’s reading affects the error in the location estimation in a minimal way compared with the 2D case.

Assuming an error of $\varepsilon$ from the users’ barometric sensors, equation (2-36) for calculating the altitude can be written as

$$h_u = \frac{T_u R}{mg} (\ln P_0 - \ln(P_u + \varepsilon))$$  \hspace{1cm} (4 - 1)

The only part affected by the error in the barometric sensor is the last part of the equation. This part can be rewritten as

Table 4.2: The Average Reduction in Location Estimation Error (m) made by a Kalman Filter in a 2D LOS Environment.

<table>
<thead>
<tr>
<th>Number of Information Users</th>
<th>Average Location Estimation Error</th>
<th>Average Reduction in Location Estimation Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o Kalman Filter (No Spoofing)</td>
<td>with Kalman Filter (No Spoofing)</td>
<td>No Spoofing</td>
</tr>
<tr>
<td>3</td>
<td>14.54268</td>
<td>13.5673</td>
</tr>
<tr>
<td>4</td>
<td>4.21334</td>
<td>3.77656</td>
</tr>
<tr>
<td>5</td>
<td>3.71672</td>
<td>3.42318</td>
</tr>
<tr>
<td>6</td>
<td>3.28156</td>
<td>2.88924</td>
</tr>
<tr>
<td>7</td>
<td>3.04654</td>
<td>2.51754</td>
</tr>
<tr>
<td>8</td>
<td>2.86814</td>
<td>2.40024</td>
</tr>
<tr>
<td>9</td>
<td>2.65432</td>
<td>2.22552</td>
</tr>
<tr>
<td>10</td>
<td>2.43274</td>
<td>1.97862</td>
</tr>
<tr>
<td>11</td>
<td>2.18562</td>
<td>1.61608</td>
</tr>
<tr>
<td>12</td>
<td>1.70852</td>
<td>1.2318</td>
</tr>
</tbody>
</table>
\[ \ln(P_u + \varepsilon) = \ln(P_u + \varepsilon) - \ln(P_u) + \ln(P_u) \]
\[ = \ln \left( \frac{P_u + \varepsilon}{P_u} \right) + \ln(P_u) \]
\[ = \ln(1 + \frac{\varepsilon}{P_u}) + \ln(P_u) \quad (4 - 2) \]

Since \( \varepsilon \) is around 12 Pascals and \( P_u \sim P_0 = 101325 \) Pascals, the ratio \( \frac{\varepsilon}{P_u} \) can be approximated to \( 10^{-4} \) which makes the first part of equation (4-2) approximately zero.

Comparing Figures 4-9 and 4-10 with Figures 4-2 and 4-3, although the four Figures show the same trend of reducing the location estimation error when processing information from a higher number of transmitting users, the average location estimation error in the 3D case is slightly higher than the 2D case because of the errors introduced by the third dimension calculation of the user’s location.

![Figure 4-9: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D LOS Environment.](image)
The change in the user density for the 3D LOS environment with mobile users without spoofing had minimal effect on the error in the location estimation when processing information from six or more transmitting users using Kalman filter (Figure 4-11). But when dealing with spoofing attacks (with a population of 2000 users/km$^2$), the Kalman filter worked effectively when processing information from seven or more transmitting users (Figures 4-12 and 4-13).

The effect of the Kalman filter under spoofing attack is shown in Table 4.3, where the reduction in the error of the location estimation was around the same for the 2D LOS case of 0.5m for good estimates and higher for bad estimates.
Figure 4-11: The Error in Location Estimation Resulting from Different User Densities in a 3D LOS Environment with a Kalman Filter.

Figure 4-12: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D LOS Environment with a Kalman Filter.
Table 4.3: The Average Reduction in Location Estimation Error (m) made by a Kalman Filter in a 3D LOS Environment.

<table>
<thead>
<tr>
<th>Number of Information Users</th>
<th>No Spoofing</th>
<th>20m Spoofing Error Attack</th>
<th>2 Spoofing Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3.135433</td>
<td>-2.0478</td>
<td>-8.00305</td>
</tr>
<tr>
<td>4</td>
<td>0.0312</td>
<td>0.616975</td>
<td>1.449325</td>
</tr>
<tr>
<td>5</td>
<td>1.098033</td>
<td>0.8988</td>
<td>0.73855</td>
</tr>
<tr>
<td>6</td>
<td>0.4873</td>
<td>0.821525</td>
<td>1.0019</td>
</tr>
<tr>
<td>7</td>
<td>0.4117</td>
<td>0.376725</td>
<td>0.417425</td>
</tr>
<tr>
<td>8</td>
<td>0.432467</td>
<td>0.4555</td>
<td>0.410325</td>
</tr>
<tr>
<td>9</td>
<td>0.423133</td>
<td>0.31</td>
<td>0.378825</td>
</tr>
<tr>
<td>10</td>
<td>0.460667</td>
<td>0.41795</td>
<td>0.418275</td>
</tr>
<tr>
<td>11</td>
<td>0.399333</td>
<td>0.4785</td>
<td>0.43185</td>
</tr>
<tr>
<td>12</td>
<td>0.441</td>
<td>0.43785</td>
<td>0.542875</td>
</tr>
</tbody>
</table>

Figure 4-13: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D LOS Environment with Kalman Filter.
4.1.2. The Indoor NLOS Case

In the indoor NLOS case, the number of information transmitting users processed, the density of the total number of users, the number of spoofing attackers, and the severity of the spoofing attack were all varied to simulate the different scenarios of a typical office building:

4.1.2.1. 2D Case in an Indoor NLOS Environment

a. 2D Case for Static Users in an Indoor NLOS Environment

Although in a 2D indoor NLOS environment cooperative localization does not perform as good as in the 2D LOS environment (Figures 4-14 and 4-15), the trend of minimizing the effect of spoofing attack when processing information from seven or more

![Figure 4-14](image_url)

Figure 4-14: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Indoor Multipath and NLOS Environment.
transmitting users still maintains. The indoor NLOS location estimation error is higher (12m) compared to LOS environment, since the presence of reflective and absorbent surfaces degrade the received signal.

b. 2D Case for Mobile Users in an Indoor NLOS Environment

The effects of spoofing attacks in a 2D mobile indoor NLOS setting can be minimized by processing information from seven or more transmitting users (Figures 4-16 and 4-17).

The improvement of using the Kalman filter in reducing the error in the location estimation can be seen in Table 4.4, where the Kalman filter reduces the error in location estimation by 0.8m compared to 0.5m in the LOS case (Table 4.2), since the error in location estimation is much worse in the indoor NLOS case (Figures 4-16 and 4-17).
Figure 4-16: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Indoor Multipath and NLOS Environment with a Kalman Filter.

Figure 4-17: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D Indoor Multipath and NLOS Environment with a Kalman Filter.
Table 4.4: The Average Reduction in Location Estimation Error (m) made by a Kalman Filter in a 2D Indoor NLOS Environment.

<table>
<thead>
<tr>
<th>Number of Information Users</th>
<th>No Spoofing</th>
<th>20m Spoofing Error Attack</th>
<th>2 Spoofing Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1.9009</td>
<td>0.99115</td>
<td>0.103725</td>
</tr>
<tr>
<td>4</td>
<td>0.4997</td>
<td>1.5289</td>
<td>2.196425</td>
</tr>
<tr>
<td>5</td>
<td>0.1149</td>
<td>0.44675</td>
<td>1.7454</td>
</tr>
<tr>
<td>6</td>
<td>0.6404</td>
<td>0.94505</td>
<td>0.81595</td>
</tr>
<tr>
<td>7</td>
<td>0.6536</td>
<td>0.86045</td>
<td>0.8687</td>
</tr>
<tr>
<td>8</td>
<td>0.8386</td>
<td>1.141325</td>
<td>0.8066</td>
</tr>
<tr>
<td>9</td>
<td>0.5899</td>
<td>0.842875</td>
<td>0.727225</td>
</tr>
<tr>
<td>10</td>
<td>0.4637</td>
<td>0.523758</td>
<td>0.717171</td>
</tr>
<tr>
<td>11</td>
<td>0.5185</td>
<td>0.574425</td>
<td>0.5202</td>
</tr>
<tr>
<td>12</td>
<td>0.4468</td>
<td>0.556175</td>
<td>0.560325</td>
</tr>
</tbody>
</table>

4.1.2.2. 3D Case in an Indoor NLOS Environment

a. 3D Case for Static Users in an Indoor NLOS Environment

By testing in a 3D indoor NLOS environment with different user densities, the error in location estimation is worse in the case of 2000 users/km² (Figures 4-18 and 4-19) compared to the case of 8000 users/km² (Figures 4-20 and 4-21), since the increase in the user density would lead to the decrease in the location estimation error.
Figure 4-18: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 2000 users/km².

Figure 4-19: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 2000 users/km².
Figure 4-20: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 8000 users/ km².

Figure 4-21: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment for a Density of 8000 users/ km².
b. 3D Case for Mobile Users in an Indoor NLOS Environment

In a 3D indoor NLOS environment, testing with different user densities using a Kalman filter for mobile users, the error in location estimation is worse in the case of 2000 users/km² (Figures 4-22 and 4-23) compared to the case of 8000 users/km² (Figures 4-24 and 4-25), since the increase in the user density would lead to an decrease the location estimation error. The Kalman filter used to correct the errors in the location estimation reduced the error by an average of 1.2 m (Table 4.5).

![Figure 4-22: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 2000 users/ km².](image-url)
Figure 4-23: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 2000 users/ km$^2$.

Figure 4-24: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 8000 users/ km$^2$. 

58
Table 4.5: The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 3D Indoor NLOS Environment.

<table>
<thead>
<tr>
<th>Number of Information Users</th>
<th>No Spoofing</th>
<th>20m Spoofing Error Attack</th>
<th>2 Spoofing Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>-1.58455</td>
<td>-1.5174</td>
<td>3.443991</td>
</tr>
<tr>
<td>4</td>
<td>3.09685</td>
<td>1.116513</td>
<td>0.900983</td>
</tr>
<tr>
<td>5</td>
<td>0.54735</td>
<td>1.101625</td>
<td>1.761063</td>
</tr>
<tr>
<td>6</td>
<td>1.16105</td>
<td>1.16884</td>
<td>1.4069</td>
</tr>
<tr>
<td>7</td>
<td>0.87855</td>
<td>1.263913</td>
<td>1.259513</td>
</tr>
<tr>
<td>8</td>
<td>0.9191</td>
<td>1.151725</td>
<td>1.22425</td>
</tr>
<tr>
<td>9</td>
<td>0.70505</td>
<td>1.06461</td>
<td>1.077973</td>
</tr>
<tr>
<td>10</td>
<td>1.13665</td>
<td>1.028275</td>
<td>0.912663</td>
</tr>
<tr>
<td>11</td>
<td>0.88585</td>
<td>1.009526</td>
<td>1.109663</td>
</tr>
<tr>
<td>12</td>
<td>0.3628</td>
<td>0.8142</td>
<td>0.89</td>
</tr>
</tbody>
</table>

Figure 4-25: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Indoor Multipath and NLOS Environment with a Kalman Filter for a Density of 8000 users/ km².

Table 4.5: The Average Error Correction in Location Estimation (m) made by a Kalman Filter in a 3D Indoor NLOS Environment.
4.1.3. The Outdoor NLOS Case

For the outdoor NLOS case, the number of information transmitting users processed, the number of spoofing attackers, and the severity of the spoofing attack were all varied to simulate a busy city environment:

4.1.3.1. 2D Case in an Outdoor NLOS Environment

a. 2D Case for Static Users in an Outdoor NLOS Environment

Figures 4-26 and 4-27 show an improvement in the error in the location estimation in a 2D outdoor NLOS case compared to the 2D indoor NLOS case. Both Figures show that to effectively minimize the effect of a spoofing attack, it is important to set the number of processed information transmitting users to seven or more, since processing a lower

![Graph showing error in location estimation](image)

Figure 4-26: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Outdoor Multipath and NLOS Environment.
number of information transmitting users would lead higher location estimation errors (Figures 4-26 and 4-27).

b. 2D Case for Mobile Users in an Outdoor NLOS Environment

In the 2D outdoor NLOS environment, the proposed method was able to minimize the effect of the spoofing attacks when processing information from six or more transmitting users (Figures 4-28 and 4-29).

The average improvement the Kalman filter produces is 1 m (Table 4.6), with it being a little less for very good location estimates, since the Kalman filter depends on the observations to find the true location estimate, the Kalman filter would need to correct less with the presence of good observations.
Figure 4-28: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 2D Outdoor Multipath and NLOS Environment with a Kalman Filter.

Figure 4-29: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 2D Outdoor Multipath and NLOS Environment with a Kalman Filter.
Table 4.6: The Average Reduction in Location Estimation Error (m) made by a Kalman Filter in a 2D Outdoor NLOS Environment.

<table>
<thead>
<tr>
<th>Number of Information Users</th>
<th>No Spoofing</th>
<th>20m Spoofing Error Attack</th>
<th>2 Spoofing Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>-0.3792</td>
<td>-14.73505</td>
<td>-30.0086</td>
</tr>
<tr>
<td>4</td>
<td>2.467</td>
<td>1.008425</td>
<td>-0.69385</td>
</tr>
<tr>
<td>5</td>
<td>-0.1019</td>
<td>0.7198</td>
<td>0.3075</td>
</tr>
<tr>
<td>6</td>
<td>0.5232</td>
<td>0.5553</td>
<td>0.73465</td>
</tr>
<tr>
<td>7</td>
<td>0.6986</td>
<td>0.6547</td>
<td>1.001675</td>
</tr>
<tr>
<td>8</td>
<td>0.7025</td>
<td>0.732425</td>
<td>0.696475</td>
</tr>
<tr>
<td>9</td>
<td>0.6892</td>
<td>0.62295</td>
<td>0.88655</td>
</tr>
<tr>
<td>10</td>
<td>0.7139</td>
<td>0.614125</td>
<td>0.690625</td>
</tr>
<tr>
<td>11</td>
<td>0.6439</td>
<td>0.72305</td>
<td>1.003747</td>
</tr>
<tr>
<td>12</td>
<td>0.9153</td>
<td>0.866225</td>
<td>0.9605</td>
</tr>
</tbody>
</table>

4.1.3.2. 3D Case in an Outdoor NLOS Environment

a. 3D Case for Static Users in an Outdoor NLOS Environment

In the 3D outdoor NLOS environment, the proposed method was able to get good location estimates when processing information from six or more transmitting users under normal condition, where no spoofing attack occurred (Figures 4-30 and 4-31). But when an attack occurs, the proposed method minimized the effect of those attacks when processing information from eight or more transmitting users. That is because the proposed method needed an extra information source to deal with both the errors produced from the NLOS environment, and the errors introduced by the third dimension calculation of the user’s location.
Figure 4-30: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Outdoor Multipath and NLOS Environment.

Figure 4-31: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Outdoor Multipath and NLOS Environment.
b. 3D Case for Mobile Users in an Outdoor NLOS Environment

Although the 3D outdoor NLOS environment for both static and mobile users is the same, the use of the Kalman filter helped in reducing both the error in the location estimation and the number of information transmitting users needed to be processed (Figures 4-32 and 4-33).

The difference in location estimation error in Table 4.7 was around 0.8 m improvement when using the Kalman filter.

![Figure 4-32: The Error in Location Estimation Resulting from 20 m Error Attack from Different Number of Spoofing Users in a 3D Outdoor Multipath and NLOS Environment with a Kalman Filter.](image)
Figure 4-33: The Error in Location Estimation Resulting from Different Errors from Two Spoofing Users in a 3D Outdoor Multipath and NLOS Environment with a Kalman Filter.

Table 4.7: The Average Reduction in Location Estimation Error (m) made by a Kalman Filter in a 3D Outdoor NLOS Environment.

<table>
<thead>
<tr>
<th>Number of Information Users</th>
<th>No Spoofing</th>
<th>20m Spoofing Error Attack</th>
<th>2 Spoofing Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.3055</td>
<td>-5.82215</td>
<td>-11.96245</td>
</tr>
<tr>
<td>4</td>
<td>0.7428</td>
<td>1.33815</td>
<td>1.9404</td>
</tr>
<tr>
<td>5</td>
<td>0.4489</td>
<td>0.635325</td>
<td>1.028775</td>
</tr>
<tr>
<td>6</td>
<td>0.7824</td>
<td>0.85605</td>
<td>1.49465</td>
</tr>
<tr>
<td>7</td>
<td>0.7174</td>
<td>1.05445</td>
<td>0.6777</td>
</tr>
<tr>
<td>8</td>
<td>0.4808</td>
<td>0.671175</td>
<td>0.813325</td>
</tr>
<tr>
<td>9</td>
<td>0.426</td>
<td>0.601</td>
<td>0.83265</td>
</tr>
<tr>
<td>10</td>
<td>0.6426</td>
<td>0.783275</td>
<td>0.77005</td>
</tr>
<tr>
<td>11</td>
<td>0.5746</td>
<td>0.6551</td>
<td>0.67265</td>
</tr>
<tr>
<td>12</td>
<td>0.4222</td>
<td>0.51665</td>
<td>0.607475</td>
</tr>
</tbody>
</table>
4.2. General Results

The main algorithm and the mathematical background for this research are the same for all the cases presented above. For that reason, all the Figures show the same pattern, where using information from more than six transmitting users for location estimation has minimum error compared to using less than six users for location estimation. In addition, when using information from three transmitting users for location estimation, the localization algorithm sometimes returns with no results (Table 4.8), since processing information from only three transmitting users will produce only one estimate and if any two circles (Figure 1-1) do not intersect the localization algorithm returns with no results.

Although using information from a high number of transmitting users for location estimation reduces the error and the chances of the program returning no results, the high run-time (Table 4.9) for processing the information from that high number of transmitting users does not warrant their use.

Comparing the proposed method to the MMSE method detailed in [27, 28], the proposed method yielded a consistent advantage over the MMSE method in all the cases (Figures 4-34 - 4-39). For the case with no attacks in a 2D LOS environment, the proposed

<table>
<thead>
<tr>
<th>Number of information transmitting users processed</th>
<th>Outage percentage (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>18</td>
</tr>
<tr>
<td>4</td>
<td>2.5</td>
</tr>
<tr>
<td>&gt; 5</td>
<td>&lt; 1</td>
</tr>
</tbody>
</table>

Table 4.8: The percentage the localization method returns with no results.
method was able to produce a location estimate with an error of 5 m less than the MMSE method (Figure 4-34). While the improvement was less when faced with a spoofing attack, the proposed method was still able to produce a location estimate with an average error less than the MMSE method (Figures 4-35 and 4-36).

Although different user environments produce different results, the common element when testing in three different 3D environments using with mobile users (Figures 4-37 – 4-39) is that the MMSE fails to produce results close to the results obtained from the proposed method. Since the MMSE method in [27, 28] did not expand to the 3D cases, Equation (2-43) was utilized to facilitate the comparison between the proposed method and the MMSE method in 3D cases. Testing the different environments after adding spoofing users would not make sense because to the large difference in the performance between the proposed method the MMSE method.

Table 4.9: The average run-time for the cooperative localization method to return with a location estimation.

<table>
<thead>
<tr>
<th>Number of information transmitting users processed</th>
<th>Average location estimation run time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>7.1</td>
</tr>
<tr>
<td>4</td>
<td>9.3</td>
</tr>
<tr>
<td>5</td>
<td>11.7</td>
</tr>
<tr>
<td>6</td>
<td>14.1</td>
</tr>
<tr>
<td>7</td>
<td>16.5</td>
</tr>
<tr>
<td>8</td>
<td>18.6</td>
</tr>
<tr>
<td>9</td>
<td>21.1</td>
</tr>
<tr>
<td>10</td>
<td>23.4</td>
</tr>
<tr>
<td>11</td>
<td>25.8</td>
</tr>
<tr>
<td>12</td>
<td>29.9</td>
</tr>
</tbody>
</table>
Figure 4-34: Comparing the Error in Location Estimation in a LOS Environment between the Proposed Method and the MMSE Method with no spoofing attacks.

Figure 4-35: Comparing the Error in Location Estimation Resulting from 20 m Error Attack from Three Spoofing Users a LOS Environment between the Proposed Method and the MMSE Method.
Figure 4-36: Comparing the Error in Location Estimation Resulting from 60 m Error Attack from Two Spoofing Users in a LOS Environment between the Proposed Method and the MMSE Method.

Figure 4-37: Comparing the Error in Location Estimation in a 3D LOS Environment with mobile users between the Proposed Method and the MMSE Method with no spoofing attacks.
Figure 4-38: Comparing the Error in Location Estimation in a 3D Indoor NLOS Environment with mobile users between the Proposed Method and the MMSE Method with no spoofing attacks.

Figure 4-39: Comparing the Error in Location Estimation in a 3D Outdoor NLOS Environment with mobile users between the Proposed Method and the MMSE Method with no spoofing attacks.
Chapter 5

Conclusion and Future Work

Cooperative localization in cellular networks provides cellular network users with location information with higher accuracy than GPS (for outdoor cases) or reliable location information (for indoor cases). Many cooperative localization researches are focused on WSNs, which are not mobile and have a short range.

Unfortunately, every human made electronic system is under attack in one way or the other, from computer viruses and website phishing to IP, email, and GPS spoofing. Cooperative localization systems are no exception; spoofing attacks on cooperative localization users, although not serious these days, will be a serious problem for systems that implement cooperative localization in the future.

Spoofing users in a cellular network implementing cooperative localization affect the users’ ability to estimate their true location by sending false information to the users. The most effective way, tested here, is sending wrong location information, which made the users think they are at a different location.

Anti-spoofing cooperative localization for cellular network users is possible by the use of cooperative localization and MLE, by eliminating the extreme estimates produced using
the corrupt information from the spoofing users. The algorithm proposed here was able to produce location estimation with acceptable average errors both in 2D and 3D cases.

Processing information from extra location transmitting users (7 transmitting users or more) reduced the effect of spoofing attacks. The proposed method produced an estimate of the user’s location with a run-time of 16 seconds for 7 transmitting users and 29 seconds for 12 transmitting users. However, target users processing information from 6 or less transmitting users are not able to eliminate the effect of the spoofing attack.

Processing information from multiple transmitting users was implemented in a different way in [27, 28]. Using the MMSE method in an iterative way, the algorithm tried to identify the spoofing users and eliminate any location information from those users. The MMSE method failed to produce the accuracy the proposed method provided in different 2D and 3D simulation environments.

More work needs to be done in this field to reduce the run-time of the location estimation program since the run-time of the program relates to the power consumption of the devices. One way is to implement parallel programming, since the process of producing each location estimate is independent from the process of producing the other location estimates.
References


[34] BMP280 Digital barometric pressure sensor, Bosch Sensortec GmbH, Reutlingen, Germany, 2012


Appendix A

The Derivation of $f_{e_{x_0}} (e_{x_0})$

The following are the detailed equations derived from Equations (2-11, 2-12) to find the pdf of the location estimation error $e_{x_0}$

$$f_{e_{x_0}, e_{y_0}} (e_{x_0}, e_{y_0}) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \left( \psi_1^2 - \left( x_1 + L_1 - (x_0 + e_{x_0}) \right)^2 \right) \psi_2^2 - \left( y_1 + L_1 - (y_0 + e_{y_0}) \right)^2 \right)$$

$$= \sum_{k=1}^{20} I_k$$

Note that the following variables are heavily referenced in this appendix

$L_1$: is the upper limit of the integration

$L_2$: is the lower limit of the integration

$u(.)$: is the unit step function

$$\psi_1 = \sqrt{D_t^2 - \left( x_1 + L_1 - (x_0 + e_{x_0}) \right)^2} - \sqrt{D_t^2 - \left( x_1 + L_2 - (x_0 + e_{x_0}) \right)^2}$$

$$\psi_2 = \sqrt{D_t^2 - \left( y_1 + L_1 - (y_0 + e_{y_0}) \right)^2} - \sqrt{D_t^2 - \left( y_1 + L_2 - (y_0 + e_{y_0}) \right)^2}$$
\[ \psi_3 = \sqrt{D_x^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} - \sqrt{D_x^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} \]

\[ \psi_4 = 2 \cdot L_2 \cdot \{1 - u[(x_0 + e_{x_0}) - (x_3 + L_2)]\} + 2\{(x_0 + e_{x_0}) - x_3\} \cdot \{u[(x_0 + e_{x_0}) - (x_3 + L_2)] - u[(x_0 + e_{x_0}) - (x_3 + L_1)]\} + 2 \cdot L_1 \cdot u[(x_0 + e_{x_0}) - (x_3 + L_1)] \]

\[ \psi_5 = \sqrt{D_x^2 - (x_1 + L_1 - (x_0 + e_{x_0}))^2} - \sqrt{D_x^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2} \]

\[ \psi_6 = \sqrt{D_x^2 - (y_1 + L_1 - (y_0 + e_{y_0}))^2} - \sqrt{D_x^2 - (y_1 + L_2 - (y_0 + e_{y_0}))^2} \]

\[ \psi_7 = \sqrt{D_x^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} - \sqrt{D_x^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2} \]

\[ \psi_8 = 2 \cdot L_2 \cdot \{1 - u[(x_0 + e_{x_0}) - (x_2 + L_2)]\} + 2\{(x_0 + e_{x_0}) - x_2\} \cdot \{u[(x_0 + e_{x_0}) - (x_2 + L_2)] - u[(x_0 + e_{x_0}) - (x_2 + L_1)]\} + 2 \cdot L_1 \cdot u[(x_0 + e_{x_0}) - (x_2 + L_1)] \]

\[ I_1 = \frac{-1}{8\pi^2 \sigma^2 (\sqrt{1-r^2})^2} \cdot \psi_1 \cdot \psi_2 \cdot \psi_3 \cdot \psi_4 \]

\[ I_2 = \frac{1}{4 \pi^2 \sigma^2 (\sqrt{1-r^2})^2} \cdot \psi_1 \cdot \psi_3 \cdot \psi_4 \cdot \left\{ 3 \cdot ((x_0 + e_{x_0}) - x_1)^2 \cdot \sqrt{D_x^2 - (y_1 + L_1 - (y_0 + e_{y_0}))^2} + \left( \sqrt{D_x^2 - (y_1 + L_1 - (y_0 + e_{y_0}))^2} \right)^3 - 3 \cdot ((x_0 + e_{x_0}) - x_1)^2 \cdot \left( L_1 + y_1 - (y_0 + e_{y_0}) \right)^2 \right\} - 3 \cdot ((x_0 + e_{x_0}) - x_1)^2 \cdot \sqrt{D_x^2 - (y_1 + L_2 - (y_0 + e_{y_0}))^2} + \left( \sqrt{D_x^2 - (y_1 + L_2 - (y_0 + e_{y_0}))^2} \right)^3 - 3 \cdot ((x_0 + e_{x_0}) - x_1)^2 \cdot \left( L_2 + y_1 - (y_0 + e_{y_0}) \right)^2 \right\} \]
\[ I = \frac{1}{16a^2 \sigma^2 (\sqrt{1 - \rho^2})} * \psi_1 * \psi_2 * \psi_3 * \{ 1 - u[(x_0 + e_{x_0}) - (x_2 + L_2)] \} \] * \\
\left\{ \frac{1}{3} * \sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} \right\} \left\{ 2 * D_2^2 + L_1^2 + L_1 * ((x_0 + e_{x_0}) - x_2) + ((x_0 + e_{x_0}) - x_2)^2 \right\} + D_2^2 * \\
((x_0 + e_{x_0}) - x_2) * \tan^{-1}\left( \frac{x_2 + L_1 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}} \right) - \left\{ \frac{1}{3} * \sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} \right\} \left\{ 2 * D_2^2 + L_2^2 + L_2 * ((x_0 + e_{x_0}) - x_2) + ((x_0 + e_{x_0}) - x_2)^2 \right\} + D_2^2 * \\
L_2^2 + L_2 * ((x_0 + e_{x_0}) - x_2) + ((x_0 + e_{x_0}) - x_2)^2 \right\} + D_2^2 * ((x_0 + e_{x_0}) - x_2) * \\
\tan^{-1}\left( \frac{x_2 + L_2 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}} \right) \right\} + \frac{1}{16a^2 \sigma^2 (\sqrt{1 - \rho^2})} * \psi_1 * \psi_2 * \psi_3 * \{ u[(x_0 + e_{x_0}) - (x_2 + L_1)] \} * \\
\left\{ \frac{1}{3} * \sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} \right\} \left\{ 2 * D_2^2 + L_1^2 + 5 * L_1 * ((x_0 + e_{x_0}) - x_2) + 7 * ((x_0 + e_{x_0}) - x_2)^2 \right\} \right\} - D_2^2 * ((x_0 + e_{x_0}) - x_2) * \\
\tan^{-1}\left( \frac{x_2 + L_1 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}} \right) \right\} + \frac{1}{16a^2 \sigma^2 (\sqrt{1 - \rho^2})} * \psi_1 * \psi_2 * \psi_3 * \{ u[(x_0 + e_{x_0}) - (x_2 + L_2)] \} * \\
u[(x_0 + e_{x_0}) - (x_2 + L_1)] * \left\{ \frac{1}{3} * D_2 * (2 * D_2^2 + ((x_0 + e_{x_0}) - x_2)^2) - 5 * ((x_0 + e_{x_0}) - x_2) * ((x_0 + e_{x_0}) - x_2) + 7 * ((x_0 + e_{x_0}) - x_2)^2 \right\} - \left\{ \frac{1}{3} * \sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} \right\} \left\{ 2 * D_2^2 + L_2^2 + 5 * L_2 * ((x_0 + e_{x_0}) - x_2) + 7 * ((x_0 + e_{x_0}) - x_2)^2 \right\} \right\} - D_2^2 * ((x_0 + e_{x_0}) - x_2) * \\
\tan^{-1}\left( \frac{x_2 + L_2 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}} \right) \right\} + \frac{1}{16a^2 \sigma^2 (\sqrt{1 - \rho^2})} * \psi_1 * \psi_2 * \psi_3 * \{ u[(x_0 + e_{x_0}) - (x_2 + L_2)] \} * \\
u[(x_0 + e_{x_0}) - (x_2 + L_1)] * \left\{ \frac{1}{3} * \sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} \right\} \left\{ 2 * D_2^2 + L_1^2 + L_1 * ((x_0 + e_{x_0}) - x_2) + 7 * ((x_0 + e_{x_0}) - x_2)^2 \right\} \right\} - D_2^2 * ((x_0 + e_{x_0}) - x_2) * \\
((x_0 + e_{x_0}) - x_2)^2 + D_2^2 * ((x_0 + e_{x_0}) - x_2) * \tan^{-1}\left( \frac{x_2 + L_1 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}} \right) \right\} - \left\{ \frac{1}{3} * D_2 * (2 * D_2^2 + \right\} * \\
((x_0 + e_{x_0}) - x_2)^2 + ((x_0 + e_{x_0}) - x_2) * ((x_0 + e_{x_0}) - x_2) + ((x_0 + e_{x_0}) - x_2)^2 \right\)} \\
80
\[ I_4 = \frac{-1}{16\pi^3\sigma^{(1-\rho^2)}} \left( \psi_1 \times \psi_2 \times \psi_3 \times \left\{ 1 - u((x_0 + e_{x_0}) - (x_3 + L_2)) \right\} \times \left( \frac{-1}{3} \times L_1^3 - \left( \frac{-1}{3} \times L_2^3 \right) \right) + \right. \\
\left. \frac{-1}{16\pi^3\sigma^{(1-\rho^2)}} \left( \psi_1 \times \psi_2 \times \psi_3 \times \left\{ u((x_0 + e_{x_0}) - (x_3 + L_1)) \right\} \times \left( 4 \times (x_0 + e_{x_0})^2 \times L_1 - 8 \times (x_0 + e_{x_0}) \right) \times x_3 \times L_1 - 2 \times (x_0 + e_{x_0}) \times L_1^2 + 4 \times x_3^2 \times L_1 + 2 \times L_1^2 \times x_3 + \frac{1}{3} \times L_1^3 - \left( 4 \times (x_0 + e_{x_0})^2 \times L_2 - 8 \times (x_0 + e_{x_0}) \right) \times x_3 \times L_2 - 2 \times (x_0 + e_{x_0}) \times L_2^2 + 4 \times x_3^2 \times L_2 + 2 \times L_2^2 \times x_3 + \frac{1}{3} \times L_2^3 \right) \right) + \frac{-1}{16\pi^3\sigma^{(1-\rho^2)}} \left( \psi_1 \times \psi_2 \times \psi_3 \times \left\{ u((x_0 + e_{x_0}) - (x_3 + L_2)) - u((x_0 + e_{x_0}) - (x_3 + L_1)) \right\} \times \left( 4 \times (x_0 + e_{x_0})^2 \times ((x_0 + e_{x_0}) - x_3) - 8 \times (x_0 + e_{x_0}) \times x_3 \times ((x_0 + e_{x_0}) - x_3) - 2 \times (x_0 + e_{x_0}) \times ((x_0 + e_{x_0}) - x_3) + 4 \times x_3^2 \times ((x_0 + e_{x_0}) - x_3) + 2 \times ((x_0 + e_{x_0}) - x_3)^2 \times x_3 + \frac{1}{3} \times ((x_0 + e_{x_0}) - x_3)^3 - \left( 4 \times (x_0 + e_{x_0})^2 \times L_2 - 8 \times (x_0 + e_{x_0}) \times x_3 \times L_2 - 2 \times (x_0 + e_{x_0}) \times L_2^2 + 4 \times x_3^2 \times L_2 + 2 \times L_2^2 \times x_3 + \frac{1}{3} \times L_2^3 \right) \right) \right) + \frac{-1}{16\pi^3\sigma^{(1-\rho^2)}} \left( \psi_1 \times \psi_2 \times \psi_3 \times \left\{ u((x_0 + e_{x_0}) - (x_3 + L_2)) - u((x_0 + e_{x_0}) - (x_3 + L_1)) \right\} \times \left( \frac{-1}{3} \times L_1^3 - \left( \frac{-1}{3} \times ((x_0 + e_{x_0}) - x_3)^3 \right) \right) \right) \\
I_5 = \frac{1}{48\pi^3\sigma^{(1-\rho^2)}} \left( \psi_2 \times \psi_3 \times \psi_4 \times \left\{ 3 \times ((y_0 + e_{y_0}) - y_1)^2 \times \sqrt{D_1^2 - (x_1 + L_1 - (x_0 + e_{x_0}))^2} - 3 \times ((y_0 + e_{y_0}) - y_1)^2 \times \sqrt{D_1^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2} - 3 \times ((y_0 + e_{y_0}) - y_1)^2 \times \sqrt{D_1^2 - (x_1 + L_1 - (x_0 + e_{x_0}))^2} \right\} \right) \\
I_6 = \frac{1}{48\pi^3\sigma^{(1-\rho^2)}} \left( \psi_1 \times \psi_2 \times \psi_4 \times \left\{ 3 \times ((y_0 + e_{y_0}) - y_2)^2 \times \sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} - 3 \times ((y_0 + e_{y_0}) - y_2)^2 \times \sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} - 3 \times ((y_0 + e_{y_0}) - y_2)^2 \times \sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} \right\} \right) \]
\[ I_7 = \frac{-1}{16\pi^2 \sigma^4(\sqrt{1-\rho^2})} \psi_1 \psi_2 \psi_3 \psi_4 (1-u((x_0 + e_{x_0}) - (x_3 + L_2))) * - (\gamma_3^2 * L_1 - (y_{\gamma} - (y_0 + e_{y_0})) * \\
(x_3 + L_1 - (x_0 + e_{x_0})) \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2 - D_3^2 * (y_3 - (y_0 + e_{y_0}))} * \\
- \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}} \right) - 2 * y_3 * L_1 * (y_0 + e_{y_0}) - x_3^2 * L_2 - \left( \frac{e_x^2 L_2^2}{2} + 2 * x_3 * L_1 * (x_0 + e_{x_0}) + D_3^2 * L_1 - \frac{L_3^3}{3} + L_1^2 * (x_0 + e_{x_0}) - L_1 * (x_0 + e_{x_0})^2 + L_1 * (y_0 + e_{y_0})^2 - (y_3^2 * L_2 - (y_3 - (y_0 + e_{y_0})) * (x_3 + L_2 - (x_0 + e_{x_0})) \sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2 - D_3^2 * (y_3 - (y_0 + e_{y_0}))} * \\
- \tan^{-1} \left( \frac{x_3 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}} \right) - 2 * y_3 * L_2 * (y_0 + e_{y_0}) - x_3^2 * L_2 - \left( \frac{e_x^2 L_2^2}{2} + 2 * x_3 * L_2 * (x_0 + e_{x_0}) + D_3^2 * L_2 - \frac{L_3^3}{3} + L_2^2 * (x_0 + e_{x_0}) - L_2 * (x_0 + e_{x_0})^2 + L_2 * (y_0 + e_{y_0})^2) + \frac{1}{16\pi^2 \sigma^4(\sqrt{1-\rho^2})} \psi_1 \psi_2 \psi_3 \psi_4 * \\
(u((x_0 + e_{x_0}) - (x_3 + L_1))) * (y_3^2 * L_1 - (y_3 - (y_0 + e_{y_0})) * (x_3 + L_1 - (x_0 + e_{x_0})) * \\
\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2 - D_3^2 * (y_3 - (y_0 + e_{y_0}))} * - \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}} \right) - 2 * y_3 * \\
L_1 * (y_0 + e_{y_0}) - x_3^2 * L_1 - \left( \frac{e_x^2 L_1^2}{2} + 2 * x_3 * L_1 * (x_0 + e_{x_0}) + D_3^2 * L_1 - \frac{L_3^3}{3} + L_1^2 * (x_0 + e_{x_0}) - L_1 * (x_0 + e_{x_0})^2 + L_1 * (y_0 + e_{y_0})^2 - (y_3^2 * L_2 - (y_3 - (y_0 + e_{y_0})) * (x_3 + L_2 - (x_0 + e_{x_0})) * \\
\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2 - D_3^2 * (y_3 - (y_0 + e_{y_0}))} * - \tan^{-1} \left( \frac{x_3 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}} \right) - 2 * y_3 * \\
L_2 * (y_0 + e_{y_0}) - x_3^2 * L_2 - \left( \frac{e_x^2 L_2^2}{2} + 2 * x_3 * L_2 * (x_0 + e_{x_0}) + D_3^2 * L_2 - \frac{L_3^3}{3} + L_2^2 * (x_0 + e_{x_0}) - L_2 * (x_0 + e_{x_0})^2 + L_2 * (y_0 + e_{y_0})^2) + \frac{1}{16\pi^2 \sigma^4(\sqrt{1-\rho^2})} \psi_1 \psi_2 \psi_3 \psi_4 * \\
(u((x_0 + e_{x_0}) - (x_3 + L_2))) * (y_3^2 * ((x_0 + e_{x_0}) - x_3) - 2 * y_3 * ((x_0 + e_{x_0}) - x_3) * (y_0 + e_{y_0}) - x_3^2 * \\
((x_0 + e_{x_0}) - x_3) - \frac{(e_x^3((x_0+e_{x_0})-x_3)^2}{2} + 2 * x_3 * ((x_0 + e_{x_0}) - x_3) * (x_0 + e_{x_0}) + D_3^2 * ((x_0 + e_{x_0}) - x_3) - \\
\frac{(x_0+e_{x_0})-x_3^3}{3} + ((x_0 + e_{x_0}) - x_3)^2 * (x_0 + e_{x_0}) - ((x_0 + e_{x_0}) - x_3) * (x_0 + e_{x_0})^2 + ((x_0 + e_{x_0}) - x_3) * \\
(y_0 + e_{y_0})^2 - (y_3^2 * L_2 - (y_3 - (y_0 + e_{y_0})) * (x_3 + L_2 - (x_0 + e_{x_0})) * \sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2 - D_3^2 * (y_3 - (y_0 + e_{y_0}))} * - \tan^{-1} \left( \frac{x_3 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}} \right) - 2 * y_3 * L_2 * (y_0 + e_{y_0}) - x_3^2 * L_2 - \left( \frac{e_x^2 L_2^2}{2} + \right)
\[\begin{align*}
2 \cdot x_3 \cdot L_2 \cdot (x_0 + e_{x_0}) + D_3^2 \cdot L_2 - \frac{l_3^2}{3} + L_2 \cdot (x_0 + e_{x_0}) - L_2 \cdot (x_0 + e_{x_0})^2 + L_2 \cdot (y_0 + e_{y_0})^2) + \\
\frac{-1}{16n^3a^4(1-p^2)} \cdot \psi_1 \cdot \psi_2 \cdot \psi_3 \cdot (u((x_0 + e_{x_0}) - (x_3 + L_2)) - u((x_0 + e_{x_0}) - (x_3 + L_1))) - \frac{y_3^2}{2} \cdot L_1 - \\
(y_3 - (y_0 + e_{y_0})) \cdot (x_3 + L_1 - x_3 + L_0) - (x_3 + L_1) - L_2 \cdot (y_0 + e_{y_0})^2 \cdot D_3^2 \cdot (x_3 + L_1 - x_3 + L_0)^2 - D_3^2 \cdot (y_3 - (y_0 + e_{y_0})) \cdot \\
- \tan^{-1}\left(\frac{x_3 + L_1 - (x_0 + e_{x_0})}{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}\right) - 2 \cdot y_3 \cdot L_1 \cdot (y_0 + e_{y_0}) - x_3^2 \cdot L_1 - \frac{(x^3 + L_1)^2}{2} + 2 \cdot x_3 \cdot L_1 \cdot (x_0 + e_{x_0}) + \\
D_3^2 \cdot L_1 - \frac{l_3^2}{3} + L_1 \cdot (x_0 + e_{x_0}) - L_1 \cdot (x_0 + e_{x_0})^2 + L_1 \cdot (y_0 + e_{y_0})^2 - \frac{y_3^2}{2} \cdot ((x_0 + e_{x_0}) - x_3) - 2 \cdot y_3 \cdot \\
((x_0 + e_{x_0}) - x_3) \cdot (y_0 + e_{y_0}) - x_3^2 \cdot ((x_0 + e_{x_0}) - x_3) - \frac{(x^3 + (x_0 + e_{x_0}) - x_3)^2}{2} + 2 \cdot x_3 \cdot ((x_0 + e_{x_0}) - x_3) \cdot \\
(x_0 + e_{x_0}) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) - \frac{((x_0 + e_{x_0}) - x_3)^2}{3} + ((x_0 + e_{x_0}) - x_3)^2 \cdot (x_0 + e_{x_0}) - ((x_0 + e_{x_0}) - x_3)^2 + ((x_0 + e_{x_0}) - x_3) \cdot (y_0 + e_{y_0})^2) \\
I_5 = \frac{1}{8n^3a^4(1-p^2)} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \psi_8 \\
I_7 = \frac{-1}{48n^3a^4(1-p^2)} \cdot \psi_5 \cdot \psi_7 \cdot \psi_8 \cdot (3 \cdot ((x_0 + e_{x_0}) - x_1)^2 \cdot \sqrt{D_1^2 - (y_1 + L_1 - (y_0 + e_{y_0})^2)} + \\
\left(\sqrt{D_1^2 - (y_1 + L_1 - (y_0 + e_{y_0})^2)}\right)^3 - 3 \cdot ((x_0 + e_{x_0}) - x_1) \cdot (L_1 + y_1 - (y_0 + e_{y_0}))^2 - (3 \cdot ((x_0 + e_{x_0}) - x_1) \cdot (L_1 + y_1 - (y_0 + e_{y_0}))^2 + \\
((x_0 + e_{x_0}) - x_1)^2 \cdot \sqrt{D_1^2 - (y_1 + L_2 - (y_0 + e_{y_0})^2)} + \left(\sqrt{D_1^2 - (y_1 + L_2 - (y_0 + e_{y_0})^2)}\right)^3 - 3 \cdot ((x_0 + e_{x_0}) - x_1) \cdot (L_2 + y_1 - (y_0 + e_{y_0}))^2) \\
I_{10} = \frac{1}{16n^3a^4(1-p^2)} \cdot \psi_5 \cdot \psi_6 \cdot \psi_8 \cdot (1 - u((x_0 + e_{x_0}) - (x_2 + L_2))) \cdot \left(-\frac{1}{3} \cdot L_1^2 - \frac{1}{3} \cdot L_2^2\right) \cdot \\
\frac{1}{16n^3a^4(1-p^2)} \cdot \psi_5 \cdot \psi_6 \cdot \psi_8 \cdot u((x_0 + e_{x_0}) - (x_2 + L_1)) \cdot (4 \cdot (x_0 + e_{x_0})^2 \cdot L_1 - 8 \cdot (x_0 + e_{x_0}) \cdot \\
x_2 \cdot L_1 - 2 \cdot (x_0 + e_{x_0}) \cdot L_1^2 + 4 \cdot x_2 \cdot L_1 + 2 \cdot L_1^2 \cdot x_2 + \frac{1}{3} \cdot L_1^3 - (4 \cdot (x_0 + e_{x_0})^2 \cdot L_2 - 8 \cdot (x_0 + e_{x_0}) \cdot \\
x_2 \cdot L_2 - 2 \cdot (x_0 + e_{x_0}) \cdot L_2^2 + 4 \cdot x_2 \cdot L_2 + 2 \cdot L_2^2 \cdot x_2 + \frac{1}{3} \cdot L_2^3) + \frac{1}{16n^3a^4(1-p^2)} \cdot \psi_5 \cdot \psi_6 \cdot \psi_8 \cdot \\
(u((x_0 + e_{x_0}) - (x_2 + L_2)) - u((x_0 + e_{x_0}) - (x_2 + L_1))) \cdot (4 \cdot (x_0 + e_{x_0})^2 \cdot ((x_0 + e_{x_0}) - x_2) - 8 \cdot \\
(x_0 + e_{x_0}) \cdot x_2 \cdot ((x_0 + e_{x_0}) - x_2) - 2 \cdot (x_0 + e_{x_0}) \cdot ((x_0 + e_{x_0}) - x_2)^2 + 4 \cdot x_2^2 \cdot ((x_0 + e_{x_0}) - x_2) \cdot \\
83
\[2 \left( (x_0 + e_{x_0}) - x_2 \right)^2 \cdot x_2 + \frac{1}{3} \left( (x_0 + e_{x_0}) - x_2 \right)^2 - (4 \cdot (x_0 + e_{x_0})^2) \cdot L_2 - 8 \cdot (x_0 + e_{x_0}) \cdot x_2 \cdot L_2
\]
\[-2 \cdot (x_0 + e_{x_0}) \cdot L_2^2 + 4 \cdot x_2^2 \cdot L_2 + 2 \cdot L_2^2 \cdot x_2 + \frac{1}{3} \cdot L_2^2) \cdot \frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_8
\]
\[\left( u((x_0 + e_{x_0}) - (x_2 + L_2)) - u((x_0 + e_{x_0}) - (x_2 + L_1)))) \cdot \left( -\frac{1}{3} \cdot L_1^3 + \frac{1}{3} \cdot ((x_0 + e_{x_0}) - x_2)^3) \right) \right)
\]
\[
I_{11} = \frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot (1 - u((x_0 + e_{x_0}) - (x_3 + L_2))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \cdot \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}} \right) + \frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot (u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}} \right) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}} \right) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}} \right) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \] 
\[
(2 \cdot D_3^2 + L_1^2 + L_1 \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) \] 
\[
\frac{1}{16\pi^2 \sigma^2 \sqrt{1 - \rho^2}} \cdot \psi_5 \cdot \psi_6 \cdot \psi_7 \cdot \left( u((x_0 + e_{x_0}) - (x_3 + L_1)))) \cdot \left( \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \right) \]
\[(u((x_0 + e_{y_0}) - (x_3 + L_2)) - u((x_0 + e_{y_0}) - (x_3 + L_1))) * (\frac{1}{3} * \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{y_0}))^2} * (2 * D_3^2 + D_1^2 + L_1 * ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - x_3)^2) + D_3^2 * ((x_0 + e_{x_0}) - x_3) * \\
\tan^{-1}\left(\frac{x_3^2 + L_1 - (x_0 + e_{y_0})}{-D_3^2 - (x_3 + L_1 - (x_0 + e_{y_0}))^2}\right) - (\frac{1}{3} * D_3 * (2 * D_3^2 + D_1^2 + ((x_0 + e_{x_0}) - x_3)^2 + ((x_0 + e_{x_0}) - x_3) * ((x_0 + e_{x_0}) - x_3)^2 + ((x_0 + e_{x_0}) - x_3))\right) * \\
\frac{1}{48\pi^2\sigma^2(\sqrt{1 - \rho^2})} * \psi_6 * \psi_7 * \psi_8 * ((3 * ((y_0 + e_{y_0}) - y_1)^2 * \sqrt{D_4^2 - (x_1 + L_1 - (x_0 + e_{x_0}))^2} - 3 * ((y_0 + e_{y_0}) - y_1)^2) * (L_1 + L_2 - (x_0 + e_{x_0}))^2 + ((x_0 + e_{x_0}) - y_1)^2 * \sqrt{D_4^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2} - 3 * ((y_0 + e_{y_0}) - y_1)^2 * (L_2 + x_1 - (x_0 + e_{x_0}))^2 + \\
\left(\sqrt{D_4^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2}\right)^3)) - y_1) * (L_1 + \frac{L_1}{3} + L_2 * (x_0 + e_{x_0}) - L_1 * (x_0 + e_{x_0})^2 + L_1 * (y_0 + e_{y_0})^2 - (y_0 + e_{y_0}) - y_1) * (L_2 + x_1 - (x_0 + e_{x_0}))^2 + \\
\left(\sqrt{D_4^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2}\right)^3)) * \\
(1 - u((x_0 + e_{y_0}) - (x_2 + L_2))) * - (Y_2^2 * L_1 - (y_2 - (y_0 + e_{y_0})) * (x_2 + L_1 - (x_0 + e_{x_0})) \sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} - D_2^2 * (y_2 - (y_0 + e_{y_0})) * \\
- \tan^{-1}\left(\frac{x_2 + L_1 - (x_0 + e_{x_0})}{-D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}\right) - 2 * y_2 * L_1 * (y_0 + e_{y_0}) - x_2^2 * L_1 - \frac{(x_2 + L_1)^2}{2} + 2 * x_2 * L_1 * (x_0 + e_{x_0}) + \\
D_2^2 * L_1 - \frac{L_1}{3} + L_2 * (x_0 + e_{x_0}) - L_1 * (x_0 + e_{x_0})^2 + L_1 * (y_0 + e_{y_0})^2 - (y_0 + e_{y_0}) - y_1) * (L_2 + x_1 - (x_0 + e_{x_0}))^2 + \\
\left(\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}\right)^3)) \psi_8 * \psi_6 * \\
\frac{1}{16\pi^2\sigma^2(\sqrt{1 - \rho^2})} * \psi_6 * (u((x_0 + e_{y_0}) - (x_2 + L_1))) * (y_2^2 * L_1 - (y_2 - (y_0 + e_{y_0})) * (x_2 + L_1 - (x_0 + e_{x_0})) * \\
\sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} - D_2^2 * (y_2 - (y_0 + e_{y_0})) * - \tan^{-1}\left(\frac{x_2 + L_1 - (x_0 + e_{x_0})}{-D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}\right) - 2 * y_2 * \\
L_1 * (y_0 + e_{y_0}) - x_2^2 * L_1 - \frac{(x_2 + L_1)^2}{2} + 2 * x_2 * L_1 * (x_0 + e_{x_0}) + D_2^2 * L_1 - \frac{L_1}{3} + L_2 * (x_0 + e_{x_0}) - L_1 * \\
(x_0 + e_{x_0})^2 + L_1 * (y_0 + e_{y_0})^2 - (y_0 + e_{y_0}) - y_1) * (L_2 - (y_2 - (y_0 + e_{y_0})) * (x_2 + L_2 - (x_0 + e_{x_0})) * \\
\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} - D_2^2 * (y_2 - (y_0 + e_{y_0})) * - \tan^{-1}\left(\frac{x_2 + L_1 - (x_0 + e_{x_0})}{-D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}\right) - 2 * y_2 * \\
\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} - D_2^2 * (y_2 - (y_0 + e_{y_0})) * - \tan^{-1}\left(\frac{x_2 + L_1 - (x_0 + e_{x_0})}{-D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}\right) - 2 * y_2 * \\
88
\[
\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} - D_2^2 \ast (y_2 - (y_0 + e_{y_0})) \ast -\tan^{-1}\left(\frac{x_2 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}}\right) - 2 \ast y_2 \ast \\
L_2 \ast (y_0 + e_{y_0}) - x_2 \ast L_2 - \frac{(x_2 + L_2^2)}{2} + 2 \ast x_2 \ast L_2 \ast (x_0 + e_{x_0}) + D_2^2 \ast L_2 - \frac{L_2^2}{3} + L_2 \ast (x_0 + e_{x_0}) - L_2 \ast (x_0 + e_{x_0})^3 + L_2 \ast (y_0 + e_{y_0})^3) + \frac{1}{\sqrt{1 + \rho^2}} \ast \psi_5 \ast \psi_6 \ast \psi_8 \ast (u((x_0 + e_{x_0}) - (x_2 + L_2))) - \\
u((x_0 + e_{x_0}) - (x_2 + L_1))) \ast (y_2^2 \ast ((x_0 + e_{x_0}) - x_2) - 2 \ast y_2 \ast ((x_0 + e_{x_0}) - x_2) \ast (y_0 + e_{y_0}) - x_2 \ast \\
((x_0 + e_{x_0}) - x_2) - \frac{(x_0 + e_{x_0})^2 - x_2^2}{2} + 2 \ast x_2 \ast ((x_0 + e_{x_0}) - x_2) \ast (x_0 + e_{x_0}) + D_2^2 \ast ((x_0 + e_{x_0}) - x_2) - \\
((x_0 + e_{x_0}) - x_2^3) + ((x_0 + e_{x_0}) - x_2^2 \ast (x_0 + e_{x_0}) - (x_0 + e_{x_0}) - (x_0 + e_{x_0}) - x_2) \ast (x_0 + e_{x_0})^2 + ((x_0 + e_{x_0}) - x_2) \ast \\
(y_0 + e_{y_0})^2 - (y_2 \ast L_2 - (y_2 - (y_0 + e_{y_0})) \ast (x_2 + L_2 - (x_0 + e_{x_0})) \ast \sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} - \\
D_2^2 \ast (y_2 - (y_0 + e_{y_0})) \ast -\tan^{-1}\left(\frac{x_2 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}}\right) - 2 \ast y_2 \ast L_2 \ast (y_0 + e_{y_0}) - x_2^2 \ast L_2 - \frac{(x_2 + L_2^2)}{2} + \\
2 \ast x_2 \ast L_2 \ast (x_0 + e_{x_0}) + D_2^2 \ast L_2 - \frac{L_2^2}{3} + L_2 \ast (x_0 + e_{x_0}) - L_2 \ast (x_0 + e_{x_0})^2 + L_2 \ast (y_0 + e_{y_0})^2) + \\
\frac{1}{\sqrt{1 + \rho^2}} \ast \psi_5 \ast \psi_6 \ast \psi_8 \ast (u((x_0 + e_{x_0}) - (x_2 + L_2))) - u((x_0 + e_{x_0}) - (x_2 + L_1))) \ast -\left(y_2 \ast \\
L_1 \ast (y_2 - (y_0 + e_{y_0})) \ast (x_2 + L_1 - (x_0 + e_{x_0})) \ast \sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2} - D_2^2 \ast (y_2 - (y_0 + \\
e_{y_0})) \ast -\tan^{-1}\left(\frac{x_2 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}}\right) - 2 \ast y_2 \ast L_1 \ast (y_0 + e_{y_0}) - x_2^2 \ast L_1 - \frac{(x_2 + L_1^2)}{2} + 2 \ast x_2 \ast L_1 \ast (x_0 + \\
e_{x_0}) + D_2^2 \ast L_1 - \frac{L_1^2}{3} + L_1 \ast (x_0 + e_{x_0}) - L_1 \ast (x_0 + e_{x_0})^2 + L_1 \ast (y_0 + e_{y_0})^2 - (y_2 \ast ((x_0 + e_{x_0}) - x_2) - \\
2 \ast y_2 \ast ((x_0 + e_{x_0}) - x_2) \ast (y_0 + e_{y_0}) - x_2 \ast ((x_0 + e_{x_0}) - x_2) - \frac{x_2^2((x_0 + e_{x_0}) - x_2)^2}{2} + 2 \ast x_2 \ast ((x_0 + e_{x_0}) - \\
x_2) \ast (x_0 + e_{x_0}) + D_2^2 \ast ((x_0 + e_{x_0}) - x_2) - \frac{(x_0 + e_{x_0})^3}{3} + ((x_0 + e_{x_0}) - x_2)^2 \ast (x_0 + e_{x_0}) - ((x_0 + \\
e_{x_0}) - x_2) \ast (x_0 + e_{x_0})^2 + ((x_0 + e_{x_0}) - x_2) \ast (y_0 + e_{y_0})^2) \right)
\[ I_{14} = \frac{-1}{48\pi^{3}a^{2}((1-p^{2})^{2})} \ast \psi_{5} \ast \psi_{6} \ast \psi_{7} \ast \left((3 \ast (y_{0} + e_{yo}) - y_{3})^{2} \ast \sqrt{D_{3}^{2} - (x_{3} + L_{1} - (x_{0} + e_{xo})^{2}} - 3 \ast \right.
\left((y_{0} + e_{yo}) - y_{3}) \ast (L_{1} + x_{3} - (x_{0} + e_{xo})^{2} + \left(\sqrt{D_{3}^{2} - (x_{3} + L_{1} - (x_{0} + e_{xo})^{2}}\right)^{2} - 3 \ast ((y_{0} + e_{yo}) - y_{3})^{2} \ast \sqrt{D_{3}^{2} - (x_{3} + L_{2} - (x_{0} + e_{xo})^{2}} - 3 \ast ((y_{0} + e_{yo}) - y_{3}) \ast (L_{2} + x_{3} - (x_{0} + e_{xo})^{2} + \left(\sqrt{D_{3}^{2} - (x_{3} + L_{2} - (x_{0} + e_{xo})^{2}}\right)^{2} \right)\right)\right))\]

\[ I_{15} = \frac{1}{32\pi^{3}a^{2}((1-p^{2})^{2})} \ast \psi_{3} \ast \psi_{4} \ast \left((\rho \ast \left(\sqrt{D_{1}^{2} - (x_{1} + L_{1} - (x_{0} + e_{xo})^{2}} - \sqrt{D_{1}^{2} - (x_{1} + L_{2} - (x_{0} + e_{xo})^{2}}\right) \ast \left(\sqrt{D_{1}^{2} - (y_{1} + L_{1} - (y_{0} + e_{yo})^{2}} - \sqrt{D_{1}^{2} - (y_{1} + L_{2} - (y_{0} + e_{yo})^{2}}\right) \ast (2 \ast (x_{0} + e_{xo}) - 2 \ast x_{1} + \sqrt{D_{1}^{2} - (x_{1} + L_{1} - (x_{0} + e_{xo})^{2}} + \sqrt{D_{1}^{2} - (x_{1} + L_{2} - (x_{0} + e_{xo})^{2}}\right) \ast \left(\sqrt{D_{1}^{2} - (y_{1} + L_{1} - (y_{0} + e_{yo})^{2}} + \sqrt{D_{1}^{2} - (y_{1} + L_{2} - (y_{0} + e_{yo})^{2}}\right) \right)\right)\right)\right)\right)\]

\[ I_{16} = \frac{-1}{48\pi^{3}a^{2}((1-p^{2})^{2})} \ast \psi_{1} \ast \psi_{2} \ast \psi_{4} \ast \left((1 - u((x_{0} + e_{xo}) - (x_{2} + L_{2})),) \ast ((\rho \ast (6 \ast \sqrt{D_{2}^{2} - (x_{2} + L_{1} - (x_{0} + e_{xo})^{2}} \ast \left((x_{0} + e_{xo}) - x_{2}) \ast ((y_{0} + e_{yo}) - y_{2}) + 3 \ast \sqrt{D_{2}^{2} - (x_{2} + L_{1} - (x_{0} + e_{xo})^{2}} \ast \left((x_{0} + e_{xo}) - x_{2}) + (x_{2} + L_{1} - (x_{0} + e_{xo})\ast \left(\sqrt{D_{2}^{2} - (x_{2} + L_{1} - (x_{0} + e_{xo})^{2}}\right)^{2} \ast (3 \ast y_{0} - 3 \ast y_{2} + 2 \ast \sqrt{D_{2}^{2} - (x_{2} + L_{1} - (x_{0} + e_{xo})^{2}} + 3 \ast e_{yo}) - 2 \ast D_{2}^{2}) + 3 \ast D_{2}^{2} \ast ((y_{0} + e_{yo}) - y_{2}) \ast \tan^{-1}\left(\frac{\sqrt{D_{2}^{2} - (x_{2} + L_{1} - (x_{0} + e_{xo})^{2}})}{x_{2} + L_{1} - (x_{0} + e_{xo})}\right)\right)\right)\right)\right)\right)\right)\right)\right)\right)\right)\right)\right)\right)\]
\[ e_{x_0} - x_2 + 3 \cdot D_2^2 \cdot ((y_0 + e_{y_0}) - y_2) \cdot \text{tan}^{-1} \left( \frac{\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}}{x_2 + L_2 - (x_0 + e_{x_0})} \right) \]
\[ I_{17} = \frac{1}{16n^2 a (1 - \rho^2)} \left( \psi_1 \cdot \psi_2 \cdot \psi_3 \cdot \left( (1 - u((x_0 + e_{x_0}) - (x_3 + L_2))) \right) \right) \left( (\rho \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \right) \]

\[ \left( x_3 + L_1 - (x_0 + e_{x_0}) \right)^2 - 2 \cdot (y_0 + e_{y_0}) - y_3) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) + \frac{1}{3} \]

\[ \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0})) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0}))} \]

\[ - 2 \cdot \left( \sqrt{D_3} \cdot D_3 \cdot (x_3 - (x_0 + e_{x_0})) \cdot \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))}} \right) \right) - \left( \rho \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \right) \]

\[ (x_3 + L_2 - (x_0 + e_{x_0}))^2 - 2 \cdot (y_0 + e_{y_0}) - y_3) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) + \frac{1}{3} \]

\[ \sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0})) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0}))} \]

\[ - 2 \cdot \left( \sqrt{D_3} \cdot D_3 \cdot (x_3 - (x_0 + e_{x_0})) \cdot \tan^{-1} \left( \frac{x_3 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))}} \right) \right) + u((x_0 + e_{x_0}) - (x_3 + L_2)) \]

\[ - u((x_0 + e_{x_0}) - (x_3 + L_1)) \cdot \left( \frac{\rho \cdot (3 \cdot (D_3 \cdot (x_3 + L_2 - (x_0 + e_{x_0}))) + \frac{1}{3} \cdot (\rho \cdot (3 \cdot (y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \cdot \left( x_3 + L_2 - (x_0 + e_{x_0}) \right)^2 + 6 \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \right) \]

\[ \left( x_3 + L_2 - (x_0 + e_{x_0}) \right)^2 - 3 \cdot \left( \sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0})) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0}))} \right) \]

\[ - 2 \cdot \left( \sqrt{D_3} \cdot D_3 \cdot (x_3 - (x_0 + e_{x_0})) \cdot \tan^{-1} \left( \frac{x_3 + L_2 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))}} \right) \right) + u((x_0 + e_{x_0}) - (x_3 + L_2)) \]

\[ - u((x_0 + e_{x_0}) - (x_3 + L_1)) \cdot \left( \rho \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_1 - (x_0 + e_{x_0}))^2 - 2 \cdot (y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \right) \]

\[ (x_3 + L_1 - (x_0 + e_{x_0}))^2 - 3 \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) - 2 \cdot \left( \sqrt{D_3} \right) \cdot D_3 \cdot \left( x_3 - (x_0 + e_{x_0}) \right)^2 \]

\[ \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))}} \right) \right) - \left( \rho \cdot (1 \cdot D_3 \cdot (D_3 \cdot (y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \cdot \left( x_3 + L_1 - (x_0 + e_{x_0}) \right)^2 + 6 \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_1 - (x_0 + e_{x_0})) \right) \]

\[ \left( x_3 + L_1 - (x_0 + e_{x_0}) \right)^2 + \left( x_3 + L_1 - (x_0 + e_{x_0}) \right)^2 \cdot (x_3 - (x_0 + e_{x_0})) \cdot \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))}} \right) \right) \]

\[ - \left( \rho \cdot (3 \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_2 - (x_0 + e_{x_0}))^2 + 6 \cdot \left( x_3 + L_2 - (x_0 + e_{x_0}) \right)^2 + 3 \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \right) \]

\[ \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))}} \right) \right) \]

\[ - \left( \rho \cdot (3 \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_3 + L_2 - (x_0 + e_{x_0}))^2 + 6 \cdot \left( x_3 + L_2 - (x_0 + e_{x_0}) \right)^2 + 3 \cdot (x_3 - (x_0 + e_{x_0})) \cdot (x_3 + L_2 - (x_0 + e_{x_0})) \right) \]

\[ \tan^{-1} \left( \frac{x_3 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))}} \right) \right) \]
\[(Y_0 + e_{y_0}) - y_3 \) \* (x_3 - (x_0 + e_{x_0})) \* (x_3 + L_2 - (x_0 + e_{x_0})) + \sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2} \* (2 \* (x_3 + L_2 - (x_0 + e_{x_0}))^2 + 3 \* (x_3 - (x_0 + e_{x_0})) \* (x_3 + L_2 - (x_0 + e_{x_0})) - 2 \* D_3^2) - 3 \* D_3^2 \* (x_3 - (x_0 + e_{x_0})) \* \tan^{-1} \left( \frac{x_3 + L_2 - (x_0 + e_{x_0})}{-\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}} \right) \)\]

\[I_{18} = \frac{-1}{32\pi^3 \alpha^2 \sqrt{1 - \rho^2}} \* \psi_7 \* \psi_8 \* (\rho \* (\sqrt{D_1^2 - (x_1 + L_1 - (x_0 + e_{x_0}))^2} - \sqrt{D_1^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2}) \* \sqrt{D_1^2 - (y_1 + L_1 - (y_0 + e_{y_0}))^2} - \sqrt{D_1^2 - (y_1 + L_2 - (y_0 + e_{y_0}))^2} \* (2 \* (x_0 + e_{x_0}) - 2 \* x_1 + \sqrt{D_1^2 - (x_1 + L_1 - (x_0 + e_{x_0}))^2} + \sqrt{D_1^2 - (x_1 + L_2 - (x_0 + e_{x_0}))^2}) \* (2 \* (y_0 + e_{y_0}) - 2 \* y_1 + \sqrt{D_1^2 - (y_1 + L_1 - (y_0 + e_{y_0}))^2} + \sqrt{D_1^2 - (y_1 + L_2 - (y_0 + e_{y_0}))^2}))\]

\[I_{19} = \frac{-1}{16\pi^3 \alpha^2 \sqrt{1 - \rho^2}} \* \psi_5 \* \psi_6 \* \psi_8 \* ((1 - u((x_0 + e_{x_0}) - (x_2 + L_2))) \* ((\rho \* ((y_0 + e_{y_0}) - y_2)) \* (x_2 + L_1 - (x_0 + e_{x_0})) - 2 \* (y_0 + e_{y_0}) - y_2) \* (x_2 - (x_0 + e_{x_0})) \* (x_2 + L_1 - (x_0 + e_{x_0})) + \frac{1}{3} \* (x_2 - (x_0 + e_{x_0})) - 2 \* D_2^2 + D_2^2 \* (x_2 - (x_0 + e_{x_0})) \* \tan^{-1} \left( \frac{x_2 + L_1 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}} \right) \* (\rho \* ((y_0 + e_{y_0}) - y_2)) \* (x_2 + L_2 - (x_0 + e_{x_0})) + \frac{1}{3} \* (x_2 - (x_0 + e_{x_0})) - 2 \* D_2^2 + D_2^2 \* (x_2 - (x_0 + e_{x_0})) \* \tan^{-1} \left( \frac{x_2 + L_2 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}} \right) \* u((x_0 + e_{x_0}) - (x_2 + L_2)) \* (y_0 + e_{y_0}) - y_2 - (x_0 + e_{x_0}) + 6 \* (y_0 + e_{y_0}) - y_2) \* (x_2 - (x_0 + e_{x_0})) \* (x_2 + L_2 - (x_0 + e_{x_0})) + \sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2} \* (2 \* (x_2 + L_2 - (x_0 + e_{x_0}))^2 + 3 \* (x_2 - (x_0 + e_{x_0})) \* (x_2 + L_2 - (x_0 + e_{x_0})) - 2 \* D_2^2 - 3 \* D_2^2 \* (x_2 - (x_0 + e_{x_0})) \* \tan^{-1} \left( \frac{x_2 + L_2 - (x_0 + e_{x_0})}{-\sqrt{D_2^2 - (x_2 + L_2 - (x_0 + e_{x_0}))^2}} \right) \) + u((x_0 + e_{x_0}) - (x_2 + L_2))\]

93
\[ I_{20} = \frac{1}{48\pi\rho \sigma (\sqrt{1 - r^2})} \psi \cdot \psi \cdot \psi \cdot \psi \cdot (1 - u((x_0 + e_{x_0}) - (x_3 + L_3))) \cdot ((\rho \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_2 + L_1 - (x_0 + e_{x_0}))^2 - 2 \cdot ((y_0 + e_{y_0}) - y_3) \cdot (x_2 - (x_0 + e_{x_0})) \cdot (x_2 + L_1 - (x_0 + e_{x_0})) + \frac{1}{3} \cdot \sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2} \cdot (2 \cdot (x_2 + L_1 - (x_0 + e_{x_0}))^2 - 3 \cdot (x_2 - (x_0 + e_{x_0})) \cdot (x_2 + L_1 - (x_0 + e_{x_0})) - 2 \cdot D_1^2) + D_2^2 \cdot (x_2 - (x_0 + e_{x_0})) \cdot \tan^{-1}\left(\frac{x_2 + L_1 - (x_0 + e_{x_0})}{\sqrt{D_3^2 - (x_2 + L_1 - (x_0 + e_{x_0}))^2}}\right)) \]
\begin{align*}
D_3^2) + 3 \cdot D_3^2 \cdot ((y_0 + e_{y_0}) - y_3) \cdot \tan^{-1}\left(\frac{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}}{x_3 + L_2 - (x_0 + e_{x_0})}\right) &+ u((x_0 + e_{x_0}) - (x_3 + L_2)) - \\
u((x_0 + e_{x_0}) - (x_3 + L_1)) &\cdot ((\rho \cdot (6 \cdot D_3 \cdot ((x_0 + e_{x_0}) - x_3) \cdot ((y_0 + e_{y_0}) - y_3) + 3 \cdot D_3^2 \cdot ((x_0 + e_{x_0}) - x_3) + 3 \cdot D_3^2) \cdot ((y_0 + e_{y_0}) - y_3) \cdot \frac{\pi}{2}) - \rho \cdot (6 \cdot D_3, \sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}) \cdot ((x_0 + e_{x_0}) - x_3) \cdot ((y_0 + e_{y_0}) - y_3) + 3 \cdot \left(\frac{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}{x_3 + L_2 - (x_0 + e_{x_0})}\right) \cdot ((x_0 + e_{x_0}) - x_3) + ((x_0 + e_{x_0}) - L_2 - x_3) \cdot \left(\frac{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}{x_3 + L_2 - (x_0 + e_{x_0})}\right)}{2} - 2 \cdot D_3^2) + 3 \cdot D_3^2 \cdot ((y_0 + e_{y_0}) - y_3) \cdot \tan^{-1}\left(\frac{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}}{x_3 + L_1 - (x_0 + e_{x_0})}\right) - \rho \cdot (6 \cdot D_3 \cdot ((x_0 + e_{x_0}) - x_3) \cdot ((y_0 + e_{y_0}) - y_3) + 3 \cdot \left(\frac{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}{x_3 + L_1 - (x_0 + e_{x_0})}\right) - 2 \cdot D_3^2) + 3 \cdot D_3^2 \cdot ((y_0 + e_{y_0}) - y_3) \cdot \tan^{-1}\left(\frac{\sqrt{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}}{x_3 + L_2 - (x_0 + e_{x_0})}\right) - \rho \cdot (6 \cdot D_3 \cdot ((x_0 + e_{x_0}) - x_3) \cdot ((y_0 + e_{y_0}) - y_3) + 3 \cdot \left(\frac{D_3^2 - (x_3 + L_2 - (x_0 + e_{x_0}))^2}{x_3 + L_2 - (x_0 + e_{x_0})}\right) - 2 \cdot D_3^2) + 3 \cdot D_3^2 \cdot ((y_0 + e_{y_0}) - y_3) \cdot \tan^{-1}\left(\frac{\sqrt{D_3^2 - (x_3 + L_1 - (x_0 + e_{x_0}))^2}}{x_3 + L_1 - (x_0 + e_{x_0})}\right)
\end{align*}
Appendix B

The Matlab Program Code

The following are the source code for the simulation Matlab program used in this dissertation. They are divided into files, with each file split into different cases. Some files are common between different cases and are listed once.

B.1. Main.m

Main.m is the main program that controls all the simulation parameters, it differs for each case.

B.1.1. The Indoor and Outdoor LOS Case

B.1.1.1. 2D Case in a LOS Environment

a. 2D Case for Static Users in a LOS Environment

```matlab
clear
for WKR=1:100
    num_users=22;
    map_size = 20;
```
users_location(1,:) = [map_size/2 map_size/2];
users_location(2:num_users,:) = map_size*(rand(num_users-1,2))+1;
height = zeros(2*map_size);
antenna_height = 1.5*ones(1,num_users);
freq_sep = 50*10^6; % frequency seperation
start_freq = 850*10^6; % lowest frequency
user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
for i=2:num_users-1
    user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    end
end

precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate
R = 0.03;
Random = mvnrnd([0 0],[1 0 R;R 10],num_users-1);
rx_locations = users_location(2:end,:) + Random; % adding an error element to the recived location information

% no_spoof = 2;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = users_location(2:no_spoof+1,:) +
% 60*rand(no_spoof,2); % adding a spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:) +
% Random(no_spoof+1:end,:)]; % adding an error element to the recived location information

[phase_shift signal_rx] = signal_simulation(users_location,height,antenna_height,user_freq, 
Fs,tx_power,bit_rate);
signals=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% 
%%   Estimation through RSS 
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
num_info_users = 7; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
signals(:,i)=signal1;
signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))))
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
    rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);
rss_users_loc=[];
for i=1:num_info_users % Matching freqs to locations
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(1),:);
end

table01=table(num_info_users); % table of possible combination of users for triangulation

est_loc_rss_ftr =
    location_estimation_ftr(rss_users_loc,rss_dis,table01);

est_loc_rss_MMSE = MMSE(rss_users_loc,rss_dis);

E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);

E1_MMSE(1) = users_location(1,1)-est_loc_rss_MMSE(:,1);
E1_MMSE(2) = users_location(1,2)-est_loc_rss_MMSE(:,2);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_MMSE.^2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   End of RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

b. 2D Case for Mobile Users in a LOS Environment

clear

map_size = 100;
num_users=22;
users_location(1,:) = [10 10];
users_location = [users_location; map_size*(rand(num_users-1,2))+1];
height = zeros(2*map_size);

% initialization for Kalman filter
L = 10^6;
LOCi_rss = [0 0 ; 0 0];
PXi_rss = [L 0 ; 0 L];
PYi_rss = [L 0 ; 0 L];
LOCi_toa = [0 0 ; 0 0];
PXi_toa = [L 0 ; 0 L];
PYi_toa = [L 0 ; 0 L];
PX0_rss = PXi_rss;
PY0_rss = PYi_rss;
n=0;
m=0;
Distenation=[90 90 ;users_location(2:end,:)];
velocity=[0.75 ;zeros(num_users-1,1)];
pause_time=zeros(num_users,1);
max_pause = 10; % Maximum pause time
%max_pause and velocity have to have the same time unit
%%
for WKR=1:100
    antenna_height = 1.5*ones(1,num_users);
    freq_sep = 50*10^6 ; % frequency separation
    start_freq = 850*10^6; % lowest frequency
    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end
    precision=10^-10;
    Fs=1/precision; % sampling frequency
    C = 299792458 ; % Speed of Light (m/s)
    tx_power=5; % transmission power=5 watt
    bit_rate=50*10^6; % transmission bit rate
    R = 0.03;
    V = 10;
    Random = mvnrnd([0 0],[V R;R V],num_users-1);
    rx_locations = users_location(2:end,:) + Random;

    % no_spoof = 1;
    % rx_locations = [];
    % rx_locations(1:no_spoof,:) = users_location(2:no_spoof+1,:) + 20*rand(no_spoof,2); % adding a spoofing user
    % rx_locations = [rx_locations; users_location(no_spoof+2:end,:)] + Random(no_spoof+1:end,:)); % adding an error element to the
    % recieved location information

    [phase_shift signal_rx] =
    signal_simulation(users_location,height,antenna_height,user_freq,
    Fs,tx_power,bit_rate);
    signals=[];

    Estimation through RSS
    num_info_users=9; % number of users to take info from
    for i=1:num_info_users
        [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate,
        freq_sep);
        signals(:,i)=signal1;
        signal_rx=signal2;
    end
end
freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
end
rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);
end
rss_users_loc=[];
for i=1:num_info_users % Matching freqs to locations
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(1),:);
end
table01=table(num_info_users); % table of possible combination of users for triangulation

est_loc_rss_ftr = location_estimation_ftr(rss_users_loc,rss_dis,table01);

Klmn_rss = location_estimation_klmn(est_loc_rss_ftr,LOCi_rss,PXi_rss,P Yi_rss);
est_loc_rss_klmn = [Klmn_rss(1,1) Klmn_rss(1,2)];
LOCi_rss = Klmn_rss(:,1:2);
PXi_rss = Klmn_rss(:,3:4);
P Yi_rss = Klmn_rss(:,5:6);

n=n+1;
if ((est_loc_rss_ftr(1)-est_loc_rss_klmn(1))^2 + (est_loc_rss_ftr(2)-est_loc_rss_klmn(2))^2)^0.5 > 10
    if n > 5
        PXi_rss = PX0_rss;
P Yi_rss = PY0_rss;
n = 0;
    end
end

E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
E1_klmn(1) = users_location(1,1)-est_loc_rss_klmn(:,1);
E1_klmn(2) = users_location(1,2)-est_loc_rss_klmn(:,2);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_klmn.^2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   End of RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%% Moving the mobile stations
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dis=((Destination(:,1)-users_location(:,1)).^2+(Destination(:,2)-users_location(:,2)).^2).^0.5;
for i=1:num_users
if dis(i)<1
    pause_time(i)=round(rand(1)*max_pause);
    Distenation(i,1)=(rand(1)*(map_size-2))+1;
    Distenation(i,2)=(rand(1)*(map_size-2))+1;
    velocity(i)=(rand(1)+0.4);%*60;        % Avg human speed =
1.4 m/s = 84 m/min
else
    if pause_time>0
        pause_time(i)=pause_time(i)-1;
    else
        users_location(i,1) = users_location(i,1) + velocity(i)
* (Distenation(i,1)-users_location(i,1))/dis(i);
        users_location(i,2) = users_location(i,2) + velocity(i)
* (Distenation(i,2)-users_location(i,2))/dis(i);
    end
end
end

B.1.1.2. 3D Case in a LOS Environment

a. 3D Case for Static Users in a LOS Environment

clear

for WKR=1:100
    num_users=22;
    map_size = 100;
    users_location(1,:) = map_size/2*(rand(1,2))+ map_size/4;
    users_location(2:num_users,:) = map_size*(rand(num_users-1,2))+1;
    height = 30*ones(2*map_size).*((rand(2*map_size))+50);

    for i=1:num_users
        user_height(i,1) = height(round(users_location(i,1)),round(users_location(i,2)));
    end
    antenna_height = 1.5*ones(1,num_users);
    UH = user_height + antenna_height';
    freq_sep = 50*10^6 ; % frequency separation
    start_freq = 850*10^6; % lowest frequency
    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end
    precision=10^-10;
    Fs=1/precision; % sampling frequency

    end

end
C = 299792458 ; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate

R = 0.03;
V = 1;
Random = mvnrnd([0 0 0],[V R R V R R V],num_users-1);
rx_locations = [users_location(2:end,:); UH(2:end)] + Random; % adding an error element to the received location information

% no_spoof = 1;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = [users_location(2:no_spoof+1,:)
UH(2:no_spoof+1,:)] + 20*rand(no_spoof,3); % adding a spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:)]
% no_spoof+2:end]) + Random; % adding an error element to the received location information

m = 0.0289644;
g = 9.80665;
r = 8.31447;
P0 = 101325;
Temp = 303.15; % temperature in kelvins = 30 C = 86 F
users_P = P0*exp(-m*g*UH/r/Temp);
UP = users_P + 12*randn(length(UH),1);

[phase_shift signal_rx] =
signal_simulation(users_location,height,antenna_height,user_freq,
Fs,tx_power,bit_rate);
signals=[];
% Estimation through RSS
num_info_users=9; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
    signals(:,i)=signal1;
    signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
    rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);
end

rss_users_loc=[];
for i=1:num_info_users % Matching freqs to locations
    freq_temp = find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:) = rx_locations(freq_temp(1),:);
    rss_UP(i,:) = UP(freq_temp(1),:);
end
table02=table2D(num_info_users); % table of possible combination of users for triangulation

rss_Tmp = m*g*rss_users_loc(:,3)./(r*(log(P0)-log(rss_UP)));
rss_user_T2 = temp_ftr(rss_Tmp);
rss_user_H2 = rss_user_T2*r*(log(P0)-log(UP(1)))/(m*g);

est_loc_rss_ftr =
location_estimation_ftr(rss_user_H2,rss_users_loc,rss_dis,table02);

E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
E1_ftr(3) = user_height(1,1)+1.5-est_loc_rss_ftr(:,3);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));

b. 3D Case for Mobile Users in a LOS Environment

clear

map_size = 100;
num_users=22;
users_location = map_size*(rand(num_users,2))+1;
height = 30*ones(2*map_size).*(rand(2*map_size))+50;

% initialization for Kalman filter
LOCi_rss = [0 0 0; 0 0 0];
PXi_rss = [10^5 0; 0 10^5];
PYi_rss = [10^5 0; 0 10^5];
PX0_rss = PXi_rss;
PY0_rss = PYi_rss;

Distenation=users_location;
velocity=zeros(num_users,1);
pause_time=zeros(num_users,1);
max_pause = 10; % Maximum pause time

for WKR=1:100
    for i=1:num_users
        user_height(i,1) = height(round(users_location(i,1)),round(users_location(i,2)));
    end
    antenna_height = 1.5*ones(1,num_users);
    UH = user_height + antenna_height';

end
freq_sep = 50*10^6; % frequency separation
start_freq = 850*10^6; % lowest frequency
user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
for i=2:num_users-1
    user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    end
end

precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate

R = 0.03;
Random = mvnrnd([0 0 0],[1 R R;R 1 R;R R 1],num_users-1);
rx_locations = [users_location(2:end,:);UH(2:end)] + Random; % adding an error element to the received location information

% no_spoof = 2;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = [users_location(2:no_spoof+1,:)
% UH(2:no_spoof+1,:)] + 60*rand(no_spoof,3); % adding a spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:)] + Random; % adding an error element to the received location information

m = 0.0289644;
g = 9.80665;
r = 8.31447;
P0 = 101325;
Temp = 303.15; % temperature in kelvins = 30 C = 86 F
users_P = P0*exp(-m*g*UH/r/Temp);
UP = users_P + 12*randn(length(UH),1);

[phase_shift signal_rx] =
signal_simulation(users_location,UH,user_freq, Fs,tx_power,bit_rate);
signals=[];

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Estimation through RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
num_info_users=3; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate,
    freq_sep);
signals(:,i)=signal1;
signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
end
rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);

rss_users_loc=[];
for i=1:num_info_users
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(1,:),:);
    rss_UP(i,:) = UP(freq_temp(1)+1,:);
end
table02=table2D(num_info_users);  % table of possible combination of users for triangulation

rss_Tmp = m*g*rss_users_loc(:,3)./(r*(log(P0)-log(rss_UP)));
rss_user_T = sum(rss_Tmp)/length(rss_Tmp);
rss_user_H = rss_user_T*r*(log(P0)-log(UP(1)))/(m*g);

est_loc_rss_ftr =
    location_estimation_ftr(rss_user_H,rss_users_loc,rss_dis,table02);

Klmn_rss =
    location_estimation_klmn(est_loc_rss_ftr,LOCi_rss,PIXi_rss,PIyi_rss);
est_loc_rss_klmn = [Klmn_rss(:,1) Klmn_rss(:,2) Klmn_rss(:,3)];

LOCi_rss = Klmn_rss(:,1:3);
PIXi_rss = Klmn_rss(:,4:5);
PIyi_rss = Klmn_rss(:,6:7);

n=n+1;
if ((est_loc_rss_ftr(1)-est_loc_rss_klmn(1))^2 +
    (est_loc_rss_ftr(2)-est_loc_rss_klmn(2))^2)^0.5 > 4
    if n > 5
        PXi_rss = PX0_rss;
        PIyi_rss = PY0_rss;
        n = 0;
    end
end

est_loc_rss_MMSE = MMSE(rss_users_loc,rss_dis);

E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
E1_ftr(3) = user_height(1,1)-est_loc_rss_ftr(:,3);
E1_klmn(1) = users_location(1,1)-est_loc_rss_klmn(:,1);
E1_klmn(2) = users_location(1,2)-est_loc_rss_klmn(:,2);
E1_klmn(3) = user_height(1,1)-est_loc_rss_klmn(:,3);
E1_MMSE(1) = users_location(1,1)-est_loc_rss_MMSE(:,1);
E1_MMSE(2) = users_location(1,2)-est_loc_rss_MMSE(:,2);
E1_MMSE(3) = user_height(1,1)-est_loc_rss_MMSE(:,3);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_klmn.^2));
error_rss(3,WKR)=sqrt(sum(E1_MMSE.^2));
B.1.2. The Indoor NLOS Case

B.1.2.1. 2D Case in an Indoor NLOS Environment

a. 2D Case for Static Users in an Indoor NLOS Environment

```matlab
clear

surface = xlsread('surfaces.xls','Building');
permit = xlsread('surfaces.xls','Permittivity'); % Air, Concrete and Glass

for WKR=1:100
    num_users=22;
    map_size = 100;

    for i=1:num_users
        users_location(i,:,:) = (map_size-10)*(rand(1,2))+5;
        while surface(ceil(users_location(i,:))) ~= 0
            users_location(i,:) = (map_size-10)*(rand(1,2))+5;
        end
    end
```
height = zeros(2*map_size);
antenna_height = 1.5*ones(1,num_users);
freq_sep = 50*10^6; % frequency separation
start_freq = 850*10^6; % lowest frequency
user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
for i=2:num_users-1
    user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    end
end

precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458 ; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate
R = 0.03;
Random = mvnrnd([0 0],[1 R;R 1],num_users-1);
rx_locations = users_location(2:end,:) + Random; % adding an error
element to the recieved location information

% no_spoof = 2;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = users_location(2:no_spoof+1,:) +
% 20*rand(no_spoof,2); % adding a spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:) +
% Random(no_spoof+1:end,:)] % adding an error element to the recieved
% location Information

[phase_shift signal_rx] = signal_simulation(users_location, height,
antenna_height, user_freq, Fs, tx_power, bit_rate, surface, perm);
signals=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Estimation through RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
num_info_users=3; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
signals(:,i)=signal1;
signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
\[
\text{rss\_dis}(i) = C \times (\text{tx\_power})^{0.5} / ((4\pi \times (f_{\text{req}}(i))) \times (\text{rx\_power}(i))^{0.5})
\]
end

\[
\text{rss\_users\_loc} = [];
\]
for \(i=1:\text{num\_info\_users}\) % Matching freqs to locations
    freq\_temp = \text{find}(\text{abs(user\_freq-freq(i))}<\text{freq}\_sep/2);
    \text{rss\_users\_loc}(i,:) = \text{rx\_locations}(\text{freq\_temp}(1,:),:);
end

table01 = \text{table}(\text{num\_info\_users}); % table of possible combination of users for triangulation

\[
\text{est\_loc\_rss\_ftr} = \text{location\_estimation\_ftr}(\text{rss\_users\_loc}, \text{rss\_dis}, \text{table01});
\]
\[
\text{E1\_ftr}(1) = \text{users\_location}(1,1) - \text{est\_loc\_rss\_ftr}(1,:);
\]
\[
\text{E1\_ftr}(2) = \text{users\_location}(1,2) - \text{est\_loc\_rss\_ftr}(2,:);
\]
\[
\text{error\_rss}(1,\text{WKR}) = \sqrt{\text{sum}(\text{E1\_ftr}^2)};
\]

b. 2D Case for Mobile Users in an Indoor NLOS Environment

clear

\[
\text{surface} = \text{xlsread('surfaces.xls','Building')};
\]
\[
\text{permit} = \text{xlsread('surfaces.xls','Permittivity')}; % Air, Concrete and Glass
\]
map\_size = 100;
num\_users = 22;

\[
\text{for } i=1:\text{num\_users}
    \text{users\_location}(i,:) = ((\text{map\_size}-4) \times \text{rand}(1,2)) + 2;
    \text{while } \text{surface(ceil(\text{users\_location}(i,:)))} \neq 0
        \text{users\_location}(i,:) = ((\text{map\_size}-4) \times \text{rand}(1,2)) + 2;
    \end
\end

\[
\text{height} = \text{zeros}(2 \times \text{map\_size});
\]
\[
\text{% initialization for Kalman filter}
\text{LOC}_{\text{i\_rss}} = [0 \ 0 \ 0 \ 0];
\text{PX}_{\text{i\_rss}} = [10^5 \ 0 \ 0 \ 10^5];
\text{PY}_{\text{i\_rss}} = [10^5 \ 0 \ 0 \ 10^5];
\text{PX0\_rss} = \text{PX}_{\text{i\_rss}};
\text{PY0\_rss} = \text{PY}_{\text{i\_rss}};
\text{n}=0;
\text{m}=0;
\text{Destination}=\text{users\_location};
\]
velocity=zeros(num_users,1);
pause_time=zeros(num_users,1);
max_pause = 10; % Maximum pause time
% max_pause and velocity have to have the same time unit
for WKR=1:100
    antenna_height = 1.5*ones(1,num_users);
    freq_sep = 50*10^6; % frequency separation
    start_freq = 850*10^6; % lowest frequency
    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end
    max_pause = 10; % Maximum pause time
    pause_time = zeros(num_users,1);
    pause_time(1) = max_pause;
    for t=2:num_users
        pause_time(t) = max(min(pause_time(t-1)+max_pause,pauser_time(t-1)),0);
    end
end

precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate

R = 0.03;
Random = mvnrnd([0 0],[1 R;R 1],num_users-1);
rx_locations = users_location(2:end,:) + Random; % adding an error element to the received location information
%
no_spoof = 2;
rx_locations = [];
rx_locations(1:no_spoof,:) = users_location(2:no_spoof+1,:) + 60*rand(no_spoof,2); % adding a spoofing user
rx_locations = [rx_locations; users_location(no_spoof+2:end,:) + Random(no_spoof+1:end,:)] % adding an error element to the received location information

(phase_shift signal_rx) = signal_simulation(users_location, height, antenna_height, user_freq, Fs, tx_power, bit_rate, surface, permit);
signals=[];

Estimation through RSS
num_info_users=8; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
    signals(:,i)=signal1;
    signals(:,i)=signal2;
end

freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
end

rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);
end

rss_users_loc=[];
for i=1:num_info_users
    % Matching freqs to locations
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(i,:),:);
end
table01=table(num_info_users); % table of possible combination of
users for triangulation

est_loc_rss_ftr =
location_estimation_ftr(rss_users_loc,rss_dis,table01);

Klmn_rss =
location_estimation_klmn(est_loc_rss_ftr,LOCI_rss,PXi_rss,PYi_rss);
est_loc_rss_klmn = [Klmn_rss(1,1) Klmn_rss(1,2)];
LOCI_rss = Klmn_rss(:,1:2);
PXi_rss = Klmn_rss(:,3:4);
PYi_rss = Klmn_rss(:,5:6);

n=n+1;
if ((est_loc_rss_ftr(1)-est_loc_rss_klmn(1))^2 +
(est_loc_rss_ftr(2)-est_loc_rss_klmn(2))^2)^0.5 > 3
    if n > 5
        PXi_rss = PX0_rss;
        PYi_rss = PY0_rss;
        n = 0;
    end
end

E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
E1_klmn(1) = users_location(1,1)-est_loc_rss_klmn(:,1);
E1_klmn(2) = users_location(1,2)-est_loc_rss_klmn(:,2);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_klmn.^2));

B.1.2.2. 3D Case in an Indoor NLOS Environment

a.  3D Case for Static Users in an Indoor NLOS Environment

clear
surface(:,:,1) = xlsread('surfaces.xls','Floor 1');
surface(:,:,2) = xlsread('surfaces.xls','Floor 2');
surface(:,:,3) = xlsread('surfaces.xls','Floor 3');
surface(:,:,4) = xlsread('surfaces.xls','Floor 4');
surface(:,:,5) = xlsread('surfaces.xls','Floor 5');
permit = xlsread('surfaces.xls','Permittivity'); % Air, Concrete and Glass

%%
for WKR=1:100
    num_users=22;
    map_size = 100;
    for i=1:num_users
        users_location(i,:) = (map_size-10)*(rand(1,2))+5;
        while surface(round(users_location(i,:))) ~= 0
            users_location(i,:) = (map_size-10)*(rand(1,2))+5;
        end
    end

    height = 0:6:24;
    antenna_height = 1.5*ones(num_users,1);
    user_height(:,1) = height(round(4*rand(num_users,1)+1));
    UH = user_height + antenna_height;
    freq_sep = 50*10^6; % frequency seperation
    start_freq = 850*10^6; % lowest frequency
    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end

    precision=10^-10;
    Fs=1/precision; % sampling frequency
    C = 299792458 ; % Speed of Light (m/s)
    tx_power=5; % transmission power=5 watt
    bit_rate=50*10^6; % transmission bit rate

    R = 0.03;
    V = 1;
    Random = mvnrnd([0 0 0],[V R R;R V R;R R V],num_users-1);
    rx_locations = [users_location(2:end,:); UH(2:end)] + Random; % adding an error element to the recieved location information

    % no_spoof = 2;
    % rx_locations = [];
    % rx_locations(1:no_spoof,:) = [users_location(2:no_spoof+1,:);
    % UH(2:no_spoof+1,:)] + 60*rand(no_spoof,3); % adding a spoofing user
    % rx_locations = [rx_locations; users_location(no_spoof+2:end,:);
    % UH(no_spoof+2:end)] + Random; % adding an error element to the recieved location information

    m = 0.0289644;
    g = 9.80665;
    r = 8.31447;
P0 = 101325;
Temp = 303.15; % temperature in kelvins = 30 C = 86 F
users_P = P0*exp(-m*g*UH/r/Temp);
UP = users_P + 12*randn(length(UH),1);

[phase_shift signal_rx] =
signal_simulation(users_location,UH,user_freq, Fs,tx_power,bit_rate,
surface, permit);
signals=[];

% Estimation through RSS

num_info_users=9; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
    signals(:,i)=signal1;
    signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
    rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);
end

rss_users_loc=[];
for i=1:num_info_users % Matching freqs to locations
    freq_temp = find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:) = rx_locations(freq_temp(1),:);
    rss_UP(i,:) = UP(freq_temp(1)+1,:);
end

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));

% End of RSS

end
b. 3D Case for Mobile Users in an Indoor NLOS Environment

clear

surface(:,:,1) = xlsread('surfaces.xls','Floor 1');
surface(:,:,2) = xlsread('surfaces.xls','Floor 2');
surface(:,:,3) = xlsread('surfaces.xls','Floor 3');
surface(:,:,4) = xlsread('surfaces.xls','Floor 4');
surface(:,:,5) = xlsread('surfaces.xls','Floor 5');
permit = xlsread('surfaces.xls','Permittivity'); % Air, Concrete and Glass

map_size = 100;
num_users=22;
for i=1:num_users
    users_location(i,:) = ((map_size-10)*rand(1,2))+5;
    while surface(round(users_location(i,:))) ~= 0
        users_location(i,:) = ((map_size-10)*rand(1,2))+5;
    end
end
height = 0:6:24;
antenna_height = 1.5*ones(num_users,1);
user_height(:,1) = height(round(4*rand(num_users,1)+1));
freq_sep = 50*10^6; % frequency separation
start_freq = 850*10^6; % lowest frequency

% initialization for Kalman filter
LOCi_rss = [0 0 0; 0 0 0];
PXi_rss = [10^5 0; 0 10^5];
PYi_rss = [10^5 0; 0 10^5];
PX0_rss = PXi_rss;
PY0_rss = PYi_rss;
n=0;
M=0;

Destination=users_location;
velocity=zeros(num_users,1);
pause_time=zeros(num_users,1);
max_pause = 10; % Maximum pause time
%max_pause and velocity have to have the same time unit
%
for WK=1:100
    UH = user_height + antenna_height;
    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end
    precision=10^-10;
    Fs=1/precision; % sampling frequency
C = 29972458 ; % Speed of Light (m/s)

\[
tx\_power=5; \quad \% \text{transmission power=5 watt}
\]

\[
bit\_rate=50*10^6; \quad \% \text{transmission bit rate}
\]

\[
R = 0.03;
\]

\[
V = 1;
\]

Random = mvnrnd([0 0 0],[V R R;R V R;R R V],num_users-1);

Rx_locations = [users\_location(2:end,:), user\_height(2:end)] + Random;
% adding an error element to the recieved location information

% no_spoof = 2;
% Rx_locations = [];
% Rx_locations(1:no_spoof,:) = [users\_location(2:no_spoof+1,:),
% user\_height(2:no_spoof+1,:)] + 60*rand(no_spoof,3); % adding a spoofing user
% Rx_locations = [Rx_locations; users\_location(no_spoof+2:end,:)
% user\_height(no_spoof+2:end)] + Random; % adding an error element to the recieved location information

m = 0.0289644;
g = 9.80665;
r = 8.31447;
P0 = 101325;
Temp = 303.15; % temperature in kelvins = 30 C = 86 F

\[
users\_P = P0*exp(-m*g*UH/r/Temp);
\]

\[
UP = users\_P + 12*randn(length(UH),1);
\]

phase_shift signal\_rx] = signal\_simulation(users\_location,UH,user\_freq, Fs,tx\_power,bit\_rate, 
surface, permit);

signals=[];

% Estimation through RSS

num_info_users=3; % number of users to take info from

for i=1:num_info_users
    [signal1 signal2]=QPSK\_filter(signal\_rx, Fs, bit\_rate, freq\_sep);
    signals(:,i)=signal1;
    signal\_rx=signal2;
end

freq=[];

for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end

for i=1:num_info_users
    rx\_power(i)=sum(signals(:,i).^2)/length(signals);
    rss\_dis(i)=C*(tx\_power).^0.5/((4*pi*(freq(i)))*(rx\_power(i)).^0.5);
end

rss\_users\_loc=[];

for i=1:num_info_users
    freq\_temp=find(abs(user\_freq-freq(i))<freq\_sep/2);
    rss\_users\_loc(i,:)=Rx\_locations(freq\_temp(1,:),);
    rss\_UP(i,:) = UP(freq\_temp(1)+1,:);
end

table02=table2D(num_info_users);  % table of possible combination of users for triangulation

rss_Tmp = m*g*rss_users_loc(:,3)./(r*(log(P0)-log(rss_UP)));
rss_user_T2 = temp_ftr(rss_Tmp);
rss_user_H2 = rss_user_T2.*r*(log(P0)-log(UP(1)))/(m*g);

est_loc_rss_ftr = location_estimation_ftr(rss_user_H2,rss_users_loc,rss_dis,table02);

Klmn_rss = location_estimation_klmn(est_loc_rss_ftr,LOCi_rss,PXi_rss,PYi_rss);
est_loc_rss_klmn = [Klmn_rss(1,1)  Klmn_rss(1,2)  Klmn_rss(1,3)];
LOCi_rss = Klmn_rss(:,1:3);
PXi_rss = Klmn_rss(:,4:5);
PYi_rss = Klmn_rss(:,6:7);

n=n+1;
if ((est_loc_rss_ftr(1)-est_loc_rss_klmn(1))^2 +
(est_loc_rss_ftr(2)-est_loc_rss_klmn(2))^2)^0.5 > 4
if n > 5
PXi_rss = PX0_rss;
PYi_rss = PY0_rss;
n = 0;
endif
endif

est_loc_rss_MMSE = MMSE(rss_users_loc,rss_dis);

E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
E1_ftr(3) = user_height(1,1)-est_loc_rss_ftr(:,3);
E1_klmn(1) = users_location(1,1)-est_loc_rss_klmn(:,1);
E1_klmn(2) = users_location(1,2)-est_loc_rss_klmn(:,2);
E1_klmn(3) = user_height(1,1)-est_loc_rss_klmn(:,3);
E1_MMSE(1) = users_location(1,1)-est_loc_rss_MMSE(:,1);
E1_MMSE(2) = users_location(1,2)-est_loc_rss_MMSE(:,2);
E1_MMSE(3) = user_height(1,1)-est_loc_rss_MMSE(:,3);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_klmn.^2));
error_rss(3,WKR)=sqrt(sum(E1_MMSE.^2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   End of RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Moving the mobile stations
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dis=((Distenation(:,1)-users_location(:,1)).^2+(Distenation(:,2)-
users_location(:,2)).^2).^0.5;
for i=1:num_users
if dis(i)<1
pause_time(i)=round(rand(1)*max_pause);
Distenation(i,:) = ((map_size-10)*rand(1,2))+5;
while surface(round(Distenation(i,:)/10)) ~= 0
  disp('Waiting for the surface to be clear...');
  pause(0.1);
end

  % Move the mobile station
  Distenation(i,:) = Distenation(i,:) + randn(1,2)*0.1;
end

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_klmn.^2));
error_rss(3,WKR)=sqrt(sum(E1_MMSE.^2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   End of RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Distenation(i,:) = ((map_size-10)*rand(1,2)+5);
end
velocity(i)=(rand(1)+0.4);*60;    % Avg human speed = 1.4 m/s = 84 m/min

else
    if pause_time>0
        pause_time(i)=pause_time(i)-1;
    else
        users_location(i,1) = users_location(i,1) + velocity(i) * (Distenation(i,1)-users_location(i,1))/dis(i);
        users_location(i,2) = users_location(i,2) + velocity(i) * (Distenation(i,2)-users_location(i,2))/dis(i);
    end
end
end

B.1.3. The Outdoor NLOS Case

B.1.3.1. 2D Case in an Outdoor NLOS Environment

a. 2D Case for Static Users in an Outdoor NLOS Environment

clear

surface = xlsread('surfaces.xls','Building');
permit = xlsread('surfaces.xls','Permittivity');    % Air, Concrete and Glass
%%
for WKR=1:100
num_users=22;
map_size = 1000;

users_location(1,:) = (map_size-200)*rand(1,2)+100;
while surface(ceil(users_location(1,:)/10)) ~= 0
    users_location(1,:) = (map_size-200)*rand(1,2)+100;
end

for i=1:num_users-1
    users_location(i+1,:) = 200*(rand(1,2)+(users_location(1,:)-
100)+1;
    while surface(ceil(users_location(i+1,:)/10)) ~= 0
        users_location(i+1,:) = 200*(rand(1,2)+(users_location(1,:)-
100)+1;
    end
end

height = zeros(2*map_size);
antenna_height = 1.5*ones(1,num_users);
freq_sep = 50*10^6; % frequency separation
start_freq = 850*10^6; % lowest frequency
user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
for i=2:num_users-1
    user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    end
end
precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458 ; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate
R = 0.03;
V = 40;
Random = mvnrnd([0 0],[V R;R V],num_users-1);
rx_locations = users_location(2:end,:) + Random; % adding an error element to the recived location information

% no_spoof = 2;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = users_location(2:no_spoof+1,:) +
60*rand(no_spoof,2); % adding a spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:) +
Random(no_spoof+1:end,:)]; % adding an error element to the recived location information
[phase_shift signal_rx] = signal_simulation(users_location, height,
antenna_height, user_freq, Fs, tx_power, bit_rate, surface, permit);
signals=[];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%   Estimation through RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
num_info_users=9; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
signals(:,i)=signal1;
signal_rx=signal2;
end
freq=[];
for i=1:num_info_users
    freqs=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
    rss_dis(i)=C*(tx_power).^0.5/(4*pi*(freq(i)))^0.5*(rx_power(i)).^0.5;
end
rss_users_loc=[];
for i=1:num_info_users  % Matching freqs to locations
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(1),:);
end

table01=table(num_info_users);  % table of possible combination of
                                % users for triangulation

est_loc_rss_ftr =
    location_estimation_ftr(rss_users_loc,rss_dis,table01);

E1_ftr(1) = users_location(1,1) - est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2) - est_loc_rss_ftr(:,2);

error_rss(1,WKR) = sqrt(sum(E1_ftr.^2));

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% End of RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

b. 2D Case for Mobile Users in an Outdoor NLOS Environment

clear

surface = xlsread('surfaces.xls','Building');
permit = xlsread('surfaces.xls','Permittivity'); % Air, Concrete and Glass

map_size = 1000;
num_users=22;

users_location(1,:) = (map_size-200)*rand(1,2)+100;
while surface(ceil(users_location(1,:)/10)) ~= 0
    users_location(1,:) = (map_size-200)*rand(1,2)+100;
end

for i=1:num_users-1
    users_location(i+1,:) = 200*(rand(1,2))+(users_location(1,:)-
                              100)+1;
    while surface(ceil(users_location(i,:)/10)) ~= 0
        users_location(i+1,:) = 200*(rand(1,2))+(users_location(1,:)-
                              100)+1;
    end
end

height = zeros(2*map_size);

% initialization for Kalman filter
LOCi_rss = [0 0 ; 0 0 ];
PXi_rss = [10^5 0; 0 10^5];
PYi_rss = [10^5 0; 0 10^5];
PX0_rss = PXi_rss;
PY0_rss = PYi_rss;
n=0;
m=0;

Destination=users_location;
velocity=zeros(num_users,1);
pause_time=zeros(num_users,1);
max_pause = 10; % Maximum pause time
% max_pause and velocity have to have the same time unit
%
for WKR=1:100
    antenna_height = 1.5*ones(1,num_users);
    freq_sep = 50*10^6; % frequency separation
    start_freq = 850*10^6; % lowest frequency
    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end
end

precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate

R = 0.03; % R
Random = mvnrnd([0 0],[1 R;R 1],num_users-1);
rx_locations = users_location(2:end,:) + Random; % adding an error
element to the recieved location information

% no_spoof = 2;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = users_location(2:no_spoof+1,:) +
% 60*rand(no_spoof,2); % adding a spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:) +
% Random(no_spoof+1:end,:)]; % adding an error element to the receieved
% location Information

[phase_shift signal_rx] = signal_simulation(users_location, height,
    antenna_height, user_freq, Fs, tx_power, bit_rate, surface, permit);
signals=[];

for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate,
        freq_sep);
    signals(:,i)=signal1;
    signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
freqs = find(abs(fft(signals(:,i))) == max(abs(fft(signals(:,i)))));
    freq(i) = freqs(1) * Fs / (length(signals));
  end
  for i = 1:num_info_users
      rx_power(i) = sum(signals(:,i).^2) / length(signals);
      rss_dis(i) = C * (tx_power).^0.5 / ((4 * pi * (freq(i))) * (rx_power(i)).^0.5);
  end
  rss_users_loc = [];
  for i = 1:num_info_users
      freq_temp = find(abs(user_freq - freq(i)) < freq_sep / 2);
      rss_users_loc(i,:) = rx_locations(freq_temp(1),:);
  end
  table01 = table(num_info_users);
  % table of possible combination of users for triangulation
  est_loc_rss_ftr = location_estimation_ftr(rss_users_loc, rss_dis, table01);
  Klmn_rss = location_estimation_klmn(est_loc_rss_ftr, LOCI_rss, PXi_rss, PYi_rss);
  est_loc_rss_klmn = [Klmn_rss(1,1) Klmn_rss(1,2)];
  LOCI_rss = Klmn_rss(:,1:2);
  PXi_rss = Klmn_rss(:,3:4);
  PYi_rss = Klmn_rss(:,5:6);
  n = n + 1;
  if ((est_loc_rss_ftr(1)-est_loc_rss_klmn(1))^2 +
     (est_loc_rss_ftr(2)-est_loc_rss_klmn(2))^2)^0.5 > 4
      if n > 5
        PXi_rss = PX0_rss;
        PYi_rss = PY0_rss;
        n = 0;
      end
  end
  E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
  E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
  E1_klmn(1) = users_location(1,1)-est_loc_rss_klmn(:,1);
  E1_klmn(2) = users_location(1,2)-est_loc_rss_klmn(:,2);
  error_rss(1,WKR) = sqrt(sum(E1_ftr.^2));
  error_rss(2,WKR) = sqrt(sum(E1_klmn.^2));
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   End of RSS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Moving the mobile stations
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dis = ((Distenation(:,1)-users_location(:,1)) .^ 2 + (Distenation(:,2)-users_location(:,2)) .^ 2).^0.5;
for i = 1:num_users
    if dis(i) < 1
        pause_time(i) = round(rand(1) * max_pause);
    end
end
Distenation(i,:) = (200-2)*(rand(1,2))+((800-2)*(rand(1,2))) + 2;
    while surface(round(Distenation(i,:)/10)) ~= 0
        Distenation(i,:) = (200-2)*(rand(1,2))+((800-2)*(rand(1,2))) + 2;
    end
velocity(i)=(rand(1)+0.4); %*60; % Avg human speed = 1.4 m/s = 84 m/min
else
    if pause_time > 0
        pause_time(i)=pause_time(i)-1;
    else
        users_lctn0 = users_location(i,:);
        users_location(i,:) = users_location(i,:) + velocity(i) * (Distenation(i,:) - users_location(i,:))/dis(i);
        if surface(round(users_location(i,:)/10)) ~= 0
            users_lctn1(:,1) = users_location(i,:);
            users_lctn2(:,1) = users_location(i,:);
            users_lctn3(:,1) = users_location(i,:);
            users_lctn4(:,1) = users_location(i,:);
            D = ((users_lctn0(1,1) - users_lctn1(1,1)).^2 + (users_lctn0(1,2) - users_lctn1(1,2)).^2).^0.5;
            n = 0;
            while surface(round(users_lctn1(1,:)/10)) ~= 0
                n=n+1;
                users_lctn1(1,2) = users_lctn0(1,2) + (D^2-(users_lctn1(1,1)+n-users_lctn0(1,1)).^2).^0.5;
                if isreal(users_lctn1(1,2))==0
                    users_lctn1(:,1)=users_lctn0;
                    break
                end
            end
            n = 0;
            while surface(round(users_lctn2(1,:)/10)) ~= 0
                n=n+1;
                users_lctn2(1,2) = users_lctn0(1,2) - (D^2-(users_lctn2(1,1)+n-users_lctn0(1,1)).^2).^0.5;
                if isreal(users_lctn2(1,2))==0
                    users_lctn2(:,1)=users_lctn0;
                    break
            end
            n = 0;
            while surface(round(users_lctn3(1,:)/10)) ~= 0
                n=n+1;
                users_lctn3(1,2) = users_lctn0(1,2) + (D^2-(users_lctn3(1,1)-n-users_lctn0(1,1)).^2).^0.5;
                if isreal(users_lctn3(1,2))==0
                    users_lctn3(:,1)=users_lctn0;
                    break
            end
            n = 0;
            while surface(round(users_lctn4(1,:)/10)) ~= 0
                n=n+1;
                users_lctn4(1,2) = users_lctn0(1,2) - (D^2-(users_lctn4(1,1)-n-users_lctn0(1,1)).^2).^0.5;
if isreal(users_lctn(4,2))==0
    users_lctn(4,:)=users_lctn0;
    break
end

dist(1) = ((Distenation(i,1) - users_lctn(1,1)).^2 + (Distenation(i,2) - users_lctn(1,2)).^2).^0.5;
+ (Distenation(i,2) - users_lctn(2,2)).^2).^0.5;
+ (Distenation(i,2) - users_lctn(3,3)).^2).^0.5;
+ (Distenation(i,2) - users_lctn(4,4)).^2).^0.5;
c = find(min(dist)==dist);
users_location(i,:) = users_lctn(c,:);
end
end
end

B.1.3.2. 3D Case in an Outdoor NLOS Environment

a. 3D Case for Static Users in an Outdoor NLOS Environment

clear

surface = xlsread('surfaces.xls','Building');
permit = xlsread('surfaces.xls','Permittivity'); % Air, Concrete and Glass

for WKR=1:100
    map_size = 1000;
    num_users=22;
    users_location(1,:) = (map_size-200)*rand(1,2)+10;
    while surface(ceil(users_location(1,:)/10)) ~= 0
        users_location(1,:) = (map_size-200)*rand(1,2)+10;
    end

    for i=1:num_users-1
        users_location(i+1,:) = 200*(rand(1,2))+(users_location(i,:)-100)+1;
        while surface(ceil(users_location(i,:)/10)) ~= 0
            users_location(i+1,:) = 200*(rand(1,2))+(users_location(i,:)-100)+1;
        end
    end

    height = 30*ones(2*map_size).*rand(2*map_size)+50;
for i=1:num_users
    user_height(i,1) = height(round(users_location(i,1)),round(users_location(i,2)));
end
antenna_height = 1.5*ones(1,num_users);
UH = user_height + antenna_height';
freq_sep = 50*10^6; % frequency separation
start_freq = 850*10^6; % lowest frequency
user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
for i=2:num_users-1
    user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
    end
end
precision=10^-10;
Fs=1/precision; % sampling frequency
C = 299792458; % Speed of Light (m/s)
tx_power=5; % transmission power=5 watt
bit_rate=50*10^6; % transmission bit rate

R = 0.03;
Random = mvnrnd([0 0 0],[1 R R;R 1 R;R R 1],num_users-1);
rx_locations = [users_location(2:end,:) UH(2:end)] + Random;

% no_spoof = 2;
% rx_locations = []; % rx_locations(1:no_spoof,:) = [users_location(2:no_spoof+1,:) UH(2:end)] + Random;

m = 0.0289644;
g = 9.80665;
r = 8.31447;
P0 = 101325;
Temp = 303.15; % temperature in kelvins = 30 C = 86 F
users_P = P0*exp(-m*g*UH/r/Temp);
UP = users_P + 12*randn(length(UH),1);

[phase_shift signal_rx] = signal_simulation(users_location,UH,user_freq, Fs,tx_power,bit_rate, surface, permit);
signals=[];
%%%%%% Estimation through RSS
num_info_users=3; % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK_filter(signal_rx, Fs, bit_rate, freq_sep);
signals(:,i)=signal1;
signal_rx=signal2;
end
freq=[];
for i=1:num_info_users
    freq=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);
    rss_dis(i)=C*(tx_power).^0.5/((4*pi*(freq(i)))*(rx_power(i)).^0.5);
end

rss_users_loc=[];
for i=1:num_info_users
    % Matching freqs to locations
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(1),:);
    rss_UP(i,:) = UP(freq_temp(1)+1,:);
end

% Table of possible combination of users for triangulation

rss_Tmp = m*g*rss_users_loc(:,3)./(r*(log(P0)-log(rss_UP)));

end

b. 3D Case for Mobile Users in an Outdoor NLOS Environment

clear

surface = xlsread('surfaces.xls','Building');
permit = xlsread('surfaces.xls','Permittivity'); % Air, Concrete and Glass

map_size = 1000;
num_users=22;
users_location(1,:) = (map_size-200)*rand(1,2)+100;
while surface(ceil(users_location(1,:)/10)) ~= 0
    users_location(1,:) = (map_size-200)*rand(1,2)+100;
end

for i=1:num_users-1
users_location(i+1,:) = 200*(rand(1,2))+(users_location(1,:)-100)+1;
    while surface(ceil(users_location(i,:)/10)) ~= 0
        users_location(i+1,:) = 200*(rand(1,2))+(users_location(1,:)-100)+1;
    end
end

height = 30*ones(2*map_size).*(rand(2*map_size))+50;

% initialization for Kalman filter
LOCi_rss = [0 0 0; 0 0 0];
PXi_rss = [10^5 0; 0 10^5];
PYi_rss = [10^5 0; 0 10^5];
PX0_rss = PXi_rss;
PY0_rss = PYi_rss;
n=0;
M=0;

freq_sep = 50*10^6 ; % frequency seperation
start_freq = 850*10^6; % lowest frequency

Destination=users_location;
velocity=zeros(num_users,1);
pause_time=zeros(num_users,1);
max_pause = 10; % Maximum pause time
%max_pause and velocity have to have the same time unit
%%
for WKR=1:100
    for i=1:num_users
        user_height(i,1) = height(round(users_location(i,1)),round(users_location(i,2)));
    end
    antenna_height = 1.5*ones(num_users,1);
    UH = user_height + antenna_height;

    user_freq(1) = start_freq + freq_sep*round(20*rand(1,1));
    for i=2:num_users-1
        user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        while sum(ismember(user_freq(1:i-1),user_freq(i)))>0
            user_freq(i) = start_freq + freq_sep*round(20*rand(1,1));
        end
    end

    precision=10^-10;
    Fs=1/precision; % sampling frequency
    C = 299792458 ; % Speed of Light (m/s)
    tx_power=5; % transmission power=5 watt
    bit_rate=50*10^6; % transmission bit rate
    R = 0.03;
    Random = mvnrnd([0 0 0],[1 R R;R 1 R;R R 1],num_users-1);
    rx_locations = [users_location(2:end,: user_height(2:end)) + Random; % adding an error element to the recived location information
% no_spoof = 2;
% rx_locations = [];
% rx_locations(1:no_spoof,:) = [users_location(2:no_spoof+1,:)
user_height(2:no_spoof+1,:),] + 20*rand(no_spoof,3);  % adding a
spoofing user
% rx_locations = [rx_locations; users_location(no_spoof+2:end,:)
user_height(no_spoof+2:end)] + Random;  % adding an error element to
the recieved location information

m = 0.0289644;
g = 9.80665;
r = 8.31447;
P0 = 101325;
Temp = 303.15;  % temperature in kelvins = 30 C = 86 F
users_P = P0*exp(-m*g*UH/r/Temp);
UP = users_P + 12*randn(length(UH),1);

[phase_shift signal_rx] =
signal_simulation(users_location,UH,user_freq, Fs,tx_power,bit_rate,
surface, permit);
signals=[];
% Estimation through RSS
num_info_users=7;  % number of users to take info from
for i=1:num_info_users
    [signal1 signal2]=QPSK filter(signal_rx, Fs, bit_rate, freq_sep);
signals(:,i)=signal1;
signal_rx=signal2;
end

freq=[];
for i=1:num_info_users
    freq=find(abs(fft(signals(:,i)))==max(abs(fft(signals(:,i)))));
    freq(i)=freqs(1)*Fs/(length(signals));
end
for i=1:num_info_users
    rx_power(i)=sum(signals(:,i).^2)/length(signals);  
    rss_dis(i)=C*(tx_power).^0.5/(4*pi*(freq(i)))*(rx_power(i)).^0.5);
end

rss_users_loc=[];
for i=1:num_info_users  % Matching freqs to locations
    freq_temp=find(abs(user_freq-freq(i))<freq_sep/2);
    rss_users_loc(i,:)=rx_locations(freq_temp(1,:),);
    rss_UP(i,:) = UP(freq_temp(1)+1,:);
end
table01=table(num_info_users);  % table of possible combination of
users for triangulation

% Matching users for triangulation

% table2D(num_info_users);  % table of possible combination of
users for triangulation

rss_Tmp = m*g*rss_users_loc(:,3)./(r*(log(P0)-log(rss_UP)));
rss_user_T2 = temp_ftr(rss_Tmp);
rss_user_H2 = rss_user_T2*r*{(log(P0)-log(UP(1)))/(m*g)};
est_loc_rss_ftr =
location_estimation_ftr(rss_user_H2,rss_users_loc,rss_dis,table02);

Klmn_rss =
location_estimation_klmn(est_loc_rss_ftr,LOCi_rss,PXi_rss,PYi_rss);
est_loc_rss_klmn = [Klmn_rss(1,1) Klmn_rss(1,2) Klmn_rss(1,3)];
LOCi_rss = Klmn_rss(:,1:3);
PXi_rss = Klmn_rss(:,4:5);
PYi_rss = Klmn_rss(:,6:7);

n=n+1;
if ((est_loc_rss_ftr(1)-est_loc_rss_klmn(1))^2 +
    (est_loc_rss_ftr(2)-est_loc_rss_klmn(2))^2)^0.5 > 4
    if n > 5
        PXi_rss = PX0_rss;
PYi_rss = PY0_rss;
n = 0;
    end
end

est_loc_rss_MMSE = MMSE(rss_users_loc,rss_dis);
E1_ftr(1) = users_location(1,1)-est_loc_rss_ftr(:,1);
E1_ftr(2) = users_location(1,2)-est_loc_rss_ftr(:,2);
E1_ftr(3) = user_height(1,1)-est_loc_rss_ftr(:,3);
E1_klmn(1) = users_location(1,1)-est_loc_rss_klmn(:,1);
E1_klmn(2) = users_location(1,2)-est_loc_rss_klmn(:,2);
E1_klmn(3) = user_height(1,1)-est_loc_rss_klmn(:,3);
E1_MMSE(1) = users_location(1,1)-est_loc_rss_MMSE(:,1);
E1_MMSE(2) = users_location(1,2)-est_loc_rss_MMSE(:,2);
E1_MMSE(3) = user_height(1,1)-est_loc_rss_MMSE(:,3);

error_rss(1,WKR)=sqrt(sum(E1_ftr.^2));
error_rss(2,WKR)=sqrt(sum(E1_klmn.^2));
error_rss(3,WKR)=sqrt(sum(E1_MMSE.^2));
% End of RSS
% Moving the mobile stations
Dis=((Destenation(:,1)-users_location(:,1)).^2+(Destenation(:,2)-users_location(:,2)).^2).^0.5;
for i=1:num_users
    if dis(i)<1
        pause_time(i)=round(rand(1)*max_pause);
        Destenation(i,:) = (200-2)*rand(1,2)+((800-2)*rand(1,2))+2;
        while surface(round(Destenation(i,:)/10)) ~= 0
            Destenation(i,:) = (200-2)*rand(1,2)+((800-2)*rand(1,2))+2;
        end
        velocity(i)=(rand(1)+0.4).*60; % Avg human speed = 1.4 m/s = 84 m/min
    else
        if pause_time>0
            % do nothing
        end
    end
end
\[
\text{pause}_{-}\text{time}(i) = \text{pause}_{-}\text{time}(i) - 1;
\]

\[
\text{else}
\]

\[
\text{users}_{-}\text{lctn}0 = \text{users}_{-}\text{location}(i,:);
\]

\[
\text{users}_{-}\text{location}(i,:) = \text{users}_{-}\text{location}(i,:) + \text{velocity}(i) \times (\text{Destination}(i,:) - \text{users}_{-}\text{location}(i,:)) / \text{dis}(i);
\]

\[
\text{if } \text{surface}(\text{round}(\text{users}_{-}\text{location}(i,:)/10)) = 0
\]

\[
\text{users}_{-}\text{lctn}(1,:) = \text{users}_{-}\text{location}(i,:);
\]

\[
\text{users}_{-}\text{lctn}(2,:) = \text{users}_{-}\text{location}(i,:);
\]

\[
\text{users}_{-}\text{lctn}(3,:) = \text{users}_{-}\text{location}(i,:);
\]

\[
\text{users}_{-}\text{lctn}(4,:) = \text{users}_{-}\text{location}(i,:);
\]

\[
D = ((\text{users}_{-}\text{lctn}(1,1) - \text{users}_{-}\text{lctn}(1,1))^2 + (\text{users}_{-}\text{lctn}(1,2) - \text{users}_{-}\text{lctn}(1,2))^2)^{0.5} -
\]

\[
\text{n} = 0;
\]

\[
\text{while } \text{surface}(\text{round}(\text{users}_{-}\text{lctn}(1,:)/10)) = 0
\]

\[
\text{n} = n + 1;
\]

\[
\text{users}_{-}\text{lctn}(1,2) = \text{users}_{-}\text{lctn}(1,2) + (D^2 - (\text{users}_{-}\text{lctn}(1,1) + n - \text{users}_{-}\text{lctn}(1,1))^2)^{0.5};
\]

\[
\text{if } \text{isreal}(\text{users}_{-}\text{lctn}(1,2)) = 0
\]

\[
\text{users}_{-}\text{lctn}(1,:) = \text{users}_{-}\text{lctn}0;
\]

\[
\text{break}
\]

\end{while}

\]

\[
\text{n} = 0;
\]

\[
\text{while } \text{surface}(\text{round}(\text{users}_{-}\text{lctn}(2,:)/10)) = 0
\]

\[
\text{n} = n + 1;
\]

\[
\text{users}_{-}\text{lctn}(2,2) = \text{users}_{-}\text{lctn}(1,2) - (D^2 - (\text{users}_{-}\text{lctn}(2,1) + n - \text{users}_{-}\text{lctn}(1,1))^2)^{0.5};
\]

\[
\text{if } \text{isreal}(\text{users}_{-}\text{lctn}(2,2)) = 0
\]

\[
\text{users}_{-}\text{lctn}(2,:) = \text{users}_{-}\text{lctn}0;
\]

\[
\text{break}
\]

\end{while}

\]

\[
\text{n} = 0;
\]

\[
\text{while } \text{surface}(\text{round}(\text{users}_{-}\text{lctn}(3,:)/10)) = 0
\]

\[
\text{n} = n + 1;
\]

\[
\text{users}_{-}\text{lctn}(3,2) = \text{users}_{-}\text{lctn}(1,2) + (D^2 - (\text{users}_{-}\text{lctn}(3,1) - n - \text{users}_{-}\text{lctn}(1,1))^2)^{0.5};
\]

\[
\text{if } \text{isreal}(\text{users}_{-}\text{lctn}(3,2)) = 0
\]

\[
\text{users}_{-}\text{lctn}(3,:) = \text{users}_{-}\text{lctn}0;
\]

\[
\text{break}
\]

\end{while}

\]

\[
\text{n} = 0;
\]

\[
\text{while } \text{surface}(\text{round}(\text{users}_{-}\text{lctn}(4,:)/10)) = 0
\]

\[
\text{n} = n + 1;
\]

\[
\text{users}_{-}\text{lctn}(4,2) = \text{users}_{-}\text{lctn}(1,2) - (D^2 - (\text{users}_{-}\text{lctn}(4,1) - n - \text{users}_{-}\text{lctn}(1,1))^2)^{0.5};
\]

\[
\text{if } \text{isreal}(\text{users}_{-}\text{lctn}(4,2)) = 0
\]

\[
\text{users}_{-}\text{lctn}(4,:) = \text{users}_{-}\text{lctn}0;
\]

\[
\text{break}
\]

\end{while}

\]

\[
\text{dist}(1) = ((\text{Destination}(i,1) - \text{users}_{-}\text{lctn}(1,1))^2 + (\text{Destination}(i,2) - \text{users}_{-}\text{lctn}(1,2))^2)^{0.5};
\]

\[
\text{dist}(2) = ((\text{Destination}(i,1) - \text{users}_{-}\text{lctn}(2,1))^2 + (\text{Destination}(i,2) - \text{users}_{-}\text{lctn}(2,2))^2)^{0.5};
\]
\[
\text{dist}(3) = ((\text{Destination}(i,1) - \text{users\_lctn}(3,1))^2 + (\text{Destination}(i,2) - \text{users\_lctn}(3,2))^2)^{0.5};
\]

\[
\text{dist}(4) = ((\text{Destination}(i,1) - \text{users\_lctn}(4,1))^2 + (\text{Destination}(i,2) - \text{users\_lctn}(4,2))^2)^{0.5};
\]

\[
c = \text{find}(\text{min}(\text{dist}) == \text{dist});
\]

\[
\text{users\_location}(i,:) = \text{users\_lctn}(c,:);
\]

\]

B.2. table.m

This subroutine is used to create a list of all possible user combinations.

```matlab
function table01=table(num_users)
    table01=[];
    for i=1:num_users
        for j=1:num_users
            if i==j
                continue
            else
                for k=1:num_users
                    if j==k
                        continue
                    elseif i==k
                        continue
                    else
                        test01=ismember(table01,[i j k]);
                        test02=sum(test01,2);
                        test03=ismember(test02,3);
                        if sum(test03)>0
                            continue
                        else
                            table01=[table01;i j k];
                            end
                        end
                    end
                end
            end
        end
    end
end
```
B.3. location_estimation_ftr.m

These subroutine are the part where the actual location estimation happens. The 3D case differs since the extra element of height is present.

B.3.1. 2D Cases

```matlab
function LOC_est=location_estimation_ftr(rx_locations,rss_dis,table01)

[R C]=size(table01); %#ok<NASGU>
for i=1:R
    x1=rx_locations(table01(i,1),1);
    x2=rx_locations(table01(i,2),1);
    x3=rx_locations(table01(i,3),1);
    y1=rx_locations(table01(i,1),2);
    y2=rx_locations(table01(i,2),2);
    y3=rx_locations(table01(i,3),2);
    D1=rss_dis(table01(i,1));
    D2=rss_dis(table01(i,2));
    D3=rss_dis(table01(i,3));

    LOC=[2*(x2-x1) 2*(y2-y1) 2*(x3-x1) 2*(y3-y1); D1^2-D2^2-x1^2+x2^2-y1^2+y2^2; D1^2-D3^2-x1^2+x3^2-y1^2+y3^2];

    loc_est(i,1)=double(LOC(1));
    loc_est(i,2)=double(LOC(2));
end
for i = length(loc_est(:,1)):-1:1
    if isnan(loc_est(i,1)) == 1 || loc_est(i,1) == Inf || loc_est(i,1) == -Inf
        loc_est(i,:)=[];
    end
end

loc = sort(loc_est(:,1));
X = hist(loc,round((length(loc))^0.5));
MX = max(X);
for i=1:length(X)
    if X(i)==MX
        BX = sum(X(1:i-1));
    end
end
```
if isempty(BX)==1 || BX<0
disp(BX)
end

XX = loc(BX+1:BX+MX);
for i=length(loc):-1:1
   if sum(loc_est(i,1)==XX)==0
      loc_est(i,:)=[ ];
   end
end

loc = sort(loc_est(:,1));
X = hist(loc,round((length(loc))^0.5));
MX = max(X);
for i=1:length(X)
   if X(i)==MX
      BX = sum(X(1:i-1));
   end
end

if isempty(BX)==1 || BX<0
disp(BX)
end

XX = loc(BX+1:BX+MX);
for i=length(loc):-1:1
   if sum(loc_est(i,1)==XX)==0
      loc_est(i,:)=[ ];
   end
end

LOC_est(1) = sum(loc_est(:,1))/(length(loc_est(:,1)));
LOC_est(2) = sum(loc_est(:,2))/(length(loc_est(:,2)));

B.3.2. 3D Cases

function
LOC_est=location_estimation_ftr(Z0,rx_locations,rss_dis,table01)

[R C]=size(table01); %#ok<NASGU>
for i=1:R
   x1=rx_locations(table01(i,1),1);
   x2=rx_locations(table01(i,2),1);
   x3=rx_locations(table01(i,3),1);
   y1=rx_locations(table01(i,1),2);
   y2=rx_locations(table01(i,2),2);
   y3=rx_locations(table01(i,3),2);
   z1=rx_locations(table01(i,1),3);
   z2=rx_locations(table01(i,2),3);
z3 = rx_locations(table01(i,3),3);
D1 = rss_dis(table01(i,1));
D2 = rss_dis(table01(i,2));
D3 = rss_dis(table01(i,3));

LOC = [2*(x2-x1) 2*(y2-y1) 2*(x3-x1) 2*(y3-y1)]...
     [D1^2-D2^2-x1^2+y2^2-z1^2+z2^2-20*2*(z2-z1); D1^2-
      D3^2-x1^2+x3^2-y1^2+y3^2-z1^2+z3^2-20*2*(z3-z1)];

loc_est(i,1) = double(LOC(1));
loc_est(i,2) = double(LOC(2));

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for i = length(loc_est(:,1)):-1:1
    if isnan(loc_est(i,1)) == 1 || loc_est(i,1) == Inf || loc_est(i,1)
      == -Inf
        loc_est(i,:) = [];
    end
end

loc = sort(loc_est(:,1));
X = hist(loc,round((length(loc))^0.5));
MX = max(X);
for i=1:length(X)
    if X(i)==MX
        BX = sum(X(1:i-1));
    end
end

if isempty(BX)==1 || BX<0
    disp(BX)
end

XX = loc(BX+1:BX+MX);
for i=length(loc):-1:1
    if sum(loc_est(i,1)==XX)==0
        loc_est(i,:)=[];
    end
end

loc = sort(loc_est(:,2));
X = hist(loc,round((length(loc))^0.5));
MX = max(X);
for i=1:length(X)
    if X(i)==MX
        BX = sum(X(1:i-1));
    end
end

if isempty(BX)==1 || BX<0
    disp(BX)
end
XX = loc(BX+1:BX+MX);
for i=length(loc):-1:1
    if sum(loc_est(i,2)==XX)==0
        loc_est(i,:)=[];
    end
end

LOC_est(1) = sum(loc_est(:,1))/(length(loc_est(:,1)));
LOC_est(2) = sum(loc_est(:,2))/(length(loc_est(:,2)));
LOC_est(3) = Z0;

B.4. QPSK_mod.m

Part of the signal simulation, QPSK_mod.m is where the signal is modulated using QPSK.

function QPSK_signal = QPSK_mod(bit_stream,Fs,fc,Eb,bit_rate)

t = 1/Fs : 1/Fs : length(bit_stream)/bit_rate;

for i=1:length(bit_stream)
    if bit_stream(i) == 0
        if mod(i,2)==0
            odd_data_stream(Fs/bit_rate*(i-1)+1:Fs/bit_rate*(i+1)) = -(Eb/2)^0.5;
        else
            even_data_stream(Fs/bit_rate*(i-2)+1:Fs/bit_rate*i) = -(Eb/2)^0.5;
        end
    elseif bit_stream(i) == 1
        if mod(i,2)==0
            odd_data_stream(Fs/bit_rate*(i-1)+1:Fs/bit_rate*(i+1)) = (Eb/2)^0.5;
        else
            even_data_stream(Fs/bit_rate*(i-2)+1:Fs/bit_rate*i) = (Eb/2)^0.5;
        end
    end
end

basic_function1 = (2*bit_rate)^0.5 .* cos(2*pi*fc.*t);
basic_function2 = (2*bit_rate)^0.5 .* sin(2*pi*fc.*t);
QPSK_signal = odd_data_stream .* basic_function1 + even_data_stream .* basic_function2;

band1=[fc-bit_rate/2 fc+bit_rate/2]; % BW for QPSK = bit rate
band1=2*band1/Fs;
[b1,a1]=butter(5,band1);
QPSK_signal = filter(b1,a1,QPSK_signal);

B.5. QPSK_filter.m

To separate the different signals from the different users the program uses QPSK_filter.m.

function [signal1 signal2]=QPSK_filter(signal, Fs, bit_rate, freq_sep)

signal_m=signal;
signal_f=fft(signal_m);

f1=find(abs(signal_f(1,:))==max(abs(signal_f(1,:))));
f1=f1(1)*Fs/(length(signal));
b1=band=[f1-freq_sep/2 f1+freq_sep/2];       % frequency seperation = 50 MHz
band1=2*band1/Fs;
[b1,a1]=butter(5,band1);
signal1=filter(b1,a1,signal_m);

F1 = carrier_recovery(signal1,f1,Fs);

band1=[F1-bit_rate/2 F1+bit_rate/2];       % BW for QPSK = bit rate
band1=2*band1/Fs;
[b1,a1]=butter(5,band1);
signal1=filter(b1,a1,signal_m);

[b2,a2]=butter(5,band1,'stop');
signal2=filter(b2,a2,signal_m);

B.6. carrier_recovery.m

The signal received by the target user arrives degraded. To properly find the correct carrier of a signal, carrier_recovery.m is used based on Costas recovery circuit.

function Fc = carrier_recovery(QPSK_signal,fc,Fs)

t=1/Fs : 1/Fs : length(QPSK_signal)/Fs;
Fc1 = [fc 0 0; fc 0 0];
VCO_freq = 1.005*fc;          % to combat the high spikes if both
                        % frequencies are exactly matched
for n=1:3

    tempI=QPSK_signal.*sin(2*pi*VCO_freq*t);
    tempQ=QPSK_signal.*cos(2*pi*VCO_freq*t);

    Wn1 = 0.05 ;
    [b1,a1]=butter(5,Wn1,'low');
    I = filter(b1,a1,tempI);
    [b2,a2]=butter(5,Wn1,'low');
    Q = filter(b2,a2,tempQ);

    IL = zeros(1,length(I)) ;
    QL = zeros(1,length(I)) ;

    if max(I)<=1
        threshold = 0.5*max(I);
    else
        threshold = 1;
    end

    for i=1:length(I)
        if I(i)>threshold
            IL(i)=threshold;
        elseif I(i)<-threshold
            IL(i)=-threshold;
        else
            IL(i)=I(i);
        end
        if Q(i)>threshold
            QL(i)=threshold;
        elseif Q(i)<-threshold
            QL(i)=-threshold;
        else
            QL(i)=Q(i);
        end
    end

    IQ1 = I .* QL;
    IQ2 = IL .* Q;
    IQ = IQ1 - IQ2;

    Wn2 = 0.05 ;
    [b3,a3]=butter(5,Wn2,'low');
    VCO_in = filter(b3,a3,IQ);
    VCO_in_F=abs(fft(VCO_in));

    [r1 c1] = find(VCO_in_F==max(VCO_in_F));
    Df = ((c1(1)-1)/4)*Fs/length(VCO_in_F);

    Fc1(:,n) = [VCO_freq + Df; VCO_freq - Df];
    if n==3
continue
else
    VCO_freq = 1.005*Fc1(n,1);
end

Fc2(1,1) = abs(Fc1(1,1) - Fc1(1,2)) ;
Fc2(1,2) = abs(Fc1(1,1) - Fc1(2,2)) ;
Fc2(1,3) = abs(Fc1(1,1) - Fc1(1,3)) ;
Fc2(1,4) = abs(Fc1(1,1) - Fc1(2,3)) ;
Fc2(2,1) = abs(Fc1(2,1) - Fc1(1,2)) ;
Fc2(2,2) = abs(Fc1(2,1) - Fc1(2,2)) ;
Fc2(2,3) = abs(Fc1(2,1) - Fc1(1,3)) ;
Fc2(2,4) = abs(Fc1(2,1) - Fc1(2,3)) ;

[r2 c2] = find(Fc2==min(min(Fc2))) ; %#ok<NASGU>

Fc = Fc1(r2(1),1) ;

B.7. signal_simulation.m

B.7.1. 2D LOS Cases

function [phase_shift signal_rx] = signal_simulation(LOC, Height, ANT_H8, fc, Fs, tx_power, bit_rate)

Num03 = size(LOC);
len = Num03(1);

for i = 2:len
    D(i-1)= sqrt((LOC(i,1)-LOC(1,1))^2+(LOC(i,2)-LOC(1,2))^2+...
    (ANT_H8(i)+Height(round(LOC(i,1)),round(LOC(i,2))))-
    ANT_H8(1)...-
    Height(round(LOC(1,1)),round(LOC(1,2))))^2);
end

bit_stream = round(rand(len,1000));
C = 299792458 ; % Speed of Light (m/s)
lamda = C./fc;
P_Tx = tx_power*ones(1,length(lamda));
P_Rx = P_Tx .* (lamda./(4.*pi.*D)).^2;
phase_shift = D./C;
Eb = P_Rx/bit_rate;
MX=max(phase_shift);
signal_rx = zeros(1,round((length(bit_stream(1,:))/bit_rate+MX)*Fs));

for n=1:length(P_Rx)
    QPSK_signal = QPSK_mod(bit_stream(n,:),Fs,fc(n),Eb(n),bit_rate);
signal_rx = signal_rx + [zeros(1,round(phase_shift(n)*Fs)) QPSK_signal zeros(1,length(signal_rx)-length(QPSK_signal)-round(phase_shift(n)*Fs))];
end

signal_rx(round(length(bit_stream(1,:))/bit_rate*Fs)+1:end) = [];

B.7.2. 2D NLOS Cases

function [phase_shift signal_rx] = signal_simulation(LOC, Height, ANT_H8, fc, Fs, tx_power, bit_rate, surface, permit)

Num03 = size(LOC);
len = Num03(1);
for i = 2:len
    D(i-1)= sqrt((LOC(i,1)-LOC(1,1))^2+(LOC(i,2)-LOC(1,2))^2+...
                 (ANT_H8(i)+Height(round(LOC(i,1)),round(LOC(i,2))))-
                 ANT_H8(1)...-
                 Height(round(LOC(1,1)),round(LOC(1,2))))^2);
end

bit_stream = round(rand(len,1000));
C = 299792458 ; % Speed of Light (m/s)
lamda = C./fc;
P_Tx1 = tx_power*ones(1,length(lamda));
for i=1:length(P_Tx1)
P_Tx(i) = NLOS(P_Tx1(i), LOC(1,:), LOC(1+i,:),surface, permit);
end
P_Rx = P_Tx .* (lamda./(4.*pi.*D)).^2;
phase_shift = D./C;
Eb = P_Rx./bit_rate;
% MX=max(phase_shift);
MX = 10 * max(phase_shift);
signal_rx = zeros(1,round((length(bit_stream(1,:))/bit_rate+MX)*Fs));
for n=1:length(P_Rx)
    QPSK_signal = QPSK_mod(bit_stream(n,:),Fs,fc(n),Eb(n),bit_rate);
signal_rx = signal_rx + [zeros(1,round(phase_shift(n)*Fs)) QPSK_signal zeros(1,length(signal_rx)-length(QPSK_signal)-round(phase_shift(n)*Fs))];
end
[Pr DIS] = multi_path(LOC(1,:), LOC(n+1,:), tx_power, surface, permit);
phase_shift_multi = DIS./C;
P_Rx_multi = Pr .* (lamda(n)./(4.*pi.*DIS)).^2;
Eb_multi = P_Rx_multi./bit_rate;
QPSK_signal_multi = QPSK_mod(bit_stream(n,:),Fs,fc(n),Eb_multi,bit_rate);
signal_rx = signal_rx - [zeros(1,round(phase_shift_multi*Fs))
QPSK_signal_multi zeros(1,length(signal_rx)-length(QPSK_signal_multi)-
round(phase_shift_multi*Fs))];
    % minus sign for 180 degree shift when signal is reflected
end

signal_rx(round(length(bit_stream(1,:))/bit_rate*Fs)+1:end) = [];

B.7.3. 3D LOS Cases

function [phase_shift signal_rx] = signal_simulation(LOC, Height,
ANT_H8, fc, Fs, tx_power, bit_rate)
Num03 = size(LOC);
len = Num03(1);
for i = 2:len
    D(i-1)= sqrt((LOC(i,1)-LOC(1,1))^2+(LOC(i,2)-LOC(1,2))^2+
        ... +
        (ANT_H8(i)+Height(round(LOC(i,1)),round(LOC(i,2)))-
        ANT_H8(1)-
        ...-Height(round(LOC(1,1)),round(LOC(1,2))))^2);
end

bit_stream = round(rand(len,1000));
C = 299792458 ; % Speed of Light (m/s)
lamda = C./fc;
P_Tx = tx_power*ones(1,length(lamda));
P_Rx = P_Tx .* (lamda./(4.*pi.*D)).^2;
phase_shift = D./C;
Eb = P_Rx/bit_rate;
MX=max(phase_shift);
signal_rx = zeros(1,round((length(bit_stream(1,:))/bit_rate+MX)*Fs));

for n=1:length(P_Rx)
    QPSK_signal = QPSK_mod(bit_stream(n,:),Fs,fc(n),Eb(n),bit_rate);
    signal_rx = signal_rx + [zeros(1,round(phase_shift(n)*Fs))
QPSK_signal zeros(1,length(signal_rx)-length(QPSK_signal)-
round(phase_shift(n)*Fs))];
end

signal_rx(round(length(bit_stream(1,:))/bit_rate*Fs)+1:end) = [];

B.7.4. 3D NLOS Cases

function [phase_shift signal_rx] = signal_simulation(LOC, Height,
ANT_H8, fc, Fs, tx_power, bit_rate, surface, permit)
Num03 = size(LOC);
len = Num03(1);
\begin{verbatim}
for i = 2:len
    D(i-1)= sqrt((LOC(i,1)-LOC(1,1))^2+(LOC(i,2)-LOC(1,2))^2+...
    (ANT_H8(i)+Height(round(LOC(i,1)),round(LOC(i,2))))^2-
    ANT_H8(1)...-
    Height(round(LOC(1,1)),round(LOC(1,2))))^2);
end

bit_stream = round(rand(len,1000));
C = 299792458 ; \% Speed of Light (m/s)
lamda = C./fc;
P.Tx1 = tx_power*ones(1,length(lamda));

for i=1:length(P.Tx1)
    P.Tx(i) = NLOS(P.Tx1(i), LOC(1,:), LOC(1+i,:),surface, permit);
end

P.Rx = P.Tx .* (lamda./(4.*pi.*D)).^2;
phase_shift = D./C;
Eb = P.Rx/bit_rate;
\% MX=max(phase_shift);
MX = 10 * max(phase_shift);
signal_rx = zeros(1,round((length(bit_stream(:,1))/bit_rate+MX)*Fs));

for n=1:length(P.Rx)
    QPSK_signal = QPSK_mod(bit_stream(n,:),Fs,fc(n),Eb(n),bit_rate);
    signal_rx = signal_rx + [zeros(1,round(phase_shift(n)*Fs))
    QPSK_signal zeros(1,length(signal_rx)-length(QPSK_signal)-
    round(phase_shift(n)*Fs))];
    [Pr DIS] = multi_path(LOC(1,:), LOC(n+1,:), tx_power, surface, permit);
    phase_shift_multi = DIS./C;
P.Rx_multi = Pr .* (lamda(n)./(4.*pi.*DIS)).^2;
    Eb_multi = P.Rx Multi/bit_rate;
    QPSK_signal_multi = QPSK_mod(bit_stream(n,:),Fs,fc(n),Eb_multi,bit_rate);
    signal_rx = signal_rx - [zeros(1,round(phase_shift_multi*Fs))
    QPSK_signal_multi zeros(1,length(signal_rx)-length(QPSK_signal_multi)-
    round(phase_shift_multi*Fs))];
    \% minus sign for 180 degree shift when signal is reflected
    PP(n) = P.Rx_multi;
end

signal_rx(round(length(bit_stream(:,1))/bit_rate*Fs)+1:end) = [];

NUM = 3;
PRP = sort(P.Tx,'descend');
disp([sum(-1*(floor(P.Tx/tx_power)-1))/length(P.Rx);
sum(PP)/length(P.Rx); sum(-1*(floor(PRP(1:NUM)/tx_power)-1))/NUM])
\end{verbatim}
B.8. location_estimation_klmn.m

Location estimation using Kalman filter.

### B.8.1. 2D Cases

```matlab
function FF = location_estimation_klmn(LOC_e,LOCi,PXi,PYi)
    warning off all
    x00 = LOCi(:,1);
y00 = LOCi(:,2);
A = [1 1;0 1];
C = [1 0];
I = [1 0;0 1];

    xn0 = A*x00;
yn0 = A*y00;
    PX0 = A*PXi*A';
    PY0 = A*PYi*A';

    KX = PX0*C'/(C*PX0*C'+1);
    KY = PY0*C'/(C*PY0*C'+1);

    xn1 = xn0+KX*(LOC_e(1)-C*xn0);
yn1 = yn0+KY*(LOC_e(2)-C*yn0);

    PX = (I-KX*C)*PX0;
    PY = (I-KY*C)*PY0;

    LOC_est = [xn1 yn1];
    FF = [LOC_est PX PY];
```

### B.8.2. 3D Cases

```matlab
function FF = location_estimation_klmn(LOC_e,LOCi,PXi,PYi)
    x00 = LOCi(:,1);
y00 = LOCi(:,2);
A = [1 1;0 1];
C = [1 0];
I = [1 0;0 1];

    xn0 = A*x00;
yn0 = A*y00;
```
PX0 = A*PXi*A';
PY0 = A*PYi*A';

KX = PX0*C'/(C*PX0*C'+1);
KY = PY0*C'/(C*PY0*C'+1);

xn1 = xn0+KX*(LOC_e(1)-C*xn0);
yn1 = yn0+KY*(LOC_e(2)-C*yn0);
zn1 = [LOC_e(3); 0];

PX = (I-KX*C)*PX0;
PY = (I-KY*C)*PY0;

LOC_est = [xn1 yn1 zn1];
FF = [LOC_est PX PY];

B.9. NLOS.m

These subroutines are used to simulate the signal passing through obstacles.

B.9.1. 2D Cases

function P_Tx1 = NLOS(P_Tx, user_loc, user_2, surface, permit)
slope=(user_loc(2)-user_2(2))/(user_loc(1)-user_2(1));
srf = 0;
if abs(slope)<1
    for x=round(user_loc(1)):(user_2(1)-round(user_loc(1)))/abs(user_2(1)-round(user_loc(1))):user_2(1)
        y=slope*(x-user_2(1))+user_2(2);
        if surface(x, ceil(y))~=0
            point = [x, ceil(y)];
            srf = surface(x, ceil(y));
            break
        end
    end
else
    if slope==0
        slope1=inf;
    else
        slope1=1/slope;
    end
    for y=round(user_loc(2)):(user_2(2)-round(user_loc(2)))/abs(user_2(2)-round(user_loc(2))):user_2(2)
        x=slope1*(y-user_2(2))+user_2(1);
        if surface(ceil(x), y)==0
            point = [ceil(x), y];
            srf = surface(ceil(x), y);
            break
        end
    end
end
if srf==0
    P_Tx1 = P_Tx;
else
    if surface(point(1)+1,point(2))==0
        theta = atan(abs(user_loc(2)-point(2))/abs(user_loc(1)-point(1)));
    elseif surface(point(1)-1,point(2))==0
        theta = atan(abs(user_loc(2)-point(2))/abs(user_loc(1)-point(1)));
    elseif surface(point(1),point(2)+1)==0
        theta = atan(abs(user_loc(1)-point(1))/abs(user_loc(2)-point(2)));
    elseif surface(point(1),point(2)-1)==0
        theta = atan(abs(user_loc(1)-point(1))/abs(user_loc(2)-point(2)));
    end

    reflaction_coeff_parallel = (-permit(srf+1).*sin(theta)+(permit(srf+1)-(cos(theta)).^2).^0.5)./(permit(srf+1).*sin(theta)+(permit(srf+1)-(cos(theta)).^2).^0.5);
    reflaction_coeff_orthogonal = (sin(theta)-(permit(srf+1)-(cos(theta)).^2).^0.5)./(sin(t

P_parallel = P_Tx * sin(theta);
P_orthogonal = P_Tx * cos(theta);

Pt_parallel = P_parallel * (1-(abs(reflaction_coeff_parallel))^2)^2; % the second power is for the second reflection
Pt_orthogonal = P_orthogonal * (1-(abs(reflaction_coeff_orthogonal))^2)^2;

P_Tx1 = (Pt_parallel^2 + Pt_orthogonal^2)^0.5;
end

B.9.2. 3D Cases

function P_Tx2 = NLOS(P_Tx, user_loc, user_2, user_h, user_2h, surface, permit)
X = abs(user_2(1)-user_loc(1));
Y = abs(user_2(2)-user_loc(2));
Z = abs(user_2h-user_h);

no_floors = floor(abs(user_h-user_2h)/6);
if no_floors ~= 0
    theta = pi/2 - acos(Z/(X^2+Y^2+Z^2)^0.5);
    srf = 1;
    reflection_coeff_parallel = (-
        permit(srf+1).*sin(theta)+(permit(srf+1)-
        (cos(theta)).^2).^0.5)./(permit(srf+1).*sin(theta)+(permit(srf+1)-
        (cos(theta)).^2).^0.5);
    reflection_coeff_orthogonal = (sin(theta)-(permit(srf+1)-
        (cos(theta)).^2).^0.5)./(sin(theta)+(permit(srf+1)-
        (cos(theta)).^2).^0.5);

    P_parallel = P_Tx * sin(theta);
    P_orthogonal = P_Tx * cos(theta);

    Pt_parallel = P_parallel * ((1-
        (abs(reflection_coeff_parallel))^2)^2)^no_floors ;
        % the second power is for the second reflection
    Pt_orthogonal = P_orthogonal * ((1-
        (abs(reflection_coeff_orthogonal))^2)^2)^no_floors ;
        % the no_floors power to account for the number of floors between
    Tx and Rx

    P_Tx2 = (Pt_parallel^2 + Pt_orthogonal^2)^0.5;
elseif no_floors == 0
    slope=(user_loc(2)-user_2(2))/(user_loc(1)-user_2(1));
    srf = 0;
    if abs(slope)<1
        for x=round(user_loc(1)):round(user_loc(1)):abs(user_2(1)-
            round(user_loc(1)))/abs(user_2(1)-round(user_loc(1))):
            y=slope*(x-user_2(1))+user_2(2);
            if surface(x, ceil(y))~=0
                point = [x, ceil(y)];
                srf = surface(x, ceil(y));
                break
            end
        end
    else
        if slope==0
            slope1=inf;
        else
            slope1=1/slope;
        end
        for y=round(user_loc(2)):round(user_loc(2)):abs(user_2(2)-
            round(user_loc(2)))/abs(user_2(2)-round(user_loc(2))):
            x=slope1*(y-user_2(2))+user_2(1);
            if surface(ceil(x), y)~=0
                point = [ceil(x), y];
                srf = surface(ceil(x), y);
                break
            end
        end
    end
else
    if srf==0
        P_Tx2 = P_Tx;
else
    if surface(point(1)+1, point(2)) ~= 0
        theta = atan(abs(user_loc(2) - point(2))/abs(user_loc(1) - point(1)));
    elseif surface(point(1)-1, point(2)) ~= 0
        theta = atan(abs(user_loc(2) - point(2))/abs(user_loc(1) - point(1)));
    elseif surface(point(1), point(2)+1) ~= 0
        theta = atan(abs(user_loc(1) - point(1))/abs(user_loc(2) - point(2)));
    elseif surface(point(1), point(2)-1) ~= 0
        theta = atan(abs(user_loc(1) - point(1))/abs(user_loc(2) - point(2)));
    end
end

reflaction_coeff_parallel = (-permit(srf+1).*sin(theta)+(permit(srf+1)-(cos(theta)).^2).^0.5)./(permit(srf+1).*sin(theta)+(permit(srf+1)-(cos(theta)).^2).^0.5);
reflaction_coeff_orthogonal = (sin(theta)-(permit(srf+1)-(cos(theta)).^2).^0.5)./(sin(theta)+(permit(srf+1)-(cos(theta)).^2).^0.5);

P_parallel = P_Tx * sin(theta);
P_orthogonal = P_Tx * cos(theta);

Pt_parallel = P_parallel * (1-(abs(reflaction_coeff_parallel))^2)^2;  % the second power is for the second reflection
Pt_orthogonal = P_orthogonal * (1-(abs(reflaction_coeff_orthogonal))^2)^2;

P_Tx2 = (Pt_parallel^2 + Pt_orthogonal^2)^0.5;
end
end

B.10. multi_path.m

These subroutines are used to simulate the signal being reflected.

B.10.1. 2D Cases and 3D Outdoor Cases

function [Pr Distance] = multi_path(user_loc, user_2, Power_i, surface, permit)
D = ((user_loc(1) - user_2(1))^2 + (user_loc(2) - user_2(2))^2)^0.5;
\( sf = \text{zeros}(4,2); \)

\( n = 0; \)
\( k = 0; \)
\( \text{while } k==0 \)
\( \quad n = n+1; \)
\( \quad k = \text{surface}(\text{round}(user\_loc(1)+n), \text{round}(user\_loc(2)))); \)
\( \text{end} \)
\( sf(1,:) = [user\_loc(1)+n \ user\_loc(2)]; \)
\( n = 0; \)
\( k = 0; \)
\( \text{while } k==0 \)
\( \quad n = n+1; \)
\( \quad k = \text{surface}(\text{round}(user\_loc(1)-n), \text{round}(user\_loc(2)))); \)
\( \text{end} \)
\( sf(2,:) = [user\_loc(1)-n \ user\_loc(2)]; \)
\( n = 0; \)
\( k = 0; \)
\( \text{while } k==0 \)
\( \quad n = n+1; \)
\( \quad k = \text{surface}(\text{round}(user\_loc(1)), \text{round}(user\_loc(2)+n)))); \)
\( \text{end} \)
\( sf(3,:) = [user\_loc(1) \ user\_loc(2)+n]; \)
\( n = 0; \)
\( k = 0; \)
\( \text{while } k==0 \)
\( \quad n = n+1; \)
\( \quad k = \text{surface}(\text{round}(user\_loc(1)), \text{round}(user\_loc(2)-n)))); \)
\( \text{end} \)
\( sf(4,:) = [user\_loc(1) \ user\_loc(2)-n]; \)

\% d1 and d2 are the distance between the users the four walls around
\% them
\( d1(1) = \text{abs}(sf(1,1)-user\_loc(1)); \)
\( d2(1) = \text{abs}(sf(1,1)-user\_2(1)); \)
\( d1(2) = \text{abs}(sf(2,1)-user\_loc(1)); \)
\( d2(2) = \text{abs}(sf(2,1)-user\_2(1)); \)
\( d1(3) = \text{abs}(sf(3,2)-user\_loc(2)); \)
\( d2(3) = \text{abs}(sf(3,2)-user\_2(2)); \)
\( d1(4) = \text{abs}(sf(4,2)-user\_loc(2)); \)
\( d2(4) = \text{abs}(sf(4,2)-user\_2(2)); \)

\% D1 and D2 are the distance between the users and the point of
\% reflection
\( D1(1) = (((D^2-(d2(1)-d1(1))^2)/(d1(1)+d2(1))^2+1)*(d1(1))^2).^0.5; \)
\( D2(1) = D1(1)*d2(1)/d1(1); \)
\( D1(2) = (((D^2-(d2(2)-d1(2))^2)/(d1(2)+d2(2))^2+1)*(d1(2))^2).^0.5; \)
\( D2(2) = D1(2)*d2(2)/d1(2); \)
\( D1(3) = (((D^2-(d2(3)-d1(3))^2)/(d1(3)+d2(3))^2+1)*(d1(3))^2).^0.5; \)
\( D2(3) = D1(3)*d2(3)/d1(3); \)
\( D1(4) = (((D^2-(d2(4)-d1(4))^2)/(d1(4)+d2(4))^2+1)*(d1(4))^2).^0.5; \)
\( D2(4) = D1(4)*d2(4)/d1(4); \)
\[ \text{DIS} = \text{D1} + \text{D2}; \]
\[ \theta = \arcsin(\frac{\text{d1}}{\text{D1}}); \quad \% \text{angle of reflection} \]
\[ [r \ c] = \text{find}(\text{DIS} = \min(\text{DIS})); \]
\[ \% \text{to check where the reflection point is compared to our user} \]
\[ X = -\frac{(\text{user\_loc(1)} - \text{user\_2(1)})}{\text{abs}(\text{user\_loc(1)} - \text{user\_2(1)})}; \]
\[ Y = -\frac{(\text{user\_loc(2)} - \text{user\_2(2)})}{\text{abs}(\text{user\_loc(2)} - \text{user\_2(2)})}; \]
\[ \% \text{sf(:,3) is the kind of surface where the reflection happens} \]
\[ \text{MP} = \text{length}([\text{surface(:,1)})]; \]
\[ \text{if} \ c=1 \]
\[ \quad \text{if} \ (\text{sf(1,2)} + Y*\text{D1(1)}*\cos(\theta(1))) < 1 \]
\[ \quad \quad \text{SF} = 1; \]
\[ \quad \text{elseif} \ (\text{sf}(1,2) + Y*\text{D1}(1)*\cos(\theta(1))) > \text{MP} \]
\[ \quad \quad \text{SF} = \text{MP}; \]
\[ \quad \text{else} \]
\[ \quad \quad \text{SF} = (\text{sf}(1,2) + Y*\text{D1}(1)*\cos(\theta(1)))); \]
\[ \quad \text{end} \]
\[ \text{sf}(1,3) = \text{surface(\text{round}(\text{sf}(1,1)), \text{round}(\text{SF}))}; \]
\[ \text{elseif} \ c=2 \]
\[ \quad \text{if} \ (\text{sf}(2,2) + Y*\text{D1}(2)*\cos(\theta(2))) < 1 \]
\[ \quad \quad \text{SF} = 1; \]
\[ \quad \text{elseif} \ (\text{sf}(2,2) + Y*\text{D1}(2)*\cos(\theta(2))) > \text{MP} \]
\[ \quad \quad \text{SF} = \text{MP}; \]
\[ \quad \text{else} \]
\[ \quad \quad \text{SF} = (\text{sf}(2,2) + Y*\text{D1}(2)*\cos(\theta(2)))); \]
\[ \quad \text{end} \]
\[ \text{sf}(2,3) = \text{surface(\text{round}(\text{sf}(2,1)), \text{round}(\text{SF}))}; \]
\[ \text{elseif} \ c=3 \]
\[ \quad \text{if} \ (\text{sf}(3,1)+ X*\text{D1}(3)*\cos(\theta(3))) < 1 \]
\[ \quad \quad \text{SF} = 1; \]
\[ \quad \text{elseif} \ (\text{sf}(3,1)+ X*\text{D1}(3)*\cos(\theta(3))) > \text{MP} \]
\[ \quad \quad \text{SF} = \text{MP}; \]
\[ \quad \text{else} \]
\[ \quad \quad \text{SF} = (\text{sf}(3,1)+ X*\text{D1}(3)*\cos(\theta(3)))); \]
\[ \quad \text{end} \]
\[ \text{sf}(3,3) = \text{surface(\text{round}(\text{SF}), \text{round}(\text{sf}(3,2)));} \]
\[ \text{elseif} \ c=4 \]
\[ \quad \text{if} \ (\text{sf}(4,2)+ X*\text{D1}(4)*\cos(\theta(4))) < 1 \]
\[ \quad \quad \text{SF} = 1; \]
\[ \quad \text{elseif} \ (\text{sf}(4,2)+ X*\text{D1}(4)*\cos(\theta(4))) > \text{MP} \]
\[ \quad \quad \text{SF} = \text{MP}; \]
\[ \quad \text{else} \]
\[ \quad \quad \text{SF} = (\text{sf}(4,2)+ X*\text{D1}(4)*\cos(\theta(4)))); \]
\[ \quad \text{end} \]
\[ \text{sf}(4,3) = \text{surface(\text{round}(\text{SF}), \text{round}(\text{sf}(4,2)));} \]
\[ \text{end} \]
\[ \text{reflaction\_coeff\_parallel} = (\quad \text{\texttt{permi}} \texttt{t(sf(:,3)+1).*sin(theta)+(permi} \texttt{t(sf(:,3)+1)} \texttt{-(cos(theta)).^2).^0.5)./(permi} \texttt{t(sf(:,3)+1).*sin(theta)+(permi} \texttt{t(sf(:,3)+1)} \texttt{-(cos(theta)).^2).^0.5); \]
\[ \text{reflaction\_coeff\_orthogonal} = (\text{\texttt{sin}}(\text{\texttt{theta}})-(\text{\texttt{permi}} \texttt{t(sf(:,3)+1)} \texttt{-(cos(theta)).^2).^0.5)./(\text{\texttt{sin}}(\text{\texttt{theta}})+(\text{\texttt{permi}} \texttt{t(sf(:,3)+1)} \texttt{-(cos(theta)).^2).^0.5)); \]
P_parallel = Power_i * sin(theta(c)) ;
P_orthogonal = Power_i * cos(theta(c)) ;

Pr_parallel = P_parallel * (abs(reflaction_coeff_parallel(c)))^2 ;
Pr_orthogonal = P_orthogonal * (abs(reflaction_coeff_orthogonal(c)))^2 ;

Pr = (Pr_parallel^2 + Pr_orthogonal^2)^0.5;
Distance = DIS(c);

B.10.2. 3D Indoor Cases

function [Pr Distance] = multi_path(user_loc, user_2, user_h, user_2h, 
Power_i, surface, permit)

D = ((user_loc(1)-user_2(1))^2+(user_loc(2)-user_2(2))^2+(user_h-
user_2h)^2)^0.5;

% Determining the floor level
if user_h < 6 * 1
   FLR = 1;
elseif user_h < 6 * 2
   FLR = 2;
elseif user_h < 6 * 3
   FLR = 3;
elseif user_h < 6 * 4
   FLR = 4;
elseif user_h < 6 * 5
   FLR = 5;
elseif user_h < 6 * 6
   FLR = 6;
end

sf = zeros(6,3);
n = 0;
k = 0;
while k==0
   n = n+1;
   if user_loc(1)+n>100
      disp(n)
   end
   k = surface(round(user_loc(1)+n), round(user_loc(2)), FLR);
end
sf(1,:) = [user_loc(1)+n user_loc(2) user_h];
n = 0;
k = 0;
while k==0
    n = n+1;
    if user_loc(1)-n<1
        disp(n)
    end
    k = surface(round(user_loc(1)-n), round(user_loc(2)), FLR);
end
sf(2,:) = [user_loc(1)-n user_loc(2) user_h];

n = 0;
k = 0;
while k==0
    n = n+1;
    if user_loc(2)+n>100
        disp(n)
    end
    k = surface(round(user_loc(1)), round(user_loc(2)+n), FLR);
end
sf(3,:) = [user_loc(1) user_loc(2)+n user_h];

n = 0;
k = 0;
while k==0
    n = n+1;
    if user_loc(2)-n<1
        disp(n)
    end
    k = surface(round(user_loc(1)), round(user_loc(2)-n), FLR);
end
sf(4,:) = [user_loc(1) user_loc(2)-n user_h];

H1 = user_h+4.5;
sf(5,:) = [user_loc(1) user_loc(2) H1];

H2 = user_h-1.5;
sf(6,:) = [user_loc(1) user_loc(2) H2];

% d1 and d2 are the distance between the users the four walls around them
d1(1) = abs(sf(1,1)-user_loc(1));
d2(1) = abs(sf(1,1)-user_2(1));
d1(2) = abs(sf(2,1)-user_loc(1));
d2(2) = abs(sf(2,1)-user_2(1));
d1(3) = abs(sf(3,2)-user_loc(2));
d2(3) = abs(sf(3,2)-user_2(2));
d1(4) = abs(sf(4,2)-user_loc(2));
d2(4) = abs(sf(4,2)-user_2(2));
d1(5) = abs(sf(5,3)-user_h);
d2(5) = abs(sf(5,3)-user_2h);
d1(6) = abs(sf(6,3)-user_h);
d2(6) = abs(sf(6,3)-user_2h);

% D1 and D2 are the distance between the users and the point of reflection
$D_1(1) = \frac{((D^2 - (d_2(1) - d_1(1))^2)/(d_1(1) + d_2(1))^2 + 1)*(d_1(1))^2)^{0.5};$

$D_2(1) = D_1(1) * d_2(1)/d_1(1);$

$D_1(2) = \frac{((D^2 - (d_2(2) - d_1(2))^2)/(d_1(2) + d_2(2))^2 + 1)*(d_1(2))^2)^{0.5};$

$D_2(2) = D_1(2) * d_2(2)/d_1(2);$

$D_1(3) = \frac{((D^2 - (d_2(3) - d_1(3))^2)/(d_1(3) + d_2(3))^2 + 1)*(d_1(3))^2)^{0.5};$

$D_2(3) = D_1(3) * d_2(3)/d_1(3);$

$D_1(4) = \frac{((D^2 - (d_2(4) - d_1(4))^2)/(d_1(4) + d_2(4))^2 + 1)*(d_1(4))^2)^{0.5};$

$D_2(4) = D_1(4) * d_2(4)/d_1(4);$

$D_1(5) = \frac{((D^2 - (d_2(5) - d_1(5))^2)/(d_1(5) + d_2(5))^2 + 1)*(d_1(5))^2)^{0.5};$

$D_2(5) = D_1(5) * d_2(5)/d_1(5);$

$D_1(6) = \frac{((D^2 - (d_2(6) - d_1(6))^2)/(d_1(6) + d_2(6))^2 + 1)*(d_1(6))^2)^{0.5};$

$D_2(6) = D_1(6) * d_2(6)/d_1(6);$

$DIS = D_1 + D_2;$

$\theta = \text{asin}(d_1/D_1); \quad \% \text{angle of reflection}$

[r c] = find(DIS==min(DIS));

$\% \text{to check where the reflection point is compared to our user}$

$X = -(\text{user}_\text{loc}(1) - \text{user}_\text{2}(1))/\text{abs}(\text{user}_\text{loc}(1) - \text{user}_\text{2}(1));$

$Y = -(\text{user}_\text{loc}(2) - \text{user}_\text{2}(2))/\text{abs}(\text{user}_\text{loc}(2) - \text{user}_\text{2}(2));$

$\% sf(:,4) is the kind of surface where the reflection happens$

$MP = \text{length}(\text{surface}(:,1));$

if $c=1$

$\text{if } (\text{sf}(1,2)+Y*D_1(1)*\cos(\theta_1)) < 1$

$SF = 1;$

$\text{elseif } (\text{sf}(1,2)+Y*D_1(1)*\cos(\theta_1)) > MP$

$SF = MP;$

$\text{else}$

$SF = (\text{sf}(1,2)+Y*D_1(1)*\cos(\theta_1));$

$\text{end}$

$\text{sf}(1,4) = \text{surface}(\text{round}(\text{sf}(1,1)), \text{round}(\text{SF}), \text{FLR});$

elseif $c=2$

$\text{if } (\text{sf}(2,2)+Y*D_1(2)*\cos(\theta_2)) < 1$

$SF = 1;$

$\text{elseif } (\text{sf}(2,2)+Y*D_1(2)*\cos(\theta_2)) > MP$

$SF = MP;$

$\text{else}$

$SF = (\text{sf}(2,2)+Y*D_1(2)*\cos(\theta_2));$

$\text{end}$

$\text{sf}(2,4) = \text{surface}(\text{round}(\text{sf}(2,1)), \text{round}(\text{SF}), \text{FLR});$

elseif $c=3$

$\text{if } (\text{sf}(3,1)+X*D_1(3)*\cos(\theta_3)) < 1$

$SF = 1;$

$\text{elseif } (\text{sf}(3,1)+X*D_1(3)*\cos(\theta_3)) > MP$

$SF = MP;$

$\text{else}$

$SF = (\text{sf}(3,1)+X*D_1(3)*\cos(\theta_3));$

$\text{end}$

$\text{sf}(3,4) = \text{surface}(\text{round}(\text{SF}), \text{round}(\text{sf}(3,2)), \text{FLR});$

elseif $c=4$

$\text{if } (\text{sf}(4,1)+X*D_1(4)*\cos(\theta_4)) < 1$

$SF = 1;$

$\text{elseif } (\text{sf}(4,1)+X*D_1(4)*\cos(\theta_4)) > MP$

$SF = MP;$

$\text{else}$


SF = (sf(4,1)+X*D1(4)*cos(theta(4))); 
end
sf(4,4) = surface(round(SF), round(sf(4,2)), FLR);

elseif c==5
sf(5,4) = 1;
elseif c==6
sf(6,4) = 1;
end

reflaction_coeff_parallel = (-permit(sf(:,4)+1).*sin(theta)+(permit(sf(:,4)+1)-
(cos(theta)).^2).^0.5)./(permit(sf(:,4)+1).*sin(theta)+(permit(sf(:,4)+1)-
(cos(theta)).^2).^0.5);
reflaction_coeff_orthogonal = (sin(theta)-(permit(sf(:,4)+1)-
(cos(theta)).^2).^0.5)./(sin(theta)+(permit(sf(:,4)+1)-
(cos(theta)).^2).^0.5);

P_parallel = Power_i * sin(theta(c));
P_orthogonal = Power_i * cos(theta(c));

Pr_parallel = P_parallel * (abs(reflaction_coeff_parallel(c)))^2;
Pr_orthogonal = P_orthogonal * (abs(reflaction_coeff_orthogonal(c)))^2;
Pr = (Pr_parallel^2 + Pr_orthogonal^2)^0.5;
Distance = DIS(c);

B.11. MMSE.m

A subroutine was created to test the MMSE location estimation method.

B.11.1. 2D Cases

function Loc = MMSE(LOC,D)

[r c] = size(LOC); %#ok<NASGU>

n = 0;
dlta = 100;
D_temp01 = D;
LOC_temp01 = LOC;

while dlta > 8
n=n+1;
delta = [];
for i=1:length(D_temp01)
    D_temp02 = D_temp01;
    D_temp02(i) = [];

LOC_temp02 = LOC_temp01;
LOC_temp02(i,:) = [];

x0 = sum(LOC_temp02(:,1))/(r-n);
y0 = sum(LOC_temp02(:,2))/(r-n);

delta(i) = (sum((D_temp02'-(LOC_temp02(:,1)-x0).^2+(LOC_temp02(:,2)-y0).^2+(LOC_temp02(:,3)-z0).^2).^0.5).^2/(r-n)).^0.5;
end
dlta = min(delta);
[R C] = find(dlta==delta);
D_temp01(C) = [];
LOC_temp01(C,:) = [];
end

Loc = sum(LOC_temp01)/(r-n-1);

**B.11.2. 3D Cases**

```matlab
function Loc = MMSE(LOC,D)
[r c] = size(LOC); %#ok<NASGU>
n = 0;
dlta = 100;

D_temp01 = D;
LOC_temp01 = LOC;

while dlta > 8
    n=n+1;
dlta = []; 
    for i=1:length(D_temp01)
        D_temp02 = D_temp01; 
        D_temp02(i) = [];
        LOC_temp02 = LOC_temp01;
        LOC_temp02(i,:) = [];
        x0 = sum(LOC_temp02(:,1))/(r-n);
        y0 = sum(LOC_temp02(:,2))/(r-n);
        z0 = sum(LOC_temp02(:,3))/(r-n);
        
        delta(i) = (sum((D_temp02'-(LOC_temp02(:,1)-x0).^2+(LOC_temp02(:,2)-y0).^2+(LOC_temp02(:,3)-z0).^2).^0.5).^2/(r-n)).^0.5;
    end
dlta = min(delta);
    [R C] = find(dlta==delta);
    D_temp01(C) = [];
    LOC_temp01(C,:) = [];
end
```

151
Loc = \text{sum}(\text{LOC\_temp01})/(r-n-1);