Improving Strength-of-Arrival Position Location Using a Neural Network on Radio Channel Impulse Responses

by

Jie Liu

BSc. Eng., (CS), Xi'an Institute of Technology, P. R. China, 1998

A Thesis Submitted in Partial Fulfillment of
the Requirements for the Degree of

Master of Science in Engineering

in the Graduate Academic Unit of
Department of Electrical and Computer Engineering

Supervisors: Brent R. Petersen, Ph.D., Electrical and Computer Engineering
Rajamani Doraiswami, Ph.D., Electrical and Computer Engineering

Examining Board: Dennis F. Lovely, Ph.D., Electrical and Computer Engineering, Chair
Richard J. Tervo, Ph.D., Electrical and Computer Engineering
Maryhelen Stevenson, Ph.D., Electrical and Computer Engineering
Virendrakumar C. Bhavsar, Ph.D., Faculty of Computer Science

This thesis is accepted.

Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

September, 2002

Abstract

This thesis focuses on evaluating an effective type of Position Location (PL) system for cellular phones.

Due to the inadequacy of existing Large-Scale-Fading (LSF) models, a new model is developed. This new LSF model introduces random changes called Splashes-Of-Change (SOC), in the root-mean-square delay spread of channel impulse responses over small regions of a cell. The new LSF model is called the SOC LSF Model (SOCLSFM) and includes propagation delay, path loss, exponentially distributed power delay profiles, and log-normal shadowing.

Strength-Of-Arrival (SOA) PL simulations were used to evaluate the SOCLSFM. SOA PL alone is often not sufficiently accurate because of the multipath. A multilayer Levenberg-Marquardt-trained feed-forward Neural Network (NN) was introduced and successfully improved accuracy compared to SOA PL. Impulse responses from the mobile to the base stations, as well as extracted features of impulse responses, are the inputs to the NN.

Acknowledgement

I would like to gratefully acknowledge Dr. B. R. Petersen and Dr. R. Doraiswami for their technical guidance, supervision, encouragement, funding support and patience throughout the research. This research was supported in part by an operating grant from the Natural Sciences and Engineering Research Council (NSERC).

I am also grateful to Geoffrey G. Messier for his valuable comments about radio channel measurements and location techniques.

Finally I would like to thank my landlady Anna Christie for her great help with my English. Also, I sincerely thank my family and my husband, Song Lin, for their understanding and encouragement.

Table of Contents

List of Tables

List of Figures

List of Abbreviations and Acronyms

2-D	Two-Dimensional
AOA	Angle-Of-Arrival
BSs	Base Stations
CDMA	Code Division Multiple Access
CTCF	Continuous-Time Continuous-Frequency
DSP	Digital Signal Processing
DTCF	Discrete-Time Continuous-Frequency
E-911	Enhanced 911
FCC	Federal Communications Commission
FDMA	Frequency Division Multiple Access
GPS	Global Positioning System
IS-95	Interim Standard 95
LM	Levenberg Marquardt
LOS	Line-Of-Sight
LSF	Large-Scale-Fading
MS	Mobile Station
NLOS	Non Line-Of-Sight
NN	Neural Network
PCS	Personal Communications Service
PL	Position Location
PLBLSFM	Path-Loss-Based Large Scale Fading Model
PSAPs	Public Safety Answering Points
RMS	Root-Mean-Square
RSS	Received Signal Strength
SOA	Strength-Of-Arrival
SOC	Splashes-Of-Change
SOCLSFM	Splashes-Of-Changes Large Scale Fading Model
SSF	Small-Scale-Fading
TDMA	Time Division Multiple Access

TDOA	Time-Difference-of-Arrival
TOA	Time-Of-Arrival

List of Mathematical Symbols

a_1	the mathematical symbol relates to P_1 and P_2. See Equation (3.7a)
a_2	the mathematical symbol relates to P_1 and P_3. See Equation (3.7b)
a_3	the mathematical symbol relates to P_2 and P_3. See Equation (3.7c)
$a_{11}, a_{12}, \ldots, a_{44}$	16 constraints that need to be determined to create the interpolation. See Equation (3.3)
$a_i^{(p)}$	the output of a function block in the multilayer feed-forward NN. See Figure 4.6
b	the mathematical symbol relates to RMS delay spread. See Equation (2.6)
$B_c(x, y)$	the bell-curve shape of the splash. See Equation (2.11)
c	the velocity of light or radio signal. See Equation (2.1)
d	distances between the three BSs. See Figure 3.5
D	the width of square grid. See Figure 2.1
d_0	the close-in reference distance. See Equation (2.8)
d_1	the distance from MS to BS1
d_2	the distance from MS to BS2
d_3	the distance from MS to BS3
d'_1	the first mean delay associated with the h_1. See Figure 4.3
d'_2	the second mean delay associated with the h_2. See Figure 4.3
d'_3	the third mean delay associated with the h_3. See Figure 4.3
dB	deciBels
d_s	the MS-BS separation distance. See Equation (2.8)
d_{in}	the fan-in (or in-degree) of a neuron
$d_i^{(p)}$	the desired output in the multilayer feed-forward NN. See Figure 4.6
e	exponential
e_1	the first error in the output of the in the multilayer feed-forward NN. See Figure 4.6

e_2 the second error in the output of the in the multilayer feed-forward NN. See Figure 4.6

f frequency. See Equation (A.1)

f_c the bandwidth of the radio channel 799.

f_m the maximum Doppler shift. See Equation (A.2)

g gradient of a NN. See Equation (F.7)

h each hidden neuron. See Equation (F.5)

H the height of one splash. See Equation (2.9)

h_1 the MS-BS1 impulse response. See Figure C.1

h_2 the MS-BS2 impulse response. See Figure C.2

h_3 the MS-BS3 impulse response. See Figure C.3

H_m Hessian matrix. See Equation (F.6)

H_s the mean height of one splash. See Equation (2.9)

h_1' the impulse response at coordinate (x, y).
See Figure (3.4) and Equation (3.2)

h_2' the impulse response at coordinate $(x+D, y)$.
See Figure (3.4) and Equation (3.2)

h_3' the impulse response at coordinate $(x, y+D)$.
See Figure (3.4) and Equation (3.2)

h_4' the impulse response at coordinate $(x+D, y+D)$.
See Figure (3.4) and Equation (3.2)

h_c the impulse response at MS. See Figure 3.4 and Equation (3.2)

hr hour

$h(\bullet)$ the $\bullet$th coefficient of the impulse response. See Figure 2.1

$h_n(k)$ a modified impulse response with corresponding exponent coefficient. See Equation (2.14)

$h_p(k)$ the old impulse response. See Equation (2.14)

$H(f)$ Doppler filter specified by Gans' model. See Figure A.5

i layer of the the multilayer feed-forward NN. See Figure 4.6

$I_m[\bullet]$	imaginary part of $\bullet$
j	node of the the multilayer feed-forward NN. See Figure 4.6
J	Jacobian matrix. See Equation (F.6) and (F.7)
k	relative time delay of the *k*th multipath component as compared to the 0th arriving component. See Equation (2.2)
km	kilometers (10^3 m)
l_1	the distance from MS to grid point (x_1, y_1). See Figure 3.4
l_2	the distance from MS to grid point $(x_1 + D, y_1)$. See Figure 3.4
l_3	the distance from MS to grid point $(x_1, y_1 + D)$. See Figure 3.4
l_4	the distance from MS to grid point $(x_1 + D, y_1 + D)$. See Figure 3.4
$\ln(\bullet)$	base *e* logarithm of $\bullet$
$\log(\bullet)$	base 10 logarithm of $\bullet$
m	meters
ms	milliseconds (10^{-3} s)
MB	megabytes (10^6 bytes)
MHz	megahertz (10^6 Hz)
MS-BS1	the impulse response calculate from the first relative coordinate
MS-BS2	the impulse response calculate from the second relative coordinate
MS-BS3	the impulse response calculate from the third relative coordinate
n	path loss exponent. See Equation (2.8)
N	width, in sample of each impulse response. See Equation (2.1)
ns	nanoseconds
N_a	the number of MS positions where the distance between the true and estimated position is less than or equal to 125 m
N_l	the number of MS locations where the PL scheme could located
N_o	the number of outages, meaning the number of MS positions where the PL scheme failed to location the MS
N_t	the total number of MS positions that are evaluated
$ND_s(x, y)$	the modified RMS delay spread. See Equation (2.13)

o	the activation of each neuron. See Equation (F.1)
p	the patterns of the multilayer feed-forward NN. See Figure 4.6
p_1	the received power at BS1
p_2	the received power at BS2
p_3	the received power at BS3
P_k	exponential power delay profile. See Equation (2.2)
P_r	mean received power by mobile. See Equation (A.1)
P_{outage}	the percentage of the time location estimate cannot be determined. See Equation (3.17)
$P\%$	the percentage of location estimates with errors less than 125 m. See Equation (3.18)
$PathL_{\mathrm{dB}}(\bullet)$	the path loss power in dB over distance of •. See Equation (2.8)
$\overline{PathL}_{dB}(\bullet)$	the path loss power in dB over reference distance of •. See Equation (2.8)
$P_{x,y}(x, y)$	2-D circular Gaussian probability density function. See Equation (2.10)
$PD_s(x, y)$	the pervious RMS delay spread. See Equation (2.12)
R	the radius of the circle ranges around BS. See Figure 2.1
$R_e[\bullet]$	real part of •
r_1	the radius of the first circle in SOA-based estimation. See Equation (3.11b)
r_2	the radius of the second circle in SOA-based estimation. See Equation (3.12b)
r_3	the radius of the third circle in SOA-based estimation See Equation (3.13b)
s	second
$R_c(x, y)$	the randomness of the splash
$S_i^{(p)}$	the output of a summer in the multilayer feed-forward NN. See Figure 4.6

$S_{bb}(f)$ a baseband power spectral density. See Equation (A.1)

t a target value in the target vector. See Equation (F.3)

T_M the number of testing patterns

T_N the number of training patterns

v velocity of mobile

W the width of one splash

w_1 the weight relates to h_1' in computing h_c. See Equations (3.2,3.5)

w_2 the weight relates to h_2' in computing h_c. See Equations (3.2,3.5)

w_3 the weight relates to h_3' in computing h_c. See Equations (3.2,3.5)

w_4 the weight relates to h_4' in computing h_c. See Equations (3.2,3.5)

w_{c} the bandwidth of CDMA. See Equation (2.1)

w_k the previous weights generated by LM method. See Equation (F.10)

W_s the mean width of one splash

w_{k+1} the newly updated weights generated by LM method.
See Equation (F.10)

$w_{i,j}^{(p)}$ the weighs of the multilayer feed-forward NN. See Figure 4.6

$\vec{w}$ vector denotes the weights linking the neuron unit to the previous neuron layer. See Equation (F.1)

x variable takes the *x* coordinate values at grid points.
See Equation (2.11)

X the number of input signals in a NN. See Figure 4.1

x_1 the fix *x* coordinated of a grid point. See Figure (3.4)

x_1' the *x* coordinate of the first intersected point of the circles.
See Equations (3.14a)

x_2' the *x* coordinate of the second intersected point of the circles.
See Equations (3.14a)

x_3' the *x* coordinate of the third intersected point of the circles.
See Equations (3.14a)

x_{r}	the *x* coordinate of a uniformly distributed random point
X_s	log-normal shadowing
x_{c1}	the *x* coordinate of the center of the first circle in SOA-based estimation. See Equation (3.11a)
x_{c2}	the *x* coordinate of the center of the second circle in SOA-based estimation.See Equation (3.12a)
x_{c3}	the *x* coordinate of the center of the third circle in SOA-based estimation.See Equation (3.13a)
$x_i^{(p)}$	input pattern of the multilayer feed-forward NN. See Figure 4.6
$\bar{x}$	*x* coordinate of the center of the distribution. See Equation (2.10)
$\hat{x}$	the *x* coordinate of PL estimate of MS. See Equation (3.14a)
$\bar{x}$	vector denotes the activation of the pervious layer. See Equation (F.1)
$\hat{x}_{SOA}$	the *x* coordinate of SOA PL estimate of MS. See Figure 4.2
$\hat{x}_{SOA+NN}$	the *x* coordinate of the PL estimates of the integrated scheme. See Figure 4.3
y	variable takes the *y* coordinate values at grid points. See Equation (2.11)
Y	the number of hidden nodes in a NN. See Figure 4.1
y_1	the fix *y* coordinated of a grid point. See Figure (3.4)
y_1'	the *y* coordinate of the first intersected point of the circles. See Equations (3.14b)
y_2'	the *y* coordinate of the second intersected point of the circles. See Equations (3.14b)
y_3'	the *y* coordinate of the third intersected point of the circles. See Equations (3.14b)
y_{r}	the *y* coordinate of a uniformly distributed random point
y_{c1}	*y* coordinate of the center of the first circle in SOA-based estimation. See Equation (3.11a)

y_{c2}	the *y* coordinate of the center of the second circle in SOA-based estimation. See Equation (3.12a)
y_{c3}	the *y* coordinate of the center of the third circle in SOA-based estimation. See Equation (3.13a)
$\bar{y}$	the *y* coordinate of the center of the distribution. See Equation (2.10)
$\hat{y}$	the *y* coordinate of PL estimate of MS. See Equation (3.14b)
$\hat{y}_{SOA}$	the *y* coordinate of SOA PL estimate of MS. See Figure 4.2
$\hat{y}_{SOA+NN}$	the *y* coordinate of the PL estimates of the integrated scheme. See Figure 4.3
z	the parameter of the z transform
a	the exponential coefficient. See Equation (2.2)
a_1	the positive exponential coefficient
a_1'	the exponential coefficient of first power delay profile. See Figures (4.4) and (4.5)
a_2'	the exponential coefficient of second power delay profile. See Figures (4.4) and (4.5)
a_3'	the exponential coefficient of third power delay profile. See Figures (4.4) and (4.5)
$a_{1,2}$	the two possible solutions of the exponential coefficient. See Equation (2.6)
a_n	the new exponential coefficient
a_p	$a_p = a_1$, old positive exponential coefficient
d_k	error term for each output neuron. See Equation (F.4)
e	the mean error of PL estimation. See Equation (3.16)
g	the scalar size. See Equation (F.10)
$h_Q(t)$	the quadrature phase component of a complex baseband Gaussian process. See Figure A.5

$h_t (t)$	in-phase component of a complex baseband Gaussian process. See Figure A.5
l	wavelength of light
m	the path loss coefficient
p	Ludolphian number, 3.1415926…
s	standard deviation. See Equation (2.8)
s_H	the standard deviation of the height of one splash. See Equation (2.9)
s_t	RMS delay spread. See Equation (2.5)
$s(y)$	activation function. See Equation (F.2)
$\bar{t}$	the first moment of power delay profile. See Equation (2.3)
$\overline{t^2}$	the second moment of power delay profile. See Equation (2.4)
∞	infinite
%	percent
$\bullet + \blacklozenge$	• plus ♦
$\bullet - \blacklozenge$	• minus ♦
$\frac{\bullet}{\blacklozenge}$	• divided by ♦
$\bullet \in \blacklozenge$	• belongs to ♦
$\bullet^{\blacklozenge}$	• raised to the power ♦
$\bullet^{\mathrm{T}}$	matrix transpose of •
$\bullet < \blacklozenge$	• less than ♦
$\sqrt{\bullet}$	square root of •
$\|\bullet\|$	absolute value of •
$[\bullet \blacklozenge]$	interval from • to ♦
$E[\bullet]$	mathematical expectation of •
$\overset{\bullet}{\longleftrightarrow}$	by • transform
$\sum_{a}^{b} \bullet$	sum of • from a to b

Chapter 1 Introduction

1.1 Background

During the past few years competition among telecommunications industries has been very intense. Mobile network operators are looking for new ways to improve their products and services and yet stay profitable. The operators compete with each other to provide customers with highly personalized service and in this context, cellular phone location systems have become a focus of intense research and development.

A cellular phone location system has potential applications, which benefit both industry and customers. The potential applications of location-based services [Drane98] can be broadly classified into four major categories: location-based information, location-sensitive billing, emergency services, and location tracking. Figure 1.1 shows some of the Position Location (PL) applications under the four categories.

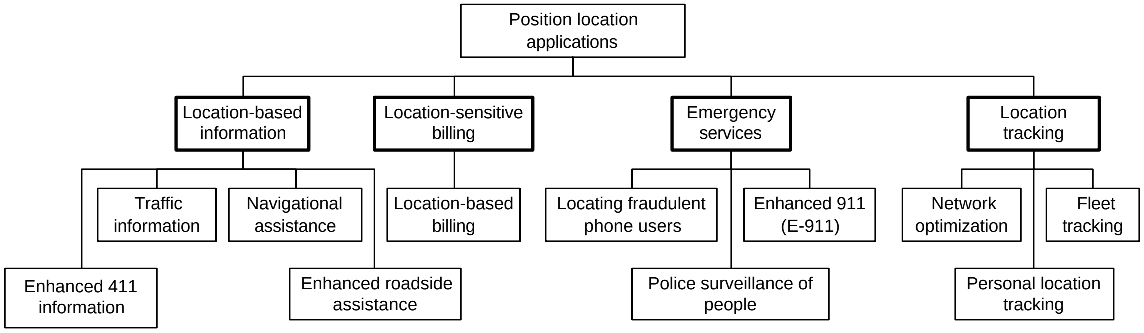

Position location applications
Location-based information
Location-sensitive billing
Emergency services
Location tracking
Traffic information
Navigational assistance
Location-based billing
Locating fraudulent phone users
Enhanced 911 (E-911)
Network optimization
Fleet tracking
Enhanced 411 information
Enhanced roadside assistance
Police surveillance of people
Personal location tracking

Figure 1.1 PL-based applications

The Enhanced 911(E-911) service, especially the location estimation, forms the main focus of this research. A recent study of wireless location by the State of New Jersey indicates that wireless E-911 calls accounted for 43% of all E-911 calls received during the trial [State of NJ97]. Location information for wireless E-911 calls permits rapid response which can help in situations where callers are disoriented, disabled, unable to speak, or do not know their location [Caffery98].

Besides providing emergency services, the position location allows location-sensitive billing. The network operators provide differential tariffs depending on the location of the mobile. Location tracking is used to update personal location information, or the long-term monitoring of mobile phone positions. It also provides excellent input to the planning of the cellular network, for improving the network performance [Drane98].

Section 1.1 will briefly introduce the requirements of location applications, mobile-aided and network-aided location techniques, three networked-aided basic existing PL techniques, the radio systems, and some problems existing in PL.

1.1.1 Position Location Requirements

It is always desirable to achieve the highest possible accuracy in location applications. However, the requirements in different applications may differ due to various reasons such as the cost and the technology. The emergency applications, such as E-911 and other applications based on location-sensitive billing would all require high accuracy. However, other location applications requiring lesser accuracy such as fleet management, can utilize lower-accuracy location techniques [Caffery98].

1.1.1.1 General FCC requirements for Enhanced 911

The accuracy requirement of E-911 location service is an important area of research, and the impetus for the development of cellular location techniques comes from the accuracy requirement of E-911 services.

In 1996, the US Federal Communications Commission (FCC) required that every wireless service provider including cellular, provide location information to 911 public safety services, Public Safety Answering Points (PSAPs). A directive (FCC Order 94-

102) [FCC97] mandated two phases of implementation of E-911 technology by wireless service providers. Phase II, beginning in October 2001, requires carriers to identify the location within 125 m at least 67% of the time [Koshima00].

The desired features of a cellular E-911 system are summarized as the following [Koshima00]:

- the FCC requirement is met (<125 m accuracy for 67% of all measurements),
- the coverage should be comparable to cellular systems,
- seamless integration with existing Base Stations (BSs),
- low-cost location receiver equipment,
- expandable, scalable and reliable, and
- an interfacing with existing E-911 terrestrial networks.

1.1.1.2 Specific FCC requirements for mobile-aided and network-aided techniques

A radio location system can be a mobile-aided PL system, network-aided PL system, or hybrid PL system [Zagami98].

In a mobile-aided PL system, the Mobile Station (MS) uses signals transmitted by the BSs to calculate its own position using a device such as a Global Positioning System (GPS) receiver embedded in the mobile handset. The drawbacks of this system include the high cost for developing a suitable low-power and economical integrated technology for use in the handsets, and the cost for deploying new handsets.

In a network-aided PL system the BSs measure the signals transmitted by the MS and relay them to a location centre for processing. This has the advantage of not requiring any modifications or specialized equipment in the MS handset, thus accommodating the large pool of handsets already in use by the existing cellular networks. Infrastructure required for implementation will not be prohibitively expensive.

The hybrid PL system combines different aspects of mobile-aided and network-aided positioning architectures.

The FCC later amended the location accuracy requirements for phase II to the following [FCC97]:

- Mobile-Aided Solution

  50 m for 67% of the calls

  150 m for 95% of the calls

- Network-Aided Solution

  100 m for 67% of the calls

  300 m for 95% of the calls

Note that the accuracy requirement of a mobile-aided location solution is different in the FCC regulations from a network-aided. Because of the drawbacks of non-network technologies, cellular carriers generally favor the use of a network-aided approach. So the non-network technologies will not be discussed further.

1.1.2 Methods of Network-Aided Position Location Techniques

There are many economical PL systems, which are implemented based on the three basic existing network-aided PL methods: signal Strength-Of-Arrival (SOA), Angle-Of-Arrival (AOA), Time-Of-Arrival (TOA) measurements, or their combinations [McGuire94]. The SOA technique will be discussed later in Section 1.2.2.

1.1.2.1 Angle-of-arrival

The AOA location method uses simple triangulation to locate the MS. The receiver BSs measure the direction of received signals (i.e. AOA) from the MS using directional antennas or antenna arrays. AOA measurements at two BSs will provide a position fix but the accuracy of the position estimation depends on distance and geometry between the MS and two BSs, as well as multipath propagation. As a result, more than two BSs are normally needed to improve the location accuracy [Caffery98].

This method suffers in the indoor Non-Line-Of-Sight (NLOS) environment. Surrounding objects or walls usually block the Line-Of-Sight (LOS) signal path, which is the clear path from the receiver to the transmitter. As a result this method is useful for macro cellular cells, and may be impractical for micro cellular cells.

1.1.2.2 Time-of-arrival

The TOA method is based on estimating the propagation time of the signals from an MS to multiple BSs. Once the TOAs are measured, the distances between the MS and BSs can be simply determined by the radio speed (speed of light). The MS's position is given by the intersection of the circles with BSs at the center once the radial distances are calculated [Caffery98].

An extension of the TOA technique is the Time Difference of Arrival (TDOA), which measures the TDOAs of a signal received at multiple pairs of BSs. The MS's position is given by the intersection of the hyperbolae.

The TOA method requires high-resolution timing measurements. Compared to TOA, the advantage of TDOA is that it does not require knowledge of the transmit time. Instead, time synchronization among multiple BSs is required. However, for both TOA and TDOA, LOS propagation conditions are necessary to achieve high accuracy.

1.1.3 Radio Systems

Pagers, cordless and cellular phones are the common mobile radio systems. However, the cellular system is the only one that provides high quality service [Rappaport96].

With thousands of cellular phone calls going on at any given time within a city, it certainly would not be possible for everyone to talk on the same channel at once.

Therefore, several different techniques were developed by cellular phone manufacturers to assign the available bandwidth into many channels, each of which is capable of supporting one conversation. The technique that allows many mobile users to share the bandwidth is called multiple access. The cellular systems are based on three existing Frequency Division Multiple Access (FDMA), Time Division Multiple Access (TDMA), and Code Division Multiple Access (CDMA) techniques [Viterbi95].

1.1.3.1 Frequency division multiple access

FDMA assigns individual channels to individual users. Each user is allocated a unique frequency band or channel and these channels are assigned on demand for users who request service. During the period of the call, no other user can share the same frequency band. FDMA is used in all analog cellular systems.

FDMA systems are often regarded as the least efficient cellular system since only one user can access a particular channel at a time. Analog signals are particularly susceptible to noise. As a consequence, analog cell phones must use higher power (between 0.3 and 3 watts) to achieve an acceptable call quality. Due to these shortcomings, FDMA is being replaced by newer digital techniques [Lee97].

1.1.3.2 Time division multiple access

TDMA systems divide the radio channel into time slots. In each slot only one user is allowed to either transmit or receive. Each user occupies a cyclically repeating time slot. Therefore, a channel may be regarded as particular time slot that reoccurs every frame.

While TDMA is a good access technique, it is still somewhat inefficient. TDMA has limited flexibility for varying digital data rates (high quality voice, low quality voice, pager traffic) or accommodations for silence in a telephone conversation. TDMA also requires strict signaling and time slot synchronization [Lee97].

1.1.3.3 Code division multiple access

CDMA provides increased capacity [Gilhousen91] and has been selected as the Interim Standard 95 (IS-95) by the U.S. Telecommunication Industry Association [TIA/EIA99]. IS-95 systems encode each call as a coded sequence across the entire frequency spectrum. Each conversation is modulated in the digital domain, with a unique code (called a pseudo-noise code) that makes it distinguishable from the other calls in the frequency spectrum. Using a correlation calculation and the code the call was encoded with, the digital audio signal can be extracted from the other signals being broadcast by other phones on the network [Garg97].

CDMA offers far greater capacity and variable data rates depending on the audio activity. CDMA has the complexity of deciphering and extracting the received signals, especially if there are multiple signal paths (reflections) between the MS and BS (called multipath interference). CDMA has many other advantages, but it still has some drawbacks. Its performance suffers from power variations with distance between the BS and MS, termed the near-far problem. However, this problem can be overcome by CDMA power control [Rappaport96]. The near-far problem and power control concepts will be explained in Section 1.1.4.

In this thesis CDMA is selected as an example of a cellular radio system in view of its popularity, efficiency and adoption as an interim industry standard. The most important aspect of CDMA for this thesis is the modeling of channel impulse responses, which is developed in Chapter 2.

1.1.4 Problems in Position Location Systems

Sources of errors in wireless location systems include multipath propagation, multiple access interference, and variation of power with distance.

1.1.4.1 Code division multiple access power control

In CDMA systems, users create interference among each other in which strong users with higher received powers (short MS-BS distances) can overwhelm the communication quality of the weak users with lower received powers (long MS-BS distance) [Mizusawa96]. The non-zero correlation between their codes aggravates the interference.

In order to combat the near-far problem, power control must be implemented in MSs. This power control assures that all transmissions from the MSs within the cell are received with the same signal power at the BS. An MS close to a BS will have a reduced transmitter power. Consequently, other BSs in neighboring cells will receive a low-power signal from the MS.

The power control scheme, which reduces the power transmitted by the MS to the neighboring BSs, however, makes it difficult for the neighboring BSs to find the distances between the MS and BSs when the SOA estimation scheme is employed. The PL estimation will be inaccurate. Thus the power control presents a major problem for a reliable and accurate geolocation of mobile phone users [Knopp95].

1.1.4.2 Multipath

Multipath is the primary source of inaccuracy in the SOA or AOA estimation schemes [Caffery98]. Multipath effects may be caused by inhomogeneity of the atmosphere, reflections from natural barriers such as hills and cliffs, and man-made obstacles such as buildings, and tunnels. The total received field strength is the sum of contributions arising from the individual multipath components (reflected, diffracted, and mixed). As a result, the received power is subject to rapid fluctuations, and several versions of same signal may arrive at different time instants. This causes inaccuracy in the PL estimation scheme.

1.1.4.3 Multiple access interference

In a CDMA system, many MSs in the system transmit signals in the same frequency band. This produces a high level of interference among the transmitted signals. A signal from an MS, which has to be located, can be obscured by the signals from other MSs, especially if the interfering signal strength is higher. In CDMA, multipath interference can only be rejected if the multipath arrives with a delay of at least one chip interval of the CDMA code. Multipath arriving within the chip interval can introduce errors in PL estimation [Mizusawa96].

1.2 Literature Review

The work described in this thesis is multidisciplinary in nature borrowing techniques from various disciplines including estimation theory, communication theory, statistics, and digital signal processing. The radio channel fading models are reviewed first, followed by PL estimation using the SOA technique. Finally related applications of Neural Networks (NNs) are discussed.

1.2.1 Radio Channel Fading Models

Radio channel models traditionally focus on predicting the average Received Signal Strength (RSS) with respect to distance from the MS, as well as the variability of the signal strength in close spatial proximity to a particular location. There are two typical radio channel models, Large-Scale-Fading (LSF) and Small-Scale-Fading (SSF) models.

1.2.1.1 Large-scale-fading

The LSF model predicts the mean signal strength for very large BS-MS separation distances (several hundred meters or thousands of meters) [Rappaport96]. This is important for predicting received power levels in cells. Adaptive antennas or power control may compensate for large-scale path losses. Also the LSF model represents an average signal power attenuation or path loss due to motion over large areas. This phenomenon is affected by prominent terrain contours such as hills, forests, billboards, and clumps of buildings. The receiver is often represented as being shadowed by such prominences. The statistics of the LSF model provides a way of computing an estimate of the path loss as a function of the distance [Liberti97].

In general, current LSF models can be classified as outdoor or indoor fading models. Some commonly used outdoor fading models are the Longley-Rice model, the Durkin model, the Okumura model, the Hata model, the Walfisch and Bertoni model, and the wideband Personal Communications Service (PCS) micro-cell model [Rappaport96]. The outdoor fading model is dominated by the large-scale path losses. However this is not the case for the indoor fading model. The indoor fading model can only cover a very small area, and the variability of the environment is much greater for a much smaller range of MS-BS distances [Rappaport96]. The indoor fading model may include the partition losses, log-distance path losses, the Ericsson multiple breakpoint model, and the attenuation factor model [Rappaport96]. For simplicity of analysis, the LSF models developed in the thesis will be applied only in the outdoor fading environment. However, a variety of deficiencies are explored by studying these existing outdoor propagation models. They are explained in the following sections.

The first deficiency is a lack of impulse responses in models. In a radio communication system, the channel determines how the electromagnetic propagation of a signal from a transmitter is received. It is possible to express the channel in terms of its impulse response, which is the signal that would be received if an impulse were to be transmitted. The impulse response characterizes the wideband radio system and contains all information, especially the multipath of the transmitted signal, necessary to analyze or simulate any type of radio transmission through the channel [Rappaport96].

The second deficiency is a lack of large-scale correlation among impulse responses. Large-scale correlation of impulse responses is caused by objects in the environment. For example, consider all the coefficients of all the BS-MS impulse responses both near to a building and far from a building. Those coefficients of the impulse responses where the MS is near the building will have a higher correlation compared to impulse responses where the MS is far from the building.

The third deficiency is geographical dependence. A geographical database, which contains a map of buildings and terrain features, is required to develop accurate propagation models in specific regions. The computational expense is enormous and sometimes cannot be avoided.

For example, the Longley-Rice model is based on a geographic database and is complex. It does not consider multipath and hence does not employ impulse responses in the model implementation [Rice67]. The Durkin's Model is simple in the LOS environment, but in the NLOS environment, it is too complicated [Dadson75]. The Okumura and Hata models are simple. They are ideal in an urban area, but they do not employ impulse responses [Okumura68, Hata90]. Ray tracing techniques employ large-scale correlated impulse responses, but require a geographic database [Schaubach92].

The existing outdoor LSF models mostly suffer from one or both of the problems of complexity and absence of impulse responses. A newly regenerated fading model, termed the Splashes-Of-Changes LSF Model (SOCLSFM), is based on the simple features of current outdoor LSF models, has impulse responses with large-scale correlations, but does not have the complexity of models with geographic databases. In Chapter 2, the SOCLSFM will be described in detail.

1.2.1.2 Small-scale-fading

The SSF model characterizes the rapid fluctuations of the RSS over very short travel distances or short times [Rappaport96]. Compared with the LSF model, the SSF model exhibits much more rapid fluctuations of received signals over a short travel distance or time. This fluctuation is caused by constructive and destructive interference between two or more versions of the same signal [Liberti97].

The theory of the SSF model and simulation results are described in Appendix A. Appendix A also contains the results of an investigation about the suitability of the SSF model for this thesis. The results of simulations show that there is no large-scale correlated multipath in the SSF. Therefore, an LSF model is more suited for an NN-based location system which uses multipath properties to improve performance.

1.2.2 Position Location Estimation using Strength-Of-Arrival Techniques

PL using SOA is a well-known location estimation method that uses a known mathematical model describing the path loss attenuation with distance [Figel69, Hata80]. Since a measurement of signal strength provides a distance estimate between the MS and BS, the MS lies on a circle centered at the BS. By using multiple BSs, the location of the

MS can be determined from the intersection of the circles [Caffery98]. However, practical propagation conditions especially in urban area are far from ideal. Although the signal-level contours are no longer circles and are not centered at the BSs, they can be used for location estimation by finding the location that produces the best fit between the predicted and the measured value.

Compared to other PL techniques, the SOA method has less accuracy than the time-based and AOA methods. However, it is easier to implement and has the advantage of not requiring any significant modifications or specialized equipment in the MS handset or BSs. On the other hand, the AOA method requires antenna arrays to determine the AOA of the signal, and the time-based TOA and TDOA techniques need timing measurements. The SOA technique is relatively independent of the modulation and multiple access method used in the network. This makes it attractive for indoor and micro-cell location applications for low cost MSs [McGuire94]. In CDMA cellular systems, the MSs are power controlled to combat the near-far effect. Therefore, for SOA systems, it is necessary that the transmitted power of the MSs be known and be controlled with reasonable accuracy. IS-95 provides BSs with the ability to poll the MSs received powers. Due to these advantages, the SOA method is chosen for PL estimation. In Chapter 3 its performance and results of evaluation using the SOCLSFM is presented.

For practical SOA location systems, LOS seldom exists between MSs in buildings or urban areas. In other words, there exists unknown signal path loss and multipath between the transmitter and receiver. Multipath fading and shadowing have major adverse effects on signal strength measurements. Additionally, the SOA method can only provide good performance in the region where the distances between the BSs is small [McGuire94]. Therefore, a lack of precision with the SOA method is a problem. Poor accuracy associated with the SOA method is overcome by including an NN with the SOA method.

1.2.3 Applications of Neural Networks

"A neural network is an interconnected assembly of simple processing elements, units or nodes, whose functionality is loosely based on the animal neuron. The processing ability of the network is stored in the inter-unit connection strengths, or weights, obtained by a process of adaptation to, or learning from, a set of training patterns." [Gurney96]. NNs are often good at solving problems that do not have an algorithmic solution or for which an algorithmic solution is too complex to be found.

Individual neurons of an NN are connected from the input pattern to the outputs. Inputs and interconnection weights are processed by a summation function (typically a weighted summation) to yield a sum that is passed to a nonlinear function (typically a sigmoid nonlinearity). The output of the nonlinearity is the output of the neuron or nodes. An NN uses a training procedure to adjust the input weights on each neuron such that the output of the network is consistent with the desired output. The process consists of presenting the given data to each input node and the correct or desired response to each of the network's output nodes. Once the network is trained, input data, which the NN has not previously encountered, termed the test or blind data, is presented to the NN. The output node gives the desired estimate [Gurney96].

NNs are increasingly employed to solve real world problems of considerable complexity. Although one may apply an NN for interpretation, prediction, diagnosis, planning, monitoring, debugging, repair, instruction, and control, the most successful applications of NNs are in categorization and pattern recognition, which includes speech and image processing.

Basically, most applications of NNs fall into the following five categories [Zahedi93]: prediction, which uses input values to predict some output, dassification, which uses input values to determine the class to which the input belongs, data association, like classification, recognizes data that contains errors, data conceptualization, analyzes inputs so that grouping relationships can be inferred, and data filtering, which smoothes an input signal.

The NN may be a single layer or multilayer perceptron. Multilayer perceptrons are feedforward NNs, which are commonly used in speech and image recognition. Because of the presence of hidden layers, the multilayer perceptron has the ability to realize any arbitrary nonlinear input-output functional relationship governing the data. These types of NNs are trained off-line, using a large set of known input-output data. Once trained, the network weights are frozen and test data can be run through the NN [Scalero92]. In this thesis, PL estimation is based on an integration of a feed forward NN and the SOA method.

Similar to speech or image processing using the NN, where the input pattern of speech or image data is fuzzy and unclear, and the input patterns to the NN obtained from the SOA method are also less accurate. By training the NN with input data, which may be inaccurate, and the output data, namely the MS's location, which is accurate, the problem of location inaccuracy stemming from the input SOA estimate is overcome. In PL, obtaining a relationship governing the input, namely SOA, and the output data, namely the location of the MS, is generally non-linear. An NN, in view of its ability to model non-linear relationships, is preferred to the traditional statistical technique, which cannot handle as easily non-linear relationships.

The integrated SOA-NN technique will be covered in Chapter 4.

1.3 Thesis Contributions

The two main contributions of this thesis are, first, the development and evaluation of the SOCLSFM and, second, PL technique using a integration of SOA and an NN.

1.3.1 Splashes-Of-Changes Large-Scale-Fading Model

The existing LSF models are either complicated, missing impulse responses, unable to handle multipath, or dependent upon a geographic database. This thesis introduces the SOCLSFM, which is based on adding, step-by-step, several radio propagation effects and features into channel impulse responses. The Splashes-Of-Changes (SOC) effect, which is caused by delayed or reflected signals from multipath fading, is introduced in the last step. The SOCLSFM overcomes the deficiencies of the current LSF models. It is simple and geographically independent.

1.3.2 Strength-Of-Arrival and Neural-Network Integrated PL Technique

The SOA location estimates are improved by processing the SOA estimates by an NN. By using an NN, there is no need to use different methods depending upon the position of the MS. Although SOA and NN are two widely known techniques, integrating these two techniques has proved to be efficient in PL estimation.

1.4 Thesis Outline

The thesis is organized in the following way. In Chapter 2, a description of three concepts related to the SOCLSFM radio channel model and results of its evaluation are given. The five steps to create the SOCLSFM model and the results of evaluation are presented in terms of both received power and Root-Mean-Square (RMS) delay spread.

In Chapter 3, an SOA-based PL estimation model is developed so as to evaluate the SOCLSFM, and the detailed descriptions of the current SOA estimation technique and algorithms used in the PL system are provided. The location estimation performance of the SOA method is demonstrated using three criteria. Finally, the simulation results of the SOA technique, which show its poor performance, are given.

In Chapter 4, the general NN structure as well as the NN learning and training algorithms are introduced, as well as how the SOA and NN techniques are integrated. The simulation results cover different NN architectures. The performance of the NN during training and testing are given. The accuracy of the integrated PL technique is investigated. Simulation results show the superior performance of the integrated scheme.

In Chapter 5, the conclusion of the thesis is presented by summarizing the results of the proposed PL estimation scheme. Future work in the area of PL estimation is outlined.

Chapter 2 Splashes-Of-Changes Large-Scale Fading Model

2.1 Introduction

The rationale for a new and improved SOCLSFM was presented in Chapter 1. The traditional outdoor LSF models are realistic but very complicated. The SOCLSFM has sufficient complexity to exploit the large-scale correlation or multipath.

This chapter introduces the SOCLSFM for the PL estimation and details of the techniques involved in the implementation of the model. The SOCLSFM is a comprehensive model generated in five steps: random coefficient values, propagation delay, exponential power delay profile and RMS delay spread, path loss and log-normal shadowing, and SOC in RMS delay spread. Implementation of each step is based on adding new features to the previous step. The Path-Loss-Based LSF Model (PLBLSFM) is created using only the first three of five steps of the SOCLSFM. The PLBLSFM is a simple model and is used only to show that the SOA-based PL estimation technique has been implemented correctly; the PL estimation error should be zero.

Relevant literature and fundamental concepts employed in developing the SOCLSFM are described in the following sections of this chapter.

2.1.1 Outdoor Fading Environment

The outdoor fading environment forms the basis of the mathematical model discussed in Section 2.2. Generating outdoor fading models involves the following concepts:

- **Multipath** refers to the various propagation paths, which the transmitted signal takes before reaching the receiver. The transmitted signal is subject to random phase, amplitude change and time delay as it traverses the various paths.

- **Impulse Responses** with random coefficients.
- **Propagation delay** is the difference between the time instant a signal departs from a BS to the time instant the signal arrives at an MS. The time delay is directly proportional to the MS-BS distance [Garg96].
- **Power delay profile** is a plot of the average power versus time of the impulse responses.
- **RMS delay spread** is a measure of the width of the power delay profile.
- **Path loss** denotes the average loss of signal power with the distance between the MS and the BS [Rappaport96].
- **Log-normal shadowing** relates to vast changes in the received power at locations having the same MS-BS distance [Rappaport96].
- **SOC** is caused by obstacles in the BS to MS propagation path. It causes changes in RMS delay spread.

These concepts of the outdoor fading models are presented in detail in Section 2.2.

2.1.2 Multipath Propagation

Multipath propagation or multipath occurs both in LOS and NLOS environments, although more often in NLOS. The radio signal from an MS is reflected from objects in the propagation path, and is scattered throughout the area. The scattering phenomenon is very complex. At the BS, radio signals from the MS arrive via a number of different paths. The actual signal picked up by the receiver antenna is a combination of all of these different signal components. It is therefore convenient to characterize a multipath radio signal in terms of its impulse response. The random phase and amplitude of the different multipath components cause fluctuations in the strength of the received signal.

Another effect of multipath is time dispersion caused by multipath propagation delay. Signal components are subject to different time delays as they take different paths from the transmitting to receiving station. A component taking a shorter path will arrive earlier than that taking a longer path. Multipath signals which arrive at the receiving station usually cause time dispersion [Garg96]. This is considered in Section 2.2.

2.1.3 Discrete-Time Impulse Response

A bandlimited sampled time-invariant channel impulse response can be expressed as a function of time, where the time delay axis is divided into equally-spaced time delay segments [Rappaport96]. The received signal in a multipath channel consists of components that are characterized by varying fading effects [Rappaport96]. A useful statistical description of the fading is given by the discrete-time power delay profile, which gives the average energy or power, in each impulse response coefficient. Furthermore, the discrete-time model allows for Digital Signal Processing (DSP) techniques.

2.2 Development and Statistical Evaluation of the Model

In this section, the development and statistical evaluation of the channel model is presented. The five steps of the model were summarized in Section 2.1. In this section, the model is described in detail.

2.2.1 Random Coefficient Values

Consider the signal propagation around a BS. It can be represented by a Two-Dimensional (2-D) image as shown in Figure 2.1. A square grid with a width of 50 m denotes the spatial sampling of the 2-D signal propagated by the BS. The choice of a 50 m separation distance, *D*, between two grid points (spatial sampling points) is a trade-off between the total number of samples over a circular area of radius *R*, 2000 m, and the resolution. The larger the separation is between grid points, the smaller the total number of samples, and the poorer the resolution. However, the complexity of design is considerably decreased because of fewer sample points.

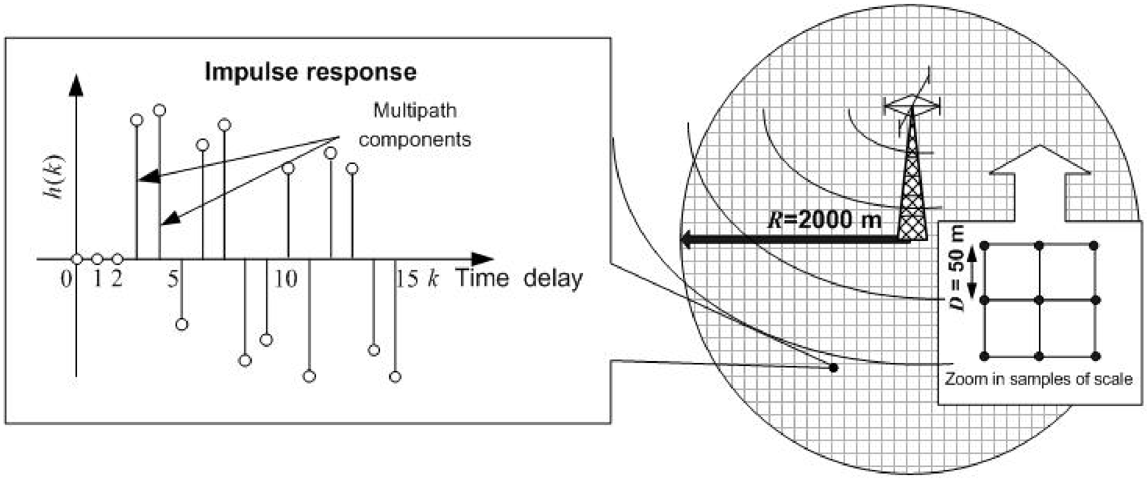


Figure 2.1 The 2-D SOCLSFM solution

Each grid point is the location where the propagated signal from the BS is received. A signal transmitted from the BS traverses various paths before arriving at the sample point and forms several signal multipath components (the reflected waves) received at different times. The propagation channel from the BS to this sample point is characterized by a discrete-time impulse response. An example of an impulse response with 16 taps of components is presented in Figure 2.1, in which the first three components have zero power received. The impulse response completely captures the behavior of the channel multipath effects.

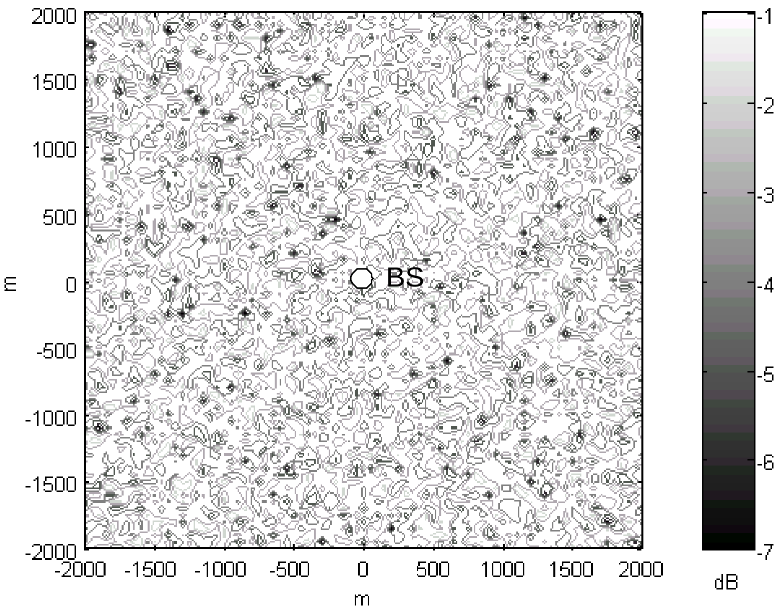


Figure 2.2 Contour plot of mean received power with random coefficient values

To develop the SOCLSFM, the first step is to produce random coefficient values at each tap of all the impulse responses throughout the cell. The number of impulse responses in a cell is 6561; see Figure 2.1. The mean received power is an important statistic, and is the sum of the squared magnitudes of the individual complex taps in the channel impulse responses. Figure 2.2 illustrates the contours of mean received power with random coefficient values generated for all 6561 impulse responses; the power received is expressed in dB. All of the important parameters employed in the generation of the impulse responses are listed in Table 2.1. N is the width, in samples, of each impulse response and is also the number of resolved multipath components. N is arbitrarily chosen to capture only multipath signals which travel a distance less than or equal to $2R$. N is calculated by:

$$N \approx \frac{2R}{c\left(\dfrac{1}{w_c}\right)} \quad . \tag{2.1}$$

It is assumed that all the impulse responses around the BS have the same number of taps, *N*.

Table 2.1 Some parameters involved in the random coefficient values generation

Parameters	Values
The velocity of light or radio signal (c)	2.998×10^{8} m/s
The bandwidth of CDMA (IS-95)(w_c) [Rappaport96]	1.25 MHz
The width of square grid (D)	50 m
The radius of the circle range around BS (R)	2000 m
The number of taps in impulse responses (N)	16

2.2.2 Propagation Delay

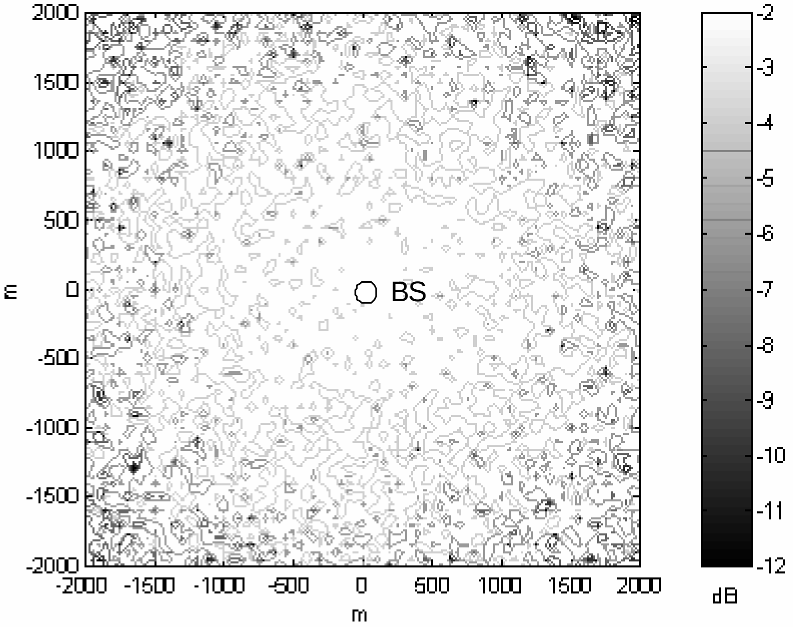


Figure 2.3 Contour plot of mean received power with propagation delay

Propagation delay is a measure of the time required for the radio signal to travel from the BS to an MS and it characterizes the MS-BS channel propagation path. The MS-BS distance is a major factor which affects propagation delay. Longer distance results in longer time for signals to be received. The propagation delay is represented by a time delay parameter in the each impulse response. An example of an impulse response is given in Figure 2.1, where the impulse response taps are zero over the initial time interval corresponding to the propagation delay. If the propagation delay is set to ‘2’, the impulse response will be zero over the time interval [0 2]. As a consequence, the received powers of taps will be zero until the time instant, ‘2’.

The second step in the procedure to generate the SOCLSFM is to zero the beginning of impulse responses according to the propagation delay. Another consequence of this step is that the mean received power, RSS, is lower at the cell boundary, as indicated in Figure 2.3.

2.2.3 Exponential Power Delay Profile and Root-Mean-Square Delay Spread

Power delay profile is generally represented as the plot of relative received power as a function of excess delay with respect to a fixed time delay reference [Rappaport96]. Excess delay is the relative delay of a multipath component as compared to the first arriving component [Rappaport96]. The time delay reference is time delay of the first arriving component, usually set to ‘0’.

In practice, the power delay profile is obtained by averaging a large set of impulse responses. The typical power delay profile is exponential and is commonly used to model an outdoor environment (urban, suburban, and rural areas). The mean excess delay (the first moment of the power delay profile) and RMS delay spread are multipath channel statistics that can be determined from a power delay profile. The main parameter, namely the RMS delay spread, is the second central moment of the power delay profile.

In micro-cellular channels, the value of the RMS delay spread is usually smaller and rarely exceeds a few hundred ns. Seidel and Rappaport reported delay spreads in four European cities of between 50 and 300 ns [Rappaport96].

Typically, the RMS delay spread measured in an NLOS indoor environment is approximately 50 to 250 ns [Medbo99]. However, in an LOS outdoor environment (rural area), this value is reduced to approximately 10 to 100 ns. NLOS often occurs in wireless communication systems, and a value of 100 ns is generally assumed for the RMS delay spread, as it represents an average RMS delay spread for the indoor and the outdoor environments. An RMS delay spread of 100 ns was chosen for the LSF model in the outdoor urban area.

An effective way to set the RMS delay spread in the exponential power delay profile is to set the parameter a , the exponential coefficient. The exponential coefficient a is an analytical function of the RMS delay spread (s_t). The derivation of a given the RMS delay spread, s_t , of 100 ns, proceeds as follows:

Excess delay, *k*, is defined as the relative time delay of the *k*th multipath component as compared to the first arriving component (*k* = 0).

The exponential power delay profile P_k is given:

$$P_k = \begin{cases} e^{-2ak} & ,k \in \{0, 1, 2, 3, ...\} \\ 0 & ,k \in \{...,-3,-2,-1\} \end{cases}. \qquad (2.2)$$

The first moment of power delay profile (mean excess delay) is:

$$\bar{t} = \frac{\sum\limits_{k=0}^{\infty} kP_k}{\sum\limits_{k=0}^{\infty} P_k}. \qquad (2.3)$$

The second moment of power delay profile is:

$$\overline{t^{2}} = \frac{\sum_{k=0}^{\infty} k^{2} P_k}{\sum_{k=0}^{\infty} P_k} . \tag{2.4}$$

The RMS delay spread is:

$$s_{t} = \sqrt{\overline{t^{2}} - \left(\bar{t}\right)^{2}} . \tag{2.5}$$

By combining Equations (2.2) to (2.5), two possible solutions, $a_{1,2}$, of the exponential coefficient may be found. The solutions are a function of s_t , as defined in Equation (2.6),

$$a_{1,2} = \frac{\ln\left(\frac{b \pm \sqrt{b^{2}-4}}{2}\right)}{2}, \quad \text{where} \quad b = \frac{1}{s_{t}^{2}} + 2 \tag{2.6}$$

The detailed computation of $a_{1,2}$ is described in Appendix B.

The positive exponential coefficient, a_1 , is chosen. Thus, the exponential power delay profile is:

$$P_k = e^{-2 a_1 k} , \tag{2.7}$$

where k indicates the excess delay. Given $s_t = 100\,\text{ns} \times w_c$, a_1 was found to be 2.09. Figure 2.4 is the plot of the power delay profile in Equation (2.7).

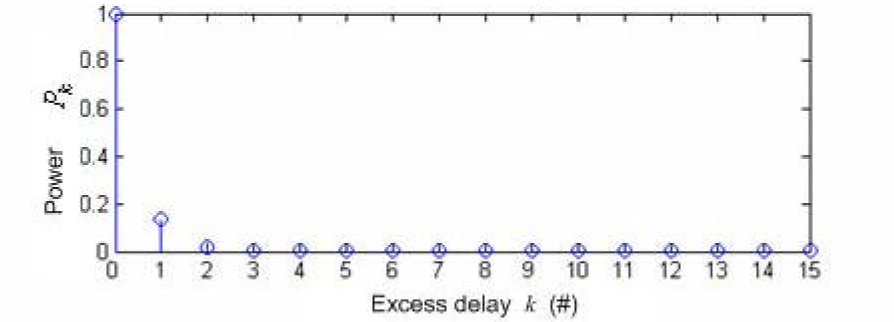


Figure 2.4 Discrete-time exponential power delay profile (a_1 : 2.09 , s_t : 0.125)

In Appendix B, the derivation of a_1 was based on the z transform, where the variable k in the formulas given by Equations (2.3) - (2.5) defined on an infinite set. However, in a physical system, k takes on finite integer values, $\{0,1,2,\ldots,N-1\}$, (N = 16 in the simulation as indicated in Figure 2.4).

RMS delay spread contours measure the time width of reflected waves, and are defined by Equations (2.3) - (2.5). It is probably the most important single measure for the time width of a multipath radio channel.

The third step in the procedure to generate the SOCLSFM is to include the effect of the power delay profile. It is included simply by taking the power delay profile from Figure 2.4, shifting it in time, and multiplying by the impulse responses from step two. For example, in Figure 2.4, the value of P_k at k = 0 multiplies the value of $h(3)$ in Figure 2.1; the value of P_k at k = 1 multiplies the value of $h(4)$; this pattern is continued.

Figures 2.5 and 2.6 present the results of mean received power, and RMS delay spread contours, respectively, after the inclusion of the exponential power delay profile. Comparing Figure 2.3 and Figure 2.5, it can be seen that the exponential power delay profile decreases the mean received power and it is evenly spread around the BS. RMS delay spread contours are computed using Equation (2.5). Figure 2.6 shows the RMS delay spread contours. The values of RMS delay spread are shown in the range from 0.2 to 1.4. It shows a random and even distribution of RMS delay because the impulse responses are random.

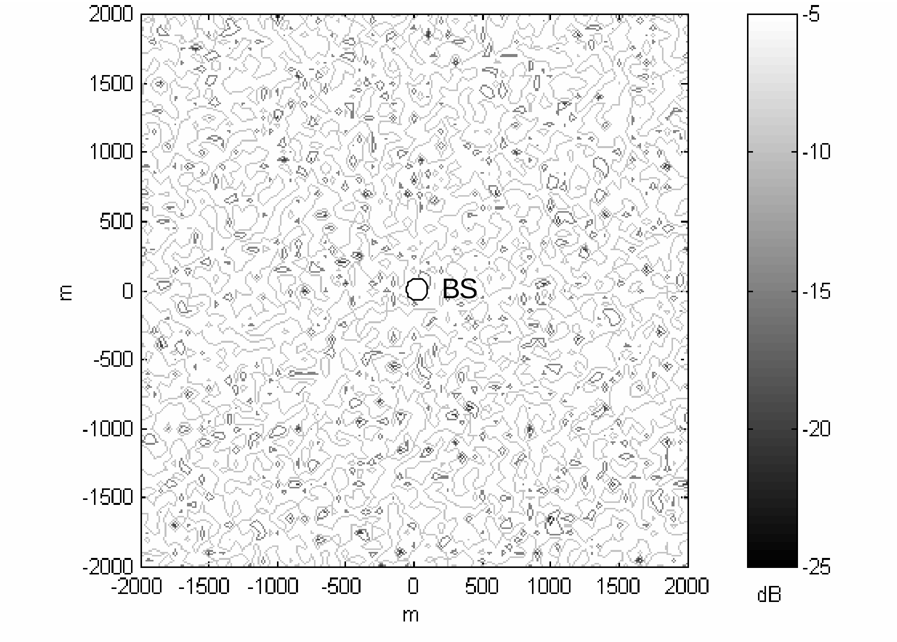


Figure 2.5 Contour plot of mean received power after adding the exponential power delay profile

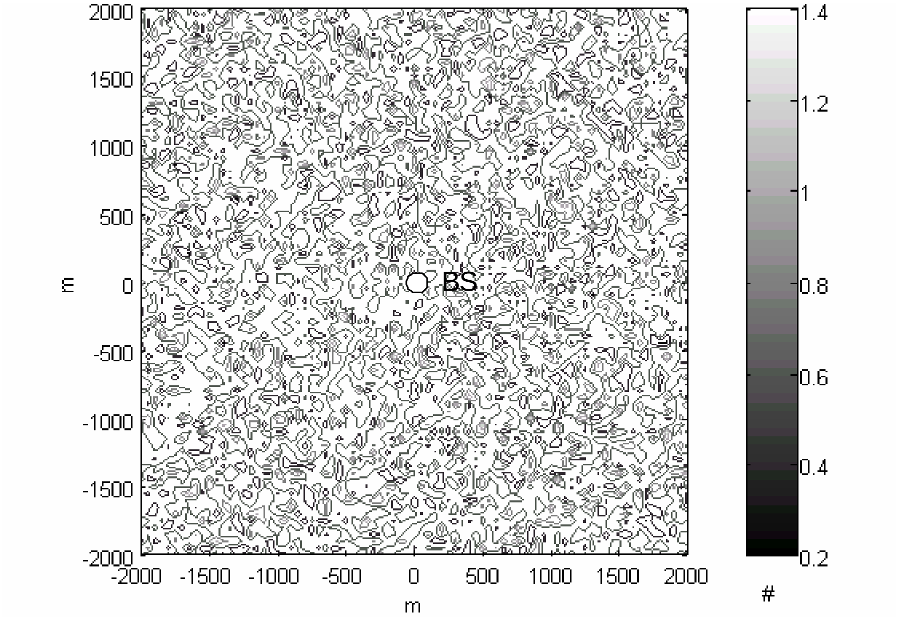


Figure 2.6 Contour plot of RMS delay spread after adding the exponential power delay profile

2.2.4 Path Loss and Log-Normal Shadowing

A theoretical propagation model is obtained by including the effect of path loss as a function of distance for the received signal power. Both theoretical and measurement-based propagation models indicate that the average received signal power attenuates or decreases logarithmically with the distance. However, a model that takes only the path loss into consideration, does not account for the effect of the propagation environment. For example, clutter of objects along two different propagation paths having the same BS-MS distance may be vastly different [Rappaport96]. As a result, actual path loss for the two paths will be different. However, the path loss predicted by the formula based on the distance will be the same. To simulate the effect of clutter, log-normal shadowing is added to the path loss. The modified formula for path loss, which takes into account the distance as well as the effect of the environment (e.g. clutter of objects in the propagation path) is derived in terms of the path loss exponent (*n*) and a Gaussian-distributed random variable termed the log-normal shadowing (X_s). The modified path loss formula in Equation (2.8) is a combination of the path loss formula relating path loss with distance and an additive term (which is the log-normal shadowing):

$$PathL_{dB}(d_s) = \overline{PathL}_{dB}(d_0) + 10n\log\left(\frac{d_s}{d_0}\right) + X_s \quad . \qquad (2.8)$$

d_s indicates the MS-BS separation distance. d_0 denotes the close-in reference distance. $PathL_{dB}(d_s)$ is the path loss power in dB. *n* is the path loss exponent, which depends on the surroundings and obstacle type X_s is a Gaussian random variable having a standard deviation of s.

As mentioned in Section 2.1, the large-scale path loss model without shadowing is referred to, in this thesis, as PLBLSFM. The variance of path loss depends highly on the environment. The path loss exponent, *n*, in the NLOS environment, outside of buildings often varies from 2.7 to 4. However, this value is smaller (1.6 – 1.8) in the LOS indoor environment [Rappaport96]. Based on experiments, the standard deviation s was found to be around 11.8 dB in urban areas [Rappaport96].

To evaluate the large-scale path loss model given by Equation (2.8), parameters in the formula are assigned reasonable values as listed in Table 2.2. The term $\overline{PathL}_{dB}(d_0)$ is arbitrarily assigned a reference value of 0 dB.

Table 2.2 Parameters selected in path loss and log-normal shadowing

Parameters	**Values**
The reference distance (d_0)	10 m
The reference path loss at distance d_0 ($\overline{PathL}_{dB}(d_0)$)	0 dB
Shadowing variance (s)	11.8 dB
Path loss exponent (n)	4

The fourth step in creating the SOCLSFM is to include the effects of path loss and shadowing. This is done by evaluating Equation (2.8) for $PathL_{dB}(d)$ to obtain specific numerical values for each location in the cell; shown in Figure 2.1. These numerical values are assigned to be the energy of each of the impulse responses obtained from step three.

The numerical values of path loss, with shadowing, are shown in a contour plot in Figure 2.7. Setting the path loss does not change the RMS delay spread since the RMS delay spread is already normalized by the power. The RMS delay spread contours would be the same as Figure 2.6. The mean received power, which is displayed as contours, decreases logarithmically (obeying the path loss with distance formula) from the BS to the cell boundary with vastly different shadowing at each contour edge, as shown in Figure 2.7.

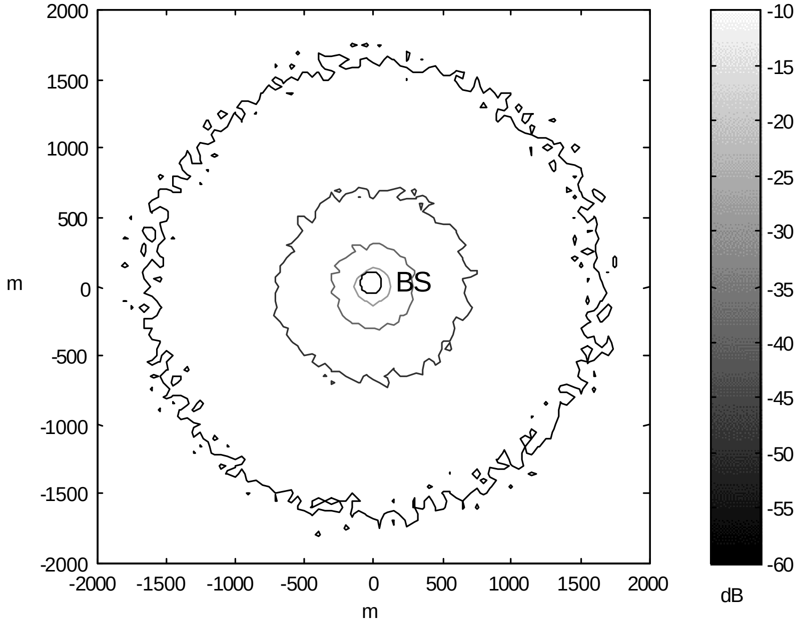


Figure 2.7 Contour plot of mean received power after adding path loss and log-normal shadowing

2.2.5 Splashes-Of-Change in Root-Mean-Square Delay Spread

The received signal may be subject to reflection, diffraction, and scattering from surrounding buildings or other obstacles. This phenomenon of so-called multipath propagation produces SOCs in the RMS delay spread. The LSF model is adjusted using the experimental data to include the SOCs in the RMS delay spread. This is the fifth and final step to generate the SOCLSFM.

SOC refers to changes made to RMS delay spread contours. Previously generated RMS delay spreads are random. In step three to generate the SOCLSFM, the constant exponential coefficient, a_1, from Equation (2.7) was assigned to all the channel impulse responses. In the multipath environment, the signal transmitted from a BS is received by the MS via multiple propagation paths. The transmitted signal may encounter obstacles, and, as a result, the received signal is subject to distortion including multipath depending upon the position, size and shape of the obstacles. The larger the number of obstacles spread randomly around the BS, the greater the number of reflected waves received by the MS located beside the obstacles. Each SOC in RMS delay spread, which will be introduced in step five, corresponds to a set of obstacles in the propagation path.

SOCs are put in by increasing the exponential coefficient, a_1, at the power delay profiles of the impulse responses over a small area of the cell called a “splash”. This effect, termed SOC, assumes that the various types of obstacles are randomly distributed in the cellular area. The SOC effect is simulated by assuming that a_1 is varied as a 2-D Gaussian random variable over the splash, which results in “bell-curve changes” in the RMS delay spread contours [Soma99]. The SOCs in RMS delay spread will not change the path loss generated in step four.

SOCs in RMS delay spread are generated using the following six elements:

- **Element 1:** Choose the location of one splash

Since an SOC occurs at random, a uniformly distributed random point, (x_r, y_r), is chosen inside a circular area of radius 2000 m for locating the center of the splash.

- **Element 2:** Generate the height of one splash, *H*

A 2-D bell curve's height, *H*, is a Rayleigh Random Variable (RV). The mean of the RV is set to $H_S = 2.0$, $H_S = E[H]$, where $E[\ ]$ denotes mathematical expectation [Iyanaga80]. The standard deviation, s_H, is derived from H_S using

$$s_H = H_S\sqrt{\frac{2}{p}}\ . \qquad (2.9)$$

A sample point from the distribution of *H* is generated using a standard numerical technique [Knuth81].

- **Element 3:** Generate the width of one splash, *W*

The width of one splash, *W*, is a Rayleigh RV with a mean $W_S = 10$ m. A sample point from the distribution of W is generated [Knuth81].

- **Element 4:** Generate the shape of the splash, $B_c(x,y)$ using x_r, y_r, H and *W*.

The shape of the splash is based on a 2-D circular Gaussian probability density function, $P_{x,y}(x,y)$, given by

$$P_{x,y}(x,y) = \frac{1}{2ps^2}\left(e^{\frac{-(x-\bar{x})^2}{2s^2}} \right)\left(e^{\frac{-(y-\bar{y})^2}{2s^2}} \right), \qquad (2.10)$$

where ($\bar{x}$, $\bar{y}$) are the coordinates of the center of the distribution and s is the standard deviation [Iyanaga80]. Generalizing Equation (2.10), the shape of the splash is given by

$$B_C(x,y) = H\left(e^{\frac{-(x-x_r)^2}{2W^2}} \right)\left(e^{\frac{-(y-y_r)^2}{2W^2}} \right), \qquad (2.11)$$

where coordinates x and y are variables. The coordinates take values at grid points, which are 50 m apart, within the circular area of radius 2000 m.

- **Element 5:** Generating the randomness of the splash, $R_c(x,y)$ using $B_c(x,y)$

$B_c(x,y)$ and $R_c(x,y)$ are used in combination to increase the RMS delay spread throughout the splash. The increases are statistically higher in the center of the splash, as determined using $B_c(x,y)$, but the increases must also be random within the splash, as determined using $R_c(x,y)$. $R_c(x,y)$ is a set of sample points from independent Rayleigh RVs with mean $B_c(x,y)$.

- **Element 6:** Increasing the RMS delay spread of impulse responses in one splash

By using the randomness $R_c(x,y)$ from step five, with the previous RMS delay spread ($PD_S\ (x,y)$), a modified RMS delay spread ($ND_S\ (x,y)$) is obtained,

$$ND_S(x,y) = (1 + R_c(x,y))PD_S(x,y)\,. \qquad (2.12)$$

In step three, the exponential coefficient as denoted using a_1 . For clarity of presentation in this section, let $a_p = a_1$. Based Equation (2.6), for the old exponential coefficient (a_p), the new exponential coefficient (a_n) can be expressed in terms of ND_S , as indicated in Equation (2.13):

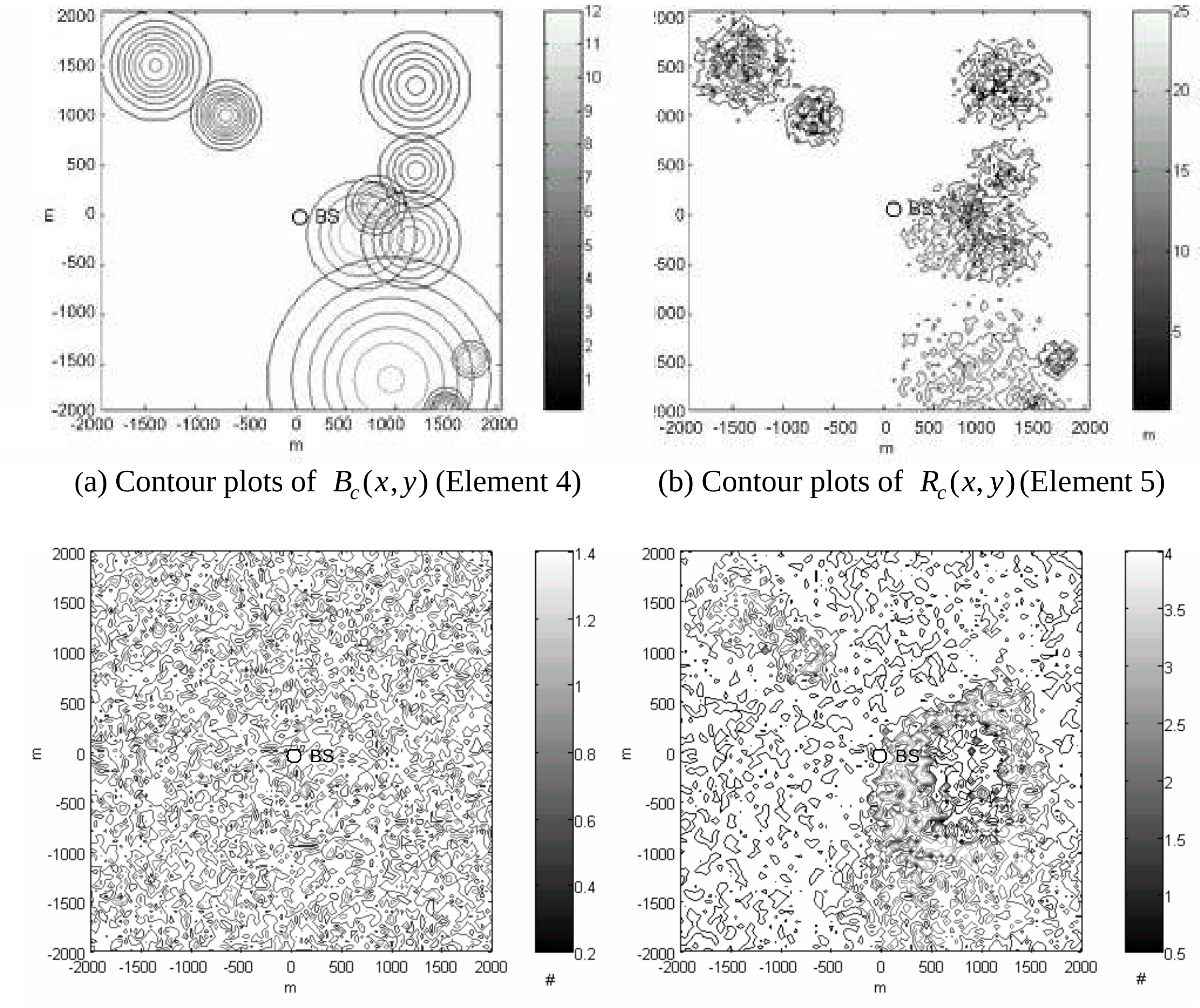
$$a_n(x,y) = \frac{\ln\left(\dfrac{b \pm \sqrt{b^2 - 4}}{2}\right)}{2}, \quad \text{where} \quad b = \frac{1}{ND_S\,(x,y)^2} + 2\,. \qquad (2.13)$$

For every sampling point with the coordinates (*x*, *y*), a modified impulse response ($h_n(k)$) is generated using the corresponding exponential coefficients given by Equations (2.6) and (2.13). $h_n(k)$, which relates to the power delay profile in Equation (2.7), is computed as

$$h_n(k) = e^{-2(a_n - a_p)k} h_p(k), \qquad (2.14)$$

where $h_p(k)$ is the previous impulse response, and k is the excess delay, $k = \{0\ ,1,2,\ldots, N-1\}$.

Elements 1 to 6 generate one SOC in RMS delay spread. However 10 SOCs are used in the SOCLSFM. Figure 2.8 shows the results of the 10 SOCs.



(a) Contour plots of $B_c(x,y)$ (Element 4)

(b) Contour plots of $R_c(x,y)$ (Element 5)

(c) RMS delay spread contours for $h_p(k)$

(d) RMS delay spread contours for $h_n(k)$ (Element 6)

Figure 2.8 Generating 10 SOCs in the RMS delay spread with many stages

The 10 bell curves for each $B_c(x,y)$ are shown in Figure 2.8 (a), with varying widths, amplitudes and locations in the cellular area. Contour plots of $R_c(x,y)$ are presented in Figure 2.8 (b), which shows the randomness in RMS delay spread changes. Figure 2.8 (c), the previously established RMS delay spreads from step four, is provided in contrast to Figure 2.8 (d). Figure 2.8 (d) contains the RMS delay spread contours with the 10 SOCs applied in step five. Figure 2.8 (d) shows the final form of the RMS delay spread used in the SOCLSFM.

The mean received power of an impulse response is modified by Equation (2.14). However, the mean received power must remain unchanged. By re-normalizing the mean received power of every impulse response, the mean received power contours remain the same shown in Figure 2.7.

2.3 Summary

In Chapter 2, five steps to develop a new LSF channel model called the SOCLSFM for the outdoor fading environment are presented. Impulse responses collect all the information used for describing the SOCLSFM inside a multipath environment. The model is developed sequentially in five steps. Although the previous four steps do provide enough information about the channel, they do not consider the SOC effects, which are critical to introduce correlation among impulse responses. Adding SOCs is a new approach in developing an LSF model. Chapter 3 presents SOA-based PL estimation using both the PLBLSFM and SOCLSFM.

Chapter 3 Evaluation of Fading Models using Strength-Of-Arrival Techniques

3.1 Introduction

In this chapter, the existing SOA technique is applied to evaluate the fading models (PLBLSFM, SOCLSFM) introduced in Chapter 2. The PLBLSFM and the SOCLSFM are used and the algorithms for the PL estimation using the SOA technique are introduced. For simplicity of analysis, it is assumed that the channel allocated to the users requesting the PL service is stationary. The SOA-based PL estimation is accomplished in two stages. The first stage involves the estimation of the received power by each BS through the use of a path-loss technique which computes the loss of power of the signal traversing a particular path. The estimated SOAs are transformed into range difference measurements between BSs. The range difference measurements form a set of difference equations. The second stage utilizes efficient algorithms to produce an unambiguous solution to this set of equations, from which the estimate of the PL is obtained. The performance evaluation of the SOA-based PL estimation scheme is presented. Both the PLBLSFM and the SOCLSFM are considered, and the limitations of using the SOA technique alone are discussed.

Sections 3.2 and 3.3 introduce the model and algorithms respectively.

3.2 Strength-Of-Arrival-Based Scheme

The SOA scheme consists of the PLBLSFM and SOCLSFM fading models and the PL estimation.

3.2.1 Splashes-Of-Change Large-Scale-Fading Models for Three Base Stations

Assuming a 2-D spatial geometry, the signal levels from three BSs are described by the PLBLSFM or the SOCLSFM. For simplicity, only the SOCLSFM is illustrated. Chapter 2 described the SOCLSFM centered around one BS. However, three SOCLSFMs are required, each centered around its respective BS; see Figure 3.1. All the SOCLSFMs are generally described in terms of fixed parameters which govern the statistical behavior of the random aspects of the SOCLSFM. However, the SOCLSFM is a time-invariant model and is location dependent. Many generated model parameters are random. As a result, at each time of generation of the SOCLSFM, a so-called unique propagation set is formed. We restrict the number of BSs to the minimum required for an unambiguous PL estimation, namely three. Figure 3.1 shows the three fading models, each having a radius of 2000 m around the BS. The three BSs are presumed to be located at each corner of an equilateral triangle, whose coordinates are BS1 = (0 m, 0 m), BS2 = (0 m, 1500 m), and BS3= (750 m, 1299 m), respectively. Each side is 1500 m in length, and the stationary MS users are assumed to be located within a shaded circular area of radius 866 m centered at (750 m, 433 m). The center of the shaded area coincides with that of the triangle, as shown in Figure 3.1.

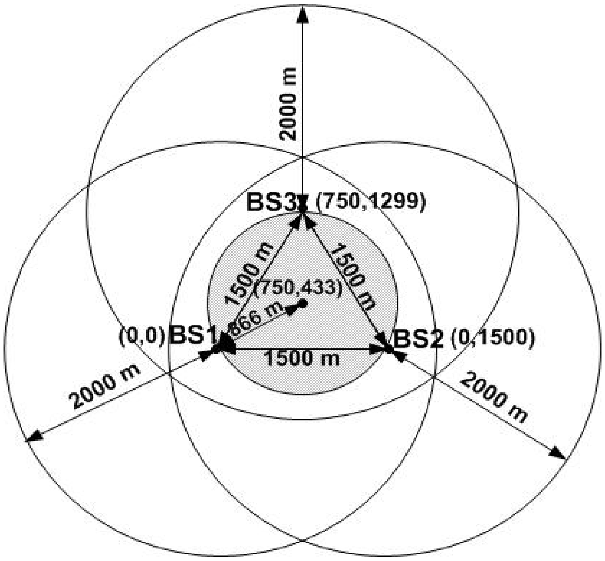


Figure 3.1 The configurations of the PL model over three BSs

3.2.2 Development of Strength-Of-Arrival-Based PL Estimation Scheme

An overview of the SOA-based scheme is as follows. The three impulse responses derived from the SOCLSFMs and the true MS location coordinates are fed to the SOA-based PL estimation scheme. The output of the SOA-based PL estimator is the estimate of the location co-ordinates of the MS. The estimation error is computed from the true and the estimated location co-ordinates and the performance of the estimator is analyzed in terms of the mean-squared estimation error. See Figure 3.2.

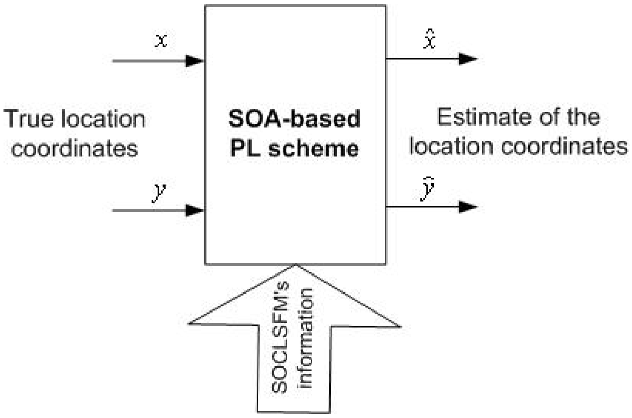


Figure 3.2 SOA-based PL estimation scheme

The detail of the SOA-based scheme is as follows:

- Repeat the following steps for each BS with different MS's relative coordinates, respectively:

  1. The signal propagation around three BSs is sampled uniformly in a 2-D space, where the sample points are located at the vertices of squares 50 m wide. An MS can be possibly located within the four sample points represented by the corners of the square. Impulse responses at these four sample points can be obtained from the SOCLSFM.

  2. The impulse response at the MS is computed from the interpolation of the

impulse responses at the corners of the square.

3. The power of the impulse response is computed.

- Using the set of difference equations governing the range difference measurements, three circles are drawn.
- The points of intersection of the three circles can form one or two triangles. The estimate of the MS location is obtained weighting the centers of the triangles.

The SOA scheme is described in detail in Section 3.3.

3.3 Mathematical Formulation of Strength-Of-Arrival-Based Scheme

This section describes the mathematical formulation of the SOA-based PL scheme.

3.3.1 Received Power Estimation using Three Propagation Sets

3.3.1.1 Mobile at some random location

It is presumed that the MSs are scattered uniformly inside a circle with a radius of 866 m and centered at the coordinate (750 m, 433 m). Let the coordinates of the MS positions be indicated by (x, y).

3.3.1.2 Measurement of received powers and development of range difference equations

Using the mathematical model governing the signal path loss with distance, the three received powers, denoted P_1, P_2 and P_3, are given by

$$P_1 = m\, d_1^{-n}, \tag{3.1a}$$

$$P_2 = m\, d_2^{-n}, \text{ and} \tag{3.1b}$$

$$P_3 = m\, d_3^{-n}, \tag{3.1c}$$

where d_1, d_2 and d_3 are the distances from the MS to three BSs; m is the path loss coefficient and n is the path loss exponent.

In order to estimate the received powers from each of the three propagated signals, the impulse response is computed from an averaging of the impulse responses at the sample points of the square grid using coordinate shifting and interpolation. The absolute coordinates of the three propagation sets are centered at the three BSs, having coordinates (0 m, 0 m) at each BS, and three circles having a 2000 m radius, as shown in Figure 2.3. An MS has three different relative coordinates for each absolute coordinate. As illustrated in Figure 3.3, the MS's relative coordinate for BS1 is (100 m,100 m), the MS's relative coordinate for BS2 is (-1400 m, 100 m), and the MS's relative coordinate for BS3 is (-650 m, -1199 m). Three relative coordinates are used to identify the three different impulse responses, namely MS-BS1, MS-BS2 and MS-BS3.

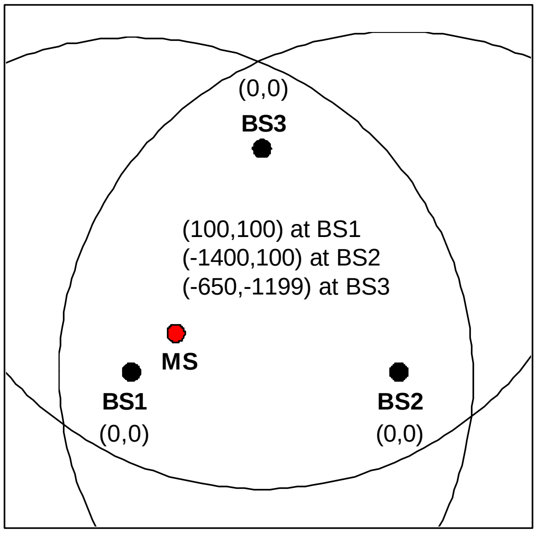


Figure 3.3 Relative coordinates of an MS

As mentioned before, the impulse responses are identified at the grid points located at the vertices of a square 50 m wide. The impulse response of the channel at the MS, which is located inside the square, is computed by interpolation of the four impulse responses at the vertices of the square.

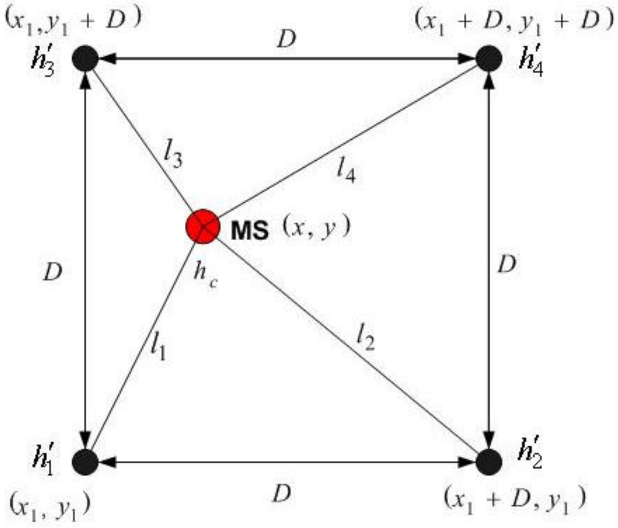


Figure 3.4 An impulse response at an MS estimated in a square pattern

Figure 3.4 shows the grid points located at coordinates (x_1,y_1), (x_1+D,y_1), (x_1,y_1+D), and (x_1+D, y_1+D). The impulse responses h_1', h_2', h_3', and h_4', are identified at these points respectively, where D is the width of the square (D = 50 m). The MS is located at the coordinate (x, y), and the distances from the MS to the four grid points are l_1, l_2, l_3, and l_4.

The impulse response where the MS is at (x, y), h_c, is obtained by interpolation as

$$h_c = w_1 h_1' + w_2 h_2' + w_3 h_3' + w_4 h_4', \tag{3.2}$$

where w_1, w_2, w_3, and w_4 are the weights. The weights are computed from the distances to the four vertices, l_1, l_2, l_3, and l_4, and the width of the square, D, as follows:

$$\begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \end{bmatrix}, \tag{3.3}$$

where $a_{11}, a_{12}, a_{13}, ..., a_{44}$ are the 16 constraints that need to be determined to create the interpolator. The 16 constraints are obtained from the MS positions at the four grid points, with four constraints on the weights at each point. The constraints at (x_1, y_1) are

$$1 = a_{11}(0) + a_{12}(D) + a_{13}(D) + a_{14}(\sqrt{2}D)\,, \tag{3.4a}$$
$$0 = a_{21}(0) + a_{22}(D) + a_{23}(D) + a_{24}(\sqrt{2}D)\,, \tag{3.4b}$$
$$0 = a_{31}(0) + a_{32}(D) + a_{33}(D) + a_{34}(\sqrt{2}D)\,, \text{ and} \tag{3.4c}$$
$$0 = a_{41}(0) + a_{42}(D) + a_{43}(D) + a_{44}(\sqrt{2}D)\,. \tag{3.4d}$$

The constraints at $(x_1 + D, y_1)$ are

$$0 = a_{11}(D) + a_{12}(0) + a_{13}(\sqrt{2}D) + a_{14}(D)\,, \tag{3.4e}$$
$$1 = a_{21}(D) + a_{22}(0) + a_{23}(\sqrt{2}D) + a_{24}(D)\,, \tag{3.4f}$$
$$0 = a_{31}(D) + a_{32}(0) + a_{33}(\sqrt{2}D) + a_{34}(D)\,, \text{ and} \tag{3.4g}$$
$$0 = a_{41}(D) + a_{42}(0) + a_{43}(\sqrt{2}D) + a_{44}(D)\,. \tag{3.4h}$$

The constraints at $(x_1, y_1 + D)$ are

$$0 = a_{11}(D) + a_{12}(\sqrt{2}D) + a_{13}(0) + a_{14}(D)\,, \tag{3.4i}$$
$$0 = a_{21}(D) + a_{22}(\sqrt{2}D) + a_{23}(0) + a_{24}(D)\,, \tag{3.4j}$$
$$1 = a_{31}(D) + a_{32}(\sqrt{2}D) + a_{33}(0) + a_{34}(D)\,, \text{ and} \tag{3.4k}$$
$$0 = a_{41}(D) + a_{42}(\sqrt{2}D) + a_{43}(0) + a_{44}(D)\,. \tag{3.4l}$$

The constraints at $(x_1 + D, y_1 + D)$ are

$$0 = a_{11}(\sqrt{2}D) + a_{12}(D) + a_{13}(D) + a_{14}(0)\,, \tag{3.4m}$$
$$0 = a_{21}(\sqrt{2}D) + a_{22}(D) + a_{23}(D) + a_{24}(0)\,, \tag{3.4n}$$
$$0 = a_{31}(\sqrt{2}D) + a_{32}(D) + a_{33}(D) + a_{34}(0)\,, \text{ and} \tag{3.4o}$$
$$1 = a_{41}(\sqrt{2}D) + a_{42}(D) + a_{43}(D) + a_{44}(0)\,. \tag{3.4p}$$

Equations (3.4a) to (3.4p) can be solved for $a_{11}, a_{12}, a_{13}, \ldots, a_{44}$, and substituted into Equation (3.3) to get the weights,

$$w_1 = -\frac{\sqrt{2}\,l_1}{2D} + \frac{l_2}{2D} + \frac{l_3}{2D}\,, \tag{3.5a}$$
$$w_2 = \frac{l_1}{2D} - \frac{\sqrt{2}\,l_2}{2D} + \frac{l_4}{2D}\,, \tag{3.5b}$$

$$w_3 = \frac{l_1}{2D} - \frac{\sqrt{2}\, l_3}{2D} + \frac{l_4}{2D}\text{, and} \tag{3.5c}$$

$$w_4 = \frac{l_2}{2D} + \frac{l_3}{2D} - \frac{\sqrt{2}\, l_4}{2D}\,. \tag{3.5d}$$

With the method to obtain one interpolated impulse response, h_c, now described, this method can be repeated for all three BSs. In summary, the MS-BS1 impulse response, called h_1, can be obtained by:

- using the MS's relative coordinates to get h_1', h_2', h_3', and h_4',
- finding w_1, w_2, w_3, and w_4 from Equations (3.5a), (3.5b), (3.5c), and (3.5d),
- finding h_c from Equation (3.2), and
- assigning this h_c to h_1.

Similarly, the MS-BS2 impulse response, called h_2, and the MS-BS3 impulse response, called h_3, are obtained. There detail graphs in generating these impulse responses are shown in Appendix C. From h_1, h_2, and h_3, their respective powers P_1, P_2, and P_3, may be obtained.

Equations (3.1a), (3.1b), and (3.1c) can be transformed into three range difference equations, as expressed in Equations (3.6a), (3.6b), and (3.6c),

$$\left(\frac{P_1}{P_2}\right)^{\frac{2}{n}} = \left(\frac{d_2}{d_1}\right)^2 \Rightarrow d_2^2 = \left(\frac{P_1}{P_2}\right)^{\frac{2}{n}} d_1^2\,, \tag{3.6a}$$

$$\left(\frac{P_1}{P_3}\right)^{\frac{2}{n}} = \left(\frac{d_3}{d_1}\right)^2 \Rightarrow d_3^2 = \left(\frac{P_1}{P_3}\right)^{\frac{2}{n}} d_1^2 \text{ , and} \tag{3.6b}$$

$$\left(\frac{P_2}{P_3}\right)^{\frac{2}{n}} = \left(\frac{d_3}{d_2}\right)^2 \Rightarrow d_3^2 = \left(\frac{P_2}{P_3}\right)^{\frac{2}{n}} d_2^2\,. \tag{3.6c}$$

Since $P_1, P_2,$ and P_3 are known, assigning

$$\left(\frac{P_1}{P_2}\right)^{\frac{1}{n}} = a_1 \ , \tag{3.7a}$$

$$\left(\frac{P_1}{P_3}\right)^{\frac{1}{n}} = a_2 \ \text{, and} \tag{3.7b}$$

$$\left(\frac{P_2}{P_3}\right)^{\frac{1}{n}} = a_3 \, . \tag{3.7c}$$

Substitute Equations (3.7a), (3.7b), and (3.7c) into Equations (3.6a), (3.6b), and (3.6c) to get:

$$d_2^2 = a_1^2 d_1^2 \ , \tag{3.8a}$$

$$d_3^2 = a_2^2 d_1^2 \ \text{, and} \tag{3.8b}$$

$$d_3^2 = a_3^2 d_2^2 \, . \tag{3.8c}$$

Each of the above equations describes a circle, and the intersections are used to find the MS's location. Ideally, the circles intersect at a point. Practically, the intersections form a region and the MS's location is estimated from that that region. Section 3.3.2 gives the details.

3.3.2 Position Estimate of MS

3.3.2.1 Solutions to the three range difference equations

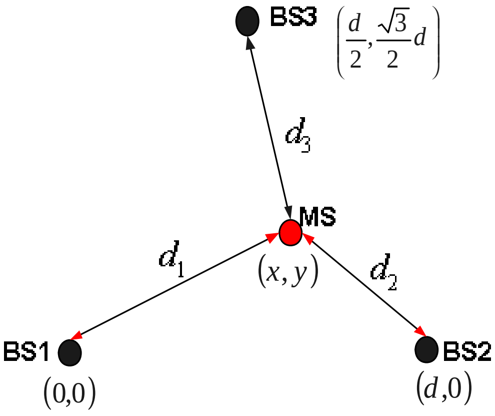


Figure 3.5 The MS-BSs related graph

The first of three pairs of circles considered is from BS1 and BS2. The coordinates of BS1, BS2, and the MS, and the length of one side of the isolates triangle formed by three BSs, d, are indicated in Figure 3.5. Equation (3.8a) represents a circle, between BS1 and BS2, whose equation is

$$(x-d)^2+(y-0)^2=a_1^2\left[(x-0)^2+(y-0)^2\right]. \tag{3.9}$$

Simplifying we get

$$\left(x-\frac{d}{1-a_1^2}\right)^2+y^2=\left(\frac{a_1 d}{1-a_1^2}\right)^2. \tag{3.10}$$

The coordinate at the center, (x_{c1},y_{c1}), and the radius, r$_1$, of the circle are given by

$$(x_{c1},y_{c1})=\left(\frac{d}{1-a_1^2},0\right)\text{, and} \tag{3.11a}$$

$$r_1=\left(\frac{a_1 d}{1-a_1^2}\right). \tag{3.11b}$$

Similarly Equation (3.8b) represents a circle between BS1 and BS3, whose coordinate at the center (x_{c2},y_{c2}) and whose radius, r$_2$, are

$$(x_{c2},y_{c2})=\left(\frac{d}{2(1-a_2^2)},\frac{\sqrt{3}d}{2(1-a_2^2)}\right)\text{, and} \tag{3.12a}$$

$$r_2=\left(\frac{a_2 d}{1-a_2^2}\right), \tag{3.12b}$$

and Equation (3.8c) represents a circle between BS2 and BS3 whose center, (x_{c3},y_{c3}), and the radius, r$_3$, are

$$(x_{c3},y_{c3})=\left(\frac{d-2da_3^2}{2(1-a_3^2)},\frac{\sqrt{3}d}{2(1-a_3^2)}\right)\text{, and} \tag{3.13a}$$

$$r_3 = \left(\frac{a_3 d}{1 - a_3^2} \right). \tag{3.13b}$$

The PL estimate of the MS is computed from the intersection points of the three circles. The estimate is a weighted average of these intersection points.

3.3.2.2 Computation of the position location estimate of mobile station

How the three circles intersect greatly influences the PL estimate of the MS. Appendix D illustrates the influence of the points of intersection. In the case where the circles do not intersect, we can use an approximate method by generating a fictitious point of intersection by giving one of the received powers small increments until the circles intersect at a point, as explained in Appendix D. The PL estimate is the centroid of the triangle formed by the three points of intersection,

$$\hat{x} = \frac{1}{3} x_1' + \frac{1}{3} x_2' + \frac{1}{3} x_3', \tag{3.14a}$$

$$\hat{y} = \frac{1}{3} y_1' + \frac{1}{3} y_2' + \frac{1}{3} y_3', \tag{3.14b}$$

where $\hat{x}$ and $\hat{y}$ are the coordinates of the PL estimate of the MS and (x_1', y_1'), (x_2', y_2') and (x_3', y_3') are the coordinates of the points of intersection of the circles.

3.4 Results of Evaluation

In the simulations in this thesis, four parameters relevant to the performance are:

- N_t , the total number of MS positions that are evaluated,
- N_l , the number of MS locations where the PL scheme could locate the MS,
- N_o , the number of outages, meaning the number of MS positions where the PL scheme failed to locate the MS, and
- N_a , the number of MS positions where the distance between the true and

estimated position is less than or equal to 125 m.

Some important relationships are:

$$N_l + N_o = N_t\text{, and} \tag{3.15a}$$

$$N_a < N_l\,. \tag{3.15b}$$

In order to evaluate the SOA-based PL estimate, three performance criteria are applied. They are based on the FCC's E-911 PL requirement (<125 m accuracy for 67 % measurements) described in Chapter 1.

The first criterion is the mean error (m) of estimation (e):

$$e = \frac{1}{N_l}\sum_{i=1}^{N_l}\sqrt{(\hat{x}_i - x_i)^2 + (\hat{y}_i - y_i)^2}\,, \tag{3.16}$$

where $\hat{x}$ and $\hat{y}$ indicate the coordinates of the estimate, and (x, y) are the coordinates of the true MS position.

The second criterion is the percentage (%) of the time location estimate cannot be determined (P_{outage}):

$$P_{outage} = \frac{N_o}{N_t} \times 100\% \tag{3.17}$$

The third criterion is the percentage (%) of estimations with errors less than 125 m ($P\%$):

$$P\% = \frac{N_a}{N_l} \times 100\% \tag{3.18}$$

The parameters used in the SOCLSFM are the same as those indicated in Chapter 2. Additional parameters used in the simulation are provided in Table 3.1.

Table 3.1 Parameters of SOA-based PL estimation

Parameters	Values
The distance between BSs (*d*)	1500 m
The square pattern length (*D*)	50 m
Path loss exponent (*n*)	4
Total numbers of MS locations simulated (N_t)	50

$N_t = 50$ MSs were chosen to evaluate the performance of the SOA-based PL estimation scheme with the SOCLSFM. 50 locations are enough to provide an acceptable variance on the performance criteria, as illustrated in Appendix E.

The results of the evaluation of the SOA-based PL estimation using both the PLBLSFM and the SOCLSFM are illustrated in Figure 3.6 and Figure 3.7, where the random locations of 50 MSs denoted by "*", the 50 estimated PLs denoted by "o", and three BSs denoted by "BS". The performance measures are presented in Table 3.2.

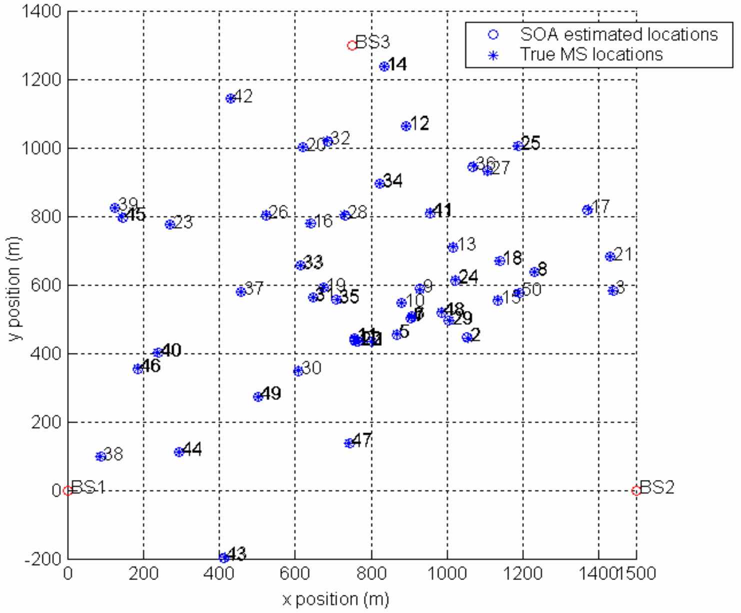


Figure 3.6 The SOA-based PL evaluated using the PLBLSFM
(50 MS locations, Mean error: 1.9 m)

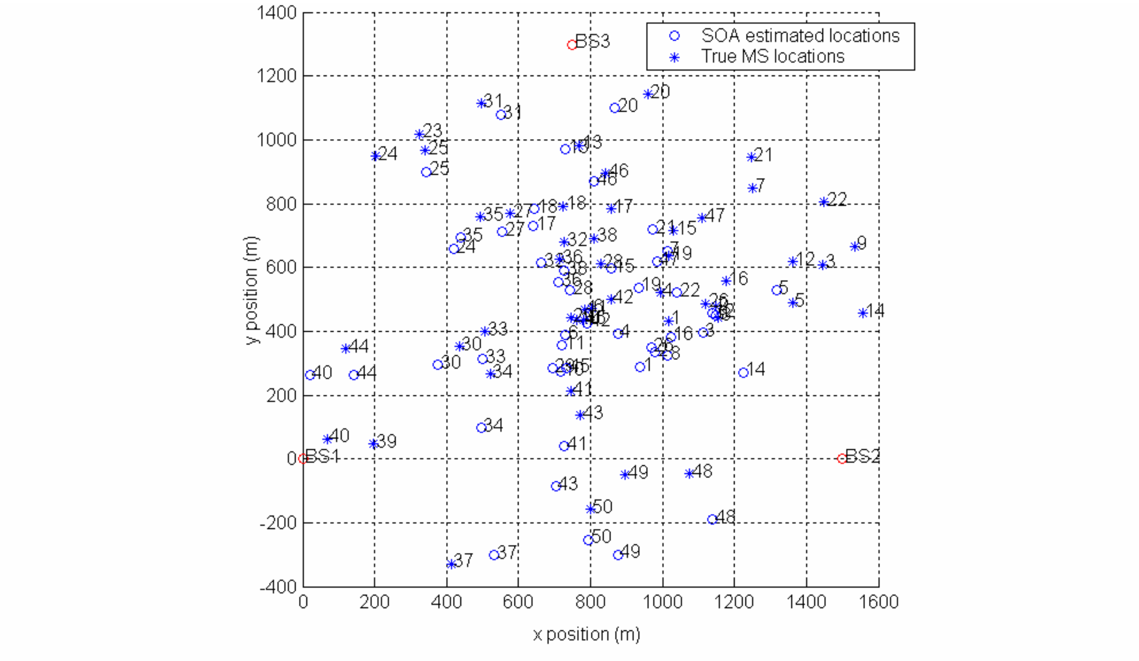


Figure 3.7 The SOA-based PL evaluated using the SOCLSFM
(50 MS locations, Mean error: 176.3 m)

Table 3.2 Performance measures for SOA-based PL evaluated using
the PLBLSFM and the SOCLSFM

SOA-based PL evaluated using the PLBLSFM	**SOA-based PL evaluated using the SOCLSFM**
$e = 1.9$ m	$e = 176.3$ m
$P\% = 100.0$ %	$P\% = 39.6$ %
$P_{outage} = 0.0$ %	$P_{outage} = 4.0$ %

The PLBLSFM is a fading model that allows for ideal performance. The SOA-based PL using the PLBLSFM gives good performance in all the locations. This demonstrates that the SOA-based PL estimation scheme is correct. Theoretically, in the absence of multipath and shadowing, a PL estimate with the PLBLSFM should give zero estimation error, that is in Figure 3.6 and Table 3.2, e should be zero. However, in the simulation, in some of the cases, the circles did not intersect, and an approximate estimate was used by creating fictitious points of intersection by increasing the received power. Although the approximation resulted in a non-zero value of the performance measure, the error was small compared to the size of the cell.

SOA-based PL estimation using the SOCLSFM gives worse performance compared with that of the PLBLSFM, as illustrated in Figure 3.7 and Table 3.2. However, propagation is subject to multipath as well as shadowing, as is commonly encountered in practice. Thus the SOCLSFM is a more realistic model.

The results of the evaluation have shown that the SOA-based PL estimation using the SOCLFSM by itself cannot yield good performance. The expected accuracy of the estimate using this method does not meet the E-911 PL requirements described in Chapter 1. There is a need to introduce a new approach to improve the performance of the SOA-based scheme.

3.5 Summary

The SOA method is a well-known technique in estimating the PL in a wireless channel. This chapter theoretically describes the SOA-based PL estimation scheme using the SOCLSFM. Mathematical formulation and the results of the evaluation are presented. For completeness, PL estimated performance based on the SOA using both the PLBLSFM and the SOCLSFM are evaluated. The performance of the SOA-based PL scheme using the PLBLSFM was shown to be superior as the PLBLSFM represents ideal conditions. However, in practice, the SOCLSFM is more realistic. Large mean errors in the PL estimate using the SOCLSFM are caused by the multipath and shadowing. PL estimation using only SOA-based scheme generally gives poor performance, as shown by the SOA-based PL scheme using the SOCLSFM. There is a room for improving the SOA-based method.

Chapter 4 Position Location for Strength-Of-Arrival and Neural-Network Techniques

4.1 Introduction

In this chapter, an accurate PL estimation scheme that integrates SOA and NN techniques is presented. The proposed PL estimation scheme is a hybrid of the traditional SOA scheme and an NN-based estimation scheme. The radio channel model uses the SOCLSFM. The NN is a multilayer feed-forward network, and uses the efficient Levenberg-Marquardt (LM) algorithm for training. The input to the NN is formed of

- PL estimates obtained using the SOA-based scheme with the SOCLSFM,
- three impulse responses estimated from the SOCLSFM, and
- features extracted from the impulse responses.

The output of the NN is a PL estimate integrating the input data.

An appropriate NN architecture was determined by essentially selecting different architectures and choosing the one whose performance was applicable to the problem. The number of hidden layers and the number of nodes in each hidden layer were varied, and the network configuration, which gave a minimum mean error in the PL estimation with respect to NN training phase, was selected. After the NN architecture was selected, two kinds of integrated schemes were evaluated by extensive experimentation using simulated training and test data, which contained various channel environments including multipath and shadowing.

The simulation results show an improved performance of the proposed integrated scheme, especially when the propagation channel is subject to multipath and shadowing.

4.2 Integrated Strength-Of-Arrival and Neural-Network PL Estimation

The motivation behind the integration of the SOA and NN schemes is to exploit the training abilities of the NN to overcome the deficiencies of the SOA scheme in handling different channel environments. This, hopefully, will improve the performance of the PL estimation scheme to a level acceptable to the FCC.

In this section, the NN architecture, the training and training of the NN, and the two integrated schemes are discussed.

4.2.1 Neural-Network Structure

General NN architectures are composed of the following [Haykin99]:

- Feed-forward connection
- Time delayed connection
- Lateral connection
- Feedback connection

An NN with a feed-forward structure and at least one hidden layer has the ability to approximate any nonlinear functional relationship, if the data is sufficiently large and representative. In other words, a multilayer feed-forward NN can approximate any arbitrary input-output function.

In view of this, a multilayer feed-forward NN is chosen for the PL estimation. The problem solved by the NN is to approximate a function that relates the inputs, namely features involved in the SOA estimation, and the outputs, namely the PL estimated by an integrated system. A multilayer feed-forward NN is shown in Figure 4.1.

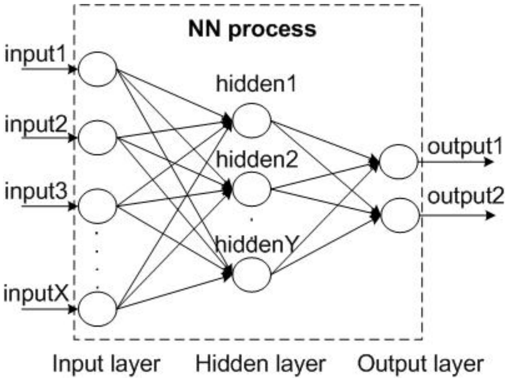


Figure 4.1 A fully connected multilayer feed-forward NN

In Figure 4.1, the neurons are grouped in layers. There are X input signals, Y hidden nodes, and two output signals. There may be many hidden layers between the input and output layers. In this figure only one hidden layer is shown.

4.2.2 Neural-Network Training Methods

The NN must undergo a training process before it can approximate a correct input-output function, that is to say, the NN can generate a correct output to a given input only after it has learnt [Mitchell97]. The algorithm that is designed to adjust the weights of the network connection in the training process is called the NN training algorithm.

4.2.2.1 Neural-network training algorithm — backpropagation

The NN training process can be simply classified as unsupervised or supervised. Unsupervised training is used when a clear link between input pattern and ideal output values does not exist [Warner96]. Supervised training involves providing an NN with specific input and ideal output values and allowing it to iteratively reach a solution [Mitchell97]. In this thesis, since the ideal outputs, namely the true PLs of the MSs, are available during the training phase, a supervised training scheme is employed. Compared with other training methods, such as Hebbian training, optimal estimation and competitive training, backpropagation is the most commonly used supervised algorithm for a multilayer feed-forward network. Details of the backpropagation training algorithm are provided in Appendix F.

4.2.2.2 Neural-network training algorithm — Levenberg-Marquardt

Since the backpropagation training algorithm was first popularized, there has been considerable research on methods to accelerate the convergence of this algorithm [Hagan94]. Many algorithms focus on standard numerical optimization, that is, using alternative methods for computing the weights associated with network connections. The most popular algorithms for this optimization are the conjugate gradient and Newton's methods. Newton's method is considered to be more efficient in the speed of convergence, but its storage and computational requirements go up as the square of the size of the network [Hagan94]. The LM algorithm is an approximation to Newton's method [Moré77]. The LM algorithm is efficient in terms of high speed of convergence and reduced memory requirements compared to the two previous methods. In general, with networks that contain up to several hundred weights, the LM algorithm has the fastest convergence [Hagan94]. This advantage is especially noticeable if accurate training is required, and hence is employed herein. The LM algorithm is presented in Appendix F.

4.2.3 The Integration of Strength-Of-Arrival and Neural-Network Techniques

The SOA-based PL estimation scheme, which is described in Chapter 3, is shown in Figure 4.2. Given an ideal fixed MS coordinate (x, y), three SOCLSFM propagation sets are used to obtain three impulse responses (h_1, h_2 and h_3). The impulse responses are used to obtain the PL estimate, denoted $(\hat{x}_{SOA}, \hat{y}_{SOA})$.

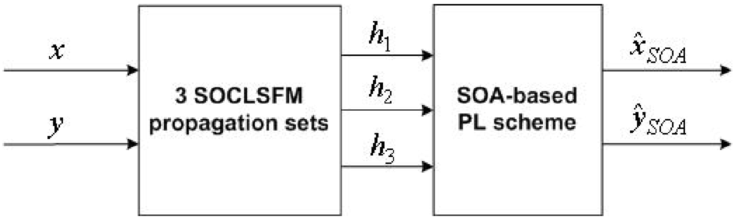


Figure 4.2 Block diagram of the SOA-based PL scheme

The integrated SOA and NN scheme is shown in Figure 4.3. Part of the integrated scheme is the SOA-based PL scheme. The impulse responses implicitly contain the features characterizing the channel. It was found that satisfactory performance could be achieved by extracting only a few features from the impulse responses. These features are the mean delay and power delay profile. Other features, such as RMS delay spread, are not considered explicitly in integrated estimation scheme. Compared to having no feature extraction, the mean delay input to the NN provides a significant improvement in the performance of the integrated scheme, as described further in Appendix G. The greater the number of inputs to the NN, the more complex is the NN and, as a result, the longer it takes to train the NN. The integration of the SOA and the NN techniques is obtained by applying to the NN, the PL estimates from the SOA-based scheme, the mean delays, and the three channel impulses responses.

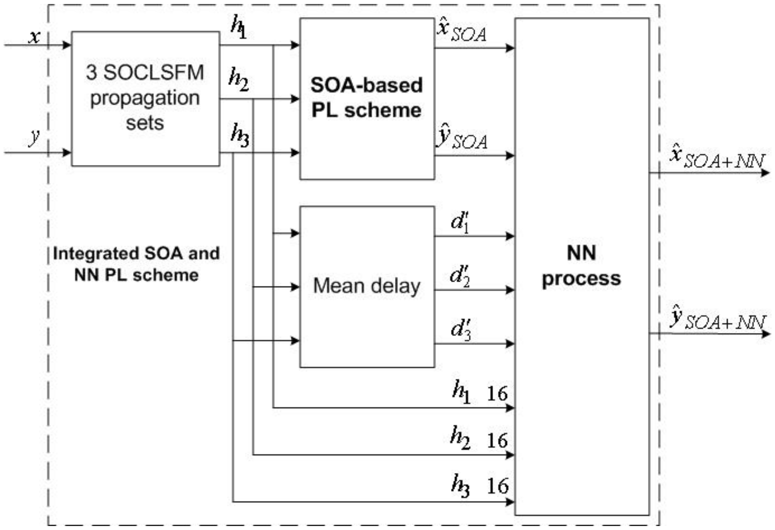


Figure 4.3 The first integrated scheme of SOA and NN techniques

As shown in Figure 4.3, called the first integrated architecture, the NN has 53 inputs:

- three 16-tap impulse response sets (*h_1*, *h_2* and *h_3*) from the SOA estimation inputs,
- three mean delays (d_1', d_2' and d_3') associated with the impulse responses, and
- the coordinate of the PL estimate using the SOA scheme , $(\hat{x}_{SOA}, \hat{y}_{SOA})$.

The NN outputs are the coordinates of the PL estimates of the integrated scheme, denoted$(\hat{x}_{SOA+NN}, \hat{y}_{SOA+NN})$. Later in this chapter, there are further references to Appendix G and Appendix H; these appendices contain results for this integrated architecture.

Figure 4.4 shows the second integrated scheme which was considered. There is a block which accepts three impulse responses, and puts out power delay profile. The details of this block are shown in Figure 4.5, where a'_1, a'_2, and a'_3 are the exponential coefficients of each power delay profile.

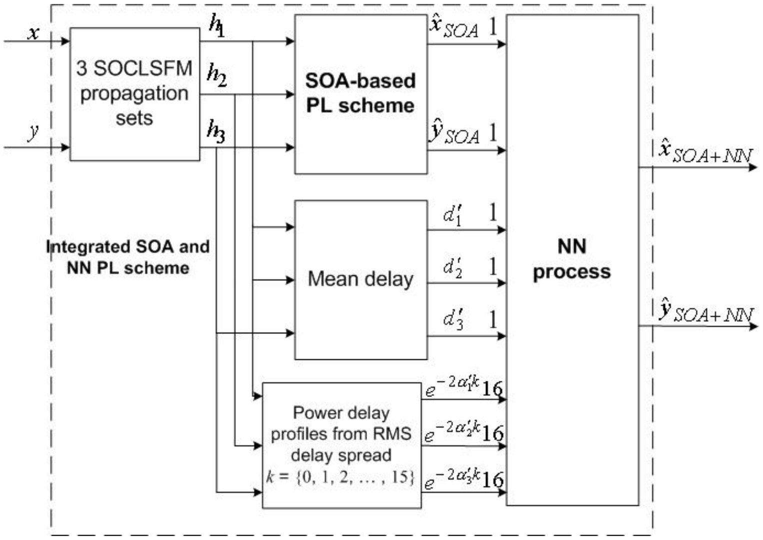


Figure 4.4 The second integrated scheme of SOA and NN techniques

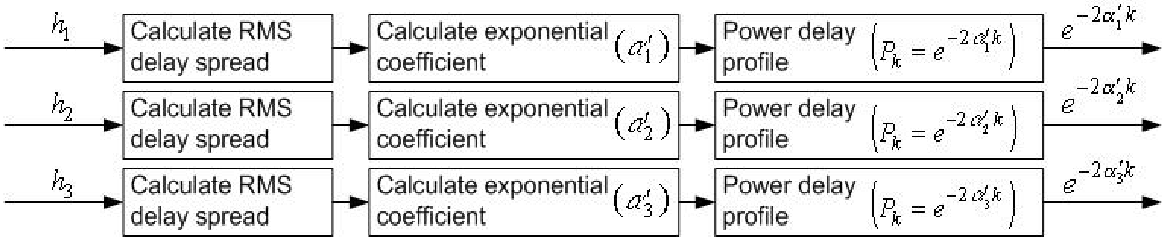

h_1
Calculate RMS delay spread
Calculate exponential coefficient (α'_1)
Power delay profile $\left(P_k = e^{-2\alpha'_1 k}\right)$
$e^{-2\alpha'_1 k}$
h_2
Calculate RMS delay spread
Calculate exponential coefficient (α'_2)
Power delay profile $\left(P_k = e^{-2\alpha'_2 k}\right)$
$e^{-2\alpha'_2 k}$
h_3
Calculate RMS delay spread
Calculate exponential coefficient (α'_3)
Power delay profile $\left(P_k = e^{-2\alpha'_3 k}\right)$
$e^{-2\alpha'_3 k}$

Figure 4.5 Using h_1, h_2 and h_3 to calculate power delay profile

The NN as shown in Figure 4.3 and Figure 4.4 is a multilayer feed-forward NN, which employs LM algorithm for training. Figure 4.6 shows the NN architecture. It has one input layer, one hidden layer, and one output layer. The hidden and output layers each have neurons. Each neuron is comprised of a set of weights, summers, and transfer functions. The weights are denoted by $\{ w_{ij}^{(p)} \}$. The outputs of the summers are passed to nonlinear transfer functions. A tangent sigmoid type nonlinear function is employed herein. The outputs of the hidden layer's nonlinear blocks are fed to the neurons in the output layer. There are two neurons in the output layer and the transfer function is linear.

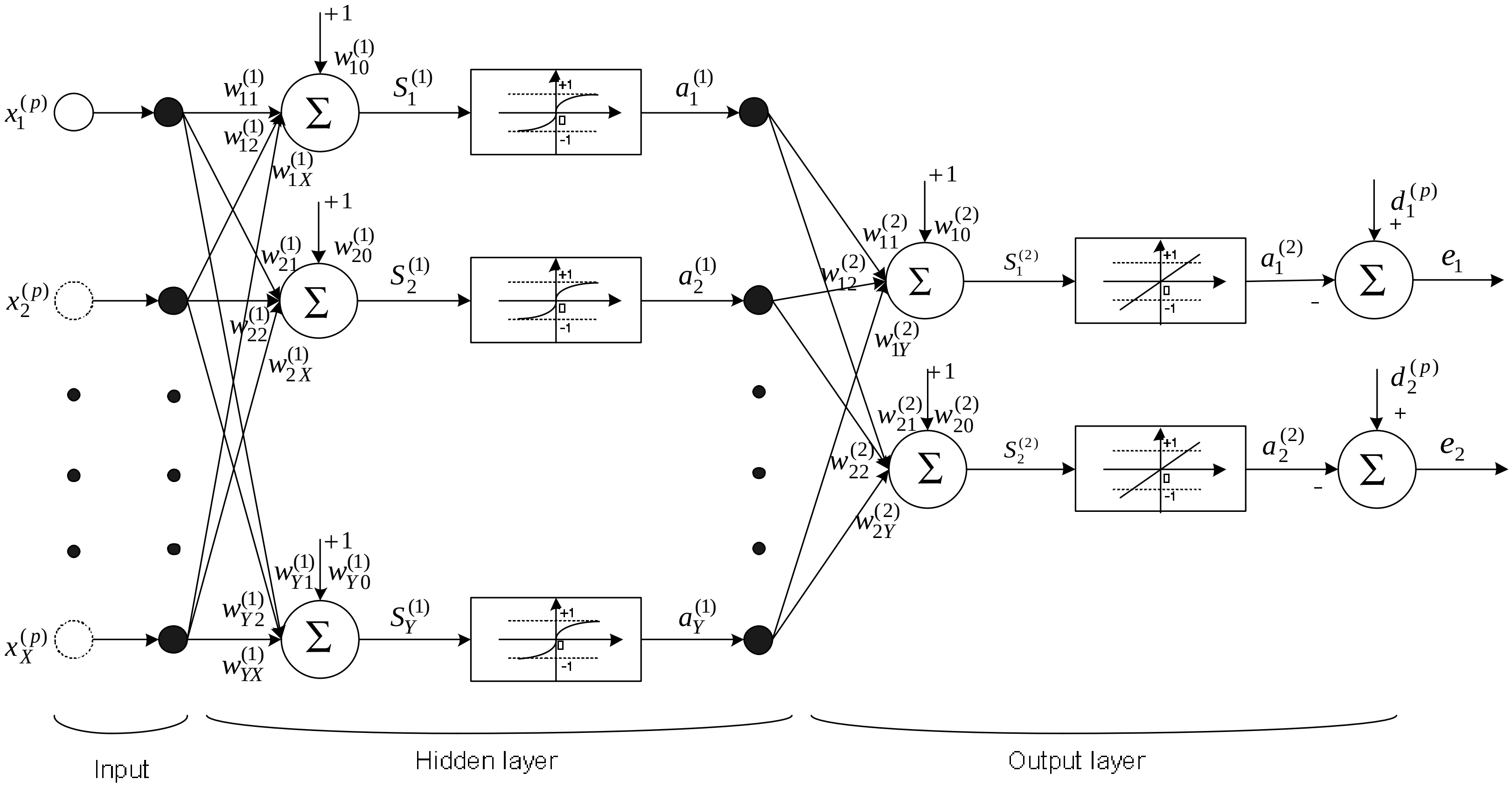


Figure 4.6 The multilayer feed-forward NN

The subscripts *p*, *i*, and *j* identify pattern, layer, and node, respectively. X denotes the number of the inputs, which in our case is 53. Y is the number of neurons in the hidden layer. The input pattern of the NN is denoted as $x_i^{(p)}$. The output of a summer and a transfer function block are denoted $S_i^{(p)}$ and $a_i^{(p)}$, respectively. The desired network output is expressed as $d_i^{(p)}$. The two errors at the output of the network are e_1 and e_2.

The backpropagation and LM algorithms are described in Appendix F.

The training of the NN proceeds as follows [Scalero92]:

- **Step1**: Initialize the weight vectors $w_{ij}^{(p)}$ randomly according to the Nguyen-Widrow algorithm [Nguyen90], described in Appendix G.
- **Step2**: Run a training pattern through the network.
- **Step3**: Evaluate the error signals and use them in the backpropagation algorithm.
- **Step4**: Update the weight vectors $w_{ij}^{(p)}$ using the LM algorithm.
- **Step5**: Repeats step 2 to 4 for all the input and output patterns in the training set.

Simulation results of the proposed integrated scheme are presented in Section 4.3.

4.3 Performance Evaluation

The size of the network depends on the number of layers and the number of neurons per layer. For obtaining the best performance, an optimal number of hidden layers and neurons per hidden layers must be determined. The NN training and the evaluation of the proposed scheme are given in Section 4.3.1, Appendix G, Section 4.3.2 and Section 4.3.3.

4.3.1 Determination of Optimal Size of the Neural-Network

The number of hidden layers, the number of neurons in each layer, the type of transfer function, the connections between layers, and the connection weights, determine the architecture of an NN. The architecture which yields the best performance depends on the type of problem handled by the NN. For example, a 21–15–3 network, with 21 inputs, 15 neurons in the hidden layer, and three outputs, is the best architecture for solving the Cholesterol data set problems [Demuth97].

The performance of the NN is evaluated using the mean error performance criteria described in the section on the SOA scheme. It is the average error distance between the estimated locations from integrated scheme and the true MS locations. A series of experiments were performed to determine the mean error for different NN sizes. As random initialization of the weights is employed, the average performance of the NN algorithms is evaluated by using several different sets of initial weights and biases. For each NN size, eight different experiments were performed with randomly chosen initial weights. Eight experiments were found to be sufficient to obtain an acceptable performance in terms of the mean variance of the mean error, as shown in Appendix G.

Table 4.1 The mean error – average over eight experiments for each NN architecture

The number of hidden layer(s)	**The architecture**	**Mean error (m)**
Zero	53–2	82.2
One	53–5–2	77.8
	53–10–2	75.6
	53–20–2	70.5
	53–25–2	72.3
	53–30–2	73.8
	53–50–2	78.7
	53–100–2	85.3
Two	53–20–15–2	236.46
	53–30–20–2	182.56

Table 4.1 provides the mean errors for the NNs with zero, one, and two hidden layers and a variable number of neurons in each layer. The NN is presented with identical input patterns for each of the chosen architectures. The input patterns to the NN were identical for each experiment associated with a given NN architecture. The number of input patterns presented was 1500 and its justification will be given in Section 4.3.2.

From Table 4.1, it is obvious that the mean errors are relatively small with one hidden layer. With one hidden layer, the error reaches a minimum value when there are 20 neurons in the hidden layer, also as shown in Figure 4.7. Additional results for 2, 5 and 10 hidden layers are given in Appendix G. It is for this reason that the number of hidden neurons, Y, is set to 20 and the evaluation of the proposed scheme is based on the 53–20–2 NN of the first integrated scheme shown in Figure 4.6 and Figure 4.3.

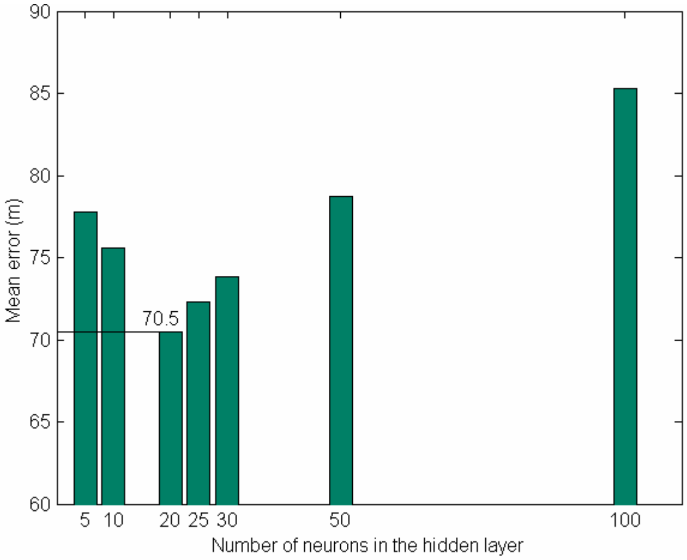


Figure 4.7 Mean error with the integrated scheme and one hidden layer

4.3.2 Neural-Network Training

The multilayer feed-forward NN with one hidden layer, 53 inputs and two outputs is trained by presenting sufficiently large and representative patterns. The patterns were split into training and test patterns. The NN was trained with training patterns, that is to say, the network weights were determined using training patterns. After the NN was trained, its performance was evaluated using the test patterns, which were not presented during the training phase. The details about the training patterns are given in Appendix H. The number of training patterns, T_{N}, was 1500. Having T_{N} greater than 1500 did not significantly improve the performance, while less than 1500 patterns degraded the performance.

The performance measures employed in the NN training are the same as those used in the SOA-based scheme described in Chapter 3. The performance measures include the mean error, denoted e, the percentage of errors less than 125 m, denoted $P\%$, and the outage of the PL estimation scheme, denoted P_{outage}. The performance measure is given in terms of an average over all the patterns rather than in terms of the estimation error for each individual pattern.

The training phases of the first and second integrated schemes are described in Sections 4.3.2.1 and 4.3.2.2, respectively.

4.3.2.1 The training phase of the first integrated scheme

The training patterns were applied to the first integrated scheme shown in Figure 4.3, and the results are shown in Table 4.2.

Table 4.2 The comparison of the training performance of the SOA-based scheme and the first integrated scheme

Performance measures of the SOA-based scheme	**Performance measures of the first integrated scheme**
$e = 175.9$ m	$e = 71.6$ m
$P\% = 36.0$ %	$P\% = 80.1$ %
$P_{outage} = 1.9\%$	

Also illustrated in Table 4.2 is the performance of the traditional SOA-based scheme. Table 4.2, clearly shows the superiority of the first integrated scheme.

4.3.2.2 The training phase of the second integrated scheme

The training patterns were applied to the second integrated scheme shown in Figure 4.4, and the results are shown in Table 4.3. Also, the performance of the traditional SOA-based scheme is presented.

Table 4.3 The comparison of the training performance of the SOA-based scheme and the second integrated scheme

Performance measures of the SOA-based scheme	**Performance measures of the second integrated scheme**
$e = 175.9$ m	$e = 66.1$ m
$P\% = 36.0\%$	$P\% = 85.5$ %
$P_{outage} = 1.9\%$	

Similarly, Table 4.3 shows the superiority of the second integrated scheme. Comparing Table 4.2 and Table 4.3, the PL performance is improved when the second integrated scheme is employed.

4.3.3 Position Location Accuracy of the Integrated Techniques

Once the NN's training phase is completed, the trained NN may be used to estimate the PL when input data not seen during the training is presented. During the testing phase, the NN is presented with T_M random sets of inputs, the true MS positions. T_M is much less than the number of training patterns T_N. Unlike the training phase, the testing phase does not involve known outputs, namely the true PLs of the MSs. However, during the testing phase, the NN is presented a reduced number of similar inputs, namely the PL estimates using the SOA, the impulse responses or power delay profiles, and the mean delays. The outputs are the estimated PLs of the MSs. The test size, T_M, is selected to be 50. This size is determined so that the variance of PL estimates is sufficiently small, as described in Appendix H.

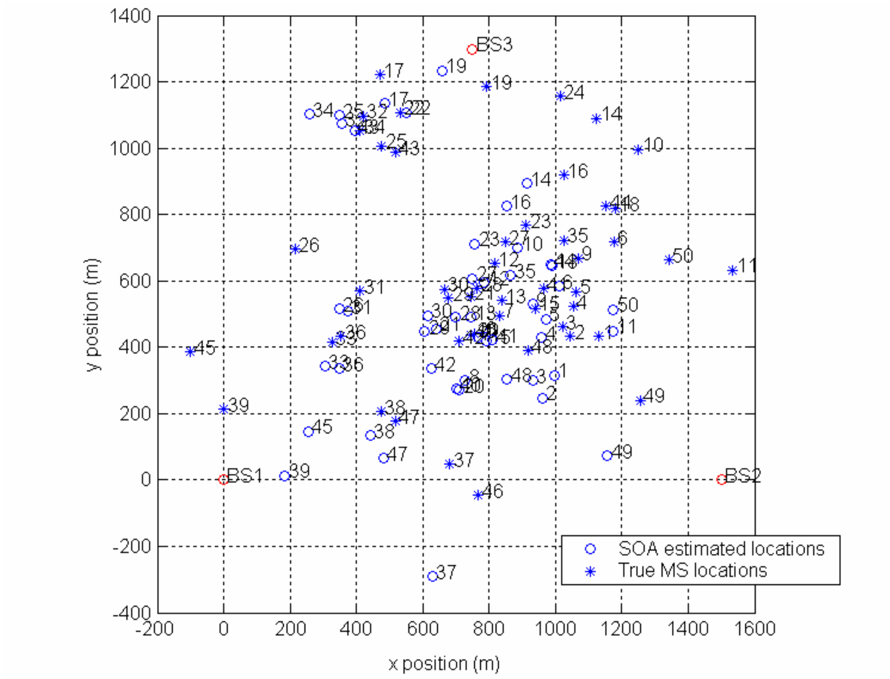


Figure 4.8 PL results of the SOA-based scheme
(50 MS users, Mean error: 173.6 m)

Sections 4.3.3.1 and 4.3.3.2 present the results obtained from the first and the second integrated schemes, respectively. The accuracy improvement is discussed in Section 4.3.3.3. For the SOA-based scheme, Figure 4.8 indicates the random locations of 50 MSs, the 50 estimated PLs, and three BSs. This figure is presented for evaluating the PL performance of the SOA-based scheme. The same MSs' locations, shown in Figure 4.8, are used to test the first and second integrated schemes.

4.3.3.1 The performance of the first integrated scheme

The performance of the first integrated scheme is evaluated by simulation. Figure 4.9 indicates the 50 random PLs of the MSs, and the corresponding estimated PLs using the first integrated scheme. For both the SOA-based scheme and the first integrated scheme, Table 4.4 provides a summary of the performance measures of 50 MS locations.

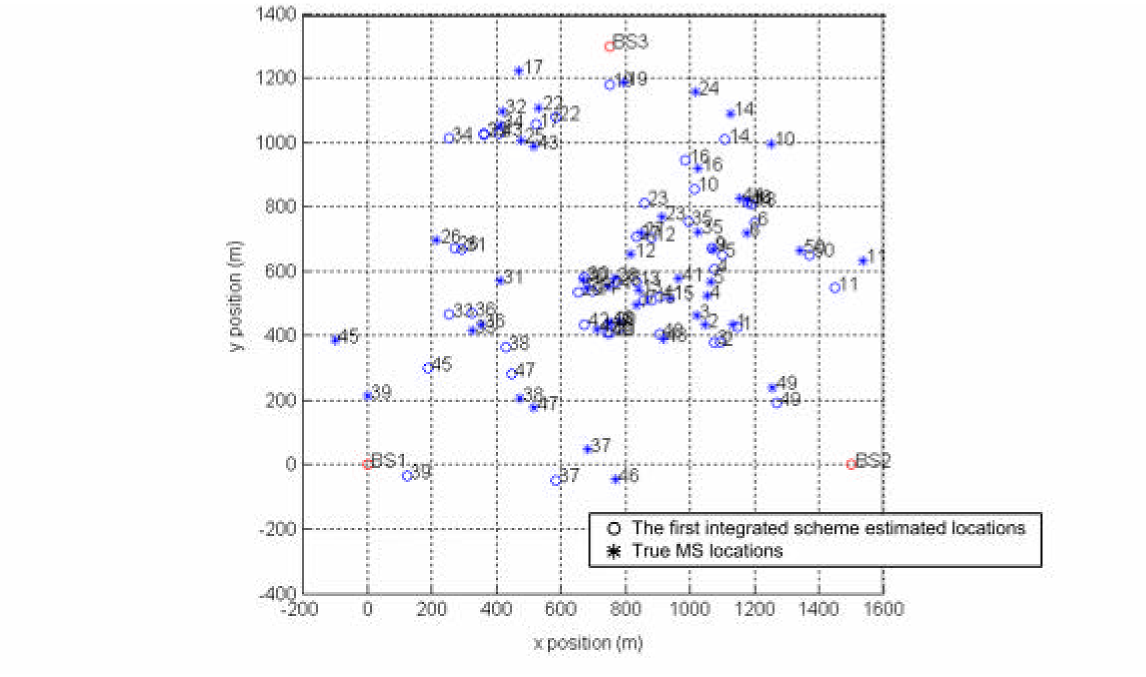


Figure 4.9 PL results of the first integrated scheme
(50 MS users, Mean error: 74.4 m)

Table 4.4 The performance comparison of the SOA-based scheme
and the first integrated scheme

SOA-based scheme	**First integrated scheme**
$e = 173.6$ m	$e = 74.4$ m
$P\% = 35.4$ %	$P\% = 81.2$ %
$P_{outage} = 4.0$ %	

4.3.3.2 The performance of the second integrated scheme

Similar to the previous simulation results, Figure 4.10 provides the true locations and the location estimates of the second integrated scheme. Table 4.5 shows the accuracy improvement of the second integrated scheme, compared to the SOA-based scheme.

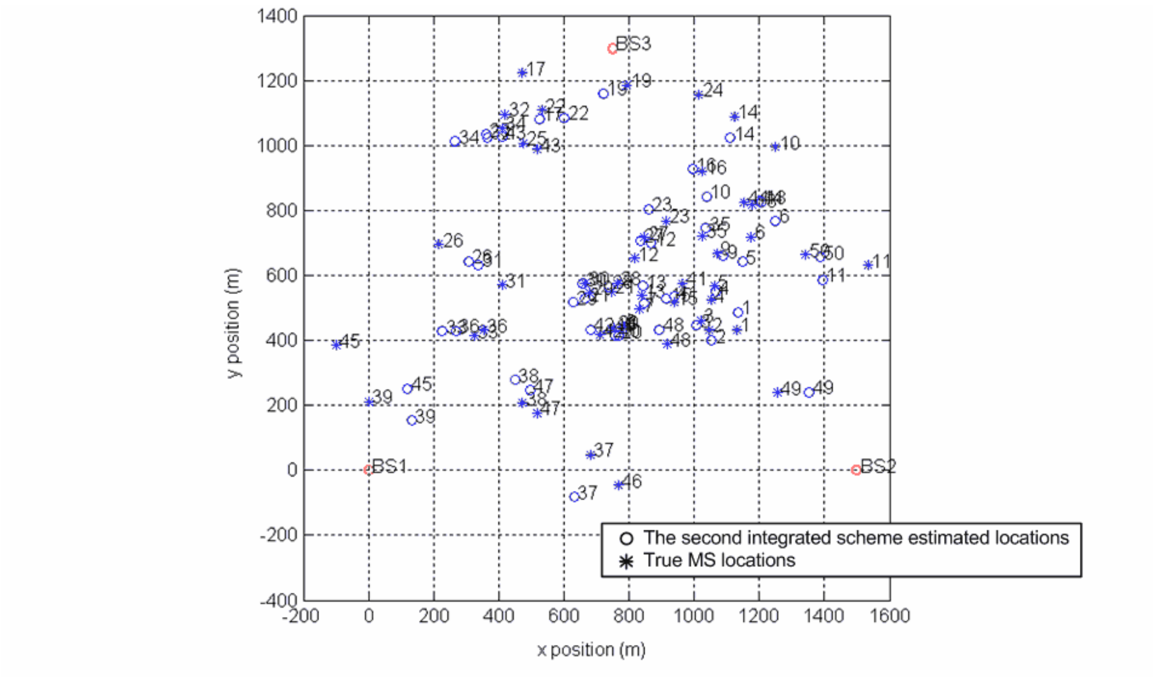


Figure 4.10 PL results of the second integrated scheme
(50 MS users, Mean error: 65.8 m)

Table 4.5 The performance comparison of the SOA-based scheme and
the second integrated schemes

SOA-based scheme	Second integrated scheme
$e = 173.6$ m	$e = 65.8$ m
$P\% = 35.4$ %	$P\% = 85.4$ %
$P_{outage} = 4.0$ %	

4.3.3.3 The accuracy improvement in the integrated techniques

Outage occurs in all the simulations. For a particular scheme, outage is the number of MSs which cannot be located, divided by the number of successful estimates. P_{outage} in Tables 4.4 and 4.5 is 4.0%, which implies that two of 50 MSs cannot be located. For these two MSs, both the SOA-based scheme and the integrated schemes fail to estimate their PLs. As indicated in Figures 4.8, 4.9 and 4.10, the locations MS_{24} and MS_{46}, could not be estimated.

From the results presented in Section 4.3.3.1, the first integrated scheme has superior performance compared to the SOA-based scheme. In Table 4.4, the first integrated scheme has a mean error, e , which is 57% less than that the SOA-based scheme. The first integrated scheme also has a 46% increase in the number of MSs within 125 m of the true MSs locations ($P\%$). This shows that the first integrated scheme, when implemented with a properly trained NN, meets the FCC requirements for an E-911 system. If we zoom in on individual MS locations, for most of the MSs such as MS_{37} in Figure 4.9, the estimates deviate slightly from the true PLs. As shown in Figure 4.8 and Figure 4.9, the deviation in the estimated location of MS_{37} is smaller for the first integrated scheme compared with the SOA-based scheme.

Similar to Section 4.3.3.1, the results of evaluation of the second integrated scheme are presented in Section 4.3.3.2. Compared to the SOA-based scheme, Table 4.5, indicates an improvement of 62% in e , and a significant improvement in $P\%$ of 50%. Figure 4.8 and Figure 4.10 also show this improvement. The PL estimates are very close to the actual in most cases.

The second integrated scheme has better performance than the first integrated scheme, as shown in Figures 4.9 and 4.10. Comparing the results presented in Tables 4.4 and 4.5, the performance improvements measured in terms of e and $P\%$ are 8.6 m and 4.2%, respectively. This shows conclusively that the second integrated scheme is superior to the first.

4.4 Summary

In this chapter two novel integrated PL estimation techniques based on combining the traditional SOA-based method and the NN technique are presented. The NN structure, the training strategy, and the network training algorithms are explained. The experimental procedure to determine the NN size is given. Simulation results of the NN training phase associated with the integrated schemes are given. The results of the training as well as the test phases are presented. The performances of the first and second integrated scheme are investigated. The results show that the power delay profiles are the preferred input to the NN, compared to the alternative, using impulse responses. Since RMS delay spread is part of the power delay profile, it is expected that including RMS delay spread as an input to the NN will not significantly improved the performance.

Based on the simulations using the SOCLSFM, the proposed integrated schemes show superior performance in a multipath environment and meet the FCC requirements.

Chapter 5 Conclusions

5.1 Conclusions

A novel scheme to estimate the location of a mobile station is proposed, as well as a new radio channel model, the SOCLSFM, to evaluate that scheme. The scheme integrates the traditional SOA-based estimation and an NN with a view to overcome the poor performance of the traditional scheme.

The first contribution of this thesis is the SOCLSFM. It is a relatively simple radio channel model which accounts for path loss, shadowing, an exponential power delay profile, propagation delay, and RMS delay spread. The most important feature about the model is that it includes SOC in RMS delay spread. The SOC introduce correlation among impulse response coefficients and thus the SOC provides valuable information for an NN. This model was the foundation for comparing the SOA-based PL estimation scheme with the integrated SOA and NN scheme.

The second contribution is the combined SOA and NN scheme. The proposed integrated scheme is shown to meet the FCC requirements based on extensive simulation. Besides the SOA-based estimates, the NN is presented with the channel characteristics including its impulse response and mean delay. Thanks to the ability of the NN to learn from examples, various signal propagation characteristics of the channel, especially those affecting the performance of the SOA, the proposed scheme overcomes the inaccuracies in the SOA estimates in the face of channel variations. Furthermore, the RMS delay spreads of the SOCLSFM provide additional information in improving the integrated PL scheme accuracy. The proposed scheme presents a new approach to wireless service providers using CDMA technology to meet the FCC E-911 regulation.

5.2 Future Work

The SOCLSFM used in this thesis puts in only increases in the splashes of RMS delay spread. However, some decreases could also be made.

Improving the channel models or using channel information that better represents the real-world mobile radio channel would provide a more realistic simulation.

Besides the multipath and shadowed environment features of the SOCLSFM, which appear to be major limiting factors in the performance of the SOA-based estimation, other factors that affect the inaccuracies need to be addressed. One such factor is the type of solution of the SOA equations.

Since the training of the NN directly affects the performance of the integrated scheme, it is important that an appropriate training algorithm be employed. Although the traditional LM training algorithm has proven to be effective in this thesis, it may be beneficial to investigate other algorithms, especially those proposed by Bogdan M. Wilamowski and Yixin Chen [Wilamowski98]. In a one hidden layer NN, those algorithms are more efficient and require less computation memory to converge compared to the LM algorithm.

Examples presented to the NN during the training and the testing phases are crucial to the performance of the estimation scheme. A technique to select a sufficient and representative data set should be the focus of further study. A smaller representative set may result in poor performance, while a large representative set would require a longer time to train.

References

[Bosch95]

E. H. Bosch, “The effects different neural network architectures have on the exploitation of hyperspectral data,” *Proceedings of the International Symposium on Spectral Sensing Research* (*ISSSR95*), vol. 1, (Melbourne, Australia), pp.21, Nov. 26 - Dec. 1, 1995.

[Bulusu00]

N. Bulusu, J. Heidemann, and D. Estrin, “GPS-less low cost outdoor localization for very small devices,” *IEEE Personal Communications Magazine*, vol. 7, no. 5, pp. 28-34. Oct. 2000.

[Caffery95]

J. J. Caffery and G. L. Stüber, “Radio location in urban CDMA microcells,” *Sixth IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications* (*PIMRC95*), pp. 858-862, Sept. 27, 1995.

[Caffery98]

J. J. Caffery and G. L. Stüber, “Overview of radiolocation in CDMA cellular systems,” George Institute of Technology, *IEEE Communications Magazine*, vol. 36, no. 4, pp. 38-45, Apr. 1998.

[Dadson75]

C. E. Dadson, J. Durkin and E. Martin, “Computer prediction of field strength in the planning of radio systems,” *IEEE Transactions on Vehicular Technology*, vol. VT-24, no. 1, pp. 1-7, Feb. 1975.

[Demuth97]

H. Demuth and M. Beale, *Neural Network Toolbox User's Guide*. The Math Works, Inc., Ver. 3.0, 1997.

[Drane98]

C. Drane, M. Macnaughtan, and C. Scott, “Positioning GSM telephones,” *IEEE Communications Magazine*, vol. 36, no. 4, pp. 46-54, Apr. 1998.

[FCC97]

Federal Communication Commission, “FCC Final Order CC Docket 94-102, Part 3,” December 23, 1997, downloaded Dec. 10, 2001, http://www.xypoint.com/we911/legisation/fccorder3.html.

[Figel69]

W. Figel, N. Shepherd, and W. Trammell, “Vehicle location by a signal attenuation method,” *IEEE Transactions on Vehicular Technology*, vol. VT-18, pp. 105-10, Nov. 1969.

[Gallant93]

S. I. Gallant, *Neural Network Learning and Expert Systems*. Cambridge, Massachusetts, USA: The MIT Press, 1993.

[Gans72]

M. J. Gans, “A power-spectral theory of propagation in the mobile-radio environment,” *IEEE Transactions on Vehicular Technology*, vol. VT-21, no. 1, pp. 27-38, Feb. 1972.

[Garg96]

V. K. Garg and J. E. Wilkes, *Wireless and Personal Communications Systems*. American Telephone and Telegraph Company Bell Laboratories Division, Upper Saddle River, USA: Prentice Hall PTR, 1996.

[Garg97]

V. K. Garg, K. Smolik, and J. E. Wilkes, *Applications of CDMA in Wireless/Personal Communications.* Upper Saddle River, NJ, USA: Prentice Hall PTR, 1997.

[Gilhousen91]

K. S. Gilhousen, I. M. Jacobs, R. Padovani, A. J. Viterbi, L. A. Weaver, Jr. and C. E. Wheatley III, “On the capacity of a cellular CDMA system,” *IEEE Transactions on Vehicular Technology*, vol. 40, no. 2, pp. 303-312, May 1991.

[Gurney96]

K. Gurney, *An Introduction to Neural Networks*. London, UK: UCL Press, June 12, 1996.

[Hagan94]

M. T. Hagan and M. Menhaj, “Training feedforward networks with the Marquardt algorithm,” *IEEE Transactions on Neural Networks*, vol. 5, no. 6, pp. 989-993, 1994.

[Hata80]

M. Hata and T. Nagatsu, “Mobile location using signal strength measurements in a cellular system,” *IEEE Transactions on Vehicular Technology*, vol. VT-29, no. 2, pp. 245-51, May 1980.

[Hata90]

M. Hata, "Empirical formula for propagation loss in land mobile radio services," *IEEE Transactions on Vehicular Technology*, vol. VT-29, no. 3, pp. 317-325, Aug. 1980.

[Haykin99]

S. Haykin, *Neural Networks – A Comprehensive Foundation*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2nd ed., 1999.

[Hellebrandt99]

M. Hellebrandt and R. Mathar, "Location tracking of mobiles in cellular radio networks," *IEEE Transactions on Vehicular Technology*, vol. 48, no. 5, pp. 1558-1562 Sept. 1999.

[Hsiung99]

S. Hsiung, E. Kao, and J. Matthews, "Multilayer feedforward network and the backpropagation algorithm," Tech. Rep., Generation 5 artificial intelligence repository, Aug. 1999.

[Iyanaga80]

S. Iyanaga and Y. Kawada, *Distribution of Typical Random Variables*, Cambridge, MA, USA: MIT Press, pp. 1483-1486, 1980.

[Knopp95]

R. Knopp and P. Humblet, "Information capacity and power control in single-cell multiuser communications," *Sixth IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications* (*PIMRC95*), pp. 331-335, Sept. 1995.

[Knuth81]

Donald E. Knuth, *The Art of Computer Programming*, vol. 2 / Seminumerical Algorithms, Reading, MA, USA: Addison –Wesley Publishing Company, 2nd ed., 1981.

[Koshima00]

H. Koshima and J. Hoshen, "Personal locator services emerge – location tracking techniques (E-911) especially in a CDMA environment," *IEEE Spectrum*, vol. 32, no. 2, pp. 41- 48, Feb. 2000.

[Lee97]

W. C. Y. Lee, *Mobile Communications Engineering: Theory and Applications*. Columbus, OH, USA: McGraw-Hill, 2nd ed., 1997.

[Liberti97]

J. Liberti and T. Rappaport, *Smart Antennas for Wireless Communications: IS-95 and Third Generation CDMA Applications*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1997.

[Linmartz96]

J. M. G. Linmartz's *Wireless Communication*, *The Interactive Multimedia CD-ROM*. Boston, USA: Baltzer Science Publishers, vol. 1, no. 1, 1996.

[McGuire94]

M. McGuire, K. N. Plataniotis, and A. N. Venetsanopoulos, "A comparison of radiolocation methods for mobile terminals by distance measurements," Tech. Rep., Edward S. Rogers Sr. Department of Electrical and Computer Engineering, University of Toronto, Canada, Jan. 1994.

[Medbo99]

J. Medbo, H. Hallenberg, and J.-E. Berg, "Propagation characteristics at 5 GHz in topical radio-LAN scenarios," *VTC 99: 49th Vehicular Technology Conference*, vol. 1, (Houston, TX, USA), pp. 185-189, 1999.

[Messier98a]

G. G. Messier, "IS-95 cellular mobile location techniques," Master's Thesis, Department of Electrical and Computer Engineering, University of Calgary, Alberta, Canada, Sept. 1998.

[Messier98b]

G. G. Messier, M. Fattouche, and B. R. Petersen, "Locating an IS-95 mobile using its signal," in Conf. Rec. *The Tenth International Conference on Wireless Communications* (*Wireless 98*), vol. 2, (Calgary, AB, Canada), pp. 562-574, July 6-8, 1998.

[Mitchell97]

T. M. Mitchell, *Machine Learning*. Boston, USA: McGraw-Hill, 1997.

[Mizusawa96]

G. A. Mizusawa, "Performance of hyperbolic position location techniques for code division multiple access," Master's thesis, Dept. of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Virginia, USA, Aug. 1996.

[Moré77]

J. J. Moré, "The Levenberg-Marquardt algorithm: Implementation and theory," *in Numerical Analysis*, Lecture Notes in Mathematics, (G. A. Watson) ed., New York, USA: Springer Verlag, vol. 630, pp. 105-116, 1977.

[Morimoto99]

A. Morimoto, M. Katayama, T. Yamazato, and A. Ograwa, “Adaptive array antennas of the base station of a multi processing gain CDMA system,” *IEICE Transactions on Fundamentals*, vol. E82-A, no. 12, Dec. 1999.

[Neskovic00]

A. Neskovic, N. Neskovic, and D. Paunovic, “Indoor electric field level prediction model based on the artificial neural networks,” *IEEE Communications Letters*, vol. 4, no. 6, pp. 190-192, June 2000.

[Nguyen90]

D. Nguyen and B. Widrow, “Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights,” *Proceedings of the International Joint Conference on Neural Networks* (*IJCNN*), vol. 3, (San Diego, CA, USA), pp. 21–26, July 1990.

[Okumura68]

T. Okumura, E. Ohmori, and K. Fukuda, “Field strength and its variability in VHF and UHF land mobile service,” *Review Electrical Communication Laboratory*, vol. 16, no. 9-10, pp. 825-873, Sept. 1968.

[Pahlavan98]

K. Pahlavan, P. Krishnamurthy, and J. Beneat, “Wideband radio propagation modeling for indoor geolocation applications,” *IEEE Communications Magazine*, vol. 35, no. 4, pp. 60-65, Apr. 1998.

[Prasad00]

R. Prasad, W. Mohr, and W. Konhäuser, *Third Generation Mobile Communication System.* London, UK: Artech House, 3rd ed., 2000.

[Rappaport96]

T. S. Rappaport, *Wireless Communication: Principles and Practice*. Upper Saddle River, NJ, USA: Prentice-Hall Inc., 1st ed., 1996.

[Rappaport98]

T. S. Rappaport and J. S. Tucker, “Last-mile wireless propagation modeling, measurement, & prediction,” Wireless Communication Alliance (WCA) Meeting, Hewlett Packard, Santa Clara, CA, USA, Nov. 17, 1998.

[Rice67]

P. L. Rice, A. G. Longley, K. A. Norton and A. P. Barsis, “Transmission loss predictions for tropospheric communication circuits,” *NBS Tech Note 101*, vol. 2, revised Jan. 1967.

[Scalero92]

R. S. Scalero and N. Tepedelenlioglu, "A fast new algorithm for training feedforward neural networks," *IEEE Transactions on Signal Processing*, vol. 40, no. 1, pp. 202-210, Jan. 1992.

[Schaubach92]

K. R. Schaubach, N. J. Davis, N. J. IV and T. S. Rappaport "A ray tracing method for predicting path loss and delay spread in microcellular environment," *in 42nd IEEE Vehicular Technology Conference*, (Denver, CO, USA), pp. 932-935, May 1992.

[Schmidt96]

A. Schmidt, "A modular neural network architecture with additional generalization abilities for high dimensional input vectors," Master's thesis, Dept. of Computing, Manchester Metropolitan University, Manchester, England, Sept. 1996.

[Sklar97]

B. Sklar and J. Maheshwar, "Rayleigh fading channels in mobile digital communication system," *IEEE Communications Magazine*, vol. 35, no. 9, pp. 90-109, July 1997.

[Soma99]

P. Soma, L. C. Ong, and Y. W. Chia, "Characterization of propagation channel in the presence of an imperfect semi-infinite half-plane using asymptotic path integral and ray tracing techniques," *In the proceedings of 29th European Microwave Conference - European Wireless 99*, vol. 1, (Munich, Germany), pp. 13-16. 1999.

[State of Montana00]

State of Montana Department of Administration, Information Services Division, "Montana wireless E9-1-1 trial: Final report," Tech. Rep., Helena, MT, USA, May 22, 2000.

[State of NJ97]

State of New Jersey, "Report on the New Jersey wireless location Enhanced 911 system trial January 22 to April 30 1997: The first 100 days," Tech. Rep., Dept. of Law and Public Safety, Div. of State Police, NJ, USA, June 16, 1997.

[Thimm97]

G. Thimm and E. Fiesler, "High order and multilayer perceptron initialization", *IEEE Transactions on Neural Networks*, vol. 8, no. 2 , pp. 349-359, Feb. 1997.

[Tholl95]

D. M. Tholl, “Characterization of the UHF indoor radio propagation channel,” Master’s Thesis, Department of Electrical and Computer Engineering, University of Calgary, Alberta, Canada, Sept. 1995.

[TIA99]

Telecommunications Industry Association, Washington, DC, USA, *Mobile Station-Base Station Compatibility Standard for Wideband Spread Spectrum Cellular Systems* (*ANSI/TIA/EIA-95-B-99*), Feb. 1, 1999.

[Viterbi95]

A. J. Viterbi, *Principles of Spread Spectrum Communication*. Massachusetts, USA: Addison-Wesley Publishing Company, June 1995.

[Warner96]

B. Warner and M. Misra, “Understanding neural networks as statistical tools,” *The American Statistician*, vol. 50, pp. 284-293, 1996.

[Wilamowski98]
B. M. Wilamowski and Y. Chen, “Efficient algorithm for training neural networks with one hidden layer,” *International Joint Conference on Neural Networks 1999* (IJCNN99), (Washington DC), pp. 1725-1728, July 10 – July 16, 1999.

[Winer91]

B. J. Winer, D. R. Brown, and K. M. Michels, *Statistical Principles in Experimental Design*. Columbus, OH, USA: McGraw-Hill, 1991.

[Zagami98]

J. M. Zagami, S. A. Parj, and K. D. Melillo, “Providing universal location services using a wireless E911 location network,” *IEEE Communications Magazine*, vol. 36, no. 4, pp. 66-71, Apr. 1998.

[Zahedi93]

F. Zahedi, *Intelligent Systems for Business: Expert Systems with Neural Networks*. Belmont, New York, USA: Wadsworth Publishing Company, 1993.

Appendix A: Limitations of SSF Model PL Solution

A.1 Introduction

Compared with the LSF model, the SSF model has much more rapid fluctuations of received signals over a short travel distance or period of time [Rappaport96]. These fluctuations are caused by constructive and destructive interference between two or more versions of the same signal.

SSF is also called Rayleigh fading, scintillation or fading. SSF changes due to multipath reflection caused by the superposition or cancellation of multipath propagation signals, the speed of the transmitter or receiver and the bandwidth of the transmitted signal. If the received signal has an LOS component, the fading envelope is given by a Rician PDF (Possibility Density Function).

The suitability of SSF model simulation is investigated in this section. The factors that influence SSF are the motion of the transmitter and receiver and multipath in the received signal. To simulate an SSF model, a set of impulse responses is generated around the BS in a grid pattern with 50 m between points. 50 m was chosen to avoid a large set of sample points inside the 2000 m radius area around BSs. Every component inside impulse responses are also called multipath components or taps, which represents the energy of every piece of multipath waves received from a same transmitted signal. A Rayleigh fading channel model is described by impulse responses, and the channel model takes into consideration both the motion of the transmitter or mobile and the multipath propagation in NLOS environment.

The radio signals transmitted by the controlling BS must be correlated to distinguish them with other transmitted signals or noise by other BSs or unknown instruments.

However, the signals simulated by the sampling impulse responses of SSF model are not correlated beyond a 50 m distance. Thus a location system using a multipath radio channel model needs an LSF model to have large-scale correlated multipath.

A.2 Small Scale Fading Model Simulation

A Rayleigh fading simulated example using the model of Clarke or Gans' model was developed based on the method described by Rappaport [Rappaport96]. This section presents the simulation results for the SSF model, and judges its suitability for PL. Figure A.1 presents the fading channel manifestation used in the simulation.

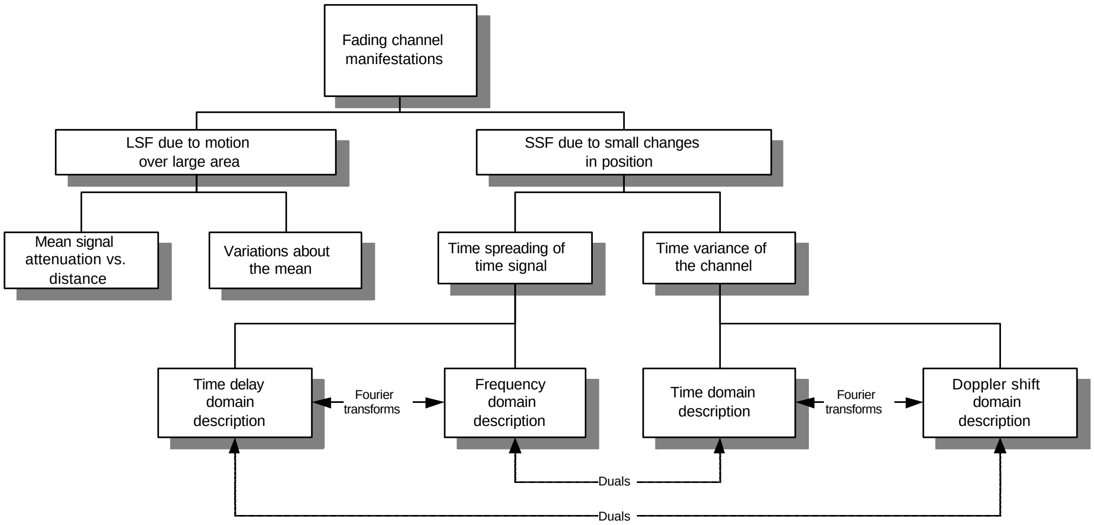

Fading channel manifestations
LSF due to motion over large area
SSF due to small changes in position
Mean signal attenuation vs. distance
Variations about the mean
Time spreading of time signal
Time variance of the channel
Time delay domain description
Fourier transforms
Frequency domain description
Time domain description
Fourier transforms
Doppler shift domain description
Duals
Duals

Figure A.1 Fading channel manifestation

A.2.1 Gans' model of the Doppler spectrum

Mobile communication systems experience Rayleigh fading and is modeled for a particular speed of the mobile by having the spectral shaping filter take the form of the Doppler filter with the maximum Doppler spread specified by the mobile speed.

Given a moving mobile, the signals from the BSs that are assumed to have the following properties:

- The multipath arrives from all directions with random phases.

- The received power is uniformly distributed with respect to the angle of arrival.

A baseband power spectral density, distribution of received power with respect to frequency, can be expressed as [Rappaport96]:

$$S_{bb}(f)=\begin{cases}\dfrac{P_r}{p\ f_m\sqrt{1-\left(\dfrac{f}{f_m}\right)^2}} & ,|f|<|f_m| \\ 0 & ,\text{otherwise}\end{cases} \tag{A.1}$$

where P_r is the mean received power by mobile, f_m is the maximum Doppler shift given by

$$f_m=\frac{v}{l}\ , \tag{A.2}$$

v is the velocity of mobile, and l is the wavelength of light.

Assuming P_r is 1 watt, and v is 50 km/hr, an illustration of Gans' model is presented in Figure A.2.

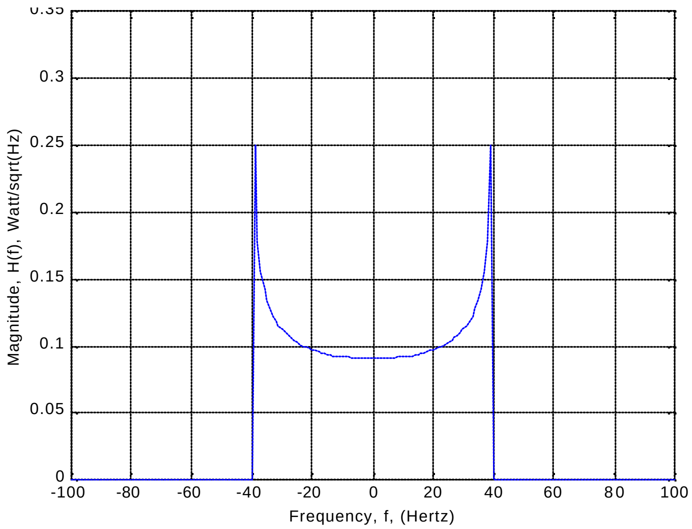


Figure A.2 A Gans' model

A.2.2 Samples of an impulse response

When a set of Rayleigh–faded baseband channel impulse responses is produced using the Gans' model, a single impulse response is generated at first. This is done in the continuous-time continuous-frequency domain, as indicated in Figure A.2, and conversion to the discrete-time continuous-frequency domain is required. Figure A.3(a) presents one discrete-time baseband channel impulse response in time domain and its Fourier transform in Figure A.3(b). Parameters for simulation are provided in Table A.1:

Table A.1 Parameters used in SSF simulation

Parameters	**Values**
The velocity of light (*c*)	2.998×10^{8} m/s
The velocity of the MS (*v*)	50 km/hr
The bandwidth of the radio channel 799 (f_c)	848.97 MHz
The received power assumed (P_r)	1 Watt
The length of sampled distance (*D*)	50 m
The radius of the circle range around BS (*R*)	2000 m

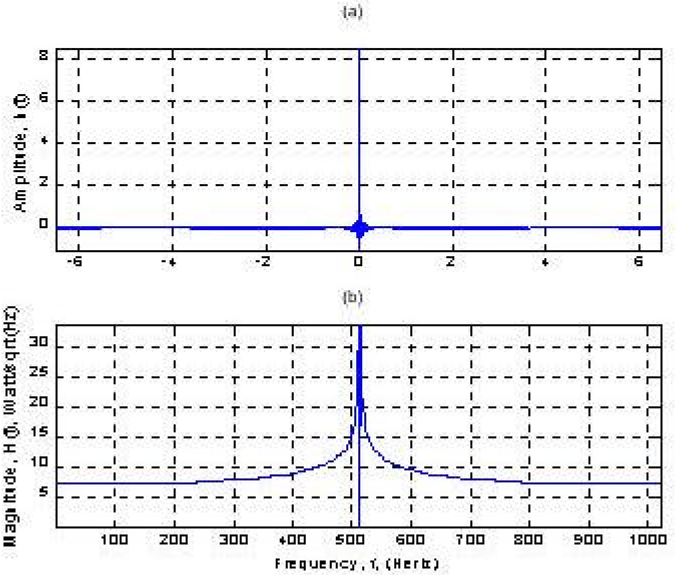


Figure A.3 Samples of an impulse response

(a) Impulse response in time domain (b) Impulse response in frequency domain

A.2.3 The outputs of one parallel line based on Gans' model

The model of the radio channel is presented in Figure A.4. Each component or tap of impulse responses can be determined by values obtained from 50 m grid scale sampling of Gans' model for each point.

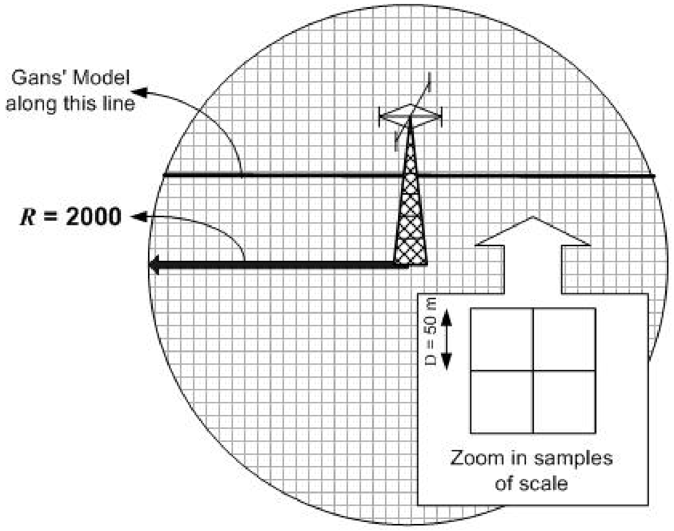


Figure A.4 Radio channel model around one BS

One Rayleigh-faded baseband channel impulse response of taps is generated by the procedure as shown in Figure A.5, where $h_I(t)$ and $h_Q(t)$ represent the in-phase and quadrature phase components of a complex baseband Gaussian process. The outputs of the system are Rayleigh-faded baseband channel impulse response taps which pass through the Doppler filter.

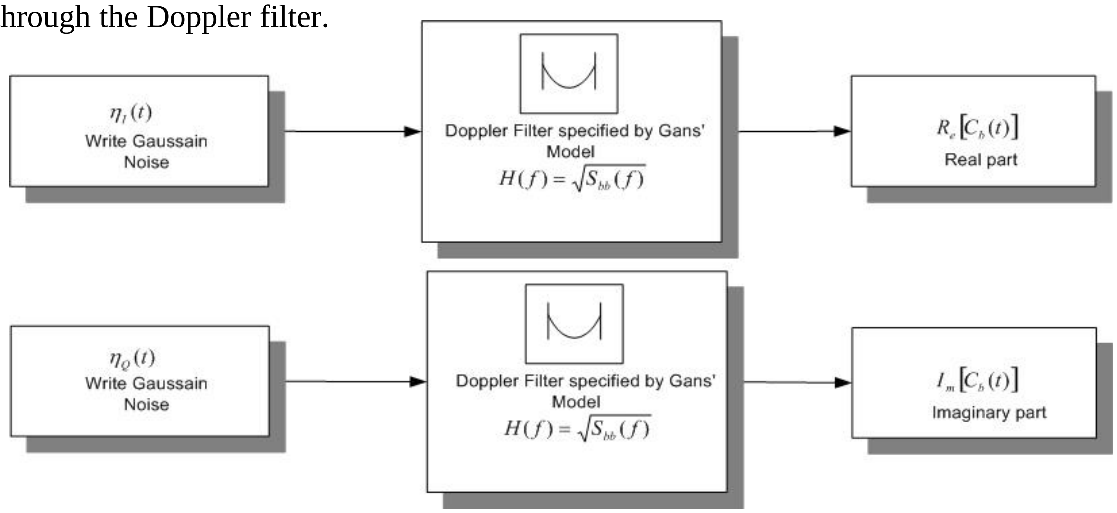


Figure A.5 The implementation steps of one impulse response

However, each impulse response generated by above procedure is in the continuous-time continuous-frequency domain, a discrete-time continuous-frequency model is required to be realized by computers, as the realization block diagram provided in Figure A.6

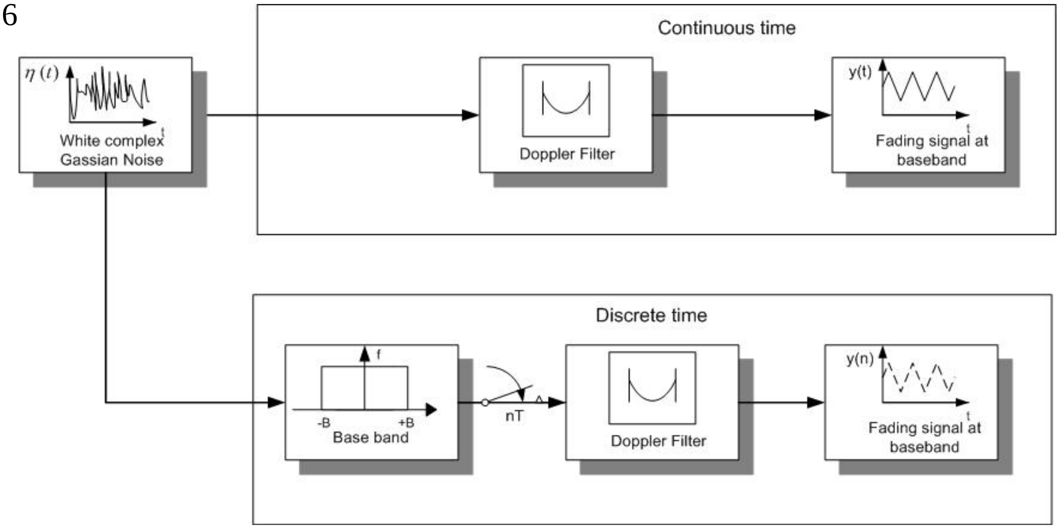


Figure A.6 Procedure to produce CTCF and DTCF fading signal through Gans' model

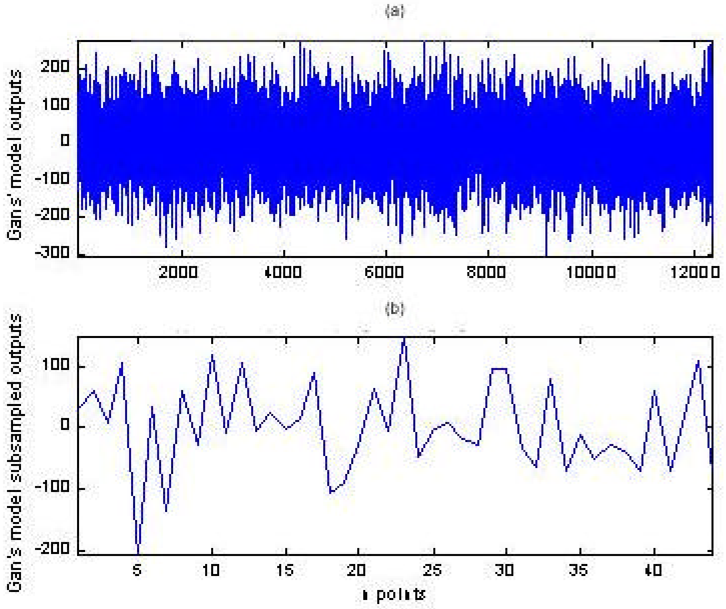


Figure A.7 One tap of Rayleigh fading signal at baseband

(a) Sampled signal (b) Subsampled signal

When a complex wide-sense-stationary Gaussian noise is sampled for a long sampling period, a sampled fading signal at baseband can be generated through low pass filter. The output discrete-time Rayleigh-faded signal at baseband and its both sampled and subsampled signals are shown in Figure A.7 (a) and Figure A.7 (b).

The Rayleigh-faded signal is generated by Gans' model. However, when the signal has been sampled over a longer period, there will be a problem of collecting too much sampling data inside a huge cellular area. In Section A.3, this problem will be discussed as the limitation of the SSF model in the PL estimation.

A.3 Small Scale Fading Model Limitation

A correlated-in-time signal is required for the multipath PL model implementation. However, if a long sampling distance (50 m) is chosen, or a long sampling time period is taken, the subsampled signal has to be correlated over the measurement. If the signal is not correlated, an NN-based PL model's implementation using an SSF model is not effective. Table A.2 indicates the correlated distance for Gans' model, as well as other parameters used in finding the distance.

Table A.2 Some factors in the SSF model sampled signal simulation

Parameters	**Value**
The sampling time period	12.7 ms
Distance over the sampling period	0.18 m
Number of samples over which "Gans'" signal is correlated*	10
Total distance over which "Gans'" signal is correlated	1.76 m

* The number of samples over which Gans' model is correlated: Once we get the sampling time period calculated (12.7 ms), arbitrarily go out 10 zero crossings of the Doppler spectrum's time response (sinc function). Assume the signal is weakly correlated after 10 zero crossings, the corresponding time is: $T_{unc} = 127$ ms, so the distance over Gans' model correlated is: $d_{unc} = vT_{unc}$=1.76 m.

The total distance over which Gans' signal is correlated is 1.76 m, but in our case a signal needs to be correlated over a much larger distance, for example, 50 m. Figure A.8 (a) and Figure A.8 (b) provide the autocorrelation for the two sampled and subsampled Rayleigh-faded signals in Figure A.7 (a) and Figure A.7 (b) respectively. Figure A.8 (b) shows a case that Gans' signal is not correlated until more than 127 ms.

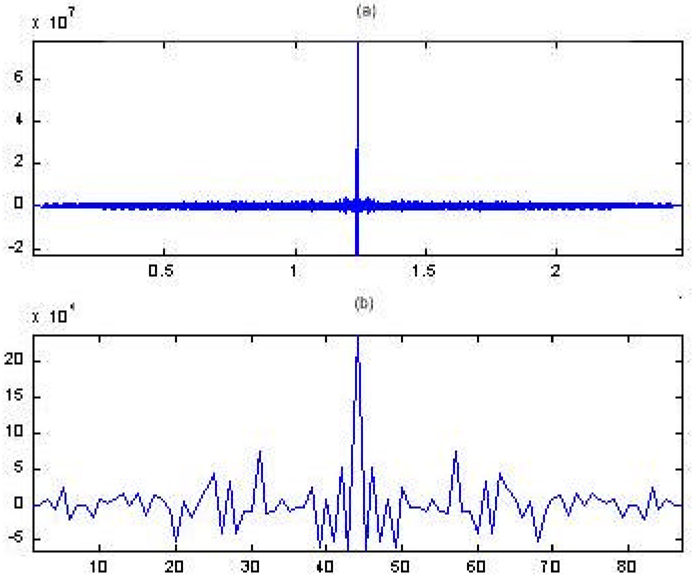


Figure A.8 Autocorrelation of Rayleigh fading model

(a) Sampled signal (b) Subsampled signal

The SSF model's signals are not correlated over a large distance. Therefore, a NN location system using multipath needs an LSF model.

Appendix B: Find Exponential Coefficient in Power Delay Profile

The first moment of the power delay profile (mean excess delay) is defined as:

$$\bar{t} = \frac{\sum_{k=0}^{\infty} kP_k}{\sum_{k=0}^{\infty} P_k} . \qquad \text{(B.1)}$$

The second moment of power delay profile is given by:

$$\overline{t^2} = \frac{\sum_{k=0}^{\infty} k^2 P_k}{\sum_{k=0}^{\infty} P_k} . \qquad \text{(B.2)}$$

The RMS delay spread is calculated using:

$$s_t = \sqrt{\overline{t^2} - \left(\bar{t}\right)^2} . \qquad \text{(B.3)}$$

The exponential power delay profile P_k is given:

$$P_k = \begin{cases} e^{-2ak} & , k \in \{0, 1, 2, 3, ...\} \\ 0 & , k \in \{..., -3. -2, -1\} \end{cases} . \qquad \text{(B.4)}$$

To calculate the relationship a with given RMS delay spread, substitute Equation (B.4) into $\bar{t}$ in Equation (B.1), $\bar{t}$ is derived as:

$$\bar{t} = \frac{\sum_{k=0}^{\infty} kP_k}{\sum_{k=0}^{\infty} P_k} = \frac{\sum_{k=0}^{\infty} ke^{-2ak}}{\sum_{k=0}^{\infty} e^{-2ak}} = \frac{\sum_{k=0}^{\infty} ke^{-2ak}}{\frac{1}{1-e^{-2ak}}} = \sum_{k=0}^{\infty} ke^{-2ak}(1-e^{-2a}) , \qquad \text{(B.5)}$$

where

$$S_1 = \sum_{k=0}^{\infty} ke^{-2ak} = \sum_{n=0}^{\infty} \left(nu(n)\right)\left(e^{2a}\right)^{-n} , \text{ representing } z = e^{2a} .$$

Thus

$$S_1(z) = \sum_{n=0}^{\infty} (nu(n)) z^{-n}.$$

By z transform,

$$n(u(n)) \overset{z}{\longleftrightarrow} \frac{z^{-1}}{(1-z^{-1})^2},$$

S_1 is finally derived as

$$S_1(z) = \frac{z^{-1}}{(1-z^{-1})^2}, \text{where } z = e^{2a} \Rightarrow S_1 = \frac{e^{-2a}}{(1-e^{-2a})^2}. \qquad \text{(B.6)}$$

Combine Equation (B.6) and Equation (B.5), $\bar{t}$ can be given as the following:

$$\bar{t} = \frac{e^{-2a}}{(1-e^{-2a})^2}(1-e^{-2a}) = \frac{e^{-2a}}{1-e^{-2a}}, \text{ where } a \geq 0, \qquad \text{(B.7)}$$

Thus

$$(\bar{t})^2 = \left(\frac{e^{-2a}}{1-e^{-2a}}\right)^2 = \left(\frac{z^{-1}}{1-z^{-1}}\right)^2 = \frac{1}{(1-z)^2}, \text{where } a \geq 0. \qquad \text{(B.8)}$$

By substituting Equation (B.4) into $\overline{t^2}$ in Equation (B.2), $\overline{t^2}$ is derived as:

$$\overline{t^2} = \frac{\sum_{k=0}^{\infty} k^2 P_k}{\sum_{k=0}^{\infty} P_k} = \frac{\sum_{k=0}^{\infty} k^2 e^{-2ak}}{\sum_{k=0}^{\infty} e^{-2ak}} = \sum_{k=0}^{\infty} k^2 e^{-2ak}(1-e^{-2a}), \text{ where } a \geq 0 \qquad \text{(B.9)}$$

where $S_2 = \sum_{k=0}^{\infty} k^2 e^{-2a\,k} = \sum_{n=0}^{\infty} (n^2 u(n))(e^{2a})^{-n}$, where assign $z = e^{2a}$.

Thus

$$S_2(z) = \sum_{n=0}^{\infty} (n^2 u(n)) z^{-n}.$$

By z transform,

$$x(n) = n\big(u(n)\big) \xleftrightarrow{Z} \frac{z^{-1}}{\left(1 - z^{-1}\right)^2} \,,$$

S_2(z) is expressed as

$$S_2(z) = \sum_{n=0}^{\infty} \big(nx(n)\big) z^{-n} \,.$$

With z transform

$$nx(n) \xleftrightarrow{Z} -z \frac{dx(z)}{dz} \,,$$

S_2(z) can be rewritten as

$$S_2(z) = -z \frac{dx(z)}{dz} = -z \frac{d \dfrac{z^{-1}}{\left(1 - z^{-1}\right)^2}}{dz} \text{, where } z = e^{2a} \,. \qquad \text{(B.10)}$$

By substituting Equation (B.10) and $z = e^{2a}$ to Equation (B.9), $\overline{t^2}$ is in the form:

$$\overline{t^2} = -z \frac{d \dfrac{z^{-1}}{\left(1 - z^{-1}\right)^2}}{dz} (1 - z^{-1}) = (1 - z) \frac{d \dfrac{z^{-1}}{\left(1 - z^{-1}\right)^2}}{dz} \,. \qquad \text{(B.11)}$$

RMS delay spread can be derived by substituting Equation (B.11) and Equation (B.8) into Equation (B.3):

$$s_t = \sqrt{\overline{t^2} - \left(\bar{t}\right)^2} = \sqrt{(1 - z) \frac{d \dfrac{z^{-1}}{\left(1 - z^{-1}\right)^2}}{dz} - \frac{1}{\left(1 - z\right)^2}} \,. \qquad \text{(B.12)}$$

Assuming

$$S_3(z) = \frac{d \dfrac{z^{-1}}{\left(1 - z^{-1}\right)^2}}{dz} = \frac{d\left(\dfrac{z}{(1 - z)^2}\right)}{dz} = \frac{z'(1 - z)^2 - z\left((1 - z)^2\right)'}{\left(\left(1 - z^2\right)\right)^2} = \cdots = \frac{1 + z}{(1 - z)^3} \text{, (B.13)}$$

and combining Equation (B.12) and Equation (B.13), RMS delay spread is obtained:

$$s_t = \sqrt{(1-z)\left(\frac{1+z}{(1-z)^3}\right) - \frac{1}{(1-z)^2}} = \sqrt{\frac{z}{(1-z)^2}} = \frac{\sqrt{z}}{1-z} \xleftrightarrow{z=e^{2a}} \frac{e^a}{1-e^{2a}} \text{. (B.14)}$$

$s_t = \dfrac{e^a}{1-e^{2a}} \Rightarrow s_t^{\,2} = \dfrac{e^{2a}}{\left(1-e^{2a}\right)^2}$, representing $z = e^{2a}$, $s_t^{\,2}$ can be rearranged as

$$s_t^{\,2} = \frac{z}{(1-z)^2} \text{ .} \qquad \text{(B.15)}$$

Rearrange Equation (B.15) as $z^2 - (2 + \dfrac{1}{s_t^{\,2}})z + 1 = 0$, and assume $b = (2 + \dfrac{1}{s_t^{\,2}})$, z is derived as:

$$z = \frac{b \pm \sqrt{b^2 - 4}}{2} \text{, where } z = e^{2a} \text{ .} \qquad \text{(B.16)}$$

Finally,

$$e^{2a} = \frac{b \pm \sqrt{b^2 - 4}}{2} \Rightarrow$$

$$\boxed{a_{1,2} = \frac{\ln\left(\dfrac{b \pm \sqrt{b^2-4}}{2}\right)}{2} \text{, where } b = (2 + \frac{1}{s_t^{\,2}}) \text{ .}} \qquad \text{(B.17)}$$

The final exponential coefficient with respect to RMS delay function is achieved as Equation (B.17).

Appendix C: Three Block Diagrams of Generating Impulse Responses for SOA-based PL Scheme

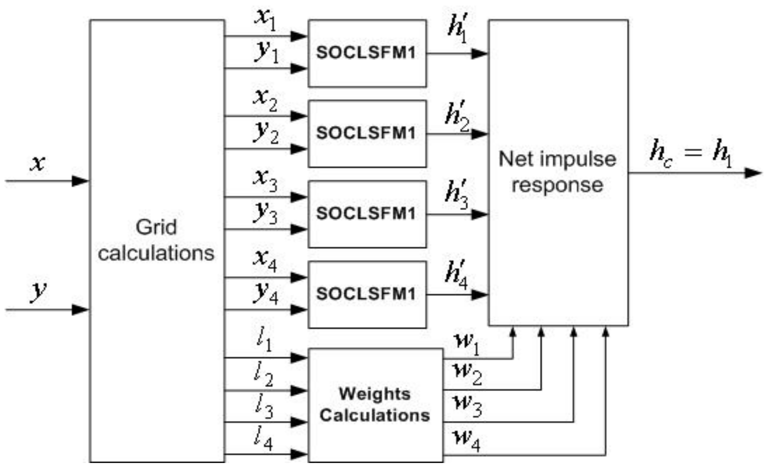


Figure C.1 The block diagram of generating impulse response h_1 for the SOA-based PL scheme

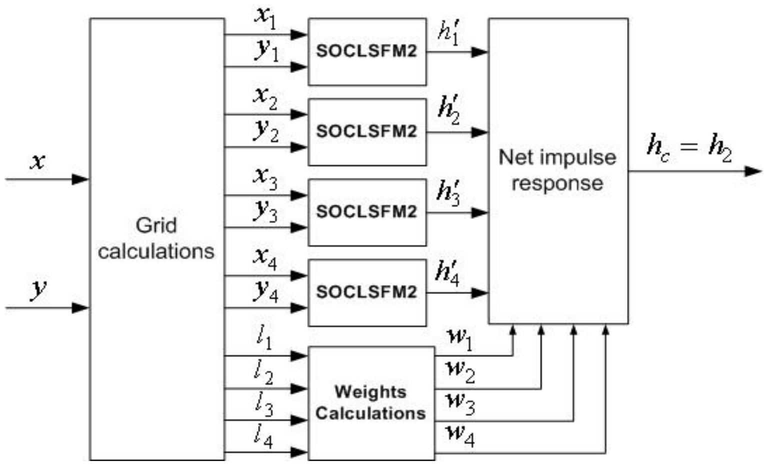


Figure C.2 The block diagram of generating imp ulse response h_2 for the SOA-based PL scheme

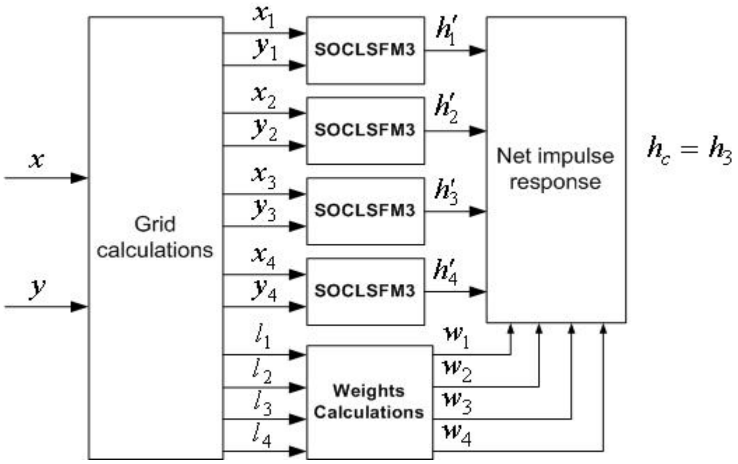


Figure C.3 The block diagram of generating impulse response h_3 for the SOA-based PL scheme

Appendix D: Detailed Explanations of the Proximate Problems in SOA Simulation

D.1 How circles intersected may result in proximate inaccuracies

How the circles intersect is important to find the MS's location. There is an interesting phenomenon shown in Figure D.1.

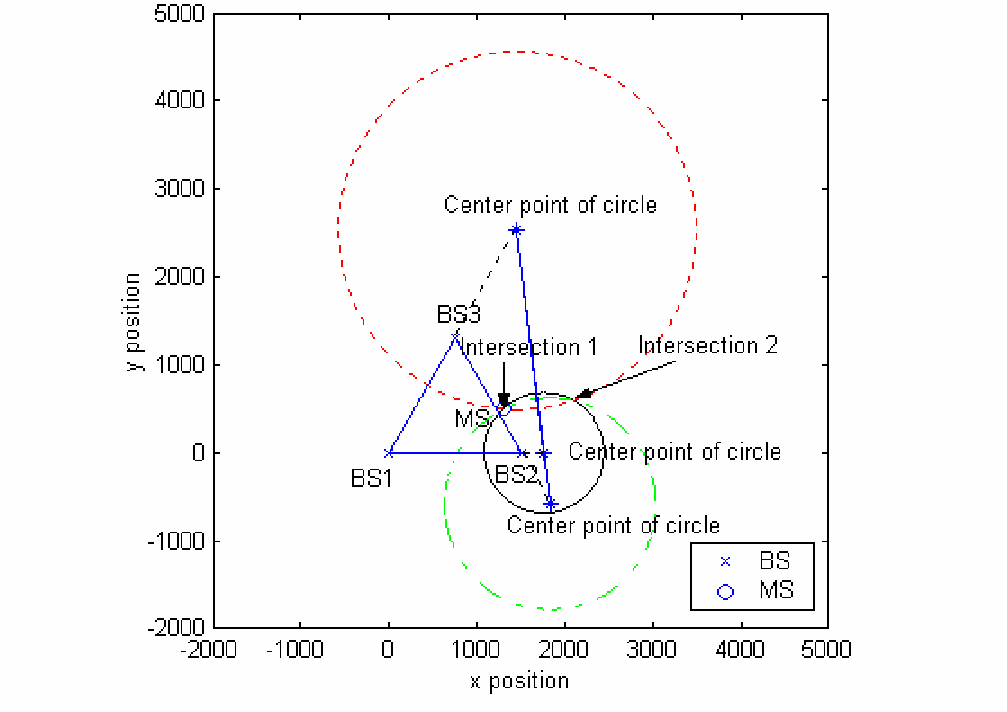


Figure D.1 SOA-based scheme estimated by three circles where MS's coordinate is (1300, 500)

As shown in Figure D.1, the centers of the circles happen to lie in the same line, the extended sides of the BSs' triangle, and the three circles have two intersections. The estimated MS position must locate inside one of the two intersections. In the case that the circles do not intersect, which usually occurs in SOA evaluated on the SOCLSFM, a proximate method might be used to approach the actual MS location. With this method one of the received powers for each BS is manually increased by small increments until finally the circles intersect. Larger received power at MS results in larger radius of the circle, which is demonstrated as following.

Three range difference equations are defined in Chapter 3:

$$\left(\frac{P_1}{P_2}\right)^{\frac{2}{n}} = \left(\frac{d_2}{d_1}\right)^2 \Rightarrow d_2^2 = \left(\frac{P_1}{P_2}\right)^{\frac{2}{n}} d_1^2\,, \tag{D.1a}$$

$$\left(\frac{P_1}{P_3}\right)^{\frac{2}{n}} = \left(\frac{d_3}{d_1}\right)^2 \Rightarrow d_3^2 = \left(\frac{P_1}{P_3}\right)^{\frac{2}{n}} d_1^2\,, \text{ and} \tag{D.1b}$$

$$\left(\frac{P_2}{P_3}\right)^{\frac{2}{n}} = \left(\frac{d_3}{d_2}\right)^2 \Rightarrow d_3^2 = \left(\frac{P_2}{P_3}\right)^{\frac{2}{n}} d_2^2\,. \tag{D.1c}$$

Since $P_1, P_2,$ and P_3 are the known variables in Equation (D.1), denoting

$$\left(\frac{P_1}{P_2}\right)^{\frac{2}{n}} = a_1\ , \tag{D.2a}$$

$$\left(\frac{P_1}{P_3}\right)^{\frac{2}{n}} = a_2\ , \text{ and} \tag{D.2b}$$

$$\left(\frac{P_2}{P_3}\right)^{\frac{2}{n}} = a_3\,. \tag{D.2c}$$

Through the derivation in Section 3.3.2, Chapter 3, the radiuses of three circles are obtained,

$$r_1 = \left(\frac{a_1 d}{1-a_1^2}\right), \tag{D.3a}$$

$$r_2 = \left(\frac{a_2 d}{1-a_2^2}\right), \text{ and} \tag{D.3b}$$

$$r_3 = \left(\frac{a_3 d}{1-a_3^2}\right). \tag{D.3c}$$

In Equation (D.3a), if a_1 is increased, obviously r_1 will be enlarged. Similarly, the larger a_2 or a_3 in Equation (D.3b) and (D.3c), the larger r_2 or r_3 respectively. As indicated in Equation (D.2a) and (D.2b), a_1 and a_2 can be enlarged by only increasing the received power P_1 at BS$_1$, and similarly a_3 can be enlarged by increasing received power P_2. Therefore, if the circles 1 and 2 do not intersect, we can either increase r_1 or r_2, resulting in manual increments in the received power P_1; If circles 1 and 3 do not intersect, increasing either r_1 or r_3 is necessary, which leads to increment in P_1 or P_2. Similarly, increment in either P_1 or P_2 leads to larger r_2 or r_3 if circle 2 and 3 do not intersect.

D.2 Equally received powers result in proximate inaccuracies

For the Equations (D.2a), (D.2b) and (D.2c), another problem possibly exists. If any two of P_1, P_2 and P_3 are equal, such as P_1=P_2, P_1=P_3, or P_2=P_3, which results in $a_1 = 1$, $a_2 = 1$ or $a_3 = 1$ respectively. For Equations (D.3a), if $a_1 = 1$, r_1 will be infinite, as well as r_2 and r_3 in Equation (D.3b) and (D.3c). If this happens, we simply assume that r_1, r_2 or r_3 has a very large value accordingly.

In general, the inaccuracies in SOA-based algorithms are caused by the above two problems in Section D.1 and D.2. These inaccuracies do not primarily affect the estimated PL for SOA evaluating on SOCLSFM, for they are usually within a distance of a few meters, as illustrated in Section 3.4. However, log-normal shadowing and SOC features in the SOCLSFM leads to much bigger PL estimation errors, usually more than a hundred meters.

Appendix E: Determination of Number of MSs in SOA-Based Scheme

Among three SOA performance criteria, mean error is the average performance that measures the mean distance between estimated MS locations and true MS PLs, and it is the major criteria that indicates the average distance error over an unknown number of MSs. The value of this number is commonly determined by the minimum size of a mean errors group with a relative stable variance compared to other groups of larger size. Variance is a measure of dispersion of a set of data points around their mean value, the mathematical expectation of the average squared deviations from the mean [Winer91]. On the other hand, it measures the "variability" (volatility) from an average, which can be analyzed to obtain the best chosen number of the MSs. If the statistical significance of the differences of means can be assessed, a more accurate comparison can be made between different mean errors groups. Statistical differences are assessed through an analysis of variance. The relationship between measurements of the variance of each group provides the information needed in determining whether the difference between groups is significant. If not, the size of corresponding group is selected to be the number of the MSs chosen in SOA-based scheme.

In the simulation, the sizes of mean errors groups are assumed to be {2, 5, 10, 50, 100}, and these groups are randomly chosen within the 100 SOA mean errors provided in Table E.1. For instance, for the group with size of 2, the group is {200.2, 46.1}, and for that of size 5, {21.7, 113.7, 46.1, 159.2, 91.2}. By calculating each group's corresponding variance, the figures of variances with respect to the groups' size are illustrated in Figure E.1, E.2, E.3 and E.4, in which different random mean errors are chosen in all groups for every attempt.

Table E.1 100 mean errors (m) of SOA-based scheme on the SOCLSFM

200.2	127.9	46.1	63.6	91.2	440.2	160.3	27.9	176.2	220.8
202.3	226.8	296.1	121.8	83.8	176.8	338.0	203.4	83.5	107.6
160.5	403.0	104.2	93.3	157.3	199.4	236.9	153.5	83.2	119.1
201.4	128.8	193.6	21.7	70.7	177.7	52.8	83.6	23.2	320.0
136.8	121.4	131.5	394.3	134.4	128.1	221.4	171.5	98.9	129.8
221.8	365.3	113.0	276.8	171.9	188.0	104.3	155.0	195.5	143.1
546.3	271.5	87.8	113.7	156.8	49.6	148.8	497.8	74.4	148.1
122.4	216.7	152.9	130.2	149.1	159.2	159.0	308.9	89.6	136.3
190.5	260.6	132.6	230.4	140.1	191.6	155.4	168.5	138.1	126.6
264.0	113.7	116.8	84.8	153.6	361.3	64.1	66.5	222.9	217.3

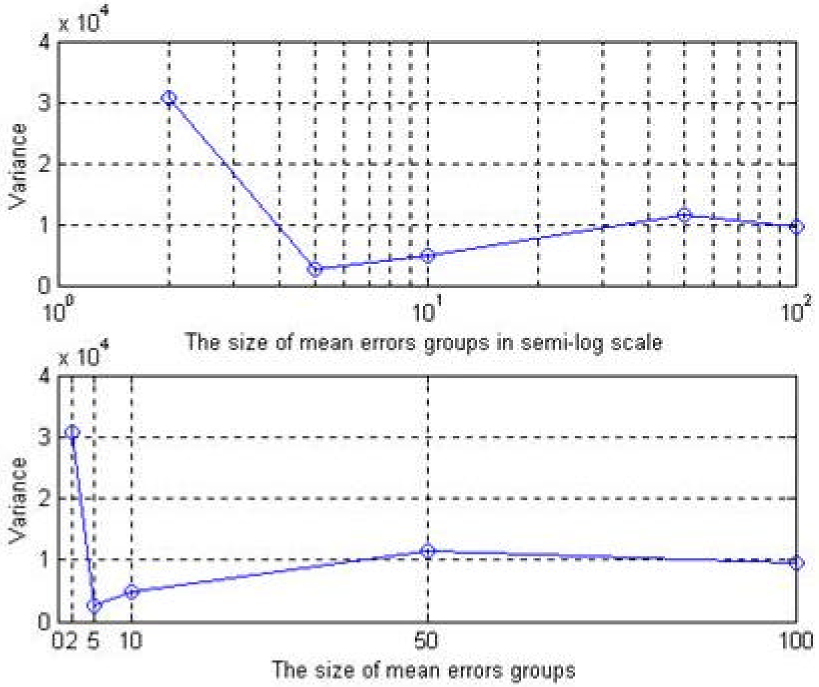


Figure E.1 Variance for mean errors in SOA-based PL estimation (Attempt # 1)

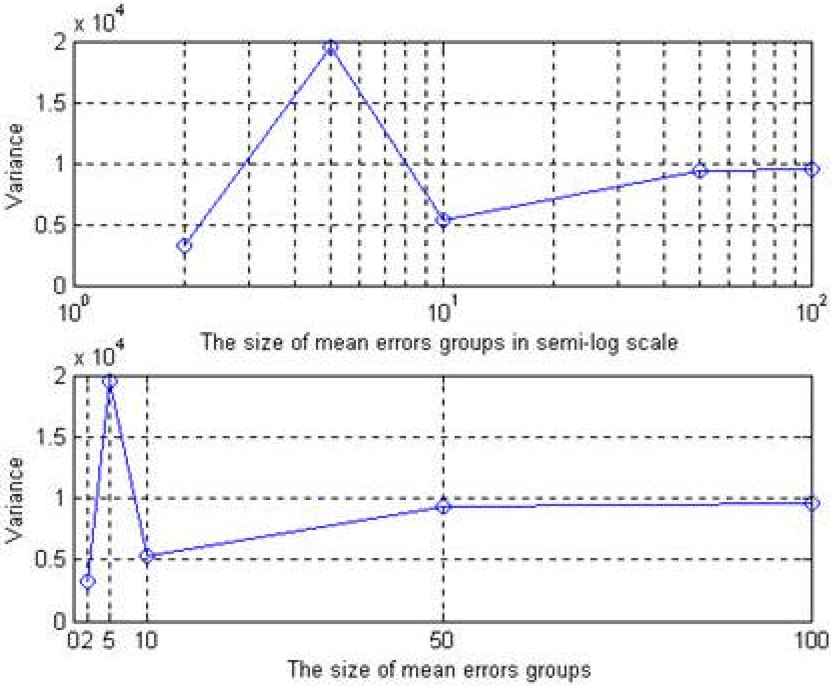


Figure E.2 Variance for mean errors in SOA-based estimation (Attempt # 2)

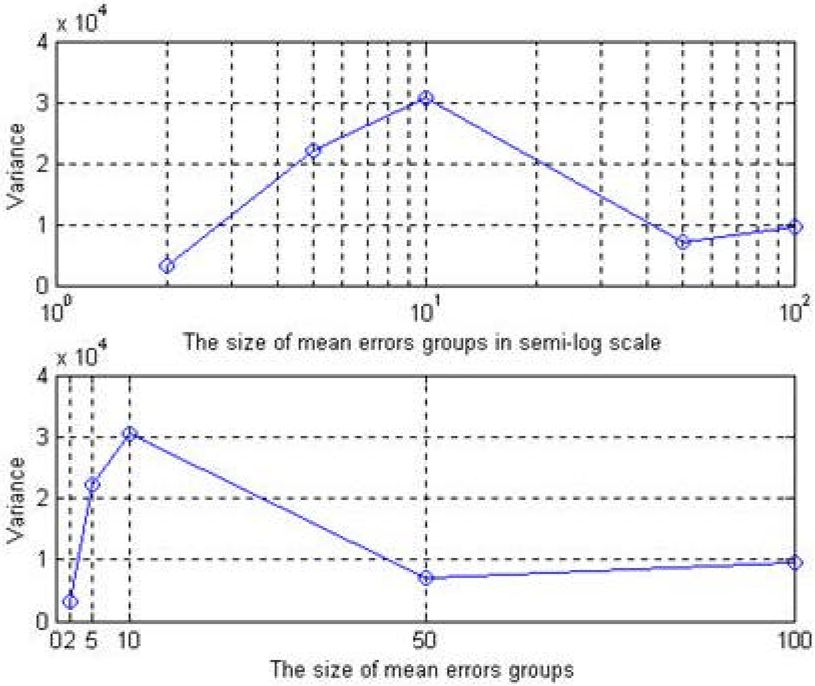


Figure E.3 Variance for mean errors in SOA-based PL estimation (Attempt # 3)

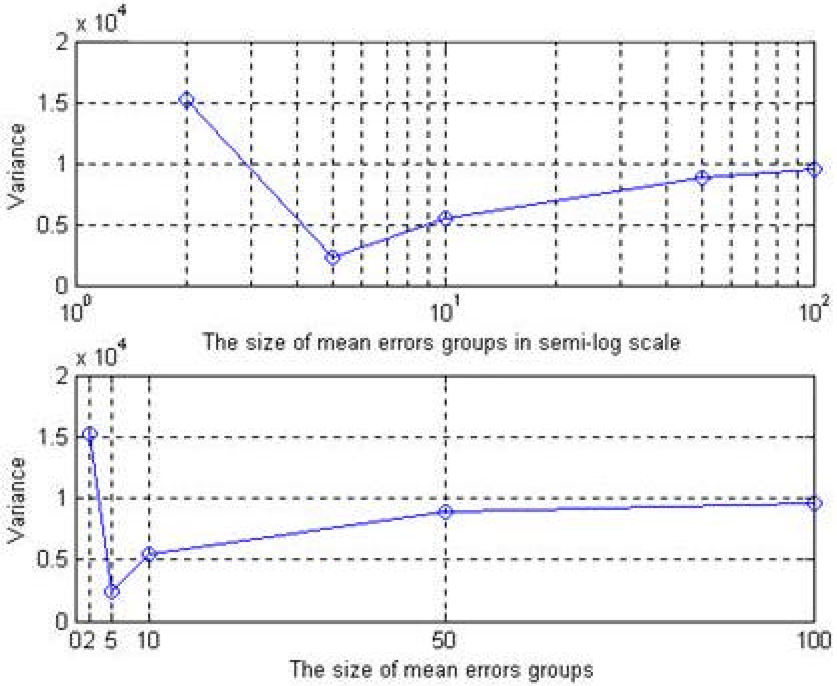


Figure E.4 Variance for mean errors in SOA-based PL estimation (Attempt # 4)

From the above figures, a group size of 100 does not appear to have a significantly influence on the mean error variances for a group with size 50. Since 50 is the minimum number that the corresponding variance starts to remain relative constant, it is regarded as the optimal number of MSs plotted in the limited area in SOA-based scheme. In our simulation, the average performance is trustful over the sampling number of 50.

Appendix F: Backpropagation Learning and LM Training Algorithms

F.1 Neural Network Learning Method — Backpropagation

There are various learning methods for multilayer NN networks. In feed-forward NNs, backpropagation is one of the methods. Backpropagation is an abbreviation for the backwards propagation of error. Figure F.1 provides a fully connected feed-forward NN example with 53 inputs, one 20 neurons hidden layer, and 2 outputs. A more detailed connected figure is specified in Figure 4.4.

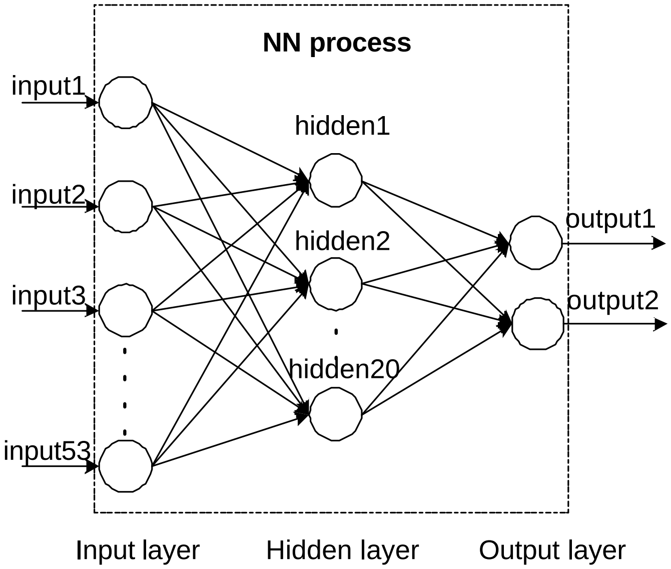


Figure F.1 A fully connected feed-forward 53-20-2 NN example

The following are the steps of using the backpropagation algorithm in a NN [Hsiung99, Gurney96, and Gallant93]:

- Create a fully connected feed-forward network with a specified number of inputs, hidden, and output sigmoid units.
- Initialize all weights $\{w_{i,j}\}$ in the network to small random values, where i, and j identify the layer and node.
- Repeat until the algorithm converges. For example, until weight changes and

changes in the mean squared error, e , become sufficiently small.

1. Forward propagation step: Take the next training example, feed the input vector through the network and compute every neuron in the network. This is done by computing the weight sum coming into a neuron and then applying the sigmoid function,

$$o = s\left(\vec{w}\vec{x}\right), \tag{F.1}$$

where the $\vec{x}$ vector denotes the activation of previous layer, the $\vec{w}$ vector denotes the weights linking the neuron unit to the previous neuron layer, and *o* is activation of each unit. In the Equation (F.1), the activation function is:

$$s\left(y\right) = \frac{1}{1+e^{-y}}\,. \tag{F.2}$$

2. Compute the square error of the network: This step is done by taking the sum of the squared error of every unit in the output layer,

$$E(\vec{w}) = \frac{1}{2}\sum_{k \in outputs}\left(t_k - o_k\right)^2 , \tag{F.3}$$

where *t* denotes a target value in the target vector (The target vector is associated with the training sample, the input vector.), and *o* denotes the activation of a neuron in the output layer.

3. Backward propagation step: Start with the outputs, make a backwards pass through the output and intermediate cells. It is computed as the following:

For each output neuron *k*, calculate the error term d_k as:

$$d_k = o_k\left(1 - o_k\right)\left(t_k - o_k\right) \tag{F.4}$$

The error term is related to the partial derivative of each weight with respect to the network error.

For each hidden node *h*, calculate the error term as:

$$\boldsymbol{d}_h = o_h(1 - o_h)\sum_{k \in outputs} w_{kh}\boldsymbol{d}_k \qquad \text{(F.5)}$$

where *h* is an integer, between 1 and the number of hidden layer. The hidden node error term depends on the error terms calculated for the output units.

4. Update each network weight as: The network weights are updated by using the Levenberg Marquardt algorithms that are mainly discussed in next Section F.2. Equation (F.10) in that section is used in the backpropagation weights training and updating.

F.2 NN Training Method — Levenberg Marquardt Algorithm

Similar to the quasi-Newton methods, the LM algorithm was designed to approach second-order training speed without having to compute the Hessian matrix. When the performance function has the form of a sum of squares, the Hessian matrix H_m can be approximated as [Gallant93], [Hagan94]:

$$H_m = J^T J \qquad \text{(F.6)}$$

and the gradient can be computed as:

$$g = J^T \boldsymbol{d} \qquad \text{(F.7)}$$

where *J* is the Jacobian matrix that contains first derivatives of the network errors with respect to the weights and biases, and d is a vector of network errors mentioned in Section F.1. The Jacobian matrix can be computed through a standard backpropagation technique [Hagan94], which is much less complex than computing the Hessian matrix.

Newton’s method can be written as:

$$w_{k+1} = w_k - H_k^{-1} g_k \,, \tag{F.8}$$

by combining Equation (F.6), (F.7) and (F.8), we get:

$$w_{k+1} = w_k - \left[J_k^T J_k \right] J_k^T d_k \,. \tag{F.9}$$

The problem with Equation (F.9) is that $J_k^T J_k$ may not be invertible. Thus, the LM algorithm warrants invertibility,

$$w_{k+1} = w_k - [J_k^T J_k + g\, I]^{-1} J_k^T d_k \tag{F.10}$$

If the scalar$g$ is zero, the equation degrades to Newton’s method using approximate Hessian matrix. If g is large, it becomes steepest descent with a small step size. Newton’s method is faster and more accurate near an error minimum.

Appendix G: Additional Details about the NN Simulation and Performance

G.1 Nguyen-Widow Weights Initialization

This section presents the Nguyen-Widrow weights initialization algorithm.

NN uses LM algorithm to train the weights. The random choice of initial weights and biases will affect the performance of LM algorithm. If the average performance of the algorithm is required, a test using several different sets of initial weights and biases will be carried out. In the thesis, for each NN architecture (fix layers and nodes in the layers), a training phase with eight random different sets of initial weights and biases experiments is performed to get the average value of mean error results. The chosen initial weights determine the starting point in the error landscape, which controls whether the learning process will end up in a local minimum or the global minimum. The easiest method is to select the weights randomly from a suitable range, such as between [-0.1, 0.1] or [-2, 2]. More sophisticated approaches to select the weights, such as the Nguyen-Widrow initialization which calculates the interval from which the weights are taken in accordance with the number of input neurons and the number of hidden neurons, can improve the learning process [Schmidt96].

D. Nguyen and B. Widrow use a multilayer perceptron with piecewise linear activation functions as an approximation of a network with logistic activation functions. Based on this simplification, they calculate an optimal length of $d_{in}\sqrt{\mathrm{Y}}$ for the randomly initialized weight vectors and an optimal bias range of $[-d_{in}\sqrt{\mathrm{Y}}, d_{in}\sqrt{\mathrm{Y}}]$ for neurons in the hidden layer, where Y is the number of hidden nodes and d_{in} is the fan-in (or in-degree) of a neuron, without justifying this interval further in a theoretical manner.The weights of the neurons in the output layer are randomly initialized in the interval [-0.5, 0.5], without any justification given [Nguyen90]. This algorithm only changes the weights and biases initailization in each training experiment, it does not effect the size or structure of the NN.

G.2 Size of Neural Network Determination

This section presents the detail simulations in determining the NN size, the process to find the optimal number of hidden layers and nodes in each hidden layers. Because of the randomly different initialized weights of NN, the average performance of the NN algorithms is evaluated by using several different sets of initial weights and biases. In the thesis, for a fixed NN architecture, each average value of mean errors represents an average of eight different training experiments, in each of which different random initial weights are used. Mean error of one training phase is evaluated on average of the eight training experiments in all tables below. The number of eight is a reasonable number to get statistical results of average, as verified in Section G.3. It is assumed that all the simulations are based on evaluating the PL criteria — mean error. The following PL mean error results in Table G.1, G.2, G.3, G.4 and G.5 are provided in the 0, 1, 2, 4 and 10 hidden layers NN simulations respectively. The simulation results clearly verified that a 53-20-2 NN is the optimal NN size to be selected in the simulation.

- No hidden layer :

Table G.1 A zero-hidden layer NN
53nodes (input) – 2 nodes (outputs) NN

Nodes in hidden layer (#)	Mean error (m) (Training phase)
0	82.2

- One hidden layer :

Table G.2 Seven one-hidden layer NNs
53nodes (input) – X nodes (hidden) – 2 nodes (outputs) NN

Nodes in hidden layer (#)	Mean errors (m) (8 experiments)	Mean error (m) (Training phase)
X = 5	77.1,78.37, 75.89, 81.38, 76.49, 76.69, 77.46,79.1	77.8
X = 10	72.7, 79.33, 78.59, 73.54 76.85,74.78,72.79, 76.23	75.6
X = 20	68.8, 68.89, 70.78, 72.80, 70.05, 71.76, 67.22, 73.34	70.5
X = 25	70.2, 73.81, 70.94, 72.11 73.67, 74.01, 75.06, 68.87	72.3
X = 30	71.3, 76.09, 75.00, 75.85, 66.99, 74.95, 77.25, 72.77	73.8
X = 50	79.1, 83.68, 73.63, 78.84, 69.47, 80.57, 84.32, 79.56	78.7
X = 100	83.6, 97.89, 80.39, 73.84, 86.37, 87.13, 77.32, 96.06	85.3

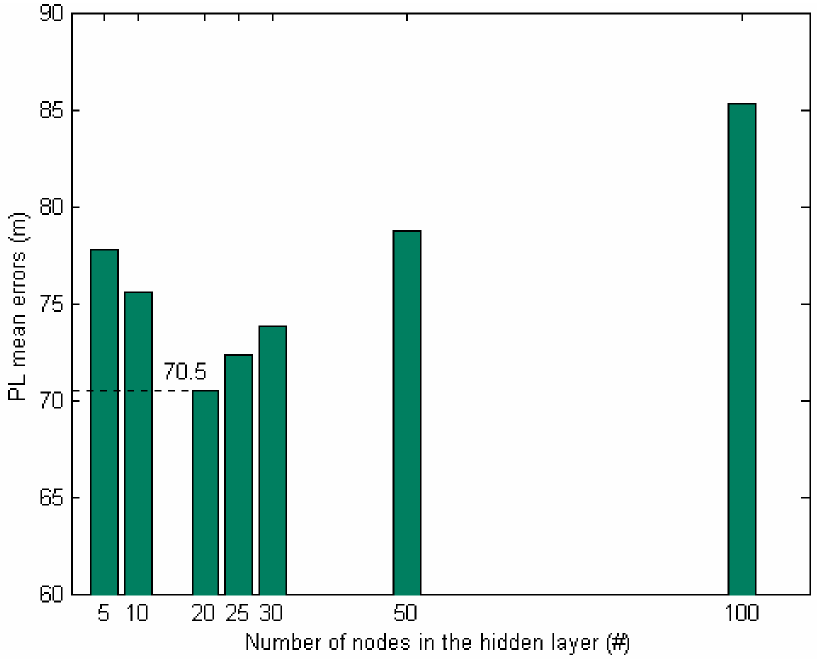


Figure G.1 PL mean errors vs. Number of nodes in one hidden layer NN simulation

Diagram in Figure G.1 shows that mean error is the smallest at 20 nodes in one hidden layer NN structure simulation.

- Two hidden layers :

Table G.3 Four two-hidden layer NNs
53 nodes (input) –X nodes (hidden1) –Y nodes (hidden 2) – 2 nodes (outputs) NN

Nodes in hidden layer1(#)	Nodes in hidden layer2 (#)	Mean error (m) (Training phase)
X = 10	Y= 10	217.02
X = 20	Y= 15	236.46
X= 30	Y= 15	178.87
X= 30	Y= 20	182.56

- Four hidden layers:

Table G.4 Two four-hidden layers NNs
53nodes (input) – X1 nodes (hidden1) – X2 nodes (hidden 2) – X3 nodes (hidden 3) – X4 nodes (hidden 4) – 2 nodes (outputs) NN

Nodes in hidden layer (#)				Mean error (m) (Training phase)
1	2	3	4	
X1 = 5	X2 = 5	X3 = 5	X4 = 5	272.4
X1 = 10	X2 = 10	X3 = 10	X4 = 10	265.3

- Ten hidden layers (1input-10hidden-1output):

Table G.5 A ten-hidden layer NN
53nodes (input) – X1 nodes (hidden1) – X2 nodes (hidden 2) – X3 nodes (hidden 3) – X4 nodes (hidden 4) – X5 nodes (hidden1) – X6 nodes (hidden1) – X7 nodes (hidden1) – X8 nodes (hidden1) – X9 nodes (hidden1) – X10 nodes (hidden1) –2 nodes (outputs) NN

Nodes in hidden layer (#)										Mean error (m)
1	2	3	4	5	6	7	8	9	10	
X1: 2	X2: 2	X3: 2	X4: 2	X5: 2	X6: 2	X7:2	X8: 2	X9: 2	X10: 2	328.5

G.3 The Chosen Number of Training Experiments Determination

This section focuses on the determination of the number of eight training experiments in a training phase.

Because the weights are randomly initialized according to the Nguyen-Widrow algorithm in every training experiment, the performance of each mean error differs from others. The statistical number of training experiments must be determined in finding the average mean error of a training phase. Similar to Appendix E, the determination of this number is assessed through an analysis of variance.

In the simulation, it is assumed that all the inputs to the NN are equivalent in every NN training experiment, but the weights are initialized differently in each experiment. A fixed NN structure chosen in every training experiment is 53–20–3. To analysis variance, the sizes of mean errors groups are assumed as: {2, 4, 8, 16, 32, 64}, and these groups are randomly chosen within the 64 NN mean errors provided in Table G.6.

Table G.6 64 mean errors (m) of 64 NN (53–20–3) training experiments

75.4	73.4	73.7	69.2	67.8	73.6	69.8	71.0
68.5	75.1	75.2	72.5	73.4	69.1	73.7	72.5
69.2	67.4	69.1	68.8	73.9	71.9	69.5	68.3
69.6	66.1	68.5	73.5	73.8	70.5	73.3	71.8
70.0	69.1	70.0	68.9	72.9	71.1	73.0	67.9
68.8	71.7	68.6	69.7	72.8	69.2	72.9	73.1
77.4	68.9	67.4	74.5	71.2	73.2	71.9	71.8
68.6	69.6	71.7	69.1	72.4	70.2	70.5	74.2

The figures of variances with respect to the groups' size are illustrated in Figure G.2, G.3, G.4 and G.5, in which the varying random mean errors are chosen in each group for every attempt.

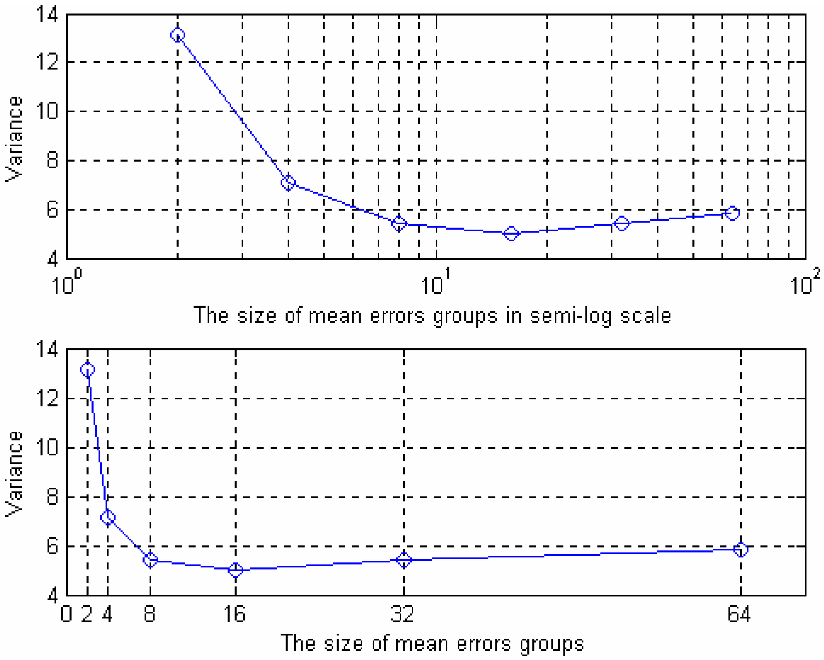


Figure G.2 The variance of mean errors in NN (53-20-2) estimation (Attempt # 1)

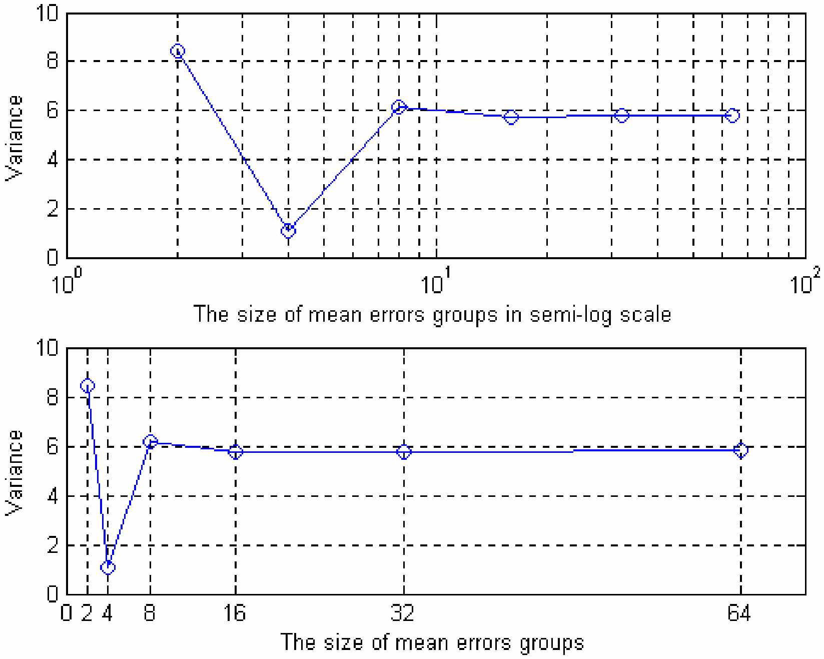


Figure G.3 The variance of mean errors in NN (53-20-2) estimation (Attempt # 2)

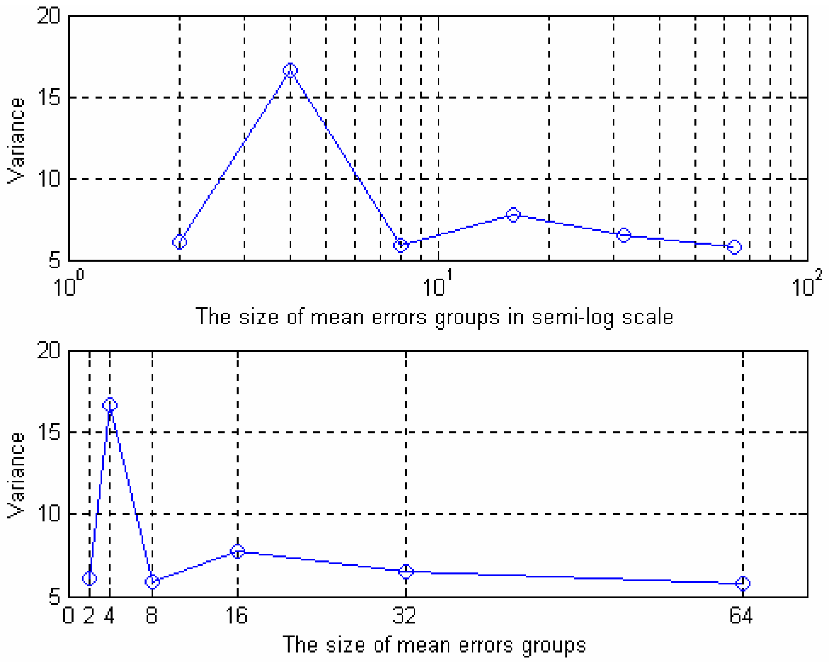


Figure G.4 The variance of mean errors in NN (53-20-2) estimation (Attempt # 3)

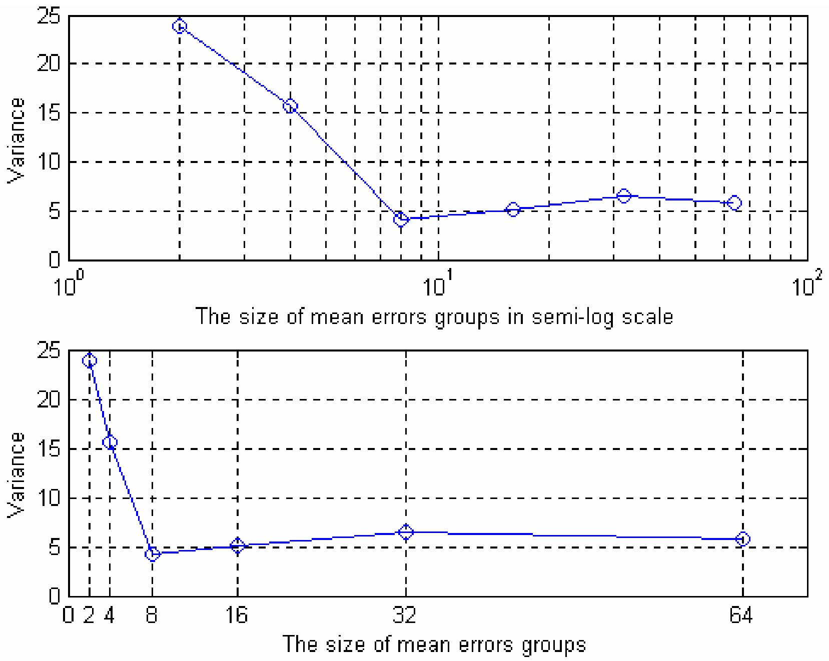


Figure G.5 The variance of mean errors in NN (53-20-2) estimation (Attempt # 4)

From the above figures, the number of eight is chosen to be the minimum number that the variance starts to remain relative constant. It is concluded that eight is the optimal number of training experiments, which determines an average mean error in a training phase.

G.4 The Evaluation of Mean Delays in NN Inputs

The results of the simulation for mean delays are presented in this simulation. In order to prove that mean delays significantly improve the NN training performance, one of the criteria so-called mean error is selected to justify the performance. The simulation of both zero mean delays and practical mean delays applied to the NN are performed. As indicated in Figure 4.3, the simulation of d_1', d_2', and $d_3' = 0$ is compared with d_1', d_2' , and $d_3' \neq 0$. The comparison of the two simulation results demonstrates that there is a great difference between the two obtained mean errors.

Both simulation results are demonstrated by the mean error of a NN training phase, which is an average computed by mean errors of the eight training experiments. Each training experiment is performed to obtain a mean error that indicates the NN performance. The NN structure used in both simulations is assumed to be 53–20–2.

Table G.7 The comparison of zero mean delays and practical mean delays inputs of the NN performance

Inputs of the NN	NN structure	Mean errors (m) (8 training experiments)	Mean error (m) (Training phase)
With 3 mean delays	53–20–2	68.8, 68.9, 70.8, 72.8, 70.1, 71.8, 67.2, 73.3	70.5
With 3 mean delays = 0	53–20–2	84.2, 86.3, 71.0, 76.5 75.9, 84.0, 83.6, 74.9	79.6

From Table G.7, the variance between 79.6 m and 70.5 m is 9.1 m which is a large different for the E-911 requirement. It is concluded that mean error can significantly help improve the NN training performance. However, it consumes more time and leads a higher complex PL scheme.

Appendix H: The Determination of Number of MSs in SOA and NN Integrated Technique Training and Testing Phases (1500, 50)

H.1 Determination of the Number of Training patterns in Integrated Technique

In this section, a number of comparisons of the various training sample size are performed based on evaluating the NN performance for every training sample size. Once the training sample size or MSs sample size is determined, a NN training experiment may provide good performance.

The simulation is performed on a Pentium II 400 MHz computer with 384 MB ram, and the NN structure is 53-20-2. The mean error is performance criteria in measuring performance of the NN training experiment.

Table H.1 summarizes the results of training the network using five different training sample sizes. Each entry of average of mean errors and mean time in the table represents the average of eight training experiments, where different random initial weights but same training sample sizes are used in each experiment.

Table H.1 Comparison of the performance for variant NN training sample sizes

Training sample sizes (#) (MSs)	Mean Time (s)	Mean error (m) (Training phase)
500	286.3	86.1
1000	1153.5	74.8
1500	2952.1	70.5
2000	5715.5	68.4
2500	8627.5	67.9

The simulation results show that the number of 1500 training sample is optimal in training the NN. Larger training sample size cannot help NN with more significant improvement and that takes much longer time in training. Results with smaller values in estimation demonstrate that they are not large enough in getting required performance.

H.2 Determination of the Number of Testing patterns in Integrated Technique

Similar to Appendix E, the determination of number of test sample size (the number of MSs) for two integrated technique is provided. The determination of this number is based on analysis of variance.

In the simulation, the sizes of mean errors groups are assumed as: {2, 5, 10, 50, 100} and these groups are randomly chosen within the 100 integrated technique mean errors provided in Table H.2. Every mean error in Table H.1 indicates a performance result of a NN training experiment. By calculating each group's corresponding variance, the figures of variances with respect to the groups' size are illustrated in Figure H.1, H.2, H.3 and H.4, where the different random mean errors are chosen in all groups for every attempt.

Table H.2 100 mean errors (m) of integrated technique with a 52-20-2 NN

85.9	35.9	84.3	102.9	193.6	43.2	34.1	42.1	87.6	77.1
168.0	43.2	54.3	83.9	47.8	38.2	90.6	81.1	102.1	56.7
14.6	76.8	56.3	87.8	33.8	140.2	11.8	26.4	78.3	77.9
37.1	41.8	33.7	36.5	154.8	56.6	46.7	41.7	98.2	299.2
77.3	123.5	56.7	111.0	37.6	157.1	41.0	54.4	35.4	10.2
49.2	45.6	54.0	274.3	72.1	46.3	38.9	75.9	30.0	88.0
218.6	166.4	66.9	16.9	68.1	229.9	11.4	19.6	56.7	45.6
26.3	55.0	30.8	56.4	28.2	75.5	12.8	57.7	163.2	67.5
28.8	31.1	12.5	21.6	67.6	184.3	88.2	78.9	64.6	23.5
39.2	60.3	18.9	52.6	19.6	60.4	94.7	12.5	50.2	110.8

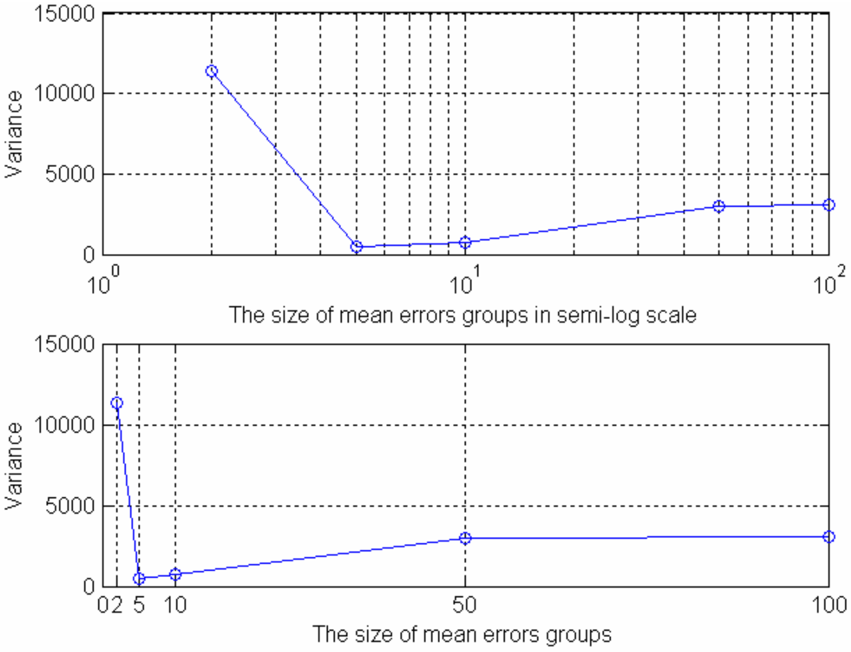


Figure H.1 The variance for mean errors in integrated technique (Attempt # 1)

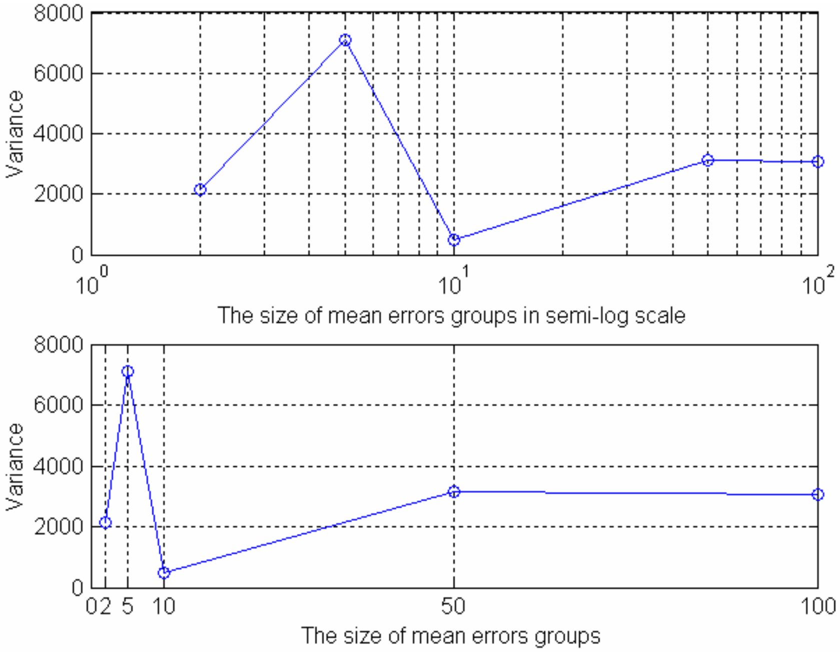


Figure H.2 The variance for mean errors in integrated technique (Attempt # 2)

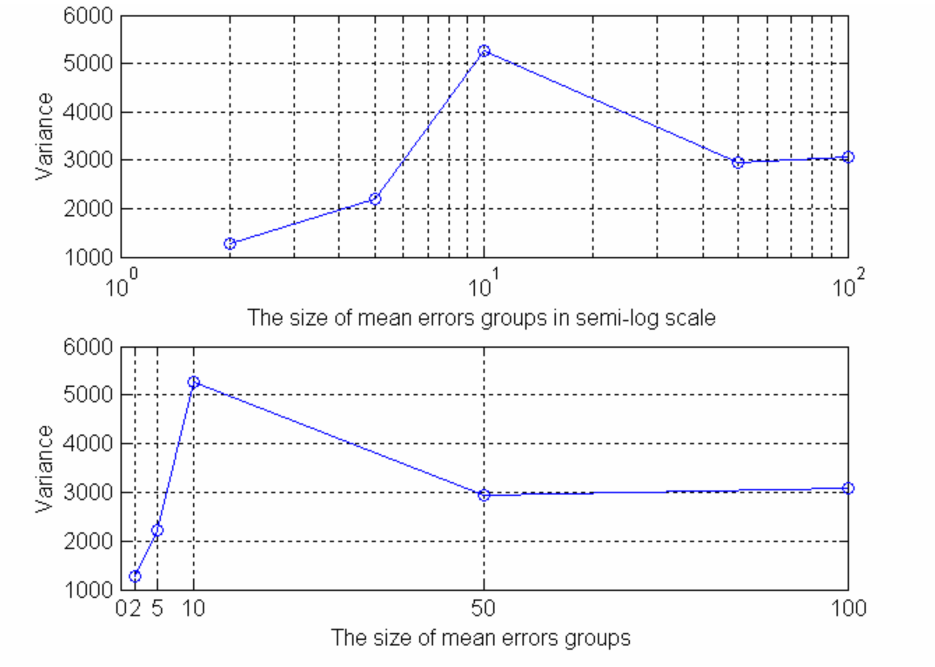


Figure H.3 The variance for mean errors in integrated technique (Attempt # 3**)**

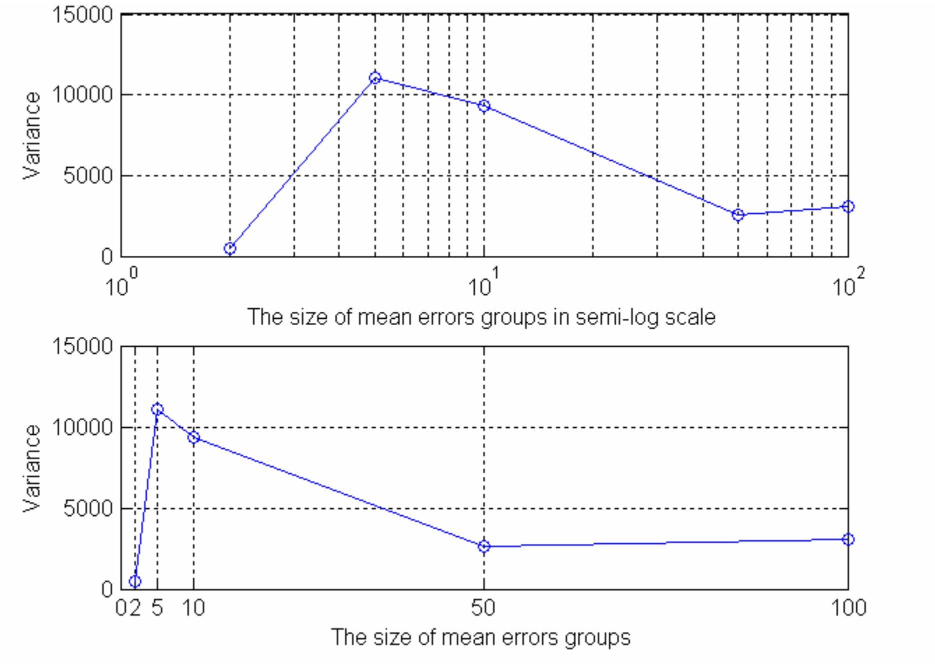


Figure H.4 The variance for mean errors in integrated technique (Attempt # 4)

From the previous figures, the number of 50 is selected because the statistical results of the variance remains constant after this point of test sample size.

Appendix I: MATLAB® Source Code

This chapter lists a few short major parts of source code involved this thesis.

I.1 SOCLSFM

Here is an example of building the first SOCLSFM propagation set. The two other propagation sets are the same except titles in labeling each step of the SOCLSFM implementation. A few of the simple functions developed in this program are not provided.

```
%######################################################################
% This program is built for the SOCLSFM for the first propagation set
% of SOCLSFM.
% h1,h2,h3,h4. hwider1 are labeled for each step of the SOCLSFM
% implementation, which includes random coefficient values, propagation
% delay, exponential power delay profile and RMS delay spread, path
% loss and log-normal shadowing, and SOC in RMS delay spread

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: December 19, 2000
%######################################################################
clear;
close all;
R=2000;%(m)
d=50;%(m)
c=2.998e8;%(m/s)
rms_delay_spread=1e-7;%(100ns)
Nover_cell_width=length([-R:d:R]);
BIS95=1.2e6;%(MHz)
T_sampling_receiver=1/BIS95;%(s)
rms_delay_spreadn=rms_delay_spread/T_sampling_receiver;
T_prop_edgeofcell=R/c;%(us)
Tmax_Delay=T_prop_edgeofcell*2;%(us)
N_taps=Tmax_Delay/T_sampling_receiver;
N_taps=round(N_taps);

%generate the Random data
h1=(1/sqrt(2))*...
  (randn(Nover_cell_width,Nover_cell_width,N_taps)+...
  1*i*randn(Nover_cell_width,Nover_cell_width,N_taps));
i_centre=find([-R:d:R]==0);
save h1 h1;

%Propagation delay
for k=1:Nover_cell_width
   for l=1:Nover_cell_width
      distance_kl=sqrt((k-i_centre)^2+(l-i_centre)^2)*d;
      time_kl=distance_kl/c;
      N_zeros=fix(time_kl/T_sampling_receiver)+1;
      h1(k,l,(1:N_zeros))=0;
   end
end
h2=h1;
save h2 h2;

%RMS_delay
b=1/(rms_delay_spreadn^2)+2;
alpha=(1/2)*log((b+sqrt(b^2-4))/2);
for k=1:Nover_cell_width
   for l=1:Nover_cell_width
      distance_kl=sqrt((k-i_centre)^2+(l-i_centre)^2)*d;
      time_kl=distance_kl/c;
      N_zeros=fix(time_kl/T_sampling_receiver)+1;
      for i=1:N_taps-N_zeros
      h3(k,l,(N_zeros+i))=h2(k,l,(N_zeros+i))...
         .*exp(-2*alpha*i);
      end
   end
end
save h3 h3;

%Path loss/log normal shadowing
n=4;%path loss exponent
sigma=11.8;%dB shadowing variance
d0=10;%(m)
Pld0bar=0;%dBW at 50(m)
for k=1:Nover_cell_width
   for l=1:Nover_cell_width
      distance_kl=sqrt((k-i_centre)^2+(l-i_centre)^2)*d;
      if distance_kl~=0
       Pldbar=Pld0bar-10*n*log10(distance_kl/d0);
      else
       Pldbar=Pld0bar;
      end
      Pld=Pldbar+sqrt(sigma)*(randn(1)/5); %log normal shadowing
      Pld=10^(Pld/10);%not in dB
        %normalize energy to 1
      h4(k,l,:)=h3(k,l,:)/sqrt(sum(abs(h3(k,l,:)).^2));
      h4(k,l,:)=h4(k,l,:)*sqrt(Pld);
   end
end
save h4 h4;

hwider1 = h4;% the impulse hwider1 has been assigned to a old impulse
response
% Add splashes of changes
%======================================
nsplashes = 10 ; % number of splashes
width_per_splashes = 10;% (m)
mean_splash_height = 2;
for i = 1: nsplashes
% Step 1 : pick a random point xr, yr.
  range = length(-R:d:R);
  xr = fix(rand(1)*range);
  yr = fix(rand(1)*range);
```

```
    while (xr == 0)|(yr == 0)
     xr = fix(rand(1)*range);
     yr = fix(rand(1)*range);
    end

% Step 2: generate a relative change, Rr.
  Ur = rand(1);
  Rr = (2/sqrt(pi))*mean_splash_height*sqrt(-(log(1 - Ur)));


% Step 3: generate a relative width, W is a Rayleigh RV, with mean
% equal to mean width per splash Ws
  Ws = width_per_splashes;
  Uw = rand(1);
  W = (2/sqrt(pi))*Ws*sqrt(-(log(1 - Uw)));

% Step 4: calculate the 2-dimensional bell-curve point using Rr and W
% bellcurve: height = Rr , width = W
  bc_bell = zeros(range,range);
    for i2 = 1:range
       for j2 = 1:range
        bc_bell(i2,j2) = Rr*exp(-((i2-xr).^2)/(2*W^2))*...
                             exp(-((j2-yr).^2)/(2*W^2));
       end
    end

% Step 5: generate a new relative changes in RMS delay spread
  rc_bell = zeros(range,range);
   for i3 = 1:range
       for j3 = 1:range
        Rc = bc_bell (i3,j3);
        Uc = rand(1);
        rc_bell(i3,j3) = (2/sqrt(pi))*Rc*sqrt(-(log(1 - Uc)));
       end
   end

 %Step 6: applying changes
 %Step(a):Given
 present_RMS_delay_spread_whole= calcrmsds(h4);
 for i4 = 1:range
    for j4 = 1:range
    onerc = rc_bell(i4,j4);%one relative change
    mean_delay_number = find_mean_delay1(i4,j4);

 %Step(b):Find hwider
    present_RMS_delay_spread = present_RMS_delay_spread_whole(i4,j4);
    alpha1 = RMS_delay_spread_to_alpha(present_RMS_delay_spread);
    new_RMS_delay_spread = (onerc+1)*present_RMS_delay_spread;
    alpha2 = RMS_delay_spread_to_alpha(new_RMS_delay_spread);
    indices_of_interest = [1 + mean_delay_number:N_taps];
     for i = min(indices_of_interest):max(indices_of_interest)
      hwider1(i4,j4,i) = hwider1(i4,j4,i).*...
                 exp(-2*(alpha2-alpha1).*(indices_of_interest(i-
mean_delay_number)-...
                 (mean_delay_number+1)));
      hwider1(i4,j4,:)                                                 =
hwider1(i4,j4,:)*(sum(abs(h4(k,l,:)).^2))/(sum(abs(hwider1(i4,j4,:)).^2
));
     end
    end
 end
end % for loop of per splash
save hwider1 hwider1;

%#######################################################################
% This program is built for plot the mean received power for each step
% of the first propagation set.

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: July 12, 2002
%#######################################################################
 clear;
 clf;
 close all;
 figure(1)
  clear ;
  load h1.mat ;
  hedB = calcmrp( h1 ) ;
  colormap(gray);
  R=2000;%(m)
  d=50;%(m)
  xset=[-R:d:R];
  yset=[-R:d:R]';
  contour(xset,yset,hedB); colorbar ;
  title('Contour plot of mean received power of random data');
  axis('square');

 figure(2)
  clear ;
  load h2.mat ;
  hedB =  calcmrp ( h2 ) ;
  colormap(gray);
  R=2000;%(m)
  d=50;%(m)
  xset=[-R:d:R];
  yset=[-R:d:R]';
  contour(xset,yset,hedB) ;colorbar ;
   title('Contour plot of mean received power after added propagation
delay signal');
  axis('square');

 figure(3)
  clear ;
  load h3.mat ;
  hedB = calcmrp ( h3 ) ;
  colormap(gray);
  R=2000;%(m)
  d=50;%(m)
  xset=[-R:d:R];
  yset=[-R:d:R]';
  contour(xset,yset,hedB) ;colorbar ;
  title('Contour plot of mean received power of added RMS delay
spread');
  axis('square');

 figure(4)
   clear ;
   load h4.mat ;
   hedB = calcmrp ( h4 ) ;
   colormap(gray);
   R=2000;%(m)
   d=50;%(m)
   xset=[-R:d:R];
   yset=[-R:d:R]';
   contour(xset,yset,hedB) ;colorbar ;
    title('Contour plot of mean received power of added pass loss and
log normal shadowing  signal');
   axis('square');

  figure(5)
   clear ;
   load hwider1.mat ;
   hedB = calcmrp ( hwider1 ) ;
   colormap(gray);
   R=2000;%(m)
   d=50;%(m)
   xset=[-R:d:R];
   yset=[-R:d:R]';
   contour(xset,yset,hedB) ;colorbar ;
   title('Contour plot of mean received power of added splashes of
changes signal');
   axis('square');

%#######################################################################
% This program is built for plot the RMS delay spread for each step of
% the first propagation set.

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: July 12, 2002
%#######################################################################
clear;
clf;
close all;

figure(1)
    clear;
    load h1.mat;
    R=2000;%(m)
      d=50;%(m)
      xset=[-R:d:R];
      yset=[-R:d:R]';
      rms_delayspread_arrary=calcrmsds(h1);
    colormap(gray);
      contour(xset,yset,abs(rms_delayspread_arrary)) ;
      axis('square');
      colorbar;
  title('Contour plot of RMS delay spread of random data');
  axis('square');

figure(2)
      clear;
    load h2.mat;
      R=2000;%(m)
      d=50;%(m)
      xset=[-R:d:R];
      yset=[-R:d:R]';
      rms_delayspread_arrary=calcrmsds(h2);
    colormap(gray);
      contour(xset,yset,abs(rms_delayspread_arrary)) ;
      axis('square');
      colorbar;
   title('Contour plot of RMS delay spread of added propagation delay
signal');
  axis('square');

 figure(3)
  clear;
  load h3.mat;
  R=2000;%(m)
  d=50;%(m)
  xset=[-R:d:R];
  yset=[-R:d:R]';
  rms_delayspread_arrary=calcrmsds(h3);
  colormap(gray);
  contour(xset,yset,abs(rms_delayspread_arrary)) ;
  axis('square');
  colorbar;
  title('Contour plot of RMS delay spread of added RMS delay signal');
  axis('square');

 figure(4)
   clear;
   load h4.mat;
   R=2000;%(m)
      d=50;%(m)
      xset=[-R:d:R];
      yset=[-R:d:R]';
      rms_delayspread_arrary=calcrmsds(h4);
    colormap(gray);
      contour(xset,yset,abs(rms_delayspread_arrary)) ;
      axis('square');
      colorbar;
   title('Contour plot of RMS delay spread of added pass loss and log
normal shadowing signal');

 figure(5)
   clear;
   load hwider1.mat;
    R=2000;%(m)
      d=50;%(m)
      xset=[-R:d:R];
      yset=[-R:d:R]';
      rms_delayspread_arrary=calcrmsds(hwider1);
    colormap('gray');
      contour(xset,yset,abs(rms_delayspread_arrary)) ;
      axis('square');
      colorbar;
   title('Contour plot of RMS delay spread of added splashes of changes
signal');

%########################################################################
% This program is to built calcmrp (calculate mean received power)
% function

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: July 12, 2002
%########################################################################
function hedB = calcmrp ( h )
%CALCMRP hedB = calcmrp ( h )
% Takes a 3 dimensional array h, which is a 2 dimensional
% array of impulse responses and makes calculations
% of mean received power, with the maximum value normalized
% to 0 dB. hedB is a two dimensional array of power values in dB.
% Example:
% clear ;
% for i = 1:10 ;
% for j = 1:10 ;
% for k = 1:10 ;
% h(i,j,k) = 1 / ( 40-(i+j+k) ) ;
% end
% end
% end
% hedB = calcmrp ( h ) ;
% mesh(hedB) ;
%
% clear ;
% load h2.mat ;
% hedB = calcmrp ( h2 ) ;
% contour(hedB) ;

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: December 19, 2000

% Find the energy of each tap.
hm = abs(h).^2 ;

% Find the energy of all the taps.
he = sum(hm,3) ;

% Express the energy in dB.
maxmaxvalue = max(max(he)) ;
hedB = 10 * log10 ( he/maxmaxvalue ) ;
```

```
return ;

%#######################################################################
% This program is to built calcmsds (calculate RMS delay spread)
% function

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: July 12, 2002
%#######################################################################
function rms_delayspread_arrary=calcrmsds(h)
%CALCMRP hedB = calcmrp ( h )
% Takes a 3 dimensional array h, which is a 2 dimensional
% array of impulse responses and makes calculations
% of 3_d rms delay spread and contour plot of rms delay spread.
% Example:
%     clear;
%     load h2.mat;
%     R=2000;%(m)
%     d=50;%(m)
%     xset=[-R:d:R];
%     yset=[-R:d:R]';
%     rms_delayspread_arrary=calcrmsds(h2);
%     mesh(xset,yset,abs(rms_delayspread_arrary));
%     xlabel( 'x' ) ;
%     ylabel( 'y' ) ;
%     zlabel( 'z' ) ;
%     title('RMS delay spead');
%
%     clear;
%     load h2.mat;
%     R=2000;%(m)
%     d=50;%(m)
%     xset=[-R:d:R];
%     yset=[-R:d:R]';
%     rms_delayspread_arrary=calcrmsds(h2);
%     contour(xset,yset,abs(rms_delayspread_arrary)) ;
%     axis('square');
%     colorbar;

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 20, 2000
% Date Modified: December 20, 2000

% Find the rms delay spread of each tap.
R=2000;%(m)
d=50;%(m)
Nover_cell_width=length([-R:d:R]);
rms_delayspread_arrary=zeros(Nover_cell_width,Nover_cell_width);
for i=1:Nover_cell_width
   for j=1:Nover_cell_width
      oneh=h(i,j,:);
      rms_delayspread_arrary_of_oneh=rms_delaycalc(oneh);
      rms_delayspread_arrary(i,j)= rms_delayspread_arrary_of_oneh;
   end
end
```

I.2 SOA

The following is the main program for SOA-based scheme evaluated on the SOCLSFM.

A few of the simple functions developed in this program are not provided.

```
%######################################################################
% This program is to setup the SOA-based scheme on SOCLSFM
% It involves the SOA-based scheme and SOA-based scheme results

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: July 12, 2002
%######################################################################
clear;
close all;
R=2000;%(m)
d=50;
i_centre=find([-R:50:R]==0);
i_centre_ref=[i_centre,i_centre];
bc1=[0,0];
bc2=[1500,0];
bc3=[750,750*sqrt(3)];
Num_try=50;
SE_SOA=zeros(1,Num_try);
%The points are chosen the central points inside the circle
%=============================================================
cofficient=rand(1,Num_try);
delta_test=linspace(0,2*pi,Num_try);
delta=rand(1,Num_try).*delta_test;
x=750+cofficient*866.*cos(delta);
y=433+cofficient*866.*sin(delta);
X_axis=x;
Y_axis=y;
%=============================================================
x_head=zeros(1,Num_try);
y_head=zeros(1,Num_try);
%For every point,find error between the estimated value and fixed value
p=0;%Initailize the number of error of estimate location less than 125m
to 0
q=0;%Initailize the outage location estimation
for i=1:Num_try
mc=[X_axis(i),Y_axis(i)];
fprintf('********************************************\n');
fprintf('The mobile is located at: %2.4d\n',mc(1));
fprintf('                          %2.4d\n',mc(2));
fprintf('The number of selected location %2.0d\n',i);
load hwider1.mat;
```

```
hedB1 = calcmrp ( hwider1) ;
load hwider2.mat;
hedB2 = calcmrp ( hwider2 ) ;
load hwider3.mat;
hedB3 = calcmrp ( hwider3 ) ;
ic=find([-R:d:R]==0);
%The calculate forms for Number 1 generation of LSF
mc_distance1=distance(bc1,mc);
%The calculate forms for Number 2 generation of LSF
mc_distance2=distance(bc2,mc);
%The calculate forms for Number 3 generation of LSF
mc_distance2=distance(bc3,mc);

%For each base station
%Get an impulse reponse by interpolation
%Call them hc1,hc2,hc3
%For the Bast Station 1
p3=[mc(1)-mod(mc(1),d),mc(2)-mod(mc(2),d)];
p1=[p3(1),p3(2)+d];
p2=[p3(1)+d,p3(2)+d];
p4=[p3(1)+d,p3(2)];
p_distance1=distance(bc1,p1);
p_distance2=distance(bc1,p2);
p_distance3=distance(bc1,p3);
p_distance4=distance(bc1,p4);
p_power1=hedB1(p_distance1(1),p_distance1(2));
p_power2=hedB1(p_distance2(1),p_distance2(2));
p_power3=hedB1(p_distance3(1),p_distance3(2));
p_power4=hedB1(p_distance4(1),p_distance4(2));
L1=len(p1,mc);
L2=len(p2,mc);
L3=len(p3,mc);
L4=len(p4,mc);
L=[L1,L2,L3,L4]';
w1=-(sqrt(2)*L1)/(2*d)+L2/(2*d)+L3/(2*d);
w2=-(sqrt(2)*L2)/(2*d)+L1/(2*d)+L4/(2*d);
w3=-(sqrt(2)*L3)/(2*d)+L1/(2*d)+L4/(2*d);
w4=-(sqrt(2)*L4)/(2*d)+L2/(2*d)+L3/(2*d);
w=[w1,w2,w3,w4];
w=sum(w);
hc1=p_power1*(w1/w)+p_power2*(w2/w)+p_power3*(w3/w)+p_power4*(w4/w);

%For the base station 2
p_distance1=distance(bc2,p1);
p_distance2=distance(bc2,p2);
p_distance3=distance(bc2,p3);
p_distance4=distance(bc2,p4);
p_power1=hedB2(p_distance1(1),p_distance1(2));
p_power2=hedB2(p_distance2(1),p_distance2(2));
p_power3=hedB2(p_distance3(1),p_distance3(2));
p_power4=hedB2(p_distance4(1),p_distance4(2));
L1=len(p1,mc);
L2=len(p2,mc);
L3=len(p3,mc);
L4=len(p4,mc);
L=[L1,L2,L3,L4]';
```

```
w1=-(sqrt(2)*L1)/(2*d)+L2/(2*d)+L3/(2*d);
w2=-(sqrt(2)*L2)/(2*d)+L1/(2*d)+L4/(2*d);
w3=-(sqrt(2)*L3)/(2*d)+L1/(2*d)+L4/(2*d);
w4=-(sqrt(2)*L4)/(2*d)+L2/(2*d)+L3/(2*d);
w=[w1,w2,w3,w4];
w=sum(w);
hc2=p_power1*(w1/w)+p_power2*(w2/w)+p_power3*(w3/w)+p_power4*(w4/w);

%For the Bast Station 3
p3=[mc(1)-mod(mc(1),d),mc(2)-mod(mc(2),d)];
p1=[p3(1),p3(2)+d];
p2=[p3(1)+d,p3(2)+d];
p4=[p3(1)+d,p3(2)];
p_distance1=round(distance(bc3,p1));
p_distance2=round(distance(bc3,p2));
p_distance3=round(distance(bc3,p3));
p_distance4=round(distance(bc3,p4));
p_power1=hedB3(p_distance1(1),p_distance1(2));
p_power2=hedB3(p_distance2(1),p_distance2(2));
p_power3=hedB3(p_distance3(1),p_distance3(2));
p_power4=hedB3(p_distance4(1),p_distance4(2));
L1=len(p1,mc);
L2=len(p2,mc);
L3=len(p3,mc);
L4=len(p4,mc);
L=[L1,L2,L3,L4]';
w1=-(sqrt(2)*L1)/(2*d)+L2/(2*d)+L3/(2*d);
w2=-(sqrt(2)*L2)/(2*d)+L1/(2*d)+L4/(2*d);
w3=-(sqrt(2)*L3)/(2*d)+L1/(2*d)+L4/(2*d);
w4=-(sqrt(2)*L4)/(2*d)+L2/(2*d)+L3/(2*d);
w=[w1,w2,w3,w4];
w=sum(w);
hc3=p_power1*(w1/w)+p_power2*(w2/w)+p_power3*(w3/w)+p_power4*(w4/w);

%Compute the power (Pr1,Pr2,Pr3)of hc1,hc2,hc3 not in dB
%find the power of hc1
Pr1=10^(hc1/10);

%Compute the power (Pr1,Pr2,Pr3)of hc1,hc2,hc3 not in dB
%find the power of hc2
Pr2=10^(hc2/10);

%Compute the power (Pr1,Pr2,Pr3)of hc1,hc2,hc3 not in dB
%find the power of hc3
Pr3=10^(hc3/10);

%Estimate location based on powers. assume n=4
%Find the first circle
n=4;
a=sqrt((Pr1/Pr2)^(2/n));
g=bc2(1);
xc1=g/(1-a^2);
yc1=bc1(2);
%r1=abs((a*g)/(1-a^2));
r1=(a*g)/(1-a^2);
```

```
%Find the second circle
n=4;
a=sqrt((Pr1/Pr3)^(2/n));
g=bc2(1);
xc2=g/(2*(1-a^2));
yc2=(sqrt(3)*g)/(2*(1-a^2));
%r2=abs((a*g)/(1-a^2));
r2=(a*g)/(1-a^2);

%Find the third circle
n=4;
a=sqrt((Pr2/Pr3)^(2/n));
g=bc2(1);
xc3=(g-2*g*a^2)/(2*(1-a^2));
yc3=(sqrt(3)*g)/(2*(1-a^2));
%r3=abs((a*g)/(1-a^2));
r3=(a*g)/(1-a^2);

%Find the intersected points of these three circles
[x1,y1,x2,y2,x3,y3]=recalculate_power(Pr1,Pr2,Pr3);
%If the circles do not overlap, then reduce the power to make the
circle bigger
Unable_locate=0;%able to be located
   if (real(x1)~=x1|real(y1)~=y1)
    k=1;
    while k<1000
    Pr1=Pr1-Pr1*0.025*k;
    [x1,y1]=recalculate_power1(Pr1,Pr2,Pr3);
      if (real(x1)~=x1|real(y1)~=y1)
         k=k+1;
         if (k>=1000|Pr1<=0)
            Unable_locate=1;%unable to be located;
            break;
         end
      else
          if
len([x1(1),y1(1)],[mc(1),mc(2)])<=len([x1(2),y1(2)],[mc(1),mc(2)])
                        c1=[x1(1),y1(1)];
             else
                        c1=[x1(2),y1(2)];
          end
      break;
      end
    end
   end

   if (real(x2)~=x2|real(y2)~=y2)
    k=0;
    while k<1000
    k=k+1;
    Pr2=Pr2-Pr2*0.025*k;
    [x2,y2]=recalculate_power2(Pr1,Pr2,Pr3);
      if (real(x2)~=x2|real(y2)~=y2)
         k=k+1;
         if (k>=1000|Pr2<=0)
            Unable_locate=1;%unable to be located;
            break;
         end
      else
         if
len([x2(1),y2(1)],[mc(1),mc(2)])<=len([x2(2),y2(2)],[mc(1),mc(2)])
                        c2=[x2(1),y2(1)];
             else
                        c2=[x2(2),y2(2)];
         end
      break;
      end
    end
   end

   if (real(x3)~=x3|real(y3)~=y3)
    k=0;
    while k<1000
    k=k+1;
    Pr3=Pr3-Pr3*0.025*k;
    [x3,y3]=recalculate_power3(Pr1,Pr2,Pr3);
      if (real(x3)~=x3|real(y3)~=y3)
         k=k+1;
         if (k>=1000|Pr3<=0)
            Unable_locate=1;%unable to be located;
            break;
         end
      else
         if
len([x3(1),y3(1)],[mc(1),mc(2)])<=len([x3(2),y3(2)],[mc(1),mc(2)])
                        c3=[x3(1),y3(1)];
            else
                        c3=[x3(2),y3(2)];
         end
      break;
      end
   end
  end
  %If the circles are overlapped, then compute axes of central three
points

 if (real(x1)==x1|real(y1)==y1)
   if len([x1(1),y1(1)],[mc(1),mc(2)])<=len([x1(2),y1(2)],[mc(1),mc(2)])
    c1=[x1(1),y1(1)];
   else
    c1=[x1(2),y1(2)];
   end
 end

 if (real(x2)==x2|real(y2)==y2)
   if len([x2(1),y2(1)],[mc(1),mc(2)])<=len([x2(2),y2(2)],[mc(1),mc(2)])
    c2=[x2(1),y2(1)];
   else
    c2=[x2(2),y2(2)];
   end
 end
 if (real(x3)==x3|real(y3)==y3)
   if len([x3(1),y3(1)],[mc(1),mc(2)])<=len([x3(2),y3(2)],[mc(1),mc(2)])
    c3=[x3(1),y3(1)];
   else
    c3=[x3(2),y3(2)];
   end
 end

 if (Unable_locate==0)
 % Find the central weight of the intersection
 %[x,y]=solve('y-y1=((y1-(y2+y3)/2)/(x1-(x2+x3)/2))*(x-x1)','y-y2=((y2-
(y1+y3)/2)/(x2-(x1+x3)/2))*(x-x2)','x','y')
            x_head(i)=1/3*c1(1)+1/3*c2(1)+1/3*c3(1);
            y_head(i)=1/3*c2(2)+1/3*c1(2)+1/3*c3(2);
            c=[x_head(i),y_head(i)];
      SE_SOA(i)=sqrt((c(1)-mc(1))^2+(c(2)-mc(2))^2);  %root  of  Square
error of SOA method
     if SE_SOA(i)<=125
      p=p+1;
        end
 % Find the mean square error between these input points and output
points
 elseif (Unable_locate==1)
      fprintf('It is out of range for mobile to be located.\n');
      q=q+1;
 end
end

MSE_SOA = sum(SE_SOA(find(SE_SOA~=0)))/length(find(SE_SOA~=0));
MSE_SOA_percent = (p*100)/(length(find(SE_SOA)~=0));
MSE_outage_percent = (q/Num_try)*100;

%The following the SOA estimate results using SOCLSFM
P=[x_head;y_head];
T=[X_axis;Y_axis];
format short;

figure(1)
hold on;
for i=1:Num_try
    s=sprintf('%d',i);
    if P(:,i)~=0
    plot( P(1,i),P(2,i),'o'),text(P(1,i)+8,P(2,i)+8,s);
    end
    plot( T(1,i),T(2,i),'*'),text(T(1,i)+8,T(2,i)+8,s);
end
plot(0,0,'ro');plot(1500,0,'ro');plot(750,750*sqrt(3),'ro');
text(0+8,0+8,'BS1');text(1500+8,0+8,'BS2');text(750+8,750*sqrt(3)+8,'BS
3');
hold off;
s=sprintf('Estimated locations, Outage is:%d\n',MSE_outage_percent);
xlabel(s);
ylabel('SOA estimated locations :o   Target locations :*');
s=sprintf('Mean Error of SOA-based scheme is %d\n',MSE_SOA);
title(s);
grid;
```

```
fprintf('=========================================================\n');
fprintf('The percentage less than 125 m after SOA-based scheme
%3.4d\n',MSE_SOA_percent);
fprintf('The outage of SOA-based scheme is %3.4d\n',MSE_outage_percent);
```

I.3 NN

The following is the main program for the first integrated NN and SOA scheme. A few of the simple functions developed in this program are not provided.

```
%######################################################################
% This program is built NN inputs for training.
% x_head, y_head: coordinates of SOA estimated PL
% mean_delay1, mean_delay2, mean_delay3: three mean delays
% impulse_response_arrary1 (16-tap): Impulse response 1
% impulse_response_arrary2 (16-tap): Impulse response 2
% impulse_response_arrary3 (16-tap): Impulse response 3

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: December 19, 2000
%######################################################################
%NN network implementation with Random inputs (the central points
%inside the circle)
%Training and testing use the same pattern of SOA-based scheme
%save the NN inputs data for the training test
clear;
close all;
R=2000;%(m)
d=50;
i_centre=find([-R:50:R]==0);
i_centre_ref=[i_centre,i_centre];
bc1=[0,0];
bc2=[1500,0];
bc3=[750,750*sqrt(3)];
Num_try = 500;
SE_SOA = zeros(1,Num_try);
%The points are chosen the central points inside the circle
%=============================================================
cofficient=rand(1,Num_try);
delta_test=linspace(0,2*pi,Num_try);
delta=rand(1,Num_try).*delta_test;
x=750+cofficient*866.*cos(delta);
y=433+cofficient*866.*sin(delta);
X_axis=x;
Y_axis=y;
save X_axis X_axis;%Network training data
save Y_axis Y_axis;%Network training data
%=============================================================
x_head=zeros(1,Num_try);
y_head=zeros(1,Num_try);
```

```
mean_delay1=zeros(1,Num_try);
mean_delay2=zeros(1,Num_try);
mean_dealy3=zeros(1,Num_try);
%For every point, the error between the estimated value and fixed value
is set to 0, p=0
%Initailize the number of error of estimate location less than 125m to
0, q=0
%Initailize the outage location estimation
for i=1:Num_try
mc=[X_axis(i),Y_axis(i)];
fprintf('********************************************\n');
fprintf('The mobile is located at: %2.4d\n',mc(1));
fprintf('                          %2.4d\n',mc(2));
fprintf('The number of selected location %2.0d\n',i);
load hwider1_new.mat;
hedB1 = calcmrp ( hwider1_new ) ;
load hwider2_new.mat;
hedB2 = calcmrp ( hwider2_new ) ;
load hwider3_new.mat;
hedB3 = calcmrp ( hwider3_new ) ;
ic=find([-R:d:R]==0);
%calcualte the mean delay
Nover_cell_width=length([-R:d:R]);
for p=1:Nover_cell_width
   for q=1:Nover_cell_width
      oneh1=hwider1_new(p,q,:);
      mean_delays_of_oneh1=mean_dealy_calc(oneh1);
      mean_delays_arrary1(p,q)= mean_delays_of_oneh1;
      impulse_response_arrary1(p,q)=oneh1;
      oneh2=hwider2_new(p,q,:);
      mean_delays_of_oneh2=mean_dealy_calc(oneh2);
      mean_delays_arrary2(p,q)= mean_delays_of_oneh2;
      impulse_response_arrary2(p,q)=oneh1;
      oneh3=hwider3_new(p,q,:);
      mean_delays_of_oneh3=mean_dealy_calc(oneh3);
      mean_delays_arrary3(p,q)= mean_delays_of_oneh3;
      impulse_response_arrary3(p,q)=oneh1;
   end
end

%For each base station
%Get an impulse reponse by interpolation
%Call them hc1,hc2,hc3
%For the base station 1
p3=[mc(1)-mod(mc(1),d),mc(2)-mod(mc(2),d)];
p1=[p3(1),p3(2)+d];
p2=[p3(1)+d,p3(2)+d];
p4=[p3(1)+d,p3(2)];
p_distance1=distance(bc1,p1);
p_distance2=distance(bc1,p2);
p_distance3=distance(bc1,p3);
p_distance4=distance(bc1,p4);
p_power1=hedB1(p_distance1(1),p_distance1(2));
p_power2=hedB1(p_distance2(1),p_distance2(2));
p_power3=hedB1(p_distance3(1),p_distance3(2));
p_power4=hedB1(p_distance4(1),p_distance4(2));
```

```
p_delay1=mean_delays_arrary1(p_distance1(1),p_distance1(2));
p_delay2=mean_delays_arrary1(p_distance2(1),p_distance2(2));
p_delay3=mean_delays_arrary1(p_distance3(1),p_distance3(2));
p_delay4=mean_delays_arrary1(p_distance4(1),p_distance4(2));
L1=len(p1,mc);
L2=len(p2,mc);
L3=len(p3,mc);
L4=len(p4,mc);
L=[L1,L2,L3,L4]';
w1=-(sqrt(2)*L1)/(2*d)+L2/(2*d)+L3/(2*d);
w2=-(sqrt(2)*L2)/(2*d)+L1/(2*d)+L4/(2*d);
w3=-(sqrt(2)*L3)/(2*d)+L1/(2*d)+L4/(2*d);
w4=-(sqrt(2)*L4)/(2*d)+L2/(2*d)+L3/(2*d);
w=[w1,w2,w3,w4];
w=sum(w);
hc1=p_power1*(w1/w)+p_power2*(w2/w)+p_power3*(w3/w)+p_power4*(w4/w);
mean_delay1(i)=p_delay1*(w1/w)+p_delay2*(w2/w)+p_delay3*(w3/w)+p_delay4
*(w4/w);

%For the base station 2
p_distance1=distance(bc2,p1);
p_distance2=distance(bc2,p2);
p_distance3=distance(bc2,p3);
p_distance4=distance(bc2,p4);
p_power1=hedB2(p_distance1(1),p_distance1(2));
p_power2=hedB2(p_distance2(1),p_distance2(2));
p_power3=hedB2(p_distance3(1),p_distance3(2));
p_power4=hedB2(p_distance4(1),p_distance4(2));
p_delay1=mean_delays_arrary2(p_distance1(1),p_distance1(2));
p_delay2=mean_delays_arrary2(p_distance2(1),p_distance2(2));
p_delay3=mean_delays_arrary2(p_distance3(1),p_distance3(2));
p_delay4=mean_delays_arrary2(p_distance4(1),p_distance4(2));
L1=len(p1,mc);
L2=len(p2,mc);
L3=len(p3,mc);
L4=len(p4,mc);
L=[L1,L2,L3,L4]';
w1=-(sqrt(2)*L1)/(2*d)+L2/(2*d)+L3/(2*d);
w2=-(sqrt(2)*L2)/(2*d)+L1/(2*d)+L4/(2*d);
w3=-(sqrt(2)*L3)/(2*d)+L1/(2*d)+L4/(2*d);
w4=-(sqrt(2)*L4)/(2*d)+L2/(2*d)+L3/(2*d);
w=[w1,w2,w3,w4];
w=sum(w);
hc2=p_power1*(w1/w)+p_power2*(w2/w)+p_power3*(w3/w)+p_power4*(w4/w);
mean_delay2(i)=p_delay1*(w1/w)+p_delay2*(w2/w)+p_delay3*(w3/w)+p_delay4
*(w4/w);

%For the base station 3
p_distance1=round(distance(bc3,p1));
p_distance2=round(distance(bc3,p2));
p_distance3=round(distance(bc3,p3));
p_distance4=round(distance(bc3,p4));
p_power1=hedB3(p_distance1(1),p_distance1(2));
p_power2=hedB3(p_distance2(1),p_distance2(2));
p_power3=hedB3(p_distance3(1),p_distance3(2));
p_power4=hedB3(p_distance4(1),p_distance4(2));
```

```
p_delay1=mean_delays_arrary3(p_distance1(1),p_distance1(2));
p_delay2=mean_delays_arrary3(p_distance2(1),p_distance2(2));
p_delay3=mean_delays_arrary3(p_distance3(1),p_distance3(2));
p_delay4=mean_delays_arrary3(p_distance4(1),p_distance4(2));
L1=len(p1,mc);
L2=len(p2,mc);
L3=len(p3,mc);
L4=len(p4,mc);
L=[L1,L2,L3,L4]';
w1=-(sqrt(2)*L1)/(2*d)+L2/(2*d)+L3/(2*d);
w2=-(sqrt(2)*L2)/(2*d)+L1/(2*d)+L4/(2*d);
w3=-(sqrt(2)*L3)/(2*d)+L1/(2*d)+L4/(2*d);
w4=-(sqrt(2)*L4)/(2*d)+L2/(2*d)+L3/(2*d);
w=[w1,w2,w3,w4];
w=sum(w);
hc3=p_power1*(w1/w)+p_power2*(w2/w)+p_power3*(w3/w)+p_power4*(w4/w);
mean_delay3(i)=p_delay1*(w1/w)+p_delay2*(w2/w)+p_delay3*(w3/w)+p_delay4
*(w4/w);

%Compute the power (Pr1,Pr2,Pr3)of hc1,hc2,hc3 not in dB
%find the power of hc1
Pr1=10^(hc1/10);

%Compute the power (Pr1,Pr2,Pr3)of hc1,hc2,hc3 not in dB
%find the power of hc2
Pr2=10^(hc2/10);

%Compute the power (Pr1,Pr2,Pr3)of hc1,hc2,hc3 not in dB
%find the power of hc3
Pr3=10^(hc3/10);

%Estimate location based on powers. assume n=4
%Find the first circle
n=4;
a=sqrt((Pr1/Pr2)^(2/n));
g=bc2(1);
xc1=g/(1-a^2);
yc1=bc1(2);
%r1=abs((a*g)/(1-a^2));
r1=(a*g)/(1-a^2);

%Find the second circle
n=4;
a=sqrt((Pr1/Pr3)^(2/n));
g=bc2(1);
xc2=g/(2*(1-a^2));
yc2=(sqrt(3)*g)/(2*(1-a^2));
%r2=abs((a*g)/(1-a^2));
r2=(a*g)/(1-a^2);

%Find the third circle
n=4;
a=sqrt((Pr2/Pr3)^(2/n));
g=bc2(1);
xc3=(g-2*g*a^2)/(2*(1-a^2));
yc3=(sqrt(3)*g)/(2*(1-a^2));
```

```
%r3=abs((a*g)/(1-a^2));
r3=(a*g)/(1-a^2);

%Find the intersected points of these three circles
[x1,y1,x2,y2,x3,y3]=recalculate_power(Pr1,Pr2,Pr3);
%If the circles do not overlap, then reduce the power to make the
circle bigger
Unable_locate=0;%able to be located
   if (real(x1)~=x1|real(y1)~=y1)
    k=1;
    while k<1000
    Pr1=Pr1-Pr1*0.025*k;
    [x1,y1]=recalculate_power1(Pr1,Pr2,Pr3);
      if (real(x1)~=x1|real(y1)~=y1)
         k=k+1;
         if (k>=1000|Pr1<=0)
            %fprintf('It is out of range for mobile to be located.\n')
            Unable_locate=1;%unable to be located;
            break;
         end
      else
          if
len([x1(1),y1(1)],[mc(1),mc(2)])<=len([x1(2),y1(2)],[mc(1),mc(2)])
                        c1=[x1(1),y1(1)];
             else
                        c1=[x1(2),y1(2)];
          end
      break;
      end
    end
   end

   if (real(x2)~=x2|real(y2)~=y2)
    k=0;
    while k<1000
    k=k+1;
    Pr2=Pr2-Pr2*0.025*k;
    [x2,y2]=recalculate_power2(Pr1,Pr2,Pr3);
      if (real(x2)~=x2|real(y2)~=y2)
         k=k+1;
         if (k>=1000|Pr2<=0)
            %fprintf('It is out of range for mobile to be located.\n')
            Unable_locate=1;%unable to be located;
            break;
         end
      else
         if
len([x2(1),y2(1)],[mc(1),mc(2)])<=len([x2(2),y2(2)],[mc(1),mc(2)])
                        c2=[x2(1),y2(1)];
             else
                        c2=[x2(2),y2(2)];
         end
      break;
      end
    end
   end
   if (real(x3)~=x3|real(y3)~=y3)
    k=0;
    while k<1000
    k=k+1;
    Pr3=Pr3-Pr3*0.025*k;
    [x3,y3]=recalculate_power3(Pr1,Pr2,Pr3);
      if (real(x3)~=x3|real(y3)~=y3)
         k=k+1;
         if (k>=1000|Pr3<=0)
            %fprintf('It is out of range for mobile to be located.\n')
            Unable_locate=1;%unable to be located;
            break;
         end
      else
         if
len([x3(1),y3(1)],[mc(1),mc(2)])<=len([x3(2),y3(2)],[mc(1),mc(2)])
                        c3=[x3(1),y3(1)];
            else
                        c3=[x3(2),y3(2)];
         end
      break;
      end
   end
  end
  %If the circles are overlapped, then compute axes of central three
points

 if (real(x1)==x1|real(y1)==y1)
   if len([x1(1),y1(1)],[mc(1),mc(2)])<=len([x1(2),y1(2)],[mc(1),mc(2)])
    c1=[x1(1),y1(1)];
   else
    c1=[x1(2),y1(2)];
   end
 end

 if (real(x2)==x2|real(y2)==y2)
   if len([x2(1),y2(1)],[mc(1),mc(2)])<=len([x2(2),y2(2)],[mc(1),mc(2)])
    c2=[x2(1),y2(1)];
   else
    c2=[x2(2),y2(2)];
   end
 end

 if (real(x3)==x3|real(y3)==y3)
   if len([x3(1),y3(1)],[mc(1),mc(2)])<=len([x3(2),y3(2)],[mc(1),mc(2)])
    c3=[x3(1),y3(1)];
   else
    c3=[x3(2),y3(2)];
   end
 end

 %if Unable_locate==1
 %   break;
 %elseif Unable_locate==0
 if (Unable_locate==0)
 % Find the central weight of the intersection
 %[x,y]=solve('y-y1=((y1-(y2+y3)/2)/(x1-(x2+x3)/2))*(x-x1)','y-y2=((y2-
(y1+y3)/2)/(x2-(x1+x3)/2))*(x-x2)','x','y')
            x_head(i)=1/3*c1(1)+1/3*c2(1)+1/3*c3(1);
            y_head(i)=1/3*c2(2)+1/3*c1(2)+1/3*c3(2);
            c=[x_head(i),y_head(i)];
      SE_SOA(i)=sqrt((c(1)-mc(1))^2+(c(2)-mc(2))^2);  %root  of  Square
error of SOA method
      if SE_SOA(i)<=125
      p = p + 1;
        end
   % Find the mean square error between these input points and output
points
 elseif (Unable_locate==1)
      fprintf('It is out of range for mobile to be located.\n');
      q = q + 1;
 end
end
MSE_SOA = sum(SE_SOA(find(SE_SOA~=0)))/length(find(SE_SOA~=0));
MSE_SOA_percent = (p*100)/(length(find(SE_SOA)~=0));
MSE_outage_percent_SOA = (q/Num_try)*100;
fprintf('==========================================================\n');
fprintf('The  percentage  less  than  125  m  after  SOA-based  scheme
%3.4d\n',MSE_SOA_percent);
fprintf('The mean square error of SOA-based scheme %3.4d\n',MSE_SOA);
fprintf('The      outage      of      SOA-based      scheme      is
3.4d\n',MSE_outage_percent_SOA);
save x_head x_head;%Network training data
save y_head y_head;%Network training data
save mean_delay1 mean_delay1;%Network training data
save mean_delay2 mean_delay2;%Network training data
save mean_delay3 mean_delay3;%Network training data
save impulse_response_arrary1;%Network training data
save impulse_response_arrary2;%Network training data
save impulse_response_arrary3;%Network training data

%######################################################################
% This program is to train the NN.
% x_head, y_head: coordinates of SOA estimated PL
% mean_delay1, mean_delay2, mean_delay3: three mean delays
% impulse_response_arrary1 (16-tap): Impulse response 1
% impulse_response_arrary2 (16-tap): Impulse response 2
% impulse_response_arrary3 (16-tap): Impulse response 3

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: December 19, 2000
%######################################################################
clear;
close all;
load x_head;%Network training data
load y_head;%Network training data
load mean_delay1;% Network training data
load mean_delay2;%Network training data
load mean_delay3;%Network training data
load impulse_response_arrary1;%Network training data
load impulse_response_arrary2;%Network training data
load impulse_response_arrary3;%Network training data
load X_axis X_axis;%Network training data
load Y_axis Y_axis;%Network training data
count =0;
Num_try =1500;
for i=1:Num_try
   while x_head(i)==0
      count = count +1;
      for k=i:Num_try-1
         x_head(k) = x_head(k+1);
         y_head(k) = y_head(k+1);
         mean_delay1(k) = mean_delay1(k+1);
         mean_delay2(k) = mean_delay2(k+1);
         mean_delay3(k) = mean_delay3(k+1);
         for i=1:Num_try
         impulse_response_arrary1(i,k)                                 =
impulse_response_arrary1(i,k+1);
         impulse_response_arrary2(i,k)= impulse_response_arrary2(i,k+1);
         impulse_response_arrary3(i,k)                                 =
impulse_response_arrary3(i,k+1);
         end
         X_axis(k) = X_axis(k+1);
         Y_axis(k) = Y_axis(k+1);
      end
    end
end

x_head = x_head(1:Num_try - count);
y_head = y_head(1:Num_try - count);
mean_delay1 = mean_delay1(1:Num_try - count);
mean_delay2 = mean_delay2(1:Num_try - count);
mean_delay3 = mean_delay3(1:Num_try - count);
for i = 1:16
impulse_response_arrary1(i)  =  impulse_response_arrary1(i,1:Num_try  -
count);
impulse_response_arrary2(i)  =  impulse_response_arrary2(i,1:Num_try  -
count);
impulse_response_arrary3(i)  =  impulse_response_arrary3(i,1:Num_try  -
count);
end
X_axis = X_axis(1:Num_try - count);
Y_axis = Y_axis(1:Num_try - count);

P=[x_head;y_head;mean_delay1;mean_delay2;mean_delay3;impulse_response_a
rrary1(1);...
impulse_response_arrary1(2);impulse_response_arrary1(3);impulse_respons
e_arrary1(4);...
impulse_response_arrary1(5);impulse_response_arrary1(6);impulse_respons
e_arrary1(7);...
impulse_response_arrary1(8);impulse_response_arrary1(9);impulse_respons
e_arrary1(10);...
impulse_response_arrary1(11);impulse_response_arrary1(12);impulse_respo
nse_arrary1(13);...
impulse_response_arrary1(14);impulse_response_arrary1(15);impulse_respo
```

```
nse_arrary1(16);...

impulse_response_arrary2(1);...
impulse_response_arrary2(2);impulse_response_arrary2(3);impulse_respons
e_arrary2(4);...
impulse_response_arrary2(5);impulse_response_arrary2(6);impulse_respons
e_arrary2(7);...
impulse_response_arrary2(8);impulse_response_arrary2(9);impulse_respons
e_arrary2(10);...
impulse_response_arrary2(11);impulse_response_arrary2(12);impulse_respo
nse_arrary2(13);...
impulse_response_arrary2(14);impulse_response_arrary2(15);impulse_respo
nse_arrary2(16);...

impulse_response_arrary3(1);...
impulse_response_arrary3(2);impulse_response_arrary3(3);impulse_respons
e_arrary3(4);...
impulse_response_arrary3(5);impulse_response_arrary3(6);impulse_respons
e_arrary3(7);...
impulse_response_arrary3(8);impulse_response_arrary3(9);impulse_respons
e_arrary3(10);...
impulse_response_arrary3(11);impulse_response_arrary3(12);impulse_respo
nse_arrary3(13);...
impulse_response_arrary3(14);impulse_response_arrary3(15);impulse_respo
nse_arrary2(16);...
]
T=[X_axis;Y_axis];
net1=newff([minmax(x_head);minmax(y_head);minmax(mean_delay1);minmax(me
an_delay2);minmax(mean_delay3);...
minmax(impulse_response_arrary1(1));
minmax(impulse_response_arrary1(2));...
minmax(impulse_response_arrary1(3));minmax(impulse_response_arrary1(4))
;minmax(impulse_response_arrary1(5));...
minmax(impulse_response_arrary1(6));minmax(impulse_response_arrary1(7))
;minmax(impulse_response_arrary1(8));...
minmax(impulse_response_arrary1(9));minmax(impulse_response_arrary1(10)
);minmax(impulse_response_arrary1(11));...
minmax(impulse_response_arrary1(12));minmax(impulse_response_arrary1(13
));minmax(impulse_response_arrary1(14));...
minmax(impulse_response_arrary1(15));minmax(impulse_response_arrary1(16
));...

minmax(impulse_response_arrary2(1));
minmax(impulse_response_arrary2(2));...
minmax(impulse_response_arrary2(3));minmax(impulse_response_arrary2(4))
;minmax(impulse_response_arrary2(5));...
minmax(impulse_response_arrary2(6));minmax(impulse_response_arrary2(7))
;minmax(impulse_response_arrary2(8));...
minmax(impulse_response_arrary2(9));minmax(impulse_response_arrary2(10)
);minmax(impulse_response_arrary2(11));...
minmax(impulse_response_arrary2(12));minmax(impulse_response_arrary2(13
));minmax(impulse_response_arrary2(14));...
minmax(impulse_response_arrary2(15));minmax(impulse_response_arrary1(16
));...

minmax(impulse_response_arrary3(1));
minmax(impulse_response_arrary3(2));...
minmax(impulse_response_arrary3(3));minmax(impulse_response_arrary3(4))
;minmax(impulse_response_arrary3(5));...
minmax(impulse_response_arrary3(6));minmax(impulse_response_arrary3(7))
;minmax(impulse_response_arrary3(8));...
minmax(impulse_response_arrary3(9));minmax(impulse_response_arrary3(10)
);minmax(impulse_response_arrary3(11));...
minmax(impulse_response_arrary3(12));minmax(impulse_response_arrary3(13
));minmax(impulse_response_arrary3(14));...
minmax(impulse_response_arrary3(15));minmax(impulse_response_arrary3(16
));...
],[20,2],{'tansig','purelin'},'trainlm');
net1.trainParam.show=20;
net1.trainParam.epochs=120;
net1.trainParam.goal=1e-10;
save net1 net1;
save P P;
save T T;
save count count;

%######################################################################
% This program is to evaluate the performance of the integrated scheme
% on SOCLSFM.

% Author: Jie Liu n925a@unb.ca
% Copyright
% Date Created: December 19, 2000
% Date Modified: December 19, 2000
%######################################################################
clear;
load net1;
load P;
load T;
load count;
Num_try=1500;
net2 = train(net1,P,T);
save net2 net2;
a= sim(net2,P);
e = T-a;
MSE_NN = mean(sqrt(e(1,:).^2+e(2,:).^2));
m=0;
for i=1:Num_try-count
    RSE_NN=sqrt(e(1,i)^2+e(2,i)^2);
    if RSE_NN<125
       m=m+1;
    end
end
MSE_NN_percent=(m/(Num_try-count))*100;
fprintf('==========================================================\n');
fprintf('Mean Error of NN test is %3.4d\n',MSE_NN)
fprintf('The percentage less than 125m after NN estimation
%3.4d\n',MSE_NN_percent);
```