DEPARTMENT OF THE AIR FORCE

AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/DS/ENG/98-11

Global Positioning System (GPS) Receiver Design For Multipaths Mitigation

DISSERTATION

El-Sayed Abdel-Salam Gadallah
Lt.Col., EGYPT

AFIT/DS/ENG/98-11

The views expressed in this dissertation are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

AFIT/DS/ENG/98-11

Global Positioning System (GPS) Receiver Design For Multipaths Mitigation

DISSERTATION

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Doctor of Philosophy

El-Sayed Abdel-Salam Gadallah, B.S.E.E., M.S.E.E.
Lt.Col., EGYPT

August, 1998

AFIT/DS/ENG/98-11

Global Positioning System (GPS) Receiver Design For Multipaths Mitigation

El-Sayed Abdel-Salam Gadallah, B.S.E.E., M.S.E.E.

Lt.Col., EGYPT

Approved:

M. Pachter — 17 Aug 98

Dr. Meir Pachter, Committee Chairman — Date

Mark E. Oxley — 17 Aug 98

Dr.Mark E. Oxley, Committee Member — Date

Stewart L. DeVilbiss — 17 Aug 98

Maj. Stewart L. DeVilbiss, Committee Member — Date

Stuart Kramer — 17 AUG 98

Lt.Col. Stuart C. Kramer, Dean's Representative — Date

Robert A. Calico, Jr
Dean

Acknowledgements

All acknowledges and thanks to ALLAH the God of the universe; Who created me, and it is Who guides me; Who gives me food and drink, And when I am ill it is Who cures me; Who will cause me to die, and then to live (again); And Who, I hope, will forgive me my faults on the Day of Judgment...

Oh my God If I disobey you, you forgive me; if I forget you, you mention me. So how I can forget you never ever forget me...

THEN,

I send my acknowledges to my country Egypt and its government that granted me this delegation for receiving this Ph.D. and I send my acknowledges to USA and AFIT for accepting my delegation. I send my acknowledges and thanks to my advisors Dr. Meir Pachter and Maj. Stewart L. DeVilbiss For their assistance, help, encourages, and advises during these work. I thanks also all the persons helped me during this work in the AFIT, in the AFIT library and in computer department. I thank Mrs Anette Robb the military international training chief and all persons in IMSO for their assistance and help. I thank my family, starting with my mother the source of save and peace in my life who permanently give me the support, the encouragement, and the praying to ALLAH for me. I thank all my brothers and my sisters for their encourages, their attention and worry about me during this work. I have to thank my wife who overloaded with much more because of me during these long days, I thank her for the patience and the encouragement. Then I ask my kids for praying to ALLAH for me. I have paid this effort because of them and because I love them.

El-Sayed Abdel-Salam Gadallah

Table of Contents

List of Figures

List of Tables

Abstract

The presence of signal reflections, termed Multipath is an inevitable source of error degrading the accuracy of DGPS signal processing. Our approach to multipath mitigation includes three necessary correlated parts: a multiple observer of the GPS received signal, an estimator unit, and a modified tracking loop. The statistical model of the multipath is differs for each possible user location and, in addition are time varying. Consequently there is no unified statistical model for the multipath signal. Therefore the solution of multipath problem using statistical models is difficult. This research introduces a new estimator that can detect the presence of multipath, can determine the unknown number of multipath components, and can estimate multipath parameters received by a GPS antenna (the time delay α, and the attenuation coefficient a). The multipath signals parameters are estimated on an epoch by epoch basis at any instant of observation. The new estimator is based on the maximum likelihood estimation applied to multiple simultaneous observations of a linear model (regression form) of the received signal. The estimator is based on a recursive deployment process of the estimated multipath time delay, α, so we called it an "α-deploying method" The observing sensors might be a correlator, in code tracking, or a multiplier, in carrier tracking. An improvement is achieved to the accuracy of multipath estimates at low signal-to-noise ratio, SNR by applying Kalman filtering as a cascaded estimator. The instantaneous observation of multipath parameters gives a high probability of tracking a dynamic multipath signal caused by environmental changes (satellites and receiver motions). A Kalman filter is presented for a stationary receiver or low speed mobile receiver. The 'α-deploying method is a good technique to estimate and to mitigate the severe effects of multipaths corrupting the received signal, and, in turn, it increases the accuracy of ranging in code tracking and ambiguity resolution in carrier phase tracking. The method also can be used as a tool to scope the multipath in the environmental area around the user, and may be a significant indicator in surveying and determining the region of high multipath.

This dissertation also includes design of new modified tracking loops,specifically, a modified Phase-Lock Loop (PLL) for carrier tracking and a Delay-Locked Loop (DLL) for code

tracking. These modified tracking loops are endowed with the α-deploying estimator. In the presence of multipath the standard PLL and DLL cannot distinguish between the direct path and the reflected signals and continue to employ a false tracking. The dynamic performance of the PLL to track the received carrier is dependent on the basic characteristic of a phase detector with positive slope crossing the zero point, namely the equilibrium point. Multipath forces the equilibrium point to move such that the local carrier signal is aligned with the received direct carrier signal.

A modified DLL can track the direct path signal code in the presence of multipath. The modified DLL is configured with a multicorrelator which is weighted by the estimated multipath components parameters. In this dissertation the investigation and analysis of the designed loop is performed. Simulation of the standard and modified tracking loops are presented in the cases of absence and presence of multipath. Tracking and performance in noise are investigated and future work is suggested.

Global Positioning System (GPS) Receiver Design For Multipaths Mitigation

I. Introduction to Multipath

1.1 What is 'Multipath'?

Reflections of a GPS satellite transmitted signal, termed multipath, are generated by reflectors such as the earth's surface, buildings, mountains, and other objects. Multipath is typically the dominant source of error in differential GPS (DGPS) positioning. Multipath induced ranging errors are normally uncorrelated between antenna locations [3]; hence, the differencing technique does not reduce multipath induced errors. Multipath introduces ranging errors by moving the equilibrium point of the tracking loops in the receiver [28]. The tracking loops in the GPS receivers are the code tracking loop and the carrier tracking loop. Erroneous estimates of the direct path or Line Of Sight (LOS) signal parameters lead to erroneous measurements of the satellite pseudorange and the carrier phase observable. Many approaches have been developed to mitigate the influence of multipath such as techniques which isolate the antenna from its surroundings [18], calibrating the environment of a static reference station [7,11], the use multiple receivers and/or antennas [4,21], and methods that modify the tracking loops of the receivers [6, 14, 29, 30]. This work is oriented toward the last approach, so it includes a GPS signal and receiver tracking analysis. This dissertation contains five proposals for mitigation of multipath induced error in carrier and code tracking within the GPS receiver.

1.2 Problem Statement

The accuracy of the range measurement the GPS based on the measurement of the time delay between the transmitted Pseudo-Random Noise (PRN) code and the received PRN code. The function of the code tracking loop is to measure this time delay. The accuracy of

the code tracking based on the cross correlation process between the received code and the local replica, in turn it is based on the chip period and the code waveform. If the received code waveform is different from the local code the accuracy of tracking degrades and an amount of bias is added to the actual tracking. There are several unwanted signal affects the waveform of the transimitted code during the downlink from the satellite into the users which are the noise, the multipath and the jamming. The noise and the jamming can be mitigated by the spread spectrum technique while the multipath can not be mitigated. Other sources induces error in the GPS such as clock bias error, ionospheric error, ephemeris error which can be removed by differencing technique as in the DGPS, but the multipath strugle against this differencing technique and still induces a severe error into the GPS ranging. Also if the carrier phase observable is employed in the presence of multipath then the tracking loop will track the received carrier signal which is a compose of the direct signal and other multpath images delay from this signal delayed. In this situation the tracking of the received signal is a false tracking of the direct path signal and no clue grauantees that tracking is not true. Commonly, the basis of the GPS signal is included in the C/A code. The chip period of the C/A code is approximately $T_c \approx 0.98$ μsec corresponding to 29.4 Km. The tracking loops looks for the time delay through $1.5T_c$, thus, the minimum accuracy of GPS range can be determined until one chip period viz, until 29.4 Km in the rang measurement. The multipath is considered as the superposition of the direct path received signal and other reflectors have the same waveform of the received code and delayed from the direct path with time delay until $1.5T_c$. In the presence of the multipath the standard tracking loop is only able to produce the weighted average of such time delays. The erroneous measure of such time delay may reach one chip period which is corresponding to 29.4 Km. So one can imagine the effect of the multipath on the accuracy of the range measurement by the GPS.

1.3 Previous Work

The mitigation approaches to multipath can be divided into three categories as follows:

1.3.1 Pre-receiver. There have been techniques developed to remove the multipath signal by isolating the antenna from its surroundings. In Tranquilla et. al., (1994 [18]),

this was achieved by using an antenna ground plane and choke ring. An empirical method of temporal modeling of multipath by calibrating the environment of static location of the receiver was presented in Ahmed El-Rabbany, (1995 [7]). Since the satellite-reflector-antenna geometry repeats every sidereal day under the same environments, he discovered that the presence of multipath can be verified using a day-to-day correlation. Once the multipath errors are obtained, they can be used to generate a series of auto-covariance functions which describe the temporal characteristics of the expected multipath error.

1.3.2 Post Receiver. There exist techniques for reducing code and carrier-phase multipath error which exploit multiple receivers and/or antennas. J. Raquet and G. Lachapelle, (1996 [21]) used multiple receivers to estimate multipath for each individual receiver/satellite pair on an epoch-to-epoch basis. A receiver/satellite double difference of measurements from receivers in very close proximity to each other was used. The task of isolating multipath and noise was assigned to specific measurements. The authors claim that the assumptions made in this method, that the multipath error distributions are Gaussian, identical, and uncorrelated between all receivers and all satellites, are not true in every situation, but are reasonable for obtaining a rough estimate of how well the method could perform. A technique to estimate the spectral parameters of multipath using the signal to noise ratio, (SNR), has been introduced by Axelrad et. al., (1996 [1]) and Christopher et. al., (1996 [4]). This method employs four antennas (one is designated as the master antenna to control the carrier phase tracking loop and up to three slave antennas to measure the differential phase). The performance is improved with increasing number of slave antennas yet there is negligible benefits from using more than 3 slave antennas. For static application, the effectiveness of the technique is limited by the capability of an adaptive notch filter to estimate multipath caused by ground reflections. This method also requires that the antenna gain patterns be known. This technique is limited to special applications wherein the use of four distributed antennas is feasible and cost effective.

1.3.3 Receiver Internal. Code phase tracking has been modified to mitigate the multipath effect, as reported in references [6, 14, 29, 30]. A. J. Van Dierendonck, (1992 [30]),

reduces multipath interference (C/A code only) by using a narrower spacing (the standard 1.0 chip is replaced by a 0.1chip) of the correlators in the Delay Locked Loop (DLL). Since the C/A code tracking errors are on the order of a decimeter, the multipath signals tracking errors can be reduced to a few meters. The 0.1 chip correlator spacing is not practical or used for the higher chip rate P-code.

The Multipath Estimating Delay Locked Loop (MEDLL) by Bryan R. Twonsend, (1995 [29]), employs multiple correlators with variable spacing to improve the tracking performance in the DLL. The received signal is correlated with many different delayed code replicas provided by the code tracking loop. The outputs from the correlators are used to estimates the gain and the phase of the direct-path and reflected multipath signals which, in turn, feed again into a standard non-coherent DLL. The literature gives no implementation details, as this is a company proprietary technique.

A modified Rake DLL (MRDLL) was designed by Mark C. Laxton, (1996 [14]), and it is an adaptation of the Rake Delay Locked Loop (RDLL) of Sheen and Stuber, (1995 [25]), which was designed for a different application. Lionel Garin et. al., (1996 [9]), has developed a technique known as the strobe and edge correlator which mitigates multipath for code tracking. This technique has a hardware structure of only one extra correlator for better code tracking or three extra correlators for enhanced strobe correlation. The idea behind this technique is to distinguish between the direct path (LOS) signal and the multipath signal. Instead of allowing the correlation of the incoming signal to be superimposed by the LOS signal and the multipath, the tracking is (as is usual with two correlators close to the peak) with a spacing close enough to hit only the slopes of the direct signal correlation curve. To achieve this, they use another type of reference signal which reduces the range for which the resultant correlation function (which is still a triangle) is not zero. Details of this design are proprietary. In this dissertation we employ multiple observations to estimate the multipath parameters.

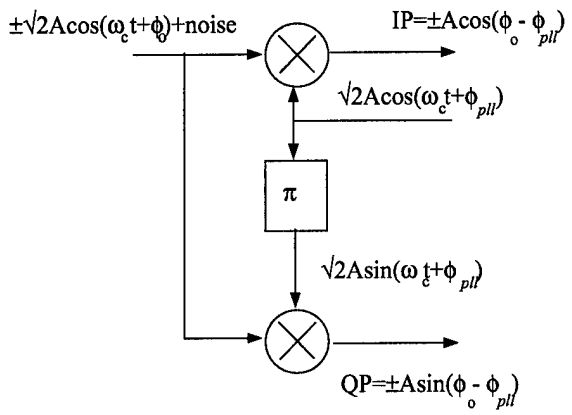


Figure 1 In-phase and In-quadrature Channel

1.4 Tracking Loops in GPS Receivers

The GPS receiver must accurately track the GPS signals received at low signal levels, usually below the thermal noise level in the receiver [27], as well as in the presence of multipath. In addition, the receiver must be able to maintain tracking in the dynamic environment created by the combined effect of satellite and receiver relative motion. For each given satellite being tracked and each of the carrier frequencies (L1, L2), two cooperating tracking loops are used inside a GPS receiver, the code and the carrier tracking loops. The data modulation technique used is a biphase shift keying (BPSK) [28]. The transmitted waveform characteristics are known to the receiver designer; it is convenient to use a phase locked loop (PLL) [27] to track the carrier signal and a delay locked loop (DLL) [28] to track the code, because the PLL and DLL yield Maximum Likelihood (ML) estimates of the carrier and code phase in the absence of multipath. Basically, two main components are required to accomplish tracking of both the carrier and the code, respectively. The first is a mixer or multiplier (see Figure 1), which multiplies the received signal by a locally generated replica of carrier; the result is either the in-phase, (IP), or quadrature phase, (QP), component of the received signal. ϕ_{pll} denotes the PLL estimate of the incoming carrier phase generated proportional to the error phase difference between the actual phase of the incoming signal and the local phase replica of the voltage controlled oscillator (VCO) output, (details will be given later). The second component is a correlator (see Figure 2), which correlates the received code with a delayed code replica.

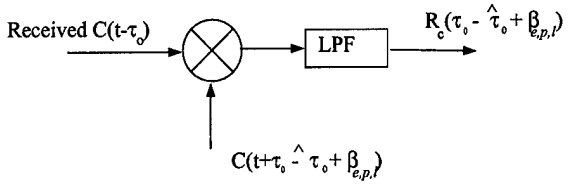


Figure 2 The Code Correlator

In the absence of multipath, the received signal is modeled as the composite of the code, the data, and the carrier. The code and the carrier tracking loops are interdependent; the code tracking loop requires an estimate of the carrier phase to enable estimation of the code phase, and the carrier tracking loop requires an estimate of the code phase to enable estimation of the carrier phase. To accomplish tracking the carrier mixer must be followed by a phase locked loop, and the code correlator must be followed by a Delay locked loop. Figure 3 shows this classical GPS tracking loop architecture and illustrates the interdependence of code and carrier tracking.

The code tracking process illustrated includes both a coherent DLL and a non-coherent DLL (only one of which is used in a given design). A coherent DLL tracks the code following carrier recovery from the PLL . The correlator outputs of the coherent DLL generates a tracking error proportional to the difference between the received signal and replicas of one half chip early and one half chip late code signals. This code tracking error is fed to a loop filter followed by a Voltage Controlled Code (VCC) which, in turn, generates the early and late codes. When the steady state of this closed loop is reached, the code tracking error becomes zero, and the center of the code replicas (defines the punctual code) is aligned with the received code. The inter-dependency between the carrier phase tracking loop and the code phase tracking loop is that the accurate carrier recovery of received signal requires accurate code tracking and vice-versa. Further details of the phase tracking loop will be given later. Unlike the coherent DLL, the non-coherent DLL does not require a pure carrier signal aligned with the received carrier. Detail of the operation of this loop are given in [10,27].

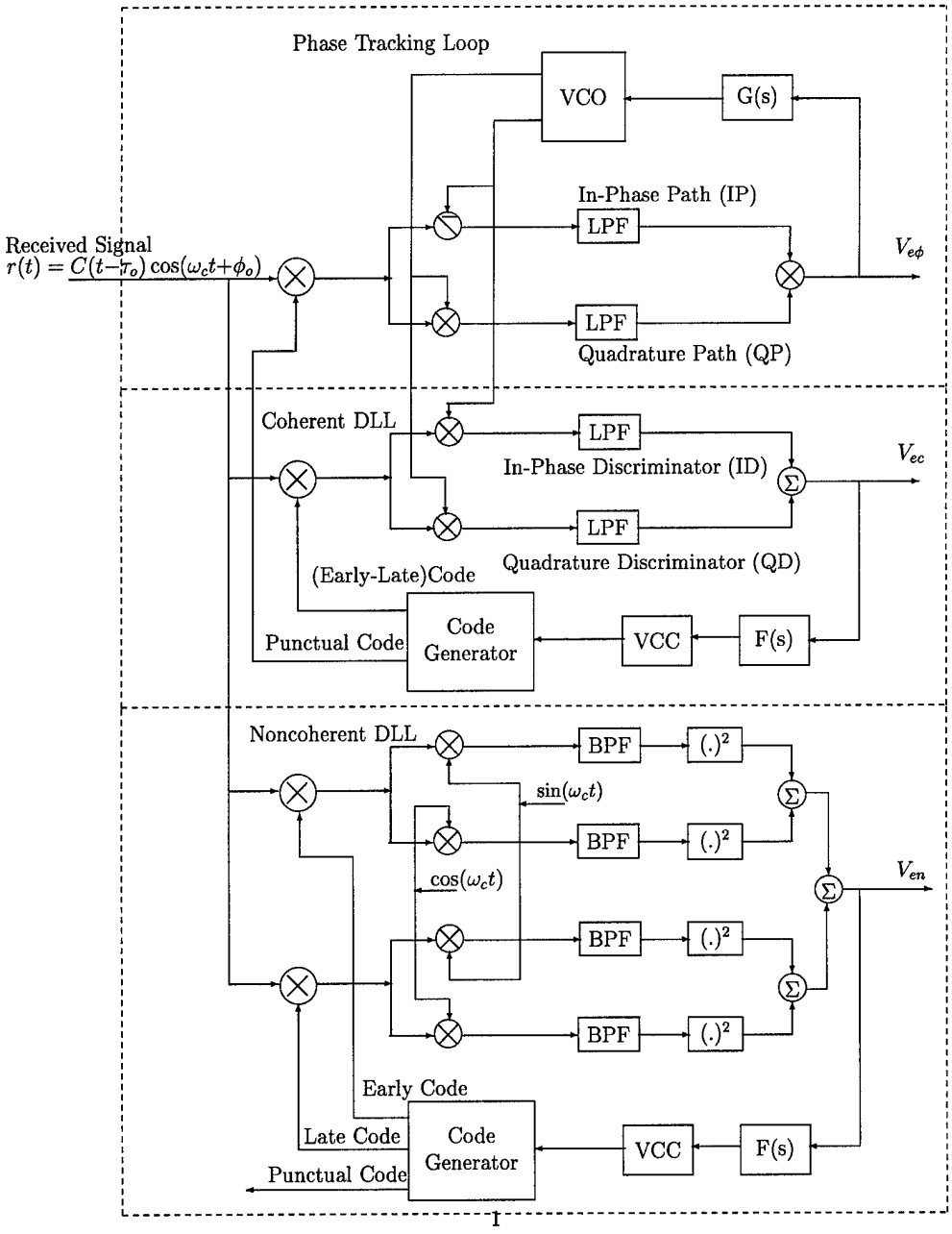


Figure 3 The classical architecture of GPS tracking loop

II. Base Band Delay Lock Loop (BBDLL)

2.1 Overview

Actually, the GPS receivers use the performance of Spread Spectrum (SS) techniques, including the acquisition and tracking of the transmitted pseudo-random noise (PRN) code. The accuracy of tracking plays an important role in supporting the acquisition and the ranging process of the GPS receivers. Furthermore, the optimum design of the overall receiver is achieved by the true assessment of the performance of the PRN code tracking loop [26]. There are several classes of the PRN tracking loops including the coherent and non-coherent categories [20,26]. We will concentrate in this chapter on the coherent DLL, or namely, Base Band DLL (BBDLL), because it is easier to be modified so as to mitigate the multipath as it will be seen in chapter VI. Analysis of the Base Band DLL in the absence of multipath is presented. The first section emphasis on the concepts of optimum tracking signals, and the second one illustrates the theory of ML estimator on the DLL. The next sections comprise the BBDLL configuration, characteristics, and noise performance. The BBDLL linear equivalent circuit including the dynamics and tracking are discussed. This chapter also presents the results of a computer simulation to verify the BBDLL theoretical and actual loop performance in the absence of multipath and in the presence of the multipath with and without noise.

2.2 Concept of Optimum Tracking Signal

The optimal closed-tracking devices is devoted to estimate the time delay, τ_o of any signal $r(t)$. For small values of delay error, the loop provides the maximum likelihood estimate of the delay in the presence of additive white Gaussian noise (AWGN) [27]. The function of the closed-tracking loop, is to multiply the received signal plus noise $r(t-\tau_o)+n(t)$ with a reference waveform $r'(t-\hat{\tau}_o)$ where $r'(t)$ is the derivative of r and $\hat{\tau}_o$ is the delay estimate, then the result of the multiplication is filtered by a Low-Pass filter (LPF). The purpose of this multiplication and the following LPF is to establish a linear operator for the difference between the time delay of the incoming signal and local signal, that is $(\tau_o-\hat{\tau}_o)$ and the purpose of the derivative also is to verify the dynamic principle of the loop into this

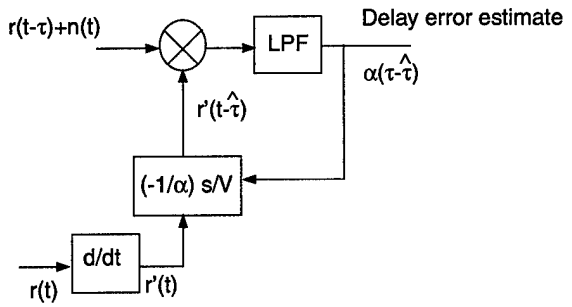


Figure 4 Tracking Loop of Any Incoming Wide-band Signal.

difference similar to a servo system, viz change the static difference into a dynamic change drives the local code to be aligned with the received code. The phase-locked loop illustrates this principle where the received signal is a pure sine wave, namely $r(t) = \sin(\omega_o t + \tau_o)$ when correlated with its derivative $r'(t) = \omega \cos(\omega_o t + \hat{\tau}_o)$ yielding a constant value proportional to $(\tau_o - \hat{\tau}_o)$, which will be discussed in detail later. Figure 4 shows the tracking loop of any incoming wide-band signal, for more details in the concept of optimum tracking signals, see-e.g., references [2,20,27].

2.3 Theory of Maximum Likelihood (ML) in the DLL

To present a demonstration of the concept of ML estimate as an optimal estimator of the delay error τ_o, consider the observation model of the received signal $r(t)$. Suppose the received signal has been recovered through the in-phase channel (IP) or the quadrature channel (QP). The received signal being observed, $\tilde{r}_I(t)$ of the IP-channel is

$$\tilde{r}_I(t) = \sqrt{P} C\,(t - \tau_o) + n(t) \tag{1}$$

where P is the input power signal (in Watts), $C\,(t - \tau_o)$ is the recovered DS/SS PRN code with T_c chip period (in seconds), $n(t)$ is zero-mean Gaussian noise with flat power spectral density (PSD) $\frac{N_o}{2}$ (W/Hz) in the frequency range $[-W_c, W_c]$ Hz and zero otherwise, and W_c is the signal bandwidth. At each instant t_i, the conditional probability density function (pdf) for $\tilde{r}_I(t_i)$ given τ_o can be written as a normal distribution with mean $C\,(t_i - \tau_o)$ and

standard deviation $\sigma > 0$.

$$f(\tilde{r}_I(t_i)/\tau_o) = \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{1}{2\sigma^2}\left(\tilde{r}_I(t_i) - C\left(t_i - \tau_o\right)\right)^2\} \tag{2}$$

where σ is the standard deviation of the noise $n(t)$. For N observations through the period time T_i, the pdf can be rewritten as [32].

$$f(\tilde{r}_I(t_i)/\tau_o) = \prod_0^N \frac{1}{\sqrt{2\pi}\sigma} \exp\{-\frac{1}{2\sigma^2}\left(\tilde{r}_I(t_i) - C\left(t_i - \tau_o\right)\right)^2\} \tag{3}$$

Now, the maximum likelihood function can be considered as the argument of the pdf of Equation (3). Denote the likelihood function by $\mathcal{L}(\tau_o)$. After expanding the square and excluding the constant terms in Equation (3) $\mathcal{L}(\tau_o)$ can be written as

$$\mathcal{L}(\tau_o) = \int_0^T \tilde{r}_I(t) C\left(t - \tau_o\right) dt \tag{4}$$

The ML estimator $\hat{\tau}_o$ is defined as the value of τ_o that maximizes the likelihood function $\mathcal{L}(\tau_o)$. Therefore, the ML estimator can be deduced from the first derivative to $\mathcal{L}(\tau_o)$ with respect to τ_o and setting the results to zero. If $\tilde{r}_I(t) C\left(t - \tau_o\right)$ is continuously durable in τ_o and $\frac{d\tilde{r}_I(t)C(t-\tau_o)}{d\tau_o}$ is Riemann integrable, then by Fundamental Theorem of Calculus the derivative of Equation 4 is

$$\frac{d\mathcal{L}(\tau_o)}{d\tau_o} = \int_o^T \tilde{r}_I(t) \frac{\partial C\left(t - \tau_o\right)}{\partial \tau_o} dt \tag{5}$$

(Weill 1995) introduces a good interpretation of the term $\frac{\partial C(t-\tau_o)}{\partial \tau_o}$, by approximating this term by $c(t+eT_c) - c(t-eT_c)$, "what he called differential code", which is a rectangular pulse of width $(2eT_c)$, centered at each chip transition and having the same polarity of $\frac{d\mathcal{L}(\tau_o)}{d\tau_o}$ at that point. Figure 5 illustrates the equivalence between the differential code with the discriminator used in the DLL.

Clearly, the spacing parameter $(2eT_c)$ specifies the performance of the ML estimator. Furthermore the differential code is easy to be generated digitally. In practice, the processing of the cross-correlation between the differential code and the incoming signal is embodied in

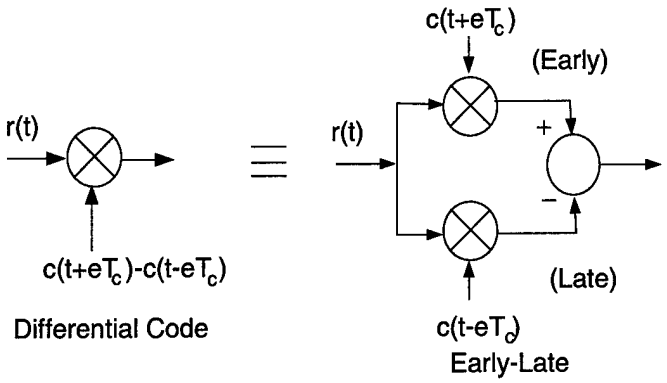


Figure 5 The Equivalence Between The Discriminator used in DLL and Differential Code

the autocorrelation function, $R_c(\Omega)$, which is defined as [20]

$$R_c(\Omega) = \frac{1}{NT_c}\int_0^{NT_c} C(t)C(t+\Omega T_c)dt \tag{6}$$

then, the correlation function in terms of the number of chips being correlated can be written as

$$R_c(\Omega) = \begin{cases} 1-\Omega(1+\frac{1}{N}) & 0 \le \Omega \le 1 \\ -\frac{1}{N} & 1 < \Omega < (N-1) \\ [\Omega-(N-1)]\left(1+\frac{1}{N}\right)-\frac{1}{N} & (N-1) \le \Omega < N \end{cases} \tag{7}$$

An approximate correlation function is given by [14, 22]

$$R_c(\Omega) \approx \begin{cases} 1-|\Omega| & \text{if } |\Omega| \le 1 \\ 0 & \text{elsewhere} \end{cases} \tag{8}$$

and Figure 6 shows its plot.

2.4 BBDLL Analysis With No Multipath

The block diagram of the tracking loop shown in Figure 7, consists of a phase discriminator, a loop filter, and a Voltage Controlled Clock (VCC).

2.4.1 The Phase Discriminator.

The input signal is the received signal defined in Equation (1).

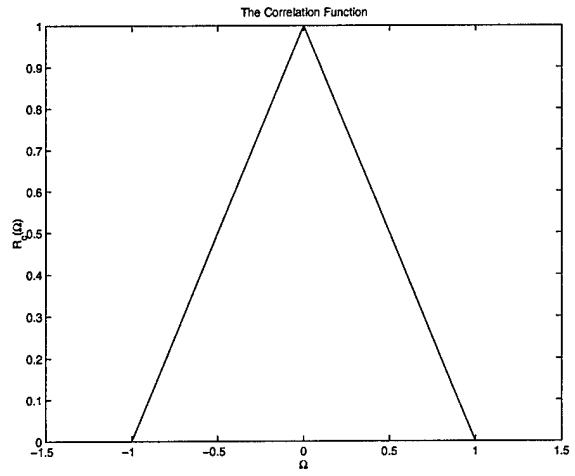


Figure 6 The Triangle Shape of the Correlation Function.

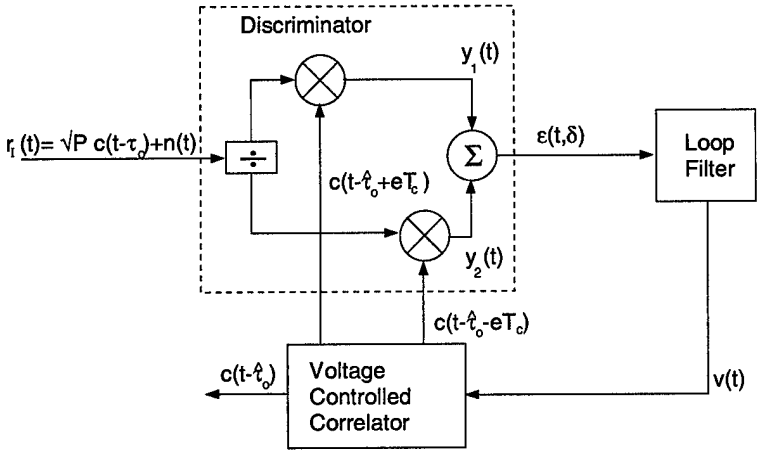


Figure 7 BBDLL.

The early-late correlator output is given by multiplying the received signal by the early and late replicas $C\,(t-\hat{\tau}_o+eT_c)$ and $C\,(t-\hat{\tau}_o-eT_c)$ respectively. The amount of the delay between these two local replicas is $2eT_c$. The early correlator output is

$$y_1(t,\tau_o,\hat{\tau}_o) = \sqrt{\frac{P}{2}} C\,(t-\tau_o)\,C\,(t-\hat{\tau}_o+eT_c) \tag{9}$$

and the late correlator output is

$$y_2(t,\tau_o,\hat{\tau}_o) = \sqrt{\frac{P}{2}} C\,(t-\tau_o)\,C\,(t-\hat{\tau}_o-eT_c)\,. \tag{10}$$

The discriminator output is the dc component of the difference between $y_1(t)$ and $y_2(t)$. The dc component is the time averaging of this difference in period of time, NT_c which is chosen such that $NT_c \gg T_c$ and N is an integer number represent the number of chips in the PN code period of integration. For C/A code the number of chips per code period length T_c is $N = 1023$ chips which corresponds to $(1msec)$. In this case as long as $N \gg 1$ the accuracy of the approximated correlator Equation 8 to the correlator Equation 7 is verified. So in the noise free case the discriminator output is given by

$$\epsilon(t,\tau_o,\hat{\tau}_o) = \frac{1}{NT_c}\int_0^{NT_c} y_2(t,\tau_o,\hat{\tau}_o) - y_1(t,\tau_o,\hat{\tau}_o)dt \tag{11}$$

In the presence of AWGN, the discriminator output $\epsilon(t,\tau_o,\hat{\tau}_o)$ can be written as

$$\epsilon\,(t,\tau_o,\hat{\tau}_o) = \sqrt{\frac{P}{2}} S_e(\delta) + \frac{1}{\sqrt{P}} n_\epsilon(t) \tag{12}$$

where $\delta = \frac{\tau_o-\hat{\tau}_o}{T_c}$ and $S_e(\delta)$ is given by

$$S_e(\delta) = R_c(\delta-e) - R_c(\delta+e) \tag{13}$$

where R_c is the autocorrelation function defined in Equation (8), and $n_\epsilon(t)$ is the noise process at the discriminator output. Equation (13) is named the S-curve of the DLL

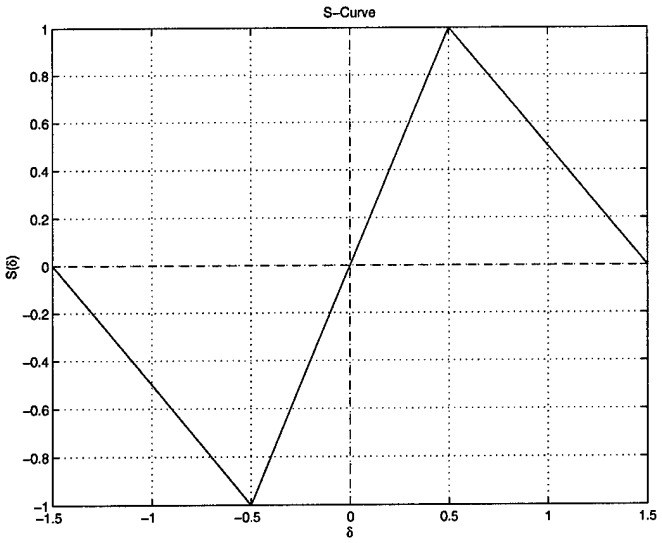


Figure 8 The BBDLL S-Curve in Deterministic case.

which represents the ideal characteristic of the DLL discriminator. Figure 8 shows the plot of the S-curve in noise free case. The noise process $n_\epsilon(t)$ can be written as

$$n_\epsilon(t) = n(t)\left[C\left(t-\hat{\tau}_o-eT_c\right)-C\left(t-\hat{\tau}_o+eT_c\right)\right] \tag{14}$$

The noise performance in the discriminator is evaluated by the calculation of PSD function. Subsequently, the autocorrelation function of $n_\epsilon(t)$ is defined as

$$R_{n\epsilon}(\tau) = E\{n_\epsilon(t)n_\epsilon(t+\tau)\} \tag{15}$$

Considering that $\hat{\tau}_o$ and eT_c are fixed, the random process $n_\epsilon(t)$ is not a wide-sense stationary process [20], however it is stationary if $\hat{\tau}_o$ is interpreted as a random variable [20]. Combining Equation (14) and (15) yields

$$\begin{aligned} R_{n_\epsilon}(\tau) \;&= E\{n(t)\left[C\left(t-\hat{\tau}_o-eT_c\right)-C\left(t-\hat{\tau}_o+eT_c\right)\right] \\ &\quad \times n(t+\tau)\left[C\left(t+\tau-\hat{\tau}_o-eT_c\right)-C\left(t+\tau-\hat{\tau}_o+eT_c\right)\right]\} \end{aligned} \tag{16}$$

The white noise $n(t)$ is independent of the spreading code C(t), so that the expected value can be written as

$$\begin{aligned} R_{n_\epsilon}(\tau) &= E\{n(t)n(t+\tau)\}E\{[C\,(t-\hat{\tau}_o-eT_c)-C\,(t-\hat{\tau}_o+eT_c)] \\ &\times [C\,(t+\tau-\hat{\tau}_o-eT_c)-C\,(t-\hat{\tau}_o+eT_c)]\} \end{aligned} \tag{17}$$

The autocorrelation function of the noise is a Dirac delta (generalized) function, that is

$$E\{n(t)n(t+\tau)\} = \frac{N_o}{2}\delta(\tau) \tag{18}$$

Substituting (18) into (17) and setting $\tau = 0$, since $\delta(\tau)$ is zero for all $\tau \neq 0$, results in [20]

$$\begin{aligned} R_{n_\epsilon}(\tau) = \; & \tfrac{N_o}{2}\delta(\tau)\,[E\{C^2\,(t-\hat{\tau}_o-eT_c)\}-2E\{C\,(t-\hat{\tau}_o-eT_c)\,C\,(t-\hat{\tau}_o+eT_c)\} \\ & +E\{C^2\,(t-\hat{\tau}_o+eT_c)\}] \end{aligned} \tag{19}$$

Since the PN code take only values of ± 1, then $C^2(t) = 1.0$. The correlation function $R_{n_\epsilon}(\tau)$ can be simplified to

$$R_{n_\epsilon}(\tau) = \frac{N_o}{2}\,[2 - 2E\{C\,(t-\hat{\tau}_o-eT_c)\,C\,(t-\hat{\tau}_o+eT_c)\}] \tag{20}$$

by results in [20] Equation (20) can be reduced to

$$R_{n_\epsilon}(\tau) = \begin{cases} N_o\delta(\tau)\left(1+\frac{1}{N}\right) & \text{for } e \geq 0.5 \\ N_o\delta(\tau)2e\left(1+\frac{1}{N}\right) & \text{for } e \leq 0.5 \end{cases} \tag{21}$$

The two-sided PSD of n_ϵ is the Fourier transform of $R_{n_\epsilon}(\tau)$, denoted by $G_{n\epsilon}(t)$,

$$G_{n\epsilon}(f) = \begin{cases} N_o\left(1+\frac{1}{N}\right) & \text{for } e \geq 0.5 \\ N_o 2e\left(1+\frac{1}{N}\right) & \text{for } e \leq 0.5 \end{cases} \tag{22}$$

2.4.2 Voltage Controlled Clock (VCC). The VCC is the element that derives the dynamics of the tracking in DLL, thus the input of the VCC is the error tracking voltage

derived from the loop filter and the outputs are two replicas, the retarded and the delayed codes which are shifted by an amount proportional to the integral of this tracking error voltage. The VCC consists of VCO and a spreading code generator which produces the code replica: the retarded, the delayed and the punctual. The mathematical representation of the VCC can be written as [20]

$$\frac{\hat{\tau}_o(t)}{T_c} = g_c \int_0^t v(\lambda) d\lambda \tag{23}$$

where $v(t)$ is the input voltage derived from the loop filter and g_c is the Voltage-Controlled Oscillator (VCO) gain in Hz/V. The Laplace transform of Equation 23 yields

$$\frac{\hat{\tau}_o}{T_c} = \frac{g_c V_c}{s} \tag{24}$$

2.4.3 Loop Filter. The tracking loop filter is necessary for the design of the damping factor and the natural frequency of the closed tracking loop, in addition, it is helpful for rejecting the high frequency components of the noise. The commonly used form of the loop filter is the lead-lag filter, perhaps passive or active according to the design requirement. The simple lead-lag transfer function of the loop filter is

$$F(s) = \frac{1+s\tau_2}{s\tau_1} \tag{25}$$

where τ_1 and τ_2 are the designed parameters of the loop filter.

2.4.4 The Linear Equivalent Circuit. The linear model of the BBDLL is valid for the small normalized phase error $\delta(t) = \frac{\tau_o - \hat{\tau}_o}{T_C}$ [20]. The equivalent linear model of the discriminator can be taken in the interval $-0.5T_c \leq \delta \leq 0.5T_c$ (see Equation 12 and Figure 8). Thus the discriminator reduces to the form

$$\epsilon(t, \delta) = K_d \delta \tag{26}$$

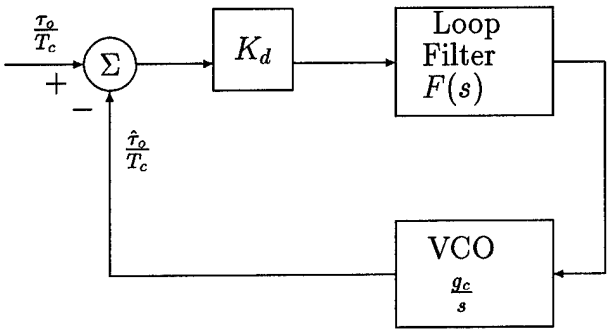


Figure 9 The Linear Equivalent Circuit

where $K_d = 2\sqrt{\frac{P}{2}}(1+\frac{1}{N})$. The linear equivalent circuit for the BBDLL can be depicted as in [20] (see Figure 9).

The classical closed-loop servomechanism transfer function $H_{dll}(s)$ is given by

$$H_{dll}(s) = \frac{\hat{\tau}_o(s)}{\tau_o(s)} = \frac{K_d g_c F(s)}{s+K_d g_c F(s)} \tag{27}$$

where $F(s)$ is the loop filter transfer function. Using the transfer function of the loop filter in Equation 25, then, the closed-loop transfer function of Equation (27) becomes

$$H_{dll}(s) = \frac{[K_d g_c(\tau_2/\tau_1)]s+K_d g_c/\tau_1}{s^2+[K_d g_c(\tau_2/\tau_1)]s+K_d g_c/\tau_1} \tag{28}$$

Equation 28 can be written in the normalized form as

$$H_{dll}(s) = \frac{2\xi\omega_n s+\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2} \tag{29}$$

where the natural frequency, ω_n and the damping factor, ξ are given by

$$\begin{aligned} \omega_n &= \sqrt{\frac{K_d g_c}{\tau_1}} \\ \xi &= \frac{\tau_2}{2}\omega_n \end{aligned} \tag{30}$$

The power spectrum of the tracking jitter is given by [20]

$$G_\delta(f) = |H_{dll}(j2\pi f)|^2 G_{n_e}(f) \tag{31}$$

The two-sided noise bandwidth is given by

$$W_L = \int_{-\infty}^{\infty} |H_{dll}(j2\pi f)|^2 df \tag{32}$$

The variance of δ, σ_δ^2, can be derived as [20]

$$\sigma_\delta^2 = \begin{cases} \frac{1}{2}\left(\frac{1}{k_d}\right)^2 N_o \left(1+\frac{1}{N}\right) W_L & \text{for } e \geq 0.5 \\ \frac{1}{2}\left(\frac{1}{k_d}\right)^2 N_o 2e \left(1+\frac{1}{N}\right) W_L & \text{for } e \leq 0.5 \end{cases} \tag{33}$$

and the loop signal-to-noise ratio, ρ_L is defined as

$$\rho_L = \frac{2P}{N_o W_L} \tag{34}$$

The two-sided noise bandwidth for a second-order loop with the loop filter defined in Equation (25) is given by

$$W_L = \omega_n \left(\xi + \frac{1}{4\xi}\right) \tag{35}$$

So far, theoretically, the BBDLL analysis is completed. An implementation of simulation model is about to be presented in the next section to verify these theoretical approaches and introducing a comparative study to investigate an actual noise performance of such a loop in the presence and in the absence of the multipath effects. The simulated results will be useful to validate the design of modified DLL (n-MRDLL) as it will be explained in details in ChapterVI.

2.5 BBDLL Computer Simulation

The simulation of the BBDLL is performed to verify the theoretical results as presented previously in this chapter. Basically the main component needs a stress investigation in the BBDLL, is the discriminator because its characteristic determines the performance of the

DLL. Typically the discriminator characteristic is the S-curve in the ideal case (see Figure 8). The successful performance of the DLL is determined by keeping the crossing of S-curve through the zero point, (equilibrium point). Other dynamical performance of the BBDLL can be designed specifying the DLL parameters by using the classical control theory. In first part of this simulation we concentrate on the characteristic function of the discriminator which is the main key to mitigate the multipath effect. A simulation is presented to investigate the BBDLL noise performance in the absence and in the presence of multipath. A significant on-time simulation is achieved for the BBDLL by using MATLAB©, SIMULINK© and GPS-toolbox. The simulation is done by modeling each block in the BBDLL using the SIMULINK which is established by constructing a templet of a realistic discriminator characteristic of the actual C/A-code considering the maximal length during the correlation process. This simulation technique is made by those templetes to save the repeation of runs and to get a faster and simpler model behavior. In addition, it gives a realistic representation of the C/A code and the BBDLL behavior as on-time simulation. The SIMULINK model implemented by using templetes of real discriminator characteristic includes the linear, nonlinear and even the noisy parts in the simulation.

2.5.1 Simulation Models. The simulation includes:

- The generation of the actual C/A-code, (1023 bit) BPSK (i.e. 1 or -1).
- The sampling time parameter, which is taken 10 sample per each chip, T_c.
- The three C/A-code replicas, early, punctual and late (spacing is one chip period, T_c).
- The correlation process, is considered as the time integration of the multiplied code by replicas, the integration time is taken the whole period of the maximal length of the C/A-code, ($NT_c = 1023T_c$).
- The AWGN generator is the (randn.m) M-file in MATLAB©.
- The frequency of C/A-code is taken, $f_c = 1.023MHz$.
- The power of the received signal at the discriminator input is taken, $P = 2$ Watt.

Discriminator Model

The simulation of the discriminator is represented as a MATLAB function block taken from the SIMULINK library, its input is the time difference $(\tau_o - \hat{\tau}_o)$ and the output is the tracking error which is theoretically represented by Equation (12). The discriminator characteristic settled inside this block is considered as the S-curve constructed from the correlation process between two C/A-codes which is generated by (cacode.m) file (GPS toolbox). The generated C/A code are sampled to be 10 sample per each chip period, and the correlation process is achieved by fixing one code and shifting the second for every one sample. The M-files have been established for this simulation are:

- **(shift.m)** is a code program for shifting the C/A-code by an intended number of sample (10 per each T_c), so its input is a number representing the shift and the code returns the shifted C/A-code with the same length (1023×10 sample plots).
- **(sctmplt.m)** is a code returns the S-curve in case of noise free and also implementing a workspace including a matrix of the C/A-code, the rows representing the shift values and the columns representing the code values.
- **(scmcnos.m)** is a code implementing a workspace including tracking error matrix which represents the S-curve performed in the presence of AWGN with rows representing the number of Monte Carlo runs and the columns representing the calculated tracking error (S-curve) in presence of noise.
- **(lintrp.m)** is the code program settled inside the discriminator block in the SIMULINLK and it uses the matrix of the S-curves from the workspace of the Mat-file (sctmplt.mat) which is generated from (scmcnos.m) files in noise free case and in AWGN respectively. This program returns also the linear interpolation for the points in-between the samples.

All the previous M-file codes are shown in Appendix A. An Evaluation of the discriminator characteristics is performed for 1000 runs in the presence of AWGN by using (scmcnos.m). The output noise of the discriminator is calculated by subtracting the noisy curves from the discriminator characteristic without noise. Figures 10 through 14 show the

simulation results for the BBDLL discriminator characteristics evaluation represented by the ensemble average mean of the noise at the discriminator output which is performed for 1000 noisy S-curves. These Figures are achieved for input signal-to-noise ratio of 10,20,..,50 dB-Hz, respectively. Figure 15 through 19 show also the Standard Deviation (SD) of noise at the discriminator output. From these Figures we observe the following facts.

1. The ensemble average mean of the noise in the tracking error at the discriminator output, (the difference between the noise free S-curve and the noisy one) is random and independent on the input tracking error $(\tau_o - \hat{\tau}_o)$, however the statistics of this random values gives a close value to the zero mean (see-e.g. Table 1). Figure 10 through 14 show the ensemble average mean of for the simulated 1000 S-curves corrupted by the noise.
2. The variance of the discriminator output noise is also random, however the mean of this randomness gives close values to the calculated theoretical variances, see Table 1.
3. As expected the output noise variance is decreased as the increase of the input SNR see-e.g. Figure 20. Figures 15 through 19 show the simulation results of the standard deviation of the 1000 sample run of the S-curve per each SNR.
4. The decrease of the input SNR makes the tracking error exceeds the threshold limits for the linear part which is the operating region of the S-Curve (from $-0.5T_c$ to $0.5T_c$), viz the operating region becomes in the nonlinear part.

Table 1 Results of Discriminator Simulation.

SNR	10	20	30	40	50
$\overline{n}_\epsilon(t)$	0.0031	0.0014	3.981e-4	1.4181e-4	2.2408e-5
σ_ϵ	1.4149	0.4472	0.1415	0.0447	0.0141
$\overline{\sigma}_\epsilon$	1.4102	0.4469	0.1408	0.0447	0.0142

Second-order BBDLL SIMULINK Model

So far, the previous simulation model of the discriminator characteristic is useful for the BBDLL simulation model which is implemented by using the SIMULINK. The objectives of the BBDLL simulation are

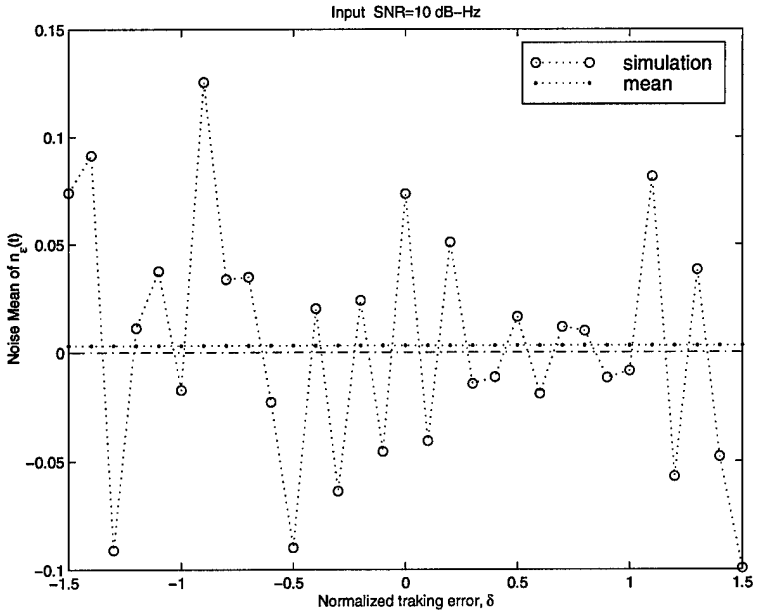


Figure 10 Simulation results showing the evaluation of the ensemble average mean of the noise $\overline{n}_\epsilon(t)$, in the discriminator characteristics (S-Curve) with input SNR=10 dB-Hz.

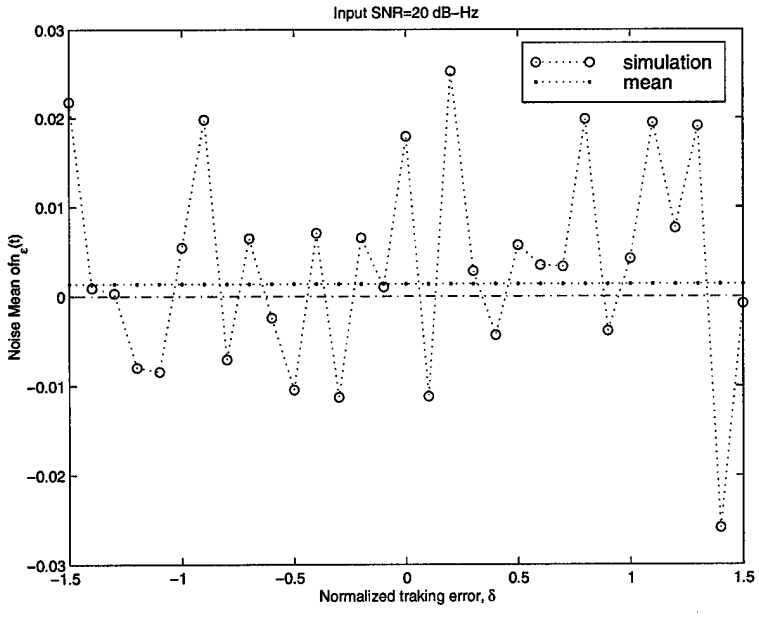


Figure 11 Simulation results showing the evaluation of the ensemble average mean of the noise $\overline{n}_\epsilon(t)$, in the discriminator characteristics (S-Curve) with input SNR=20 dB-Hz.

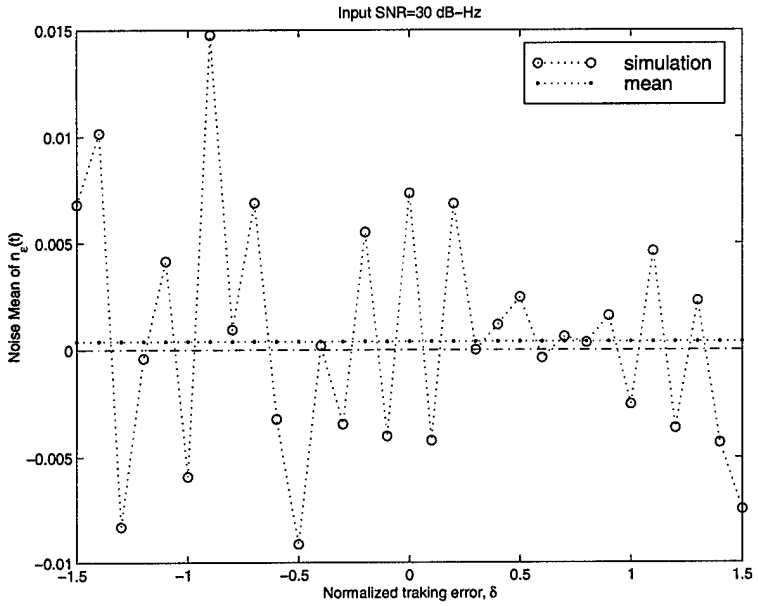


Figure 12 Simulation results showing the evaluation of the ensemble average mean of the noise $\overline{n}_\epsilon(t)$, in the discriminator characteristics (S-Curve) with input SNR=30 dB-Hz.

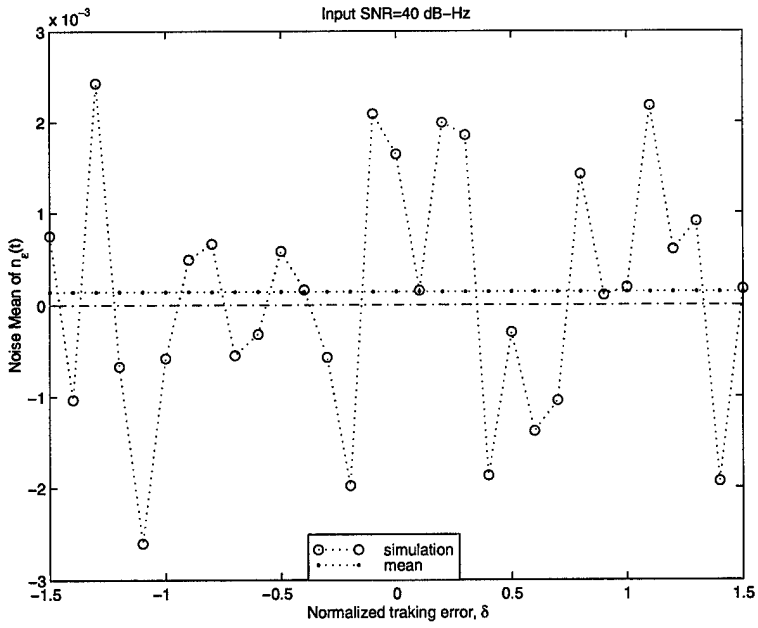


Figure 13 Simulation results showing the evaluation of the ensemble average mean of the noise $\overline{n}_\epsilon(t)$, in the discriminator characteristics (S-Curve) with input SNR=40 dB-Hz.

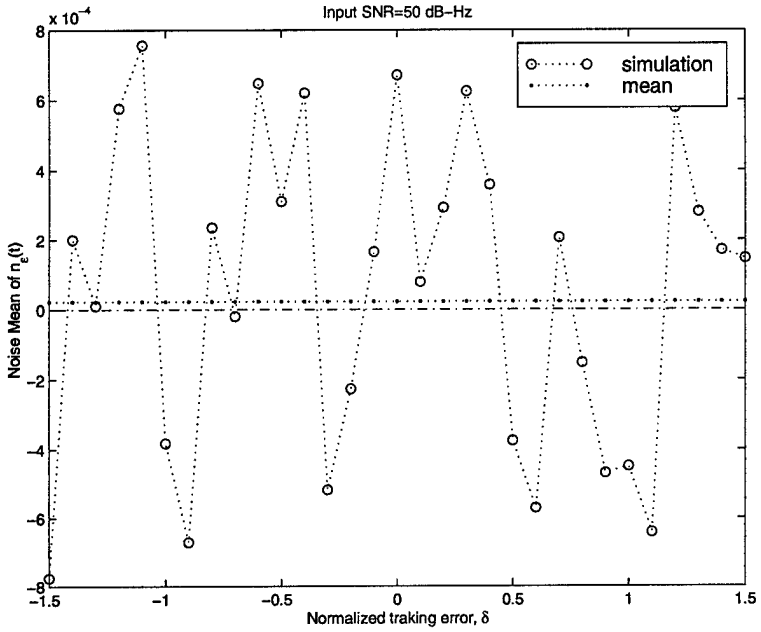


Figure 14 Simulation results showing the evaluation of the Noise mean $\overline{n}_\epsilon(t)$, in the discriminator characteristics (S-Curve) with input SNR=50 dB-Hz.

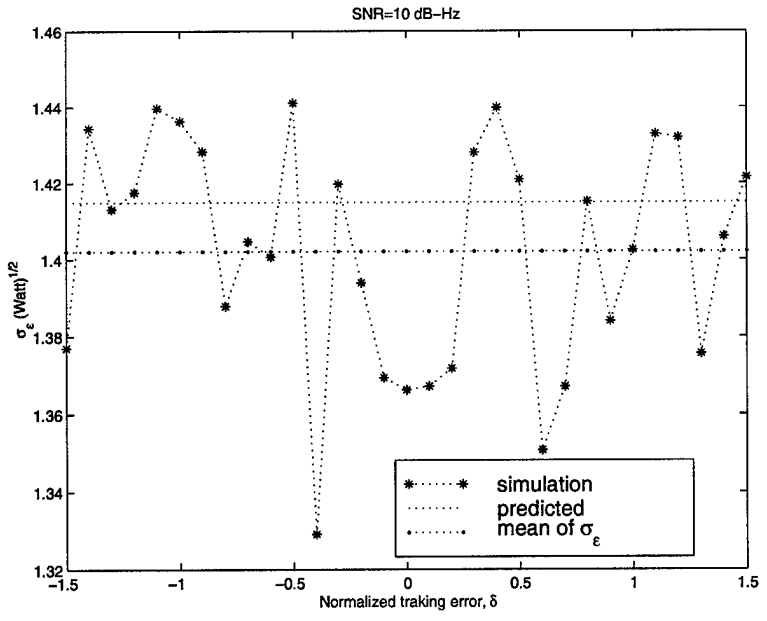


Figure 15 Simulation results showing the evaluation of the Noise SD σ_ϵ, in the discriminator characteristics (S-Curve) with input SNR=10 dB-Hz.

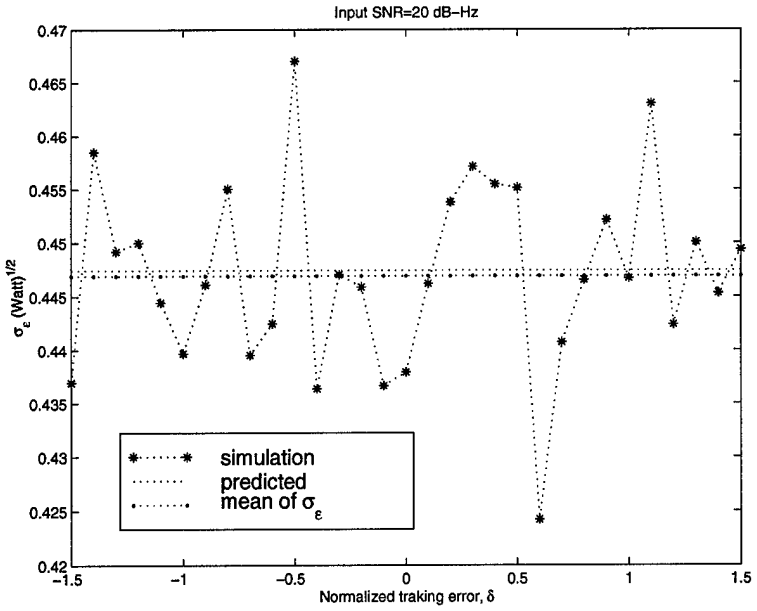


Figure 16 Simulation results showing the evaluation of the Noise SD σ_ϵ, in the discriminator characteristics (S-Curve) with input SNR=20 dB-Hz.

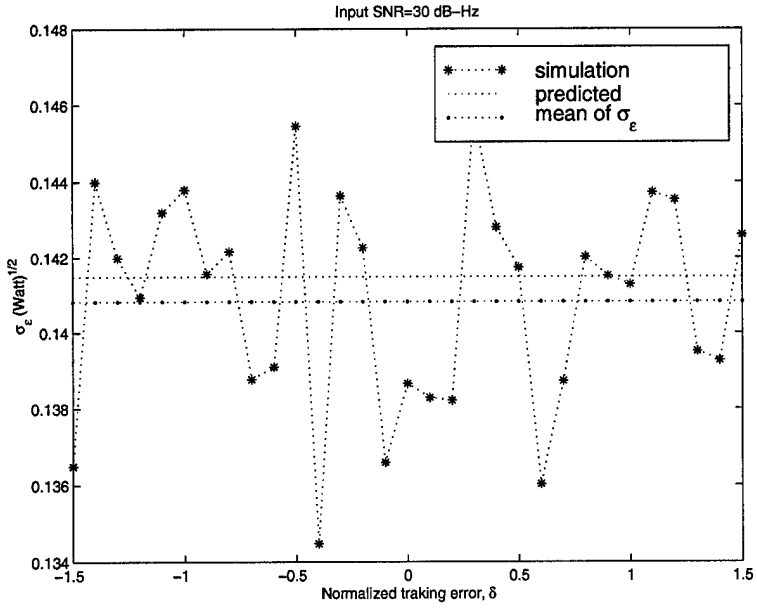


Figure 17 Simulation results showing the evaluation of the Noise SD σ_ϵ, in the discriminator characteristics (S-Curve) with input SNR=30 dB-Hz.

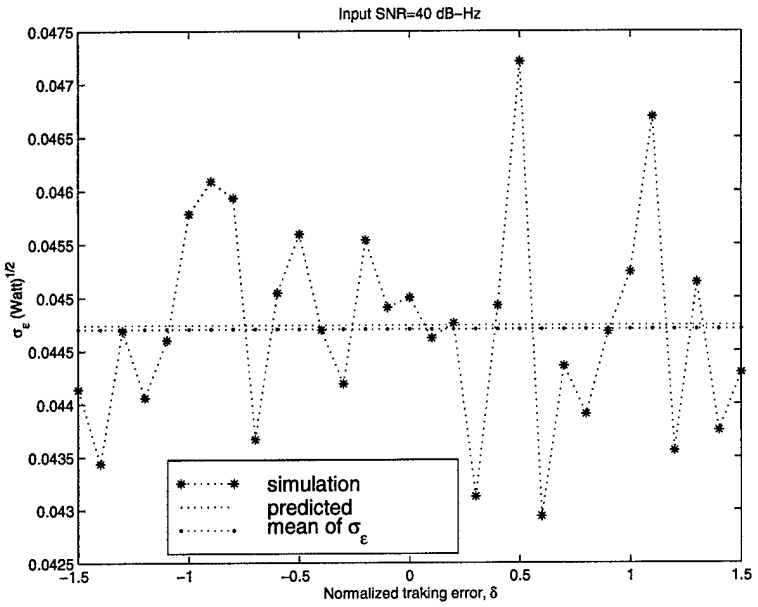


Figure 18 Simulation results showing the evaluation of the Noise SD σ_ϵ, in the discriminator characteristics (S-Curve) with input SNR=40 dB-Hz.

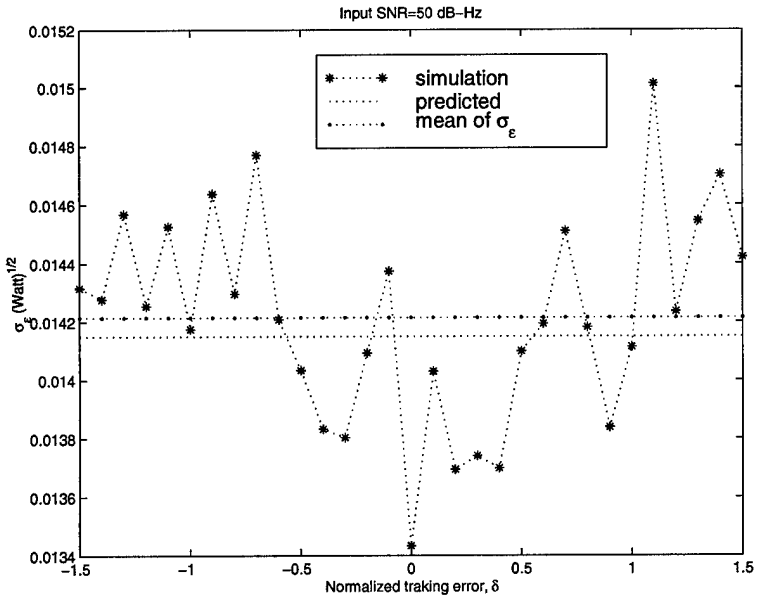


Figure 19 Simulation results showing the evaluation of the Noise SD σ_ϵ, in the discriminator characteristics (S-Curve) with input SNR=50 dB-Hz.

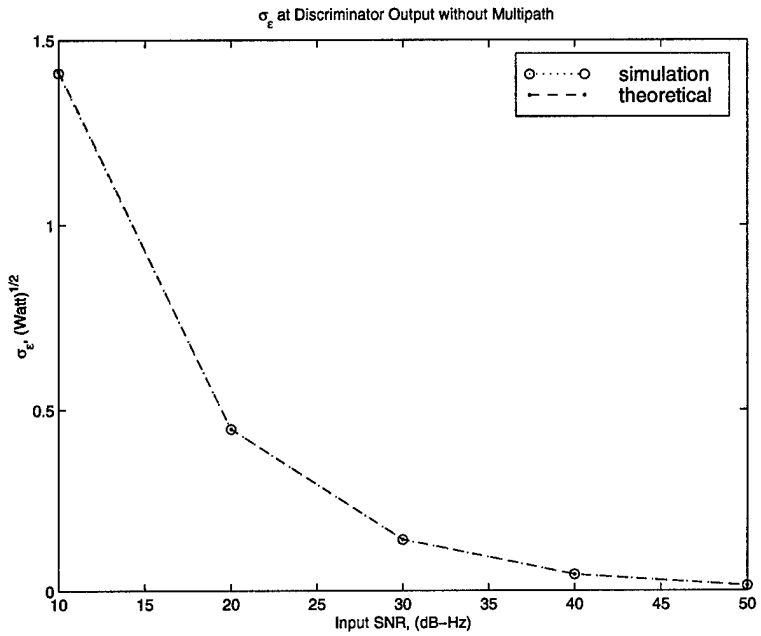


Figure 20 The simulation results of $\sigma_{\epsilon_{sim}}$ and $\sigma_{\epsilon_{th}}$ at the discriminator output.

- To examine the realisity of the servomechanism performance for BBDLL simulation model.
- To investigate the tracking and noise performance of the BBDLL.
- To prepare a templet of a simulated standard BBDLL model in order to be used in the case of the presence of multipath, as well as the evaluation results of this model will be used for a comparative assessment with the modified DLL which will be introduced in Chapter VI.

Particularly the investigation of the tracking and noise performance is performed by computing the relation between the mean-square tracking error or tracking jitter, σ_{rms} and the received signal-to-noise ratio in the loop bandwidth, ρ_L . theoretically, ρ_L is a function of the input PSD N_o, the loop bandwidth, W_L, and the received power signal, P see Equation (34). The loop bandwidth is determined by the design parameters of the loop components (discriminator gain, loop filter time constants, and VCC gain). The PSD N_o depends on ratio between the noise power level to the received signal input power per Hertz. In this simulation we introduce the relations between the root mean-square error of the tracking jitter σ_{rms} versus the signal-to-noise ratio, ρ_L. The simulation includes this relation when fixing the

loop bandwidth and changing the the input PSD, then introducing the relation when fixing the input PSD and changing the loop bandwidth. Two methods can be used for calculating the root mean-square steady state tracking jitter, the first, by using the data processing formula as

$$\sigma_{rms} = \sqrt{\frac{1}{T_{sum}}\left(\frac{1}{2}[d_1^2\Delta t_1 + \sum_{i=2}^{n-1} d_i^2(\Delta t_{i-1}+\Delta t_i) + d_n^2\Delta t_{n-1}]\right)} \tag{36}$$

where d_i is the on-time tracking jitter obtained by saving the tracking jitter values of the simulation model into a Mat-file, (it is recorded in the steady state time of the step response, ramp response), Δt_i is the time difference between the instant at d_{i+1} and d_i, T_{sum} is the period of summation in the selected part of the steady state time, and n is the number of data samples in this interval. The derivation of this formula is given in Appendix A. The second method is followed by using the Matlab command function "mean.m" applied on the squares of the tracking jitter d_i in a period of steady state, The M-file code (rms.m) is given in Appendix A. These two methods give a close values of the tracking jitter root mean-square see-e.g., Figures 28 and 29.

Figure 21 shows the SIMULINK block diagram implemented for the simulation of the standared BBDLL which includes

- the simulated system is a continuous time system.
- the discriminator gain $K_d = \sqrt{\frac{P}{2}}2(1+\frac{1}{N}) = 2.002$
- the VCC gain $g_c = \frac{1}{2.002} = 0.4995 Hz/volt.$
- the loop filter is chosen $\frac{1+\tau_2 s}{\tau_1 s}$, so we have a simulation of second-order loop as in Equation (29).
- in Equations (30) and (35) the numerical values of the closed tracking loop natural frequency, ω_n, damping factor, ξ and the loop bandwidth, W_L are chosen as in Table 2.
- the noise is simulated by using the results of previous discriminator simulation.

The most common value of the damping frequency is $\xi = 0.707$, so we do not change this value.

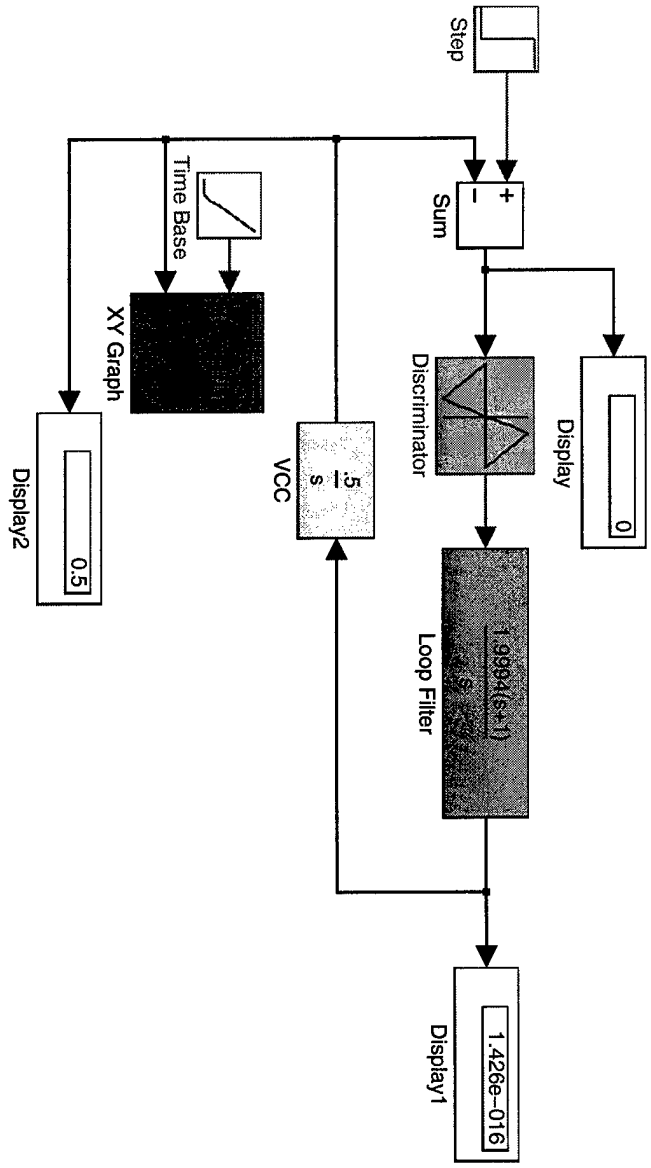


Figure 21 Simulink Block Diagram of the BBDLL.

Table 2 Second-Order BBDLL Design parameters.

ω_n (rad/sec)	ξ (rad)	W_L (Hz)	τ_1 (sec)	τ_2 (sec)
1	0.7070	1.0606	1.0000	1.4140
2	0.7070	2.1212	0.2500	0.7070
3	0.7070	3.1818	0.1111	0.4713
4	0.7070	4.2424	0.0625	0.3535
5	0.7070	5.3030	0.0400	0.2828
6	0.7070	6.3636	0.0278	0.2357
7	0.7070	7.4242	0.0204	0.2020
8	0.7070	8.4849	0.0156	0.1768
9	0.7070	9.5455	0.0123	0.1571
10	0.7070	10.6061	0.0100	0.1414

SIMULINK Parameters

The Simulink parameters involve the numerical integration of the system of ordinary differential equations (ODEs). SIMULINK has a choice between to solvers variable step and fixed step solvers. Variable-step solvers can modify their step sizes during the simulation. They provide error control and zero crossing detection. Fixed-step solvers take the same size during the simulation. They provide no error control and do not locate zero crossings. In this simulation we used the default variable-step solver with ode45 (Dormand-Prince). The ode45 is based on an explicit Runge-Kutta (4,5) formula, the Dormand-Prince pair which is the best solver for model has continuous states. (For more details see MATLAB manuals). The default parameters for this solver were

- Relative tolerance = 1E-3
- Absolute tolerance = 1E-6
- refine output factor = 1

Simulation Results

Two trade-off parameters are accounted in the design of the BBDLL which are the closed loop bandwidth; W_L and the the closed loop step response. Thus, as the wider of the loop bandwidth is the faster of the step response. In addition, there is another trade-off which is the lower of the loop signal-to-noise ratio with the increasing of the loop bandwidth. In case of noise free, Figure 22 shows the BBDLL model steps responses for different loop filter

configurations. The simulation model of the BBDLL can be validated also by comparing the theoretical calculation of the step response versus the on-time step response simulation, (see e.g., Figure 23). Figure 24 shows the simulation model which contains 3 "XY Graph3." One is connected to the loop filter output, the second is connected to the discriminator output, the last one connected to the VCC output which are plotted on Figure 22. The corresponding plots of these outputs at the discriminator output and the loop filter output are in Figures 25 and 26. Obviously the plotting curves of the discriminator output has a reverse rate with the VCC output while the loop filter output is different according to its gain parameter τ_2/τ_1. The simulation results regarding the relation between the RMS of the steady state tracking error are introduced in Figure 28 when fixing the loop bandwidth W_L, while Figure 29 shows the relation between the RMS tracking error versus the loop bandwidth. Obviously, the noise performance of the BBDLL simulation model is almost close to the theoretical loop, so as expected the RMS error of the loop is decreased as the loop bandwidth is increased, the simulation model gets a little higher error than the theoretical one when ρ_L is lower. As expected also we observe from the plot of σ_{rms} versus W_L that σ_{rms} is increased as the loop bandwidth becomes wider, and these plots also validate the performance of the simulation model in another values of loop bandwidth.

2.6 Multipath Effect on BBDLL

The operation of the BBDLL in the presence of multipath components can be interpreted from the discriminator characteristic. In the presence of multipath, the received signal can be developed as the In-Phase coherent channel as Equation (50) which is considered as the summation of the direct path code and the reflected multipath components to the receiver. Basically, The accuracy of tracking the direct path signal is dependent on the positive linear part around the zero point (equilibrium point) on the S-curve of Figure 8. Clearly the construction of the S-curve, that reserves the accurate tracking, depends on the identity of time waveform between the input PN code and the local replicas generated by the VCC in the DLL. Therefore any distortion in the time waveform of the input code follows a distortion in the triangle shape of the correlation function (Figure 6), subsequently, causing shift in the equilibrium point of the S-curve. The multipath components which are

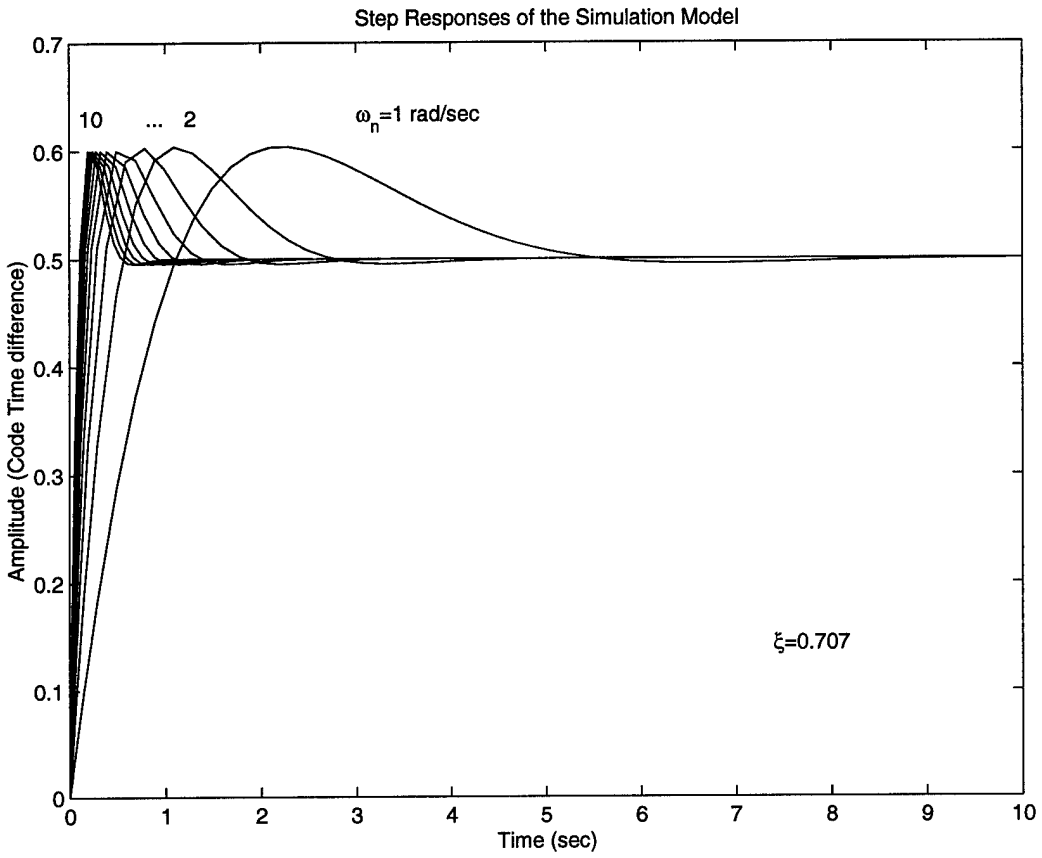


Figure 22 On-Time step Response Simulation for the closed loop model with natural frequency ω_n=1:10 rad/sec and damping factor ξ=0.707.

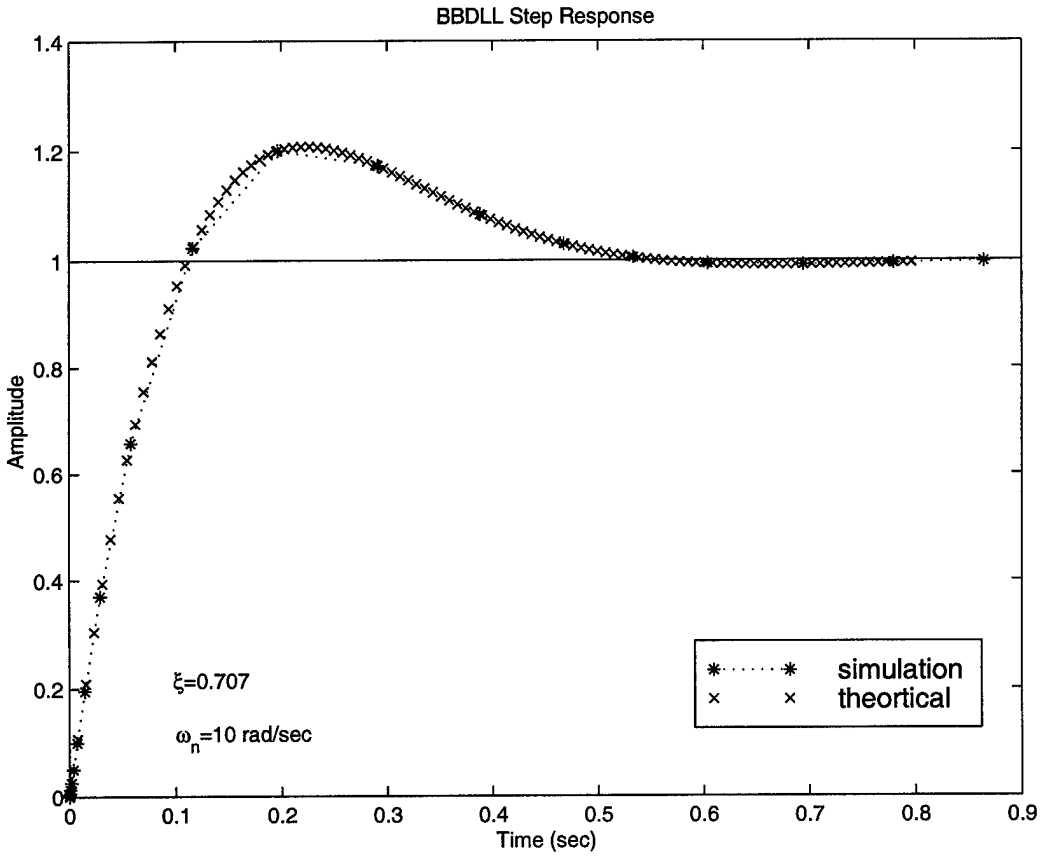


Figure 23 On-time step response simulation versus theoretical step response plot.

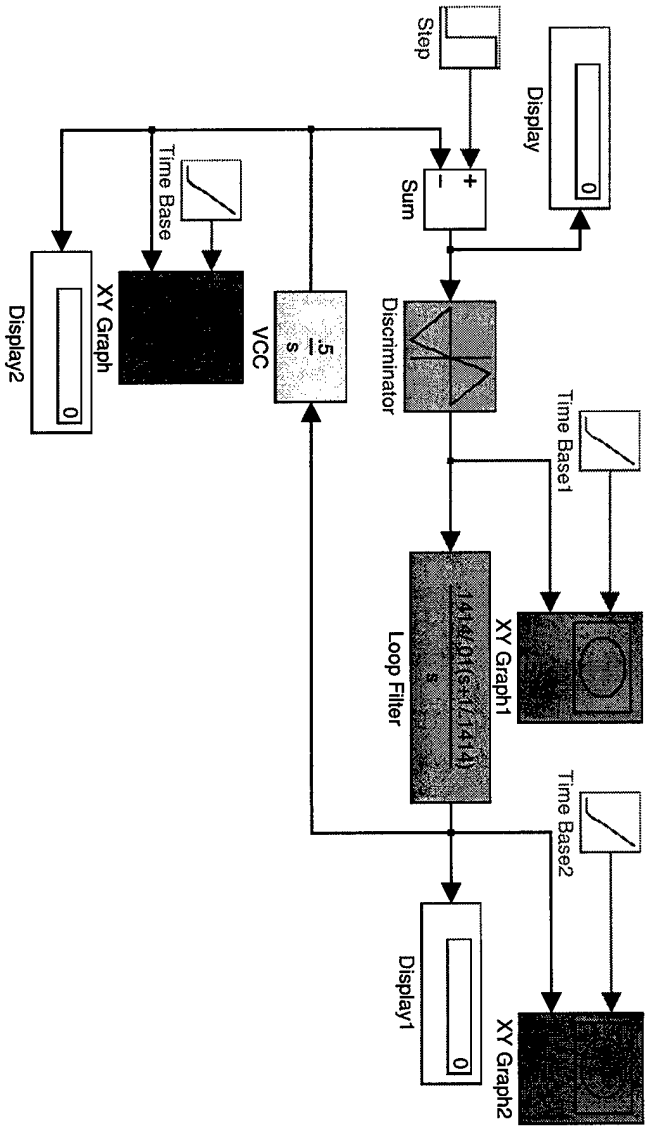


Figure 24 "XY Graphs" for on-time simulation plots.

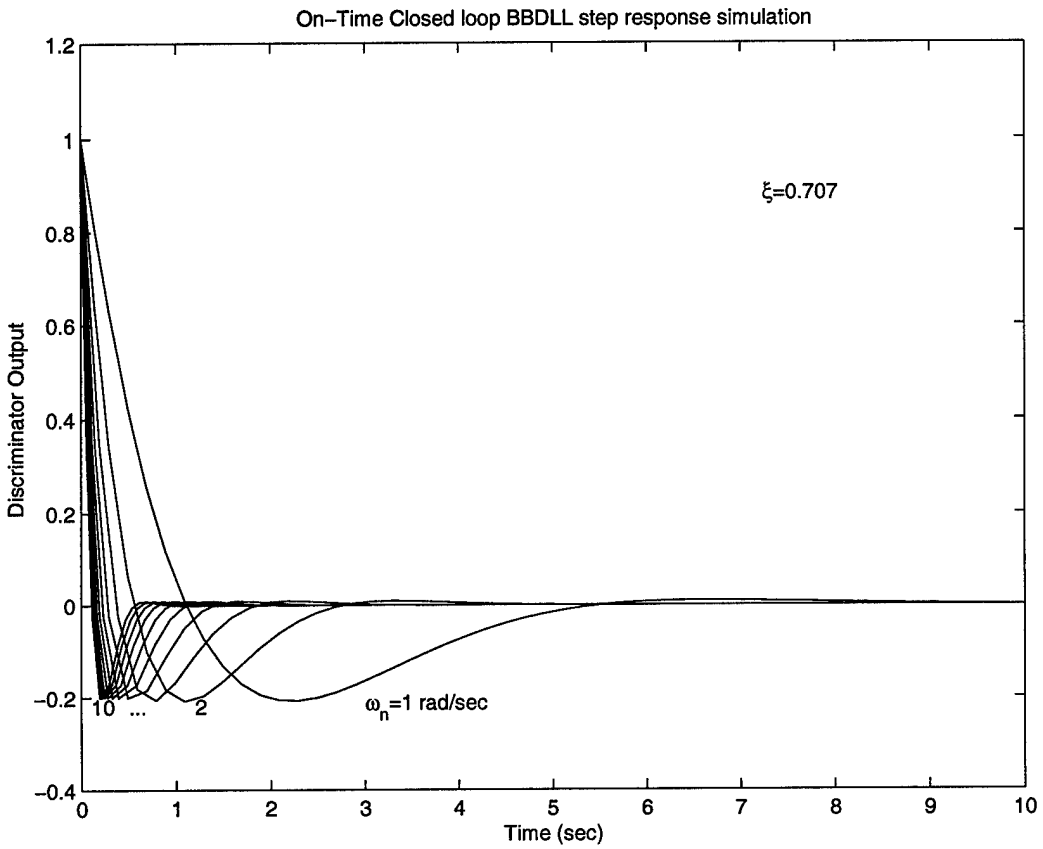


Figure 25 On-time simulation of the closed loop step response at the discriminator output.

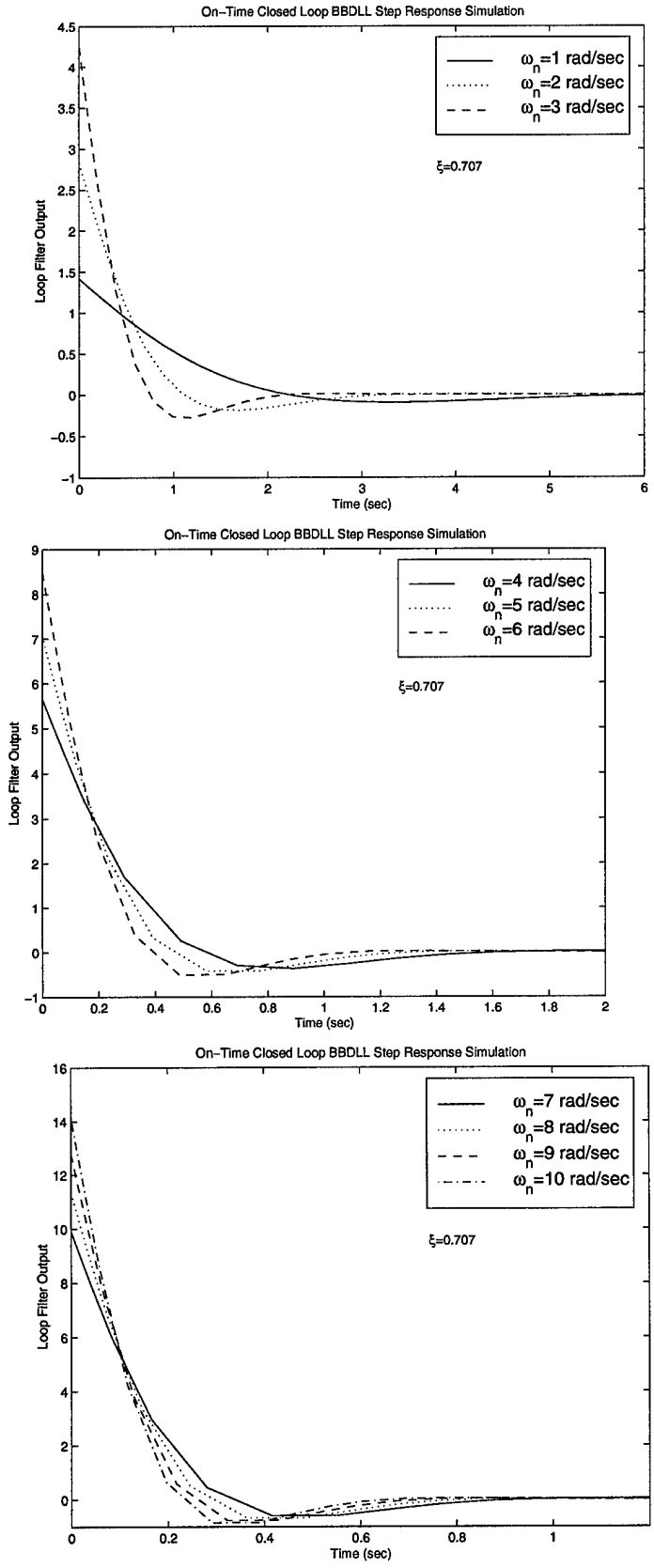


Figure 26 On-time Loop filter output of step response simulation.

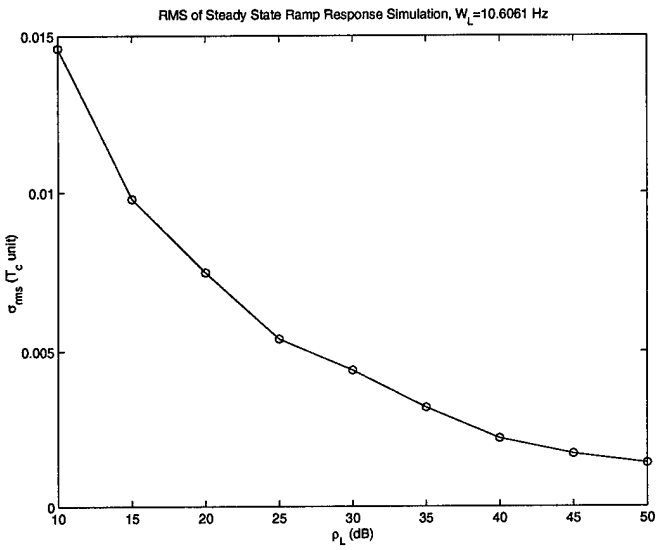


Figure 27 Simulation showing the steady state RMS Tracking error of the ramp response, σ_L versus the closed loop signal-to-noise ratio, ρ_L in loop bandwidth, $W_L = 10.6061$ Hz.

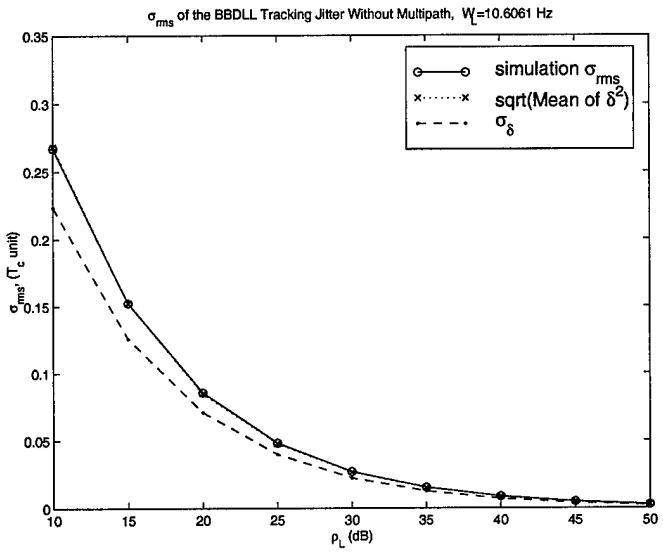


Figure 28 The simulation results showing the steady state tracking error σ_{rms} versus the loop signal-to-noise ratio ρ_L, the loop bandwidth $W_L = 10.6061Hz$.

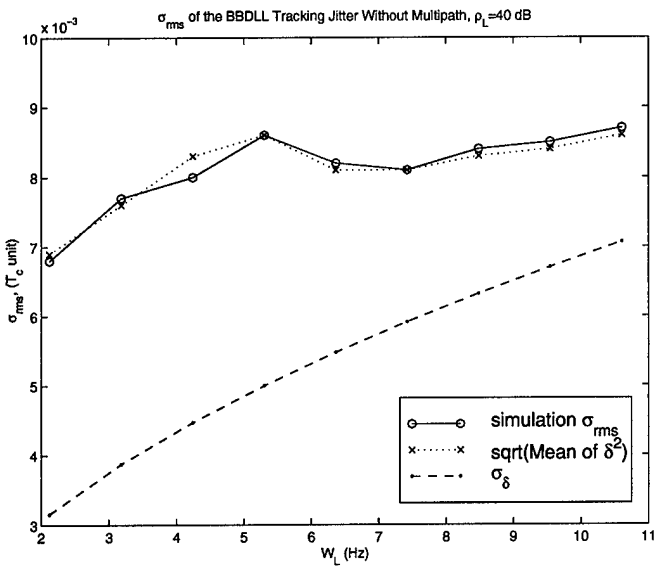


Figure 29 The simulation results showing the steady state tracking error σ_{rms} versus the loop bandwidth W_L, $\rho_L = 40dB$.

added to the direct path signal severely change the time waveform of the input code into the discriminator. In turn, the accurate tracking of the DLL can be disabled . For n multipath components, the S-curve can be derived from the difference between early-late correlators outputs, y_{1mp} and y_{2mp}, $\epsilon(\delta)$, so for noise free correlators, it can be concluded as

$$\epsilon(\delta) = y_{2mp} - y_{1mp} = \sum_{i=0}^{n} x_i S_{\frac{1}{2}}(\delta - \alpha_i) \tag{37}$$

where $\alpha_i > 0$ are the time delays due to multipath (see Chapter III) and the $S_{\frac{1}{2}}$ function is defined as

$$S_{\frac{1}{2}}(\delta) = R_c(\delta - \frac{1}{2}) - R_c(\delta + \frac{1}{2}) \tag{38}$$

A sample environment can be made by a numerical example to monitor the shift of the equilibrium point into the S-curve because of multipath.

Suppose there exist $n = 10$ reflectors around the receiver, their time delays, $\alpha_i = 0.1i$, with strengths $x_i = e^{-\alpha_i}$ (the exponential representation of the strength x_i can interpreted

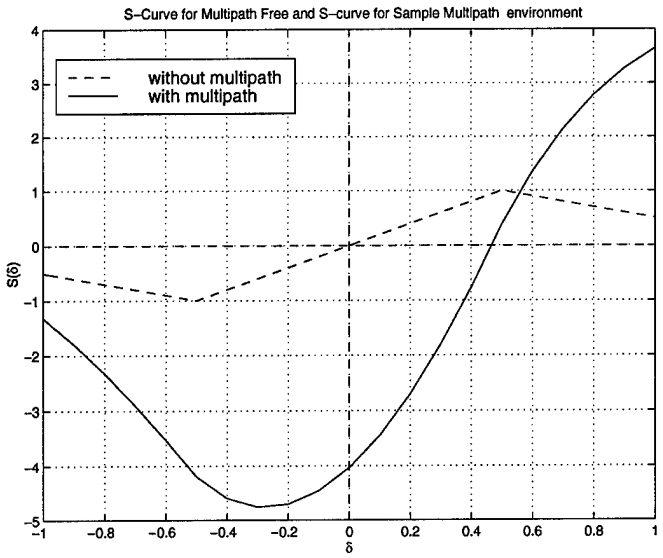


Figure 30 The S-Curve for Sample Multipath environment.

as the reflections travels a longer distance, have an extra free space loss [31]). Table 3 shows the assumed numerical values of the multipath components around the receivers.

Table 3 Parameters of Multipath Components

i	0	1	2	3	4	5	6	7	8	9	10
α_i	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
x_i	1.00	0.90	0.82	0.74	0.67	0.60	0.55	0.50	0.45	0.41	0.37

Figure 30 illustrates the difference between the S-curve in the ideal case without multipath and the case with multipath in Table 3. The Figure shows the shift of the S-curve with multipath from the zero point and a distortion in the linearity of the discriminator characteristic.

2.7 BBDLL Simulation in the Presence of Multipath

Figure 31 shows the block diagram of the simulation process followed in this section for the BBDLL investigation in the presence of multipath. The BBDLL Simulation is accomplished in the following six steps

1. The simulation starts with the calculations of the discriminator characteristics in case of noise free with multipath, as shown in Figure 31. The multipath parameters is taken

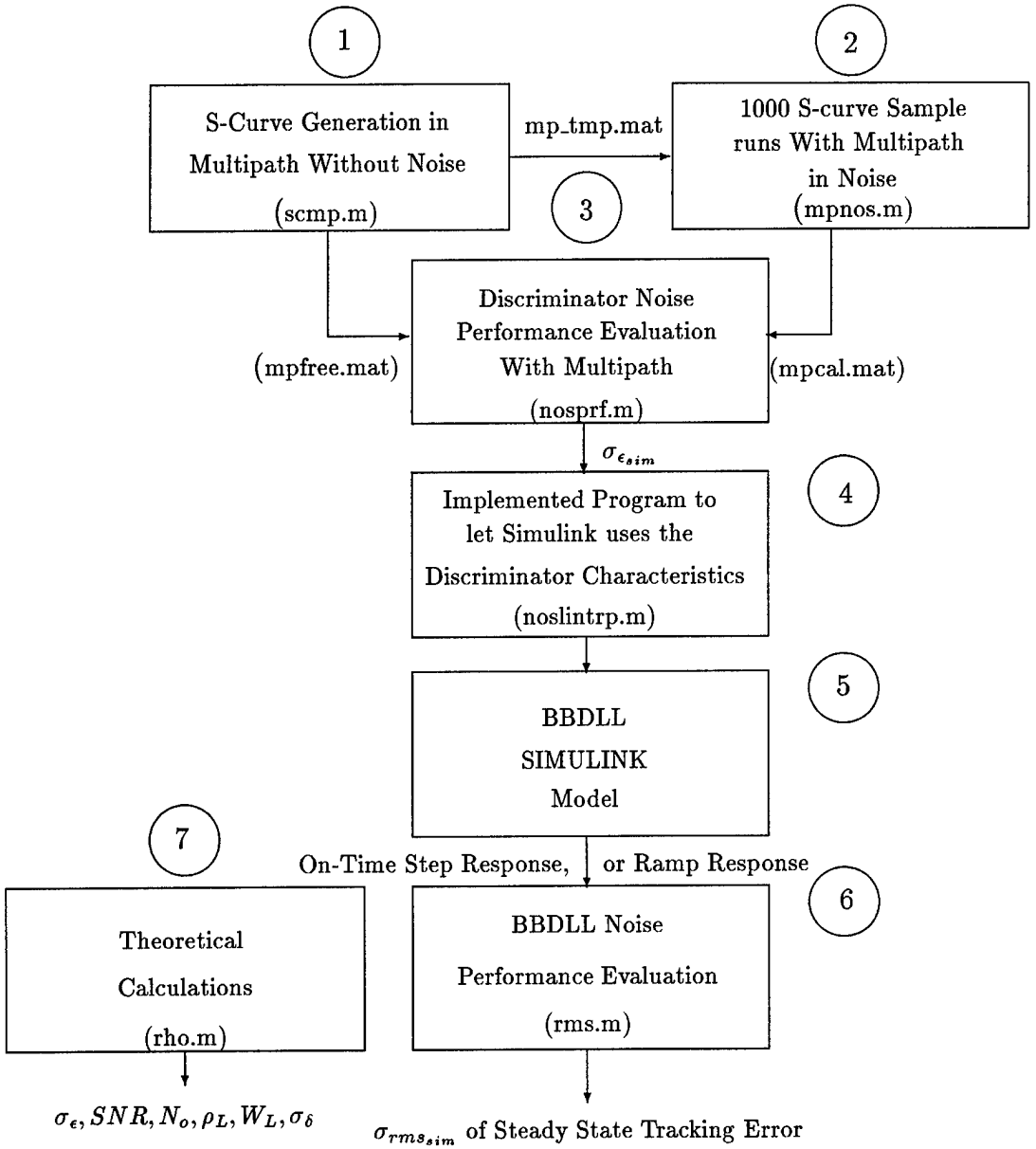


Figure 31 Simulation Block diagram of the standard BBDLL in the the presence of multipath.

for five reflections (or five multipath components plus the direct path signal) which is chosen for the time delay as, α=[0.0, 0.2, 0.5, 0.7, 1.0 1.3], and the corresponding strengths are considered as, x=[1.0, 0.7, 0.6, 0.4, 0.8, 0.3]. These multipath parameters are always taken as a reference example of the multipath in this dissertation. The simulation at this step has the program file (scmp.m) which includes the generation of the C/A-codes replicas of the DLL with its maximal length (1023 T_c), and six delayed codes with the previous α, being correlated with these replicas. These delayed replicas is representing the direct path received code and the multipath components at the discriminator input. The correlation process is achieved for each sample shift ($\frac{1}{10}T_c$). The shift process is achieved by the M-file (shift.m) as presented in Section 2.5.1. The data of the shifted codes per each shift are saved in a Mat-file (mptmplt.mat) in order to be loaded in the next step of the simulation.

2. The transfered data into this step, includes templetes of the C/A codes to be added with an AWGN for each shift in these code, which gives a realistic approach to the correlation process in the noise and in the multipath. The simulation accomplished 1000 complete S-curves, in noise to be implemented for the Monte Carlo experiment for an evaluation of the noise performance in the discriminator characteristic. The process in this step is achieved by the M-file (mpnos.m) which transfers the 1000 simulated curves into step 3 for the discriminator noise performance.

3. In this step the noise performance evaluation in the presence of the multipath is done by, evaluating the variance and the ensemble average mean of the noise at the discriminator output. The evaluation of the noise performance is performed for input signal-to-noise ratio (SNR) 10, 15, 20,...,50 db-Hz. The variance and the mean are calculated for the afore mentioned 1000 runs. The simulation uses the program (noprf.m) which loads two Mat-file (mpcal.mat and mpfree.mat) from the previous two steps, one includes the S-curve in multipath without noise and the second includes 1000 curves with multipath in noise, and this process also is done for each of the mentioned input signal-to-noise ratio. The process of noise performance evaluation is the computation of the variance and the ensemble average of the resulted difference between the 1000 noisy S-curve and

the noise free one per each SNR. So far the simulation at this step is yet to be finished, but it is also preparing the evaluated parameters of the variance at the discriminator output σ_ϵ to be used in the next step.

4. The on-time simulation begins at this step, whereas the implemented BBDLL model of Simulink described in Section 2.5.1 is employed except the discriminator characteristic is changed according to each S-curve previously calculated in step 3, in which case they were established in the multipath as well as in the noise. The Simulink model has a discriminator block representing the current discriminator characteristic (S-curve) with multipath and the AWGN is added using its statistics from the evaluated values previously determined in step 3 with $\sigma_{\epsilon_{sim}}$ per each SNR. Program (noslintrp.m) which is located inside the discriminator block, (see Figure 21), performs the current simulation of the S-curve. The input of this program is the closed loop time difference ($\delta_{sim} = \frac{\tau_o - \hat{\tau}_o}{T_c}$) and it returns the tracking error as the input of the loop filter, in addition, it returns the interpolation missed points in the simulated S-curve.

5. The BBDLL Simulink model has the same closed loop parameter as in Section 2.5.1, namely, a damping factor $\xi = 0.707$ and a natural frequency $\omega_n = 2\xi/\tau_2$, (see Table 2). The simulation in this step provides the on-time step response or the ramp response and saves it in a Mat-file for the noise performance evaluation in the next step.

6. Similar to Section 2.5.1 the noise performance in the closed loop is evaluated by the parameter $\sigma_{rms_{sim}}$ of the steady state tracking error which is achieved by program (rms.m). There is another block in Figure 31 which is implemented for the theoretical calculation of the simulation such as the the variance of the tracking error at the discriminator output σ_ϵ, the closed loop signal-to-noise ratio ρ_L, the PSD N_o, the loop bandwidth W_L, and the variance of the tracking jitter σ_δ. All the previously mentioned M-files are given in Appendix B

2.7.2 Simulation Results of the BBDLL with Multipath. *Closed loop tracking performance*

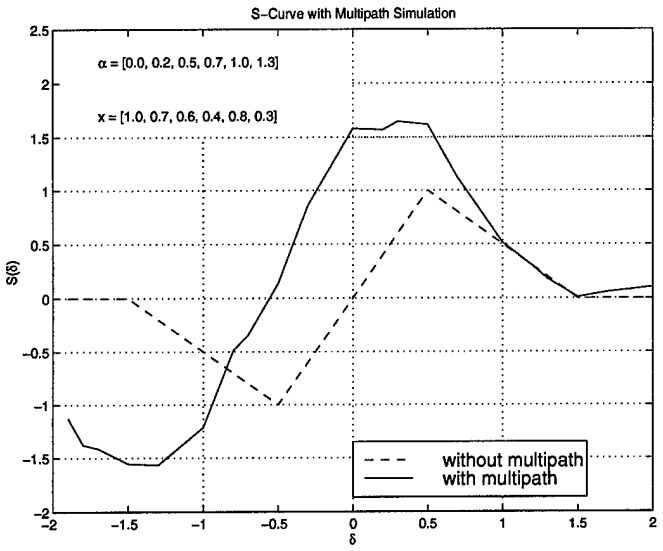


Figure 32 Typical Discriminator characteristic (S-Curve) in the Presence of Multipath.

Typically the simulation of the S-curve with the multipath (see Figure 32), that illustrates the distortion happens to the one without multipath. The plot of the S-curve with multipath shows a magnification is delivered to the original one and this is logical because the additional power that the receiver is gained from the other reflections is added to the received power signal. Figure 33 shows the on-time simulation of the step response (noise free case) taken for the step tracking error ($\delta = 0.5T_c$ at t=0 sec). The plots of the tracking error illustrates that the effect of multipath is translated into a bias error in the steady state. The resulted value of the bias in our example in this simulation is greater than the step value itself, which means the severe error may be happened by the multipath. Figure 34 also shows the on-time simulation of the step response of $\hat{\tau}_o$ at the VCC output when the input step is $\tau_o = 0.5T_c$. Figure 35 and 36 show the ramp response of the input ramp of τ_o and the tracking error δ, respectively. The significant observation on these plots are the asymptotic behavior of the ramp response with the multipath with the input ramp function, which means that the DLL can track nothing if the rate of motion of the receiver meets the value of the multipath effect.

Discriminator noise performance

Figures 37 through 41 shows the simulation results of the ensemble average of the discriminator output noise, the results is similar to the case of noise free case (see Section 2.5.1),

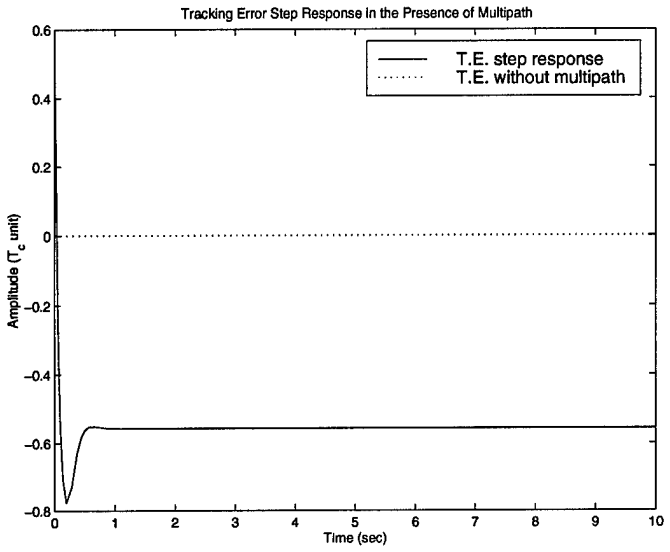


Figure 33 On-time simulation of the closed loop tracking error during the step response of the BBDLL in the presence of multipath.

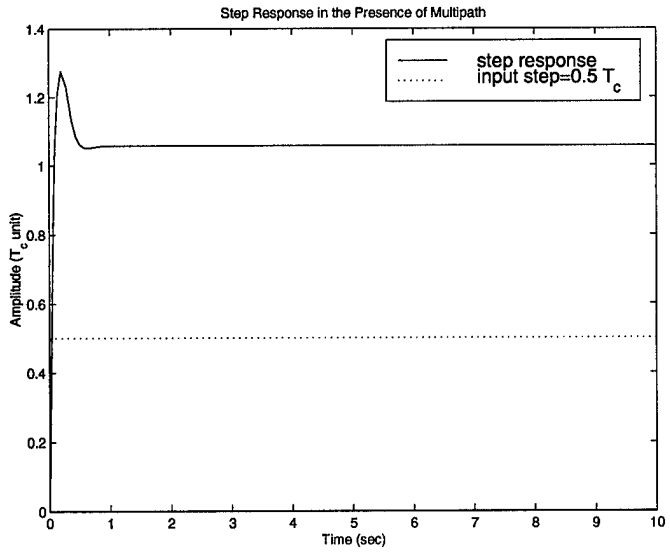


Figure 34 On-time simulation of the step response at the VCC output.

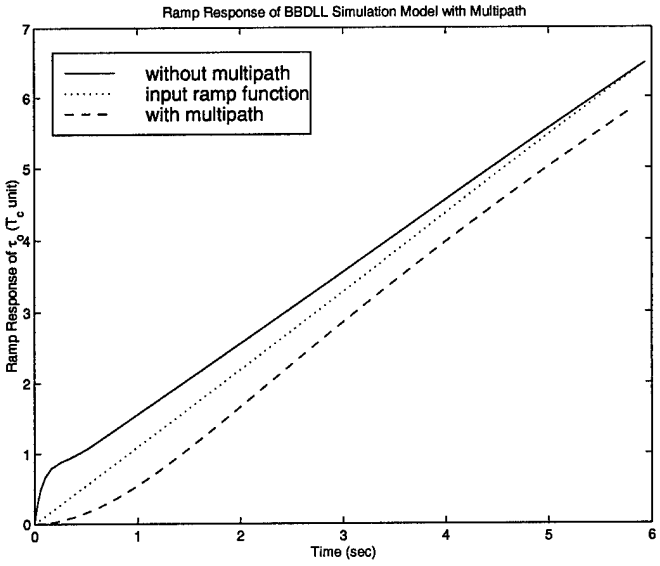


Figure 35 On-time simulation of the ramp response at the VCC output with multipath and without multipath.

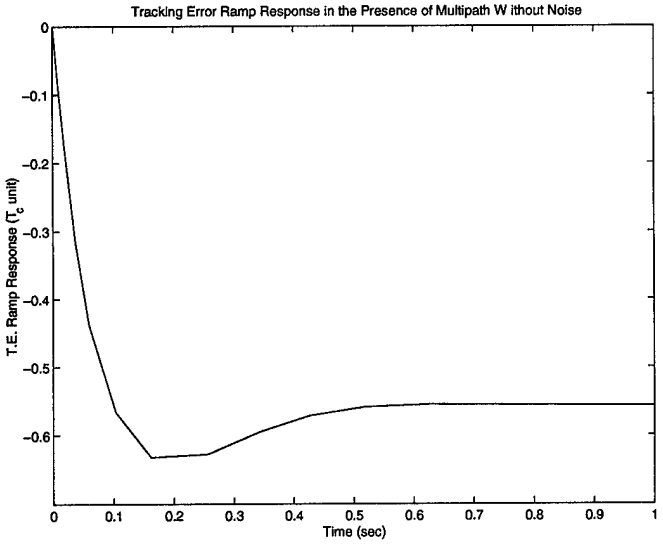


Figure 36 On-time simulation of the closed loop tracking error during the ramp response of the BBDLL in the presence of multipath.

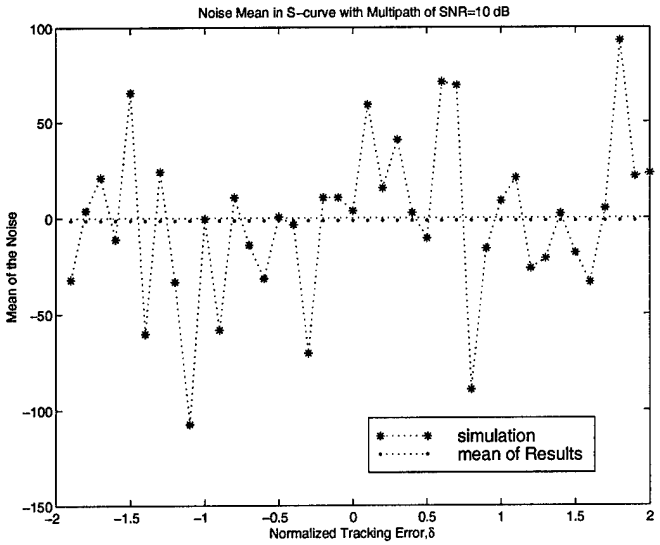


Figure 37 Simulation results showing the ensemble average mean of the noise $\overline{n}_\epsilon(t)$ at the discriminator output in the presence of the multipath, the input SNR=10 dB-Hz.

where it is random, but the mean of the results is no longer close to zero as in the noise free case. In addition the values is getting worse when the input SNR is decreased, (see Table 4). Thus, we can conclude from these plots that there exits a bias added to the S-curve because of the presence of multipath. In other words, the multipath can be interpreted as a magnifier of the input noise in the BBDLL. Figures 42 through 46 confirm this result where the noise variance becomes very large if compared with the theoretical variance $\sigma_{\epsilon_{th}}$ of noise free case Figure 47 shows these results.

Table 4 Parameters of Multipath Components

SNR	$\sigma_{\epsilon_{th}}$	$\sigma_{\epsilon_{sim}}$	$mean(\overline{n}_\epsilon(t))$
10	1.4149	24.054	-1.3956
20	0.4474	8.2722	-0.5681
30	0.1415	2.3197	0.3619
40	0.0447	0.7803	0.0980
50	0.0141	0.2461	0.0294

Noise performance in the closed tracking loop

Figure 48 shows the steady state tracking error of the closed loop. The evaluation of the variance σ^2_{rms} is achieved by using Equation 36 taken in a steady state interval from 1 to 10 seconds. The important result of the plot is that the steady state tracking error has a bias

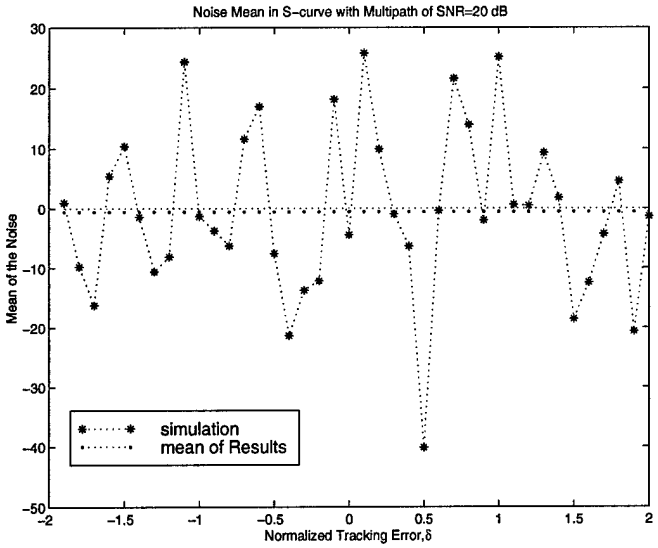


Figure 38 Simulation results showing the ensemble average mean of the noise $\overline{n}_\epsilon(t)$ at the discriminator output in the presence of the multipath, the input SNR=20 dB-Hz.

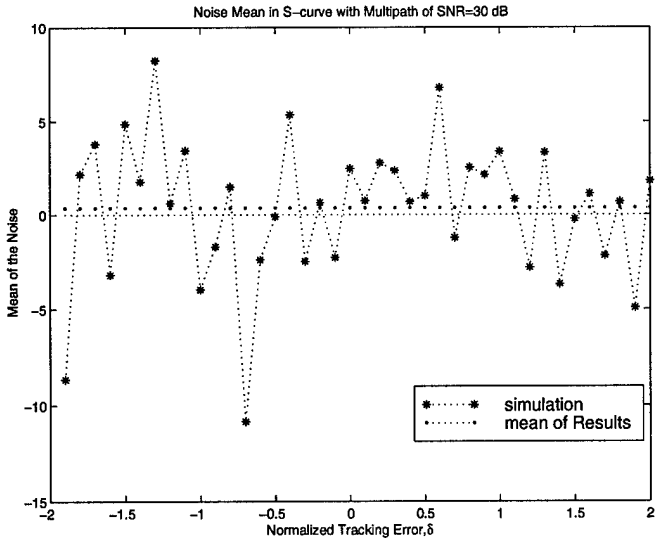


Figure 39 Simulation results showing the ensemble average mean of the noise $\overline{n}_\epsilon(t)$ at the discriminator output in the presence of the multipath, the input SNR=30 dB-Hz.

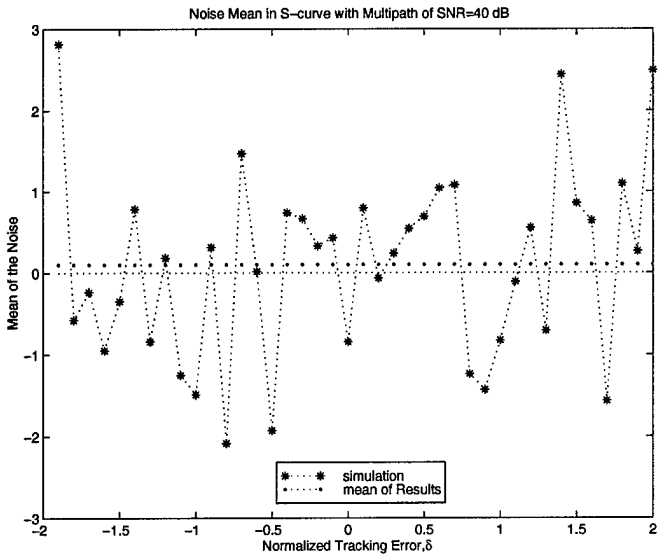


Figure 40 Simulation results showing the ensemble average mean of the noise $\overline{n}_\epsilon(t)$ at the discriminator output in the presence of the multipath, the input SNR=40 dB-Hz.

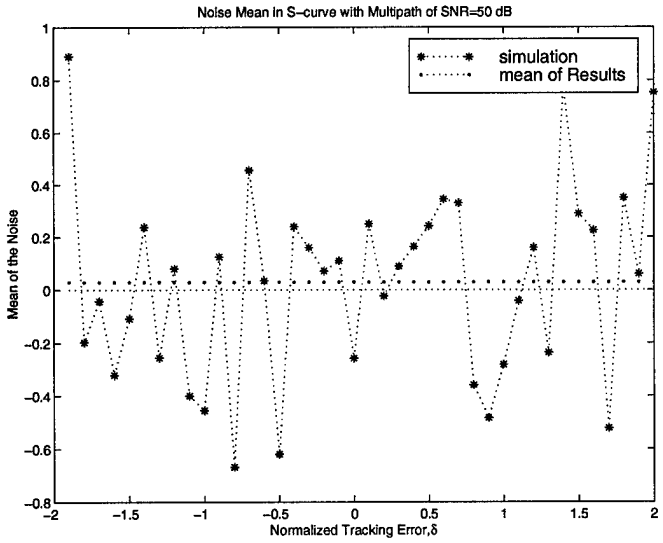


Figure 41 Simulation results showing the ensemble average mean of the noise $\overline{n}_\epsilon(t)$ at the discriminator output in the presence of the multipath, the input SNR=50 dB-Hz.

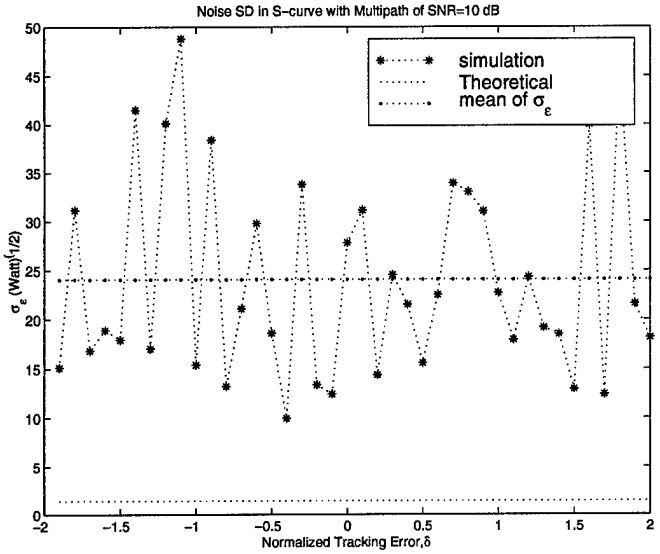


Figure 42 Simulation results showing the SD of the noise σ_ϵ at the discriminator output in the presence of the multipath, the input SNR=10 dB-Hz.

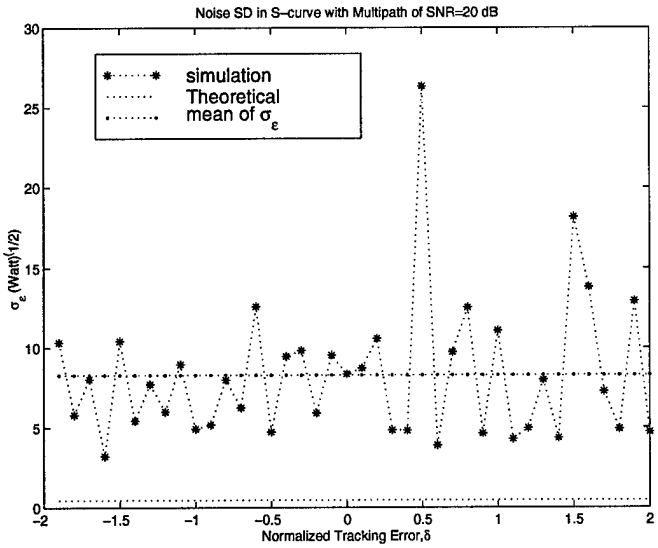


Figure 43 Simulation results showing the SD of the noise σ_ϵ at the discriminator output in the presence of the multipath, the input SNR=20 dB-Hz.

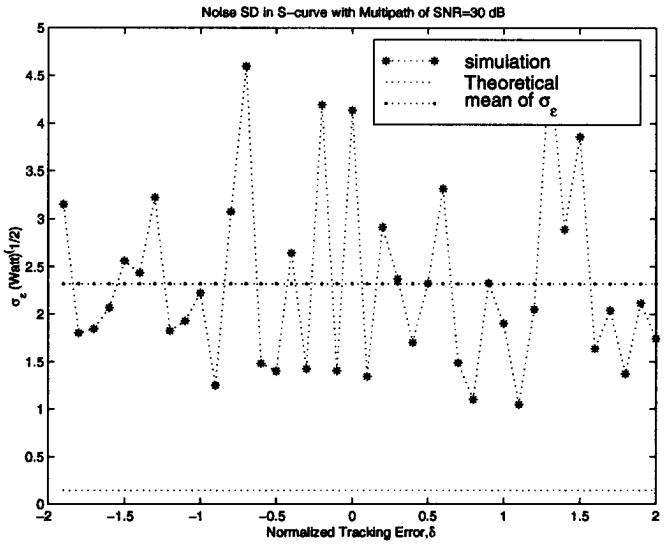


Figure 44 Simulation results showing the SD of the noise σ_ϵ at the discriminator output in the presence of the multipath, the input SNR=30 dB-Hz.

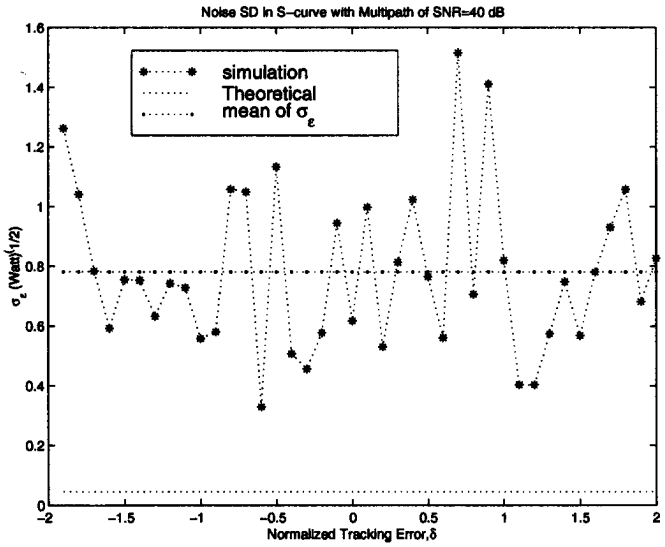


Figure 45 Simulation results showing the SD of the noise σ_ϵ at the discriminator output in the presence of the multipath, the input SNR=40 dB-Hz.

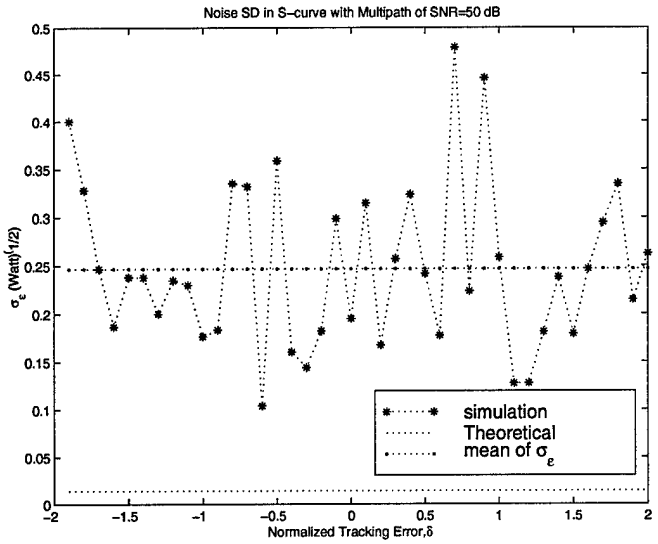


Figure 46 Simulation results showing the SD of the noise σ_ϵ at the discriminator output in the presence of the multipath, the input SNR=50 dB-Hz.

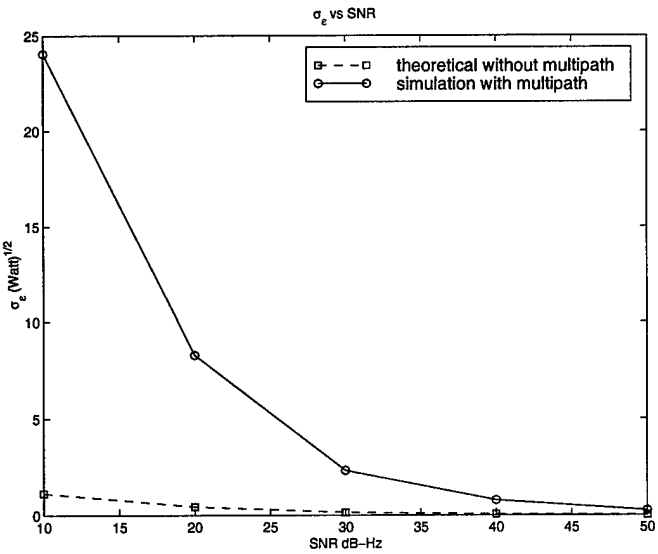


Figure 47 The Simulation results of σ_{rms} versus the input SNR.

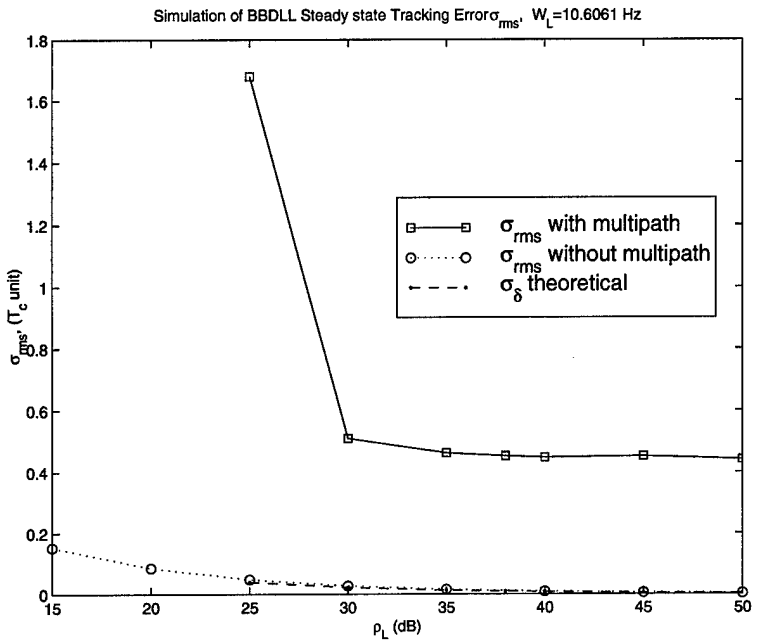


Figure 48 Simulation results shows the comparison of $\sigma_{rms_{sim}}$ with multipath and without multipath in the BBDLL

of $0.5572T_c$ caused by the multipath. the tracking error with bias looks having the same rate of the tracking error without multipath for the loop signal-to-noise ratio ρ_L from 30 dB and greater, however it jump up at $\rho_L = 25$ dB and less. The meaning of this result is that the multipath not only magnifying the noise but cause instability in the closed tracking loop especially for low signal-to-noise ratio.

2.8 Summary

1. The BBDLL simulation model using the Simulink© is close to the theoretical model in the deterministic case and with higher input SNR.
2. The BBDLL simulation shows that the multipath not only introduces a bias error on the tracking loop but also amplifies the noise. Thus, an extra bias is induced due to the noise.
3. The noise performance in the presence of multipath becomes worser with the decrease in the SNR.

III. Multipath Investigation and Modeling

3.1 Introduction

The general feature of the GPS system is its ability to transmit a simultaneous composite signal of spreading code plus data navigation message modulated with carrier signal from satellites into all users even though in the presence of any additional environmental unwanted signals. Basically we can classify the unwanted signals into three kinds: the noise, the jamming and the multipath. By communication theory, the environment or the unwanted signal can be characterized as a stationary additive white Gaussian noise (AWGN), however the jamming (whatever intended or not) and the multipath do not fit this model. The most popular technique to overcome these disturbances is the spread spectrum techniques. Several references [5, 10, 19, 20, 26] give a good analysis of the spread spectrum (SS) techniques. Typically, SS techniques are described by SS modulations such as direct sequence (DS) modulation, frequency Hopping, time Hopping ,Pulsed FM, and others. The GPS system uses the DS technique. The design of DS/SS modulation of GPS signal includes important parameters such as carrier frequency, (L1 and L2 signal), chip rate of the used code (C/A code or P code) reflecting the current accuracy of the GPS signal observation. A significant parameter that measures the receiver performance during these interferences is the process gain. The process gain is defined as the ratio between the bandwidth of the transmitted signal and the information rate. In other words, the process gain is the difference between system performance using SS techniques and system performance not using SS techniques. A probability of detection of the received signal can be enlarged by magnifying the process gain so as to improve the system performance in the presence of jamming. Nevertheless the accurate synchronization of SS techniques is the comprehensive issue of system performance . It has been shown in Chapter II that the maximum likelihood approach is an optimal estimator to achieve observation of the incoming signal immersed in AWGN and how the accuracy of observation is improved by decreasing the spacing of the differential code, however, there is a limit into the signal-to-ratio applied. The third unwanted signal (multipath) is defined also by the fading channel. Unfortunately there is no complete or

general model has been formulated for this unwanted environment such that it would be able to work as efficient as a standered model, Except for the Rayleigh distribution model which works under the assumption that for any coherent system the random phase of the incoming signal is tracked exactly. Furthermore, it is not easy to treat Rayleigh distribution like a Gaussian distribution for the standared linear estimators. Therefore the multipath is a dominant and severe error incapacitates the SS techniques to reject it from the received signal. We can summarize the process of the system performance into the following types

Rejection of the jamming signal , developed by the SS modulation design, which embodied in the accuracy of the synchronization, in acquisition and tracking, and in the processing gain, G_p, which can be calculated for the C/A code as

$$\begin{aligned} G_p &= \frac{W_{C/A}}{W_d} = \frac{1/T_c}{W_d} \\ &= \frac{1.03\times10^6 Hz}{50 Hz} = 20,600 \end{aligned} \tag{39}$$

AWGN which is a proper model of the receiver noise to detect the incoming signal by a linear estimator. The receiver noise level limits the accuracy of the signal observation, in turn it limits the performance of the associated estimators such as any additional multipath estimators. Therefore the AWGN or SNR is a very vital and advanced parameter let the signal observers or any additional technique work properly.

Multipath, or namely, fading channel , our topic in this dissertation, is the most severe unwanted signal degrades the receiver performance and accuracy, in turn it degrades the ranging process of the GPS system.

In this chapter, a complete investigation of the GPS signal model is introduced in case of code tracking and carrier phase tracking. The noise effect into the signal is also presented. In case of carrier tracking, several alternative representations of the signal in carrier loop are also introduced whether quadrature/in-phase channels. This chapter is terminated with Table 5 which summarizes the representations of the received signal processing in the tracking loops. This summary will be a useful reference for multipath mitigation work in the next chapters.

3.2 Fading Channel

In general, the typical channels that include the fading channel is the channel medium. The channel medium attenuates the signal so that the noise level of the channel cause the delivered information to be deteriorated from that of the source (satellite in the GPS). The fading channel provides multiple paths between its input and output that we named (multipath). Multipaths have different time delays and attenuation characteristics. Even worse these characteristics may vary with time [12]. Multipath produces a signal fading at output channel which struggle against the SS technique. The fading channel produces a random amplitude variation as well as a random phase variation on the received signal. The fading channel has no unified model at all, because of the varying environment between transmitters and receivers as well as the different applications used. Some communication channels can be modeled as the Rayleigh distribution known as Rayleigh channel, however this model is restricted to some assumptions which are

1. The random medium is a single surface;
2. The channel can be modeled as a linear time-invariant system.

With this assumption holding true, then the received signal can be shown to have a Rayleigh distribution for amplitude and a uniformal distribution for phase over $[0, 2\pi]$. A complete discussions and statistical modelling are well documented in [20, 31].

3.3 Multipath Modeling

It is not practical to address the multipath problem by attempting to model the environment, e.g., using a Rayleigh model, because these models are dependent on the relative motion among the satellite, the receiver, and the reflectors. In addition, the Rayleigh distribution does not meet the standard assumption of linear estimator design of Gaussianness. So the solution of this problem will be introduced in this dissertation by using an approach based on the maximum likelihood estimation applied on multiple observations of the received signal. Thus, the received signal is viewed as the superposition of all the reflected signals in

the area at the antenna. The next sections present the GPS signal model with and without multipath.

3.4 GPS Signal in Absence of Multipath

The transmitted GPS signal from one satellite can be represented as [28]

$$s(t) = A_o C(t) D(t) \cos(2\pi f_c t + \phi_o) \tag{40}$$

where f_c is the carrier frequency (either L1=1575 MHz or L2=1227 MHz), ϕ_o represents the combined effect of carrier phase noise, Doppler shift, and oscillator drift, $C(t)$ is the direct sequence spread spectrum (DS/SS) pseudo-random noise (PRN) code (in GPS, C/A code or P-code), $D(t)$ is the navigation message data. The C/A code is a ± 1 PRN sequence (Gold code) of period 1023 bits (termed "chips" to distinguish from a data bit of $D(t)$) and has a chip rate of 1.023 Mbps; so, it has period of 1 msec. The P-code is also an amplitude ± 1 PRN sequence, yet has a chip rate of 10.23 Mbps and a period of exactly 1 week. $D(t)$ is broadcast by the satellite at the relatively slow rate of 50 bps, and it has an amplitude of ± 1 as well. A_o is the transmitted signal power in watts.

3.5 GPS Signal in The Presence of Multipath

Multipath is the composite received GPS signal due to reflection, refraction, or diffraction from the local surroundings [3]. Each component of the multipath signal is delayed relative to the direct path (LOS) signal, and the relative strength for each component signal is typically less than the direct path signal, if it is not shadowed [31]. The actual received signal at the antenna of is the superposition of all of these signals and with n reflectors can be represented as:

$$r_{mp}(t) = \sum_{i=0}^{n} a_i(t) s\left(t - \tau_i(t)\right) \tag{41}$$

where $s(t)$ is the GPS signal transmitted by the satellite (see Equation 40), $a_i(t)$ is the time-varying attenuation coefficient of the i^{th} multipath component , and $\tau_i(t)$ is the time-varying propagation delay of the i^{th} multipath component. The functions $a_i(t)$ and $\tau_i(t)$ are

dependent on time due to the time-varying nature of the surroundings generated by motion of the satellites and the receiver.

3.6 GPS Signal Analysis

From Equations (40) and (41), the received signal at the antenna in the presence of multipath can be written as

$$r_{mp}(t) = \sum_{i=0}^{n} A_o a_i(t) C\left(t-\tau_i(t)\right) D\left(t-\tau_i(t)\right) \cos\left(\omega_c t+\phi_i(t)\right) \tag{42}$$

where $\omega_c = 2\pi f_c$, $\phi_i(t) = 2\pi f_c \tau_i(t)$, $\tau_i(t) = \tau_o+\alpha_i(t)T_c$, τ_o is the direct path signal propagation delay (considered as the raw range measurement, termed pseudo-range, of the code tracking loop), and α_i is a delay coefficient such that $\alpha_o = 0$ and $\alpha_i > 0$ for $i \geq 1$. In the GPS code tracking loop, $0 < \alpha_i(t) < 1.5$ is the range of $\alpha_i(t)$ which can corrupt the pseudorange measurement; This follows from the correlation properties of $C(t)$ as discussed in [31]. T_c denotes the duration of a chip of the PRN code. r_{mp} also can be written in the form of sinusoids as $r_{mp}(t) = A_m(t)\cos\left(\omega_c t+\phi_i(t)+\phi_m(t)\right)$, where the phase error, ϕ_m, is due to the composite effect of all of the reflected signals. By using trigonometry, and suppressing the time dependence on τ_i and ϕ_i an expression for the multipath generated carrier phase error, ϕ_m, is

$$\phi_m(t) = \arctan\left(\frac{\sum_{i=0}^{n} a_i A_o C\left(t-\tau_i\right) D\left(t-\tau_i\right) \sin\left(\phi_i-\phi_o\right)}{\sum_{i=0}^{n} a_i A_o C\left(t-\tau_i\right) D\left(t-\tau_i\right) \cos\left(\phi_i-\phi_o\right)}\right) \tag{43}$$

where arctan is the principal inverse of tangent, i.e. $-\frac{\pi}{2} < \phi_m(t) \leq \frac{\pi}{2}$. The time-varying amplitude, A_m, of the multipath signal can be represented as

$$A_m(t) = \sqrt{\left(\sum_{i=0}^{n} A_i \cos\left(\phi_i-\phi_o\right)\right)^2 + \left(\sum_{i=0}^{n} A_i \cos\left(\phi_i-\phi_o\right)\right)^2} \tag{44}$$

where $A_i = a_i A_o C\left(t-\tau_i\right) D\left(t-\tau_i\right)$. This alternative sinusoid form is potentially useful for the mitigation of multipath carrier phase error, and will be discussed later.

3.7 Multipath Signal Model in GPS Tracking

3.7.1 Code Tracking Loop. From Section 1.4, the received signal at the correlation process for the in-phase and quadrature channels, respectively, are given by

$$\begin{aligned} R_I &= R_c\left(\tau_o - \hat{\tau}_o + \beta_{e,p,l}\right)\cos\left(\phi_o - \hat{\phi}_o\right) \\ R_Q &= R_c\left(\tau_o - \hat{\tau}_o + \beta_{e,p,l}\right)\sin\left(\phi_o - \hat{\phi}_o\right) \end{aligned} \tag{45}$$

where $\beta_{e,p,l}$ is the correlator delay corresponding to early, punctual, and late values of $-e$, 0, $+e$ respectively, and e is the time separation of the early and late correlators (see Section 2.4). $R_c(\Omega)$ is the autocorrelation function and is defined in Equation (8). $\hat{\phi}_o$ is the estimated phase of the received carrier signal which is the phase of the local carrier generated by the voltage controlled oscillator in the steady state of the PLL (i.e., $\hat{\phi}_o = \phi_{pll}$).

From Equation (42) the received multipath signal plus noise (neglecting data modulation) is given by

$$r_{mp}(t) = \sum_{i=0}^{n} a_i(t) A_o C\left(t - \tau_o - \alpha_i T_c\right)\cos\left(\omega_c t - \phi_i\right) + n(t) \tag{46}$$

where $n(t)$ is assumed to be band-limited white Gaussian noise with a two-sided power spectral density (PSD) of $\frac{N_o}{2}$ W/Hz [27] represented as

$$n(t) = \sqrt{2} n_I(t)\cos\omega_c t - \sqrt{2} n_Q(t)\sin\omega_c t \tag{47}$$

Using the representation in Equation (47), the in-phase and quadrature components of the noise, ($n_I(t)$ and $n_Q(t)$, respectively) are independent zero-mean low-pass Gaussian noise processes each having a two-sided PSD of $\frac{N_o}{2}$ W/Hz [10, 20, 27]. The carrier phase recovery is characterized by conversion to base band using a multiplier type phase detector. (Multiplying the received signal by local carrier from PLL , $2\cos(\omega_c t - \phi_{pll})$, followed by a low pass filter. See Figure 49 and see Chapter V for more details). The output signal is

$$r(t) = \sum_{i=0}^{n} x_i C\left(t - \tau_o - \alpha_i T_c\right) + n'(t) \tag{48}$$

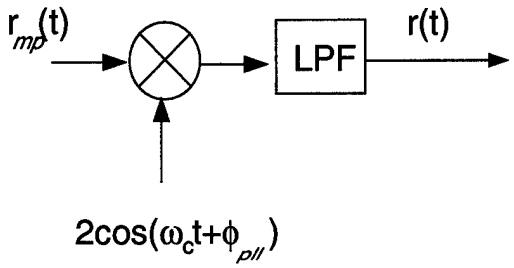


Figure 49 Sinusoidal Multiplier

where $x_i = a_i A_o \cos(\phi_i - \phi_{pll})$, $\phi_{pll} \triangleq \phi_o + \phi_m$ and $n'(t)$ is a low-pass, zero-mean additive white Gaussian noise (AWGN) process given as

$$n'(t) = \sqrt{2} n_I(t) \cos\phi_{pll} - \sqrt{2} n_Q(t) \sin\phi_{pll} \tag{49}$$

In the presence of multipath, the input to the correlator is the signal defined in Equation (48). The correlation process given in Equation (45) is altered due to the presence of the additional multipath component. These corrupted measurements for IP and QP are denoted as R_{Imp} and R_{Qmp} and given by

$$\begin{aligned} R_{Imp} &= \textstyle\sum_{i=0}^{n} x_i R_c(\tau_o - \hat{\tau}_o + \beta_{e,p,l} - \alpha_i T_c) + \nu_I \\ R_{Qmp} &= \textstyle\sum_{i=0}^{n} y_i R_c(\tau_o - \hat{\tau}_o + \beta_{e,p,l} - \alpha_i T_c) + \nu_Q \end{aligned} \tag{50}$$

where $y_i = a_i A_o \sin(\phi_i - \phi_{pll})$. The values ν_I and ν_Q are the correlator's output noises in both in-phase and quadrature channels, respectively. They are assumed to be zero-mean, lowpass AWGN [14] represented as

$$\nu_I = n'(t) C(t - \hat{\tau}_o - \beta_{e,p,l} T_c) * h_{lpf}(t)$$

$$\nu_Q = n''(t) C(t - \hat{\tau}_o - \beta_{e,p,l} T_c) * h_{lpf}(t)$$

where $n'(t)$ is the received in-phase base band signal noise defined in Equation (49), and $*$ denotes convolution. $n''(t)$ is the received quadrature base band signal noise given as [10].

$$n''(t) = \sqrt{2} n_I(t) \cos\phi_{pll} - \sqrt{2} n_Q(t) \sin\phi_{pll} \tag{51}$$

The sinusoidal form of Equation (50) is given by

$$\begin{aligned} R_{Imp} &= A_{m_\beta} \cos\left(\phi_{m_\beta}+\phi_o-\phi_{pll}\right) \\ R_{Qmp} &= A_{m_\beta} \sin\left(\phi_{m_\beta}+\phi_o-\phi_{pll}\right) \end{aligned} \tag{52}$$

where ϕ_{m_β} is given by

$$\phi_{m_\beta} = \arctan\left(\frac{\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o-\hat{\tau}_o+\beta_{e,p,l}-\alpha_i T_c\right)\sin(\phi_i-\phi_o)}{\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o-\hat{\tau}_o+\beta_{e,p,l}-\alpha_i T_c\right)\cos(\phi_i-\phi_o)}\right) \tag{53}$$

and A_{m_β} is given by

$$A_{m_\beta} = \sqrt{[\sum_{i=0}^{n} a_i A_o R_c(.)\sin(\phi_i-\phi_o)]^2+[\sum_{i=0}^{n} a_i A_o R_c(.)]^2} \tag{54}$$

where $(.) = (\tau_o-\hat{\tau}_o+\beta_{e,p,l}-\alpha_i T_c)$. The local carrier is generated in the GPS receiver by using a phase-locked loop (PLL) provided with an estimate of the punctual code phase, (see Figures 3 and 50).

The local carrier phase, ϕ_{pll}, is approximately equal to the direct path signal phase, ϕ_o, when no multipath is present. The signal in the PLL can be written as

$$R'_I(t) = \sum_{i=0}^{n} a_i A_o R_c\left(\tau_o-\hat{\tau}_o-\alpha_i T_c\right)\cos\left(\omega_c t-\phi_i\right) \tag{55}$$

and in the QP form is

$$R'_Q(t) = \sum_{i=0}^{n} a_i A_o R_c\left(\tau_o-\hat{\tau}_o-\alpha_i T_c\right)\sin\left(\omega_c t-\phi_i\right) \tag{56}$$

Equation (55) and (56) can be represented in the form

$$R'_I(t) = A_m \cos\left(\omega_c t+\phi_o+\phi_m\right) \tag{57}$$

$$R'_Q(t) = A_m \sin\left(\omega_c t+\phi_o+\phi_m\right) \tag{58}$$

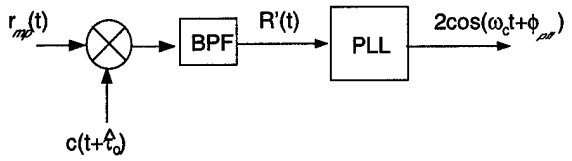


Figure 50 Reference Carrier

where the phase error, ϕ'_m, is a function of the reflected signal parameters, a_i and α_i, and it is also dependent on the code tracking error, $\tau_o - \hat{\tau}_o$, and is completely described as

$$\phi'_m = \arctan\left(\frac{\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o - \hat{\tau}_o - \alpha_i T_c\right)\sin\left(\phi_i - \phi_o\right)}{\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o - \hat{\tau}_o - \alpha_i T_c\right)\cos\left(\phi_i - \phi_o\right)}\right) \tag{59}$$

The equivalent multipath amplitude, A'_m, is given by

$$A'_m = \sqrt{\left(\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o - \hat{\tau}_o - \alpha_i T_c\right)\sin\left(\phi_i - \phi_o\right)\right)^2 + \left(\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o - \hat{\tau}_o - \alpha_i T_c\right)\cos\left(\phi_i - \phi_o\right)\right)^2} \tag{60}$$

Note that $\phi_{pll} = \phi_o + \phi'_m$. The carrier phase tracking of the PLL (see Figures 50 and 3) is achieved by setting the quadrature phase component to zero; the details will be given later. If the punctual code is employed in Equation (52), then Equations (53) and (54) will be equivalent to Equations (59) and (60), respectively.

3.7.2 Carrier Tracking Loop. The main loop to track the carrier phase is the PLL which includes an ideal multiplier-type Phase Detector (PD) , a loop filter and a VCO (see Chapter V for more details). The VCO generates two pure carriers, namely the quadrature/phase components (sine and cosine wave) and it must be aligned with the received carrier. To model the multipath signal in the carrier tracking loop, recall Equations (46) and (49) The ideal multiplier-type PD output V_d can be represented as (see Figure 49)

$$V_d = K_d\left(\sum_{i=0}^{n} a_i A_o R_c\left(\tau_o - \hat{\tau}_o - \alpha_i T_c\right)\sin\left(\phi_i - \phi_{pll}\right)\right) + n'(t) \tag{61}$$

where $\phi_{pll} = \widehat{\phi_o + \phi_m}$, is the estimate of the phase of the received carrier by the PLL in the presence of multipath which include the estimate of ϕ_m and ϕ_o in Equation (59) and Equation (40) respectively. ϕ_i is the multipath component signal, and $n'(t)$ is defined in Equation (49). K_d is the PD gain. Also, the alternative form of Equation (61) is

$$V_d = A_m \sin\left(\phi_m + \phi_o - \phi_{pll}\right) \tag{62}$$

Obviously, at the steady state of the PLL Equation (62) becomes zero, which means that the equilibrium point is biased by the multipath phase ϕ'_m. Details of this point and the carrier phase mitigation will be later. The resulting IP and QP sinusoidal signal from PLL can be written as

$$\begin{aligned} QP &= A_m \cos\left(\widehat{\phi_m + \phi_o} - \phi_{pll}\right) \\ QP &= A_m \sin\left(\widehat{\phi_m + \phi_o} - \phi_{pll}\right) \end{aligned} \tag{63}$$

3.8 Summary

The operation of code phase and carrier phase tracking loops in the GPS receivers is inter-dependent. So the equation representing the code tracking is completely related to the carrier tracking. Table 5 shows the alternatives of the signal model in the tracking loops for the cases with and without multipath.

Table 5 Multipath Signal models in DLL and PLL

Signal Model	Code Phase Tracking	Carrier Phase Tracking
without Multipath	**Correlator** $R_I = R_c(\tau_o - \hat{\tau}_o + \beta_{e,p,l})\cos(\phi_o - \hat{\phi}_o)$ $R_Q = R_c(\tau_o - \hat{\tau}_o + \beta_{e,p,l})\sin(\phi_o - \hat{\phi}_o)$	**Phase Detector** $V_d = K_d R_c(\tau_o - \hat{\tau}_o)\left(\phi_o - \hat{\phi}_o\right)$
with Multipath	$R_{I_{mp}} = \sum_{i=0}^{n} x_i R_c(\tau_o - \hat{\tau}_o + \beta_{e,p,l} - \alpha_i T_c)$ $R_{Q_{mp}} = \sum_{i=0}^{n} y_i R_c(\tau_o - \hat{\tau}_o + \beta_{e,p,l} - \alpha_i T_c)$ where $x_i = a_i A_o \cos(\phi_i - \phi_{pll})$ $y_i = a_i A_o \sin(\phi_i - \phi_{pll})$	$V_d = \sum_{i=0}^{n} u_i \left(\phi_i - \phi_{pll}\right)$ where $u_i = K_d a_i A_o R_c(\tau_o - \hat{\tau}_o - \alpha_i T_c)$
	Reference carrier $R'_I = A_m \cos\left(\omega_c t + \phi_o + \phi_m\right)$ $R'_Q = A_m \sin\left(\omega_c t + \phi_o + \phi_m\right)$	**Local carrier** $IP = \hat{A}_m \cos\left(\omega_c t + \widehat{\phi_m + \phi_o}\right)$ $QP = \hat{A}_m \sin\left(\omega_c t + \widehat{\phi_m + \phi_o}\right)$
	Multipath phase and amplitude $\phi_m = \arctan\left(\frac{QV_{d_o}}{IV_{d_o}}\right)$ $A_m = \sqrt{[IV_{d_o}]^2 + [QV_{d_o}]^2}$ where $IV_{d_o} = \sum_{i=0}^{n} A_i \cos(\phi_i - \phi_o)$ $QV_{d_o} = \sum_{i=0}^{n} A_i \sin(\phi_i - \phi_o)$ $A_i = a_i A_o R_c\left(\tau_o - \hat{\tau}_o - \alpha_i T_c\right)$	**Estimated phase of PLL** $\phi_{pll} = \widehat{\phi_m + \phi_o}$

IV. Multipath Estimation

4.1 Introduction

A promising tool to detect and mitigate the multipath effect is statistical estimation. The main techniques of estimation theory are least-squares estimation, maximum likelihood, and Bayesian techniques. The last two techniques usually require a complete statistical description of the problem variables in terms of joint probability distribution or density functions. In complicated multivariable problems the equations resulting from these two techniques are often nonlinear, difficult to solve, and impractical to implement. On the other hand, least-squares estimation only requires knowledge of the mean, variance, and covariance. When all variables have Gaussian statistics, these techniques produce linear equations, in which case each estimate is identical with that obtained by least-squares [15–17]. In GPS receiver tracking loops, the two observed parameters are the code phase, τ_o, and the carrier phase, ϕ_o . The carrier phase (PLL) and the code phase (DLL) tracking loops produce maximum likelihood estimates of these two parameters from measurements which are corrupted only by Gaussian noise [27]. However, in the presence of multipath, they provide erroneous (biased) estimates. Alike to this maximum likelihood estimator we will introduce another estimator depending on multiple correlators added to the tracking loops (carrier or code) in order to estimate the multipath separately. So we propose to reconfigure these loops with a multipath estimator to achieve the best estimate of the true tracking point in the presence of multipath. Figure 51 shows the proposed multipath approach for the code phase and carrier phase tracking.

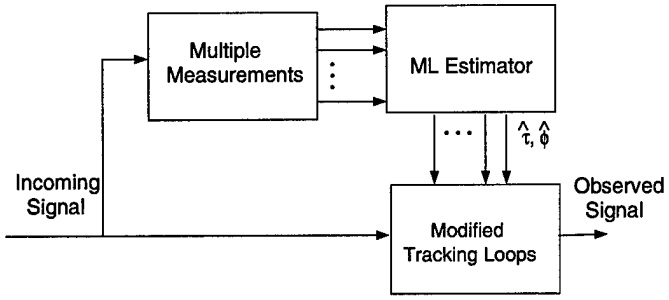


Figure 51 Multipath Approach

From Table 5, we notice that the incoming signal model in the presence of multipath, either at the correlator of the code phase tracking loop or at the multiplier-type phase detector of the carrier phase loop can be represented as a linear equation in parameters x or y, and u respectively. In the code tracking, the coefficients of the x's or y's parameters are autocorrelation functions (triangle shape), each is function in the multipath delay α's and the correlator spacing β. Ideally the physical equivalence of the multipath in the code tracking is a summation of triangles as a function of β shifted by α_i and scaled by x_i. For using a multiple observations, bank of correlators in case of code tracking under condition that the number of measurements greater than the number of multipath reflectors, then a construction of an estimator can be achieved. Namely, a least-square estimator in noise free environment or a ML estimator in the presence of noise (AWGN). In this chapter two method are developed for estimating and counting the multipath parameters: the search method and what so called the α-deploying method. A kalman filter is designed as a cascaded estimator to the α-deploying estimator to improve the estimation of the multipath in low signal-to-noise ratio. The Kalman filtering analysis, simulation and results are presented.

4.2 LS Algorithm (Noise Free Case)

The construction of this algorithm starts with Equation (50) such that it is repeated m times the number of observations, i.e., repeated per each correlator in the bank. Figure 52 shows the bank of correlators and the LS estimator. The received signal is correlated with a number of delayed code replicas $C(t-\tau_o+\beta_j)$; $j = 1, \cdots, m$ provided by the code tracking loop (see Figure 51). In this algorithm, we will use the notations "$\hat{}$" for the estimated parameters and "$\tilde{}$" for the measured parameters by the correlators. Whenever the tracking error is negligible (i.e.;$\tau_o \approx \hat{\tau_o}$), the in-phase correlation process, of Equation (50), can be written as

$$R_{Imp} = \sum_{i=0}^{n} x_i R_c \left(\beta_{e,p,l} - \alpha_i T_c\right) + \nu_I \tag{64}$$

Equation (64) contains $n+1$ values of x_i to be estimated as well as n values of α_i ($\alpha_o = 0$). This yields the linear regression

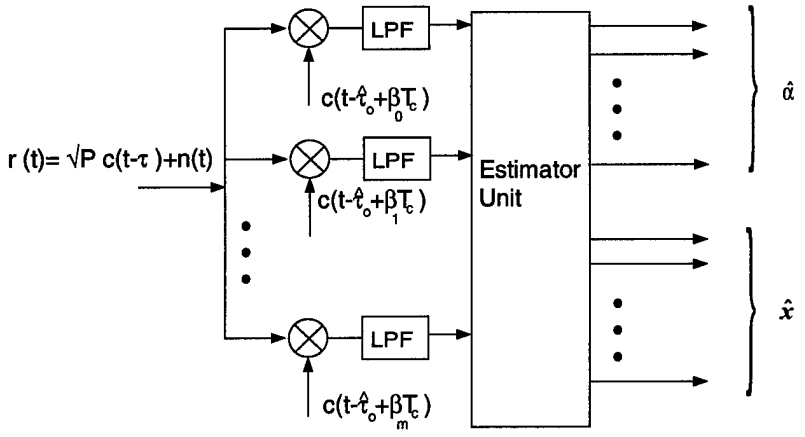


Figure 52 The Bank of correlators and LS Estimator

$$\tilde{R}_I = Hx+\nu \tag{65}$$

where $\tilde{R}_I$ is vector of $m+1$ observations with time delays $\beta_j; j = 0, 1, ..m.$ that is, $\tilde{R}_I = [R_o, R_1, \cdots, R_m]$ and x is a vector of the multipath signal parameters to be estimated, which is given by

$$x = \begin{bmatrix} x_o & \cdots & x_n \end{bmatrix}^T \tag{66}$$

The matrix H is the regressor matrix, formed for a vector of m observations, $\tilde{R}_I$, is given by

$$H = \begin{pmatrix} R_c(\beta_o) & R_c(\beta_o-\alpha_1) & \cdots & R_c(\beta_o-\alpha_n) \\ R_c(\beta_1) & R_c(\beta_1-\alpha_1) & \cdots & R_c(\beta_1-\alpha_n) \\ \vdots & \vdots & \ddots & \vdots \\ R_c(\beta_m) & R_c(\beta_m-\alpha_1) & \cdots & R_c(\beta_m-\alpha_n) \end{pmatrix} \tag{67}$$

and $\nu = [\nu_o, \nu_1, \cdots, \nu_m]^T$, is the noise vector (zero if no noise is presented).

In the modified tracking loop block of Figure 51, the number of correlators or multipliers will change according to the number of the estimated parameters, multipath time delays or multipath carrier phases. In the least-squares method, the problem formulation can be

posed as the best estimate of x that minimizes $\|\tilde{R}_I - Hx\|^2$. Mathematically, the solution of this problem is achieved by taking the derivative of this norm with respect to vector x and setting the resulting equation to zero. This yield an equation that the estimated vector $\hat{x}$ must satisfy and $\hat{x}$ is given in terms of the pseudo-inverse of matrix $H(\alpha)$ which is given by

$$\hat{x}(\hat{\alpha}) = (H^T H)^{-1} H^T \tilde{R}_I \tag{68}$$

Of course we assumed that $H^T H$ is invertible. since we consider the case of no noise, the LS estimator is interpreted as best fitting of our signal model of Equation (50).

The solution of this problem requires two steps to estimate the multipath parameters. The first is to estimate $\hat{\alpha}$ by substituting the expression of $\hat{x}$ in Equation (68) into $\|\tilde{R}_I - Hx\|^2$, then searching for the set of $\hat{\alpha}$ that minimizes the norm $\|\tilde{R}_I - H\hat{x}\|^2$. This is a nonlinear multidimensional function in the multipath parameters α. The second step entails the calculation of $\hat{x}(\hat{\alpha})$ which is determined by $\hat{\alpha}$, (see Equation 68). Unfortunately the number of multipath reflectors n is unknown. However the domain of the delayed time parameter α is known in the GPS viz $\alpha \in [0, 1.5T_c]$, which is considered as the best key to estimate the number of the reflectors, n. Two method has been developed to solve this problem, a search method and what we called "α-deploying method " the latter terminology is motivated by the deploying of the parameter α wherever the existence of the multipath reflectors is unlike the distribution of α uniformly on the entire domain $\alpha \in [0, 1.5T_c]$ for each recursion. The sequential quadratic programming optimization method can be used can be used, however in the case of multiple reflections it is inefficient.

4.2.1 Search Method. The search method entails the search of the delayed time parameters of the reflectors, α_i through its domain of $\alpha \in [0, 1.5T_c]$ and it also includes the search of the number of reflectors, n. Two trade-off parameters involved in this method, the resolution of the search, or the step size of the search, which effects the accuracy of the search and the time spent searching. The search algorithm is illustrated in flow chart of Figure 53. Appendix C.1 shows a simulation experiment when the search is performed in an example with a scenario of multipath parameters varying exponentially with time. The

results are not accurate at few points because the matrix $(H^T H)$ is close to being singular. The investigation illustrates that the columns of the matrix $(H^T H)$ are linearly dependent at points with small relative differences between the search vector α and the spacings vector β. The singularities can be avoided at these points by resetting the vector β at that point.

4.2.2 α-Deploying Method. In the α-deploying method, we are still using the bank of correlators as a multiple observers and the LS Algorithm in case of no noise (deterministic case) and ML estimator in case of existing noise. This method is useful for detecting the number of reflectors, n, (multipath components). Furthermore, the multipath parameters are estimated at each instant of observation. The time delay of multipath components, denoted by α vector, is deployed through its range $0 \leq \alpha \leq 1.5T_c$. In the bank of correlators, each correlator is driven by one replica delayed by β yielding m observations, $(m \geq n)$. This is motivated by the possible ability of the correlators' bank to highlight any multipath component effect on the deployed α around α_{true} embodied in $\hat{x}$ and $\hat{\alpha}$; viz., in the window between each chosen α and its adjacent α may lie on the deployed α itself. Mathematically, the algorithm can entails the on-time solution of an algebraic linear system of equations obtained from the output of the bank of correlators. The unknown vector in this linear system is x, and the solution of the linear system of equations is repeated at each instant of observation by the bank. Because of the link between x and α the existence of multipath components can be determined by knowing the estimated values of the multipath components' strengths denoted by $\hat{x}$, for each element of α-deployed vector. When the strength of a component is zero, this means that there is no multipath component at the corresponding deployed α. Thus, the number of multipath components in the environment can be determined by counting all nonzero values on the deployed α. The estimated multipath time delay of each reflector is determined by using the order of the $\hat{x}$ components according to the order of the deployed α. If the estimated multipath component is in-between the deployed α (i.e. not located on the chosen or deployed α), then the multipath component can be detected by determine the ratio between the estimated nonzero values before and after. This leads to redeploy α such that $\hat{x}$ coincide with x_{true}. Details of this redeployment will be later. The α-deploying method can be extended to be a simultaneous tracking technique of the

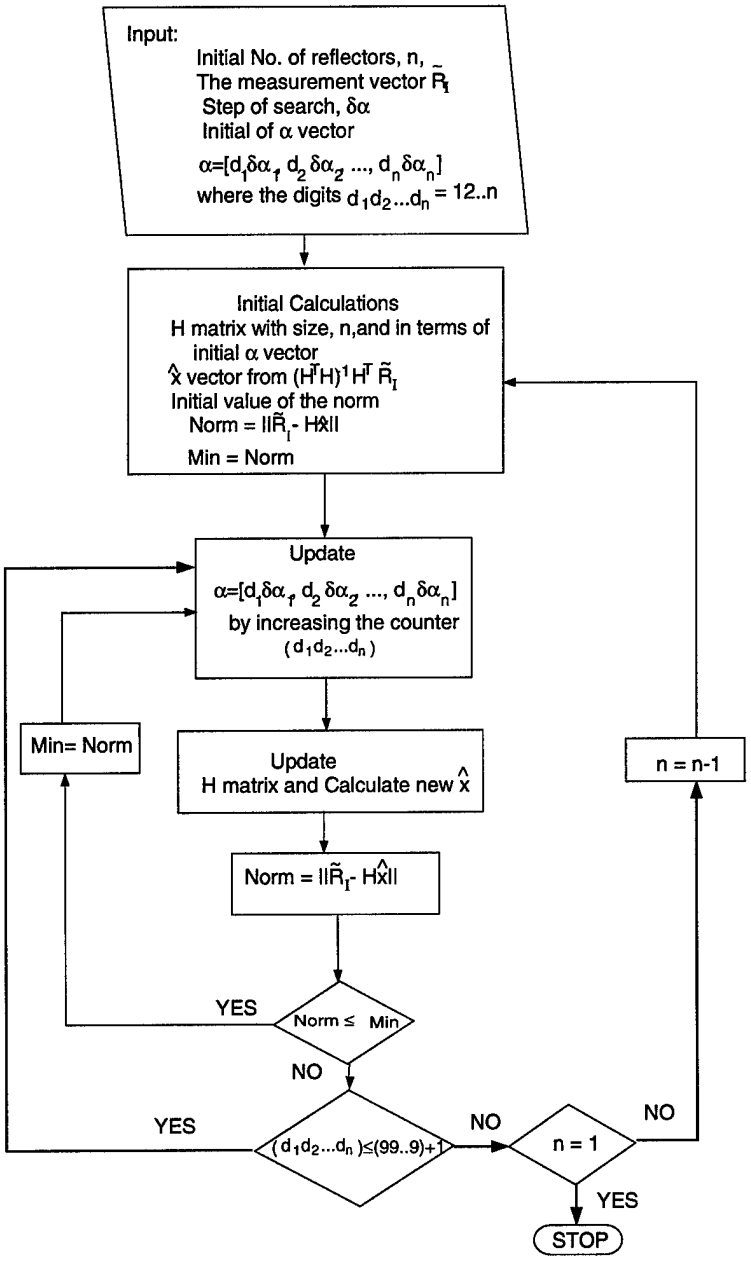


Figure 53 Search Algorithm

multipath components. To introduce this method, start with noise free case, then we will present the method under the limitations imposed by the presence of AWGN.

4.3 α-Deploying Method–Deterministic Case

The main sequence of α-deploying method includes a simultaneous multipath detection, identification, and tracking in the environment of the receiver. The detection of the multipath reflectors is achieved by considering the initial setting of the time delay vector α. The identification is done by redeploying α between two adjacent components of $\hat{\alpha}$ vector without including nonzero component in-between, and the tracking can be considered as the tracking of the multipath motion due to the relative motion between the receiver and satellites, and the satellites and reflectors. In this approach, We have assumed that the receiver is stationary. So the relativity of the dynamic multipath comes out only from the satellite motion, (detail will be later).

4.3.1 Initial Detection of Multipath. Initially α can be deployed uniformly through its range $0 \leq \alpha \leq 1.5T_c$ and discrete by $0.1T_c$ (see Table 6).

Table 6 Initial deploying of α.

order	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
α	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5

The purpose of using the initial setting of α is to detect initial locations of the existing multipath components in the environment, i.e., detect the initial presence of $\hat{x}$ imposed on the range of α between 0 and $1.5T_c$. By considering Equations (67), (68) and (8), and also considering the number of correlators, m, in the bank such that $m \geq n$ the algebraic linear system of equations can be written as

$$\begin{bmatrix} \hat{R}(\beta_o) \\ \hat{R}(\beta_1) \\ \vdots \\ \hat{R}(\beta_m) \end{bmatrix} = \begin{bmatrix} R_c(\beta_o) & R_c(\beta_o-0.1) & R_c(\beta_o-0.2) & \cdots & R_c(\beta_o-1.5) \\ R_c(\beta_1) & R_c(\beta_1-0.1) & R_c(\beta_1-0.2) & \cdots & R_c(\beta_1-1.5) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ R_c(\beta_m) & R_c(\beta_m-0.1) & R_c(\beta_m-0.2) & \cdots & R_c(\beta_m-1.5) \end{bmatrix} \begin{bmatrix} x_o \\ x_1 \\ \vdots \\ x_{16} \end{bmatrix} \tag{69}$$

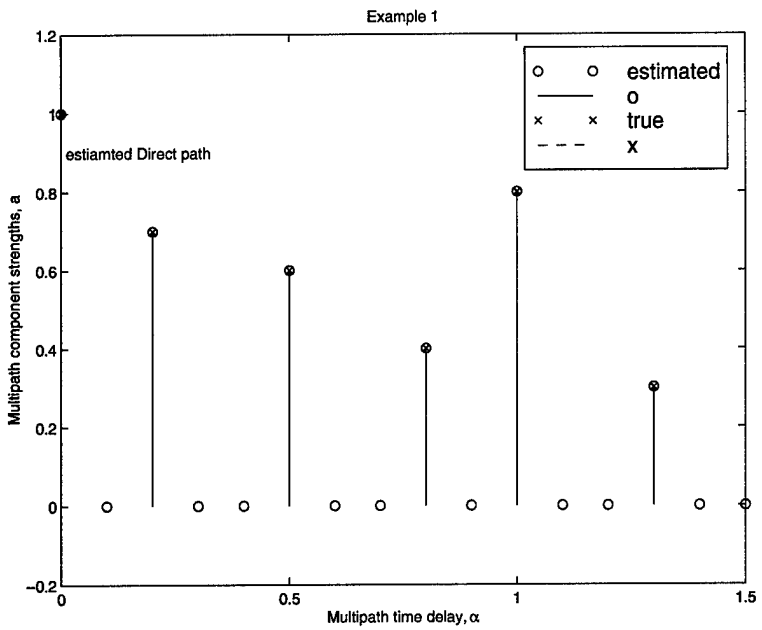


Figure 54 α-Deploying Method, Example 1

Example 1.

Suppose there exist multipath components at $\alpha_{true} = (0.0, 0.2, 0.5, 0.7, 1.0, 1.3)^T$ with strengths $x_{true} = (1, 0.7, 0.6, 0.4, 0.8, 0.3)^T$. Notice that $x(1.0) > x(0.7)$ which means, no further distribution model is specified. The strength of the direct signal component at zero delay time is taken as unity. It is required to estimate $\hat{x}$ and the corresponding $\hat{\alpha}$.

Solution

The solution of Equation (69) is Equation (68). After choosing a uniform set of β constructing our observations in the bank of correlators and using the code in Appendix C.3, the results is shown in Figure 54 and given by

$$\hat{x} = \begin{bmatrix} 1.0 & 0.0 & 0.7 & 0.0 & 0.0 & 0.6 & 0.0 & 0.4 & 0.0 & 0.0 & 0.8 & 0.0 & 0.0 & 0.3 & 0.0 & 0.0 \end{bmatrix}^T \tag{70}$$

Obviously, the result is accurate because of two reasons. Firstly, we have a noise free situation, and secondly, we assumed the existence of multipath components which completely coincide with the deployed α. The order of the multipath strength components can be determined by the order of the deployed α. Then, the number of multipath components can be determined by counting the number of nonzero elements in $\hat{x}$. The time delay for

each multipath component can also be estimated by calculating the order of each nonzero component (i.e., the order number of deployed α times the defined separation of deploying α which is taken $0.1T_c$ in this case). Indeed, we have a number of inferences from Example 1.

1. The significance of the preceding results because all the coefficients in the algebraic system of linear equations, (Equation 69), take places only for the nonzero elements of vector x_{true}.

2. No change to the preceding results happened with changing the deploying of α as long as α_{true} coincides with the deployed α. This property is true because of the nature of linearity of the autocorrelation function in Equation (8) (triangle shape) and the distribution of the correlator bank spacings vector β is equivalent to the distribution of deployed α.

3. The singularity of the matrix $H^T H$ is strongly related to the discretization values of both α and β vector, denoted by $\Delta\alpha$ and $\Delta\beta$. that is the separation value of the components of α (or the discretized value) must not be less than the separation value of β vector. i.e. $\Delta\alpha \geq \Delta\beta$. In Example 1, $\Delta\alpha = \Delta\beta = 0.1T_c$.

4. As it was known that the correlation function is a triangle shape in case of no multipath, however in the presence of multipath the triangle shape of the correlation function is distorted and becomes like a polygon shape or like a broken lines versus the delay time between the received PN code and the local replica. The bank of correlators can also represent the triangle shape in case of no multipath when the spacing vector β covers an entire one chip duration. Moreover when multipath is existed, the correlation function can be represented as a polygon shape over the range of β vector. Nonetheless each vertex of the polygon corresponds to a multipath component on the time delay β. Figure 55 shows the correlation function as a function of the spacing vector β of the bank in case of no multipath and in the presence of multipath.

   In Figure 55 the distribution of the spacings delay time β through the correlators in the bank not necessary to be in the negative part because the effective part of the multipath is in the positive side, consequently the work of our method will be included

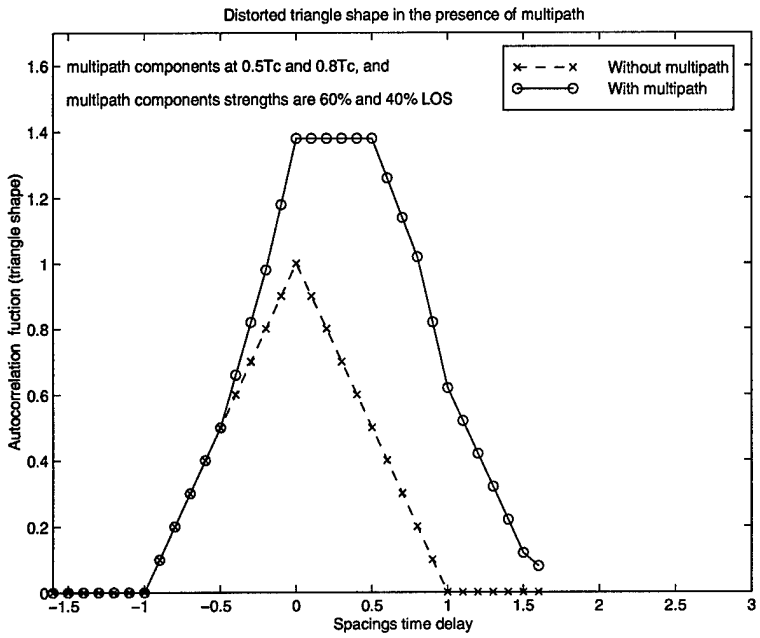


Figure 55 Correlation function vs spacings vector, β.

only in the positive part of β axis. Thus Figure 56 has to be considered in α-deploying method.

The slops of each line in the polygon plays a vital and an important role in the α-deploying method. Certainly each one of the direct path signal and the multipath components, constructs a triangle shape, and each one is shrinked or magnified according to the strength of the signal components. The superposition of all of these triangles constructs the shape of a polygon as shown in Figure 56. Subsequently each line slop in the polygon is the summation of two or more slops of other triangles according to the relative locations of the multipath components in the time delay interval.

The question now is what happens if we abandon the second assumption? that is if we suppose that the true existence of multipath components do not coincide with the deployed one. The answer will be in the next section.

4.3.2 Identification Algorithm of α. We have reached a method can detect the initial estimation of the multipath in Section 4.3.1. The current section is devoted to identify exactly where the places of multipath component strengths imposed on its true multipath

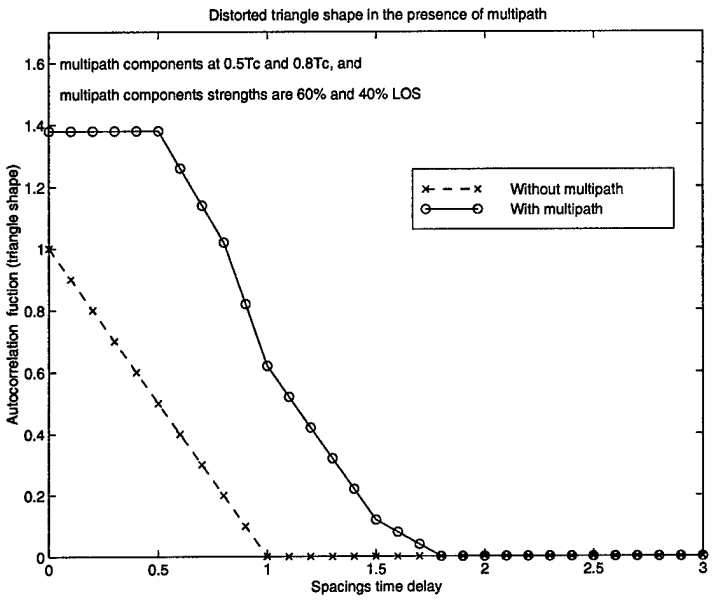


Figure 56 Correlation function vs spacings vector, β in the positive part.

time delay α_{true}. The setting of β vector in the bank such that the separation $\Delta\beta$ is small the higher accuracy of multipath detection and identification. Notwithstanding, the number of the correlators in bank should be increased, so far the identification algorithm as we are about in this section introduces a systematic method save the number of correlators used in the bank and a high bulk of computations devoleped. The identification algorithm is considered as a redeploying of the vector α using some criteria based on the previous estimation or as we mentioned, the initial estimation of the multipath, besides the nature of the polygon slops. Then, the question now is what happens if the existence of the true multipath components does not coincide with the deployed α. This answer is illustrated in Example 2.

Example 2

Suppose that $\alpha_{true} = (\ 0.0,\ 0.2,\ 0.57,\ 0.7,\ 1.05,\ 1.3,\ )$, i.e the true multipath component at 0.57 and 1.05 no longer coincide with the deployed α at 0.5 and 1.0 as in Example 1. What are $\hat{x}$ and $\hat{\alpha}$ when using the same algorithm in Section 4.3.1?

Solution

Figure 57 shows the results of Example 2 and the estimated multipath component strengths $\hat{x}$ is given in order as in 71. Notice that the estimate of $x_{true}(0.57)$ is divided into two parts at the ends of the interval $\Delta\alpha_{0.5}$ (i.e. at $\alpha_{0.5}$ and $\alpha_{0.6}$), and $x_{true}(1.05)$ is divided into two

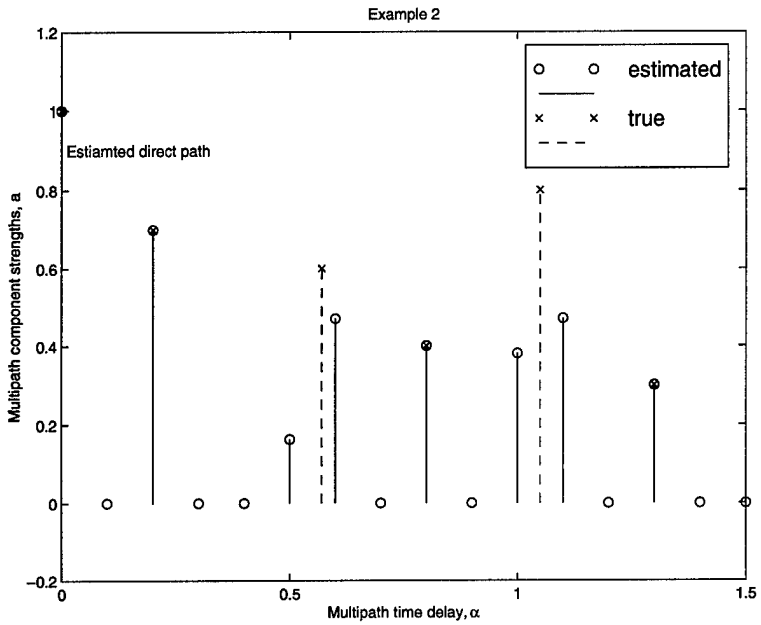


Figure 57 α Deploying Method, Example 2.

parts at the ends of the interval $\Delta\alpha_{1.0}$ (i.e. at $\alpha_{1.0}$ and $\alpha_{1.1}$).

$$\hat{x} = \begin{bmatrix} 1.0 & 0.0 & 0.7 & 0.0 & 0.0 & 0.18 & 0.42 & 0.4 & 0.0 & 0.0 & 0.4 & 0.4 & 0.0 & 0.3 & 0.0 & 0.0 \end{bmatrix} \tag{71}$$

Therefore, our inferences from Example 2 are:

1. If the true component of the multipath lies between two adjacently deployed α, then the estimate of this component is divided into two values and their ratio is equal to the ratio between the relative intervals of the time delay from the true delay α_{true} and these two adjacently deployed α. We illustrate this result.

$$\begin{aligned} \alpha_{0.57} &= \frac{\hat{x}_{0.6}}{\hat{x}_{0.5}+\hat{x}_{0.6}} \times 0.1+\alpha_{0.5} \\ &= \frac{0.42}{0.18+0.42} \times 0.1+0.5 \\ &= 0.57 \\ \alpha_{1.05} &= \frac{\hat{x}_{1.0}}{\hat{x}_{1.1}+\hat{x}_{1.0}} \times 0.1+\alpha_{1.0} \\ &= \frac{0.4}{0.4+0.4} \times 0.1+1.0 \\ &= 1.05 \end{aligned} \tag{72}$$

2. The true multipath component strength is predictable in such case. A criteria can be constructed in such situation similar to the results of Equation (72). Thus, if the true component lies in-between α_i and α_{i+1}, we can repeat the execution of the algorithm with the old values and the new redeployed α_i, now denoted by α_{in}, and given by

$$\alpha_{in} = \frac{x_{i+1}\Delta\alpha}{x_{i+1}+x_i}+\alpha_i \tag{73}$$

Where initially $\Delta\alpha = 0.1T_c$, and T_c is taken to be a unity. (We will call $\Delta\alpha$ an interval, so the entire time delay interval of multipath is 16 intervals).

3. Next, if the true value actually exists, then the nonzeros of $\hat{x}_i$ and $\hat{x}_{i+1}$ will disappear and the estimated components will transfer to $\hat{x}_{in}$ only. If it does not exist, then $\hat{x}_{in}$ and $\hat{x}_i$ will change linearly in α_{in}.
4. If two or more multipath components lies in a single interval, $\Delta\alpha_i$, the results at the end of this interval will be the weighted average of all components inside that $\Delta\alpha_i$.

Hence, we can redeploy α in the regions where the nonzero multipath strength components has appeared. Figure 58 shows the results of Example 2 after the redeploying process has been applied. The usefulness of this results is obvious, as the more α redeploying is undertaken, the higher the accuracy of the estimated multipath parameters. Thus, the number of multipath components can be determined, and, in addition, the strength and time delay for each component can be estimated.

4.3.3 Analysis of the α-Deploying Method. This section is devoted to introduce a deeper analysis in the nature of the α-deploying method. The objectives intended are to investigate the method with redeploying the time delay parameter α only without changing the vector β. We start this section to illustrates the results if there is one or more multipath components in the same interval, then, the section includes another idea of using the previous estimation of $\hat{x}$, $\hat{\alpha}$ or the previous deployed α to be used as a creative measurement or a next generation of a creative bank beside the actual bank. This idea can be useful to reduce the number in case of existed limits number of correlators in the bank.

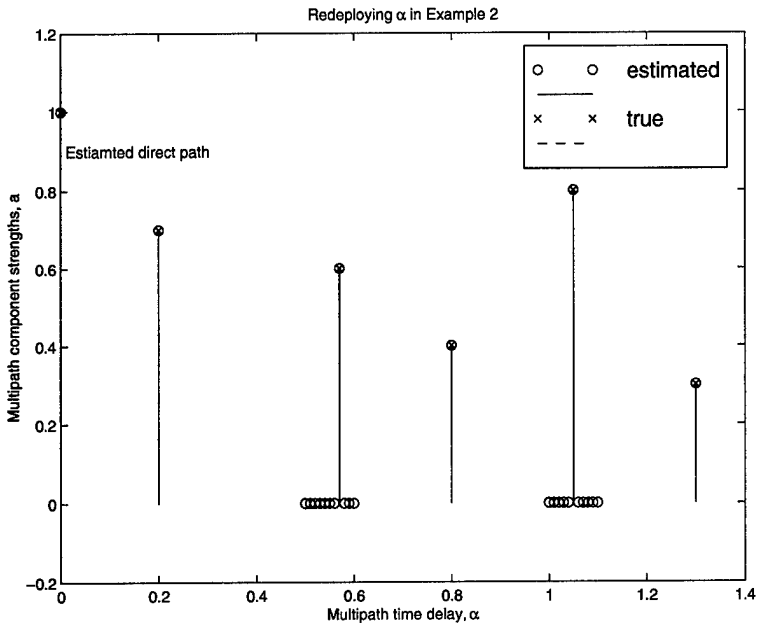


Figure 58 Redeploying of α at $0.01T_c$

Example 3.

When three components $(\ 0.7 \quad 0.6 \quad 0.8\ )^T$ are located in $\Delta\alpha_{0.3}$ as $(\ 0.32 \quad 0.34 \quad 0.37\ )^T$ then Figure 59 shows the results of the initial detection. By using the criteria in Equation (73) we get $\alpha_{in} = 0.34476$. Certainly the result is $x_{in} = 2.1$ which is the summation of the true values (see Figure 60).

The conclusions of example 3 are two fold:

1. No clues can be determined to specify what the multipath components inside each panel except those two estimated on the ends of such intervals.
2. The estimated strengths of the direct path signal and the multipath components are distributed over the the deployed α with the proportional of the relative intervals among them.

- Therefore, if the determination of multipath is necessary per each interval contains multipath in the time delay interval then the criteria in Example 2 cannot be used, and the best way to identify those multipath components is to transfer all measurements in the bank into the inside of the interval that have two or more multipath components as

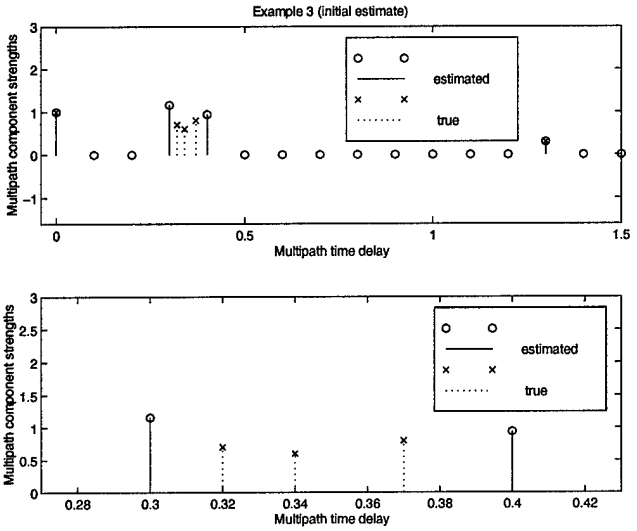


Figure 59 Three Multipath component in one panel, $\Delta\beta_{0.3}$

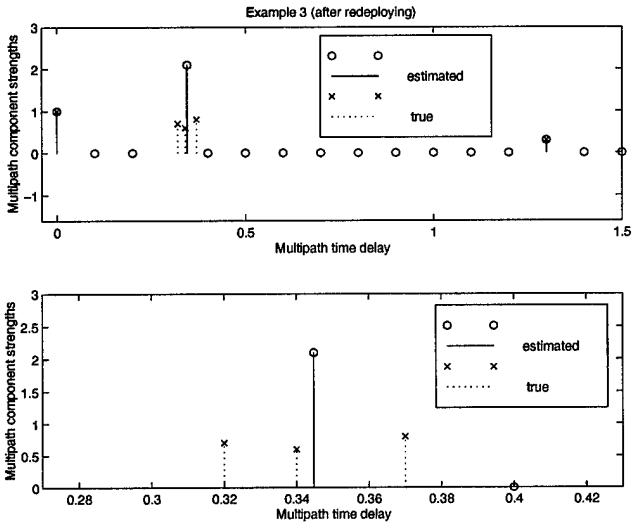


Figure 60 Three true component equivalent to one inside one panel, $\Delta\beta_{0.3}$

well as redeploying α inside such intervals and achieving the process again until a fixed number of multipath components is reached. The main functions of the tracking loops is to track the direct path signal avoiding the effect of any multipath components. Thus the main function of the multipath estimation is to separate the multipath components from the direct path. This seperatation can be explained also as a way of extracting the direct path signal only from the multipath whatever the description or the specification of the multipath. Therefore, it is not necessary to detect the exact number of the multipath components and the true locations in the time interval during the tracking loop process, but we must look for some tools to help extract and to track the direct path signal separately (this point left for future research). Indeed, by the second conclusion of Example 3, the estimation of the direct path signal can be achieved if we make an intensive measurements and deploying α in the first interval, $\Delta\alpha_{0.0}$, i.e.. from 0 to $0.1T_c$.

- As previously mentioned, if $\Delta\alpha$ becomes finer (i.e. smaller), then the estimation of the multipath parameters will become more accurate, but this requires a huge number of correlators because the increase of the deployed α vector length implys an increase in the number of unknowns, n, assumed in Equation (69) which logically must be less than the number of measurements, m. Figure 61 shows the results when deployed α vector with $\Delta\alpha = 0.01T_c$ as well as $\Delta\beta = 0.01T_c$. The result in Figure 61 is exactly x_{true} because the digit number of $\alpha_{0.57} = 0.57$ is rounded for accuracy of two decimal digits, that meet the finer accuracy.

- If the nonzero estimated components lies between two zero estimated component, then this means that the nonzero estimated component is the true value of the multipath strength x_{true} and lies on the true time delay α_{true}. But, Perhaps this result is useful only when the true strengths of the multipath components x_i are positive. However a significant way to guarantee that x_i be positive, is by redefining the elements of the H matrix in Equation 67 as $R_c(\beta_j - \alpha_i)\cos(\phi_o - \phi_i)$ and let $x_i = a_i A_o$ instead. The matrix H still keeping its non-singularity by this new definition because the autocorrelation function R_c and the cosine function are even functions, in turn its product is even as

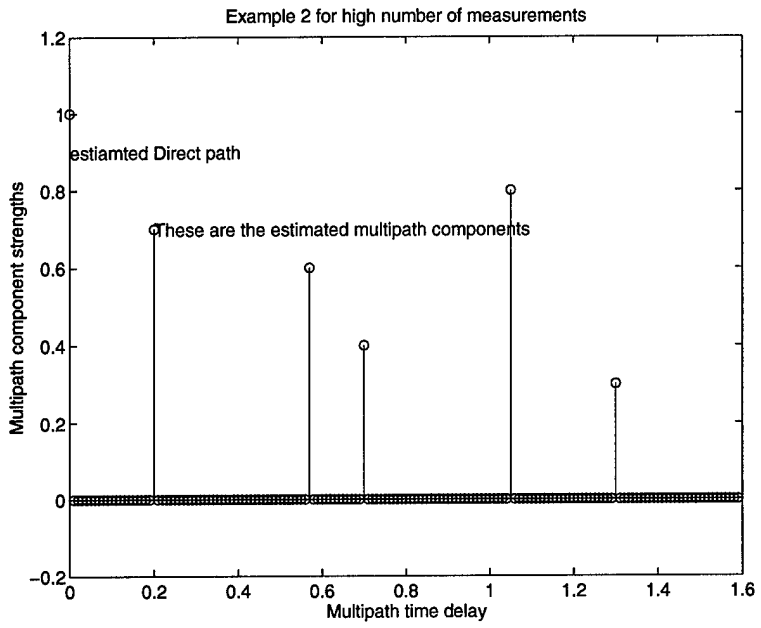


Figure 61 Example 2 with increasing the number of measurements

well. Same results of Example 1 and Example 2 are obtained with this change. Notice that $\phi_i - \phi_{pll} = 2\pi f_c \alpha_i$.

- An important demonstration of the α-deploying method can be obtained by redrawing the polygon curve using the initial estimate of the multipath components. Figure 63(a) shows the plotting of the polygon curve constructed from the true measurements and the creative polygon constructed from the estimation values of the multipath parameters. In this application we take x_{true} and α_{true} as in example 2. Figure 62 illustrates that the creative polygon is completely coincide with the measured one except those intervals $\Delta\beta_i$ contains multipath components inside. Figure 63(a) and 63(b) illustrates the zoom on the part of polygon contains x_{true} which is located in-between deployed α or as we said 'intervals'. In example 2 $x_{0.57}$ located on $\Delta\alpha_{0.5}$ and $x_{1.05}$ located on $\Delta\alpha_{1.0}$. It has been shown that the value of creative polygon at the estimated point $\alpha_{0.3}$ and $\alpha_{0.5}$ lead the slop of actual lines toward the inside of interval $\Delta\alpha_{0.5}$ or $\Delta\alpha_{1.0}$. Also Figure 64(a) and Figure 64(b) shows the polygon curve of Example 3 in the initial estimate and after deploying α at $\alpha_{in} = 0.34476$, in these Figures that notice the three components inside interval $\Delta\alpha_{0.3}$ make three broken lines the two line outside can be

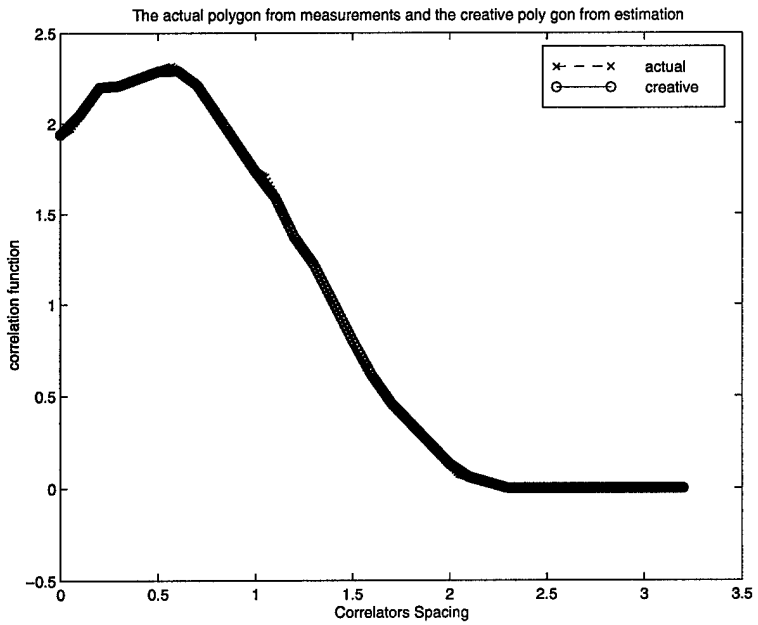


Figure 62 The actual and the creative polygon curves.

leaded by the estimated points at $\alpha_{0.3}$ and $\alpha_{0.4}$ respectively. Also in Figure 64(b) the creative polygon not exactly coincide with α_{in}, because the finer accuracy of this plot is taken 0.01 which a rounded number for two decimal digits that not meet the accuracy of α_{in} that rounded to 5 digits after the decimal point.

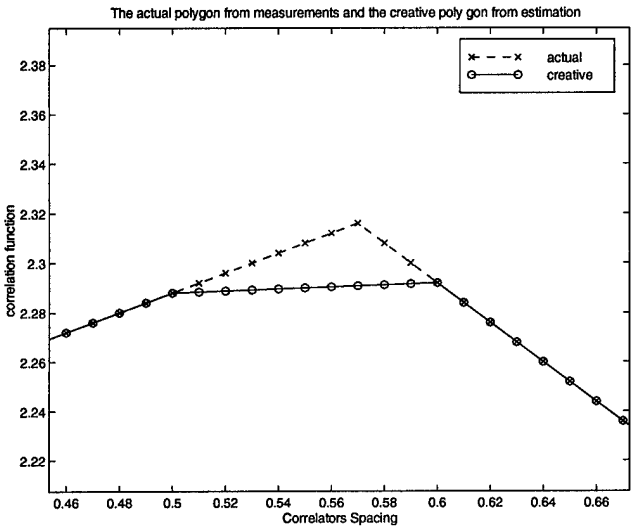
- In Example 2 the criteria in item 1 can be also created from the intersection of the outside lines of the intervals $\Delta\alpha_{0.5}$ or $\Delta\alpha_{1.0}$ as the following:

  - The slopes of those lines can be calculated from the line reached between Cartesian points at $(\beta_{0.5}, R(\beta_{0.5}))$ and $(\beta_{0.51}, R(\beta_{0.51}))$ and the line reached between $(\beta_{0.6}, R(\beta_{0.6}))$ and $(\beta_{0.59}, R(\beta_{0.59}))$. Advantageously, those Cartesian points can be taken from the creative polygon.

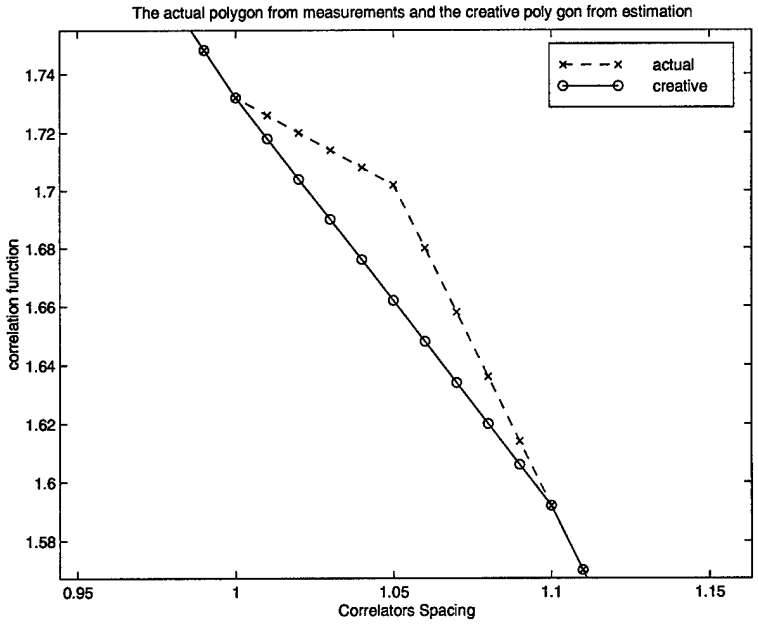
  - The point of intersection can be used as redeployed α to confirm the presence of the multipath component inside the interval.

Thus, a big advantage obtained from using the initial estimate of multipath parameters to construct a creative polygon.

The aspects of the creative polygon are:

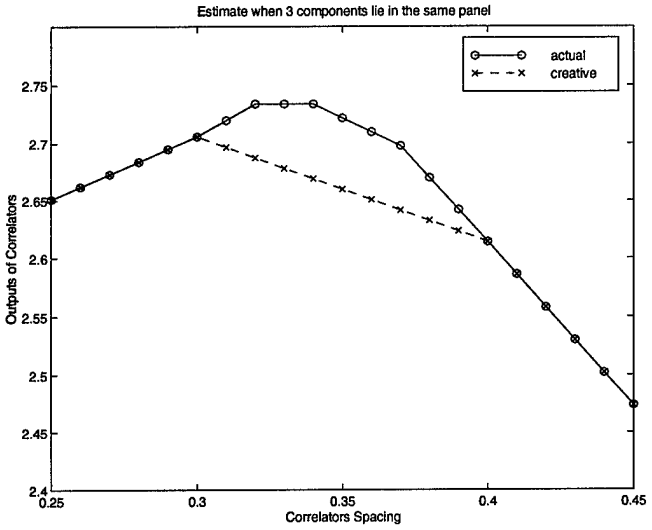


(a)

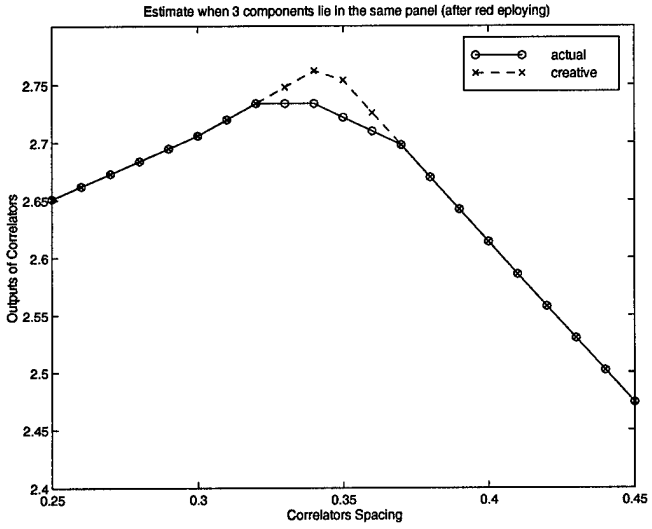


(b)

Figure 63 (a) zoom at 0.57Tc and (c) zoom at 1.05Tc



(a)



(b)

Figure 64 (a) Polygon curve of example 3 (initial estimate), (b) Polygon curve of example 3 (after redeploying) into α_{in}.

- The parts of the creative polygon are coincided with the actual one as long as its outside points are estimated with accuracy meets the true location.
- In the inside of those intervals have multipath components, yet not be estimated exactly, if the redeploying of α is coincided with the true location of α then the creative polygon will coincide with the actual one without the need of the extra measurements inside those intervals again (i.e. resetting β in the bank with values inside those intervals).

Therefore the creative polygons is also useful, if we let the tracking of the signal be achieved in the operating regions such that the true polygon is coincide with the creative polygon and avoiding the in-coincidence parts (details will be later). At all events the deploy of α is achievable into the true one when α deploying meet the β transfer at the overall intervals of the delay time contains multipath to be estimated.

Example 4

Suppose there exist two true components which lie in successive intervals $\Delta\alpha_{0.4}$ and $\Delta\alpha_{0.5}$, $\alpha_{true} = (\ 0.0,\ \ 0.2,\ \ 0.46,\ \ 0.57,\ \ 1.3\ )^T$ and $x_{true} = (\ 1.0,\ \ 0.7,\ \ 0.6,\ \ 0.8,\ \ 0.3\ )^T$. What is the estimated multipath components?

Figure 65 shows the initial estimate of the multipath. The estimation of the two values lie in the two successive intervals $\Delta\alpha_{0.4}$ and $\Delta\alpha_{0.5}$, are three value located on $(0.4,\ \ 0.5,\ \ 0.6)$.

The polygon shape for the creative and the actual measurements are shown also in Figure 65. The results of example 4 show the following

1. The true multipath components at $\alpha_{0.46}$ and $\alpha_{0.57}$ are divided into estimated components at $\alpha_{0.4}$, $\alpha_{0.5}$ and $\alpha_{0.6}$ respectively.
2. In the polygon curve the outside slops from $\beta_{0.4}$ and $\beta_{0.6}$ can be determined.
3. There is no clue can help to determine the slops into/or from $\beta_{0.5}$.
4. The shift of $\alpha_{0.4}$ and $\alpha_{0.6}$ into the inside of the intervals implys increase of the outside estimated components, i.e. from $\alpha_{0.4}$ and $\alpha_{0.6}$ into the inside of $\Delta\alpha_{0.4}$ and $\Delta\alpha_{0.5}$ respectively. Furthermore the shift implys a decrease in the middle estimated component at $\alpha_{0.5}$.

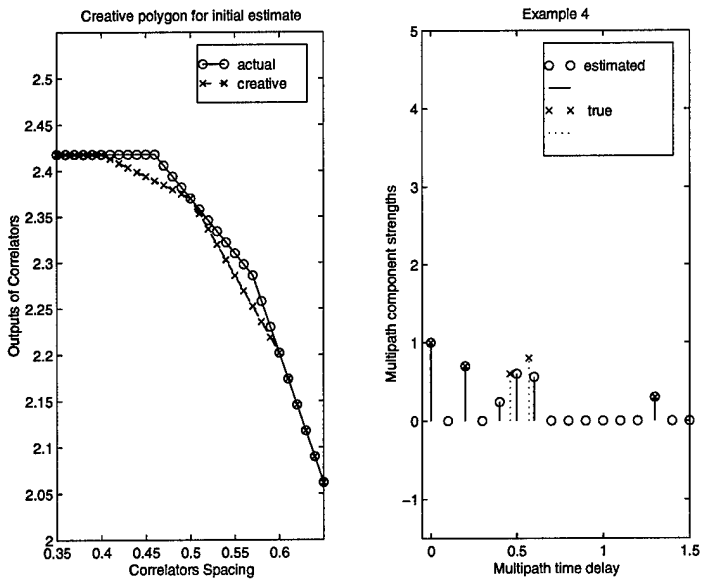


Figure 65 Example 4

5. When the results of the shift get the middle estimated component at $\alpha_{0.5}$ zero, then the point of the shift at the outside components are the true estimate of multipath inside the interval, Figure 66 through 68 shows the steps of the estimated components to the inside of the interval until settling on the true location.

6. When the result of the shift get the middle component negative, then it means that the shift of either the outside components exceeds the true estimate locations. Therefore the middle components at $\alpha_{0.5}$ is the key to recognize an accurate estimate of the multipath components in such case.

In spite of shifting α inside the interval as proceeded in example 2, 3, and 4 respectively, the best way to recognize the multipath inside those intervals bounded with two nonzero estimate components, is to redeploy the measurements and the parameter α inside those interval and continuing in this process until the multipath is completely identified. By this process the interval size can be changed from $0.1T_c$ to $0.01T_c$ to $0.001T_c$ and so on. Thus the more redeploying the more accuracy of identifying multipath.

The identification algorithm can be concluded as :

1. Run the algorithm in Section 4.3.1

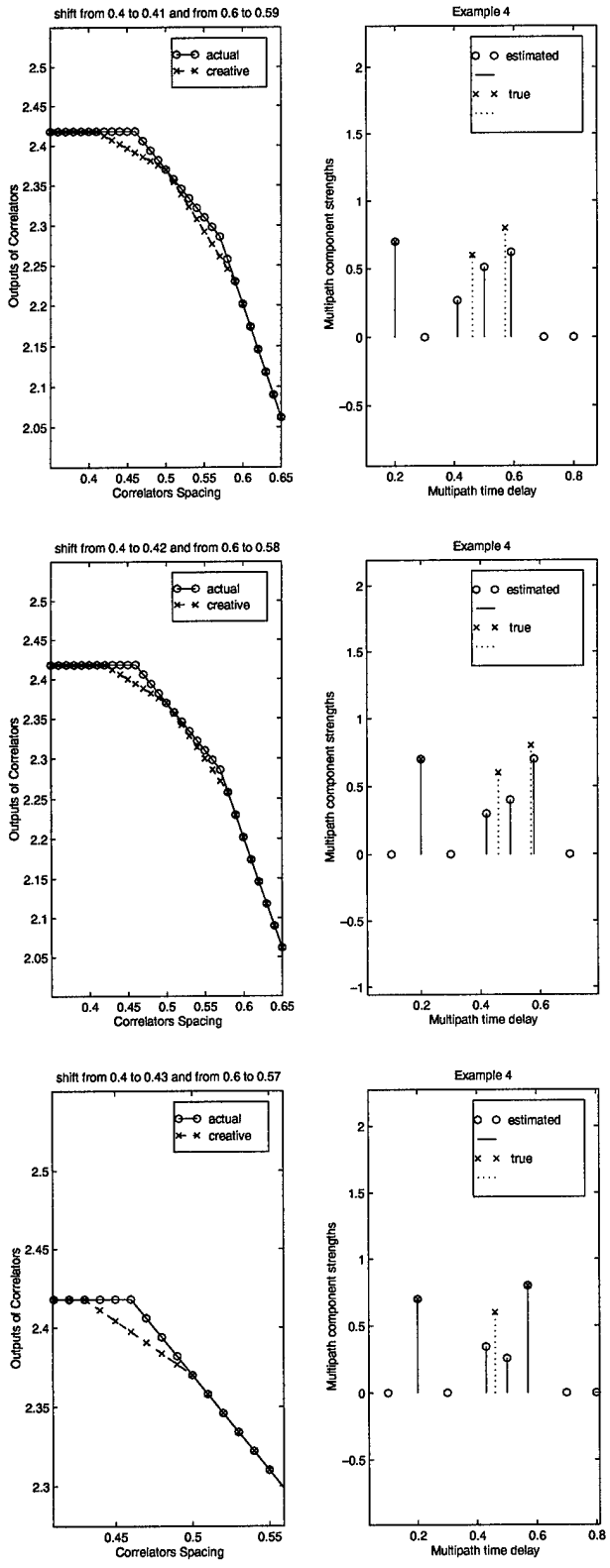


Figure 66 the steps of the estimated components to the inside of the interval until settling on the true location.

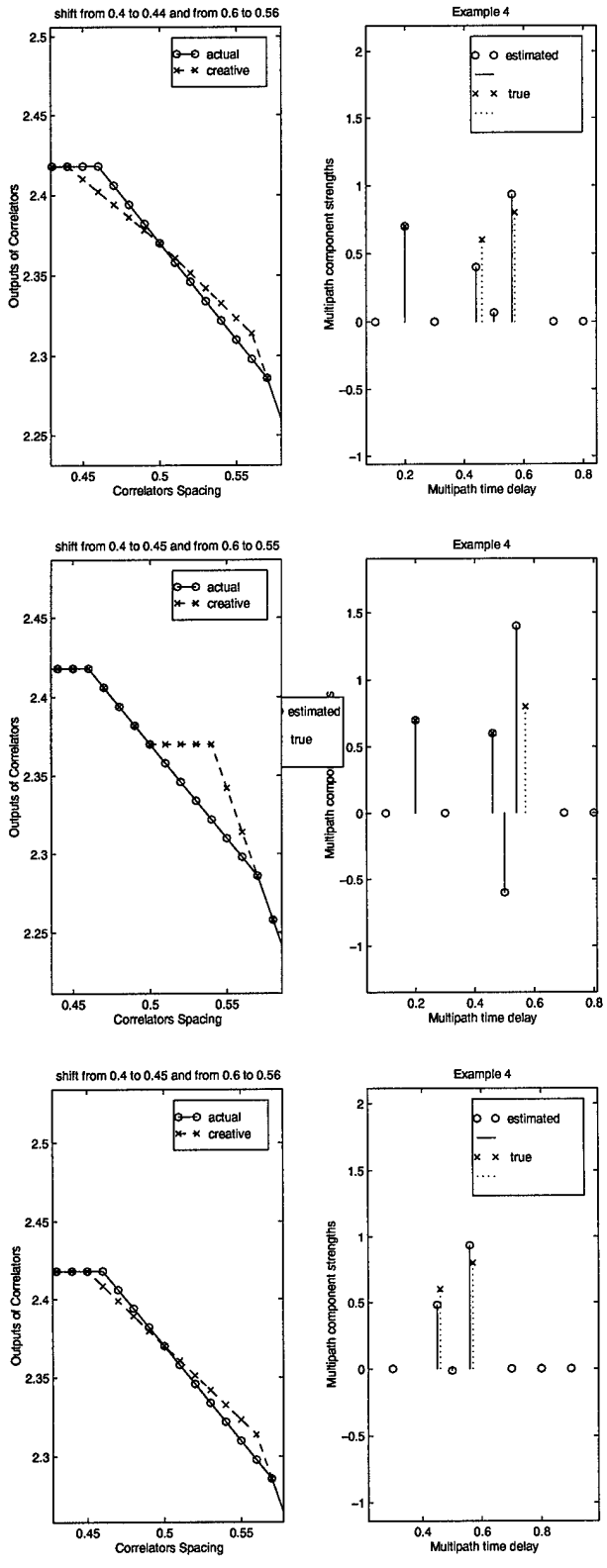


Figure 67 the steps of the estimated components to the inside of the interval until settling on the true location.

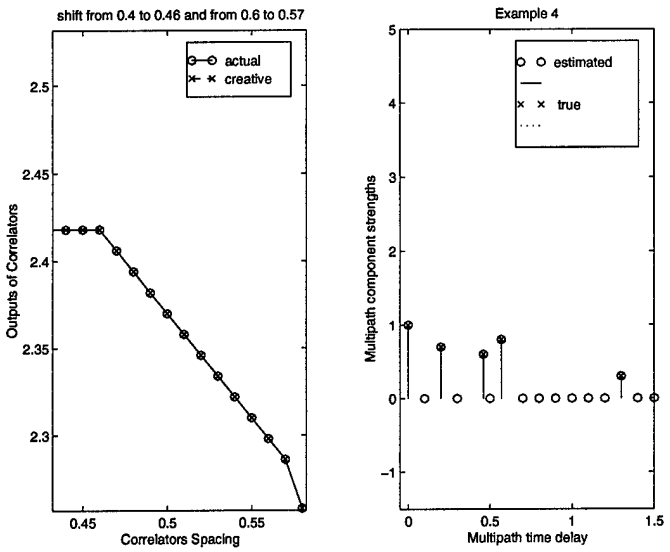


Figure 68 the steps of the estimated components to the inside of the interval until settling on the true location.

2. If there are zero values before and after the detected components count them as a true multipath.
3. If there is two or more successive nonzeros components without zero in-between, then count the corresponding interval $\Delta\alpha$ component.
4. Redeploy the measurements for those counted intervals by setting the correlators such that β_i settled inside those counted intervals
5. Redeploy α in the nonzero $\hat{x}$ and at those counted intervals also.
6. run the program in item 1 again with the new parameters in item 4 and 5.
7. Compare the number of estimated multipath components with the previous run. If no change stop, if a change occurred return into item 2

Notwithstanding, one run of the program can be useful also, if our intention is to separate the direct path signal from multipath without any care of the multipath recognition or specification. The necessary separation can be achieved by concentrating the deploying of most of measurements in the first interval $\Delta\alpha_{0.0}$ and the rest of the time delay interval would be as in example 1.

The α-Deploying Method–Additional Insights

Four rules are established to achieve a high accuracy in the α-deploying algorithm.

1. The first α deployment must be uniformly distributed in the time delay interval of $0 \leq \alpha \leq 1.5T_C$ in order to discover any multipath components in this interval.
2. The order of the deployed α determines the order of multipath components.
3. If a nonzero estimated multipath component lies between two zero estimated multipath components, then the nonzero answer yields a multipath component.
4. If the estimated nonzero multipath component is preceded, and/or followed by a nonzero component, then the number of multipath components is equal to these estimated components or less; and the exact number can be reached if the redeploying of both the α and β vectors is repeated around these nonzero components by cyclings through $0.1T_c$, $0.01T_c$, $0.001T_c$, ... etc, until the number of estimated components becomes constant. Then the number of estimated multipath components is the correct answer.

Now, We have arrived at the logical question: what is the behavior of α-deploying method in the presence of AWGN? The answer will be in next section.

4.4 α-Deploying in the Presence of AWGN

The α-deploying method in the presence of AWGN is also applicable, whenever there is some limits imposed by the Signal-to-Noise Ratio (SNR) and Multipath-to-Noise Ratio, (MNR). Since the α-deploying method entails a linear transformation of the input noise, a useful separation between the estimated multipath and noise can be achieved. In this section we introduced a Kalman filtering technique able to mitigate the effect of AWGN in multipath. In addition, the performance analysis of the Kalman filter is validated by Monte Carlo simulation. This section also includes a design model of shaping filter which represents the noise model of the LPFs, followed by the bank of correlators. The Kalman filter algorithm has been applied in the case of a stationary receiver.

4.4.1 Noise Performance in the Bank of Correlators. The AWGN corrupting the received signal has a two-sided power spectral density (PSD) of $\frac{N_o}{2}$ W/Hz , the width of the two sided PSD varies according to the bandwidth of the received signal. The bandwidth of the received signal is shrunk into the baseband of each part in the receiver that the signal has been passed. The SNR is evaluated at each baseband and it is defined as

$$SNR = (\frac{P}{N_o})\frac{1}{BW} \tag{74}$$

where P is the coherent received signal power, BW is the bandwidth of the received signal. The correlator process can be considered as a code multiplier multiplying the received PN code and the local replica generated in the receiver, beside an integration process to average the result of this multiplier. The output noise from the j^{th} correlator in the bank $\nu_j(t)$, is a low-pass AWGN by the transfer function of the integrator of the correlator (see Figure 52). Then $\nu_j(t)$ is given by

$$\nu_j(t) = n(t)c\,(t-\tau_o-\beta_j T_c) * h_{lpf}(t) \tag{75}$$

where, $n(t)$ is defined in Equation (47), the index j ranges from 1 to m, and m is the number of correlators in the bank. h_{lpf} represents the impulse response of the integrator, namely, the low-pass filter. The operator * denotes convolution. The autocorrelation function for each correlator per Equation (75) is given by

$$\begin{aligned} R_\nu(\tau) &= E\{\nu_j(t)\nu_j(t+\tau)\} \\ &= E\{[n(t) * h_{lpf}(t)][n(t+\tau) * h_{lpf}(t+\tau)]\} \times E\{c\,(t-\hat{\tau}_o-\beta_j T_c)\,c\,(t+\tau-\hat{\tau}_o-\beta_j T_c)\} \end{aligned} \tag{76}$$

Because $\delta(\tau)$ is zero for all $\tau \neq 0$ then

$$R_\nu(\tau) = E\{[n(t) * h_{lpf}(t)][n(t+\tau) * h_{lpf}(t+\tau)]\}E\{c^2\,(t-\hat{\tau}_o-\beta_j T_c)\} \tag{77}$$

Since $c^2 = 1$ then the autocorrelation function is

$$\begin{aligned} R_\nu(\tau) &= E\{[n(t) * h_{lpf}(t)][n(t+\tau) * h_{lpf}(t+\tau)]\} \\ &= \tfrac{N_o}{2} h_{lpf}(\tau) \end{aligned} \tag{78}$$

The PSD is obtained after taking the Fourier transform of Equation (78). Thus, the PSD of the noise $\nu_j(t)$ for each correlator is $\frac{N_o}{2}$ flatten in the range of frequency $[-B_{lpf}, B_{lpf}]$ and B_{lpf} is the bandwidth of the integrator, or the low-pass filter, followed by the correlator in the bank (see Figure 52). The period of integration in the integrator (i.e., $T_i = 1/B_{lpf}$) must be chosen such that $T_i \gg T_c$. that is, it must be much greater than the chip period. In turn, the variance per each correlator is given by

$$\sigma^2 = \frac{N_o}{2}\, Watt/Hz \tag{79}$$

Also there is a cross-correlation between correlators in the bank which can be given as follows. The cross-correlation between the i^{th} and the j^{th} noise correlator is given by

$$\begin{aligned} R_{\nu_{i,j}}(\tau) &= E\{\nu_i(t)\nu_j(t+\tau)\} \\ &= E\{[n(t) * h_{lpf}(t)][n(t+\tau) * h_{lpf}(t+\tau)]\} E\{c\,(t-\hat{\tau}_o-\beta_i T_c)\, c\,(t+\tau-\hat{\tau}_o-\beta_j T_c)\} \\ &= R_\nu(\tau) R_c\,(\tau+\Delta\beta_{ij} T_c) \end{aligned} \tag{80}$$

where $\Delta\beta_{ij} = \beta_i - \beta_j$. Again setting $\tau = 0$, and because $\delta(\tau)$ is zero for all $\tau \neq 0$ then the noise autocorrelation function can approximated as

$$R_{\nu_{i,j}}(0) \approx \sigma^2 R_c\,(\Delta\beta_{ij} T_c) \tag{81}$$

So far the noise covariance matrix of the vector $\nu = [\nu_o, \nu_1, \cdots, \nu_m]^T$ results as [14]

$$C_\nu = \sigma^2 P_\nu \tag{82}$$

where P_ν is the $(m \times m)$ correlation matrix

$$P_\nu = \begin{pmatrix} 1 & R_c(\beta_o-\beta_1) & \cdots & R_c(\beta_o-\beta_{m-1}) \\ R_c(\beta_1-\beta_o) & 1 & \cdots & R_c(\beta_1-\beta_{m-1}) \\ \vdots & \vdots & \ddots & \vdots \\ R_c(\beta_{m-1}-\beta_o) & R_c(\beta_{m-1}-\beta_1) & \cdots & 1 \end{pmatrix} \tag{83}$$

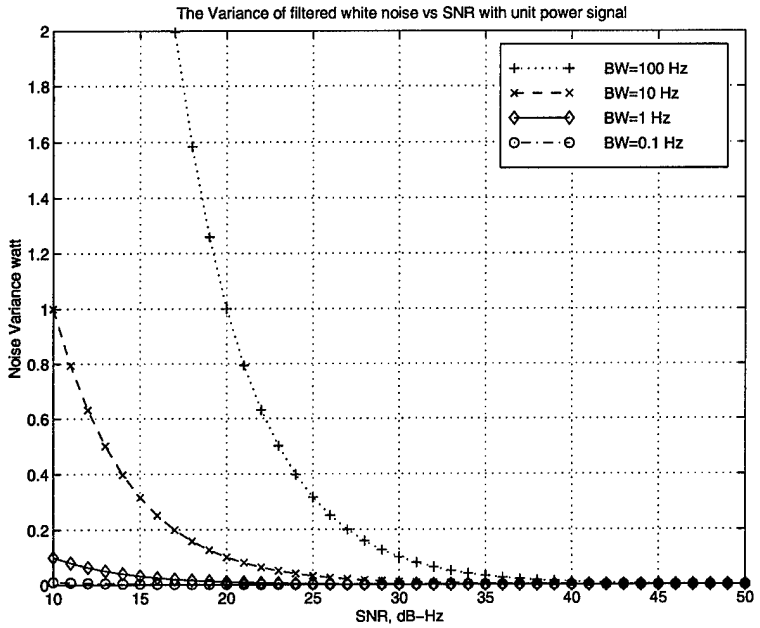


Figure 69 Noise variance of the low-pass filters with BW=100, 10, 1, 0.1 Hz.

Figure 69 shows the noise variance of the output of the low-pass filter with BW=100, 10, 1 and 0.1 Hz, respectively, versus the input SNR ranges from 10 to 50 dB-Hz when $P = 1$ Watt.

The importance of the curves in Figure 69 returns to the trade-off between the bandwidth of the low-pass filter and the input SNR which control the variance of the AWGN. Obviously, the curve of $B_{lpf} = 0.1Hz$ is the best one for all the range values of SNR. Curves for $B_{lpf} = 1.0$, 10, 100 and 1000 Hz asymptotically approach the curve $B_{lpf} = 0.1Hz$ at $SNR = 18$, 28, and 38 dB-Hz, respectively.

4.4.2 Maximum Likelihood Estimator for Multipath Parameters. The ML estimator corresponds to the least-squares estimator that has been used in Section 4.2 whilst the linear model includes the noise process vector $\nu(t)$. The MLE principle is based on the likelihood function developed from the pdf of the underlying parameters. The observed parameter in Equation (65) is the vector $\tilde{R}_I$, the unknown parameter is vector x, and the mean of the observed parameter is Hx. The covariance matrix of the noise process vector $\nu(t)$ that is deduced by Equation (82), reflects the uncertainty in the observed vector $\tilde{R}_I$, that

is, distributed as $\mathcal{N}(Hx, C_\nu)$ and the correlator noise outputs $\nu(t)$ vector is distibuated as $\mathcal{N}(0, C_\nu)$. Then the pdf of the observed vector $\tilde{R}_I$ given the unknown parameter x is represented as

$$f(\tilde{R}_I|x) = \frac{1}{(\sqrt{2\pi})^m det(C_\nu)^{1/2}} \exp\{-\frac{1}{2}[\tilde{R}_I - Hx]^T C_\nu^{-1}[\tilde{R}_I - Hx]\} \tag{84}$$

The likelihood function is defined as

$$\mathcal{L}(\tilde{R}_I, x) = \ln f(\tilde{R}_I|x) \tag{85}$$

Then, the maximum likelihood estimate is given as [24]

$$\hat{x} = (H^T P_\nu^{-1} H)^{-1} H^T P_\nu^{-1} \tilde{R}_I \tag{86}$$

Notice that σ^2 is canceled in Equation (86), $\hat{x} = \hat{a}$ when we consider the cosines in the elements of H matrix (See Example 2).

Example 5.

Examine the estimates of multipath parameters using the α-deploying method, when the low-pass filter bandwidth taken as $B_{lpf} = 1$ kHz to 0.1 Hz and SNR ranges from 0 to 50 dB-Hz. Suppose multipath components represented as $\alpha_{true} = (0.0, 0.2, 0.5, 0.7, 1.0, 1.3)^T$ with attenuation coefficients $a_{true} = (1.0, 0.7, 0.6, 0.4, 0.8, 0.3)^T$.

To study the whole ranges of both SNR from 0 to 50 dB-Hz and the B_{lpf} from 0.1 Hz to 1 kHz let us pick up some discretized values of corresponding variance of noise that is plotted in Figure 69. Table 7 summarizes the values of the SD of the noise in the chosen values of B_{lpf} at 0.1, 1.0, 10, 100 and 1000 Hz and discretized values of SNR at 0, 10, 20, 30, 40 and 50 dB-Hz, respectively.

All the values of σ in Table 7 are symmetric and can be ordered as 31.6228, 10.000, 3.1623, 1.0000, 0.3162, 0.1000, 0.0316, 0.0100, 0.0032, 0.0010. The values of SD $\sigma \geq 1$ will be discarded, because the noise level is higher than the signal level in such cases and the multipath components will disappear at these levels, so no characterization of multipath can

Table 7 Standard deviation, σ of correlator output noise

B_{lpf} Hz SNR db-Hz	0.1	1.0	10	100	1000
0	0.3162	1.0000	3.1623	10.000	31.6228
10	0.1000	0.3162	1.0000	3.1623	10.000
20	0.0316	0.1000	0.3162	1.0000	3.1623
30	0.0100	0.0316	0.1000	0.3162	1.0000
40	0.0032	0.0100	0.0316	0.1000	0.3162
50	0.0010	0.0032	0.0100	0.0316	0.1000

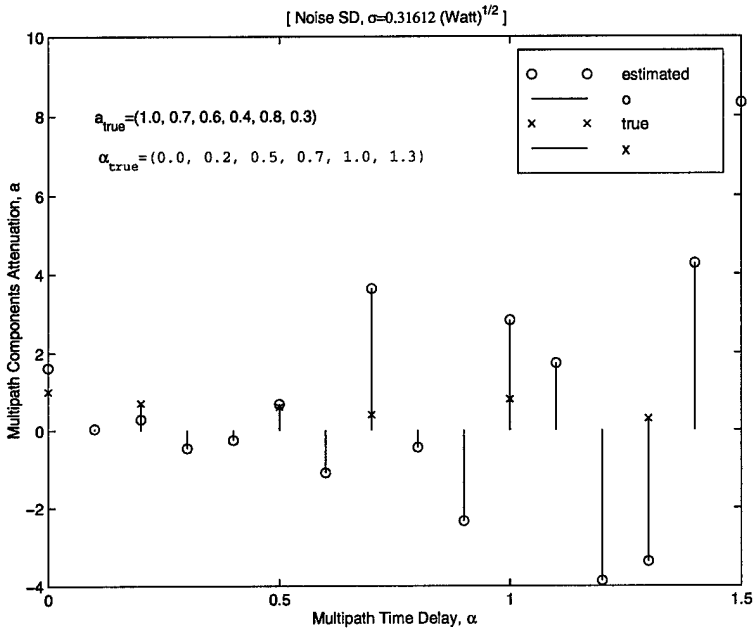


Figure 70 α-Deploying with SD, σ= 0.3162 Watt$^{\frac{1}{2}}$, Example 5.

be detected because the noise dominates the existence of the multipath. This justification can be differentiated between the polygon curves in case of noise free and the presence of noise. In turn, the solution of the linear form in Equation (65) will not fit the desired estimation, in Equation (86).

Figures 70 through 74 show that the multipath parameters can be estimated for $\sigma \leq$ 0.03612 Watt$^{\frac{1}{2}}$. By considering the numbers under the diagonal of values $\sigma \leq 0.03612$ Watt$^{\frac{1}{2}}$ on Table 7, the low-pass filter bandwidth can be determined for each correlator in the bank with the required input SNR. Therefore, α deploying method can be employed as the list in Table 8.

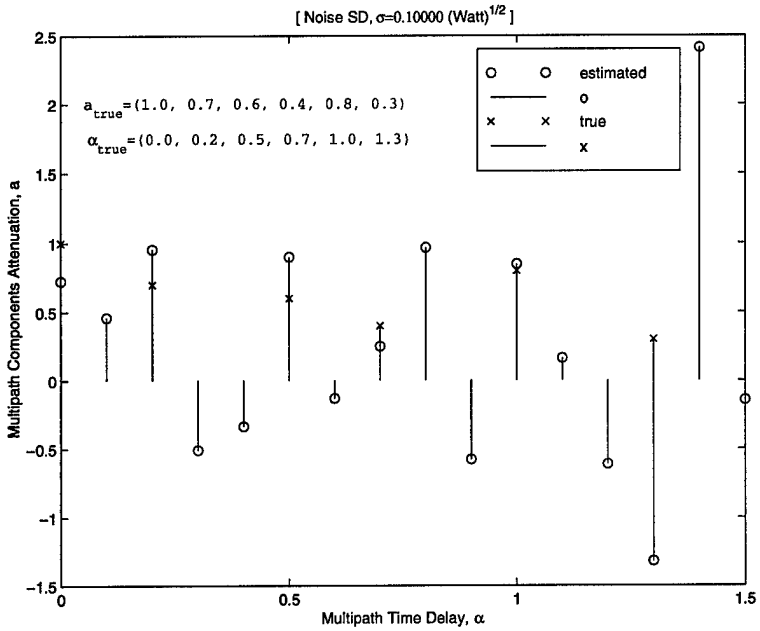


Figure 71 α-Deploying with SD, σ= 0.1000 Watt$^{\frac{1}{2}}$, Example 5.

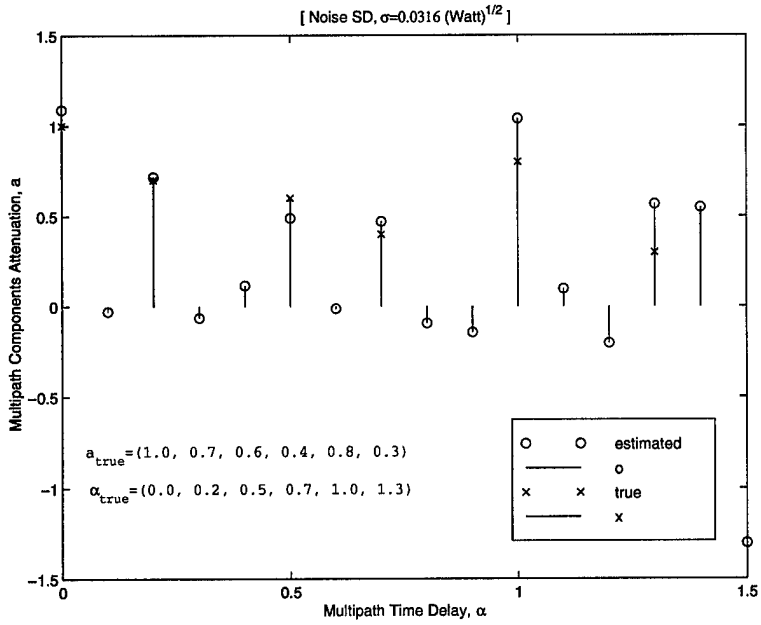


Figure 72 α-Deploying with SD, σ= 0.0316 Watt$^{\frac{1}{2}}$, Example 5.

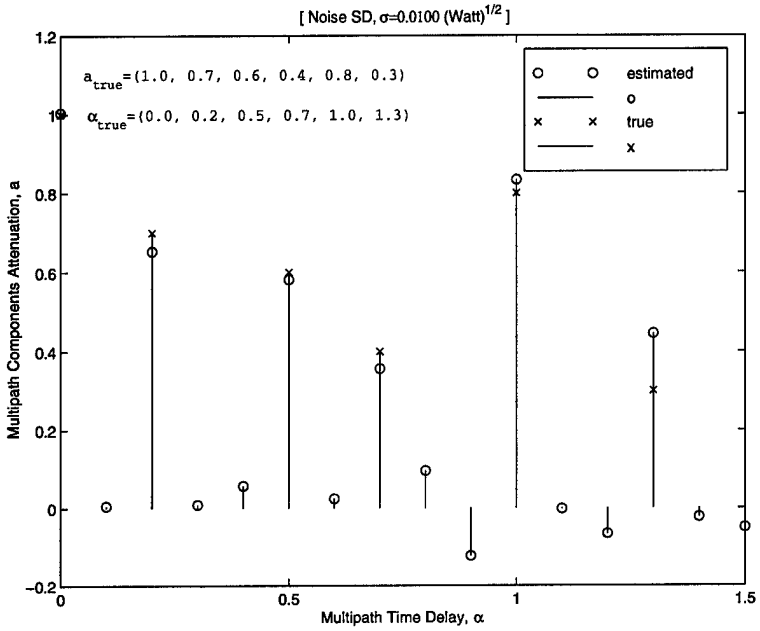


Figure 73 α-Deploying with SD, σ= 0.0100 Watt$^{\frac{1}{2}}$, Example 5.

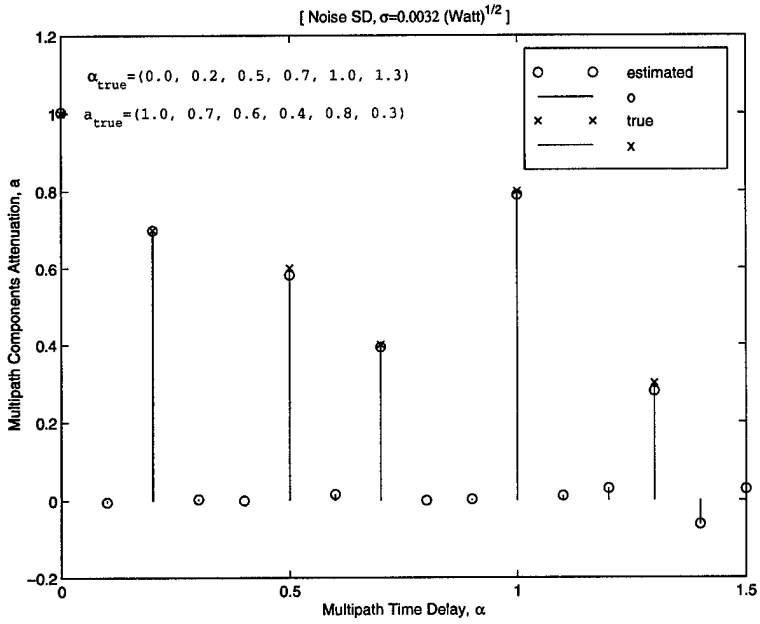


Figure 74 α-Deploying with SD, σ= 0.0032 Watt$^{\frac{1}{2}}$, Example 5.

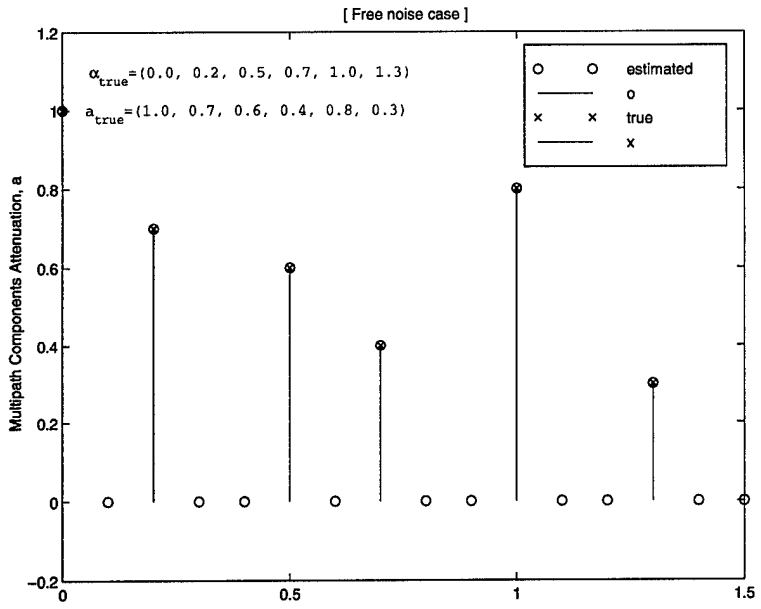


Figure 75 α-Deploying in the Deterministic Case, Example 5.

Table 8 list of B_{lpf} and corresponding range of SNR for the α-Deploying method

B_{lpf}	with input SNR $\geq$
100 Hz	50 db-Hz
10	40
1	30
0.1	20
40	10

In general, the results of Figures 70 to 74 illustrate that the vector $\hat{a}$ can be estimated in noise provided that $\sigma \leq 0.03162$ Watt$^{\frac{1}{2}}$. An attempt can be made to separate the noise from multipath, under the assumption that the multipath-to-noise ratio is high. Simply, the way to accomplish multipath separation from noise is to set all the estimated elements of the vector $\hat{a}$ under some level to zero such that the rest of the vector elements of $\hat{a}$ is the multipath components only. Figure 76 shows an example of $\alpha_{true} = [0.0, \ 0.2, \ 0.5]^T$ with $a_{true} = [1.00, \ 0.82, \ 0.61]^T$ and with $\sigma = 0.0080 \ (Watt)^{\frac{1}{2}}$. In this example the intended level to make the separation is taken at 0.4 the direct path level.

The α-deploying method can be considered as a tool to scope the multipath in the environmental area around the user, especially in the cases of High SNR or High MNR. Therefore, it could be a significant indicator in surveying and determining the region of high

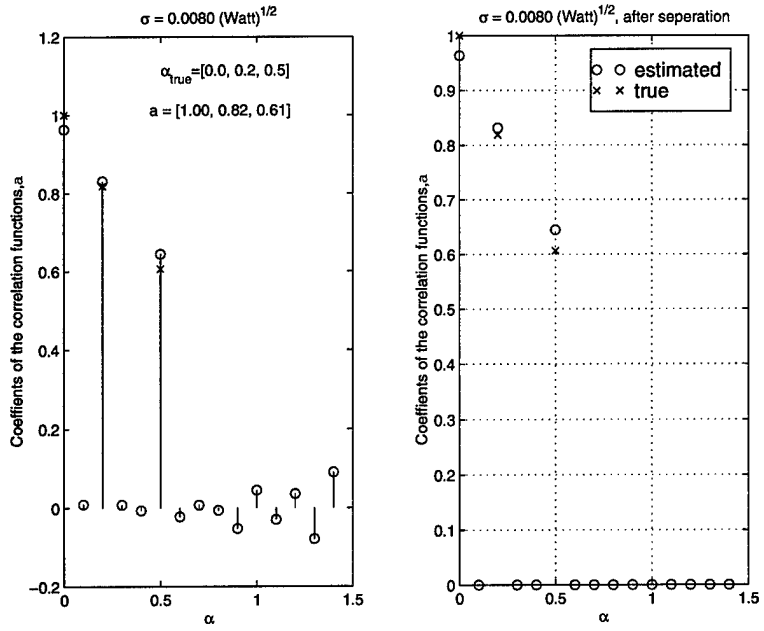


Figure 76 The truncation at the noise level

multipath. The separation of the multipath from the noise could be achievable also by using a Kalman filter. Details of this will be in next section.

4.5 Kalman Filtering Application

The estimated multipath strength parameter, $\hat{x}$, when using α-deploying method, is treated as a random variable. The idea is to use the distribution of this estimated parameter in a Kalman filter to remove the corrupting AWGN throughout the receiving time (see Figure 77).

4.5.1 System Modeling for Kalman Filtering. The α-deploying estimator can be considered as a linear transformation of the measurement vector $\tilde{R}_I$ into the vector $\hat{x}$. The linear transformation can be written as a matrix multiplication

$$\hat{x} = P_s \tilde{R}_I \tag{87}$$

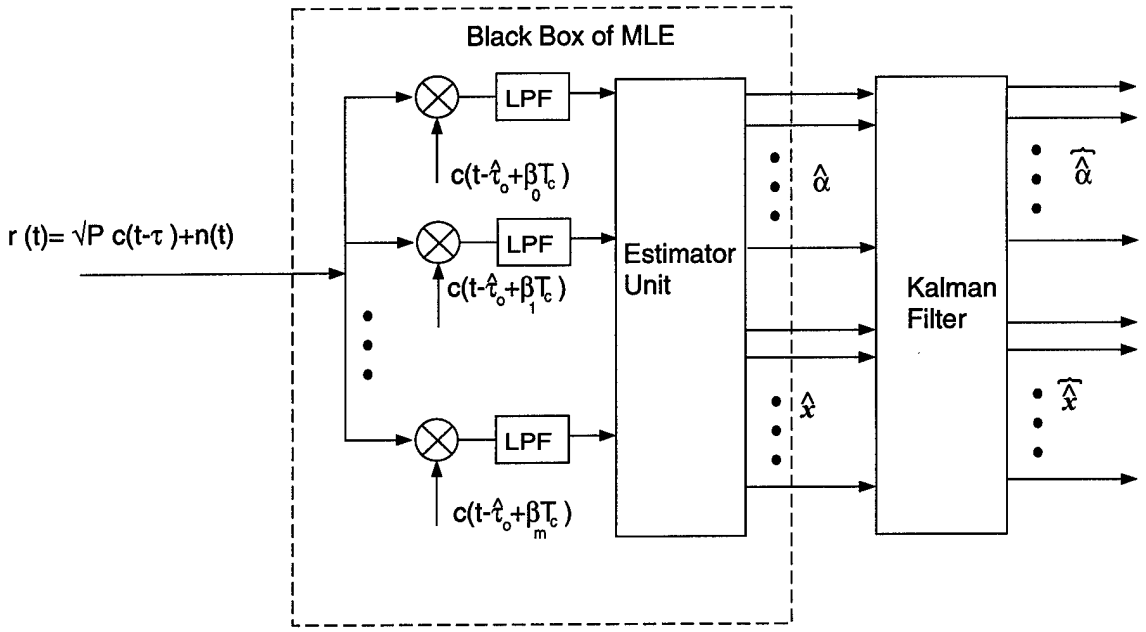


Figure 77 Configuration of Kalman Filter Application.

where P_s is the MLE (i.e. $P_s = (H^T P_\nu^{-1} H)^{-1} H^T P_\nu^{-1}$). Then, the characteristic function of $\hat{x}$ is given by

$$\begin{aligned} \Phi(\omega) &= E\{e^{-j\omega^T \hat{x}}\} \\ &= E\{e^{-j\omega^T P_s \tilde{R}_I}\} \\ &= \exp\{-j\omega^T P_s R_{free} - \tfrac{1}{2}\omega^T P_s C_\nu P_s^T \omega\} \end{aligned} \tag{88}$$

where R_{free} is the mean of $\tilde{R}_I$, considered as the measurement vector in the case of AWGN free . Thus, $\hat{x}$ is distributed as

$$\hat{x} \in \mathcal{N}(P_s R_{free}, P_s C_\nu P_s^T) \tag{89}$$

Let $\hat{x}_{free}$ denotes the MLE of x without AWGN. Let the uncertainty in $\hat{x}$ be denoted by μ, then $\hat{x}$ can be written in terms of $\hat{x}_{free}$ as

$$\hat{x} = \hat{x}_{free} + \mu \tag{90}$$

The uncertainty μ is a colored Gaussian process derived from ν with statistics concluded as follows From Equation 90, the expectation of μ can be expressed in terms of the expectation

of vector ν as

$$\begin{aligned} E\{\mu\} &= E\{\hat{x}\}-E\{\hat{x}_{free}\} \\ &= P_s E\{\tilde{R}_I\}-\hat{x}_{free} \\ &= P_s E\{R_{free}\}+P_s E\{\nu\}-\hat{x}_{free} \\ &= P_s H\hat{x}_{free}+P_s E\{\nu\}-\hat{x}_{free} \\ &= P_s E\{\nu\} \end{aligned} \tag{91}$$

Similarly, the covariance of the noise process μ can be given as

$$\begin{aligned} E\{\mu\mu^T\} &= E\{(P_s\tilde{R}_I-\hat{x}_{free})(P_s\tilde{R}_I-\hat{x}_{free})^T\} \\ &= P_s E\{\tilde{R}_I\tilde{R}_I^T\}P_s^T-\hat{x}_{free}\hat{x}_{free}^T-P_s\tilde{R}_I\hat{x}_{free}^T-\hat{x}_{free}(P_s\tilde{R}_I)^T \\ &= P_s H\hat{x}_{free}\hat{x}_{free}^T H^T P_s^T+P_s E\{\nu\nu^T\}P_s^T-\hat{x}_{free}\hat{x}_{free}^T \\ &= P_s E\{\nu\nu^T\}P_s^T \\ &= P_s C_\nu P_s^T \end{aligned} \tag{92}$$

Equations (91) and (92) can be considered also as a proof of the characteristic function in Equation (88). The conclusion now is that the model can be taken as the estimator. Its input is $\tilde{R}_I$ vector and its output is the vector $\hat{x}$. $\hat{x}$ distributed as $\mathcal{N} \in (\hat{x}_{free}, P_s C_\nu P_s^T)$, subsequently μ is distributed as $\mu \in \mathcal{N}(0, P_s C_\nu P_s^T)$.

4.5.2 Kalman Filter Modeling. The Kalman filter is an algorithm for recursively estimating the dynamical underlying state of the vectors. Therefore the time is very important issue in this application, which includes the multipath parameters and AWGN as a function of time according to the effects of both Doppler shift and the receiver motion.

The modelling of a typical Kalman filter requires [16]

1. Knowledge of the system and measurement device dynamics.
2. The statistical description of the system noise, measurement errors, and uncertainty in the dynamics model, and
3. any available information about initial conditions of the variables of interest.

- In our case the system dynamic can be expressed by the change of the state vector $\hat{x}(t)$ with respect to time t. Then the vector $\hat{x}(t)$ can be evolved according to the difference equation

$$\hat{x}(t_{i+1}) = \Phi(t_{i+1}, t_i)\hat{x}(t_i) + n_s(t_i) \tag{93}$$

where t_i represents the discrete time. $\Phi(t_{i+1}, t_i)$ is the state transition matrix, mostly this matrix is a diagonal matrix, its element represent the rate of change of the fading channel. The fading channel is characterized by fading bandwidth which is determined by the differences in carrier Doppler frequencies between reflections and the line of sight [31]. the fading bandwidth depends on the satellite-receiver-reflector geometry and the velocity of the receiver. Van Nee in [31] gives a rough estimate for the maximum Doppler difference for the C/A-code in most geometrical situation at the equator of 0.62 Hz. In this section we assume that the state transition matrix is an identity matrix, (i.e. no dynamics is assumed) then we concentrate the study of the Kalman filter when it approaches the steady state. Once, the improvement by the Kalman filter is investigated at the steady state, a comparison will be made between the steady state time versus the time of maximum Doppler difference. Let the notation $n_s(t_i)$ be the noise added to the system, then, we assume this noise to be zero because there is no clue shows that $\hat{x}(t)$ is corrupted with additional noise.

- the measurement device is represented by α-deploying estimator unit of the state vector $\hat{x}(t_i)$ as a black box, see Figure 77.

$$Z(t_i) = \hat{x}(t_i) + \mu(t_i) \tag{94}$$

Where t_i is the discrete time, which is the instant of observation in the bank of correlators. $\mu(t_i)$ is the noise process which is expressed by $\mu(t) = P_s\nu(t)$ and has a mean and a covariance defined in Equation (91) and (92). As previously mentioned The noise process $\nu_j(t)$ for each correlator j has PSD, $\frac{N_o}{2}$ flatten in the range of frequency $[-B_{lpf}, B_{lpf}]$, Thus, the noise $\nu(t)$ is a first-Markov process, and simply the PSD can

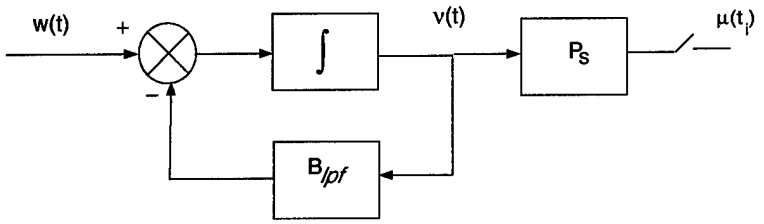


Figure 78 Shaping Filter Model

be written as [16]

$$\Psi_\nu(\omega) = \frac{2\sigma^2 B_{lpf}}{\omega^2 + B_{lpf}^2} \tag{95}$$

- A first-Markov process is modeled in the Kalman filter as a shaping filter driven by a white Gaussian noise. The shaping filter model to generate the process $\nu(t)$ can be described as [16]

$$\dot{\nu}(t) = -B_{lpf}\nu(t) + w_f(t) \tag{96}$$

where $w_f(t)$ is the zero mean white Gaussian noise. The shaping filter expresses the correlation of the noise process in time while there exit a cross-correlation established from the correlators (see Equation 82), so as previously mentioned the noise output of the shaping filter is distributed as $\nu \in \mathcal{N}(0, C_v)$. Consequently μ is distributed as $\mu \in \mathcal{N}(0, P_s C_\nu P_s^T)$. It is preferable to establish the cross correlation in the shaping filter model for the driving white noise process $w_f(t)$, thus $w_f(t)$ has a mean and covariance likewise

$$\begin{aligned} E\{w_f(t)\} &= 0 \\ E\{w_f(t) w_f(t')^T\} &= C_\nu \delta(t - t') \end{aligned} \tag{97}$$

Figure 78 shows the shaping filter model.

The descritized form of Equation (96) is

$$\nu(t_{i+1}) = \Phi_f(t_{i+1}, t_i)\nu(t_i) + w_{d_f}(t_i) \tag{98}$$

where the state transition matrix $\Phi_f(\Delta t) = e^{-\Delta t B_{lpf}}$, $\Delta t = (t_{i+1} - t_i)$ and $w_{d_f}(t_i)$ has

$$\begin{aligned} E\{w_{d_f}(t_i)\} &= 0 \\ E\{w_{d_f}(t_i)w_{d_f}(t_i)^T\} &= C_\nu\left(1-e^{-2\Delta t B_{lpf}}\right) \end{aligned} \tag{99}$$

To express Equation (98) in terms of $\mu(t_i)$, multiplying by P_s matrix in both sides

$$P_s\nu(t_{i+1}) = e^{-\Delta t B_{lpf}} P_s\nu(t_i)+P_s w_{d_f}(t_i) \tag{100}$$

then,

$$\mu(t_{i+1}) = e^{-\Delta t B_{lpf}}\mu(t_i)+w_{d_\mu}(t_i) \tag{101}$$

so the covariance of $w_{d_\mu}(t_i)$

$$E\{w_{d_\mu}(t_i)w_{d_\mu}(t_i)^T\} = P_s C_\nu P_s^T\left(1-e^{-2\Delta t B_{lpf}}\right) = Q_d \tag{102}$$

- Now define the augmented state vector process $x_a(t_i)$ as

$$x_a(t_i) = \begin{bmatrix} \hat{x}(t_i) \\ \mu(t_i) \end{bmatrix} \tag{103}$$

Equations (93) and (101) are written as an augmented state of difference equation

$$\begin{bmatrix} \hat{x}(t_{i+1}) \\ \mu(t_{i+1}) \end{bmatrix} = \begin{bmatrix} I_{n+1} & 0 \\ 0 & I_{n+1}e^{-\Delta t B_{lpf}} \end{bmatrix} + \begin{bmatrix} 0 \\ w_{d_\mu}(t_i) \end{bmatrix} \tag{104}$$

Equation (104) is

$$x_a(t_{i+1}) = \Phi_a(\Delta t)x_a(t_i)+w_{d_a}(t_i) \tag{105}$$

and the associated measurement equation

$$Z(t_i) = [I_n|I_n]\begin{bmatrix} \hat{x}(t_i) \\ \mu(t_i) \end{bmatrix} \tag{106}$$

Also Equation (106) is

$$Z(t_i) = H_a(t_i)x_a(t_i) \tag{107}$$

- The augmented form of the time-correlated measurement in Equation (107) gives a problem of a perfect measurements [16] which is treated by using a difference of consecutive measurements to generate a pseudo-measurements in which the corrupting noise is white. Then the augmentation is abandoned and the descritze-time system and shaping filter representations are

$$\hat{x}(t_{i+1}) = \hat{x}(t_i) \tag{108}$$

$$\mu(t_{i+1}) = e^{-\Delta t B_{lpf}}\mu(t_i) + w_{d_\mu}(t_i) \tag{109}$$

The pseudo-measurement process is represented as

$$Z_d(t_i) = Z(t_{i+1}) - \Phi_f(t_{i+1}, t_i)Z(t_i) \tag{110}$$

Equation (110) yields the original system states, corrupted by a zero-mean white Gaussian noise of strength $R_d(t_i)$,

$$R_d(t_i) = Q_d \tag{111}$$

- Now the Kalman filter algorithm is summarized as
using the notations $\widehat{(.)}$ for the Kalman filter mean and $\hat{(.)}$ for the α-deploying estimator.

the propagation state

$$\begin{aligned} \widehat{\hat{x}}(t_i^-) &= \widehat{\hat{x}}(t_{i-1}^-) \\ P(t_i^-) &= P(t_{i-1}^+) \end{aligned} \tag{112}$$

The Kalman filter update equations

$$\begin{aligned} K(t_i) &= P(t_i^-)\left[P(t_i^-)+R_d(t_i)\right]^{-1} \\ \widehat{\hat{x}}(t_i^+) &= \widehat{\hat{x}}(t_i^-)+K(t_i)\left[Z_d(t_i)-\widehat{\hat{x}}(t_i^-)\right] \\ P(t_i^+) &= P(t_i^-)-K(t_i)P(t_i^-) \end{aligned} \tag{113}$$

The initial conditions

we take the initial condition as the α-deploying estimate of the state vector $\hat{x}$ at the initial time t_o, i.e. the first measurement of the state $\hat{x}$

$$\begin{aligned} \widehat{\hat{x}}(t_o) &= \hat{x}(t_o) \\ P(t_o) &= P_s C_\nu P_s^T \end{aligned} \tag{114}$$

4.5.3 Simulation Results. In the computer usually the noise model of random vector say, w, vector can be generated with a normal distribution $w \in \mathcal{N}(0, \sigma^2 I)$, where I is the identity matrix. The actual distribution of the vector ν vector is $\mathcal{N}(0, \sigma^2 P_\nu)$, so it requires a linear transformation from a white noise vector, w to a color noise vector, ν. The transformation well documented in [24] and examined by [14]. The technique of the transformation needs to synthesis matrix equation as

$$BIB^T = P_\nu \tag{115}$$

where the matrix B is the Cholesky factorization of matrix P_ν. The algorithm of the transformation is:

1. Cholesky factorization of matrix $BB^T = P_\nu$
2. generation of the random vector by the computer as a normal distribution $w : N(0, I)$
3. achieve the linear transformation $\nu = B^T w$
4. weighting the noise level by the standared deviation σ, so $\nu = \sigma B^T w$

Nonetheless, the simulation requires also a generation of time-correlated noise which can be achieved as in Equation (101). Since the time taken to estimate the state vector $\hat{x}$ by

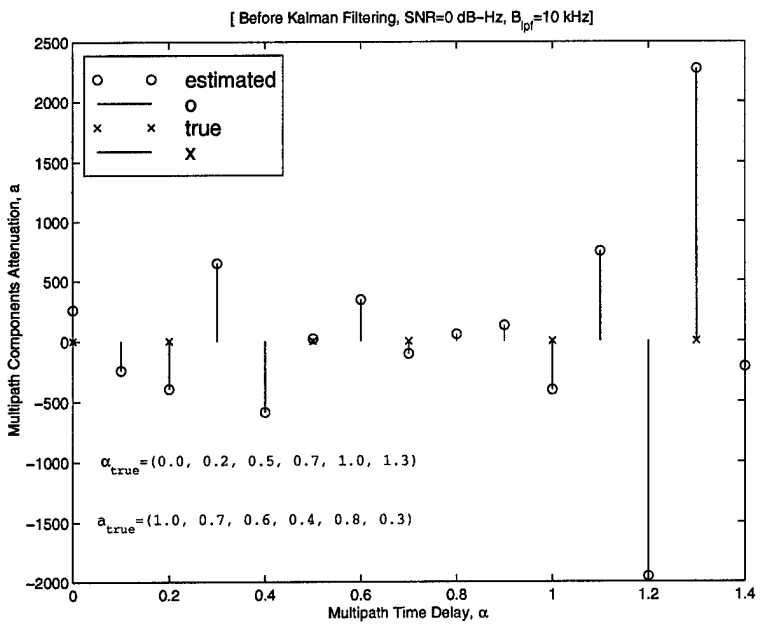


Figure 79 Before Applying the Kalman Filter

α-deploying estimator unit is related to the correlation process time by the correlators in the bank, so the sample period $\Delta t = (t_{i+1} - t_i)$ is taken $1/B_{lpf}$. The application of the Kalman filter improves the performance of the α-deploying estimate of the multipath parameters in the face of AWGN. There are three trad-off parameters in this application: the bandwidth of the integrator which control the speed of the tracking loop of the receiver, the SNR which depends on the noise sources into the receiver, and the unknown dynamics of the environment which limits the run time of Kalman filtering algorithm. The maximum rate taken such that multipath change is mentioned before 0.62 Hz for the stationary receiver, so in this simulation, the operation of Kalman filter is applied through one interval of 1 second. The chosen bandwidths 10 kHz and 1 kHz are large enough to meet the fast response requirements of the tracking loop for C/A-code. Figure 81 through 92 show the results of the multipath estimates before and after applying Kalman filtering for SNR = 0, 1, 10 ,20, 30, 40 and 50 respectively. The Kalman filter gives an improvement for SNR 40 and 50. Consequently the improvement is also apparent for bandwidth of 1 kHz-see e.g., Figure 93 through 96.

Consequently, an effort to conduct with performance analysis of the Kalman filter application is the results of a simulation plots are based on Monte Carlo simulation taken for 50 run per each plot contains the ensemble mean of the simulation error. The format

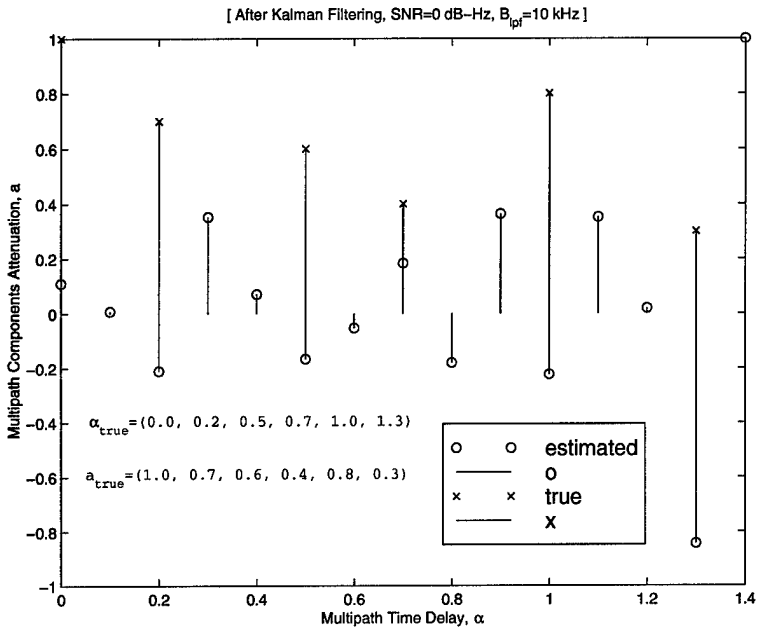


Figure 80 Kalman Filter applied in interval of 1 sec.

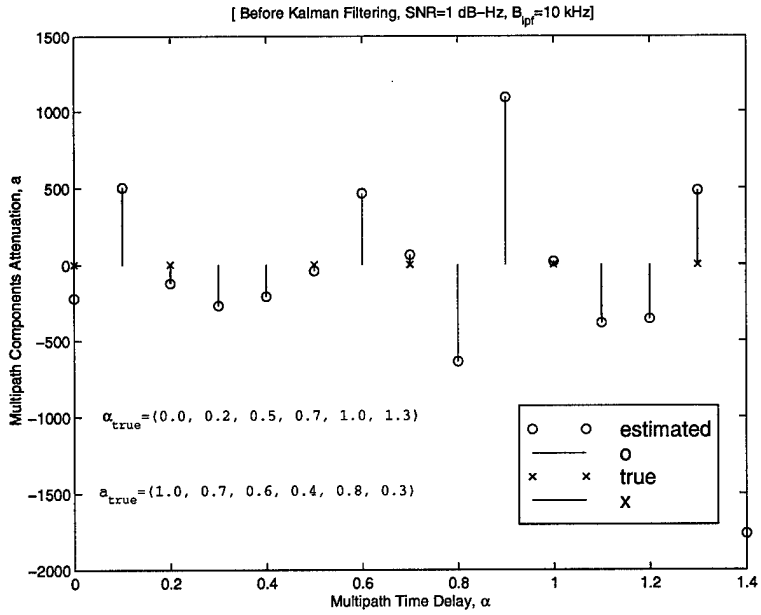


Figure 81 Before Applying Kalman Filter

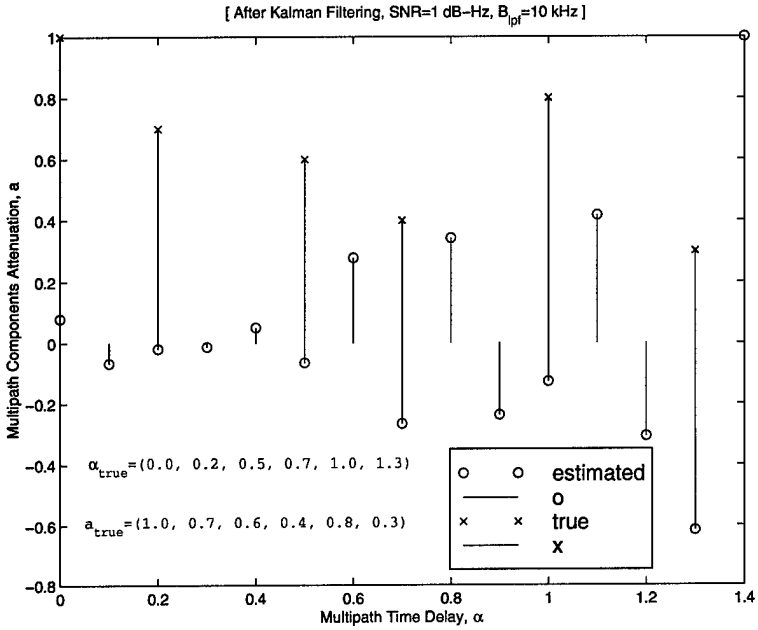


Figure 82 Kalman Filter applied in interval of 1 sec.

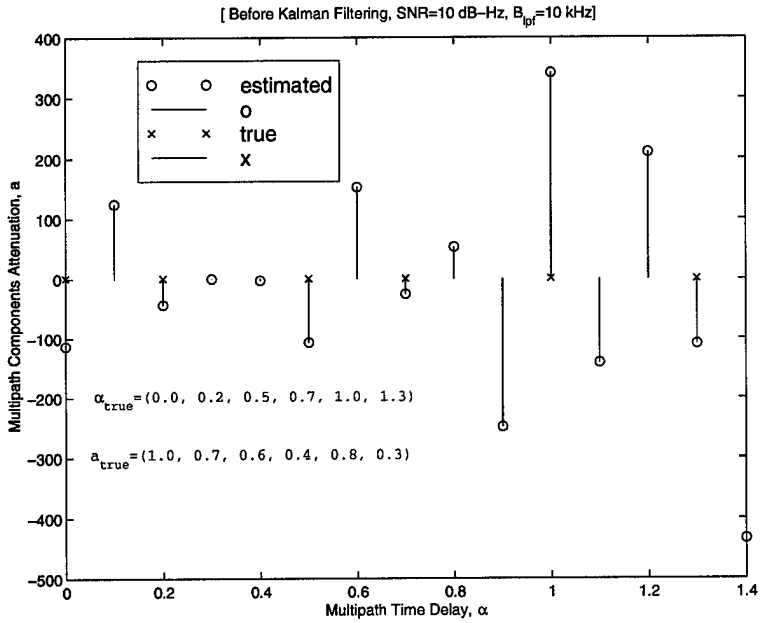


Figure 83 Before Applying Kalman Filter

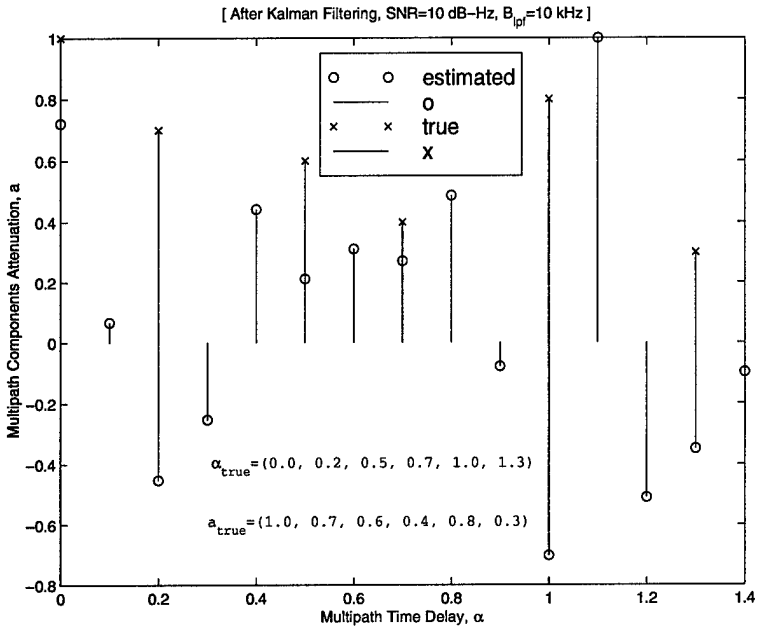


Figure 84 Kalman Filter applied in interval of 1 sec.

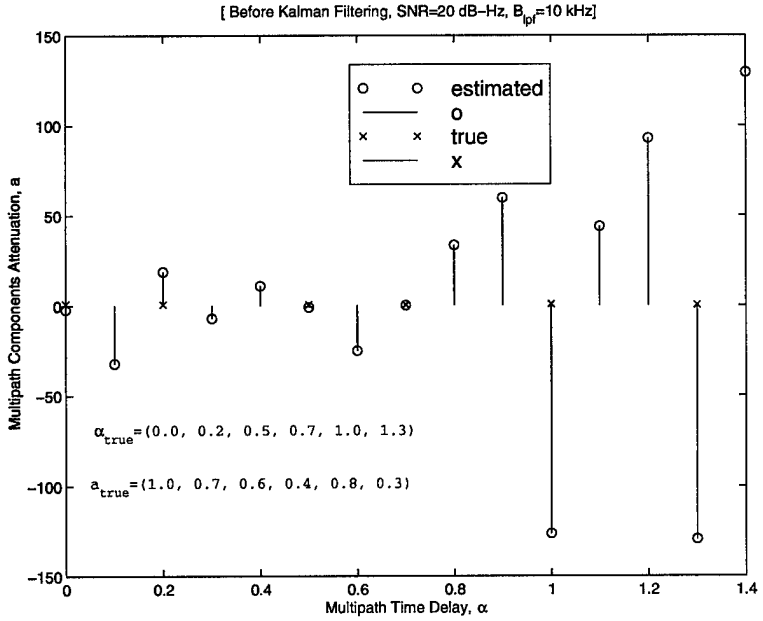


Figure 85 Before Applying Kalman Filter

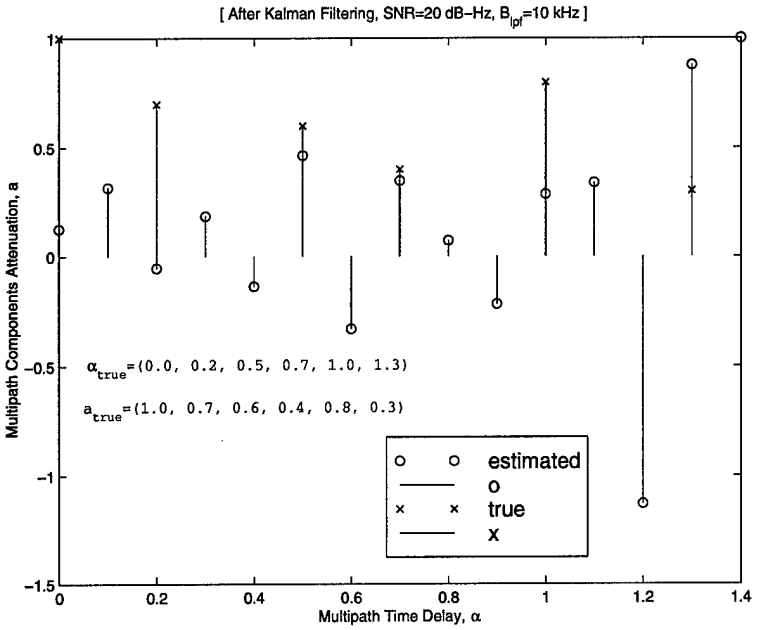


Figure 86 Kalman Filter applied in interval of 1 sec.

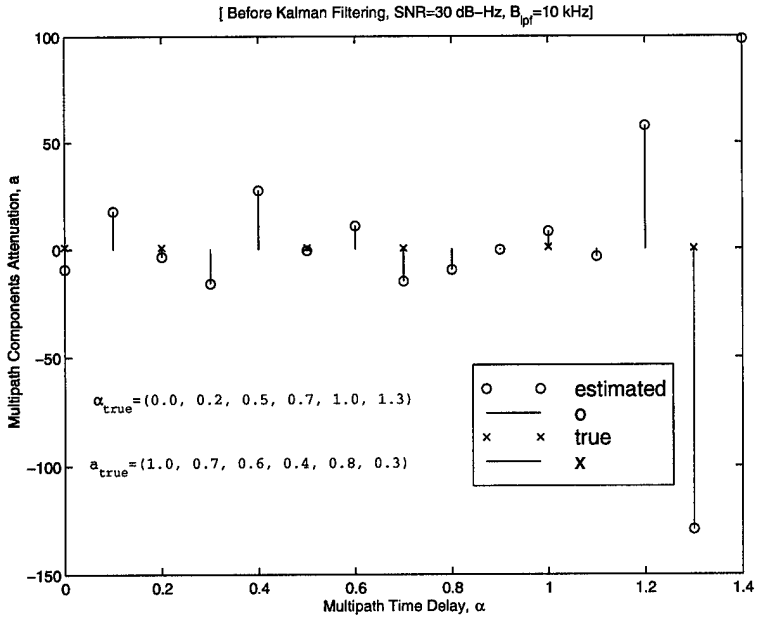


Figure 87 Before Applying Kalman Filter

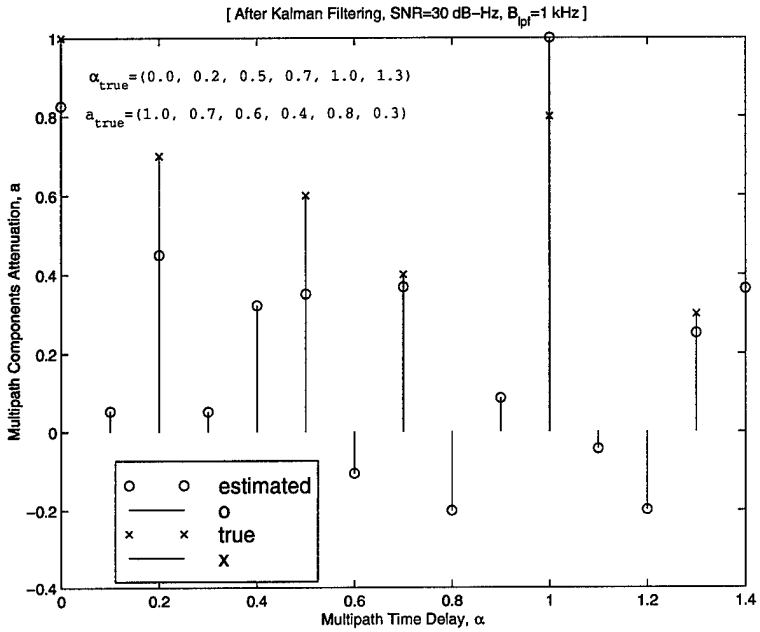


Figure 88 Kalman Filter applied in interval of 1 sec.

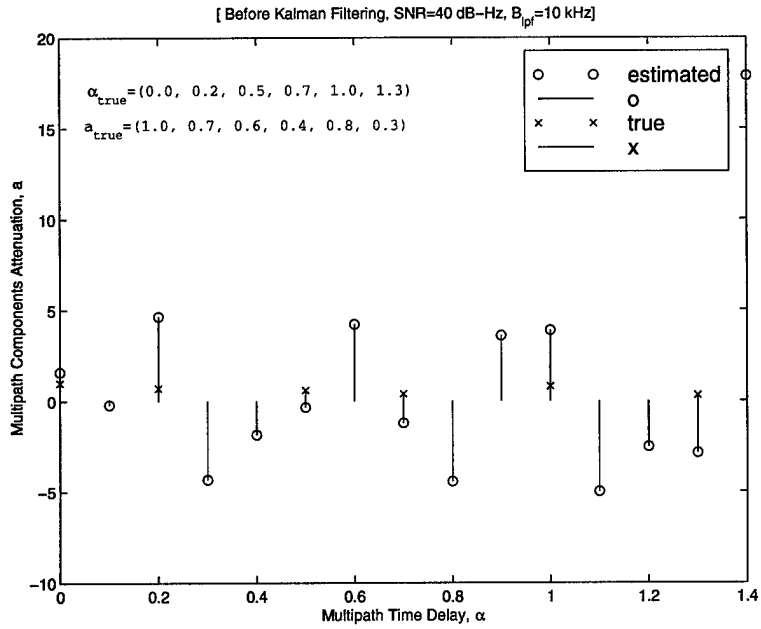


Figure 89 Before Applying Kalman Filter

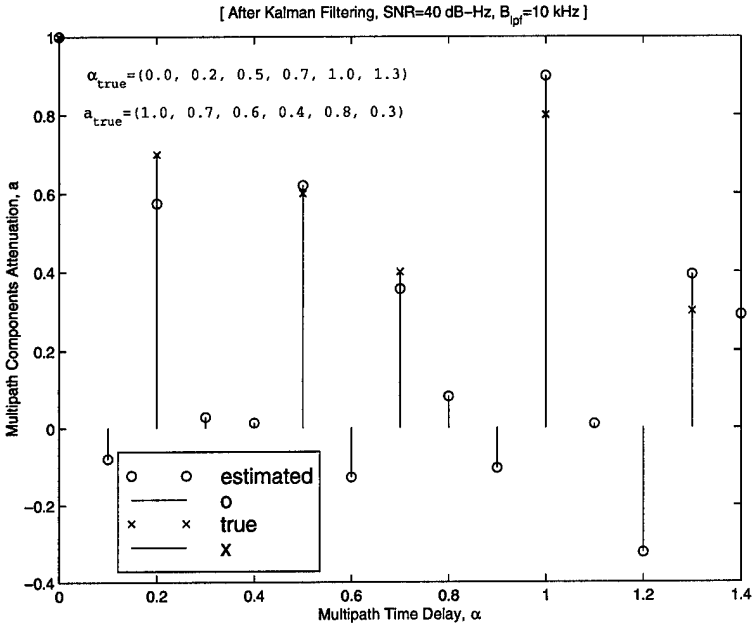


Figure 90 Kalman Filter applied in interval of 1 sec.

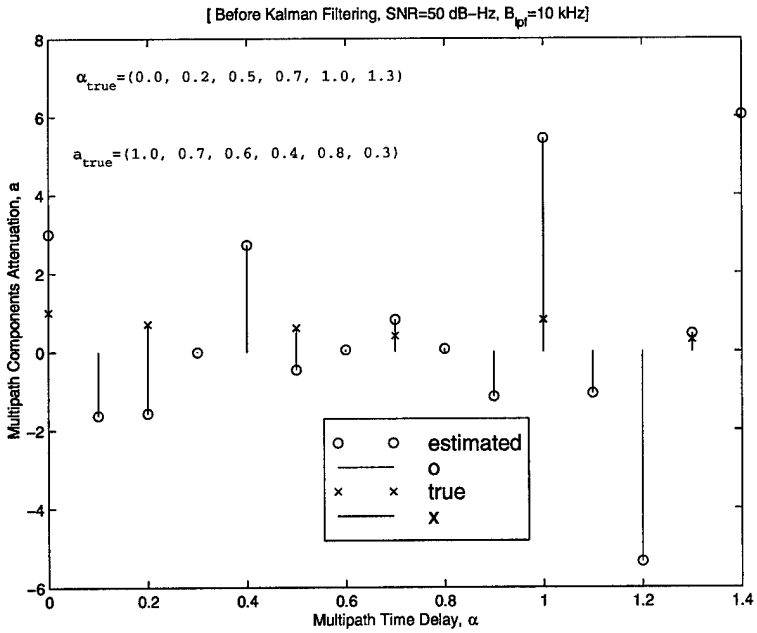


Figure 91 Before Applying Kalman Filter

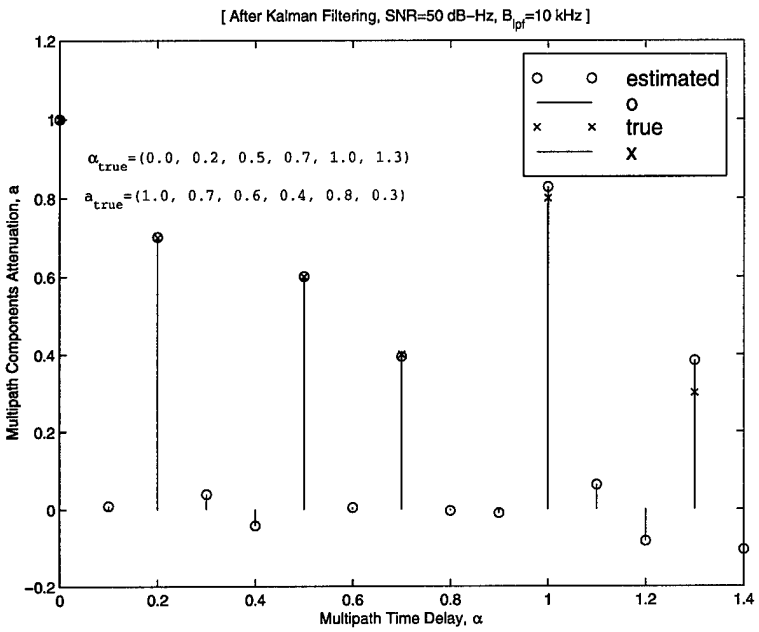


Figure 92 Kalman Filter applied in interval of 1 sec.

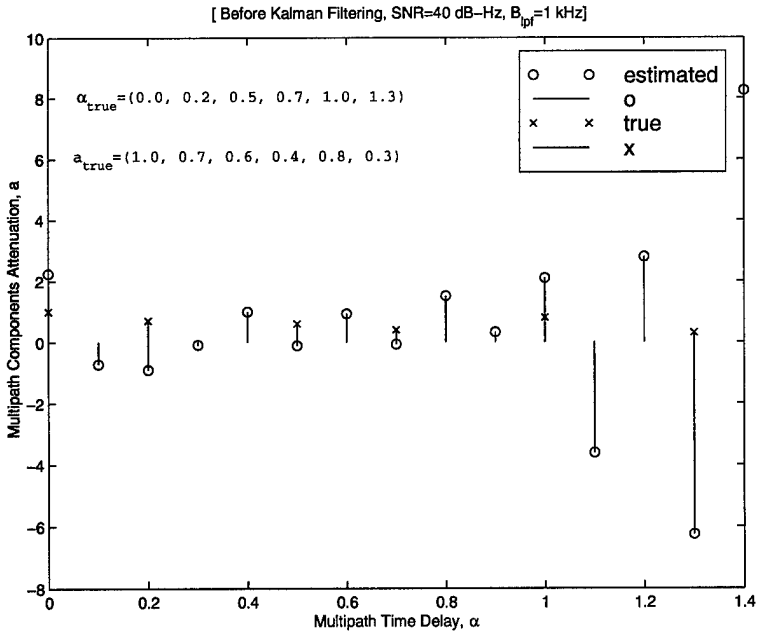


Figure 93 Before Applying Kalman Filter

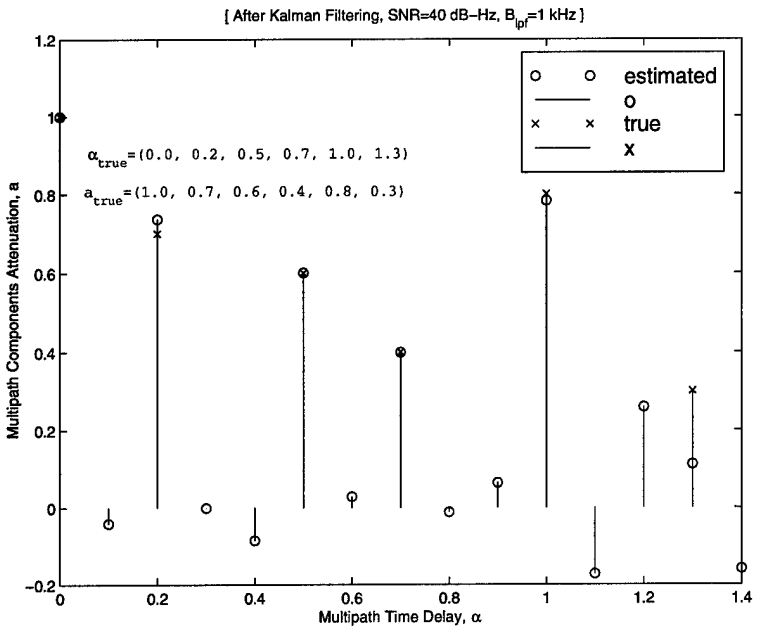


Figure 94 Kalman Filter applied in interval of 1 sec.

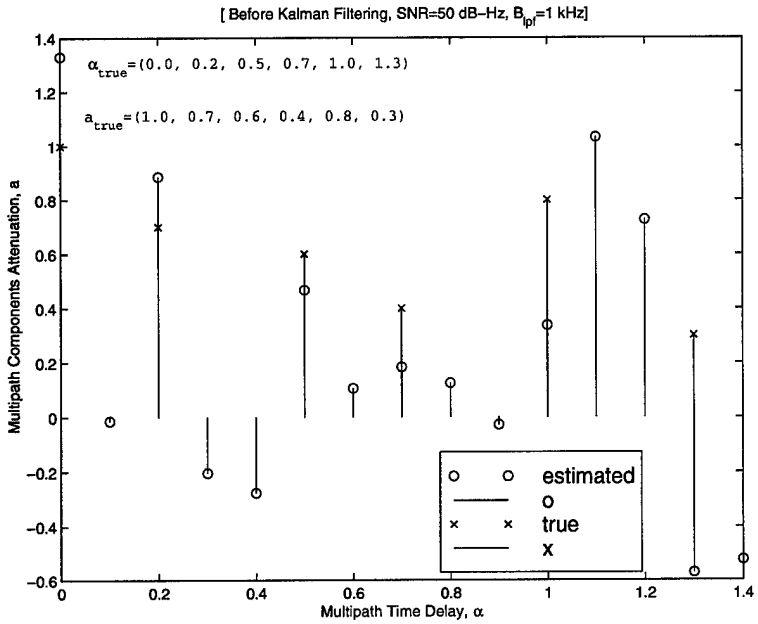


Figure 95 Before Applying Kalman Filter

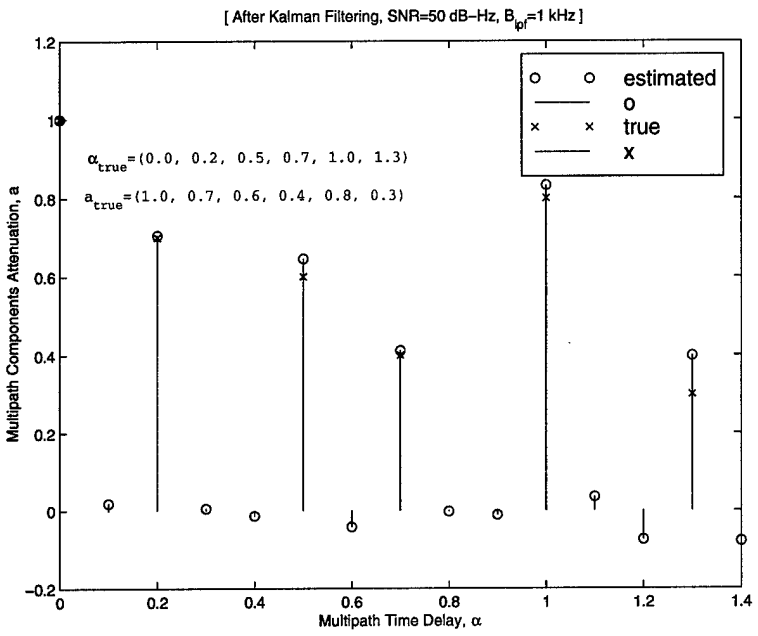


Figure 96 Kalman Filter applied in interval of 1 sec.

has been applied into the plots is made for different parameter such as the bandwidth of the low pass filter, the Kalman filter sample period, and the SNR. The bandwidth is chosen 10 kHz and 1kHz that is wide enough to be useful for quicker tracking loops. The SNR is limited to do not exceed the standared accuracy of the GPS receivers. The complete plots format are shown in Appendix D. The results of Monte Carlo simulation reveal that the significance of the Kalman filter as a second linear estimator cascaded into the α-deploying estimator, whereas the run of the KF is achieved through one second to get a chance to avoid the change in the receiver environments according to the satellite motion. In general The simulation illustrates the following notes

- The results have been improved after the first interval from 200 to 300 m sec of starting run of the KF
- the attenuation coefficient errors are always small corresponding to the multipath delay from 0 to T_c and it become bigger from $1.1T_c$ to $1.5T_c$. Fortunately, the DLL in the GPS can remove the multipath signal for long delay by using a correlation process with spacing $0.1T_c$. In practice to use this approach for removing the multipath error and the multipath dynamics as will, we have to reset the Kalman filter each second.

- we can conclude from the simulation that a useful chance to apply KF for low SNR with wide low pass filter bandwidth, comparing the values for SNR=50 with B_{lpf} = 10kHz with values lower than this.

Appendix D show also the codes designed for the Kalman Filter application using m file code in the MATLAB.

4.6 Summary

In this chapter a new method called α-deploying have been developed for solving the multipath problem. The proposed algorithm decomposes the multipath received signal into the direct path signal and the multipath components, the number of the multipath components is detected and the corresponding multipath parameters for each multipath component is estimated. Furthermore the multipath signal's parameters can be estimated at any instant of observation, which implies that the Doppler shift is implicitly incorporated in the multipath estimates. Because of the instantaneous high probability of observing the multipath parameters, the α-deploying method can be used to track the dynamic multipath signal caused by environmental changes (satellites and receiver motion). Also the method can be suggested as a tool to scope the multipath environment in the area around the user, and may be a significant indicator in surveying and determining the region of high multipath.

The search method can work when a restricted number of multipath signal is presented in the environmental area around the receiver. The α-deploying method is applicable for noise bandwidth $BW_{noise} \leq$ 100, 10, 1 and 0.1 Hz, corresponding to input $SNRs \geq$ 50, 40, 30, and 20 dB Hz, respectively, which is the accuracy of the standard C/A code tracking loop. The application of the Kalman filter as a cascaded estimator into the α-deploying estimator unit adds the ability to achieve an accurate estimate of multipath in a high noise level. Hence the Kalman filter can be considered as an important tool for separating the direct signal from multipaths in noise.

V. *Phase-Locked Loop Analysis and Simulation*

5.1 Introduction

Phase-Locked Loop plays an important roles in establishing coherent references in the receiver tracking loops. Nevertheless, it tracks the received carrier signal in the GPS receiver emphasis a high accuracy in the ranging process and ambiguity resolution. In this chapter a PLL simulation model is introduced. The purpose of this simulation is to achieve a significance investigation of the standard PLL as a reference loop for the modified PLL that will be introduced in the next chapter. and The objective of the modified PLL is to mitigate the multipath upon carrier phase. An analysis to PLL is accomplished, the carrier tracking performance is investigated and the effect of multipath has been studied in the simulated PLL.

5.2 Basic Phase-Locked-Loop

The PLL is a negative feedback subsystem for tracking the phase of the received signal. basically the PLL consists of three components: (1) a phase detector, (2) a low -pass filter, and (3) a voltage-controlled oscillator (VCO). Figure 97 illustrates the PLL block diagram.

The VCO is an oscillator that produces a periodic wave form, the frequency of which is varied about free-running frequency, f_c, according to the value of its input applied voltage. f_c is the frequency when the input applied voltage is zero. The phase detector produces an output signal function, $\sin(\epsilon)$; where $\epsilon = (\phi_o - \phi_{pll})$ is the phase difference between the incoming signal that is used to change the frequency of the VCO output. The output of the detector is low pass filtered provides the control signal that is used to change the frequency of the VCO. The PLL configuration may be designed so that it acts as a narrow-band tracking filter when the low-pass filter is a narrow-band filter. The PLL tracks the input signal in either of these ways [12]:

- If the applied signal has initial frequency f_c, the VCO will track the input frequency over some range, namely lock range, provided that the input signal frequency changes slowly.

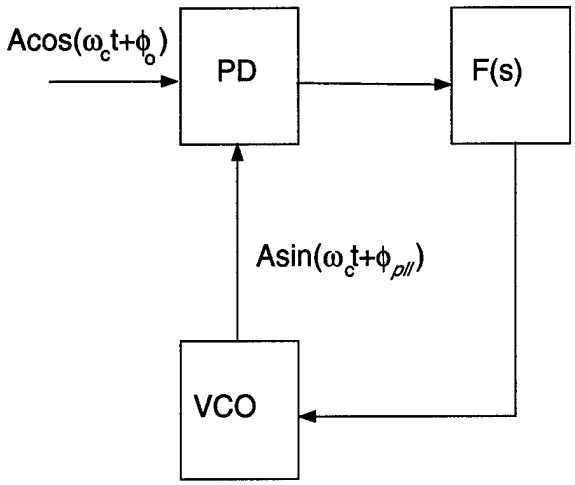


Figure 97 Phase-lock loop: basic block diagram

- If the applied signal has an initial frequency not equal to f_c, the loop may not acquire lock, even though the input frequency is within the lock range.
- The *maximum locked sweep rate* which is defined as the maximum rate of the input frequency for which the loop will remain locked. If the input frequency changes faster than this rate, the loop will drop out of lock.

5.3 Analysis and Simulation Modeling of PLL

This section includes analysis of the main components in PLL and how the construction of each part is achieved in SIMULINK.

5.3.1 Voltage-Controlled Oscillator (VCO). It is initially assumed that the loop is operating in the frequency-synchronized mode [20]; that is, only the phase of the VCO must be synchronized with the input signal phase. The input signal is represented as

$$R_{IP}(t) = A\cos(\omega_c t + \phi_o) \tag{116}$$

and the VCO output is

$$R_{VCO}(t) = A_o \sin(\omega_c t + \phi_{pll}) \tag{117}$$

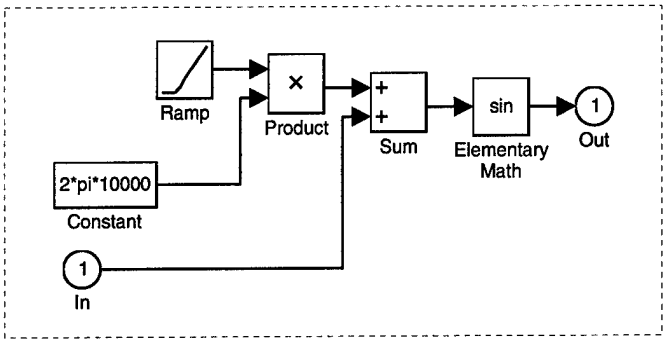


Figure 98 Voltage-Controlled Oscillator (VCO) Model

The VCO is a local signal generator able to change the output phase signal according to the input voltage, V_o so as to lock the reference input phase signal. The oscillator phase change due to this voltage is given by

$$\frac{d\phi_{pll}}{dt} = K_o V_o \tag{118}$$

where K_o is the VCO constant with units rad/sec per volt [8, 23]. In the notation of the Laplace transformation

$$\phi_{pll} = \frac{K_o}{s} V_o \tag{119}$$

In the SIMULINK, Equation (116) and (117) are modeled as elementary math blocks a sine and a cosine functions, its arguments are the product of ramp function by constant $(2\pi f_c)$. The phase ϕ_o can be added to the argument as well. This model is illustrated in Figure 98. The input phase to the model can be controlled using slider gain block weighted by constant (π). The model shows that the VCO gain K_o is equivalent to the unity. The choice of the frequency (f_c) can be controlled by changing the constant block of $(2\pi f_c)$. In this simulation we have taken $(f_c = 1kHz)$.

5.3.2 Phase Detector (PD). The PD circuits used determines the types of the PLL which are analog or digital types. The properties of the PD circuits have a strong influence on the dynamic performance of the PLL system. Four types of PDs are the most frequently used [2], the linear analog computation (four-quadrant multiplier, type1), the

digital types (The exclusive-OR gate,type 2, the edge-triggered JK flip flop, type 3, and the phase/frequency detector, type 4).

A phase detector is a device whose output is a function of the instantaneous phase difference between two input signals. The defining equation is

$$V_o = K_d \sin(\phi_o - \phi_{pll}) \tag{120}$$

where V_o is the phase detector output signal, ϕ_o and ϕ_{pll} are the instantaneous phase angles of the two inputs, respectively, and K_d is the phase detector sensitivity in volts per radian. Phase detectors used in angular feedback loops either of the balanced or the doubly-balanced type. the balanced types suppresses one of the input signals, while the doubly-balanced type is used when suppression with respect to both input signals is desired. For the mode of operation, with one of two input signals having a much larger amplitude than the other, both the balanced and the doubly-balanced circuits have the same, essentially sinusoidal, input-output relationship [13]. The characteristics of the four PDs types are sinusoidal, triangular , sawtooth and sawtooth with full frequency range, respectively.

The type 1 phase detector can be assumed as an ideal multiplier followed by a lowpass filter whose sole effect is to remove the double-frequency component at the multiplier output. The classical digital PLL (DPLL) is not always a pure digital but it is characterized by the appearance of intermediate analog signal. The model we are about to design in this chapter is chosen as the type 1 PLL model.

In the SIMULINK the multiplier of the phase detector is represented as product block and the lowpass filter is represented as the first order linear transfer function which is given in Laplace domain as

$$H_{PD}(s) = \frac{-2\pi^2 f_{lpf}}{s + 2\pi f_{lpf}} \tag{121}$$

where f_{lpf} is the desired cutoff frequency of the LPF of PD. Figure 99 shows the model of the PD designed in the SIMULINK.

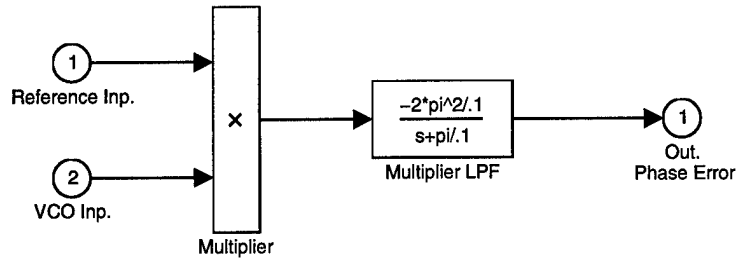


Figure 99 Model of Phase Detector (PD)

The product of the two PD input signals are the product of the sinusoidal signals defined in Equations (116) and (117) which is given as

$$R_{IP} \times R_{VCO} = \frac{1}{2} A A_o \left[\sin(2\omega_c t + \phi_o + \phi_{pll}) - \sin(\phi_o - \phi_{pll})\right] \tag{122}$$

Obviously, the function of the LPF following the multiplier is to remove the first term of second harmonic frequency in Equation (122) and the desired output will be the second term or Equation (120). The numerical values of the LPF gain in Equation (121) is adjusted such that the PD input output (I/O) is 1:1 phase difference into volts, so the PD gain becomes $K_d = 2\pi^2 f_{lpf}$. The PD characteristic can be obtained from the simulation, by SIMULINK implementation as in Figure 100. Some scopes of the PD waveforms are illustrated in Figure 101 for input phase differences of π, 0.5π, 0.25π, 0, -0.25π and -0.5π, respectively.

The resulted PD characteristic simulation is equivalent to the analytical PD characteristic (the sinusoidal form of Equation (120), see Figure (102).

The simulation of the transient response can also be implemented as in Figure 103. Figure 104 shows the PD transient response in case of cut-off frequency $f_{lpf} = 10Hz$.

In the PD design, it is noticed that the trade-off parameters effect the proper PD characteristic versus the PD step response, are the LPF gain and bandwidth. The LPF gain can be adjusted by changing its value according to the cut-off frequency, f_{lpf}. Then the adjusted LPF gain is taken as $K_d = 2\pi^2 f_{lpf}$. As long as the LPF bandwidth is narrower the accuracy of PD characteristic is improved, It can be shown in Figure 105 that as long

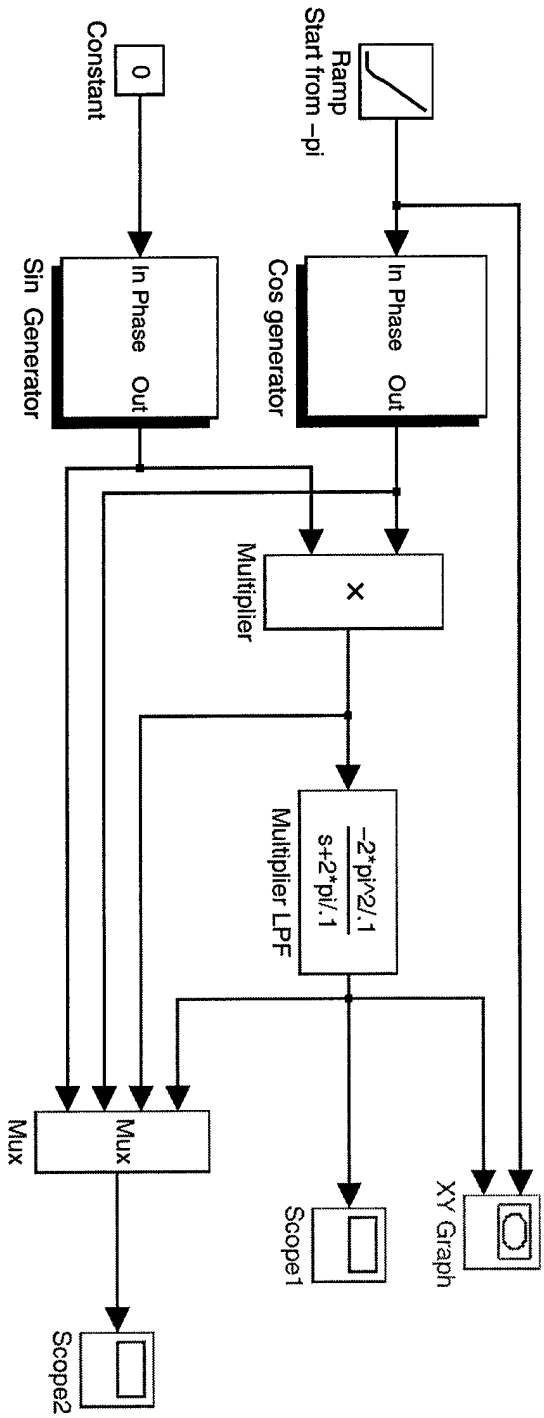


Figure 100 The PD characteristic implementation

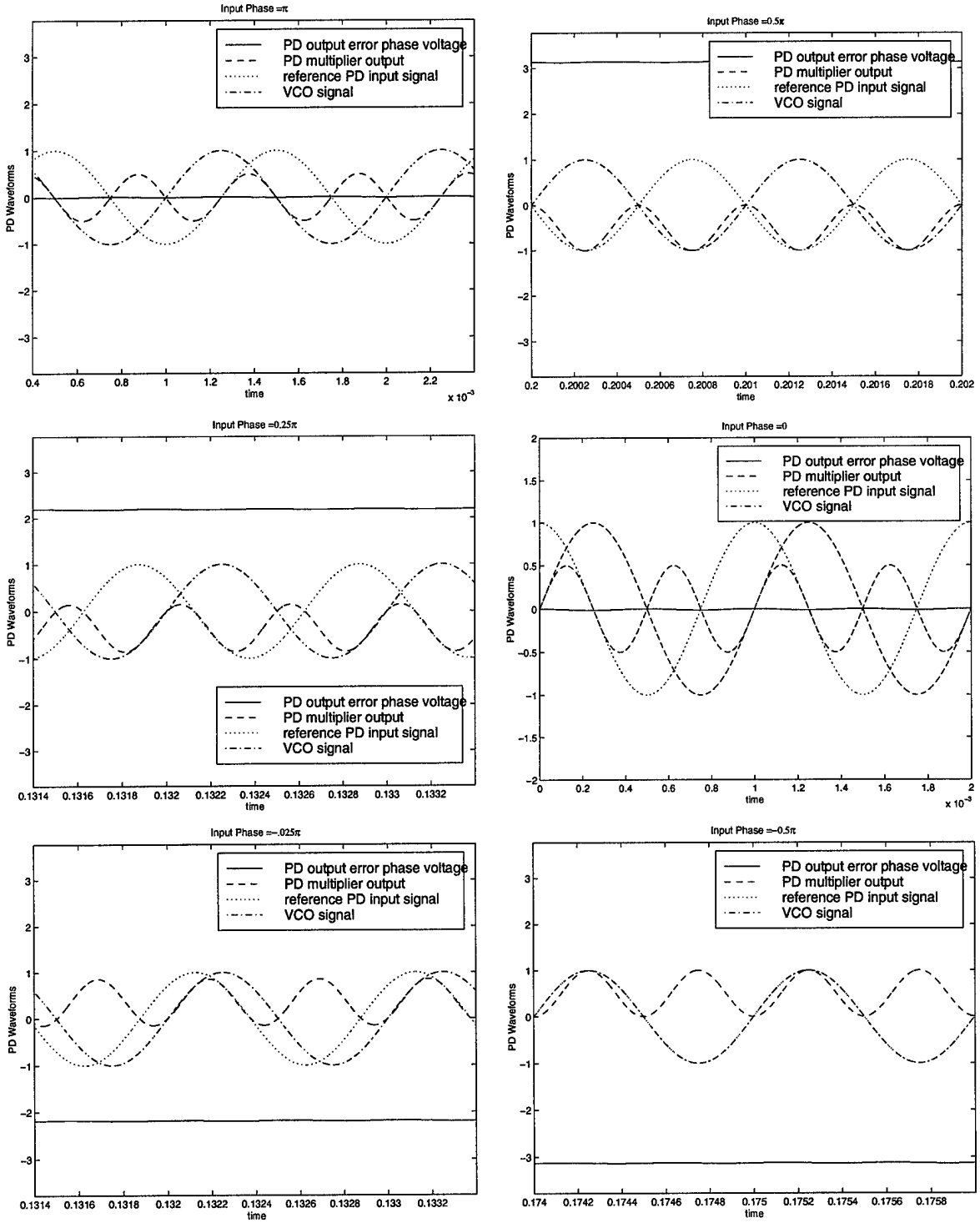


Figure 101 The PD waveforms for input phase error of π, 0.5π, 0.25π, 0, -0.25π and -0.5π

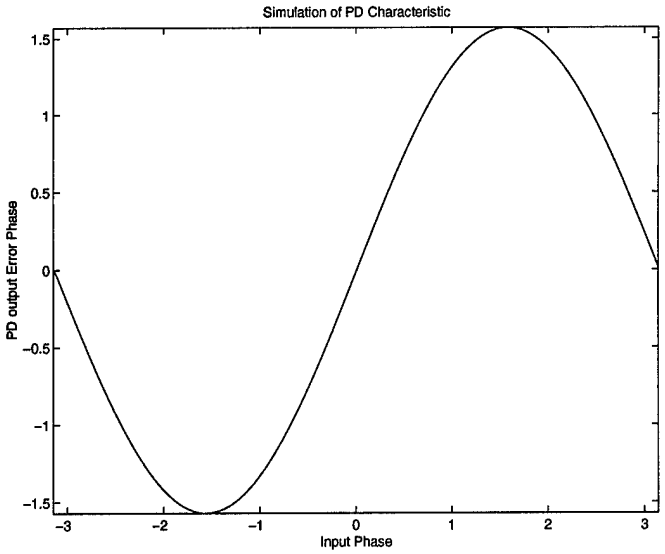


Figure 102 Simulation of Phase Detector Characteristic

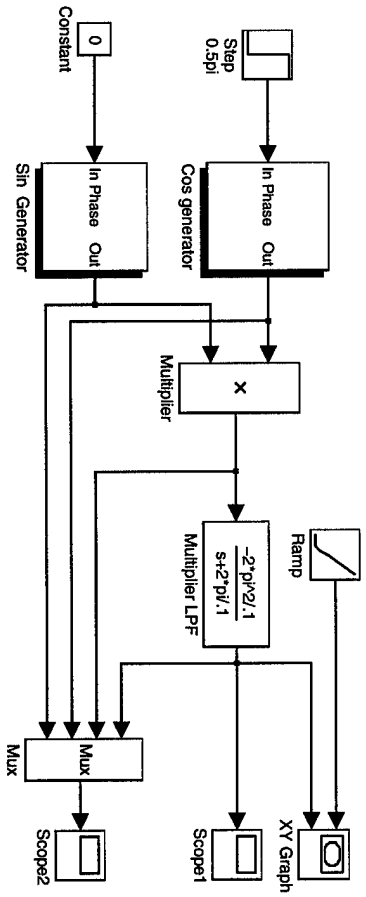


Figure 103 Implementation of Step Response Simulation

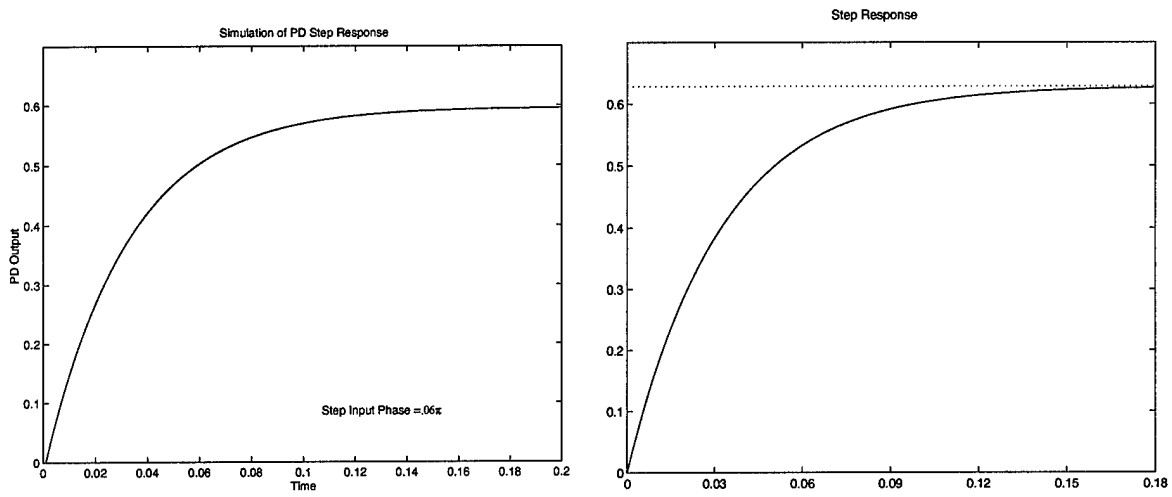


Figure 104 (a) Simulation of Step Response by SIMULINK (b) Analytical Step Response by MATLAB

as the cut-off frequency is become wider the PD characteristic lose its symmetry the phase mapping of the PD I/O becomes less accurate. The demonstration of this asymmetry because of increasing the harmonic in the PD outputs, as it is shown in Figure 107. However the PD transient response is faster when the LPF bandwidth is wider, see Figure 106

The frequency response of the lowpass filter of bandwidth ($BW_{lpf} = 1kHz$) is illustrated in Figure 108

5.3.3 Linear Model and Loop Filter. It is necessary to provide a low pass filter after the PD, to reject carrier frequency components and high frequency noise which is mentioned in the above section. Thus the dynamic performance of the PLL is influenced not only by the type of PD chosen, but also by the type of loop filter used in particular application. In most cases the loop filter is given by a first and second, order low pass filter. The most general form of transfer function for a first-order filer is realized by lead-lag filter:

$$F(s) = \frac{as+b}{cs+d} \tag{123}$$

In designing a PLL system we are free to combine any type of PD with any type of realizable loop filter.The possible combination of the first order LPF are taken for, $b = 0$, $(a, b, c, d \neq 0)$,

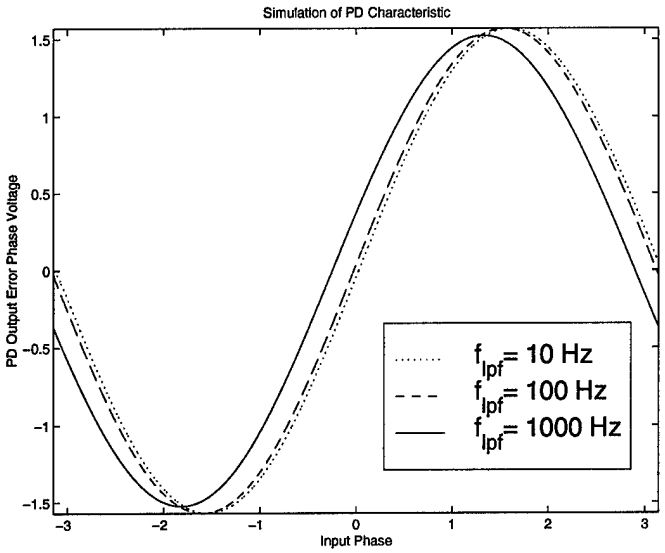


Figure 105 PD characteristics for f_{lpf} =10, 100 ,1000 Hz respectively

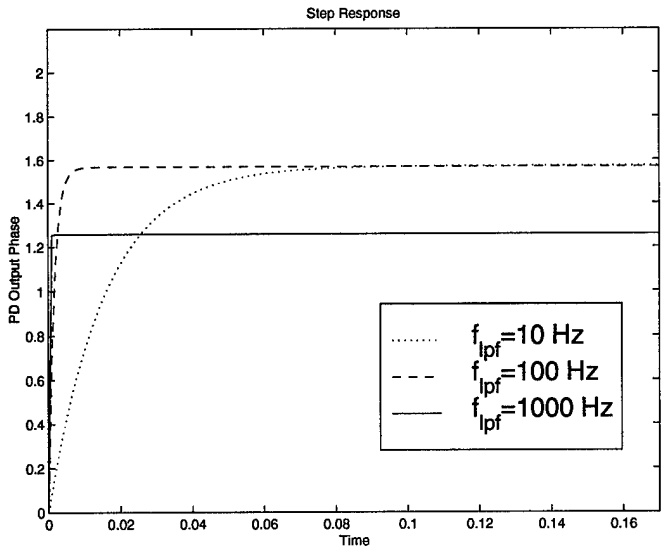


Figure 106 PD Step Response for f_{lpf} =10, 100 ,1000 Hz respectively

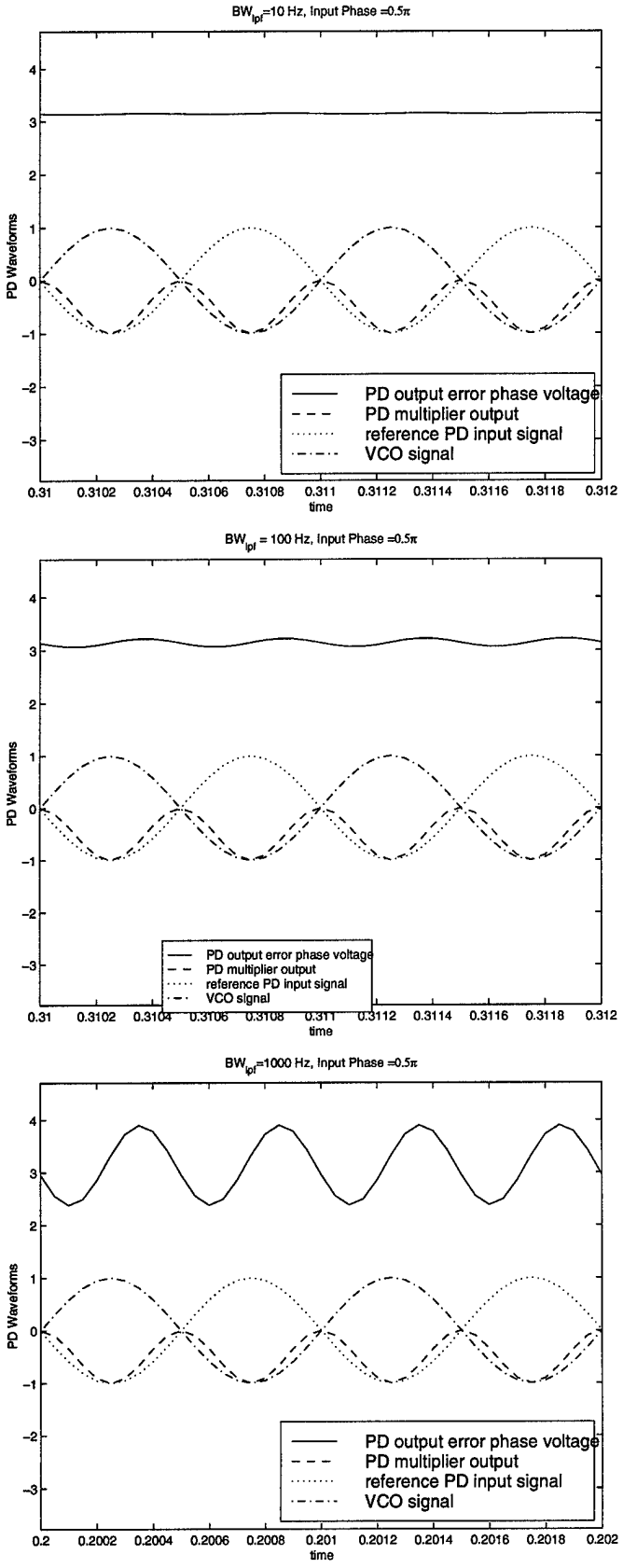


Figure 107 The scopes of PD waveforms for f_{lpf} =10, 100 ,1000 Hz respectively

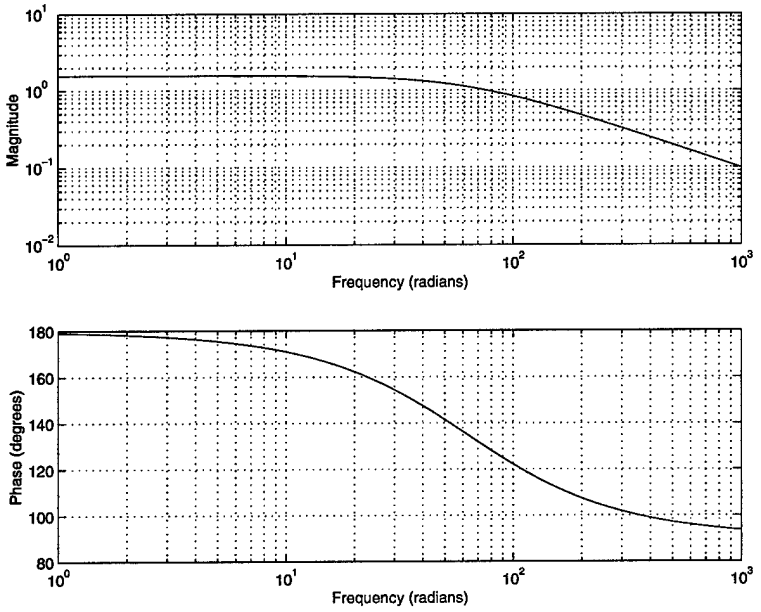


Figure 108 Frequency response of PD Lowpass filter

$c = 0$, or $b = c = 0$. Details of the dynamic response for the PLL system for all of the types of filters and applications are enhancently demonstrated in references [2,8,12,13,23].

The design of the loop filter can be achieved from the dynamic performance analysis of the PLL using the classical control theory. In turn, the PLL transfer function is necessary to be represented in the Laplace domain. The PD characteristic exhibits nonlinearity because the average output signal is a sine function of the phase error see Equation (120) and Figure (102). It is approximated that, the most linear part of the PD characteristic is $\sin(\epsilon) \approx \epsilon$, which let the PLL to be locked at all times. This approximation part is a quite large region of phase error which is valid from $-\pi/3$ to $\pi/3$ (see [2]). A linearized block diagram of the PLL in the complex frequency domain is illustrated in Figure 109.

The Phase transfer function is

$$H(s) = \frac{\Phi_o(s)}{\Phi_{pll}(s)} = \frac{K_o K_d F(s)}{s + K_o K d F(s)} \tag{124}$$

The order of the PLL system is equal to the order of the loop filter plus 1 [2]. The second order PLLs are the most commonly used, then in order to obtain the phase transfer function of the linear second order PLL, we have to substitute the first order transfer function $F(s)$

of Equation (123) in Equation (124). In this section the loop filter is chosen, $F(s) = \frac{s-a}{s}$. Then we get the transfer function $H(s)$

$$H(s) = \frac{K_o K_d(s+a)}{s^2+K_o K_d s + K_o K_d a} \tag{125}$$

Equation (125) can be written in the normalized form as

$$H(s) = \frac{2\xi\omega_n s+\omega_n^2}{s^2+2\xi\omega_n s+\omega_n^2} \tag{126}$$

where ω_n is the natural frequency and ξ is the damping factor. Thus the equivalent substitutions is given by

$$\begin{aligned} \omega_n &= \sqrt{aK_oK_d} \\ \xi &= \frac{K_oK_d}{2\sqrt{aK_oK_d}} \end{aligned} \tag{127}$$

The term K_oK_d is called the loop gain and has the dimension (s^{-1}). If the condition $K_oK_d \gg \omega_n$
is true, this PLL system is said to be a high gain loop [2]. If the reverse is true, the system is called a low gain loop.

Now in the design of the loop filter, we have the constants $K_o = 1$ and $K_d = 2\pi^2 f_{lpf}$, as seen in the previous section. This implys that $\omega_n \ll 2\pi^2 f_{lpf}$. Choose $\xi = 0.707$, then substitute in Equation (127). Then, the loop filter parameter,$a = \frac{2\pi^2 f_{lpf}}{4\xi^2}$. So for $f_{lpf} = 10Hz$, $F(s) = \frac{s+98.73}{s}$. and, the designed PLL transfer function $H(s)$ is given by

$$H(s) = \frac{20\pi^2(s+98.73)}{s^2+20\pi^2 s+1974.6\pi^2} \tag{128}$$

Figure 110(a) shows the simulated step response of the PLL and Figure 110 shows SIMULINK realization. Figure 111 shows the analytical step and frequency responses of the PLL which proofs the performance dynamics of the designed model.

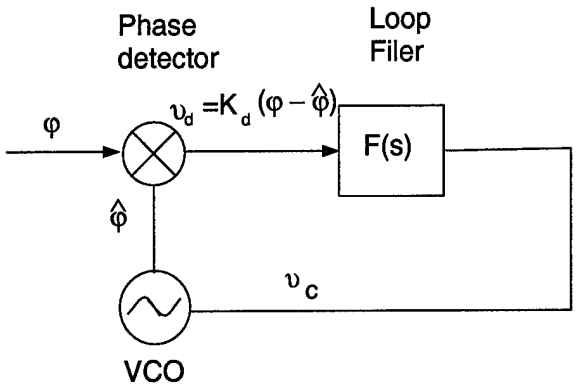


Figure 109 Linear Model of the PLL

5.4 Multipath Effect Upon PLL

In the presence of multipath, carrier tracking cannot distinguish between the direct and reflected signals and continue to employ null tracking; thus, the PLL erroneously estimates the parameters of the direct signal. The PLL aligns a local carrier signal (VCO) with the received carrier signal regardless the shift in its original phase from multipath, Doppler, thermal noise, or whatever other effects. Table 9 shows the PLL performance in the cases of absence and presence of multipath. Figure 112 illustrates the tracking curves for the direct path and multipath cases.

Table 9 Carrier tracking error in the absence and the presence of the multipath

PLL	In the absence of multipath	in the presence of multipath
PLL input (1)	$\sqrt{2}A_oR_c(\tau_o-\hat{\tau}_o)\cos(\omega_ct+\phi_o)$	$R'(t)=\sum_{i=0}^{n}a_iA_oR_c(\tau_o-\hat{\tau}_o-\alpha_iT_c)\cos(\omega_ct-\phi_i)$ or $R'(t)=A_m\cos(\omega_ct+\phi_o+\phi_m)$
VCO output (2)	$\sqrt{2}\sin(\omega_ct+\phi_o)$	$\sqrt{2}\sin\left(\omega_ct+\hat{\phi}_{pll}\right)$ $\phi_3=\phi_o+\phi_m$
Tracking error (3) S-curve	$A_oR_c(\tau_o-\hat{\tau}_o)\sin\left(\phi_o-\hat{\phi}_o\right)$ $\sin\left(\phi_o-\hat{\phi}_o\right)=\sin(\varepsilon)$	$A_m\sin\left(\phi_{pll}-\hat{\phi}_{pll}\right)$ $\sin\left(\phi_o-\hat{\phi}_o+\phi_m-\hat{\phi}_m\right)$

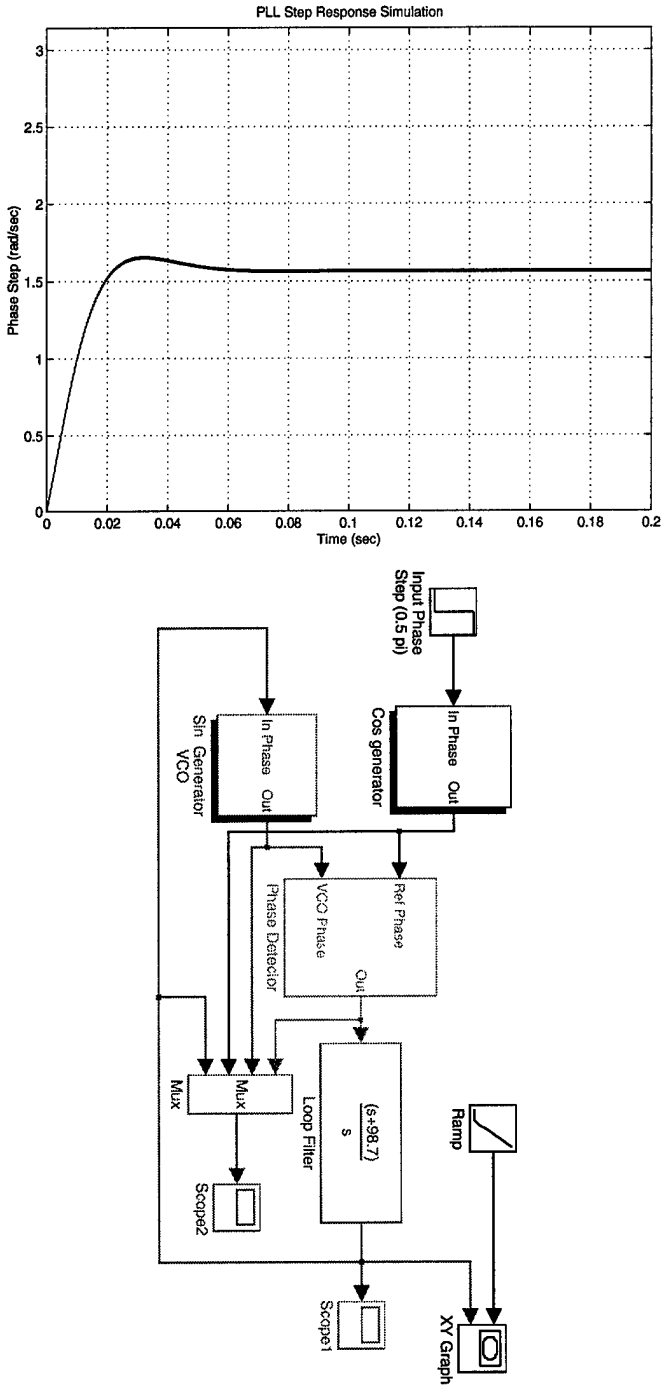


Figure 110 (a)PLL Step Response simulation (b)SIMULINK PLL Realization

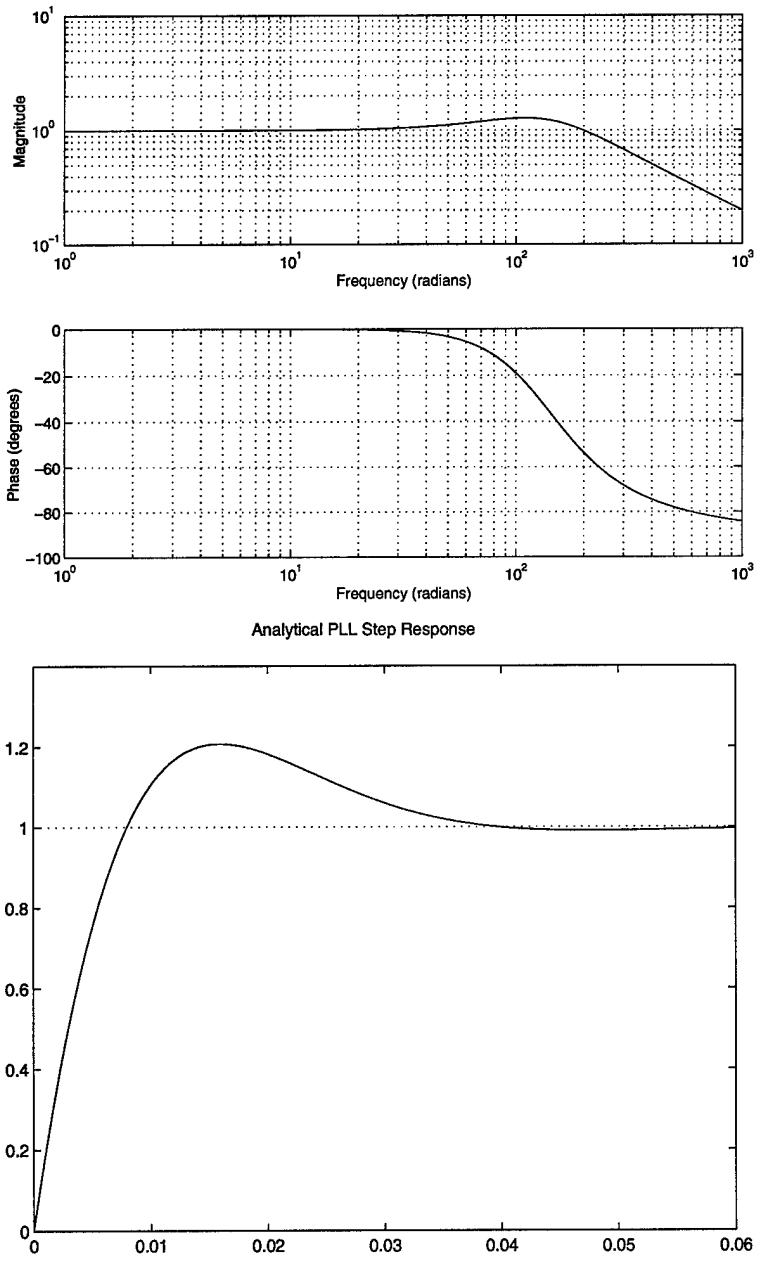


Figure 111 (a)Analytical PLL Step Response (b) PLL Frequency Response

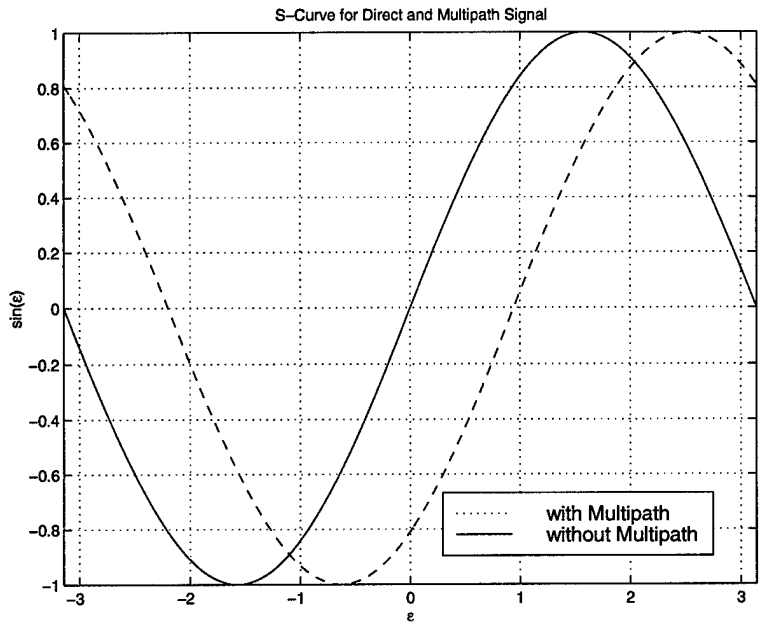


Figure 112 S-Curve of the PLL

VI. Multipath Mitigation Using the Modified Tracking Loops

6.1 Overview

The willfulness of the multipath mitigation approach in this chapter is based on the multipath estimation. The approach is configured as in Figure 51 where three units are essentially used: The multiple measurements, the estimator, and the modified tracking loop. The tracking loops must be modified to exploit the use of the estimated parameters. This chapter includes a new design of a modified tracking loops for both the carrier (modified PLL) and the code tracking (modified DLL). The modified DLL in namely, n-MRDLL, can track the direct path code signal in the presence of multipath.The n-MRDLL is configured with multicorrelator; it works with the coherent channel (in-phase or quadrature). The tracking performance is weighted with the estimated multipath parameters. This modified DLL endowed with the α-deploying estimator that is explained in details in chapter IV. This chapter includes a new general formula of the adaptive loop gain, is used to correct the discriminator slope in cases of weak received direct path signal. The analysis and investigation of n-MRDLL is presented, the tracking and the noise performance is introduced for both the discriminator and the n-MRDLL closed loop. A computer simulation with the results are described and a comparative study between the modified DLL and the standard DLL is performed. The chapter is ended with a new idea of a modified PLL which can track the direct path signal in the presence of the multipath.

6.2 n-MRDLL

The idea behind the n-MRDLL is to force the false equilibrium point due to multipath to be shifted into the true point. Basically this process is achieved in the discriminator characteristic. Laxton, 1996 [14] used this idea for a single multipath component by weighting the the correlator outputs of the discriminator with the estimated multipath component parameters such that the mathematical summation of the correlator outputs is zero as long as the tracking error is zero as well. In this chapter we introduce a generalization of this modified DLL, can work with any number n of the multipath components exploiting the benefits of

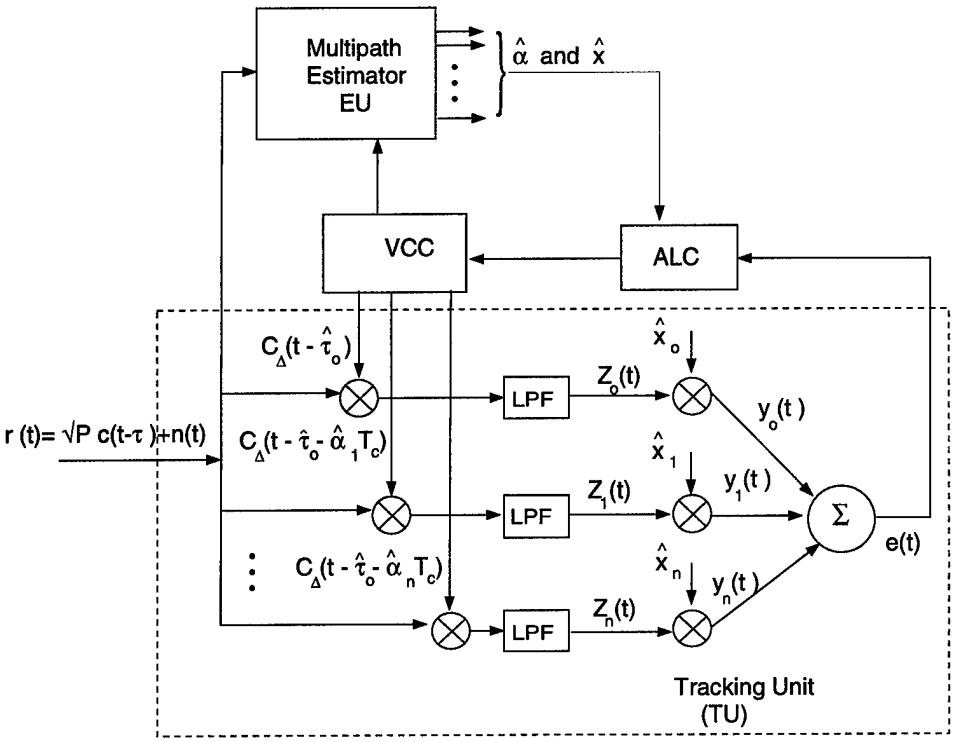


Figure 113 n-MRDLL

the α-deploying estimator which is explained in detail in chapter IV. Figure 113 shows the proposed implementation of the multicorrelator tracking loop for n-Multipath Reflections DLL (n-MRDLL). The multipath estimator block EU is taken as the α-deploying estimator see chapter IV. The tracking unit (TU) must have n correlators to enable mitigation of n reflections. The n number can be estimated using the α-deploying method also. In each correlator, the received signal is correlated with early and late replicas of the PRN code. For an early-late spacing of one code chip period, these shifted replicas are defined as

$$C_\Delta\left(t-\tau_o-\hat{\alpha}_i T_c\right) \triangleq C\left(t-\tau_o-\hat{\alpha}_i T_c-T_c/2\right)-C\left(t-\tau_o-\hat{\alpha}_i T_c+T_c/2\right) \tag{129}$$

In this modification to the MRDLL, the center time of each correlator pair is replaced by an estimated multipath delay $\hat{\alpha}$ which is the deployed α corresponding to nonzero strengths $\hat{x}$. Thus, the number n is changed according to the multipath situation.

6.3 n-MRDLL Analysis

For the purpose of analysis, let us consider the idealized case wherein the EU is providing perfect estimates of the multipath parameters, x, α. In this special case the outputs from the $(n+1)$ lowpass filters in the TU, Figure 113, are

$$\begin{aligned} z_o(t,\delta) &= x_o D(\delta) + \sum_{i=1}^{n} x_i D(\delta+\alpha_i) + \eta_o \\ z_k(t,\delta) &= x_o D(\delta-\alpha_k) + x_k D(\delta) + \sum_{\substack{i=1 \\ i\neq k}}^{n} x_i D(\delta-[\alpha_k-\alpha_i]) + \eta_k; \quad k = 1,\cdots,n \end{aligned} \tag{130}$$

The D-curve is as so called in [10,14]defined as

$$D(\delta) \triangleq R_c\left(\delta-\frac{1}{2}\right) - R_c\left(\delta+\frac{1}{2}\right) \tag{131}$$

and

$$\begin{aligned} \eta_k &= \sqrt{2}\left[\eta_k^I(t)\cos(\phi_3) + \eta_k^Q(t)\sin(\phi_3)\right] \\ \eta_k^I(t) &\triangleq n_I(t) C_\Delta\left(t-\hat{\tau}_o-\alpha_k T_c\right) * h_{lpf}(t) \\ \eta_k^Q(t) &\triangleq n_Q(t) C_\Delta\left(t-\hat{\tau}_o-\alpha_k T_c\right) * h_{lpf}(t) \end{aligned} \tag{132}$$

where * denotes convolution, and h_{lpf} is the impulse response of the LPFs. The PSD of $\eta_k^I(t)$ and $\eta_k^Q(t)$ is equivalent to the PSD derived in Equation (22), thus the PSD $G_{\eta_k^I}$ and $G_{\eta_k^Q}$ is derived for correlator spacing of $0.5T_c$ which is given by

$$G_{\eta_k^I}(f) = G_{\eta_k^Q}(f) = N_o\left(1+\frac{1}{N}\right) \tag{133}$$

Therefore, the output noise of the k^{th} LPF has a two-sided PSD of

$$G_{\eta_k}(f) = \begin{cases} N_o\left(1+\frac{1}{N}\right) & f \leq \frac{B_{LPF}}{2} \\ 0 & \text{elsewhere} \end{cases} \tag{134}$$

Assuming again perfect estimation, after mixing with $\hat{x}_i$, the outputs of the gain phase correlators are

$$\begin{aligned} y_o(t,\delta) &= x_o^2 D(\delta) + \sum_{i=1}^{n} x_o x_i D(\delta+\alpha_i) + x_o \eta_o \\ y_k(t,\delta) &= x_o x_k D(\delta-\alpha_k) + x_k^2 D(\delta) + \sum_{\substack{i=1 \\ i\neq k}}^{n} x_k x_i D(\delta-[\alpha_k-\alpha_i]) + x_k \eta_k; \quad k = 1, \cdots, n \end{aligned} \tag{135}$$

Thus, the output of the TU discriminator is

$$e(t,\delta) = \sum_{i=0}^{n} y_i(t,\delta) = S(\delta) + n_e(t) \tag{136}$$

The tracking loop or S-curve of the TU is given by

$$S(\delta) = \sum_{i=1}^{n} x_i^2 D(\delta) + \sum_{i=1}^{n} x_o x_i \left[D(\delta+\alpha_i) + D(\delta-\alpha_i) \right] + \sum_{k=1}^{n} \sum_{i=k+1}^{n-1} x_k x_i \left[D(\delta+\alpha_{ki}) + D(\delta-\alpha_{ki}) \right] \tag{137}$$

where $\alpha_{ki} = \alpha_k - \alpha_i$ and the discriminator output noise, $n_e(t)$, is given by

$$n_e(t) = \sum_{i=0}^{n} x_i \eta_i \tag{138}$$

The discriminator output noise PSD can be given by taking the Fourier transform of the autocorrelation function of the noise $E\{n_e(t) n_e(t+\tau)\}$, thus the PSD can be derived as

$$G_{n_e}(f) = \begin{cases} \sum_{i=0}^{n} x_i^2 \left(2N_o \left(1 + \frac{1}{N}\right)\right) & f \leq \frac{B_{LPF}}{2} \\ 0 & \text{elsewhere} \end{cases} \tag{139}$$

The steady-state tracking error of the n-MRDLL (neglecting noise effects) is represented by the S-curve of the TU. The S-curve represented in Equation (137) consists of 3 terms, each of which is symmetric around the tracking point and is zero when $\delta = 0$; therefore, with perfect estimation, the n-MRDLL will track the direct signal in the presence of

multiple reflected signals. To illustrate this situation let us present the following example

Figure 114 illustrates the difference between the S-curve of standard DLL and the n-MRDLL wherein the signal parameters are given as
$n = 10$, $i = 1 \cdots 10$, $\alpha_i = i \times 0.1$, $x_i = e^{-\alpha_i}$

Table 10 A Numerical Example of 10 reflectors applied into n-MRDLL

i	0	1	2	3	4	5	6	7	8	9	10
α_i	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
x_i	1.0	0.90	0.81	0.74	0.67	0.61	0.55	0.50	0.45	.41	.37

Let

$$S_3(\delta) = \sum_{k=1}^{n} \sum_{i=k+1}^{n-1} x_k x_i \left[D\left(\delta + \alpha_{ki}\right) + D\left(\delta - \alpha_{ki}\right) \right]$$

$$S_2(\delta) = \sum_{i=1}^{n} x_o x_i \left[D\left(\delta + \alpha_i\right) + D\left(\delta - \alpha_i\right) \right]$$

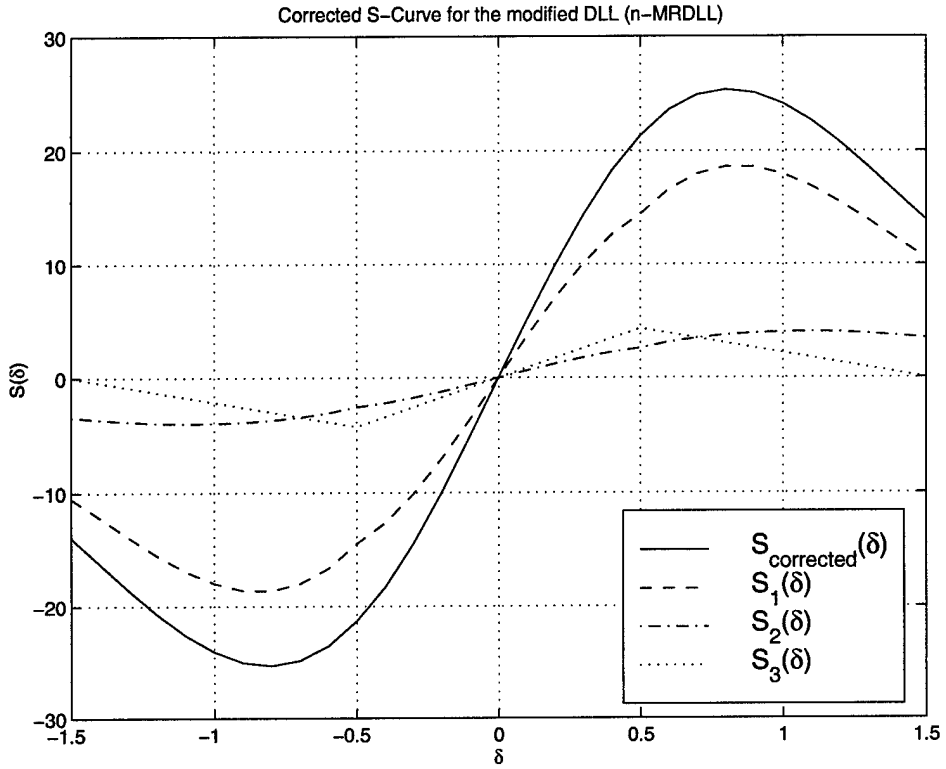
$$S_1(\delta) = \sum_{i=1}^{n} x_i^2 D(\delta)$$

S then,

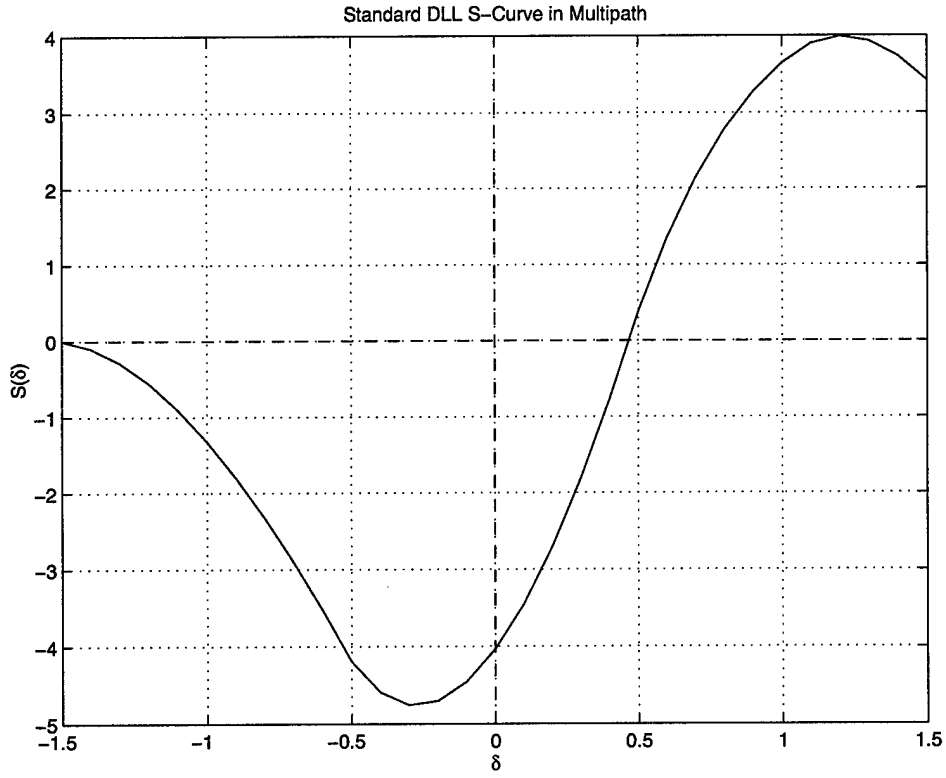
$$S(\delta) = S_1(\delta) + S_2(\delta) + S_3(\delta)$$

6.4 n-MRDLL Discriminator Characteristics

The investigation of the n-MRDLL discriminator characteristic is important. The idea of the n-MRDLL is to remove the bias due to multipath and to permit the characteristic of the discriminator to properly cross the zero point or the equilibrium point. However the problem yet be solved so far, because of the additional multipath power added to the direct path signal and the way of this addition is different according to each multipath scenario situation. Indeed, the n-MRDLL discriminator characteristic crosses the equilibrium point, while a nonlinearity might be induced and the S-curve slope could be changed, one might pose questions about the n-MRDLL S-curve, which are: 1) How far does nonlinearity approach



(a)



(b)

Figure 114 (a) The n-MRDLL S-curve (b)The Standard DLL S-curve, in the presence of multipath

the operating region around the equilibrium point due to multipath?, 2) How much is the change in the S-curve slope due to multipath?, 3) Is the slope change either harmful or useful during the tracking?. The answer to these questions will be introduced in next sections.

6.4.1 The Effect of the Multipath Strength Parameter x. The two multipath parameters influencing the S-curve are the time delay α and the strength parameter x. The strength parameter x effects the power added to the direct path signal. Thus, it can magnify the noise as it was explained in Chapter II, as well as it makes the slope rotate around the equilibrium point. In order to illustrates these insights, we use two additional parameters namely, κ and λ. The parameter κ represents a reduction factor of the multipath strength, and the parameter λ represents an additional shift added to the time delay vector, α. Actually, the parameter λ is taken zero in this section because the shift of α is not yet considered and the change of the parameter κ is taken from 90% to 0% of multipath strength. The multipath scenario is taken as in Example 1 of α=[0.0 0.2 0.5 0.7 1.0 1.3] and x=[1.0 0.7 0.6 0.4 0.8 0.3]. Thus, the reduction of x can be considered in all elements of vector x, including the direct path as $x_{reduction} = \kappa \ . \ x$ or the reduction can be performed to the multipath signal only as $x_{reduction}$=[1 κ . x_{mp}] and x_{mp}=[0.7 0.6 0.4 0.8 0.3]. In addition, the consideration of reduction in all of the x elements is also useful because in some situations the received signal sometimes hasn't the required power to establish the proper characteristic of the discriminator.

Figure 115 shows the S-curve for the standard DLL without multipath, the standard DLL with multipath and the n-MRDLL S-curve when no reduction or shift is considered for the given α or x. Clearly the plots in this Figure show that the S-curve of the standard DLL without multipath is smaller than the S-curve of the n-MRDLL. This magnification of the S-curve occurs because of the power added by the multipath reflectors. The magnification of the n-MRDLL is caused by the weighting factor of the estimated multipath strength which is multiplied by the output signal of the correlator in each branch of the TU. The second observation is that the n-MRDLL S-curve slope is greater than the standard S-curve which means that the n-MRDLL discriminator characteristic has a strong performance of the tracking process as well. Thus, the n-MRDLL S-curve slope depends on the values of the strength

parameter x. Thus, the n-MRDLL is useful when strong multipath components are detected.

Now, consider the reduction of the multipath only, viz $x_{reduction}$=[1 $\kappa \cdot x_{mp}$] The reduction factor κ is taken from 90% to 0%, the percentage 0% meaning the operation of the n-MRDLL without multipath. In order to make a significant comparison, let's superimpose the punch of S-curves, as in Figure 116, which shows the standard DLL S-curve without multipath, the punch of standard S-curves with multipath as κ changed from 90% to 0%, and the S-curves of the n-MRDLL as well. We observe the following:

- The n-MRDLL S-curve slope at $\delta = 0$ is always greater than the standard curve as long as the multipath components have a significant values.
- The n-MRDLL S-curve slope is always greater than the standard one. It converges to the standard curve as the multipath is reduced.
- No further distortion to the original n-MRDLL appeared with the reduction of x.

From these observations we conclude the following

1. The normalised S-curve has a slope of 2 in $-0.5 \leq \delta \leq 0.5$. Typically, the normalized n-MRDLL S-curve becomes the same plotting curves as in Figure 116. Thus the slopes of these normalized n-MRDLL S-curves is greater than 2 and mostly these curves are linear around the zero point. The calculation of these slopes will be presented in the next section.
2. The study of the standard DLL shows that, the relative values between the discriminator input signal-to-noise ratio, SNR (which corrupts the input code amplitude) and the size of the S-curve causes a high degradation in the noise performance at the discriminator output. In turn, this causes a severe instability of the closed tracking loop, especially in the presence of multipath. Thus, if this relativity is gets smaller, then the tracking performance gets better even in the case of low SNR. Therefore, the large n-MRDLL discriminator characteristic due to multipath may be useful for tracking even though there is a trade-off between the noise and the number of correlators in the n-MRDLL.

3. According to the discussion in 2), we conclude that the stronger multipath is useful in the n-MRDLL rather than the weaker multipath.

Changes in the multipath strength's only have been discussed, so the question now is: What happens if the reduction is in the vector x including the direct signal. Figure 117 shows the punch of curves of the standard and the n-MRDLL with multipath when $x_{reduction} = \kappa \cdot x$ The significant observation in this curves is that the n-MRDLL S-curves are reducted until it becomes close to zero. This result is intuitively obvious, for the cases where a weak signal even in the standard case. The n-MRDLL S-curves in Figure 117 under the standard S-curve (doted one) have corresponding gains less than the discriminator gain. Basically, the gain of the discriminator is the slope of the S-curve at the origin and this gain depends on the strength of direct signal and multipath. As mentioned in the previous paragraph that the multipath increases the slope but not affect the direct path signal or the standard S-curve. Laxton, 1996 [14] introduced a correction to such slopes and he called it the adaptive loop gain A, and he divided the loop filter transfer function by this gain, so that the resulting gain compensated the change in the modified loop due to multipath of one reflector.

In the next section we will derive a general formula of this adaptive gain A valid for any number of multipath components n.

6.4.2 Adaptive Loop Gain. The objective of the adaptive loop gain is to undo the changes in n-MRDLL discriminator characteristics (S-curves) due to multipath. The n-MRDLL S-curve slope can be calculated by taking the derivative of the S-curve w.r.t δ at the zero point. Recall Equation (137) and taking its derivative w.r.t δ at $\delta = 0$ yields

$$\frac{dS(\delta)}{d\delta}|_{@\delta=0} = \frac{dS_1(\delta)}{d\delta}|_{@\delta=0} + \frac{dS_2(\delta)}{d\delta}|_{@\delta=0} + \frac{dS_3(\delta)}{d\delta}|_{@\delta=0} \tag{140}$$

Equation (140) can also be written as

$$A = A_1 + A_2 + A_3 \tag{141}$$

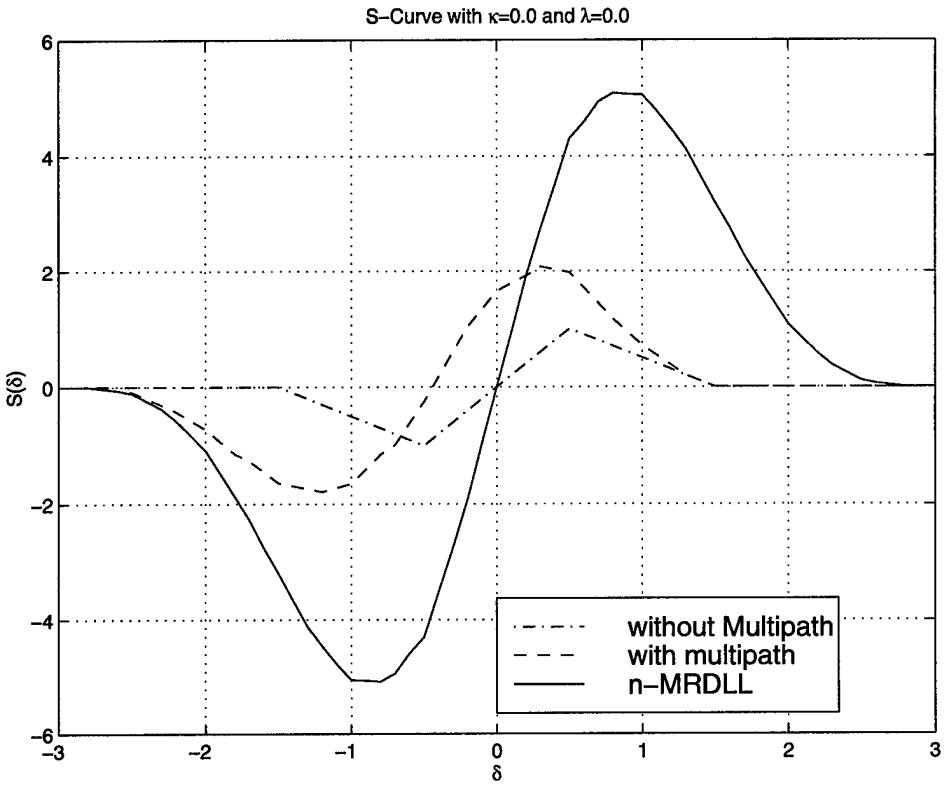


Figure 115 The Typical S-curve for the standard DLL and the n-MRDLL

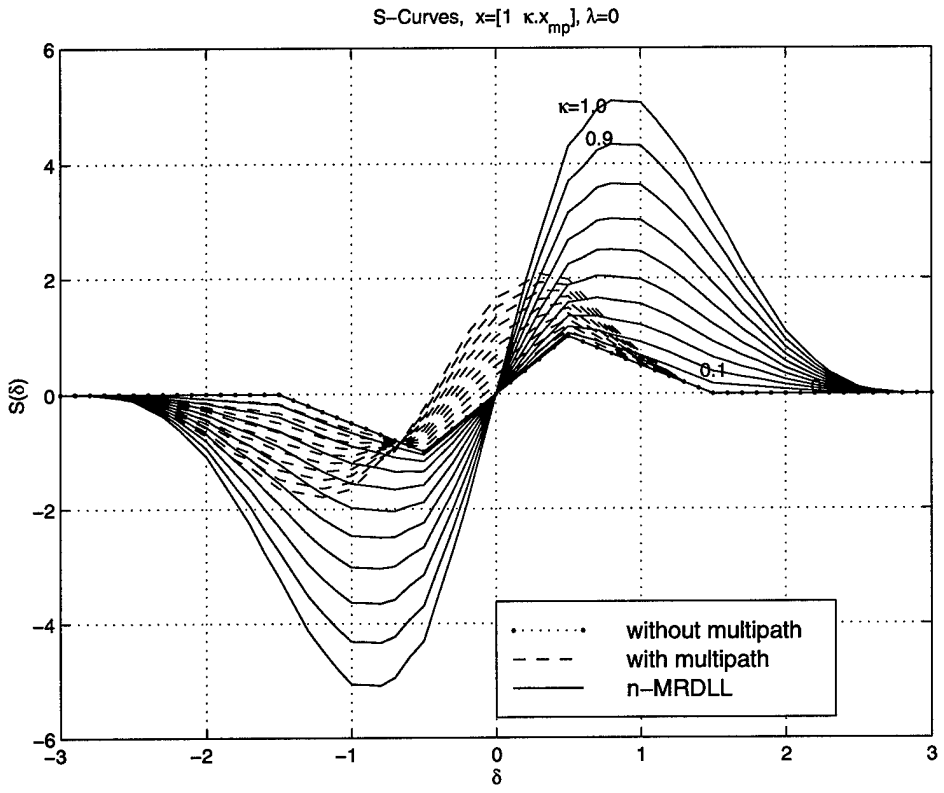


Figure 116 Typical S-curves of the standard DLL with and without multipath, and n-MRDLL, the reducing factor, $\kappa = 0, 0.1, ..., 0.9$, x=[1, $\kappa \cdot x_{mp}$]

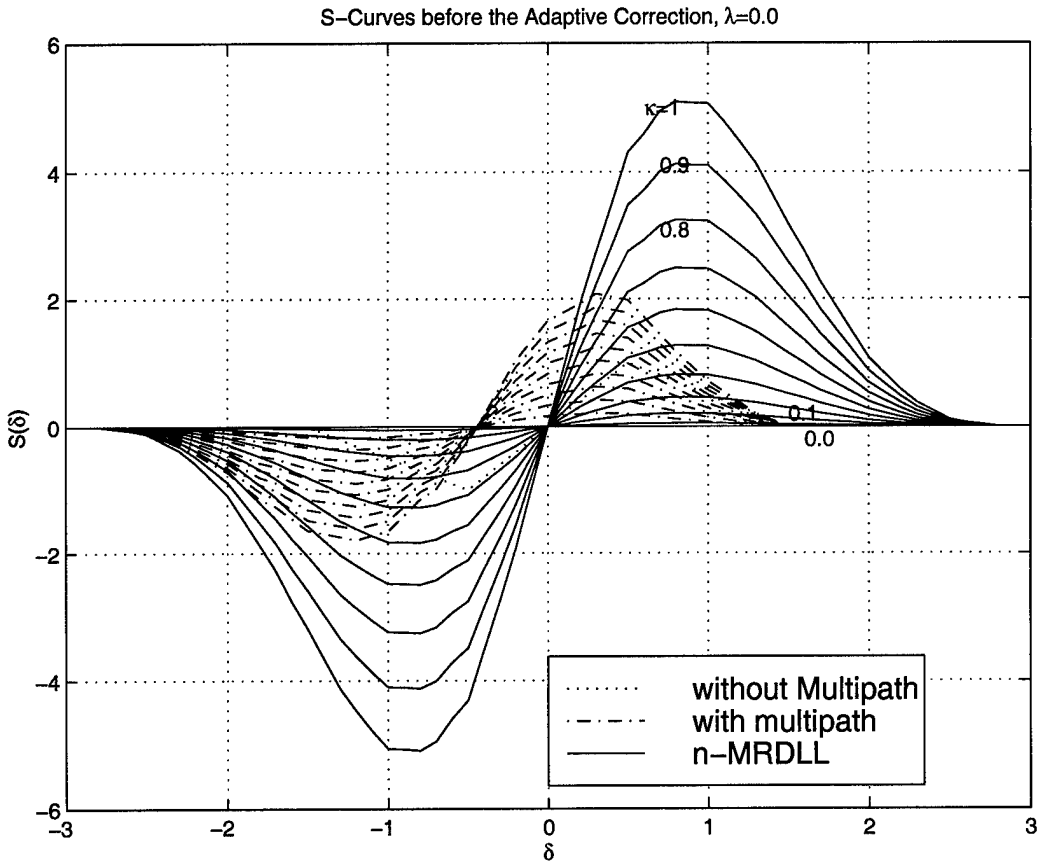


Figure 117 Typical S-curves of the standard DLL and n-MRDLL with and without multipath, $x_{reduction} = \kappa \cdot x$, $\kappa = 0.0, 0.1, ..., 1.0$

where

$$
\begin{aligned}
A_1 &= \tfrac{dS_1(\delta)}{d\delta}|_{@\delta=0} = \sum_{i=1}^{n} x_i^2 \frac{dD(\delta)}{d\delta}|_{@\delta=0} \\
A_2 &= \tfrac{dS_2(\delta)}{d\delta}|_{@\delta=0} = \textstyle\sum_{i=1}^{n} x_o x_i \frac{d[D(\delta+\alpha_i)+D(\delta-\alpha_i)]}{d\delta}|_{@\delta=0} \\
A_3 &= \tfrac{dS_3(\delta)}{d\delta}|_{@\delta=0} = \sum_{k=1}^{n} \sum_{i=k+1}^{n-1} x_k x_i \frac{d\left[D\left(\delta+\alpha_{ki}\right)+D\left(\delta-\alpha_{ki}\right)\right]}{d\delta}|_{@\delta=0}
\end{aligned}
\tag{142}
$$

Using the D-curve of Equation (131) then,

$$
\frac{dD(\delta)}{d\delta}|_{@\delta=0} = 2 \tag{143}
$$

Using Equation (142) and (143), the first term is given as

$$
A_1 = \frac{dS_1(\delta)}{d\delta}|_{@\delta=0} = 2\sum_{i=1}^{n} x_i^2 \tag{144}
$$

$$
\frac{dD\left(\delta+\alpha_i\right)}{d\delta}|_{@\delta=0} = \begin{cases} 2(1+\alpha_i) & -0.5 \le \alpha_i \le 0.5 \\ 0.5-\alpha_i & 0.5 \le \alpha_i \le 1.5 \end{cases} \tag{145}
$$

$$
\frac{dD\left(\delta-\alpha_i\right)}{d\delta}|_{@\delta=0} = \begin{cases} 2(1-\alpha_i) & -0.5 \le \alpha_i \le 0.5 \\ 0.5+\alpha_i & 0.5 \le \alpha_i \le 1.5 \end{cases} \tag{146}
$$

Since we have $\alpha_i \ge 0$, always then for $0 \le \alpha_i \le 0.5$

$$
\frac{dD\left(\delta+\alpha_i\right)}{d\delta}|_{@\delta=0} = 2(1+\alpha_i) \tag{147}
$$

and for $0.5 \le \alpha_i \le 1.5$

$$
\frac{dD\left(\delta+\alpha_i\right)}{d\delta}|_{@\delta=0} = 0.5-\alpha_i \tag{148}
$$

Similarly, for $0 \le \alpha_i \le 0.5$

$$
\frac{dD\left(\delta-\alpha_i\right)}{d\delta}|_{@\delta=0} = 2(1-\alpha_i) \tag{149}
$$

and for $0.5 \le \alpha_i \le 1.5$

$$\frac{dD(\delta+\alpha_i)}{d\delta}|_{@\delta=0} = 0.5+\alpha_i \tag{150}$$

let $\alpha_m = 0.5$ then the second term is given as

$$A_2 = \frac{dS_2(\delta)}{d\delta}|_{@\delta=0} = 4\sum_{i=1}^{m} x_o x_i + \sum_{i=m}^{n} x_o x_i \tag{151}$$

Equations (144) and (151) implies that the slopes for both S_1 and S_2 are independent on α while they are dependent on x. By using the similarity of the third term with the second term, then, the slope of S_3 is apparent as

$$A_3 = \frac{dS_3(\delta)}{d\delta}|_{@\delta=0} = 4\sum_{k=1}^{m}\sum_{i=k+1}^{n-m} x_k x_i + \sum_{k=m}^{n}\sum_{i=k+1}^{n-1} x_k x_i \tag{152}$$

The slope, A, of the n-MRDLL S-curve is independent on the time delay α and is dependent on the multipath strength x, and the index of $\alpha_m = 0.5$ which means that the slope of the discriminator characteristic is changed according to the change in the vector $\hat{x}$ only.
So far the normalized discriminator characteristic of the n-MRDLL is changed according to the change of A per each multipath situation. Figure 118 shows the relation between the adaptive loop gain for both $x_{reduction} = [1, \kappa \cdot x_{mp}]$ and $x_{reduction} = \kappa \cdot x$ respectively. Thus, dividing the S-curve of Equation (140) by the adaptive gain, then the resulting n-MRDLL S-curves corrected by this adaptive gain is shown in Figure 119 and 120 for the reduction is made on multipath only. Figure 121 shows the resulting n-MRDLL when the reduction is made for the whole x. Thus, we observe the following

1. For $x_{reduction} = [1, \kappa \cdot x_{mp}]$.
   - when x_{mp} is reduced to zero (i.e. no multipath is accounted for) the slope of the n-MRDLL is increased from 2 to 4
   - All the n-MRDLL slopes are approximately the same and coincide with the standard S-curve.

- All the n-MRDLL slopes of κ start to increase from 0.4 to 0.0 and all of those are greater than the standard one.

2. For $x_{reduction} = \kappa \cdot x$ all the n-MRDLL slopes before adaptive correction are reduced to the same slope and approximately coincide with the standard slope.

The conclusions are taken from the previous observations are:

- From item 1) the adaptive loop gains with κ changing from 1.0 to 0.5 are greater than 2 (the standard slope); then, the division by those A's gets the n-MRDLL slope smaller, and vice versa (see Figures 119 and 120). Therefore, it is recommended to use the adaptive loop gain correction when $A < 2$ in order to get a steeper slope for n-MRDLL. However it is better to keep the actual strong slopes without change (i.e. for $A \geq 2$). Again, this recommendation can be complemented by using the α-deploying estimator because it is able to give a layout of the multipath strengths, and the parameter x at any instant of measurement.
- From item 2) the same situation as in item 1) holds, except the correction here is includes the correction of the discriminator gain.

6.4.3 The Effect of the Time Delay α. In this section we will choose $\kappa = 0$ viz the multipath strength parameter x is constant and we will change the time delay α by a shift parameter, namely, λ such that $\alpha_{shift} = \alpha + \lambda$. The shift will lead the last component to be greater than the threshold value of $1.5T_c$, so in this study we will discard these components once they exceeds $1.5T_c$. By using our example, Table 11 shows the shift of α which is taken as α=[0.0, 0.2, 0.5, 0.7, 1.0, 1.3]. Figures 122 through 124 show the n-MRDLL and the standard DLL S-curves before applying the adaptive gain correction and Figure 125 through 127 show the S-curves after the adaptive correction. The important observation on these curves before adaptive correction is that the slope of the S-curves doesn't change with the change of α, except that there appeares a distortions in the nonlinear parts of these S-curves. The lack of change in the n-MRDLL S-curves slopes before the adaptive correction proves that the adaptive gain is not a function of α as seen in the previous section. Therefore, the n-MRDLL slopes are always greater than the standard slopes. Hence, it is recommended

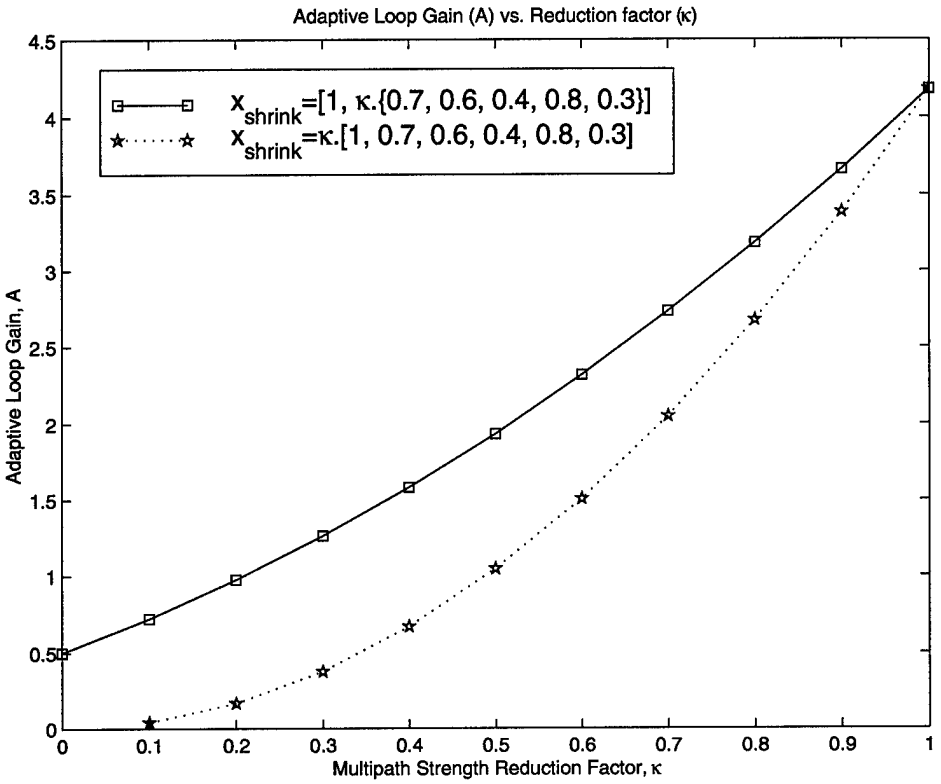


Figure 118 The adaptive gain A versus the the reduction factor κ

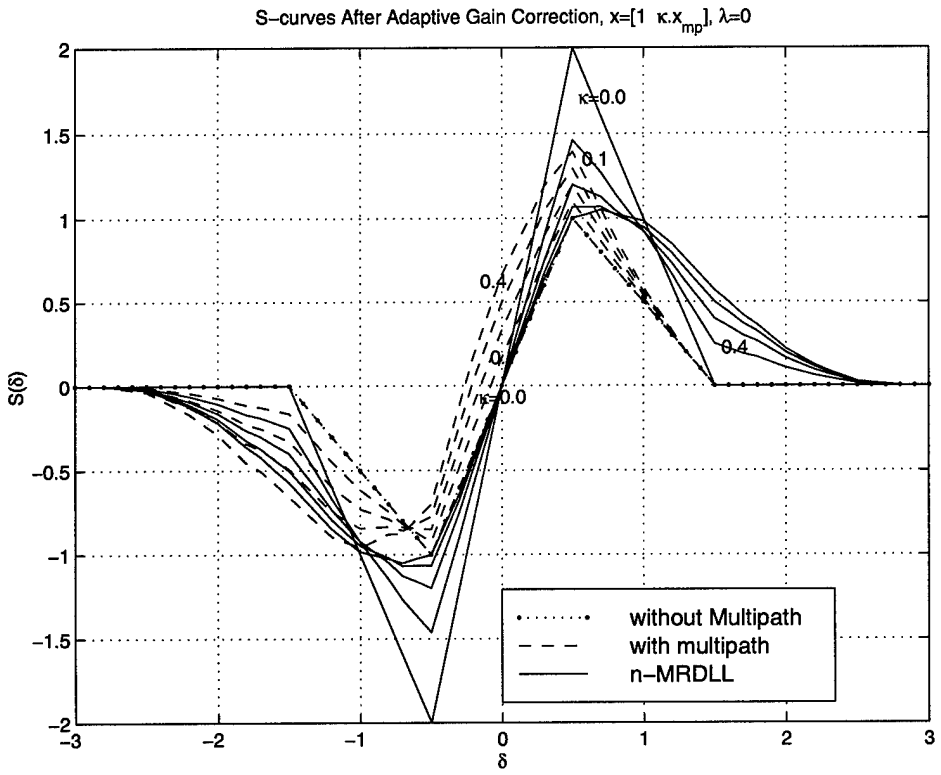


Figure 119 Typical n-MRDLL and standard DLL S-curves, after the adaptive gain correction, $\kappa = (0 : 0.4)$, x=[1, $\kappa \cdot x_{mp}$]

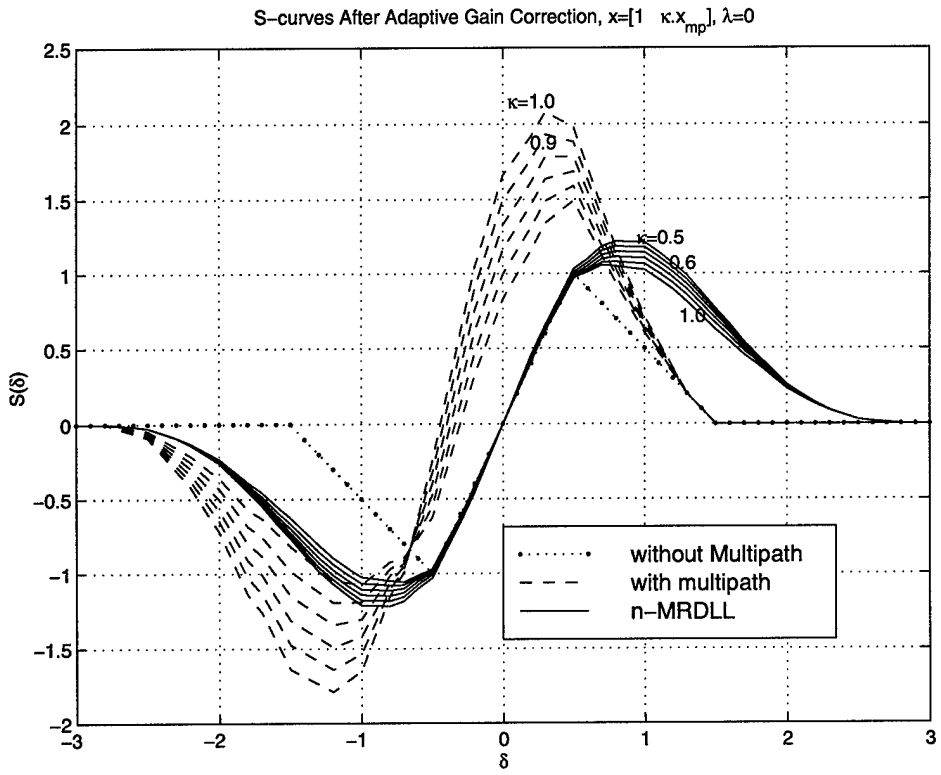


Figure 120 Typical n-MRDLL and standard DLL S-curves, after the adaptive gain correction, $\kappa = (0.5 : 1.0)$, $x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

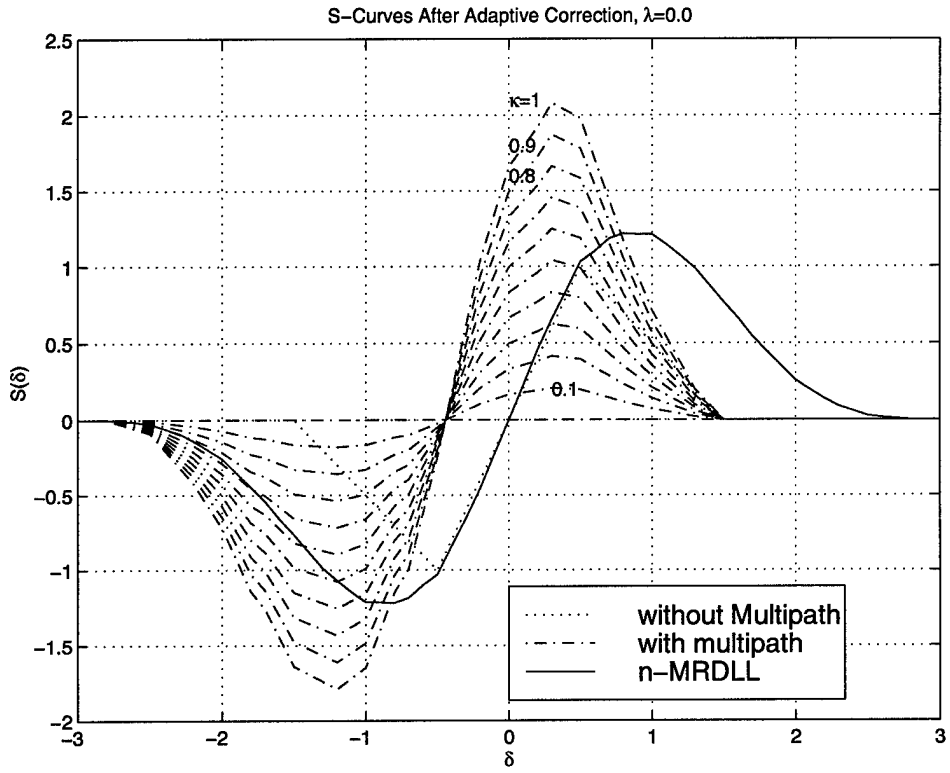


Figure 121 Typical n-MRDLL and standard DLL S-curves, after the adaptive gain correction, $x_{reduction} = \kappa \cdot x$, $\kappa = 0.0 : 1.0$

to not use the adaptive correction if the multipath components are moved with no change in their strengths

Table 11 The shift of α versus λ

λ	α_1	α_2	α_3	α_4	α_5	α_6
0.1	0	0.3	0.6	0.8	1.1	1.4
0.2	0	0.4	0.7	0.9	1.2	1.5
0.3	0	0.5	0.8	1.0	1.3	0
0.4	0	0.6	0.9	1.1	1.4	0
0.5	0	0.7	1.0	1.2	1.5	0
0.6	0	0.8	1.1	1.3	0	0
0.7	0	0.9	1.2	1.4	0	0
0.8	0	1.0	1.3	1.5	0	0
0.9	0	1.1	1.4	0	0	0
1.0	0	1.2	1.5	0	0	0

6.5 n-MRDLL Linear Equivalent Circuit

The n-MRDLL discriminator characteristic have been investigated in the previous section, and we have found that the slope of the n-MRDLL S-curve is different from the standard. This difference reflects the changes to the normalized n-MRDLL discriminator characteristic, versus the standard, which always entails an increasing in the n-MRDLL slopes. This increase because of the number of n-MRDLL correlators is greater than the standard DLL. The increase in the discriminator gain in n-MRDLL changes the tracking performance of the closed loop. In this section we will introduce the analysis of the n-MRDLL closed loop. When considering the discriminator operating region is the most linear part in its characteristic viz, δ close to the zero point, the output of the discriminator can be represented as

$$e(\delta) = A\delta \tag{153}$$

Using the linear equivalent circuit of Figure 9 the n-MRDLL closed loop transfer function H_{nr} can be written as

$$H_{nr} = \frac{\hat{\tau}_o(s)}{\tau_o(s)} = \frac{Ag_cF(s)}{s+Ag_cF(s)} \tag{154}$$

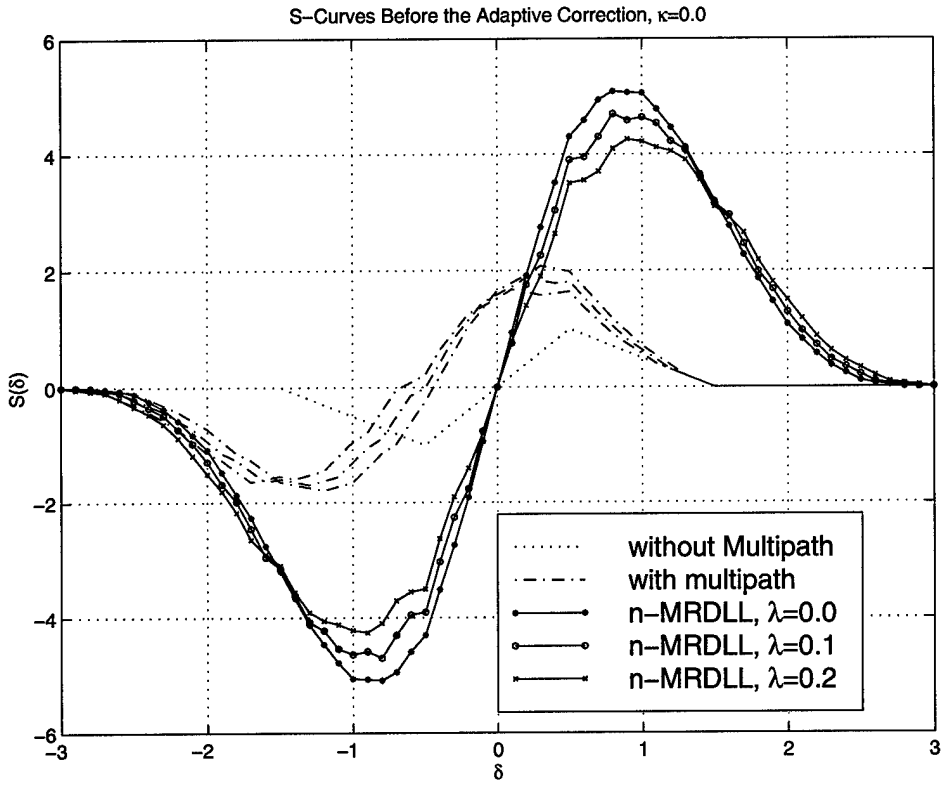


Figure 122 Typical S-curves of the standard DLL and n-MRDLL, $\lambda = 0.0, 0.1, 0.2$, $\kappa = 0.0$

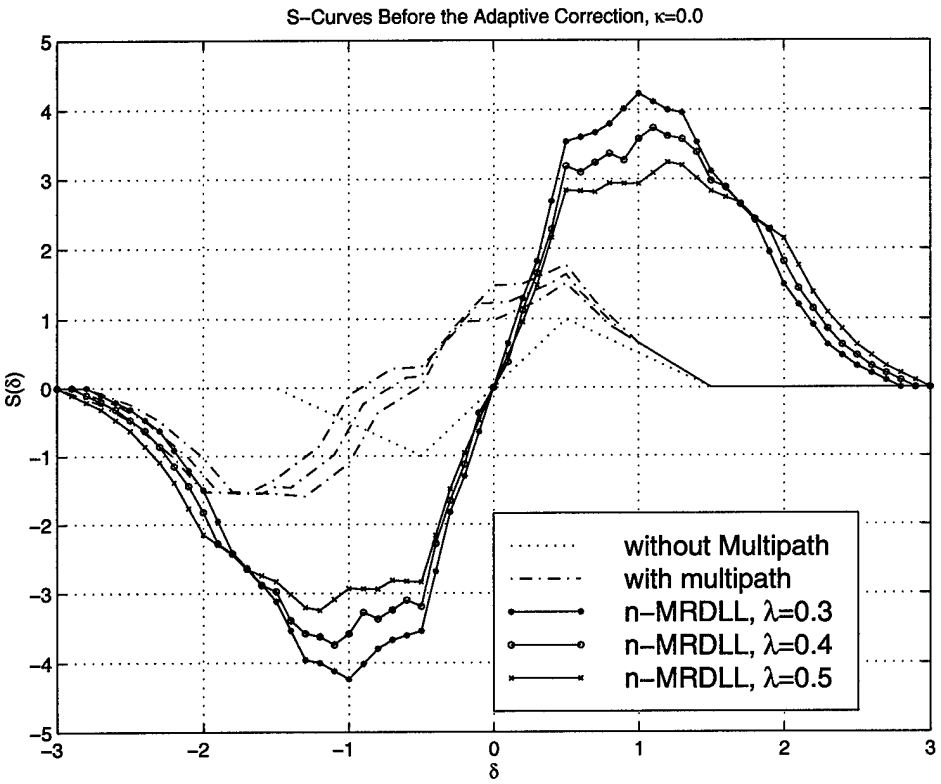


Figure 123 Typical S-curves of the standard DLL and n-MRDLL, $\lambda = 0.3, 0.4, 0.5$, $\kappa = 0.0$

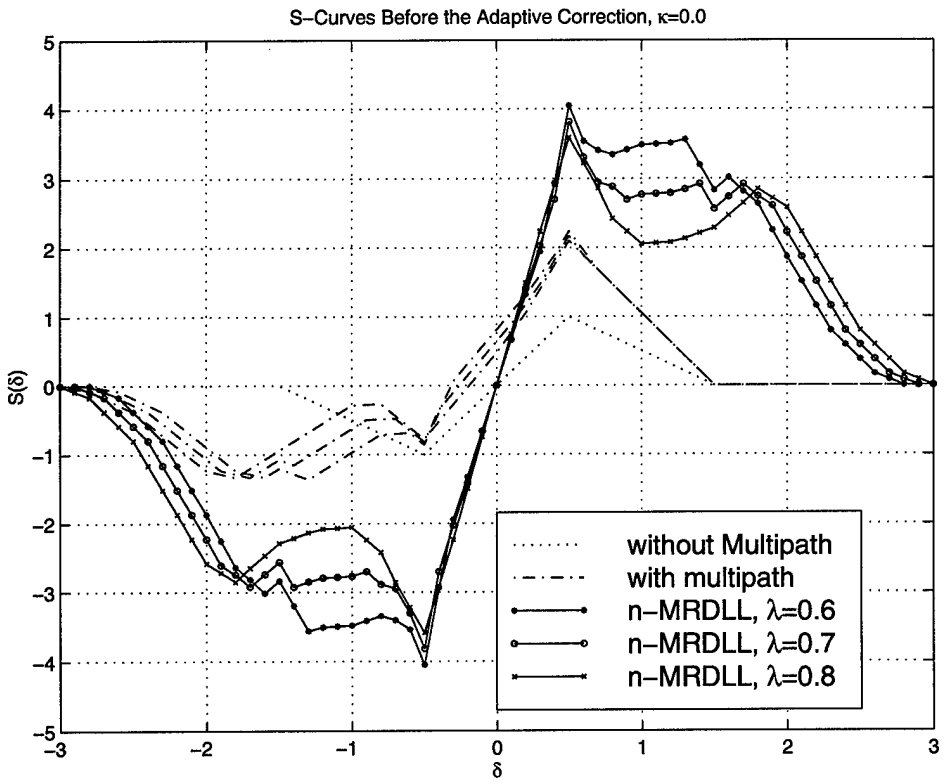


Figure 124 Typical S-curves of the standard DLL and n-MRDLL, $\lambda = 0.6, 0.7, 0.8$, $\kappa = 0.0$

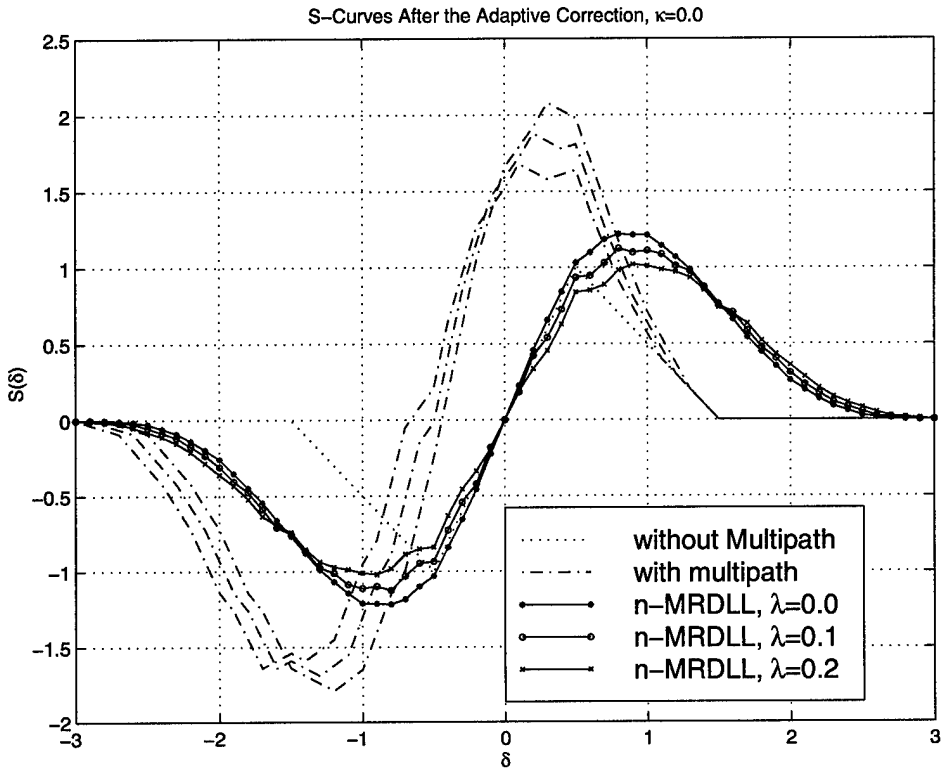


Figure 125 Typical S-curves of the standard DLL and n-MRDLL after adaptive correction, $\lambda = 0.0, 0.1, 0.2$, $\kappa = 0.0$

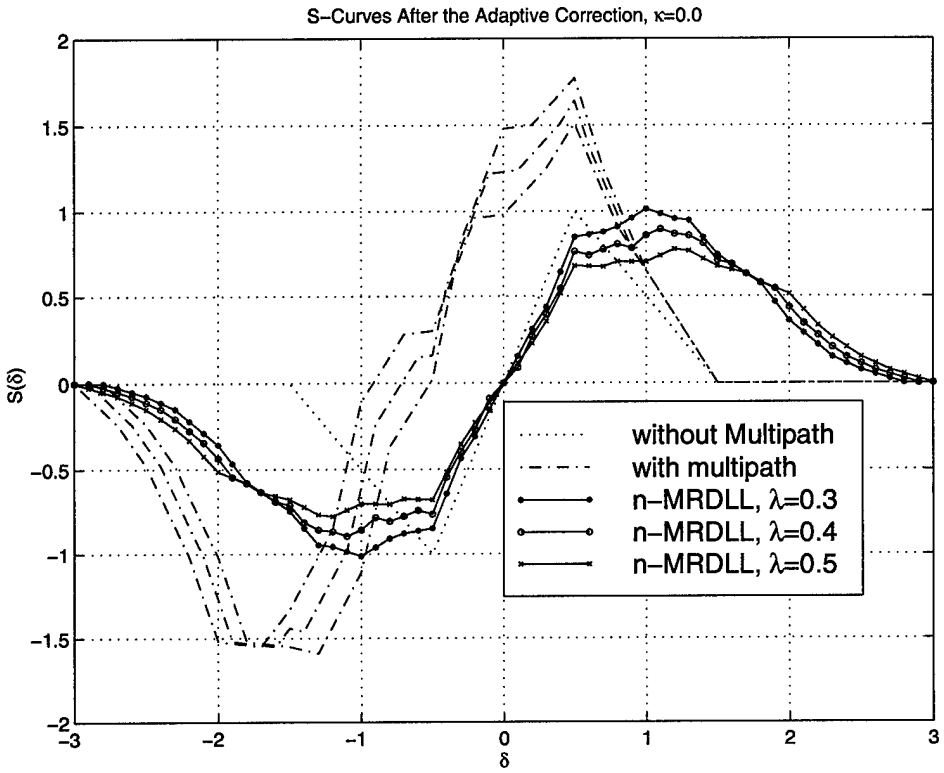


Figure 126 Typical S-curves of the standard DLL and n-MRDLL after adaptive correction, $\lambda = 0.3, 0.4, 0.5$, $\kappa = 0.0$

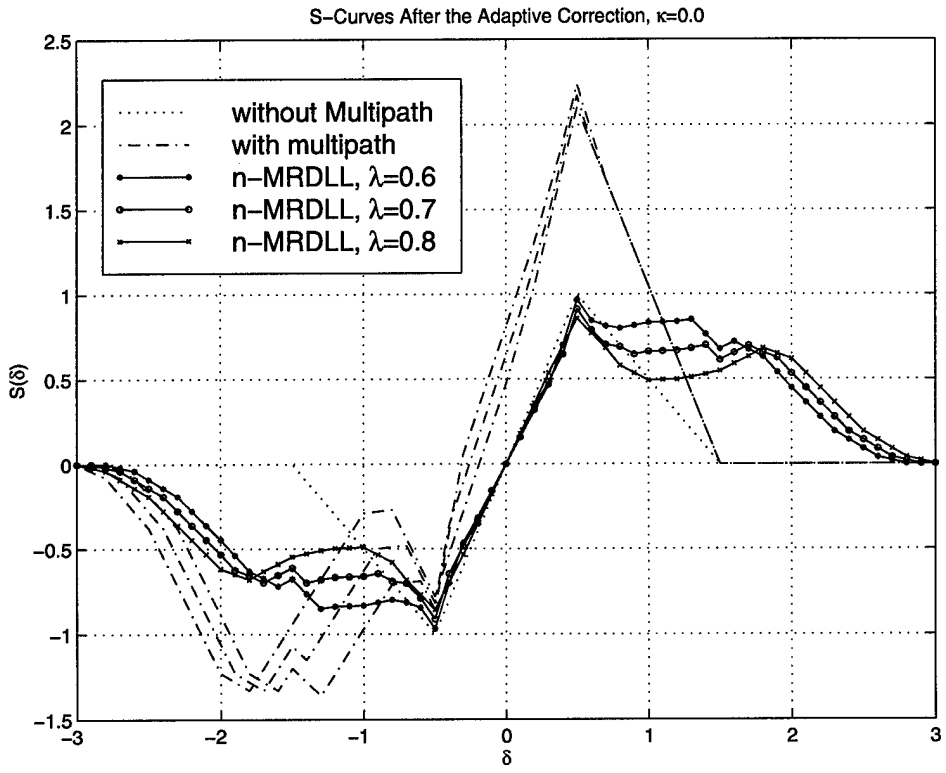


Figure 127 Typical S-curves of the standard DLL and n-MRDLL after adaptive correction, $\lambda = 0.6, 0.7, 0.8$, $\kappa = 0.0$

where F(s) is the transfer function of the loop filter. The second-order transfer function is chosen for the n-MRDLL; the loop natural frequency in rad/sec and the damping factor are given as

$$\omega_n = \sqrt{\frac{Ag_c}{\tau_1}} \tag{155}$$

$$\xi = \frac{\tau_2}{2}\omega_n \tag{156}$$

where τ_1 and τ_2 are the loop filter time constants. The loop filter is defined by [14] as an adaptive loop control and its transfer function is taken as $\frac{F(s)}{A}$; where $F(s)$ is the loop filter transfer function.

The n-MRDLL power spectrum of the tracking jitter is given by [14]

$$G_\delta(f) = |H_{nr}(j2\pi f)|^2 \frac{G_{n_e}(f)}{A^2} \tag{157}$$

where $G_{n_e}(f)$ is defined in Equation (139). Thus, the variance of the tracking jitter can be approximated as

$$\begin{aligned} \sigma_\delta^2 &= \frac{G_{n_e}(0)}{A^2} W_L \\ &= \frac{4\sum_{i=0}^{n} 2N_o x_i^2}{A^2} W_L \end{aligned} \tag{158}$$

The change in the discriminator slope follows changes in the design parameters, ω_n and ξ of the closed loop transfer function. In turn, these changes influence the step response of the closed loop. An n-MRDLL simulation model is done similar to the simulation model of Chapter II except the discriminator characteristic is evaluated for the n-MRDLL. Figure 128 shows on-time simulation step response of n-MRDLL before the adaptive correction when $x_{reduction}$=[1, $\kappa \cdot x_{mp}$] and Figure 129 shows the step responses after the adaptive correction. Figure 130 shows the n-MRDLL step responses before and after the adaptive correction when $x_{reduction} = \kappa \cdot x$. From these plots we observe the following

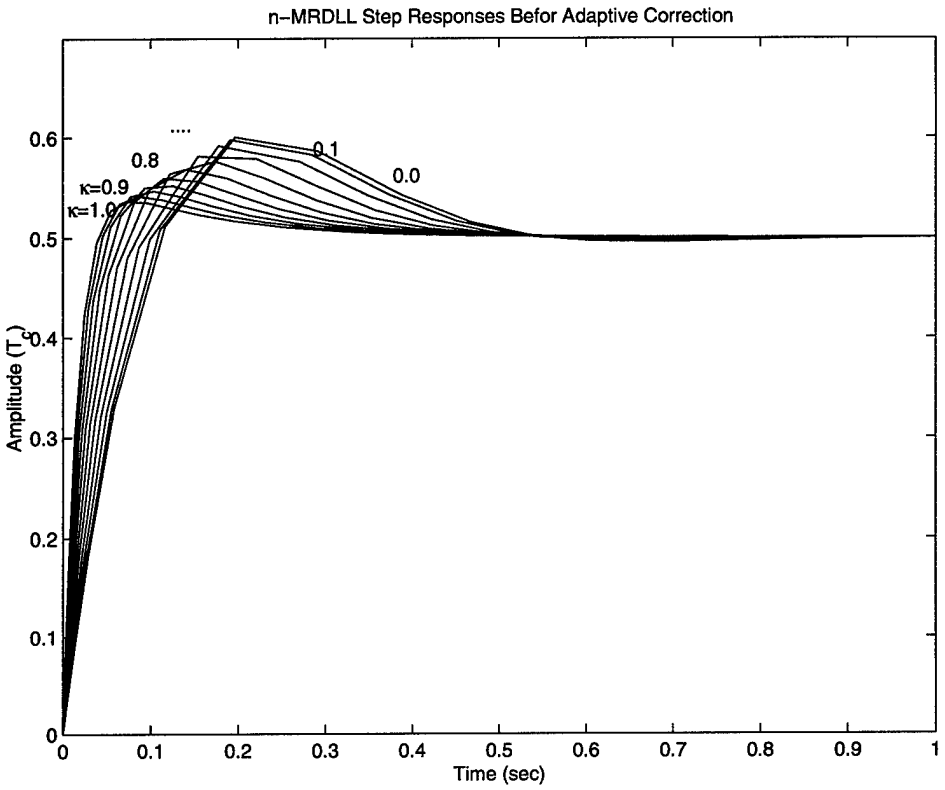


Figure 128 The on-time simulation of the n-MRDLL step response before adaptive gain correction, $\kappa = (0.5 : 1.0)$, $x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

1. In the case of $x_{reduction}$=[1, $\kappa \cdot x_{mp}$] and before the adaptive correction is applied, the n-MRDLL step response is becoming slower as x_{mp} decreases.
2. The opposite of item 1) is happening after the adaptive correction is applied
3. In the case of $x_{reduction} = \kappa \cdot x$ and before the adaptive correction, the step response becomes slower as x is decreased and the natural frequency of the loop is increased and the loop begins to enter the instability mode.
4. In the case of $x_{reduction} = \kappa \cdot x$ and after the adaptive correction, all the curves of the step responses approximately coincide with the step response curve of $\kappa = 0.5$.

The previous results confirm our previous recommendation to use the adaptive correction only when the loop gain is less than the standard loop gain i.e. $A < 2$, and not to use the adaptive gain correction for $A \geq 2$.

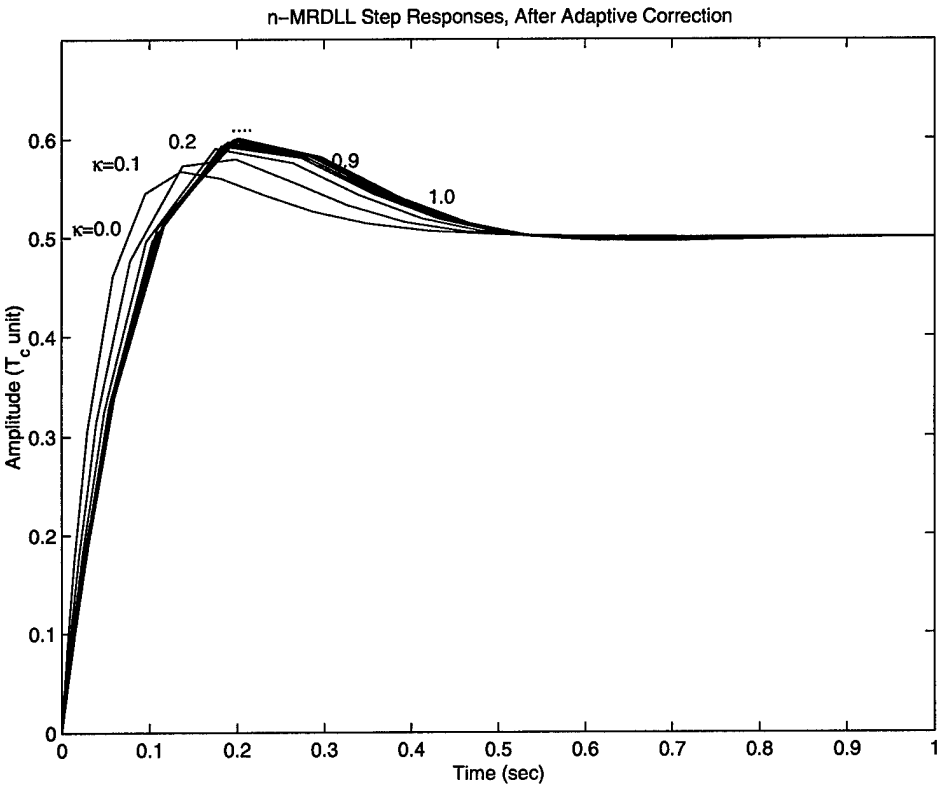


Figure 129 The on-time simulation of the n-MRDLL step response after adaptive gain correction, $\kappa = 0.0, 0.1, ..., 1.0$, $x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

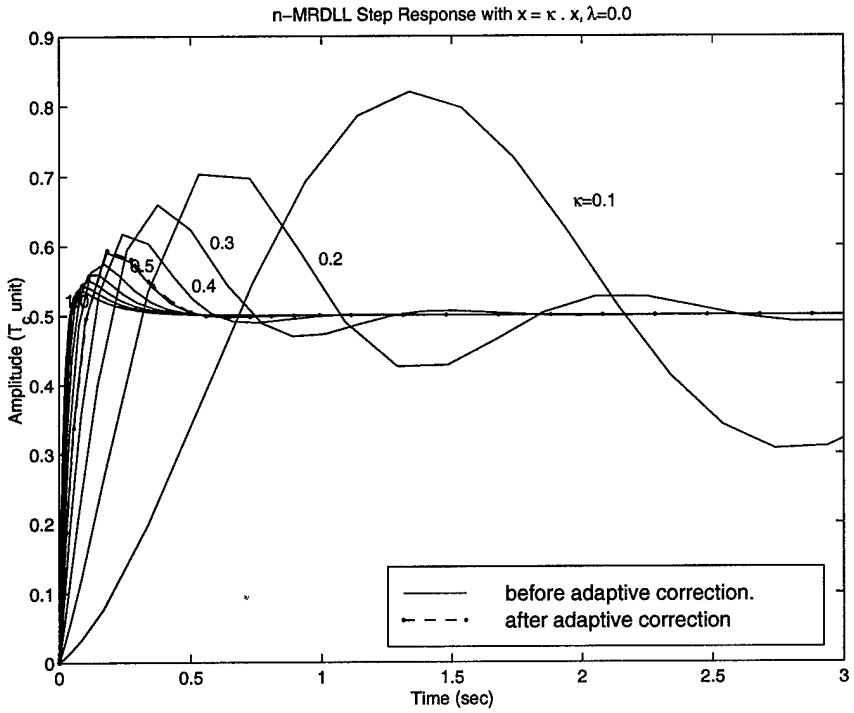


Figure 130 The on-time simulation of the n-MRDLL step response before and after the adaptive gain correction, $\kappa = 0.1, 0.2, ..., 1.0$, $x_{reduction} = \kappa \cdot x$

6.6 n-MRDLL Discriminator Noise Performance

In this section the computer simulation of the n-MRDLL follows the same technique of the standard model as it is described in chapter II except the M-files is altered. The program (nmrdll.m) prepares a six templet of difference C/A codes with maximal length, each one is delayed according to the time delay α. The program also prepares a matrix of C/A-code, its rows represents the shift number of the code which is taken one sample that is corresponding to $0.1T_c$ and the columns represents the code, the number of the columns is corresponding to the maximal code length (1023). This prepared matrix represents the received code signal at the discriminator input. All these data are saved in a Mat-file to be used with the next step of the simulation, which is performed by the program (nrmc.m). The program (nrmc.m) performs 50 runs of the n-MRDLL S-curve corrupted with noise according to the noise level specified by the input SNR. The program achieves the correlation process over the maximal length of the C/A-code and each chip of the code is corrupted with the noise before the start of the correlation process. The resulting n-MRDLL S-curves are evaluated

by using the ensemble average mean and variance. Appendix E includes the program codes of this simulation (nmrdll.m) and (nrmc.m). Appendix E includes also the Figures of the discriminator simulation results, which show the ensemble average mean and the standard deviation $\sigma_{\epsilon_{sim}}$. The ensemble mean and SD are performed for input SNR from 10 to 50 dB-Hz. Figure 131 shows simulation results of the relation between n-MRDLL $\sigma_{\epsilon_{sim}}$ and the input SNR, besides another plots of the theoretical SD of n-MRDLL $\sigma_{\epsilon_{th}}$, the theoretical SD of the standard DLL $\sigma_{\epsilon_{th}}$, standard $\sigma_{\epsilon_{sim}}$ without multipath and standard $\sigma_{\epsilon_{sim}}$. From these plots we observe and conclude the following

- The noise performance of the n-MRDLL discriminator is better than the noise performance of the standard DLL with multipath, However it is lower than the standard especially for low input SNR. The n-MRDLL SD of the noise at the discriminator output is lower than the standard because the number of correlators in n-MRDLL is greater than the number of the standard DLL. Furthermore these correlators are correlated in its noise process as illustrated in Chapter IV.
- The simulation results of the n-MRDLL SD, $\sigma_{\epsilon_{sim}}$ are close to the theoretical SD $\sigma_{\epsilon_{th}}$, but the simulated SD is not close enough to the theoretical SD as in the case of the standard DLL (the n-MRDLL $\sigma_{\epsilon_{th}}$ is calculated by Equation (22) and the standard DLL $\sigma_{\epsilon_{th}}$ is calculated by Equation (139)). The reason for this is the number of sample runs of 50, not 1000, as in the standard case of chapter II.
- All the plots of Figure 131 are improved as the SNR increased.
- Obviously the plot of the standard $\sigma_{\epsilon_{sim}}$ with multipath illustrates that the multipath magnifies the noise at the discriminator output and it can lead the loop to instability in low SNR.

6.7 n-MRDLL Closed Loop Noise Performance

At this point we complete the investigation of the n-MRDLL, where we introduce the tracking and the noise performance of the n-MRDLL closed loop. The simulation of the n-MRDLL model is done. The relation between the RMS tracking jitter and the loop

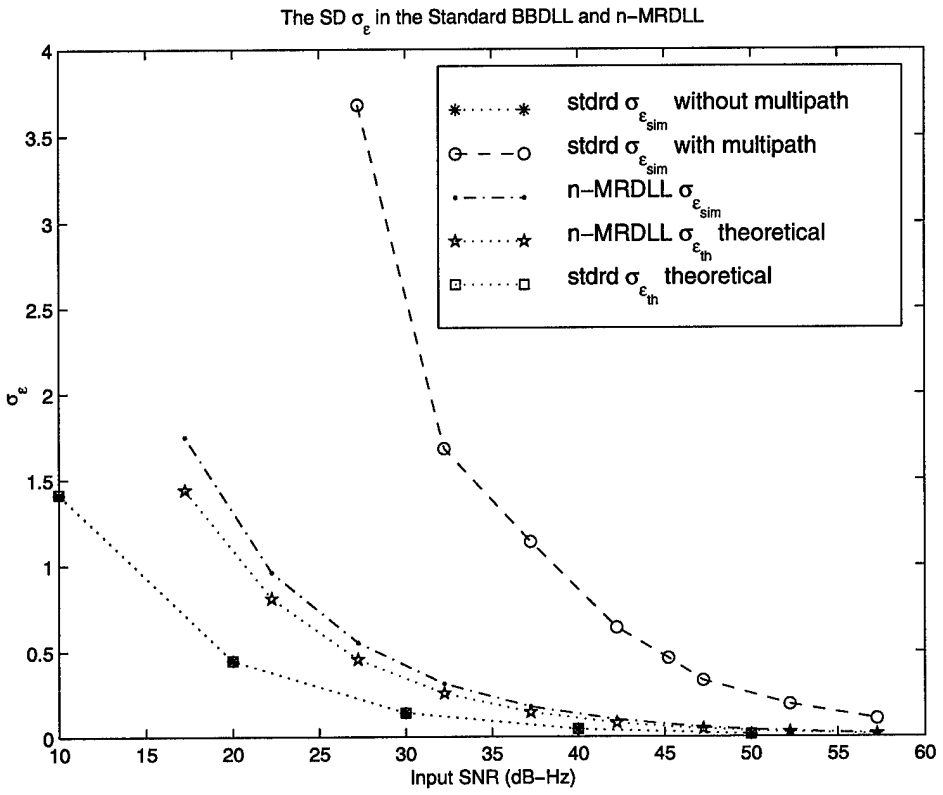


Figure 131 Simulation results shows the n-MRDLL and the standard DLL SD σ_ϵ at the discriminator output

signal-to-noise ratio is presented. The simulation includes the change of the discriminator characteristic as the reduction factor κ is changed for both cases (reduction in the complete x vector or reduction in the multipath components only, x_{mp}). The section is ended with a comparative study between the standard loop and n-MRDLL from the point of view of tracking performance, noise performance, and the recommendation for better usage of both.

6.7.1 n-MRDLL Simulation Model. The n-MRDLL simulation model is similar to the standard DLL which is presented in chapter II. The simulation model is established by using the Simulink as shown in Figure 132. There is an m-file program inside the discriminator block (nrintrp.m) in order to to make on time loading of the values of the n-MRDLL S-curves which is previously calculated by the M-file program (nrmc.m), see section 6.6. Program (nrintrp.m) also provide the linear interpolation of the missed point in the S-curve so, it present a realistic simulation of the S-curve like continuous characteristic. The noise added to the noise free S-curves is performed by the random number block taken from the Simulink library which generates an AWGN like the real one. The statistics of this noise is taken from the evaluated ensemble average mean and variance at the discriminator output which is previously simulated in section 6.6. The loop filter parameters and VCC gain is taken as the values of the standard DLL of chapter II, but the transfer function of the loop filter $F(s)$ is modified to use the adaptive gain in order to correct the S-curves, so the n-MRDLL loop filter or namely, the Adaptive Loop Controller (ALC) is $\frac{F(s)}{A}$.

6.7.2 Simulation Results. Two significant results are included in this section which are the case of the reduction of the multipath strength as previously presented and the reduction including the direct path as well. Figure 133 shows the relation between the RMS tracking error versus the loop signal-to-noise ratio ρ_L as the reduction factor κ is change in the case of $x_{reduction}$=[1, $\kappa \cdot x_{mp}$]. The plots of Figure 133 are performed before the adaptive correction. Figure 134 and 135 shows these plots after the adaptive correction. Figure 136 shows the simulation results of $\sigma_{rms_{sim}}$ versus ρ_L when $x_{reduction} = \kappa \cdot x$. Figure 137 shows an alternative plots of Figure 136 where it shows the RMS error $\sigma_{rms_{sim}}$ versus the reduction factor κ. We observe the following:

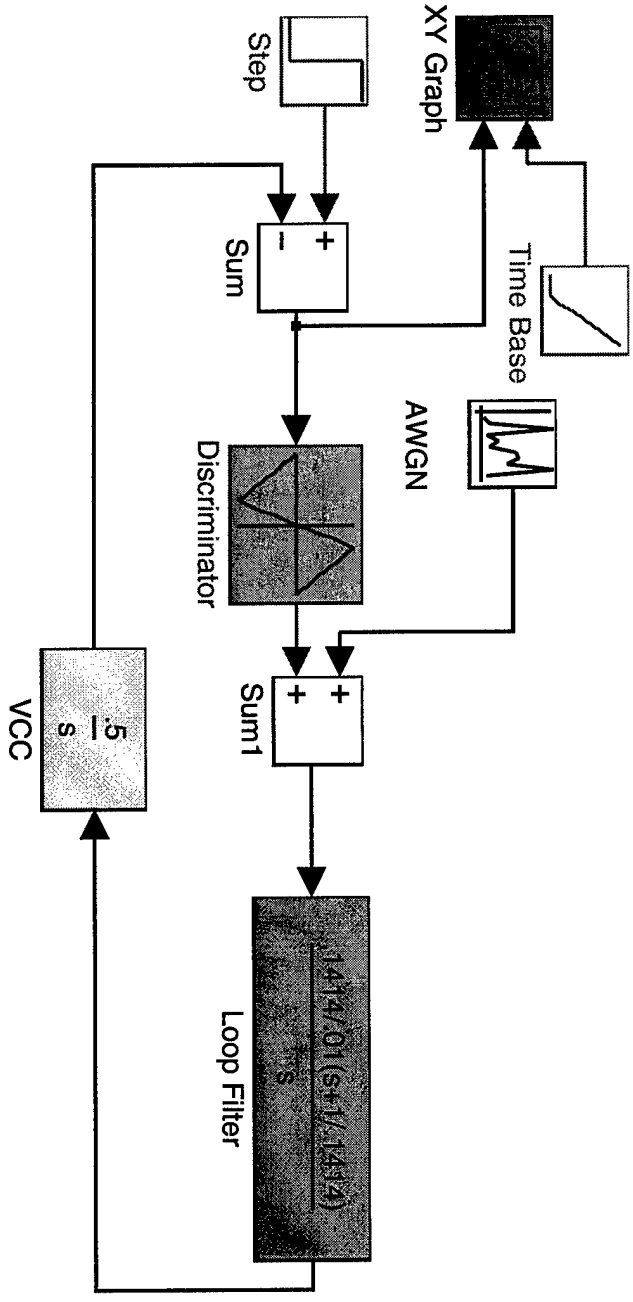


Figure 132 n-MRDLL Simulation Model

- For $x_{reduction}$=[1, $\kappa \cdot x_{mp}$] and before the adaptive correction

    1. The noise performance of the n-MRDLL closed loop is becoming better as the slope of the n-MRDLL S-curve is increased.

    2. The plots corresponding to κ=0.0, 0.1, and 0.2 have lower noise performance than the theoretical parameter σ_δ of the n-MRDLL, however for values of $\kappa > 0.2$ are better than the theoretical plots of the n-MRDLL σ_δ curve .

    3. The plots for $\kappa > 0.7$ are better than the standard theoretical curve embodied in the parameter σ_δ of the standard DLL.

    4. In some cases, the n-MRDLL becomes unstable when the loop signal-to-noise ratio is lower than $\rho_L = 15$ dB-see e.g., the plots for $\kappa < 0.6$.

    5. In general, the noise performance becomes better as ρ_L is increased and the noise performance also becomes better as the multipath strength parameter is increased and even becomes lower than the standard limit embodied in the standard σ_δ curve see the plots of $\kappa < 0.5$. Thus the surprising result is that the n-MRDLL becomes more efficient in high level of noise as the multipath becomes stronger, and the n-MRDLL succeeds to overcome the severe effect of the multipath on the standard DLL due to its magnification of noise effects.

- The case of applying the adaptive correction and $x_{reduction}$=[1, $\kappa \cdot x_{mp}$]:

    1. All the curves are approximately drawn at a close location, except the curve of $\kappa = 0.0$ becomes less than the n-MRDLL theoretical limit of the σ_δ curve.

    2. The plots with $\kappa \leq 0.5$ are better than the plots before adaptive correction while the curves of $\kappa > 0.5$ get lower than the curves before the adaptive correction. This result confirms the same results of the noise performance which were discussed in Section 6.6.

    3. We are still addressing the recommendation concerning the necessity of applying the adaptive gain correction, only for S-curve slopes less than 2; it is not required to use the adaptive correction for the slopes greater than 2, because the closed loop

performance is getting worse than before the adaptive correction. Furthermore, it will change the closed loop parameters, it will provide a slower step response, and it will cause instability in the loop at low signal-to-noise ratio.

- For the case of $x_{reduction} = \kappa \cdot x$

  1. Since the direct path strength is included to be reduced in this case, the curves of κ small have higher values of σ_{rms} than in the previous case.
  2. After the adaptive correction, all the curves are transfered to the same curve, approximately the curve of slope 2, or have $\kappa = 0.5$. This correction is useful only for the curves which have $\kappa < 0.5$ because σ_{rms} becomes lower, while the curves of $\kappa < 0.5$ do not need the correction because σ_{rms} becomes higher.
  3. We also observe that the curves with $\kappa \leq 0.8$ have noise performance better than the theoretical values of the standard DLL and the curves with $\kappa \leq 0.6$ have noise performance better than the n-MRDLL theoretical.

6.8 The Modified PLL

In this section we use the notation "$\hat{}$" for the estimation of the multipath parameters by the estimator unit, the notation"$\wideparen{}$" for the estimated carrier phase by the PLL, the subscript "m" for multipath, and "o" for direct path. As previously mentioned, the PLL aligns a local carrier signal with the received carrier signal regardless of the shift in its original phase caused by multipath, Doppler, thermal noise, or other effects. The modified PLL can correct the equilibrium point of phase tracking by canceling the multipath effect and align the direct carrier signal only. Since the phase detector plays an important role in the dynamic performance of the PLL, an attempt is made to shift its characteristic according to the multipath estimates. The estimation of multipath parameters is performed , then is introduced to the modified PLL. The modified PLL cancels the effect of multipath and accomplishes the tracking of the direct signal only. In order to visualize this idea, Figure 138 shows the simulation block diagram proposed to align the estimated carrier phase with the true direct path incoming carrier phase and hence the multipath effect will be canceled.

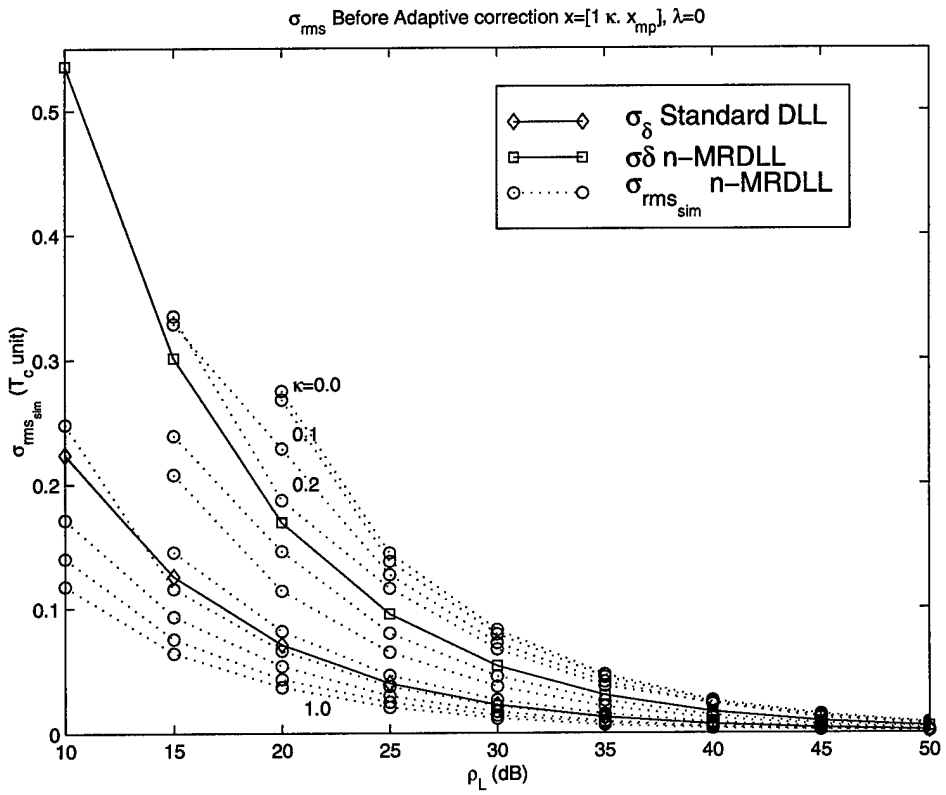


Figure 133 Simulation Results showing the RMS Steady State tracking error σ_ϵ before the adaptive correction,$x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

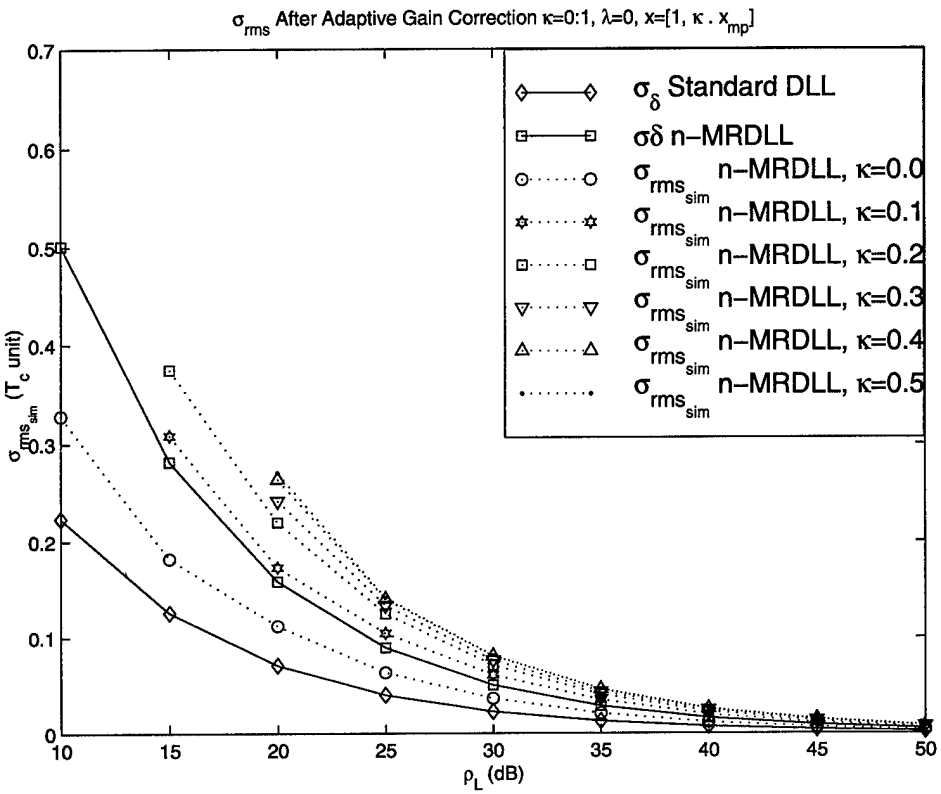


Figure 134 Simulation Results showing the RMS Steady State tracking error σ_ϵ after the adaptive correction,$x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

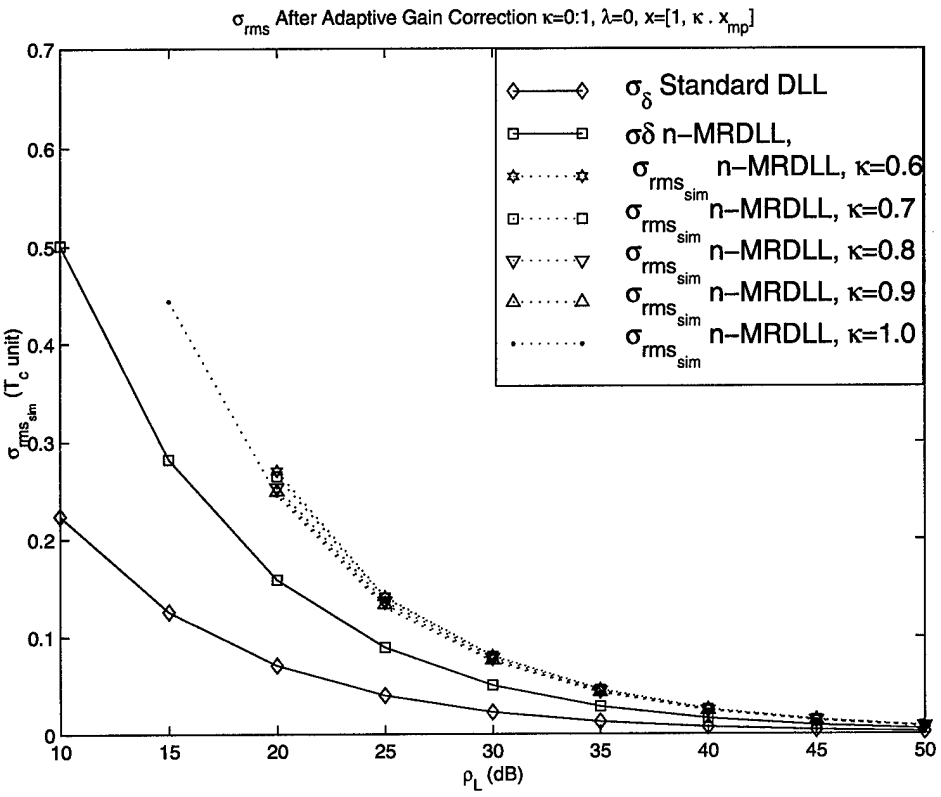


Figure 135 Simulation Results showing the RMS Steady State tracking error σ_ϵ after the adaptive correction,$x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

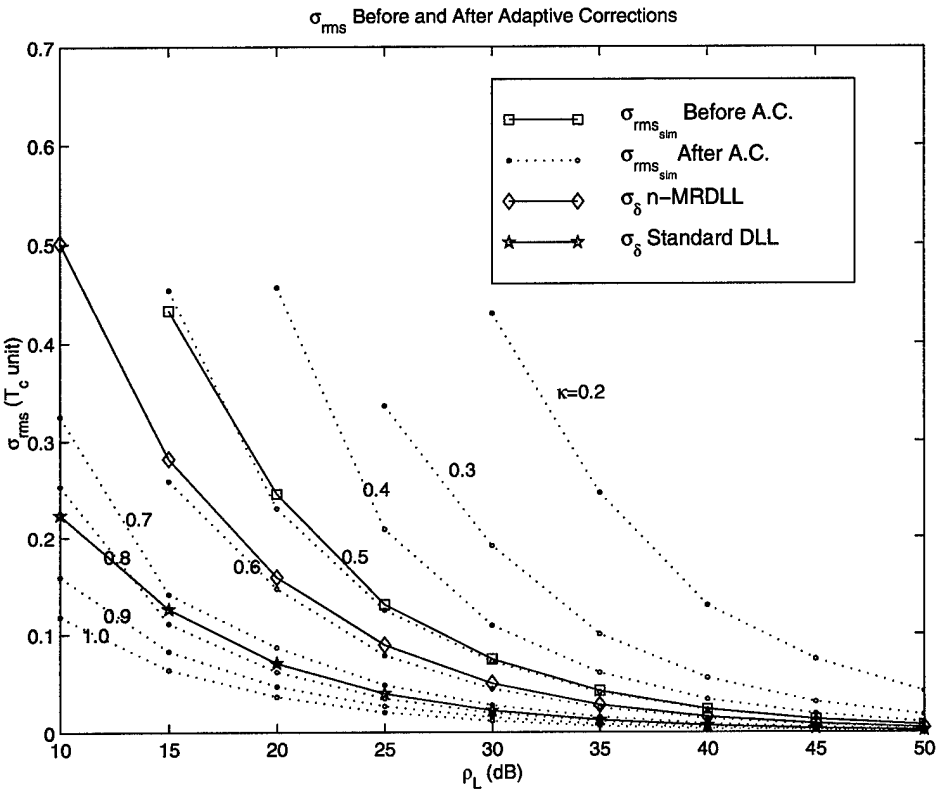


Figure 136 Simulation Results showing the RMS Steady State tracking error σ_ϵ before and after the adaptive correction,$x_{reduction} = \kappa \cdot x$

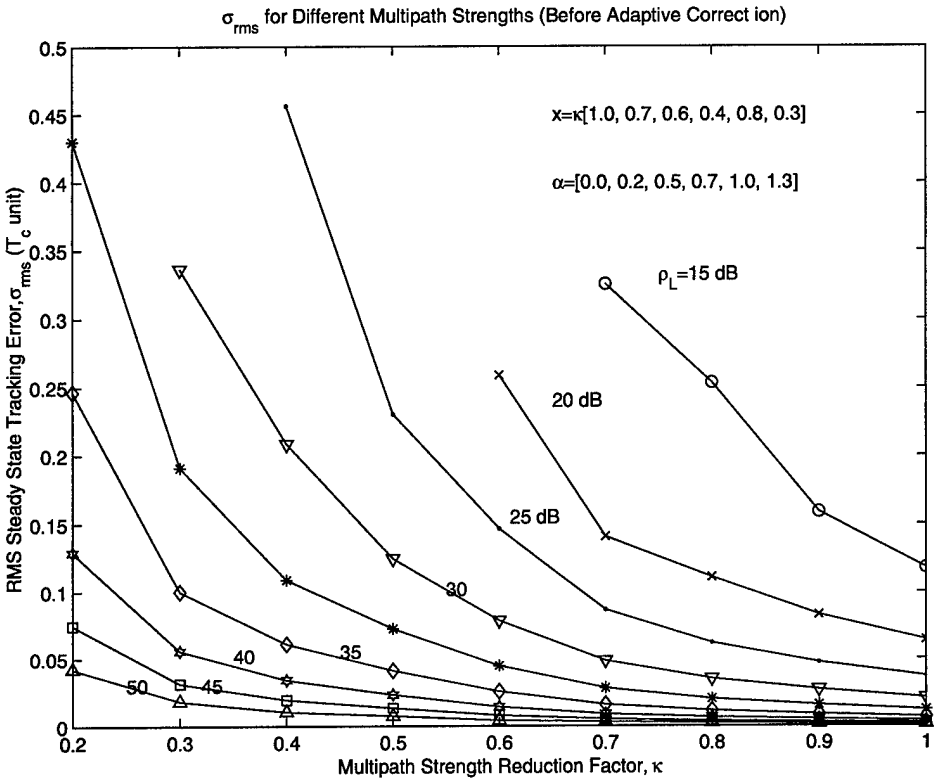


Figure 137 Simulation Results showing the RMS Steady State tracking error σ_ϵ versus the reduction factor κ, $x_{reduction}$=[1, $\kappa \cdot x_{mp}$]

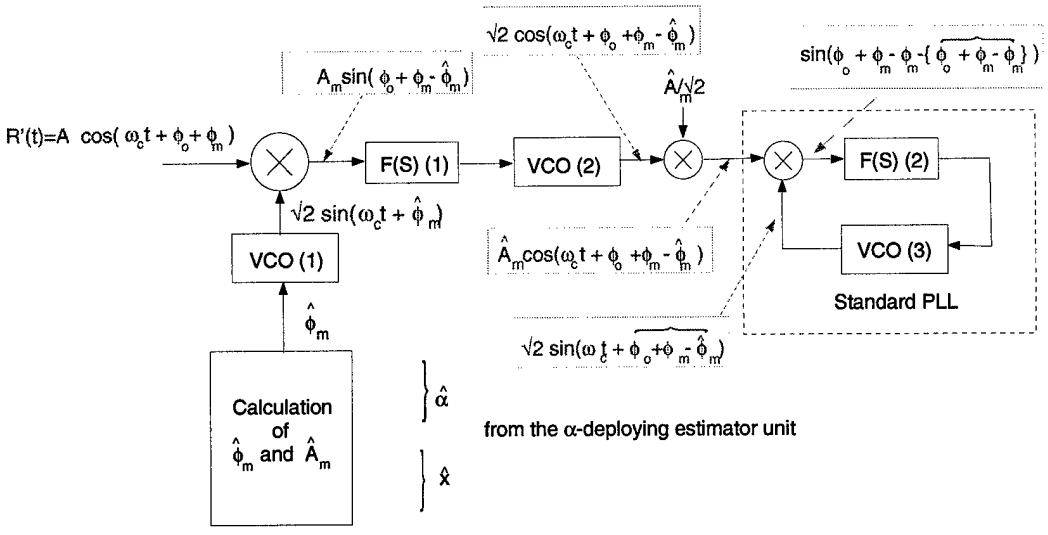


Figure 138 The modified PLL

The steps for processing the incoming signal in this simulation start with generating $\sqrt{2}\sin\left(\omega_c t+\hat{\phi}_m\right)$ from VCO (1), whose input is the calculated phase $\hat{\phi}_m$ of Equation (164). The first multiplier from the left multiplies the incoming signal with this sine wave, yielding $A_m \sin\left(\phi_o+\phi_m-\hat{\phi}_m\right)$. The loop filter F(s) (1) and VCO (2) generate a cosine signal with phase equivalent to the argument of the sine wave at the input of loop filter (1). The next step is to multiply the cosine signal by the estimated multipath amplitude, $\hat{A}_m/\sqrt{2}$. The actual signal into the PLL becomes

$$\hat{A}_m \cos\left(\omega_c t+\phi_o+\phi_m-\hat{\phi}_m\right) \tag{159}$$

and the local carrier generated would be

$$\sqrt{2}\sin\left(\omega_c t+\overbrace{\phi_o+\phi_m-\hat{\phi}_m}\right) \tag{160}$$

where $\overbrace{\phi_o+\phi_m-\hat{\phi}_m}$ is the phase estimated by the PLL. Therefore, the tracking curve (S-curve) of the PLL can be written from Equation (159) as

$$\sin\left(\phi_o+\phi_m-\hat{\phi}_m-\overbrace{(\phi_o+\phi_m-\hat{\phi}_m)}\right) \tag{161}$$

The argument of Equation (161) can also be written as

$\phi_o-\overbrace{\phi_o}+\phi_m-\overbrace{\phi_m}-(\hat{\phi}_m-\overbrace{\hat{\phi}_m})$.

If steady state tracking of the PLL is reached and an exact estimate of ϕ_m is available, the complete alignment of the direct path signal with the local carrier signal is achievable.

6.9 Multipath Carrier Phase Error Estimation Using n-MRDLL

The idea here is to exploit the estimated multipath parameters α_i and $\hat{x}_i$ which are estimated by the α-deploying method, then compute $\hat{\phi}_m$ in terms of these parameters. An attempt to reject ϕ_m from the processing of the modified PLL is introduced in the previous section. Assuming that the code phase tracking error is negligible (i.e. $\hat{\tau}_o \approx \tau o$), substitute

the estimates $\hat{\alpha}_i$ and $\hat{x}_i$ into the Equation (59); the estimated $\hat{\phi}_m$, is given by

$$\hat{\phi}_m = \arctan\left(\frac{\sum_{i=0}^{n} \hat{a}_i R_c\left(\hat{\alpha}_i T_c\right) \sin\left(\hat{\phi}_i - \phi_o\right)}{\sum_{i=0}^{n} \hat{a}_i R_c\left(\hat{\alpha}_i T_c\right) \cos\left(\hat{\phi}_i - \phi_o\right)}\right) \tag{162}$$

The correlation function of Equation (8) is an even function, i.e. $R_c(\hat{\alpha}_i T_c) = R_c(-\hat{\alpha}_i T_c)$. Also we observe that $\hat{\phi}_i - \phi_o = 2\pi f_c \hat{\alpha}_i T_c$, so from Equation (48) we have

$$\hat{a}_i = \frac{\hat{x}_i}{A_o \cos\left(2\pi f_c \hat{\alpha}_i T_c\right)} \tag{163}$$

thus,

$$\hat{\phi}_m = \arctan\left(\frac{\sum_{i=0}^{n} \hat{x}_i R_c\left(\hat{\alpha}_i T_c\right) \tan\left(2\pi f_c \hat{\alpha}_i T_c\right)}{\sum_{i=0}^{n} \hat{x}_i R_c\left(\hat{\alpha}_i T_c\right)}\right) \tag{164}$$

Similarly, the estimated multipath amplitude, $\hat{A}_m$, is given by

$$\hat{A}_m = \sqrt{\left(\sum_{i=0}^{n} \hat{x}_i R_c(\hat{\alpha}_i T_c) \cos\left(2\pi f_c \hat{\alpha}_i T_c\right)\right)^2 + \left(\sum_{i=0}^{n} \hat{x}_i R_c(\hat{\alpha}_i T_c) \sin\left(2\pi f_c \hat{\alpha}_i T_c\right)\right)^2} \tag{165}$$

The simulation of the modified PLL is done By using Simulink© and we verified accurate tracking and alignment of the direct path signal combined with the multipath components.

6.10 Summary

1. The n-MRDLL noise performance improves as the slope of the discriminator characteristic is increased.
2. The slope of the discriminator characteristic namely the adaptive loop gain, is a function of the multipath strength's parameters, x, which includes also the direct path strength.
3. The α-deploying estimator is useful for pre-calculating the slope of the discriminator characteristic.

4. The standard DLL and n-MRDLL becomes unstable in low signal-to-noise ratios, (simulation results show instability for input SNR < 15 dB-Hz). However the n-MRDLL can work in low SNR if it is corrected by the adaptive loop gain. In addition the steeper the discriminator's slope, the faster the step response of the closed tracking loop is.

5. The adaptive loop gain correction is necessary for discriminator characteristic slope less than 2 and not necessary for slope greater than 2.

6. The n-MRDLL noise performance is better than the theoretical standard noise performance if its discriminator characteristic slope is greater than 3.1832 and becomes better than the theoretical n-MRDLL noise performance if it is greater than 2.3168.

7. The slope of the discriminator becomes better when the multipath strength becomes stronger. So, the stronger the multipath in the n-MRDLL is, the better the tracking and noise performance improvement is.

8. The stronger multipath magnifies the noise effects in the standard DLL; in addition, it causes the closed loop to become unstable for low SNR.

9. The modified PLL gives a reference carrier phase aligned with the direct path carrier only in the presence of multipath. The modified PLL is useful in GPS for carrier phase observble.

VII. Conclusion and Recommendations

7.1 Overview

This dissertation contributed a useful new method for solving the GPS multipath problem. The proposed algorithm can detect the presence of the multipath, can determine the unknown the number of multipath components and identify the required information about multipath parameters (time delay,α, and attenuation coefficients, a, (multipath component's strengths) in the GPS receivers. Furthermore the multipath signal's parameters can be estimated at any instant of observation, which implies that the Doppler shift is implicitly incorporated in the estimates of multipath. The proposed method can be considered as an estimator based on a recursive deploying process of the multipath time delay, α, so we called it an"α-deploying method". Because of the instantaneous high probability of observing the multipath parameters, the α deploying method can be proposed to track the dynamic multipath signal caused by environmental changes (satellites and receiver motion). The method also can be suggested as a tool to scope the multipath in the environmental area around the user, and may be a significant indicator in surveying and determining the region of high multipath. Another theoretical proposal was investigated for the estimation of the multipath called "search method". The search method can work when a restricted number of multipath signals. The search method was considered as a regular search of the estimated multipath parameters that minimizes the quadratic form of the likelihood function see Equation (85). The GPS multipath problem was analytically investigated in cases of noise free and in the presence of noise. Another suggestion to make a separation between the multipath and the AWGN was performed by applying a Kalman filter. Once the Kalman filter was completed a validation of the results and a performance analysis were executed by a Monte Carlo simulation. In addition an improvement was verified in low SNR by using Kalman filter. Prior to introducing these methods, an investigation and modelling of the GPS signal were introduced in case of code tracking and carrier phase tracking. Finally the multipath mitigation proposal was demonstrated as a three correlated parts: the multiple observations unit, the estimator unit and the modified tracking loop unit. The estimator analysis and investigation

were performed for the α-deploying method and the search method, in the code tracking. The modified tracking loops unit were designed for both the carrier tracking (PLL) and the code tracking (DLL). This chapter provides a summary of the thesis work and it also includes an important proposals to how to modify the DLL and PLL to work with those algorithms. This chapter includes also the useful recommendation for the future research work.

7.2 Search Method Summary

The search method can be summarized as follow:

- The accuracy of search method depends on the step of search (i.e. search resolution).
- The interval of running the search method becomes longer as long as the search step is shortened.
- A singularity can be established during the search because of the relative close between α and β vector.

7.3 α-Deploying Method Results

- In the cases of noise free or high SNR, the accuracy of α- deploying method can be exact as long as the redeploying of α is smaller.
- In the presence of noise the accuracy of α-deploying method has two trade-off parameters: the bandwidth of the LPF in the bank and the SNR. The increasing of the input SNR gives the chance to widen the LPF bandwidth, subsequently, the time response of the signal observation becomes faster.
- The simulation results show that the α deploying method is applicable for noise bandwidth $BW_{noise} \leq$ 100, 10, 1 and 0.1 Hz, corresponding to input $SNR \geq$ 50, 40, 30, and 20 dB Hz, respectively, which is the significant accuracy of the standard C/A code tracking loop.
- No noise reduction is achieved during the estimate process by the MLE. Consequently, the α-deploying estimator can be considered as a linear transformation of the corrupting noise at the estimator output to the incoming noise at the correlator's bank input.

- The increasing of the number of measurements in the bank improves the multipath estimates but does not improve the noise performance.
- The results also showed that in the range of estimation for code tracking [0, T_c], the noise corrupting the estimated multipath is always small relative to to the range [T_c, $1.5T_c$] (this point is left for future work).

7.4 Rules Discovered to the α-Deploying Method

Four rules are established to achieve high accuracy with the α-deploying algorithm.

- The first α deployment must be uniformly distributed in the time delay interval of $0 \leq \alpha \leq 1.5T_C$ in order to discover any multipath components in this interval.
- The order of the deployed α determines the order of multipath components.
- If a nonzero estimated multipath component lies between two zero estimated multipath components, then the nonzero answer yields a multipath component.
- If the estimated nonzero multipath component is preceded, and/or followed by a nonzero component, then the number of multipath components is equal to these estimated components or less; and the exact number can be reached if the redeploying of both the α and β vectors is repeated around these nonzero components by cyclings through $0.1T_c$, $0.01T_c$, $0.001T_c$, ... etc, until the number of estimated components becomes constant. Then the number of estimated multipath components is the correct answer.

7.5 Kalman Filtering Application Results

- The application of the Kalman filter as a cascaded estimator to the α-deploying estimator unit reveals the ability to achieve an accurate estimates of multipath immersed in the noise. So the Kalman filter can be considered as an important tool for separating the multipath from noise.
- The Monte Carlo simulation validates the performance analysis of the designed Kalman filter.

- The Kalman filter can improve the reduction of the AWGN during its run interval of 1 second for noise bandwidth $BW_{noise} = 1$ kHz and 10 kHz with input $SNR = 40$ and 50 dB Hz, respectively.
- As long as the interval of Kalman filter run is longer, the improvement of noise reduction is increased, however we have to limit the dynamics of the multipath in the area.
- The relative comparison between the sample period of the Kalman filter versus the time constant of the LPF in the bank influences the estimated multipath-to-noise ratio, which is considered as a helpful tool to detect the multipath parameters in low SNR.

7.6 BBDLL Results

1. The BBDLL simulation model using Simulink© is close to the theoretical model in the deterministic case and with higher input SNR.
2. The BBDLL simulation shows that the multipath not only introduces a bias error on the tracking loop but also amplifies the noise. Thus an extra bias is induced due to the noise.
3. The noise performance in the presence of multipath becomes worse with the decrease in the SNR.

7.7 Modified DLL and PLL Results

7.7.1 Modified DLL (n-MRDLL).

1. The n-MRDLL noise performance is improved as the slope of the discriminator characteristic is increased.
2. The slope of the discriminator characteristic or namely the adaptive loop gain is a function of the multipath's strength parameters, x which also includes the direct path strength.
3. α-deploying method is useful for pre-calculating the slope of the discriminator characteristic.

4. The standard DLL and n-MRDLL becomes unstable in low signal-to-noise ratio, (simulation results show that instability for input SNR < 15 dB-Hz). However the n-MRDLL can work in low SNR if it is corrected by the adaptive loop gain. In addition the larger magnitude of the discriminator slope, the faster of the step response of the closed tracking loop.
5. The adaptive loop gain correction is necessary for discriminator characteristic slope less than 2 and not necessary for slope greater than 2.
6. The slope of the discriminator becomes better when the multipath strength increases. So, the stronger multipath in the n-MRDLL is the tracking and noise performance improvement.
7. The stronger multipath magnifies the noise and additionally causes the closed loop to become unstable for low SNR.

7.7.2 Modified PLL. The modified PLL gives a reference carrier phase aligned with the direct path carrier only in the presence of multipath. The modified PLL is useful in GPS for carrier phase observable.

7.8 Summary and Recommendations

1. search method is useful only for few number of multipath reflectors.
2. When the singularity occures in the search method , the accuracy of the estimates of the multipath parameters degrades.
3. The singularity can be removed by reseting the correlators spacings in the bank.
4. Increasing the number of measurements (the number of correlators in the bank) improves the accuracy of multipath estimates but does not improve the noise performance in the estimator unit.
5. The α-deploying estimator unit performes a linear transformation upon the corrupting noise.

6. Kalman filter is applied to reduce the noise corruption of the estimates of multipath parameters.
7. α-deploying method is followed by a Kalman filter is considered as a useful configuration to separate the estimated multipath parameters from the corrupting noise.
8. The α-deploying method can be used as a tool to scope the multipath in the environmental area around the user and can significantly aid the determination of the regions of high level of multipath.
9. The modified PLL for correcting the false tracking of the carrier signal, by the standard PLL, due to multipath.
10. The n-MRDLL modifyies the standard DLL to enable tracking of the direct path signal code corrupted by the presence of multipath
11. The use of an adaptive loop gain for the n-MRDLL is only recommended when $A < 2$ (it is not necessary for $A \geq 2$).
12. we recommend the use of an ALG for the standard DLL to improve the noise performance and the stability of the closed tracking loop when the direct path is weak (the discriminator slope is less than 2).
13. The stronger multipath improves the tracking and noise performance of the n-MRDLL whereas it degrades the tracking and noise performance in the standard DLL.

Appendix A. BBDLL Simulation

A.1 The shift.m M-file

```
%This program function has two argument input the first is the C/A code
%the second is the value of shift. the program returns the same code
%with the same length but with this shift
%
%Written by EL-Sayed A. Gadallah 13 Oct 97. Last update: 1 June 1998.
function [c_shift]=shift(code,rkm)
%(c_shift) is the shifted code, code is the original code,
%(rkm) is the value of shift.
shift=rkm;
for i=1:length(code)-shift
   c_shift(shift+i)=code(i);
end

for i=1:shift
   c_shift(shift-i+1)=code(length(code)-i+1);
end
```

A.2 The sctmplt.m M-file

```
% This code returns the S-curve in case of noise free also it returns
% a matrix with rows representing the shift and the columns representing
%the shifted code to be used as a templet of the shifted code
%(from -1.5T_c to 1.5T_c). The matrix size is 35x1023.
clear
PRN=cacode(1);%C/A-code generator GPS-Toolbox function
%Extension process of the C/A-code such that 10 sample per each chip
for i=1:length(PRN)
```

```
   c(10*i-9:10*i)=PRN(i)*ones(1,10);
end

c_early=shift(c,25);%fixed early replica setted at sampling point #25
c_late=shift(c,15); %late replica setted at sampling point #15 so
    %the zero is setted at #20
for shift=1:35
r(shift,1:length(c))=shift(c,shift);%representing the shifted received code
  y1=mean(r(shift,1:length(c)).*c_early);%correlation process taken as
%the mean of the results of the two code product
  y2=mean(r(shift,1:length(c)).*c_late);
  epsilon(shift)=y1-y2;%tracking error at the discriminator output
end
```

A.3 The scmcnos.m M-file

```
%This code returns the S-curve in the presence of the AWGN for 1000
%sample runs.

clear
load scfr.mat% Tamplet shifted codes of S-curve

CNR=10;% Input signal-to-noise ratio (dB-Hz)
P=2;%power P=2 Watt of the incoming PRN code waveform at
%the discriminator input

BW=.1023*10^6;%Bandwidth of the received signal at the discriminator
              % considered as 1.023MHz/sampling rate
No=P/(10^(CNR/10));%AWGN PSD
sgma=sqrt(BW*No);  %Noise SD
```

```
for mc=1:1000 %# of sample runs=1000
for shift=1:35% performing code shift
noft=sgma*randn(1,length(c_early));%noise signal

g=r(shift,1:length(c))+(noft/(sqrt(2)));%called from workspace
  y1=mean(g.*c_early);
  y2=mean(g.*c_late);
  epsilon(mc,shift)=y1-y2;%Tracking error per each shift
end
end
```

A.4 The lintrp.m M-file

```
%This code is used in the SIMULINK interface to perform the discriminator
%characteristic even in the presence of noise. This m.file is setted inside
% the MATLAB function block taken from the nonlinear SIMULINK library
%the input of this code is a real number representing the time difference
%between the received code and the local replicas and returns the
%discriminator characteristic, tracking error (the S-curve), the
%program also able to output the the tracking error even if the input
%difference in time is in-between the sampling points, using the linear
%interpolation.
%

function sc=lintrp(dif)

load lint.mat

sc=linterp(x,y,dif);
```

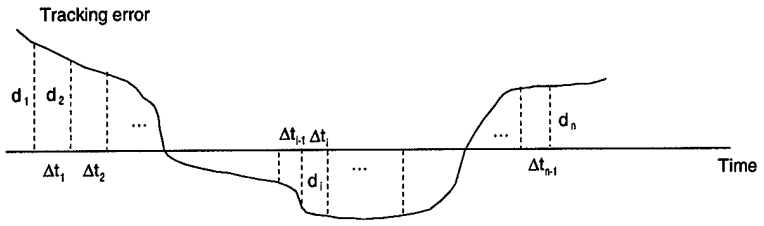


Figure 139 The steady state tracking error in the presence of noise

A.5 The Tracking Jitter RMS Formula

As it seen from Figure 139 is swing around the input step value at the steady state and after squaring it becomes positive random signal. The actual simulated tracking jitter squares is a digitizing data denoted, d_i^2 and separated with a variable samples Δ_i as explained from the Simulink behavior in chapter II. By the definition of the root mean-square, the RMS is the square root of the normalized value of the area under the curve. Thus, the area under the curve in the tracking jitter signal can be calculated as in.

$$\sigma_{rms} = \sqrt{\frac{1}{2}(d_1^2+d_2^2)\Delta t_1+\frac{1}{2}(d_2^2+d_3^2)\Delta t_2+\cdots+\frac{1}{2}(d_{n-1}^2+d_n^2)\Delta t_{n-1}} \tag{166}$$

then,

$$\sigma_{rms} = \sqrt{\frac{1}{T_{sum}}\left(\frac{1}{2}[d_1^2\Delta t_1+\sum_{i=2}^{n-1} d_i^2(\Delta t_{i-1}+\Delta t_i)+d_n^2\Delta t_{n-1}]\right)} \tag{167}$$

A.5.1 (rms.m) M-file.

```
%This program returns the rms tracking error, after loading
%the workspace with Mat-file created during the on-time simulation
%of the BBDLL model in Simulink
%Written By El-Sayed A. Gadallah, last update 25 June 1998

clear
load prma1.mat%the created mat file from the BBDLL step response simulation
x=prm(1,:);
y=prm(2,:);%prm is a define vector in loaded mat file representing
```

```
          %the step response of the closed loop
indx1=find(x<=2);%To choose the  start time of steady state tracking error
id1=max(indx1);
id2=length(x);

v1=x(id1-1:id2-1);
v2=x(id1:id2);
Dt=v2-v1;%To determine the intervals between the sample points
te=y(id1-1:id2);% the tracking error vector
sgta=std(te)%the SD of the tracking error
te2=te.^2;% the tracking error squares
te_rms=sqrt(mean(te2))%the sqrt of the mean of T.E. squares
      %it can also represent the RMS T.E.
n=length(te);
mg=Dt(1:n-2)+Dt(2:n-1);
%RMS tracking error formula
rm=sqrt((.5/(v2(n-1)-v1(1)))*(te2(1)*Dt(1)+sum(te2(2:n-1).*mg)+te2(n)*Dt(n-1)))
```

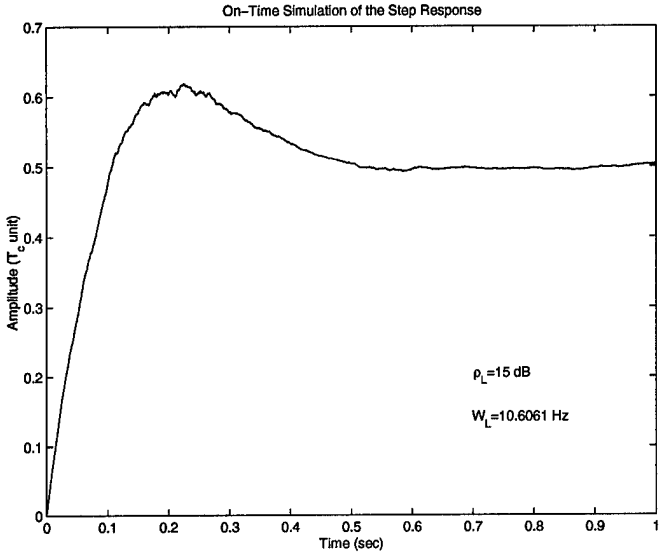


Figure 140 Simulation showing the Step Response of the closed Loop of the BBDLL with, loop Bandwidth $W_L = 10.6061$ Hz, and closed loop signal-to-noise ratio $\rho_L = 15dB$

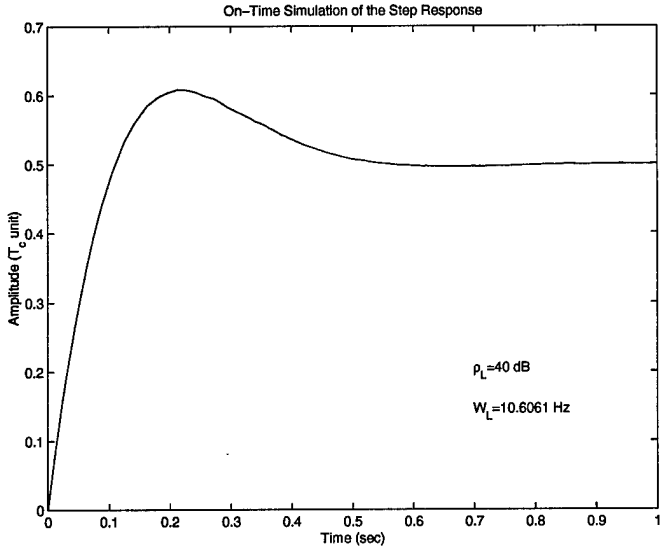


Figure 141 Simulation showing the Step Response of the closed Loop of the BBDLL with, loop Bandwidth $W_L = 10.6061$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

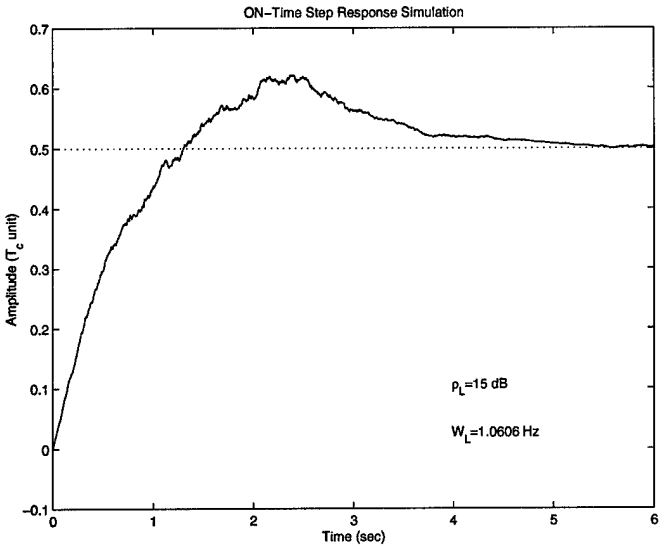


Figure 142 Simulation showing the step response of the closed loop of the BBDLL with, loop bandwidth $W_L = 1.0606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 15dB$

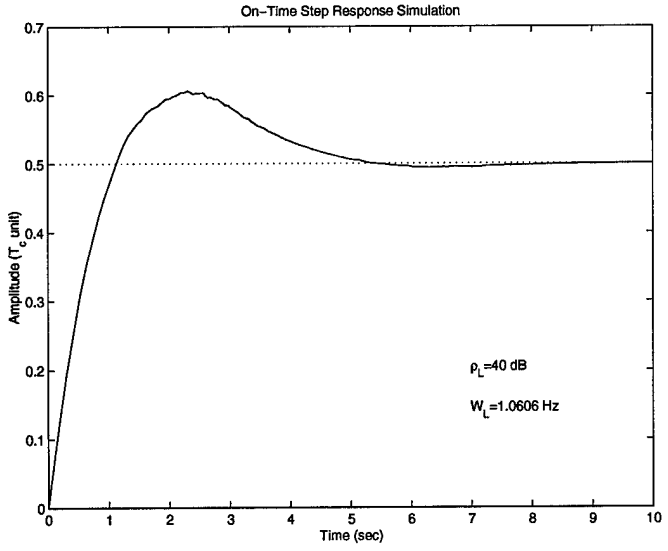


Figure 143 Simulation showing the step response of the closed loop of the BBDLL with, loop Bandwidth $W_L = 1.0606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

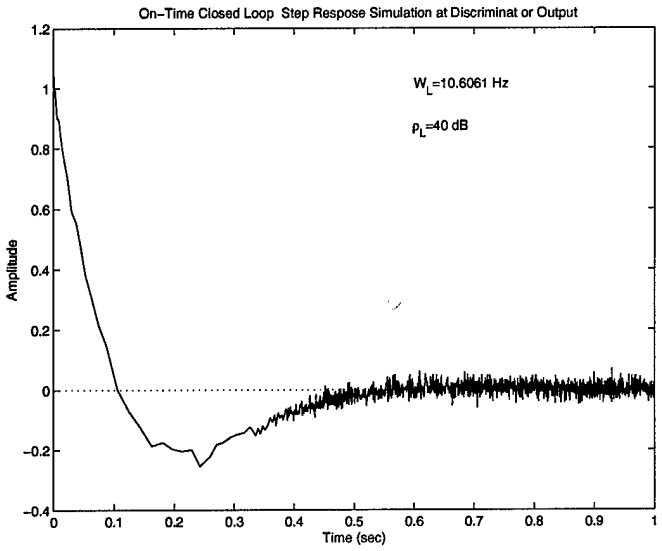


Figure 144 Simulation showing the step Response of the BBDLL closed loop at the discriminator output with, loop bandwidth $W_L = 10.606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

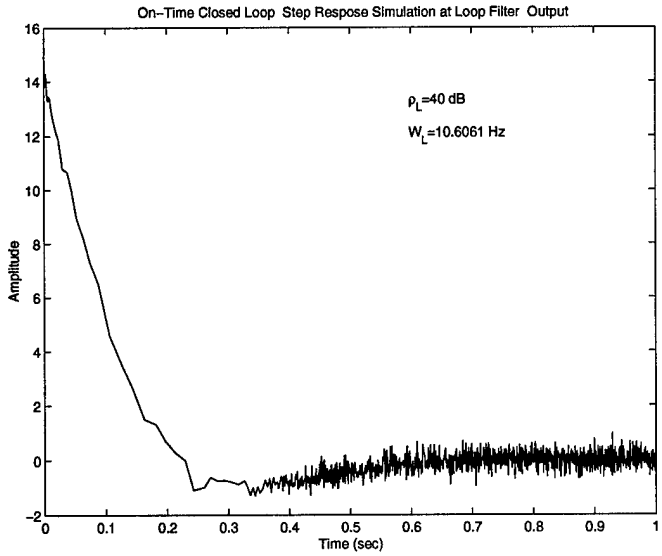


Figure 145 Simulation showing the step Response of the BBDLL closed loop at the loop filter output with, loop bandwidth $W_L = 10.606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

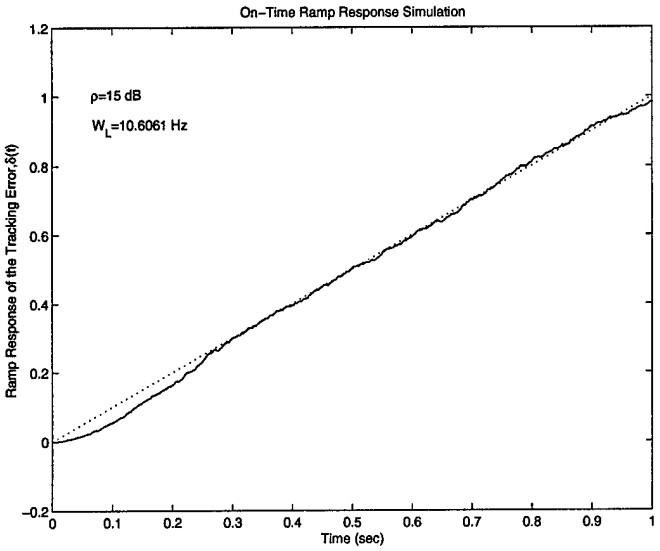


Figure 146 Simulation showing the ramp Response of the closed loop of the BBDLL with, loop bandwidth $W_L = 10.606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

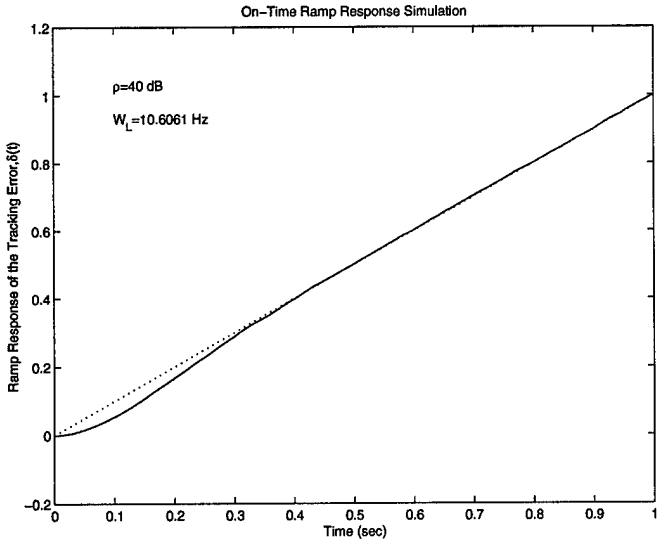


Figure 147 Simulation showing the ramp Response of the closed loop of the BBDLL with, loop bandwidth $W_L = 10.606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 15dB$

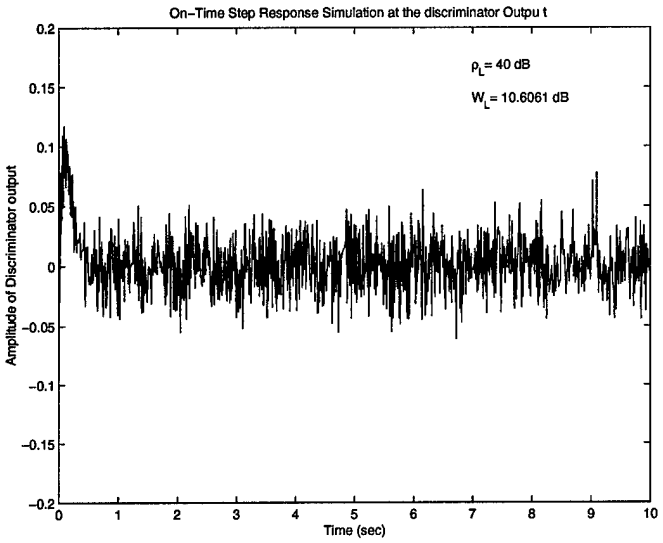


Figure 148 Simulation showing the ramp Response of the BBDLL closed loop at the discriminator output with, loop bandwidth $W_L = 10.606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

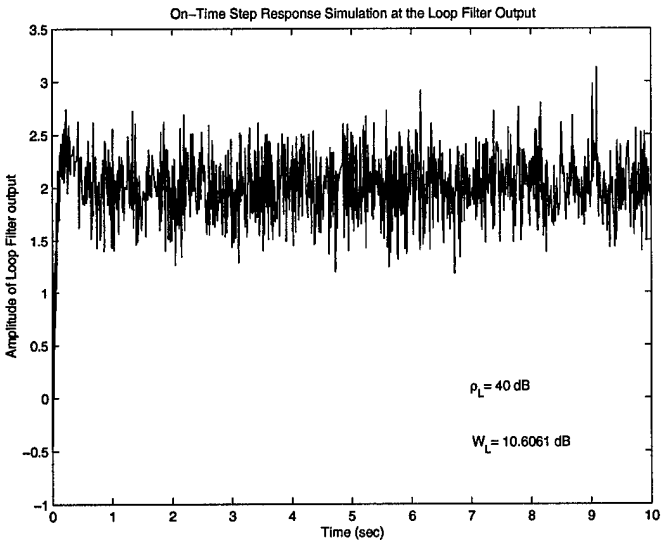


Figure 149 Simulation showing the ramp Response of the BBDLL closed loop at the loop filter output with, loop bandwidth $W_L = 10.606$ Hz, and closed loop signal-to-noise ratio $\rho_L = 40dB$

Appendix B. Multipath Simulation Codes

B.1 The scmp.m M-file

```
%This program establishes the S-curve of the BBDLL in the
%deterministic case in the presence of multipath
%the multipath parameter taken in this program is
% \alpha=[0.0, 0.2, 0.5, 0.7, 1.0 ,1.3]
%and the multipath strengths components vector is
%x=[1.0, 0.7, 0.6, 0.4, 0.8, 0.3].
%The program returns the correlation
%process for the maximal length of the C/A code
%constructing the intended  S-curve in
%the presence of multipath.
%
%Written by El-Sayed Abdel-Salam Gadallah, last Update 1 April 1998
%


clear
PRN=cacode(1); %C/A code generation
%Sampling process 10 sample per each chip.
for i=1:length(PRN)
   c(10*i-9:10*i)=PRN(i)*ones(1,10);
end
%shifted code due to multipath
c_early=zih(c,25);
c_late=zih(c,15);
c_a1=zih(c,13);
c_a2=zih(c,10);
c_a3=zih(c,8);
```

```
c_a4=zih(c,5);
c_a5=zih(c,2);

for shift=1:40

r=zih(c,shift);
r1=0.7*zih(c_a1,shift);
r2=0.6*zih(c_a2,shift);
r3=0.4*zih(c_a3,shift);
r4=0.8*zih(c_a4,shift);
r5=.3*zih(c_a5,shift);
r=r+r1+r2+r3+r4+r5;
g(shift,1:length(c))=r+r1+r2+r3+r4+r5;

  y1=mean(r.*c_early);
  y2=mean(r.*c_late);
  epsilon(shift)=y1-y2;% tracking error at the discriminator output


end
k=1:length(epsilon);
plot(k,epsilon)
```

B.2 The mpnos.m M-file

```
clear
load mp_tmplt.mat
CNR=22.2453;
P=2;
BW=.1023*10^6;
```

```
No=P/(10^(CNR/10));
sgma=sqrt(BW*No);

for mc=1:1000
for shift=1:40
noft=sgma*randn(1,length(c_early));
g(shift,1:length(c))=g(shift,1:length(c))+noft;
  y1=mean(g(shift,1:length(c)).*c_early);
  y2=mean(g(shift,1:length(c)).*c_late);
  epsilon(mc,shift)=y1-y2;
end
end
```

B.3 The nosprf.m M-file

```
load mpcal.mat
load mpfree.mat
eps=epsilon(:,15:45);
n=length(eps)
for i=1:n
   nos(i,:)=eps(i,:)-y;
end
sgmaeps_sim=std(nos);
sgmaeps_th=sqrt(2*(1+.7^2+.6^2+.4^2+.8^2+.3^2)*No*(1+1/1023)*10)
mnsga_sim=mean(sgmaeps_sim)
nosmn=mean(nos);
mnofmn=mean(nosmn)
sdnosmn=std(nosmn)
```

B.4 The noslintrp.m M-file

```
%This code is used in the SIMULINK interface to perform
%the discriminator characteristic even in the presence
%of noise. This m.file is nested inside the MATLAB
%function block taken from the nonlinear SIMULINK library
%the input of this code is a real number
%representing the time difference between the received
%code and the local replica and returns the discriminator
%characteristic, tracking error (the S-curve), the program
%also able to output the the tracking error even if the input
%difference in time is in-between the sampling points, using
%the linear interpolation.
%

function sc=lintrp(dif)

load lint.mat

sc=linterp(x,y,dif);
```

B.4.1 (rms.m) M-file.

```
%This program returns the rms tracking error, after loading
%the workspace with Mat-file created during
%the on-time simulation of the BBDLL model in Simulink
%
%Written By El-Sayed A.S. Gadalla, last update 25 June 1998
%
clear
load prma1.mat%the created mat file from the BBDLL step response simulation
```

```
x=prm(1,:);
y=prm(2,:);%prm is a define vector in loaded mat file representing
          %the step repose of the closed loop
indx1=find(x<=2);%To choose the  start time of steady state tracking error
id1=max(indx1);
id2=length(x);

v1=x(id1-1:id2-1);
v2=x(id1:id2);
Dt=v2-v1;%To determine the intervals between the sample points
te=y(id1-1:id2);% the tracking error vector
sgta=std(te)%the SD of the tracking error
te2=te.^2;% the tracking error squares
te_rms=sqrt(mean(te2))%the sqrt of the mean of T.E.
      %squares it can also represent the RMS T.E.
n=length(te);
mg=Dt(1:n-2)+Dt(2:n-1);
%RMS tracking error formula
rm=sqrt((.5/(v2(n-1)-v1(1)))*(te2(1)*Dt(1)+sum(te2(2:n-1).
*mg)+te2(n)*Dt(n-1)))
```

Appendix C. Multipath Estimation Methods

C.1 Multipath Estimation by The Search Method

Two reflectors are employed in this example. The multipath strength parameter is considered as function of time, and it is considered also as exponentially decayed with increasing the multipath time delay. The scenario is taken in this example of the time delay parameter is a ramp with slope $0.1t$. Table 12 shows summaries the time functions and the numerical values that were chosen for the multipath parameters.

Table 12 The Numerical Example of 2 reflectors

Multipath parameters	Numerical values	Description
T_c	unity	duration of PRN chip code
t	$0.0 \leq t \leq 15sec.$	time duration
α_1	$0.1t$	delay of the first reflector
α_2	$0.5+0.1t$	delay of the second reflector
$x_{o_{true}}$	unity	the direct path strength
$x_{1_{true}}$	$e^{-\alpha_1}$	the first reflector strength
$x_{2_{true}}$	$e^{-\alpha_2}$	the second reflector strength

Figure 150 through 152 show the estimation for both $\hat{\alpha}_1$ and $\hat{\alpha}_2$, the standared deviation of a AWGN is taken, 0.1, 0.01, and 0.001 respectively. Figure 153 through 155 show illustrate the results of estimation of the parameters $\hat{x}_o, \hat{x}_1$, and $\hat{x}_2$ respectively. Because of the singularity of the matrix $(H^T P_\nu^1 H)$, where the P_ν is the correlation matrix, is defined in Equation 83 some points in these figures were outliers. A correction for this singularity condition was made by adding a nonsingular matrix R; the matrix R is chosen as $\epsilon_{pseudo} I_3$ where ϵ_{pseudo} is a small real number and I_3 is the identity matrix.

C.2 Search Method Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   "Multipath Estimation by Search Method"
% This code for search of the multipath parameters $hat{\alpha}$ and
% $\hat{x}$. The time delay parameter $\alpha$ is taken for two reflectors
```

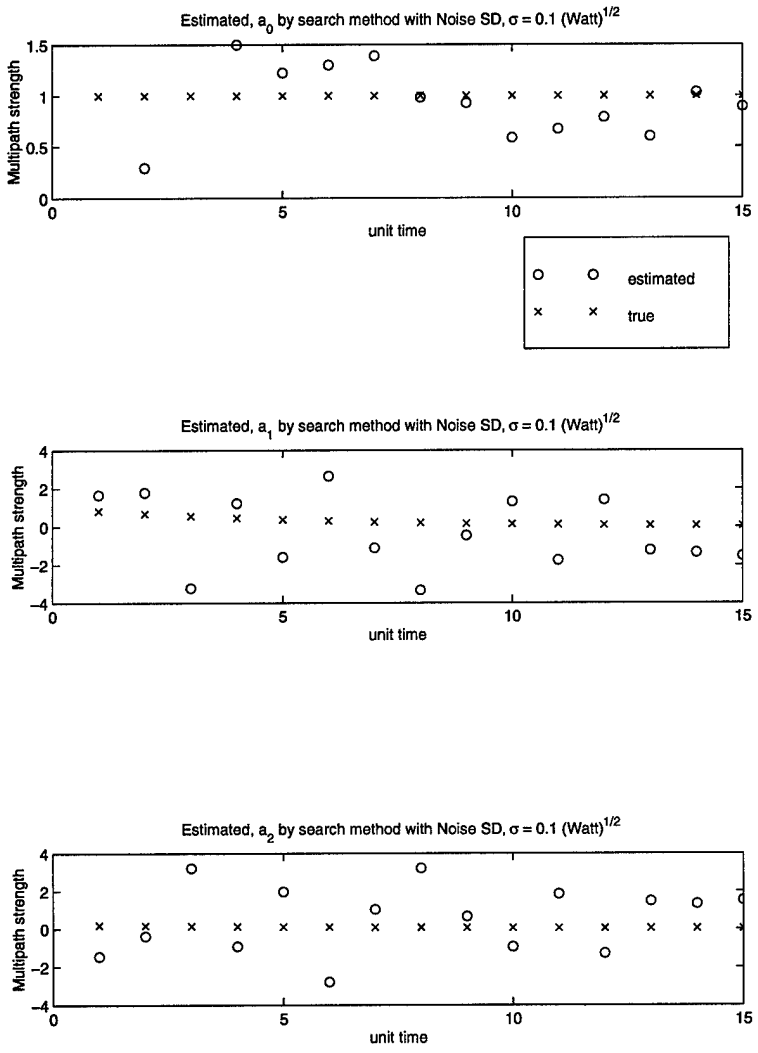


Figure 150 Estimated multipath signals strength $\hat{x}_o$, $\hat{x}_1$, and $\hat{x}_2$ by search, AWGN has $\sigma = 0.100$, $(Watt)^{1/2}$

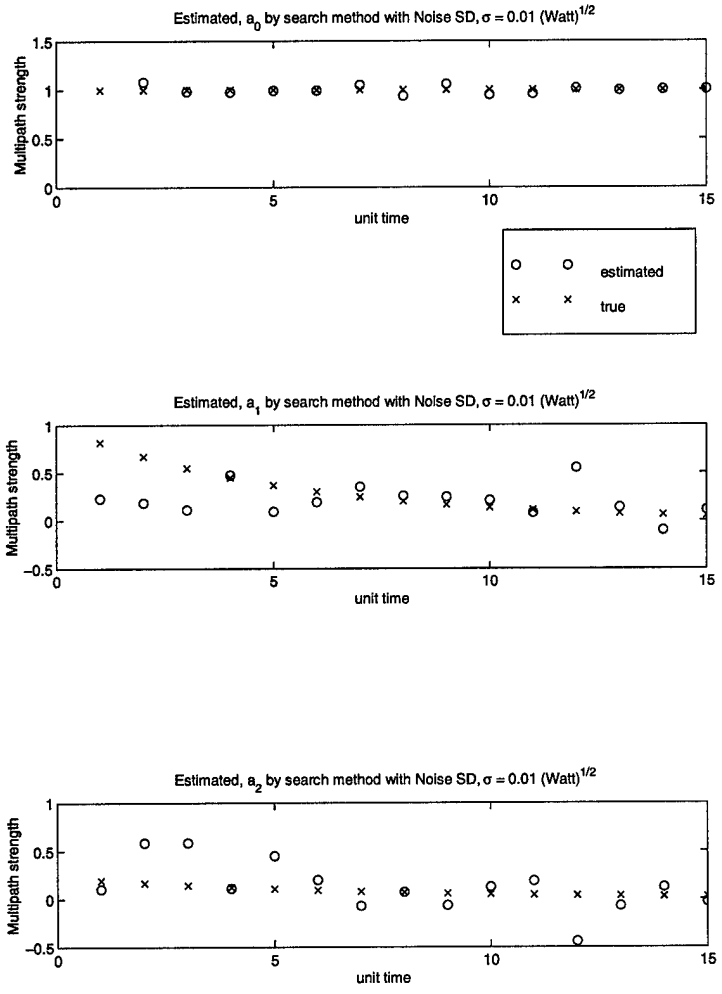


Figure 151 Estimated multipath signals strength $\hat{x}_o$, $\hat{x}_1$, and $\hat{x}_2$ by search, AWGN has $\sigma = 0.010$, $(Watt)^{1/2}$

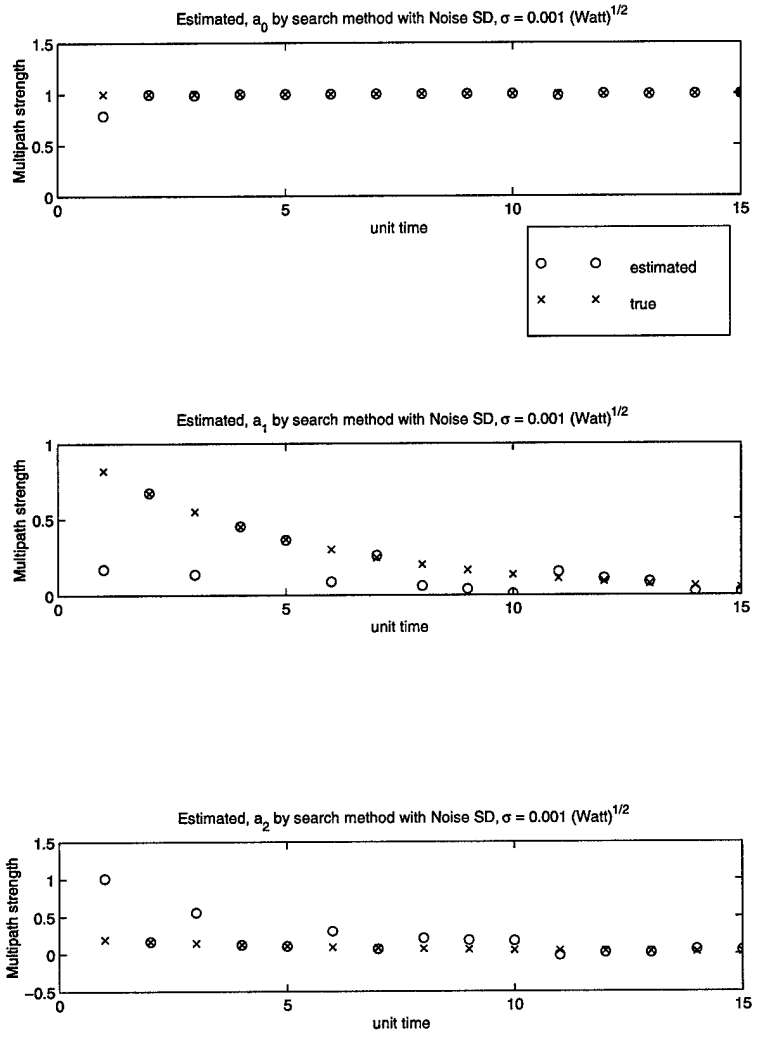


Figure 152 Estimated multipath signals strength $\hat{x}_o$, $\hat{x}_1$, and $\hat{x}_2$ by search, AWGN has $\sigma = 0.001$, $(Watt)^{1/2}$

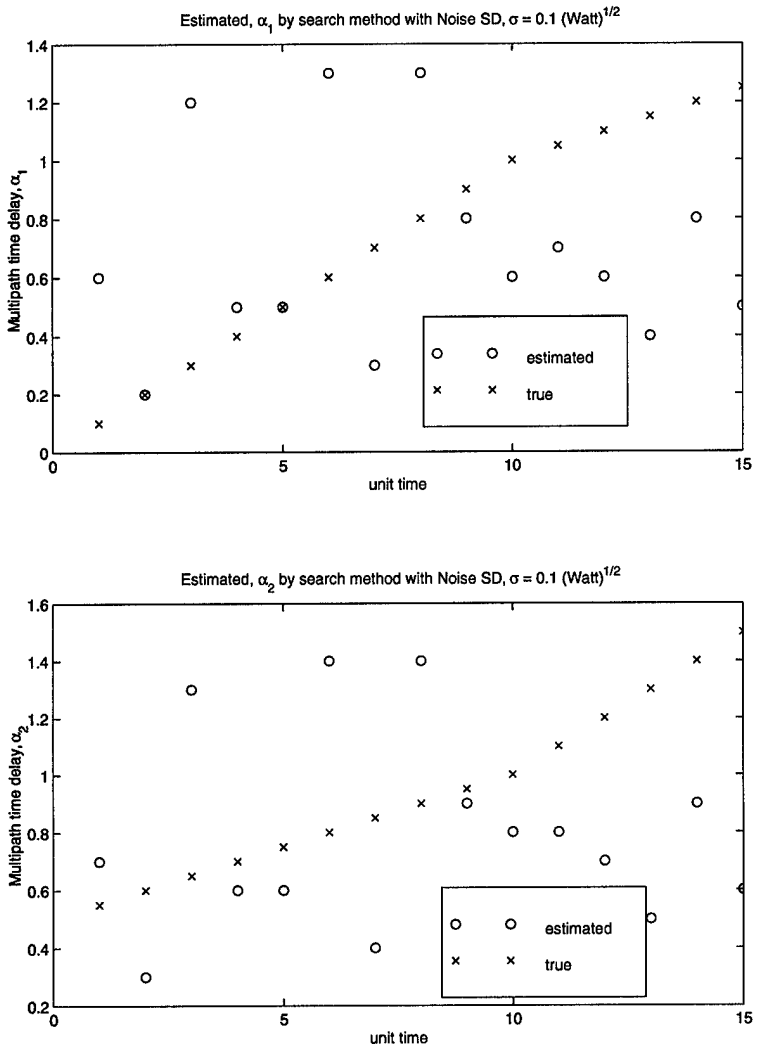


Figure 153 Estimated time delay for two reflectors $\hat{\alpha}_1$, $\hat{\alpha}_2$ by search, AWGN has $\sigma = 0.100$, $(Watt)^{1/2}$

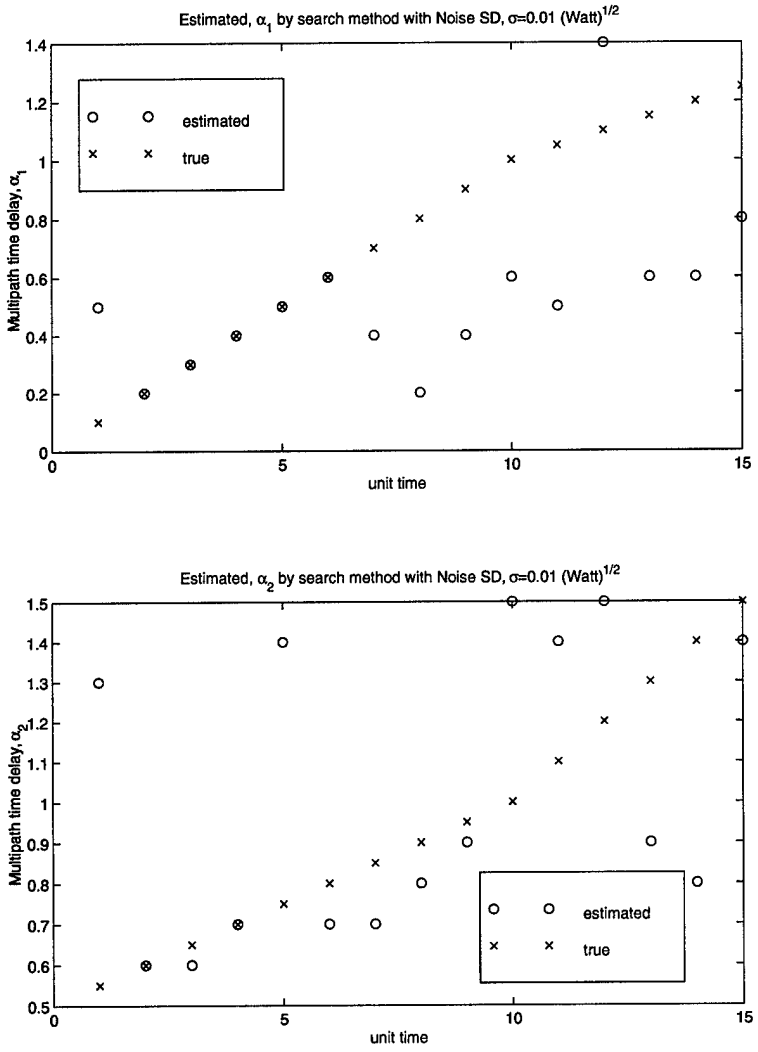


Figure 154 Estimated time delay for two reflectors $\hat{\alpha}_1$, $\hat{\alpha}_2$ by search, AWGN has $\sigma = 0.010$, $(Watt)^{1/2}$

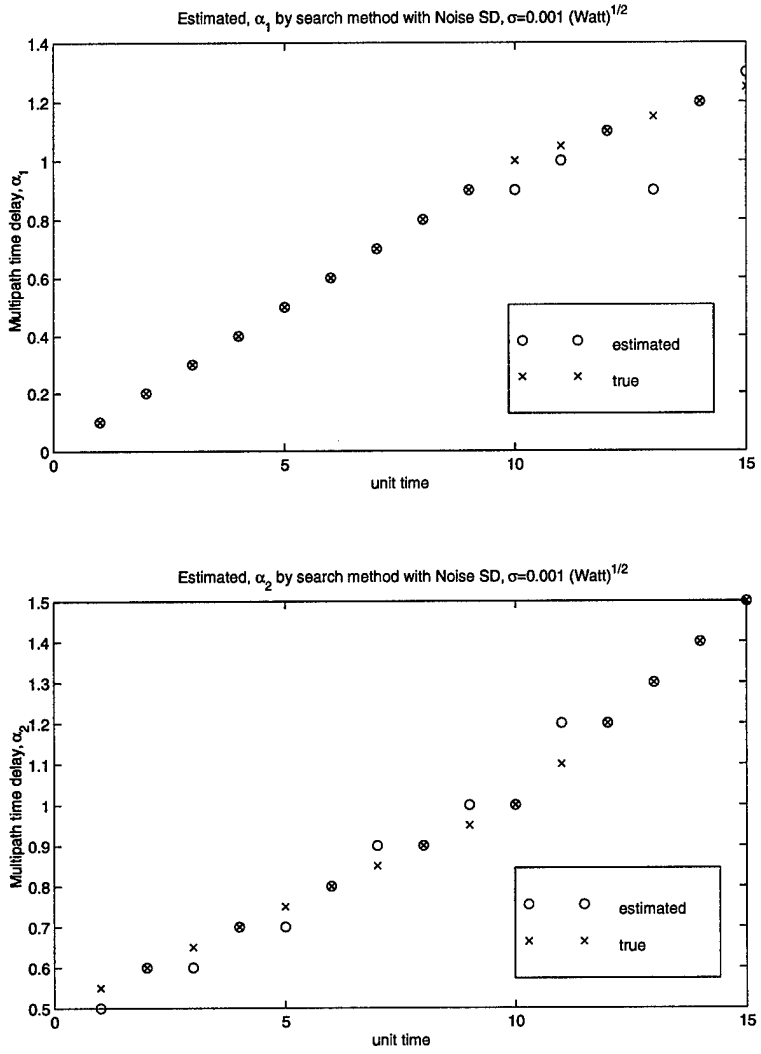


Figure 155 Estimated time delay for two reflectors $\hat{\alpha}_1$, $\hat{\alpha}_1$ by search, AWGN has $\sigma = 0.001$, $(Watt)^{1/2}$

```
% which is varying with time. The multipath strengths $x$ is decayed
% exponentially as long as $\alpha$ increased. The inputs of the code
% are the output correlations values from bank of correlators as
% a multiple measurements. These inputs are processed in the vector $R$
% by the (measv.m) code.
%Designed by El-Sayed Abdel-Salam Gadalla
%last version 06/25/98
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear
%Input Information
%%%%%%%%%%%%%%%%%

% n the number of reflectors
n=3;

% m the number of measurement
m=20;

%step of beta
delta=1/m;

%variance of AWGN
sgmav=0.1;

%pseudo number added to correct the singularity
pseudo=0.00000001;

% assumed $\beta$
for i=1:m
```

```
beta(i)=delta*i;
end

%step of search
step=.1;

%maximum alfa in GPS, Tc=1
alfamax=1.5;

%number of searching steps
nss=alfamax/step;

%loop for moving reflectors
for s=1:15

% true alfa to be estimated
alfatr=[0.0 .1*s .5+.05*s];

% true x to be estimated
for i=1:n
x(i)=exp(-alfatr(i)*i);
end

%generation of the vector of correlator measurements, R,
R=measv(alfatr,a,beta,P);


%the correlation matrix C
C=CorrMat(beta);
```

```
% The searching process
l=0.0;

for k3=1:nss
for k2=1:nss

 if k2 == k3
 alfa=step*[0.0 0.001 2*k3];
 else
 alfa=step*[0.0 k3 k2];
 end

 % the H matrix
   H=Hmat(alfa,beta);

% the estimated x, xhat
        l=l+1;
R=R+sgmav*randn(m,1);

D=pseudo* eye(n);
if rcond(H'*inv(C)*H)>=0.1e-10
xsrch=inv(H'*inv(C)*H+D)*H'*inv(C)*R;
end


% the function to be minimized
f=(R-H*xsrch)'*inv(C)*(R-H*xsrch);

if l==1 min=f;
alfhat=alfa;
```

```
xhat=xsrch;
else
if f <= min
min=f;
alfhat=alfa;
xhat=xsrch;
end
end

%two end of search
end
end

alfatr=sort(alfatr);

alftr1(s)=alfatr(2);
alftr2(s)=alfatr(3);
alfhat=sort(alfhat);
alfhat1(s)=alfhat(2);
alfhat2(s)=alfhat(3);

x1(s)=x(1);
x2(s)=x(2);
x3(s)=x(3);

xhat=xhat';

xhat1(s)=xhat(1);
xhat2(s)=xhat(2);
xhat3(s)=xhat(3);
```

```
%end of relative moving of reflections
end
```

C.3 α-Deployed Method (Noise free)

```
\%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% $\alpha$-Deploying method with noise free
%This code executes the initial detection  of multipath
%The input is the output of the bank of correlators
%The code returns the estimated multipath strength vector $\hat{x}$
%and the corresponding multipath delay vector $\hat{\alpha}$
%Designed by El-Sayed Abdel-Salam Gadalla
%last version 07/19/97
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%Input Information
%%%%%%%%%%%%%%%%%
clear


% n the true number of reflectors
n=6;


%step of alfa
step=.1;
p=log10(1/step)-1;
%number of alfa
```

```
nalfa=16*(10^(p));

% m the number of measurement (should be >= nalfa)
%m=nalfa+4;
m=2*nalfa;


%step of beta
delta=step;



% assumed beta
l=0.0;
for i=-m/2:m/2
l=l+1;
beta(l)=delta*i;
end

% number of correlators in the bank (adjusting No. of measurements)
m=length(beta);

% true alfa to be estimated

alfatr=[0.0 .2 .5 .7 1.0 1.3];

% true x to be estimated
x=[1 .7 .6 .4 .8 .3];

%generation of the vector of correlator measurements, R,
```

```
R=measv(alfatr,a,beta,P);

l=0.0;
alfa(1)=0.0;

for k=2:nalfa
alfa(k)= (.1^(p+1))*(k-1);
end;

% the H matrix
H=Hmat(alfa,beta);

xhat=inv(H'*H)*H'*R;
```

C.4 The Corresponding M File Functions

C.4.1 The Correlation Measurement Vector, R (measv.m).

```
%strength vector and spacing correlators bank vector
%the code returns the measurement vector, R.
function R=measv(alfatr,x,beta)
m=length(beta);
n=length(alfatr);
nx=length(x);
if nx~=n
   error('first two input vectors must be the same length')
end

   R=[];
for i=1:m
```

```
   sum=0.0;
    for j=1:n
if abs(beta(i)-alfatr(j))<=1;
         sum=sum+x(j)*(1-abs(beta(i)-alfatr(j)));
      end
      end
   R(i)=sum;
 end
R=R';
```

C.4.2 The correlation matrix P_ν (CorrMat.m). The correlation matrix P_ν is defined in the this code by C

```
%this code inputs the beta vector and returns
%the correlation matrix C=P_{\nu}
function Cee=CorrMat(beta)
m=length(beta);
Cee=[];
for i=1:m
for j=1:m
                        Cee(i,j)=Rc(beta(i)-beta(j));

end
end
```

C.4.3 The H matrix (Hmat.m).

```
%this code inputs the vector \alpha and \beta and returns the H matrix
% the H matrix
function H=Hmat(alfa,beta)
nalfa=length(alfa);
```

```
m=length(beta);
H=[];
for i=1:m
for j=1:nalfa
if abs(beta(i)-alfa(j))<=1;
H(i,j)=(1-abs(beta(i)-alfa(j)));
else
H(i,j)=0.0;
end
end
end
```

C.5 α-Deploying Method in the Presence of Multipath

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ''$\alpha$-Deploying Method in the presence of noise''
% Estimation of errors parameters of multiple reflectors assuming the
% presence of all possible reflectors. This code executes the initial
% detection of the multipath parameter. The input of the code is the
% outputs of the correlator bank (msmntv.m) function. The code returns
% the multipath parameters, the estimated multipath strength, $\hat{\x}$.
% The code contains the transformation from the white noise to color
% noise according to the cross-correlation among the correlators
% in the bank.
%
% Designed by E-Sayed Abdel-Salam Last version 10/28/97
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear
clf
```

```
%Input Information
%%%%%%%%%%%%%%%%%
BW=10;     %low-pass filter (Integrator) bandwidth (Hz)
CNR=40;    %Input signal-to-noise ratio (dB-Hz)
P=1;       %Input power signal (watt)

% true time delay $\alpha_{true}$ to be estimated

alfatr=[0.0 0.2 0.5 0.7 1.0 1.3];

% true multipath strength $x_{true}$ to be estimated
x=[1 .7 .6 .4 .8 .3];
a=x;


%assumed $\beta$ vector
beta=0:.1:1.5;
m=length(beta);


%generation of the vector of correlator measurements, R,
%The Input of the $\alpha$-Deploying method
R=msmntv(alfatr,a,beta,P);

%the correlation matrix C

C=CorrMat(beta);

% AWGN generation, we need to transform a white noise
% vector distributed as N~[0,var*I] into a noise vector, Nu,
```

```
% distributed as N~[0,var*C], where C is defined above
%

CHO=chol(C); %use cholesky factorization

%$\alpha$ to be deployed
alfa=0:.1:1.5;

% the H matrix
H=cosHmat(alfa,beta);
H=H*P;

%the PSD, No for the AWGN
No=P/(10^(CNR/10));

%the standared deviation $\sigma$ of the AWGN as LPF output
sgmav=sqrt(BW*No);

%the color noise vector
Nu=CHO'*randn(m,1);

%weighting the color noise by the SD $\sigma$
Nu=sgmav*Nu;
R_noise=R+Nu;

%MLE
PS=inv(H'*inv(C)*H)*H'*inv(C);

%estimated multipath strengths $\hat{x}$
xhat_noise=PS*R_noise;
```

```
%LS estimation (noise free case)
PS_free=inv(H'*H)*H';

%$\hat{x}_{free}
xhat_free=PS_free*R;
%end of the code
```

Appendix D. Monte Carlo Simulation for the Kalman Filter Application

D.1 Simulation Results

This Appendix shows the results of the Monte Carlo simulation of the ensemble average mean of the KF error. The bandwidth of the LPF is considered 10 kHz, the plots are conducted with SNR = 40 and 50 dB-Hz. The multipath delay paramters are taken at α_{true} =[0.0, 0.2, 0.5, 0.7, 1.0, 1.3] with a_{true}=[1.0, 0.7, 0.6, 0.4, 0.8, 0.3]. The number of the plots are drawn for all the assumed deployed α vector elements (16 elements). The Kalman filter input is the vector $\hat{a}$. Each curve in each plot shows the ensemble mean error of the expected $\hat{a}$ evaluated for 50 run of the Monte Carlo simulation. to validate the performance sensitivty of the Kalman filter designed, the simualtion is repeated for SNR=40 and 50 dB Hz which are corresponding to $\sigma = 1.00$ and $\sigma = 0.316$ $(Watt)^{\frac{1}{2}}$. The input power signal P is taken $P = 1$ $Watt$.

D.2 Simulation Codes

D.2.1 Monte Carlo Code.

```
%This code executes the Monte Carlo simulation and displays
%the plots of the ensemble average mean for the Kalman filter errors.
BW=10000;  %LPF bandwidth
CNR=50;    %signal-to-noise ratio
nofrun=50; %number of MC runs
rtfctr=.0001;% rate factor to change KF the sample period
[alfa,alfatr,hatxo,x,xhat_free,xhat_noise,e0,e1,e2,e3,e4,
e5,e6,e7,e8,e9,e10,e11,e12,e13,e14,e15]=Mntcrlo(BW,CNR,nofrun,rtfctr);
CNR=40;
[alfat,alfatrt,hatxot,xt,xhat_freet,xhat_noiset,et0,et1,et2,et3,et4,
et5,et6,et7,et8,et9,et10,et11,et12,et13,et14,et15]
=Mntcrlo(BW,CNR,nofrun,rtfctr);
```

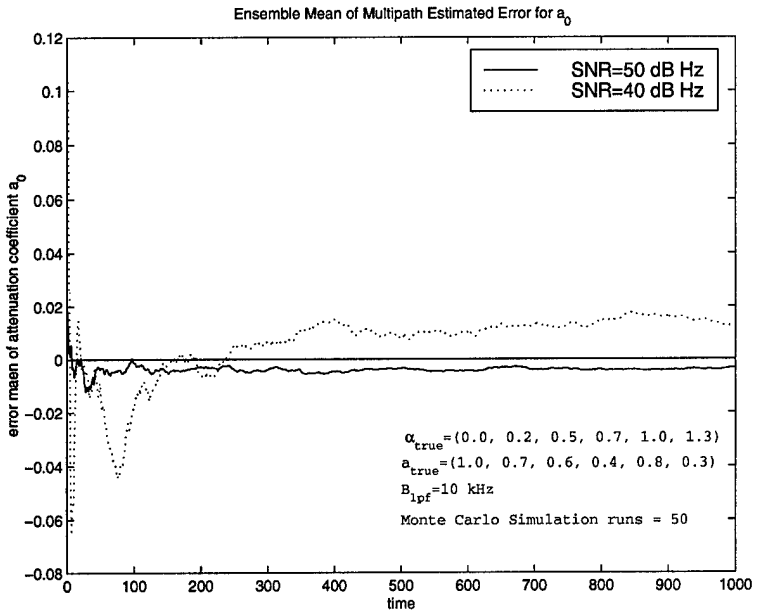


Figure 156 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_0

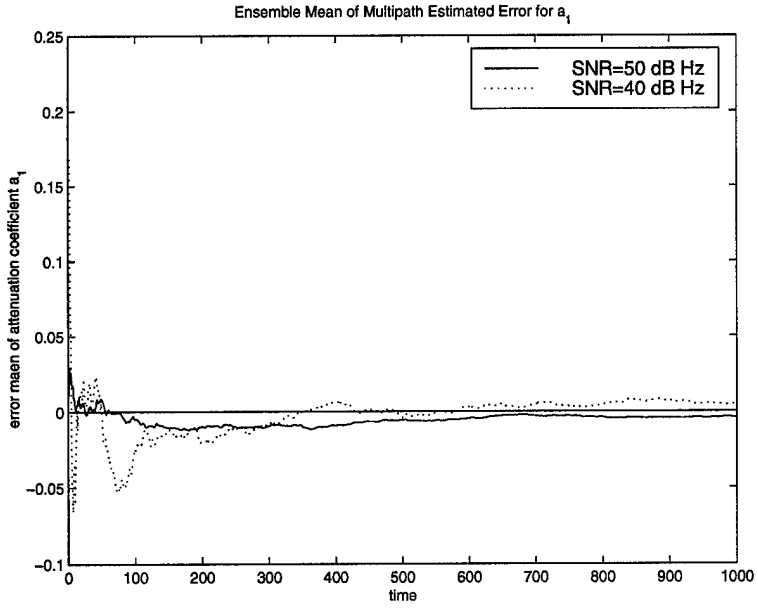


Figure 157 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_1

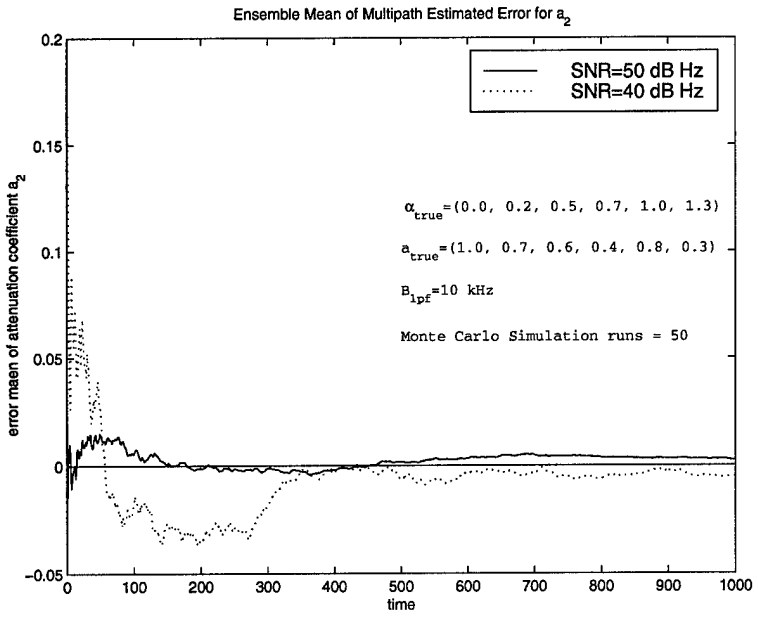


Figure 158 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_2

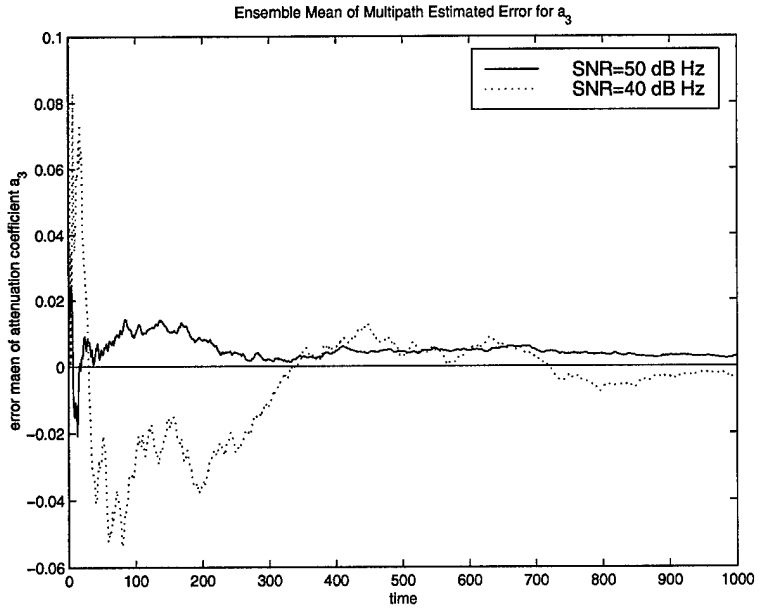


Figure 159 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_3

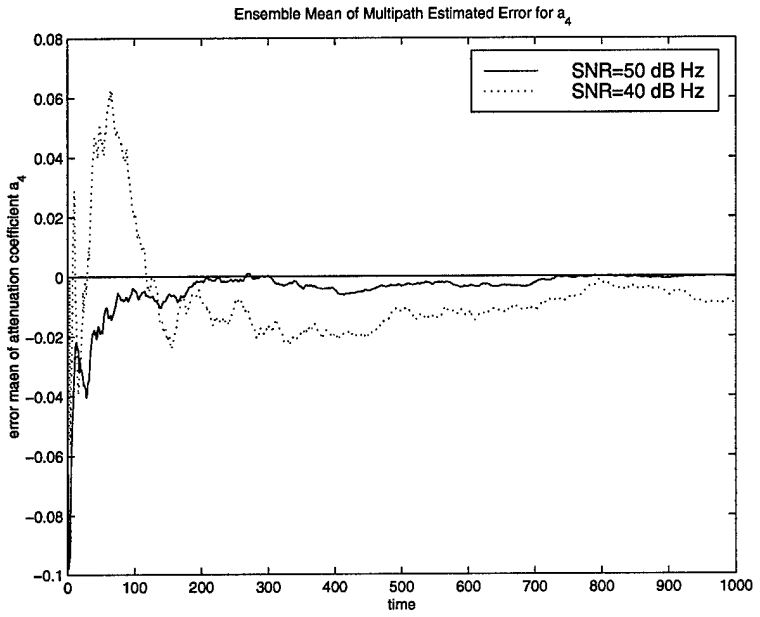


Figure 160 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_4

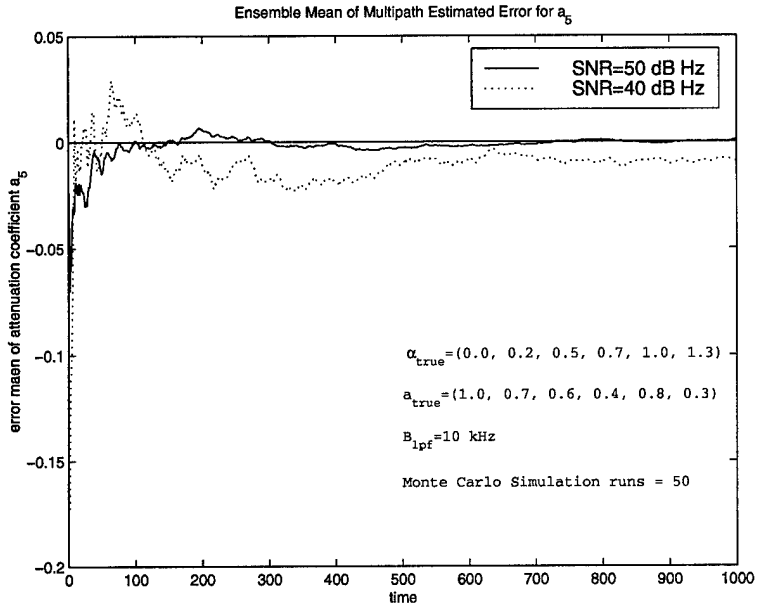


Figure 161 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_5

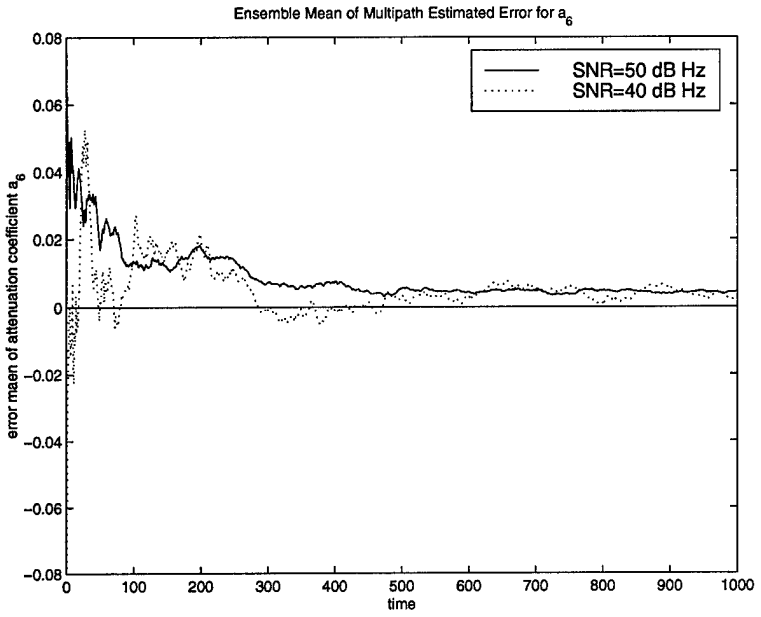


Figure 162 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_6

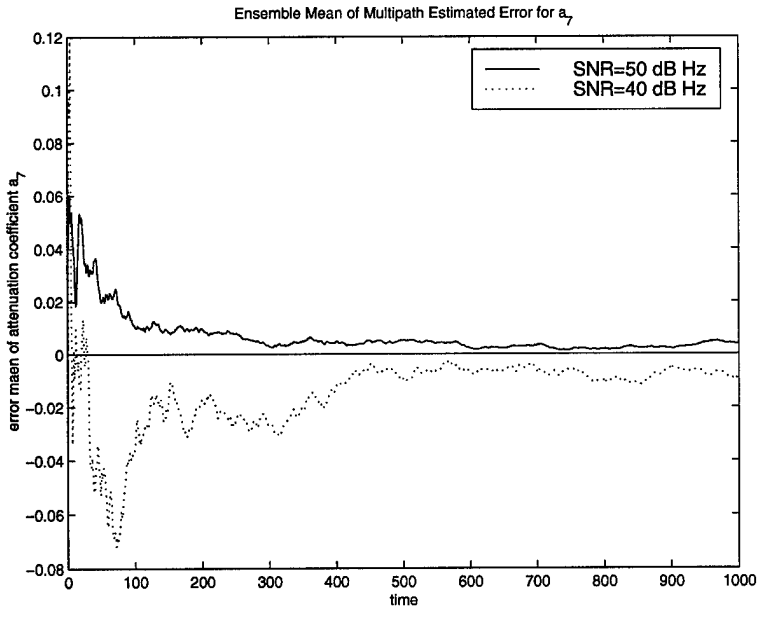


Figure 163 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_7

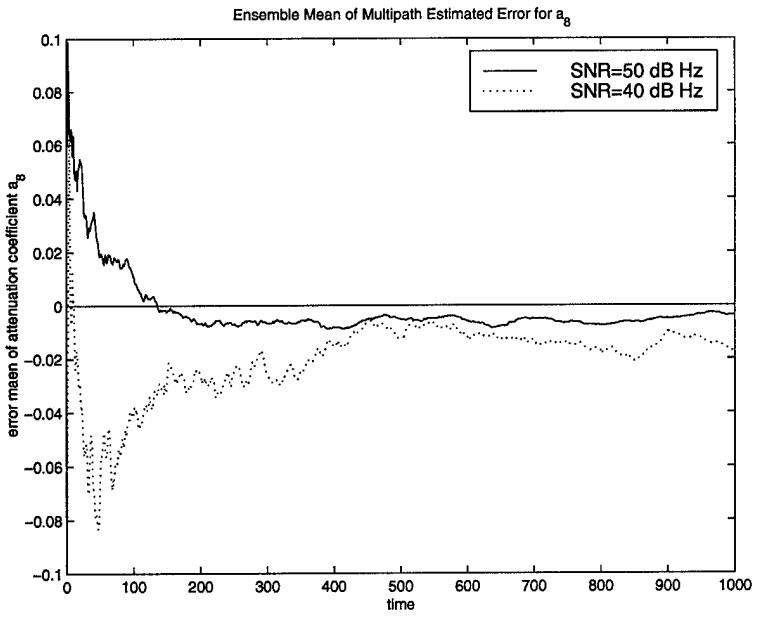


Figure 164 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_8

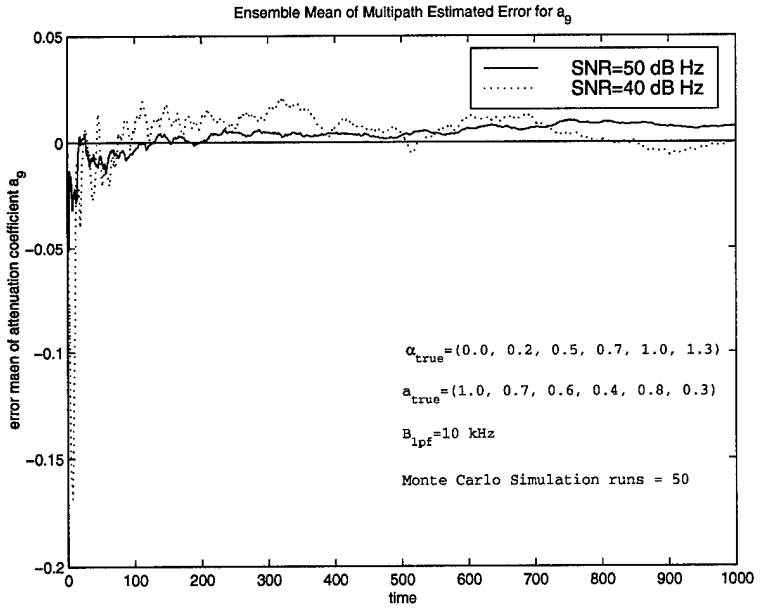


Figure 165 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_9

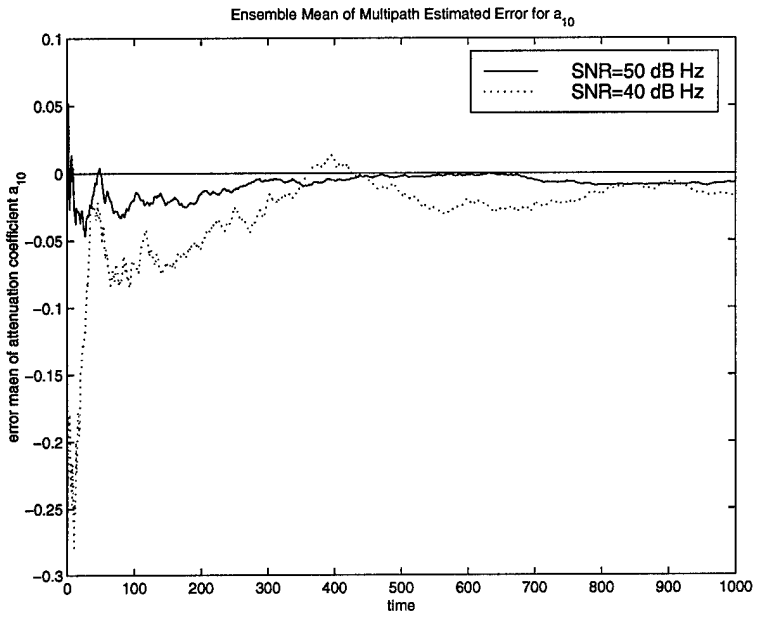


Figure 166 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_{10}

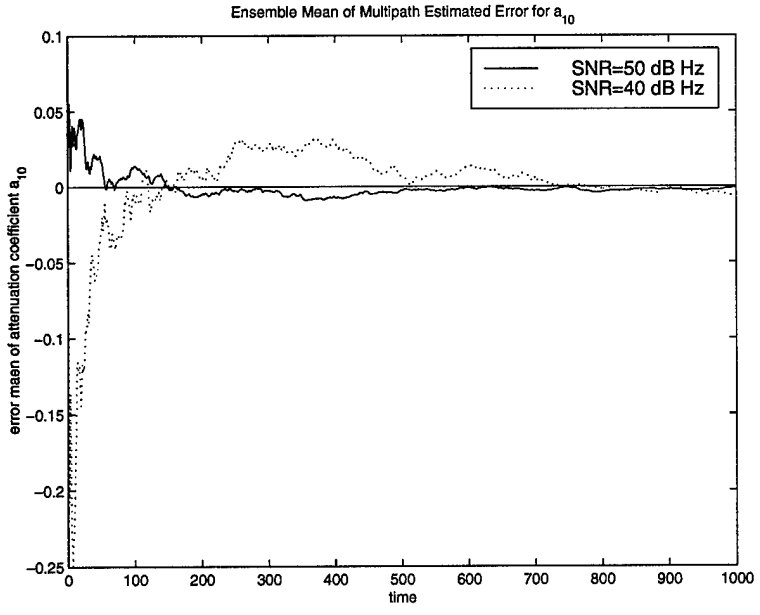


Figure 167 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_{11}

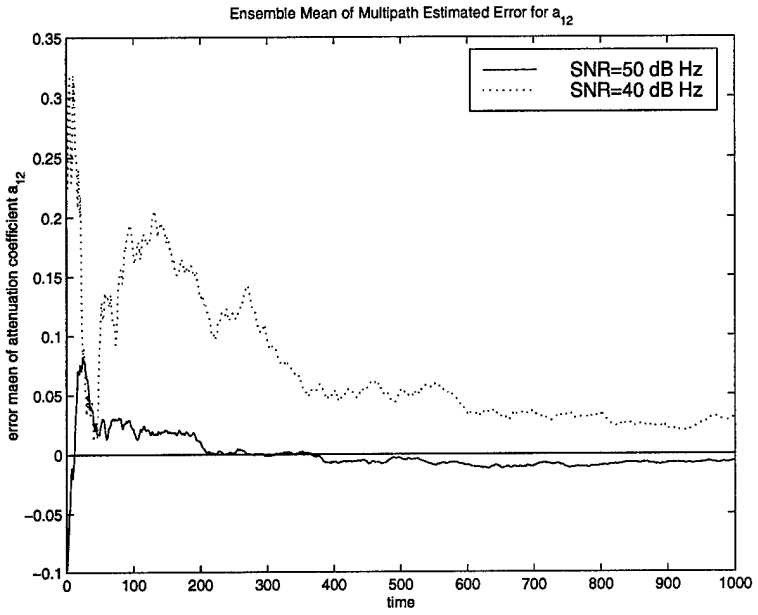


Figure 168 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_{12}

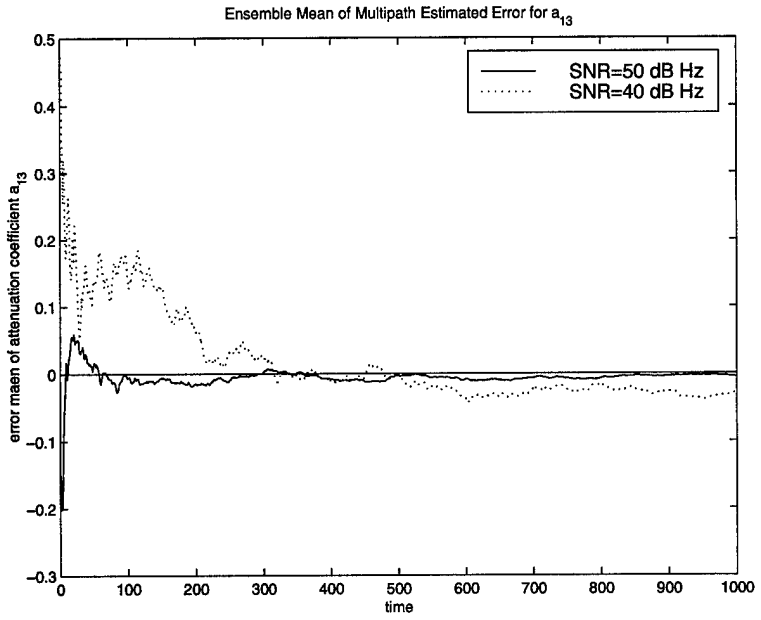


Figure 169 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_{13}

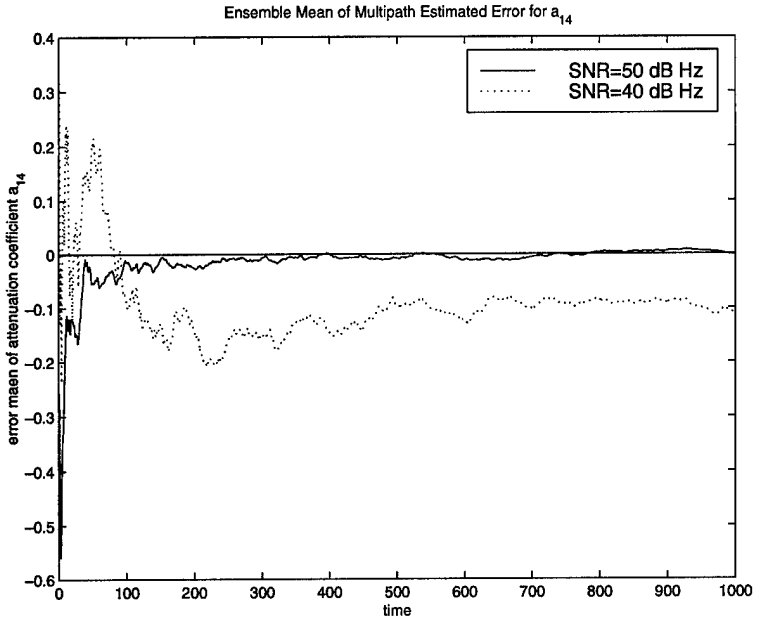


Figure 170 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_{14}

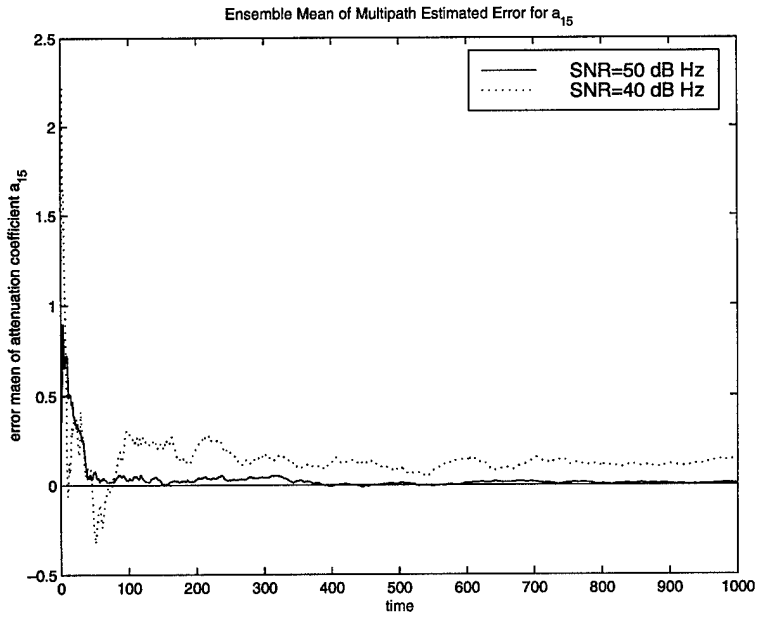


Figure 171 Kalman filtering simulation for multipath attenuation factor, ensemble mean of a_{15}

```
vec=1:length(e0);
zer=zeros(1,length(e0));
figure(1)
stem(alfa,xhat_free,'ro')
hold
stem(alfatr,x,'bx')
figure(2)
stem(alfa,xhat_noise,'ro')
hold
stem(alfatr,x,'bx')
xlabel(['\fontname{Helvetica}','Multipath Time Delay, ',
'\fontname{symbol}',' a'])
title([' [ \fontname{Helvetica}','After Kalman Filtering, SNR=30 dB-Hz,
B_{lpf}=1 kHz ]'])
ylabel(['\fontname{Helvetica}',' Multipath Components Attenuation, a'])
text(0.05,0.9,['\fontname{symbol}',' a','\fontname
{MS Sans Serif}_{true}=(0.0, 0.2, 0.5, 0.7, 1.0, 1.3)'])
text(.05,0.8,['\fontname{MS Sans Serif}','a_{true}=(1.0, 0.7, 0.6,
0.4, 0.8, 0.3)'])
legend('estimated','o','true','x')
figure(3)
stem(alfa,hatxo,'ro')
hold
stem(alfatr,x,'bx')
title([' [ \fontname{Helvetica}','Before Kalman Filtering,
SNR=30 dB-Hz, B_{lpf}=1 kHz]'])
xlabel(['\fontname{Helvetica}','Multipath Time Delay, ',
'\fontname{symbol}',' a'])
ylabel(['\fontname{Helvetica}',' Multipath Components Attenuation, a'])
text(0.05,15,['\fontname{symbol}',' a','\fontname{MS Sans Serif}_{true}
```

```
=(0.0, 0.2, 0.5, 0.7, 1.0, 1.3)'])
text(.05,10,['\fontname{MS Sans Serif}','a_{true}=(1.0, 0.7,
0.6, 0.4, 0.8, 0.3)'])
legend('estimated','o','true','x')
```

D.2.2 Kalman Filter Codes.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This program achives the Monte Calro  simultion process of the
% Kalman filter the measuring unit is the MLE using the
% $\alpha$-deploying method. the %program inputs are the
% LPF bandwidth (BW), signal-to noise ratio (CNR), the
% number of runs (nofrun), and
% (rtfctr). This program returns all the estimated vectors,
% and 16 ensemble %mean of the Kalman filter output.
%Designed by El-Sayed Abdel-Salam Gadallah
% Last version 11/13/97
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [alfa,alfatr,hatxo,x,xhat_free,xhat_noise,e0,e1,e2,e3,
e4,e5,e6,e7,e8,e9,e10,e11,e12,e13,e14,e15]
=Mntcrlo(BW,CNR,nofrun,rtfctr)

%Input Information
%%%%%%%%%%%%%%%%%%
BW=10000; %LPF bandwidth
CNR=50;   %SNR
Pow=1;    %Input power signal

% true alfa to be estimated

alfatr=[0.0 0.2 0.5 0.7 1.0 1.3];
```

```
% true x to be estimated
x=[1 .7 .6 .4 .8 .3];
a=x;

% assumed beta
beta=0:.1:1.5;
%m is the number of measurements
m=length(beta);

%generation of the vector of correlator measurements, R,
R=msmntv(alfatr,a,beta,Pow);

%the correlation matrix C
C=CorrMat(beta);
% AWGN generation, we need to transform a white noise
% vector distributed as N~[0,var*I] into a noise
% vector, Nu, distributed as N~[0,var*C], where C
% is defined above.
CHO=chol(C); %use cholesky factorization
%deployed alfa
alfa=0:.1:1.5;
% the H matrix
H=cosHmat(alfa,beta);
H=H*Pow;
% PSD No
No=Pow/(10^(CNR/10));
%variance of the AWGN
sgmav=sqrt(BW*No);
%PS matrix
```

```
PS=inv(H'*inv(C)*H)*H'*inv(C);
%Start of Monte Carlo run
for j=1:nofrun
Mu=0.0;
[zi,Muplus]=mleunit(sgmav,CHO,PS,R,m,Mu); %MLE unit
Mu=Muplus; %updating noise time correlated $\mu$
%the KF propagation states
hatxi=zi;
%the KF initial state is taken as the first run of the MLE
hatxo=zi;
%the KF propagation covariance matrix
Pi=PS*C*PS';
Dt=rtfctr*(1/BW); %sample period of the Kalman filter
Q_d=1-exp(-2*Dt*BW); %covariance matrix of \mu
interv=1000;%the interval of KF application (sec)
Rkf=PS*C*PS'*(Q_d);%the noise covariance matrix of \mu
for i=1:interv
   [ziplus1, Muplus]=mleunit(sgmav,CHO,PS,R,m,Mu);
%the update KF measurements and \mu
   Mu=Muplus; %the update noise time correlated $\mu$
   zd=ziplus1+zi*exp(-Dt*BW); %the update pseudo-measurement
   Ki=Pi*inv(Pi+Rkf);  %the update KF gain
   hatx_plus=hatxi+Ki*(zd-hatxi); %the update KF states
   P_plus=Pi-Ki*Pi; %the update KF covariance matrix
%the KF error outputs for the expected 16 strength multipath parameter
er15(j,i)=hatx_plus(16);
er14(j,i)=hatx_plus(15);
er13(j,i)=a(6)-hatx_plus(14);
er12(j,i)=hatx_plus(13);
er11(j,i)=hatx_plus(12);
```

```
er10(j,i)=a(5)-hatx_plus(11);
er9(j,i)=hatx_plus(10);
er8(j,i)=hatx_plus(9);
er7(j,i)=a(4)-hatx_plus(8);
er6(j,i)=hatx_plus(7);
er5(j,i)=a(3)-hatx_plus(6);
er4(j,i)=hatx_plus(5);
er3(j,i)=hatx_plus(4);
er2(j,i)=a(2)-hatx_plus(3);
er1(j,i)=hatx_plus(2);
er0(j,i)=a(1)-hatx_plus(1);
%return the propagation state for KF next run
   hatxi=hatx_plus;
   Pi=P_plus;
   zi=ziplus1;
end %end of KF run
end %end of Monte Carlo run

e0=mean(er0);
e1=mean(er1);
e2=mean(er2);
e3=mean(er3);
e4=mean(er4);
e5=mean(er5);
e6=mean(er6);
e7=mean(er7);
e8=mean(er8);
e9=mean(er9);
e10=mean(er10);
e11=mean(er11);
```

```
e12=mean(er12);
e13=mean(er13);
e14=mean(er14);
e15=mean(er15);

%the KF estimation
xhat_noise=hatxi;

%noise free MLE estimation
PS_free=inv(H'*H)*H';

xhat_free=PS_free*R;
```

D.2.3 The α-Deploying Estimator Unit Code (alfaunit.m). this m file is necessray to represent the update measurement of the Kalamn filter code in Section D.2.2

```
%this program represents the MLE unit. It returns the MLE of
% x̂ and the updated noise vector, μ


function [xhat_noise,Muplus]=alfaunit(sgmav,CHO,PS,R,m,Mu)
Nu=CHO'*randn(m,1);

Nu=sgmav*Nu;
j=0;
for i=length(Nu):-1:1
   j=j+1;
   tem(j)=Nu(i);
end
Nu=tem';
```

```
Muplus=.368*Mu+Nu;
R_noise=R+Muplus;
xhat_noise=PS*R_noise;
```

Appendix E. n-MRDLL Simulation

E.1 n-MRDLL Discriminator Noise Perormance Simulation

Figure 172 through 179 show the simulation of the n-MRDLL discriminator output by the ensemble average mean, input $SNR = 45, 40, \cdots 10$. Figure 180 throught 188 show the simulation results of the SD σ_ϵ at the discriminator output of n-MRDLL, for the same input SNR.

E.2 nmrdll.m (M-file)

```
clear
%C/A code generation
PRN=cacode(1);
%Sampling of the generated Code (10 sample per each chip)
for i=1:length(PRN)
   c(10*i-9:10*i)=PRN(i)*ones(1,10);
end
maxl=length(c);%code length which is 1023x10
CNR=30; %signal-to-noise ratio db.Hz
P=1;    %unity received power signal
BW=.1023*10^6; %BW at DLL processing it is considered the C/A code BW=1.023 MHz
% divided by the sampling rate 10 sample per chip
No=P/(10^(CNR/10)); %PSD W/Hz
sgma=sqrt(BW*No);% calculated SD of the AWGN

%c_early=zih(c,15);
%c_late=zih(c,5);
c_D=zih(c,35)-zih(c,25);%the difference (Early-Late) replica
c_D2=zih(c,35-2)-zih(c,25-2);% ,, of the first MP reflector (0.2Tc)delay
c_D5=zih(c,35-5)-zih(c,25-5);% ,, of the second MP reflector (0.5Tc)delay
```

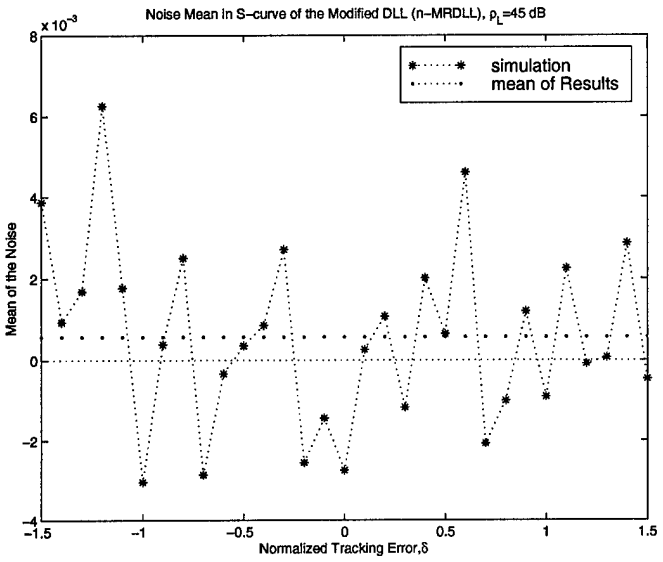


Figure 172 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 45$ dB-Hz

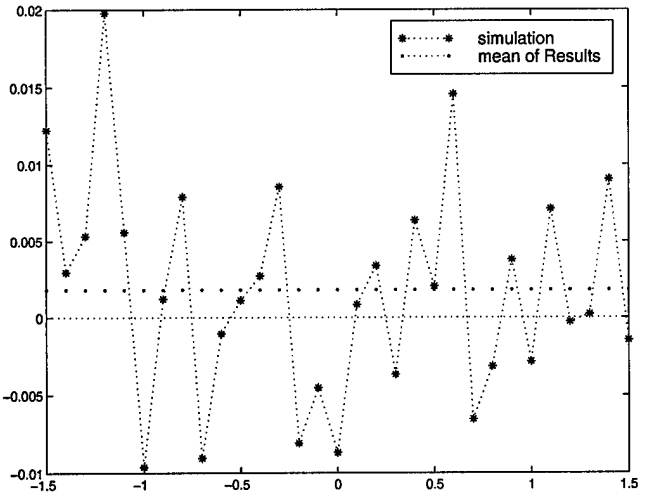


Figure 173 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 40$ dB-Hz

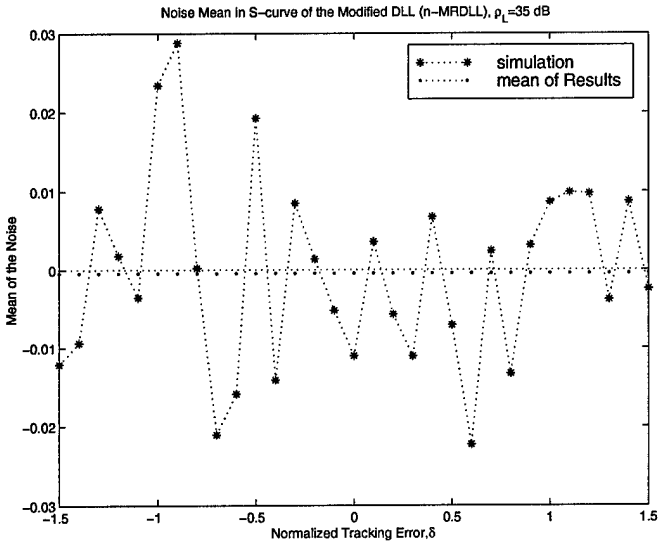


Figure 174 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 35$ dB-Hz

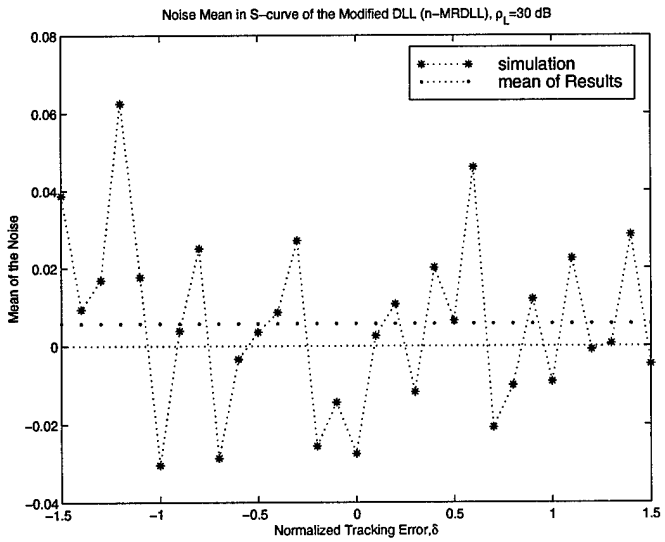


Figure 175 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 30$ dB-Hz

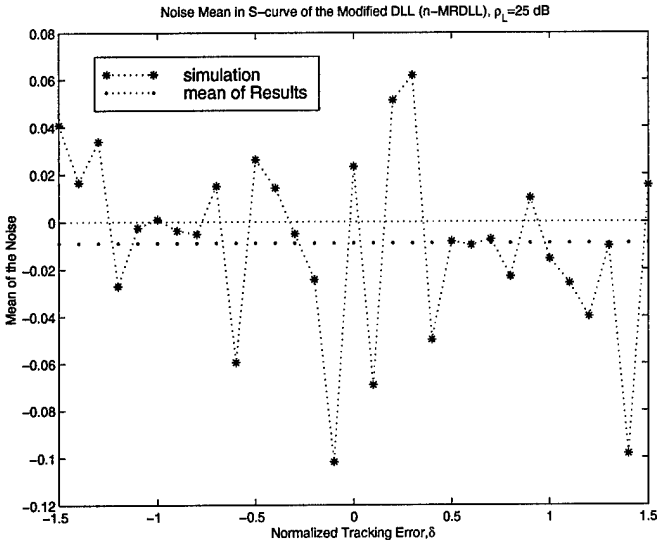


Figure 176 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 25$ dB-Hz

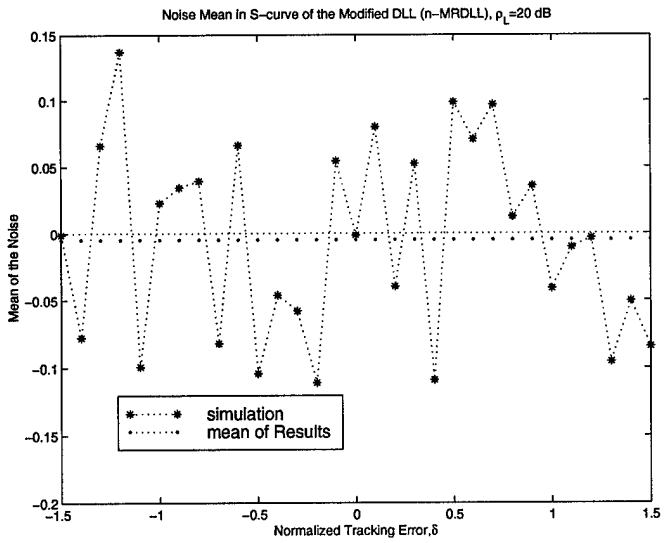


Figure 177 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 20$ dB-Hz

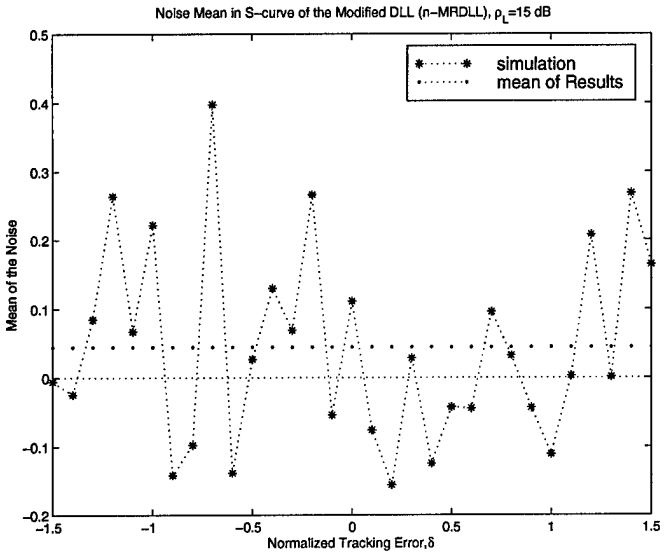


Figure 178 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 15$ db-Hz

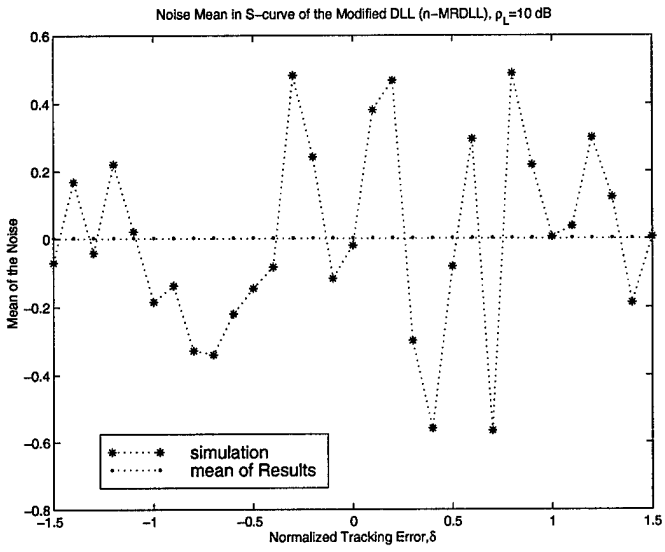


Figure 179 Simulation showing The ensemble average mean at the discriminator output of n-MRDLL, input $SNR = 10$dB-Hz

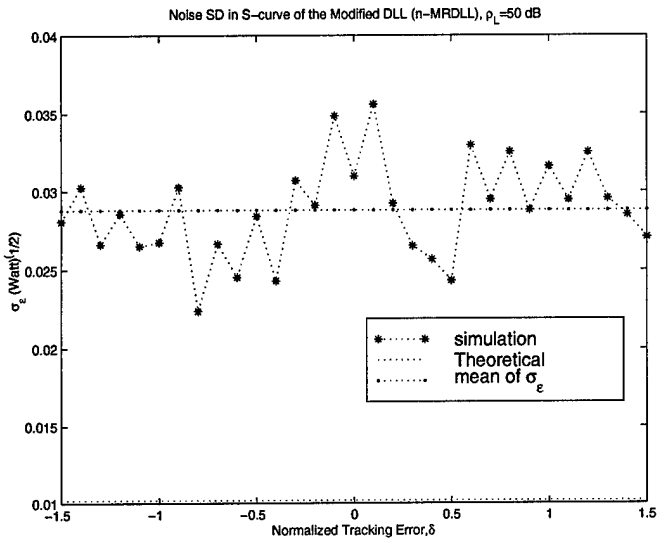


Figure 180 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 50$ dB-Hz

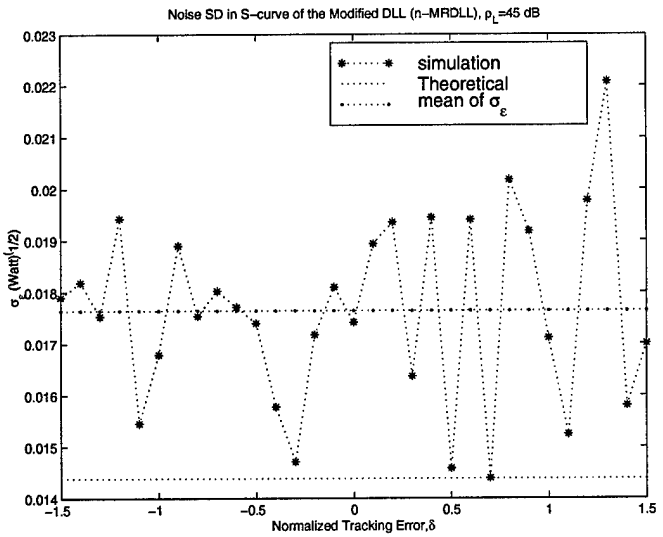


Figure 181 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 45$ dB-Hz

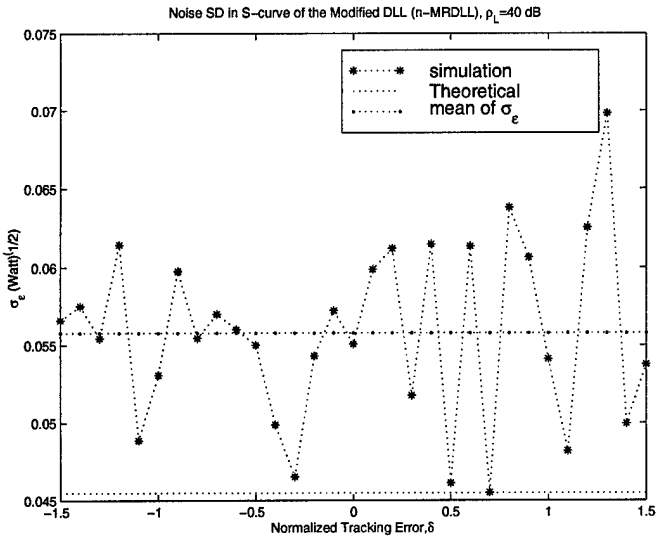


Figure 182 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 40$ dB-Hz

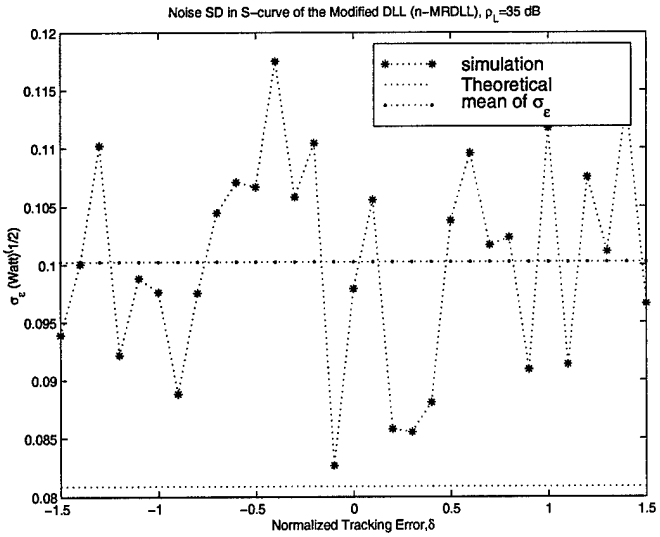


Figure 183 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 35$ dB-Hz

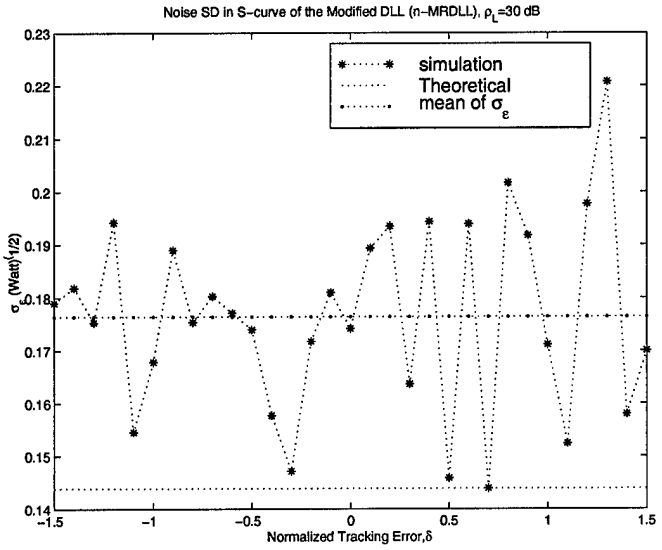


Figure 184 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 30$ dB-Hz

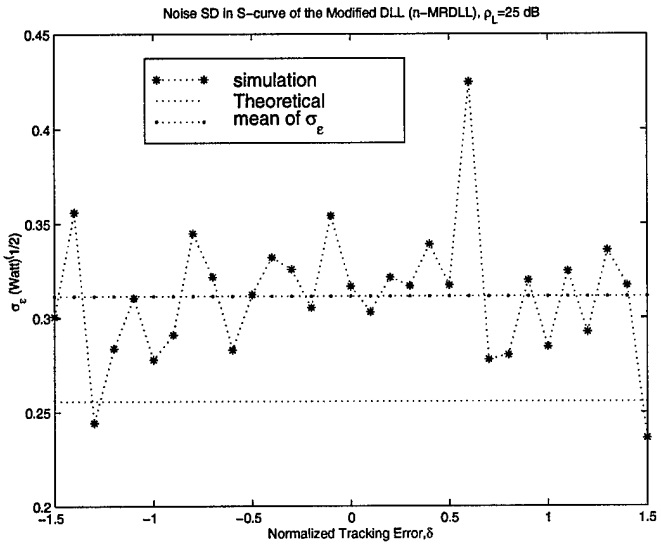


Figure 185 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 25$ dB-Hz

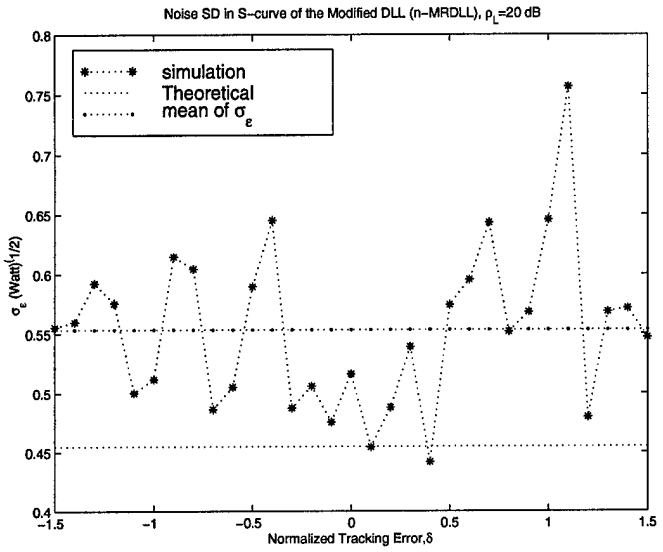


Figure 186 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 20$ dB-Hz

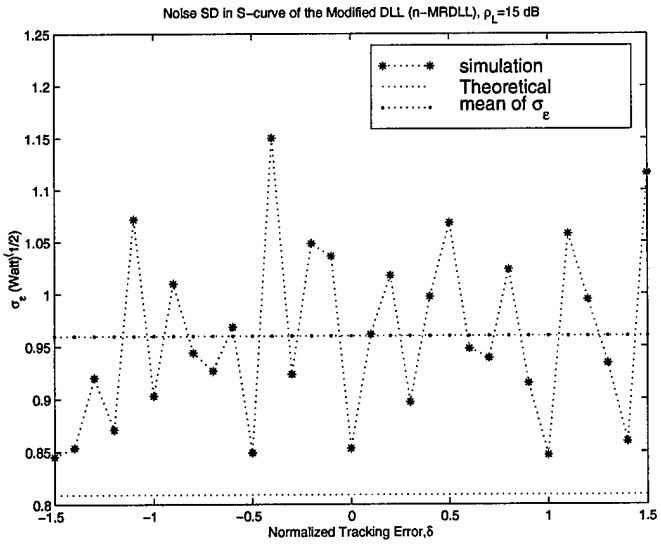


Figure 187 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 15$ dB-Hz

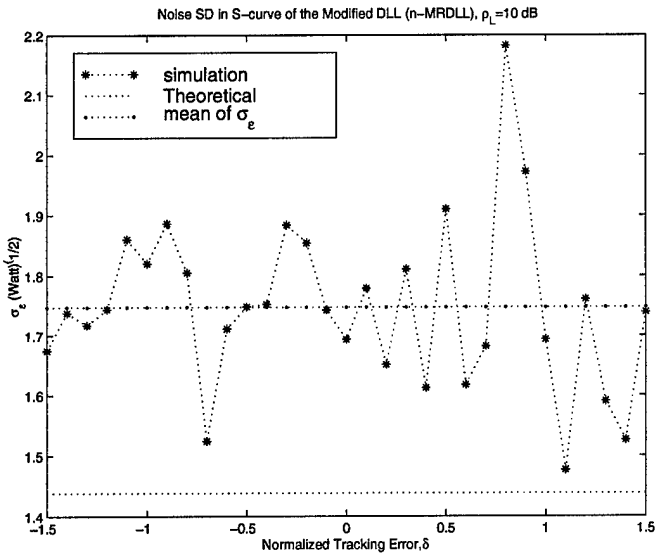


Figure 188 Simulation showing the SD σ_ϵ at the discriminator output of n-MRDLL, input $SNR = 10$ dB-Hz

```
c_D7=zih(c,35-7)-zih(c,25-7);% ,, of the third  ,,    ,,    (0.7Tc) ,,
c_D10=zih(c,35-10)-zih(c,25-10);% ,, ,, the fouth ,, ,, (1.0Tc) ,,
c_D13=zih(c,35-13)-zih(c,25-13);% ,, ,, the fifth ,, ,, (1.3Tc) ,,


%for mc=1:10
%nMRDLL Correlation process simulation
%Estimated MP time delay      = [0.0 0.2 0.5 0.7 1.0 1.3]
%Estimated Multipath Strength = [1.0 0.7 0.6 0.4 0.8 0.3]
for shift=15:45 %shift represenying the shift of the received signal
   %noft=sgma*randn(1,length(c_early));
r=zih(c,shift);%direct path signal
  B(shift,1:maxl)=r;
  r1=0.7*zih(c,shift-2);% the first MP component
  B1(shift,1:maxl)=r1;
  r2=0.6*zih(c,shift-5);%the second MP component
  B2(shift,1:maxl)=r2;
  r3=0.4*zih(c,shift-7);
```

```
  B3(shift,1:maxl)=r3;%the third MP component
  r4=0.8*zih(c,shift-10);
  B4(shift,1:maxl)=r4;%the fourth MP component
  r5=.3*zih(c,shift-13);
  B5(shift,1:maxl)=r5;%the fifth MP component
r=r+r1+r2+r3+r4+r5;%the actual received signal
%+noft;
%the correlation process is the mean of the code product
  y=mean(r.*c_D);
  y2=.7*mean(r.*c_D2);
  y5=.6*mean(r.*c_D5);
  y7=.4*mean(r.*c_D7);
  y10=.8*mean(r.*c_D10);
  y13=.3*mean(r.*c_D13);
  epsilon(shift)=y+y2+y5+y7+y10+y13;%the S-Curve for the corresponding
  %time difference "shift"
end
%end
k=1:length(epsilon);
plot(k,epsilon)
```

E.3 nrmc.m (M-file)

```
clear
load nr1.mat
CNR=47.2453;
P=2;
BW=.1023*10^6;
No=P/(10^(CNR/10));
sgma=sqrt(BW*No);
```

```
for mc=1:50
for shift=15:45
  noft=sgma*randn(1,length(c_D));

  r=B(shift,1:maxl);
    r1=B1(shift,1:maxl);
   r2=B2(shift,1:maxl);
   r3=B3(shift,1:maxl);
   r4=B4(shift,1:maxl);
   r5=B5(shift,1:maxl);
  r=r+r1+r2+r3+r4+r5+noft;
  y=mean(r.*c_D);
  y2=.7*mean(r.*c_D2);
  y5=.6*mean(r.*c_D5);
  y7=.4*mean(r.*c_D7);
  y10=.8*mean(r.*c_D10);
  y13=.3*mean(r.*c_D13);
  epsilon(mc,shift)=y+y2+y5+y7+y10+y13;
end
end
```

Bibliography

1. Axelard, P. and others. "SNR-Based Multipath Error Correction for GPS Differential Phase," *IEEE Transactions on Aerospace and Electronic Systems*, *32*(2):650–659 (1996).
2. Best, Roland E. *Phase-Locked Loops Theory, Design, and Application*. New York: McGraw-Hill,Inc, 1984.
3. Brodin, Gary. "GNSS Code and Carrier Tracking in Presence of Multipath." *Proceedings of the ION GPS-96*. 1996.
4. Comp, Christopher J. and Penina Axelard. "An Adaptive SNR-Based Carrier Phase Multipath Mitigation Technique." *The Institute of Navigation, Proceedings of the ION GPS-96*. Part 1. 1996.
5. Dixon, Rebort C. *SPREAD SPECTRUM SYSTEMS* (Second edition Edition). New York: John Wiley & Sons, Inc, 1984.
6. Doris, Dolphin and Abdelahad Benhallam. "On Correlation Processes Reducing Multipath Error in the L1 GPS Receiver." *The Institute of Navigation, Proceedings of the ION GPS-96*. 1996.
7. El-Rabbany, Ahmed. "Tempral Characteristics of Multipath Errors." *Proceedings of the ION GPS-95 8th International Technical Meeting*. 133–141. September 1995.
8. Gardner, Floyd M. *Phaselock Techniques* (second Edition). New York: Johon Wiley and Sons, 1979.
9. Grain, Lionel and others. "Strobe and Edge Correlator Multipath Mitigation for Code." *Proceedings of the ION GPS-96*. Part 1. 1996.
10. Ha, Tri T. *Digital Satellite Communications*. Macmillan Publishing Company A Division of Macmillan, 1986.
11. Hardwick, C. D. and J. Liu. "Characterization of Phase and Multipath Errors for an Aircraft GPS Antenna." *Proceedings of the ION GPS-95 8th International Technical Meeting*. 491–498. September 1995.
12. II, Leon W. Couch. *Digital and Analog Communication Systems*. New York: Macmillan Publishing Co., Inc., 1983.
13. Klapper, Jacob and John T. Frankle. *Phase-Locked and Frequency-Feedback Systems*. New York: Academic Press, Inc, 1972.
14. Laxton, Mark C. *Analysis and Simulation of a New Code Tracking Loop For GPS Multipath Mitigation*. WPAFB, Ohio: Master Thesis, Air Force Institute of Technology, December 1996.
15. Luenberger, David G. *Optimization by Vector Space Methods*. New York: John Wiley & Sons, 1969.

16. Maybeck, Peter S. *Stochastic Models, Estimation and Control, 1*. New York: Academic Press, Inc., 1979.

17. Maybeck, Peter S. *Stochastic Models, Estimation and Control, 2*. New York: Academic Press, Inc., 1982.

18. M.Tranquilla, J. and others. "Analysis of a Choke Ring Ground Plane for Multipath Control In Global Positioning System (GPS) Applications," *IEEE Transactions on Antennas and Propagation, 42*(7):905–911 (1994).

19. Nicholson, David L. *SPREAD SPECTRUM Signal Design LPE & AJ Systems*. Rockville, Maryland: Computer Science Press, Inc, 1988.

20. Peterson, Roger L. and others. *Introduction to Spread Spectrum Communications*. Englewood Cliffs, New Jersey: Prentice-Hall, 1995.

21. Raquet, J. and G. Lachaplle. "Determination and Reduction of GPS Reference Station Multipath Using Multiple Receivers." *The Institute of Navigation, Proceedings of the ION GPS-96*. Part 1. 1996.

22. Riggin, Robert N. and Stewart L. DeVilbiss. *Satellite Navigation Using the Global Positioning System*. Air Force Institute of Technology, Textbook Manuscript, 1996.

23. Robins, W.P. *Phase Noise in Signal Sources*. London: Peter Peregrinus, 1982.

24. Scharf, Louis L. *Statistical Signal Processing*. New York: Addison-Wesley, 1991.

25. Sheen, Wern-Ho and Gordon L. Stuber. "A Coherent Tracking Loop for Direct Sequence Spread Spectrum Systems on Frequency Selective Fading Channels." *Proceedings of the IEEE International Conference on Communications*. 1364–1368. June 1995.

26. Simon, Marvin K. and others. *SPREAD SPECTRUM COMMUNICATIONS*. Rockville, Maryland: Computer Science Press, Inc, 1985.

27. Spilker, J. J. *Digital Communication by Satellite*. New Jersey: Prentic-Hall, INC Englewood Cliffs, 1977.

28. Spliker, J. J. "Signal Structure and Performance Charcteristics," *NAVIGATION, 1* (1980).

29. Townsend, Bryan R. and others. "Peformance Evaluation of the Multipath Estimating Delay Lock Loop," *NAVIGATION: Journal of the Institute of Navigation, 42*(3) (fall 1995).

30. VanDierendonck, A.J. and others. "Theory and performance of narrow correlator spacing in a GPS receiver," *NAVIGATION: Journal of the Institute of Navigation, 39*(3):265–283 (Fall 1992).

31. VanNee, Richard. "Multipath effects on GPS code phase measurements," *NAVIGATION: Journal of the Institute of Navigation, 39*(2):177–190 (Summer 1992).

32. Weill, Lawrence. "C/A Code Pseudoranging Accuracy - How Good Can It Get?." *Proceedings of the ION GPS-94 7th International Technical Meeting*. 133–141. September 1994.

Vita

Lt.Col. El-Sayed Abdel-Salam Gadallah was born in Suez, Egypt, if you know Suez Canal, then you can locate where this lovely city, it is in the top of Suez Gulf of the red sea. Lt.Col. Gadallah has three Children Mona (girl), Mohamed (boy), and Asmaa' (girl).
In 1983, he received his Bachelor of Science in Electrical Engineering (BSEE) with honor from Military Technical College (MTC), Cairo, Egypt. In 1987, he was appointed as an instructor assistance in MTC. In 1992, he received his Master of Science in Electrical Engineering (MSEE) from MTC. He work as Assistant Lecturer in MTC since 1992. In September 1994, he granted a delegation to USA, AFIT, Electrical and Computer Engineering Department for receiving Ph.D. degree in Electrical Engineering, that this dissertation is introduced for. He introduced two papers, the first is presented at the ION conference, May 1998 in Denver, Colorado, entitled "Signal Processing For Multipath Mitigation in GPS". The second is introduced in ION GPS conference, September 1998 in Nashville, Tennessee, entitled "Design of GPS Receiver Code and Carrier Tracking Loops for Multipath Mitigation".

Permanent address: 41 Block No. 506
Imtedad Zahraa Nasr City
Cairo, EGYPT

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE
Global Positioning System (GPS) Receiver Design For Multipath Mitigation

5. FUNDING NUMBERS

6. AUTHOR(S)
Lt.Col(Egypt) El-Sayed Abdel-Salam Gadallah

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
AFIT/ENG

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/DS/ENG/98-11

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFRL/SNAR
Capt Jaun R. Vasques
WPAFB, OH, 45433-7318
DSN 785-5668x4014

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
Dr. Meir Pachter,
AFIT/ENG, 2950 P Street, Bldg 640
WPAFB, OH, 45433-7765 email: mpachter@afit.af.mil

12a. DISTRIBUTION AVAILABILITY STATEMENT

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

Multipath effects are a source of error degrading the accuracy of DGPS signal processing. The statistical models of multipath are determined by user location and, in addition are time varying. There is no unified statistical model for the multipath signal. Therefore the solution of the multipath problem using statistical models is difficult. This research introduces a new estimator that can detect the presence of multipath, can determine the unknown number of multipath components and can estimate multipath parameters in the GPS receiver (time delay and attenuation coefficients). Furthermore the multipath signal parameters are estimated at any instant of observation. The new estimator is based on maximum likelihood estimation applied to multiple observations of a linear model (regression form) of the received signal. In addition, the estimator is based on a recursive deployment of the multipath time delay. An improvement is achieved to the accuracy of multipath estimates at a low signal-to-noise level by applying Kalman filtering as a cascaded estimator. Kalman filtering application can be considered as an important tool for separating the direct path signal from multipath in noise. This dissertation also includes the design of new modified tracking loops endowed with the mentioned estimator: a modified Phase Lock Loop (PLL) for carrier tracking and a modified Delay Locked-Loop (DLL) in the code tracking. The modified loops can properly track the received direct signal in the presence of multipaths where the standard tracking loops are disabled. Simulations of the standard and the modified loops are presented. Tracking and performance in noise are investigated and a future work is suggested.

14. SUBJECT TERMS	15. NUMBER OF PAGES
multipath mitigation, tracking loops, GPS receiver	270
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	