Air Force Institute of Technology
AFIT Scholar

Theses and Dissertations | Student Graduate Works

12-1996

Analysis and Simulation of a New Code Tracking Loop for GPS Multipath Mitigation

Mark C. Laxton

Follow this and additional works at: https://scholar.afit.edu/etd

 Part of the Software Engineering Commons

Recommended Citation

Laxton, Mark C., "Analysis and Simulation of a New Code Tracking Loop for GPS Multipath Mitigation" (1996). *Theses and Dissertations*. 5925.
https://scholar.afit.edu/etd/5925

This Thesis is brought to you for free and open access by the Student Graduate Works at AFIT Scholar. It has been accepted for inclusion in Theses and Dissertations by an authorized administrator of AFIT Scholar. For more information, please contact AFIT.ENWL.Repository@us.af.mil.

AFIT/GE/ENG/96D-10

ANALYSIS AND SIMULATION OF A NEW CODE TRACKING LOOP FOR GPS MULTIPATH MITIGATION

THESIS

Mark C. Laxton
Captain, USAF

AFIT/GE/ENG/96D-10

Approved for public release; distribution unlimited

19970108 054

DTIC QUALITY INSPECTED 5

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U. S. Government.

AFIT/GE/ENG/96D-10

ANALYSIS AND SIMULATION OF A NEW CODE TRACKING LOOP FOR GPS MULTIPATH MITIGATION

THESIS

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

Master of Science in Electrical Engineering

Mark C. Laxton, B.S.

Captain, USAF

December, 1996

Approved for public release; distribution unlimited

Acknowledgements

First of all, I would like to thank the 'Comm guys' (Rod, Rob, Jeff, Al, and Eric) for keeping me sane through the entire AFIT experience; your humor, homework help, and especially your friendships are much appreciated. I'd like to thank my advisor, Capt Stew DeVilbiss, for his patience, guidance, and his lively discussions on Big Ten football. Finally, and most importantly, I want to thank my precious wife, whose love and support made the rough times easier and the good times even better; I love you, Tina.

Mark C. Laxton

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation Page

List of Symbols

Symbol Page

Abstract

This thesis proposes a new direct-sequence spread spectrum (DS/SS) code phase tracking loop which mitigates the effects of multipath interference on code phase tracking error; such errors can translate to significant range measurement errors in DS/SS ranging systems such as Global Positioning System (GPS). The new code tracking loop, called the modified RAKE delay-lock loop (MRDLL), uses maximum-likelihood (ML) signal parameter estimation to determine the amplitude, carrier phase, and relative propagation delay of both a direct-path and a reflected signal; a multiple-correlator code phase tracking loop then exploits these ML signal estimates to remove the tracking error introduced by the reflection. A preliminary analysis showed that the MRDLL's linear tracking region varied with the reflected signal parameters; therefore, an adaptive loop controller (ALC) was introduced to allow the loop designer to fix dynamic specifications such as loop natural frequency. Analysis and computer simulations demonstrated that, when multipath was present, the MRDLL exhibited a significantly lower steady-state code phase tracking error than that of the standard non-coherent delay-lock loop (NCDLL), which is typically used in GPS receivers. In an ideal multipath-free environment, the NCDLL is still the best choice for code phase tracking.

This GPS receiver design technology will benefit the entire aviation community by eliminating or reducing the dominant source of error in differential GPS-based instrument landing systems, resulting in improved ILS safety, reliability, and integrity. In relative GPS applications, such as precision guided munitions, lethality is improved via elimination of multipath contribution to targeting error. In all GPS applications, the (no longer dominant) error contribution of multipath can be eliminated, yielding enhanced positioning accuracy.

ANALYSIS AND SIMULATION OF A NEW CODE TRACKING LOOP FOR GPS MULTIPATH MITIGATION

I. Introduction

1.1 Overview

The satellite-based NAVSTAR Global Positioning System (GPS) has ushered in a new era in world-wide precision navigation. Both the military and civilian communities use GPS in a variety of applications as users rely on GPS satellite signals to estimate current latitude, longitude, and elevation. The United States Air Force (USAF), for example, relies on GPS receivers to provide accurate position data for many of its aircraft navigation systems.

In the ongoing search for greater navigation accuracy, research efforts continue to focus on understanding and reducing the effects of real-time errors at the GPS receiver. Three significant error sources are satellite and receiver clock biases, atmospheric effects on the GPS signal, and multipath signal reflections. Two of these error sources (clock bias and atmospheric effect) can often be significantly reduced by using differencing techniques such as differential GPS (DGPS). Multipath effects, however, generally cannot be reduced by differencing techniques; therefore, reducing the effects of multipath interference on the GPS receiver has received much attention. This thesis examines a proposed GPS receiver code tracking loop designed to mitigate the effects of multipath interference.

1.2 Background Information

1.2.1 The Multipath Signal. Multipath signals occur when the original GPS line-of-sight (LOS) satellite signal reflects off surrounding objects or the ground plane (Figure 1). At the GPS

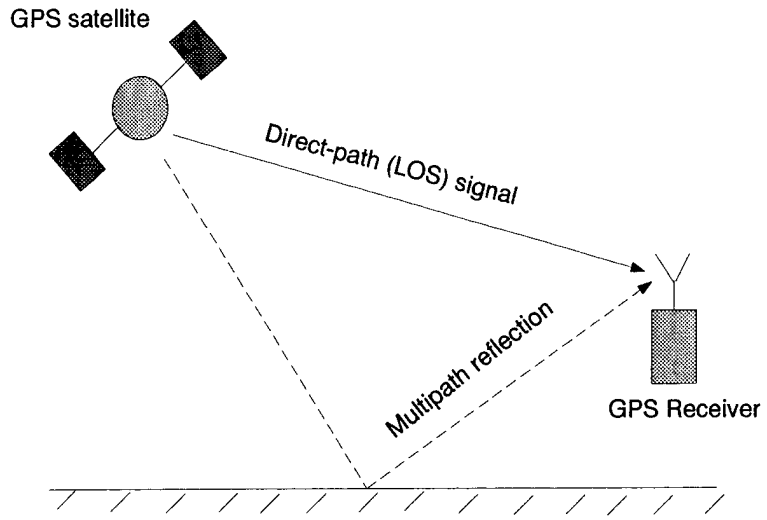

Figure 1. Multipath signal reflection.

receiver, this reflection produces an attenuated replica of the LOS signal which is delayed due to the longer propagation paths.

The general multipath signal can be expressed as the sum of several time-shifted and attenuated versions of the original transmitted signal:

$$\sum_{i=0}^{L} a_i s\left(t-\tau_i\right) \tag{1}$$

where $s(t)$ is the original transmitted signal, L is the number of reflected paths present, a_i is the attenuation coefficient of the ith signal, and τ_i is the propagation delay of the ith signal ($i = 0$ corresponds to the direct path LOS signal).

For a single GPS satellite, the transmitted signal can be represented as

$$s(t) = c(t)m(t)\sqrt{2P}\cos(\omega_c t + \phi) \tag{2}$$

where $c(t)$ is a pseudo-random noise (PN) direct-sequence spread-spectrum (DS/SS) code, $m(t)$ represents binary phase-shift keyed (BPSK) data modulation at 50 bits per second (referred to

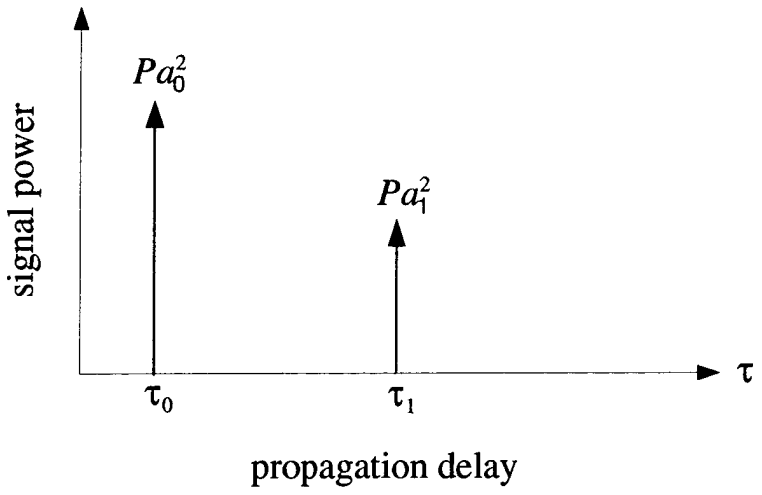

Figure 2. Average power-delay profile for a multipath signal with a single reflection.

as the 'navigation message'), P is the signal power in Watts, and ϕ is the initial transmitted phase of the signal in radians. GPS transmits at two different carrier frequencies, $\omega_c = 2\pi f_c$ (in radians/sec); the first carrier, designated L1, is at $f_c = 1575.42$ MHz while the second carrier, L2, is at $f_c = 1227.6$ MHz (4).

When modeling the attenuation coefficients, a_i, it is important to consider both their distribution and their values relative to each other. There are two scenarios to be considered when modeling the amplitude distribution of the received signal (11):

1. **A single reflection is present**. In this case, $L = 1$ in Equation 1. A typical power-delay profile for a_0 and a_1 is shown in Figure 2. This scenario is valid when there is one dominant reflector present such as a large building or body of water. Often, a_0 and a_1 are assumed to be uniformly distributed between 0 and 1.

2. **Multiple reflections are present**. In this case, $L > 1$ in Equation 1, and the a_i's are distributed according to a Rayleigh distribution. This scenario is valid when many reflectors are present, such as buildings or trees. A typical power-delay profile for the multiple-reflection case is shown in Figure 3

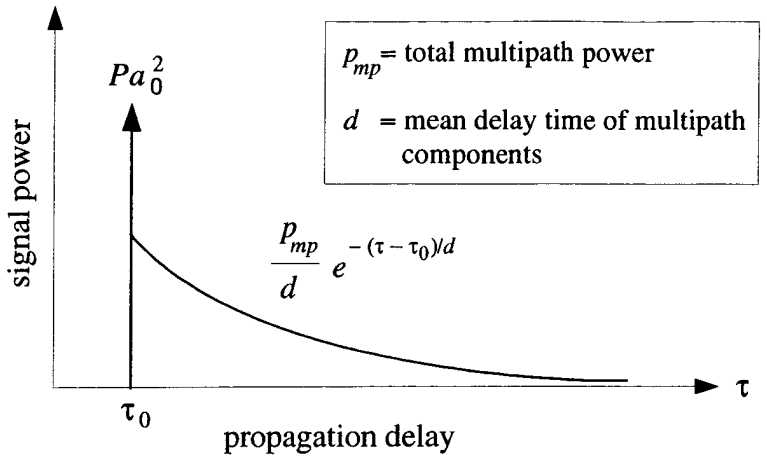

Figure 3. Average power-delay profile for a multipath signal with multiple reflections.

Notice that the amplitudes of the reflected signals are generally less than the direct path signal. According to (11), this condition will occur for the following reasons:

1. **The reflected signal travels a greater distance.** Therefore, it experiences a greater free space loss. However, this attenuation has a minimal effect on GPS receiver ranging errors because, as will be seen, only reflections with a propagation delay just slightly greater than that of the direct-path signal have a significant impact.

2. **The reflector surface causes attenuation.** The amount of this attenuation is dependent upon the material of the reflector and the incident angle of the reflection. For very low incident angles, the attenuation is negligible.

3. **The receiver antenna causes attenuation.** This is due to the antenna's gain pattern and the orthogonal polarization between the received signals. Generally, in the case of GPS, a single reflection has a left-hand circular polarization while the direct path signal has a right-hand polarization. Therefore, the two received signal components are orthogonal to one another.

1.2.2 The GPS Spreading Code. In order to understand the operation of the GPS receiver and the effects of multipath, it is first necessary to examine the properties of the DS/SS code, $c(t)$. This code is a BPSK code generated at a very high frequency relative to the data message rate. To distinguish the spreading code from the data message, each code pulse of is called a 'chip' (as opposed to a 'bit') and the code is said to have a 'chip period' of T_c seconds. Furthermore, the spreading code itself is periodic with period NT_c seconds, where N is the number of chips in a period. The actual values of T_c and N differ depending on whether the user is receiving the lower-rate coarse/acquisition (C/A) code or the higher-rate precision code (P-code). The properties of these two codes are summarized in Table 1 which is based on information presented in (4).

Table 1. GPS spreading code parameters.

Parameter	**C/A code**	**P-code**
chip rate $= 1/T_c$	1.023 MHz	10.23 MHz
code period $= N$	1023 chips; (1ms)	$\approx 6 \times 10^{12}$ chips; (1week)
carrier band	L1	L1, L2

At the transmitter, mixing $c(t)$ with the data-modulated carrier has the effect of 'spreading' the GPS signal's bandwidth and significantly reducing the effective received signal-to-noise ratio (SNR). A receiver can recover the transmitted signal by mixing the received signal with an internally generated and synchronized replica code; this is known as 'despreading' the signal. The replica code is generated by a code tracking loop in the receiver which is responsible for tracking changes in the propagation delay (or code phase) of the incoming DS/SS code.

An important property of any DS/SS code is its autocorrelation function, $R_c(\Omega)$, which is defined as

$$R_c(\Omega) = \frac{1}{NT_c} \int_0^{NT_c} c(t)c(t + \Omega T_c)dt \ . \tag{3}$$

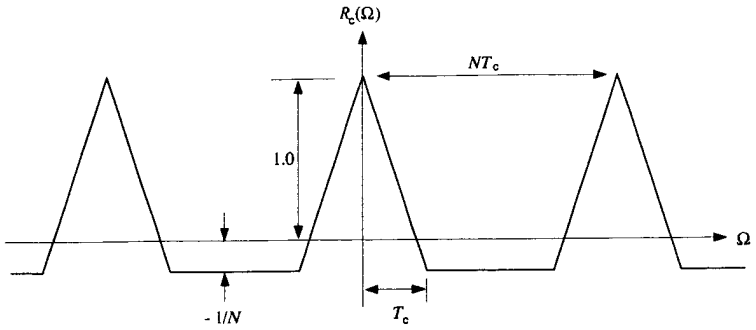

Figure 4. DS/SS spreading code autocorrelation function.

This function is illustrated in Figure 4 for a maximal-length PN code (similar to the Gold codes used by GPS); it is periodic with period N and can be written as (5)

$$R_c(\Omega) = \begin{cases} 1 - \Omega\left(1 + \frac{1}{N}\right) & 0 \le \Omega \le 1 \\ -\frac{1}{N} & 1 < \Omega < (N-1) \\ [\Omega - (N-1)]\left(1 + \frac{1}{N}\right) - \frac{1}{N} & (N-1) \le \Omega < N \end{cases} . \tag{4}$$

As can be seen in Table 1, $N \gg 1$ for the GPS signal. Therefore, the code autocorrelation function can be approximated as

$$R_c(\Omega) \approx \begin{cases} 1 - |\Omega| & |\Omega| \le 1 \\ 0 & \text{elsewhere} \end{cases} . \tag{5}$$

1.2.3 The code tracking loop. As mentioned before in Section 1.2.2, the GPS receiver's code tracking loop tracks the code phase of the incoming DS/SS signal, $c(t)$. Tracking the code phase serves two main purposes:

1. Tracking provides an estimate of the magnitude of time shift required to maximize the correlation between the incoming signal and the receiver's internally generated 'on-time' code;

this information is used by the receiver to calculate an initial user-to-satellite range estimate, known as the *pseudorange* (PR) measurement.

2. The synchronized replica signal is used to despread the GPS signal; this despread signal is then passed to the receiver's carrier tracking loop for demodulation of the data message, $m(t)$. The data message contains satellite information that is needed to enable the receiver to compute its position, velocity, and clock bias.

The typical GPS receiver uses the standard non-coherent delay-lock-loop (NCDLL) as its code tracking loop. This loop is often called the 'early-late' delay-lock loop, referring to the advanced and delayed versions of the code replica generated by the voltage-controlled clock (VCC). Furthermore, the relative delay spacing, ΔT_c, between the two generated codes is called the 'early-late spacing' (typically, $\Delta = 1.0$). The the theory of operation and an analysis of the NCDLL will be presented in Chapter II.

1.2.4 Effects of multipath on the code tracking loop. The code tracking loop's mission is to accurately track the code phase of the direct path LOS signal. However, when multipath signals arrive at the GPS receiver, they introduce a phase tracking error in the NCDLL. The significance of these tracking errors is larger than one might initially think. For instance, for C/A code, a code phase tracking error, ϵ, of one-tenth of a chip period, T_c, gives:

$$\epsilon = 0.1T_c \approx 98 \text{ ns}, \tag{6}$$

which may seem rather small. However, representing this tracking error in terms of distance (ϵ_d) by multiplying by the speed of light, c, gives

$$\epsilon_d = c\epsilon \approx \left(3 \times 10^8 \text{ m/s}\right)\left(9.8 \times 10^8 \text{ s}\right) = 29.4 \text{ m}. \tag{7}$$

Therefore, a code tracking error of only 98 ns corresponds to a range measurement error of almost 30 m! It has been shown (11) that multipath tracking errors are significant only when the relative delay between the multipath and direct-path signals is less than $1.5T_c$; this is due to the inherent multipath resistance of DS/SS spreading codes and is based on properties of the code autocorrelation function defined in Equation 3. The effects of multipath on the NCDLL's code phase tracking performance is examined in detail in Chapter II.

1.2.5 Proposed Mitigation Techniques. A significant amount of research has gone into the mitigation of GPS multipath effects. This section looks at three mitigation techniques that have been proposed by various authors:

1. Narrow correlator spacing
2. Minimum mean-square error (MMSE) estimation
3. Multiple-correlator DLL

1.2.5.1 Narrow Correlator Spacing. It was shown by Van Dierendonck, Fenton, and Ford that reducing the early-late spacing in the NCDLL could reduce GPS multipath errors under certain conditions (10). Experiments performed with a C/A code GPS receiver demonstrated that reducing the standard 1.0-chip spacing to 0.1-chip spacing reduced the tracking error caused by multipath interference. However, the authors in (10) admit that the problem with the narrow correlator spacing technique is that it currently can only be used with C/A code, which is intended primarily for civilian use. The NCDLL with narrow spacing cannot accurately track the higher-frequency P-code, which is used in important military applications.

1.2.5.2 MMSE Estimation. A second multipath mitigation approach proposed replacing the NCDLL altogether with an alternative direct-path delay estimator. This alternative estimator was derived by Weill as the minimum mean-square error (MMSE) estimator for direct

path delay (14). It had already been shown that the NCDLL approached the performance of the maximum-likelihood estimator (MLE) for one-path signals (13). However, while the MLE was optimum for the one-path case, Weill showed in (14) that the MMSE estimator performed better in the presence of multipath because no other estimator exhibited a lower root mean-square (rms) estimation error; the only exception to this occurred when multipath/LOS path separations were extremely small (approximately 1-2 meters). The main disadvantage of the MMSE estimator lies in implementation. It is currently difficult to implement the true MMSE estimator for direct-path delay, because the MMSE requires two quadruple integrations which are very computationally intensive.

1.2.5.3 Multiple-correlator estimation. A third multipath mitigation scheme is the multiple correlator estimation (MCE). As illustrated in Figure 5, the received signal is correlated with many delayed code replicas ($c(t - \widehat{\tau} - \beta_k T_c)$ in Figure 5) provided by the code tracking loop. As will be seen in Chapter III, this effectively 'samples' the code autocorrelation function, $R_c(\Omega)$, at $\Omega = \beta_k$; the resulting correlator outputs, R_k, are sampled and digitally processed to estimate the gain and phase of the direct-path and reflected signal components. These estimates are then used by the code tracking loop to reduce any code tracking errors introduced by multipath interference. One example of the MCE approach is the Multipath Estimating Delay Lock Loop (MEDLL). The MEDLL uses MCE to remove the estimated multipath signal components and feed the estimated direct-path signal to a standard NCDLL (9).

The MCE approach was also used by Sheen and Stuber in their proposed RAKE delay-lock loop (RDLL) (7) (8). This approach differed from the MEDLL in that the NCDLL was not used; instead, the code tracking loop was completely redesigned. The RDLL featured a relatively simple multiple-correlator loop design that demonstrated a significant improvement in tracking error performance over the standard NCDLL in the presence of multi-user interference. However, unlike

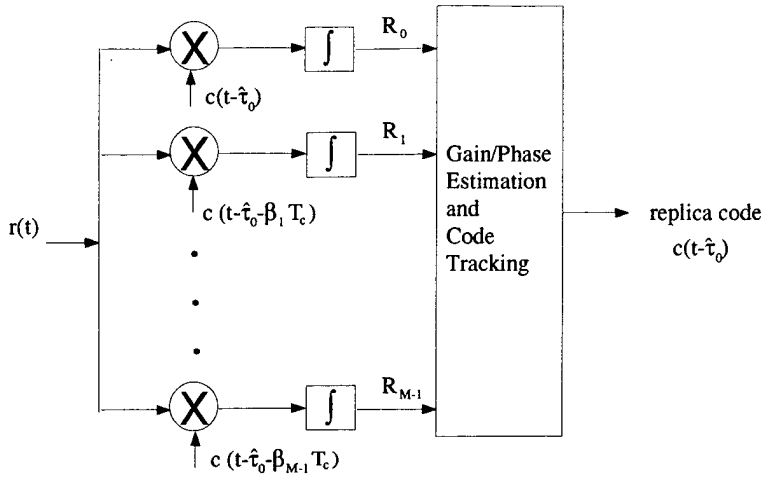

Figure 5. Multiple-correlator estimation.

the MEDLL, the RDLL was not designed for GPS applications, but for code-division-multiple-access (CDMA) mobile radio communications.

1.3 Problem Statement

The RDLL was designed to operate in a multiple-user CDMA environment with relative delay spacings that were *integer* multiples of the chip period, T_c. As stated in Section 1.2.4, GPS code tracking loop errors due to multipath occur only for multipath delay spacings that are very small (i.e., *fractional* multiples of T_c). Therefore, for the RDLL to be effective in GPS multipath environments, it must be modified to mitigate the effects of multipath signals with small delays relative to the direct-path signal.

1.4 Thesis Objectives

This thesis proposes a GPS receiver code tracking loop, the modified RAKE delay-lock loop (MRDLL), designed to mitigate the effects of multipath interference. This new tracking loop is a modified version of the RDLL introduced by Sheen and Stuber. The proposed loop includes a

multiple-correlator estimation unit (MCEU) and introduces an original adaptive loop controller (ALC) not present in the RDLL.

The main objectives of this research were to:

1. Verify the effects of multipath on NCDLL code tracking performance.
2. Adapt the existing RDLL design for solution of the GPS multipath problem.
3. Design an original multipath gain/phase estimation scheme for use with the MRDLL.
4. Compare predicted and simulated performance of the MRDLL to that of the NCDLL in the presence of multipath.
5. Investigate performance of the MRDLL in the presence of additive white Gaussian noise (AWGN).

A secondary research objective was to design computer simulation models for each of the code tracking loops being evaluated; such models will be useful for any follow-on research.

1.5 Assumptions

For this thesis, the following assumptions were made:

1. The received GPS signal consists of a direct-path component and one reflected signal (i.e., two-path case).
2. The code tracking loop is already tracking the received code prior to the introduction of the multipath signal (i.e., the acquisition phase is complete).
3. The received GPS signal has been down-converted from the GPS carrier frequency to the intermediate frequency (IF).
4. The received GPS signal corresponds to only one GPS satellite (i.e., one satellite per receiver channel).

5. Multipath mitigation is achieved through signal processing alone; no special antennas or spatial processing techniques are used.

1.6 Approach

This thesis presents results based on the theoretical analysis and computer simulation of the NCDLL and MRDLL code tracking loop designs. Analyses and simulations were performed for a variety of GPS direct-path, multipath, and additive white Gaussian noise (AWGN) environments.

All computer simulations are described in Chapter IV. The majority of the code loop simulation was performed using the Signal Processing Workstation (SPW®) software from Comdisco Systems, Inc. of Foster City, California. Some preliminary simulation required using the MATLAB® computational software from The MathWorks, Inc. of Natick, Massachusetts. Simulations were performed on the Sun Workstations provided at the Air Force Institute of Technology (AFIT).

II. NCDLL Analysis

2.1 Overview

This chapter presents an analysis of the non-coherent delay-lock loop (NCDLL). The first section investigates the operation of the NCDLL when no multipath is present; this investigation leads to the development of the NCDLL linear model. The second section examines the effects of multipath on the NCDLL with an emphasis on code phase tracking error performance.

A block diagram of the NCDLL is shown in Figure 6. In the analysis that follows, the following assumptions are made.

1. The spreading code *self-noise* is negligible. Code self-noise is defined as the time varying component of the NCDLL discriminator output, $\epsilon(t, \delta)$. Generally, almost all of the self-noise power is at frequencies which are well outside the NCDLL loop bandwidth (5).

2. System *processing gain* , G, is high. Processing gain is defined as

$$G = \frac{W_c}{W_d} , \tag{8}$$

which is the ratio between the GPS spreading code chip rate, $W_c = 1/T_c$, and the data modulation bit rate, W_d. For all spread-spectrum systems like GPS, $W_c \gg W_d$; the processing gain for the C/A code, for example, is calculated as

$$G_{c/a} = \frac{(1/T_c)_{c/a}}{W_d} = \frac{1.023 \times 10^6 \ Hz}{50 \ Hz} \approx 2 \times 10^4 , \tag{9}$$

and it can be seen that the processing gain for the P-code is even greater.

3. The bandpass filters (BPFs) in Figure 6 are assumed to be ideal 'brick-wall' filters centered at IF with a single-sided noise bandwidth of $B_{IF} \ll W_c$ Hz. This assumption gives a filter

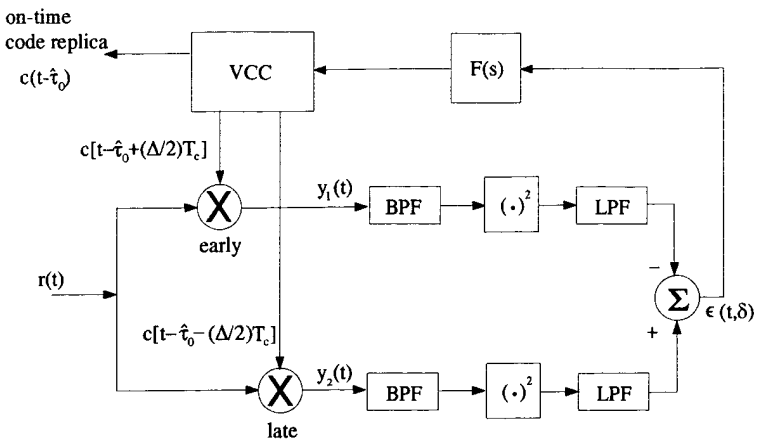

Figure 6. Non-coherent delay lock loop block diagram.

transfer function, $H_{IF}(f)$, with magnitude

$$|H_{IF}(f)| = \begin{cases} 1 & \text{for } |f \pm f_c| \leq \left|\frac{B_{IF}}{2}\right| \\ 0 & \text{elsewhere} \end{cases} . \tag{10}$$

Similarly, the lowpass filters (LPFs) are assumed to be ideal with two-sided noise bandwidth, $B_{LPF} \ll W_c$ Hz and transfer function

$$|H_{LPF}(f)| = \begin{cases} 1 & \text{for } |f| \leq \left|\frac{B_{LPF}}{2}\right| \\ 0 & \text{elsewhere} \end{cases} .$$

4. The NCDLL early-late chip offset has a value $\Delta = 1.0$. This is the typical value used in the NCDLL.

5. The BPSK data modulation on the received signal is ignored. This assumption is made because the NCDLL's squaring circuits remove the effects of phase modulation. Also, it can be shown that removing the data modulation results in the maximum possible noise power at the NCDLL phase discriminator output (5). Therefore, ignoring data modulation provides a worst-case scenario for the loop noise analysis.

6. The period, N, of the spreading code is large. Therefore, the code autocorrelation function, $R_c(\Omega)$, is well-approximated by Equation 5 in Chapter I.

7. Unless otherwise stated, code Doppler shift is ignored.

2.2 NCDLL analysis with no multipath

2.2.1 The received signal. The received direct-path GPS signal, $r_{dp}(t)$, is a modulated carrier in bandlimited AWGN; the received signal (neglecting data modulation) is given as

$$r_{dp}(t) = \sqrt{2P}a_0 c(t-\tau_0)\cos(2\pi f_c t + \theta_0) + n(t) \tag{11}$$

where P is the transmitted signal power in Watts, a_0 is the direct-path attenuation coefficient, $c(t)$ is the DS/SS BPSK spreading code, τ_0 is the direct-path signal propagation delay in seconds, ω_c is the received carrier frequency in rad/sec, and θ_0 is the received carrier phase in radians given by $\theta_0 = -\omega_c\tau_0$.

The noise, $n(t)$, is assumed to be band-limited zero-mean AWGN with a two-sided power spectral density (PSD) of $N_0/2$ W/Hz, and is represented as

$$n(t) = \sqrt{2}n_I(t)\cos\omega_c t - \sqrt{2}n_Q(t)\sin\omega_c t\ . \tag{12}$$

Using the representation in Equation 12, the in-phase and quadrature components of the noise, ($n_I(t)$ and $n_Q(t)$, respectively) are independent zero-mean lowpass white Gaussian noise processes each having a two-sided PSD of $N_0/2$ W/Hz.

2.2.2 The early-late correlator outputs. Next, consider the outputs of the early/late correlators, $y_1(t)$ and $y_2(t)$. Under our previously stated assumptions, it can be shown that the

signal components of interest are (5)

$$\begin{aligned} y_1(t) &= \sqrt{2P}a_0 R_c \left(\delta + \frac{1}{2}\right) \cos\left(\omega_c t + \theta_0\right) + n_{y1}(t) \\ y_2(t) &= \sqrt{2P}a_0 R_c \left(\delta - \frac{1}{2}\right) \cos\left(\omega_c t + \theta_0\right) + n_{y2}(t) \end{aligned} \tag{13}$$

where δ is the normalized code phase estimation error defined as

$$\delta(t) \triangleq \frac{\tau_0(t) - \widehat{\tau}_0(t)}{T_c} \tag{14}$$

and $\widehat{\tau}_0(t)$ is the NCDLL's estimate of the direct-path propagation delay. From this point forward, the implicit time dependence of δ will be dropped to simplify notation.

The correlator output noises, $n_{y1}(t)$ and $n_{y2}(t)$, are given by

$$\begin{aligned} n_{y1}(t) &= c\left(t - \widehat{\tau}_0 + \frac{1}{2}T_c\right) n(t) \\ n_{y2}(t) &= c\left(t - \widehat{\tau}_0 - \frac{1}{2}T_c\right) n(t) \end{aligned} \tag{15}$$

where $n(t)$ is defined in Equation 12. Both $n_{y1}(t)$ and $n_{y2}(t)$ are Gaussian since $n(t)$ is zero-mean Gaussian and $c(t)$ equal $+1$ or -1 with equal probability; however, these correlator output noises are obviously not independent (5).

2.2.3 Discriminator output. To determine the NCDLL discriminator output, $\epsilon(t, \delta)$, first consider the signal component (neglecting noise). This signal component is obtained by squaring the signal components of the correlator outputs, taking their difference, and taking the lowpass component to give

$$\epsilon(t, \delta)_{sig} = \left[y_2^2(t) - y_1^2(t)\right]_{LP} = P a_0^2 S(\delta) \tag{16}$$

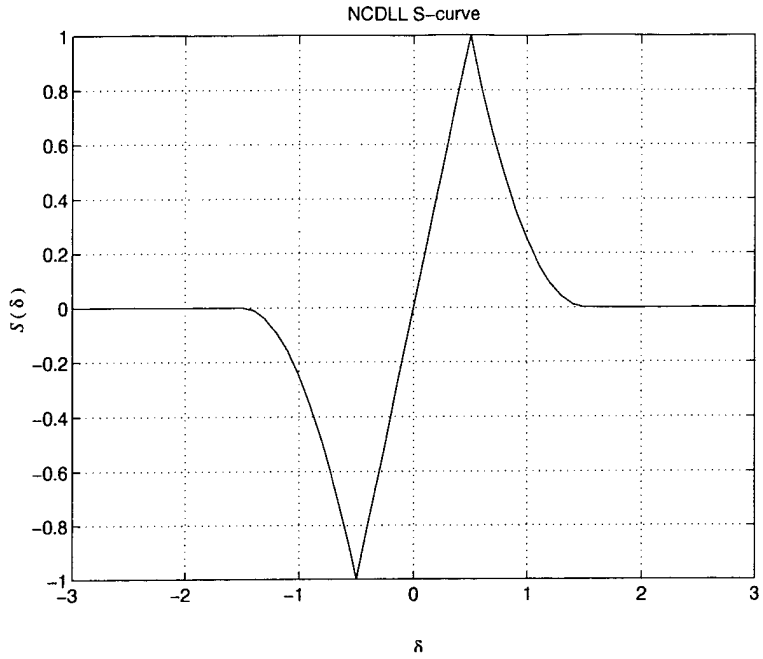

Figure 7. NCDLL S-curve.

where $S(\delta)$ is referred to as the NCDLL 'S-curve' and is defined as

$$S(\delta) \triangleq R_c^2\left(\delta - \frac{1}{2}\right) - R_c^2\left(\delta + \frac{1}{2}\right) . \tag{17}$$

This curve is plotted in Figure 7. Note that there is a linear region around $\delta = 0$. This region is always chosen as the operating region of the NCDLL, and the loop will tend to operate at the point where $S(\delta) = 0$ and the slope is positive (5). Figure 7 shows that this operating point corresponds to zero steady-state code phase tracking error (i.e., $\delta_{ss} = 0$).

Next, consider the noise component, $n_\epsilon(t)$, of the discriminator output. Because the loop filter following the discriminator has a bandwidth much smaller than that of $n_\epsilon(t)$, only the lowpass component is of interest. This lowpass noise component is assumed to be AWGN and can be shown

to have a two-sided PSD, $G_{n\epsilon}(f)$, given by (5)

$$G_{n\epsilon}(f) \approx G_{n\epsilon}(f)|_{f=0} = 2N_0^2 B_{IF} + 2Pa_0^2 N_0 \left[R_c^2 \left(\delta - \frac{1}{2} \right) + R_c^2 \left(\delta + \frac{1}{2} \right) \right] \tag{18}$$

which is defined over the loop bandwidth and is in units of W/Hz. The combined signal-plus-noise expression for the NCDLL discriminator output is

$$\epsilon(t, \delta) = Pa_0^2 S(\delta) + n_\epsilon(t) \ . \tag{19}$$

2.2.4 NCDLL linear model. The analysis in the preceding paragraphs allows us to form a linear model for the NCDLL. This linear model is valid for small tracking errors ($\delta \approx 0$) and is formed by first considering the operation of the voltage-controlled clock (VCC) described by

$$\frac{\widehat{\tau}_0(t)}{T_c} = g_c \int_0^t \epsilon(\lambda, \delta) * f(\lambda) \, d\lambda \tag{20}$$

where $*$ denotes convolution, g_c is the VCC constant in Hz/volt, and $f(t)$ is the impulse response of the NCDLL loop filter. For small tracking errors and large code period, N, the linear region of the S-curve can be expressed via Equations 17 and 5 as

$$S(\delta) = 2\delta = 2 \left(\frac{\tau_0 - \widehat{\tau}_0}{T_c} \right) \ . \tag{21}$$

Therefore, applying Equation 21 to Equation 19, Equation 20 can be rewritten as

$$\frac{\widehat{\tau}_0(t)}{T_c} = K_{nc} g_c \int_0^t \left[\left(\frac{\tau_0(\lambda) - \widehat{\tau}_0(\lambda)}{T_c} \right) + \frac{n_\epsilon(\lambda)}{K_{nc}} \right] * f(\lambda) \, d\lambda \tag{22}$$

where $K_{nc} \triangleq 2Pa_0^2$.

Equation 22 represents a NCDLL linear equivalent circuit and is shown in Figure 8 with corresponding Laplace transform shown in Figure 9.

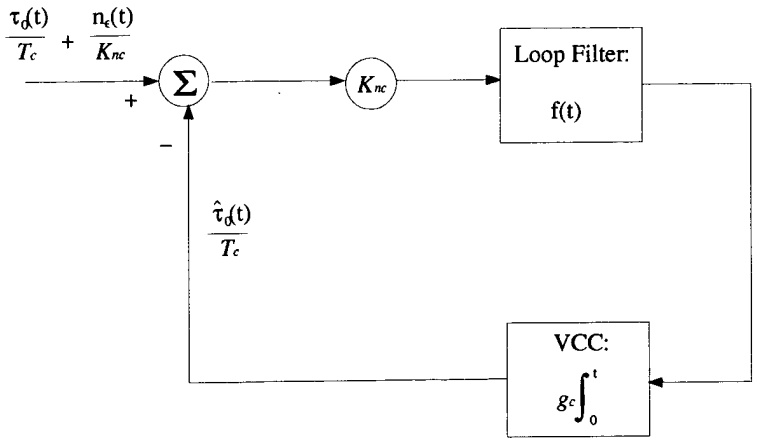

Figure 8. NCDLL linear equivalent circuit.

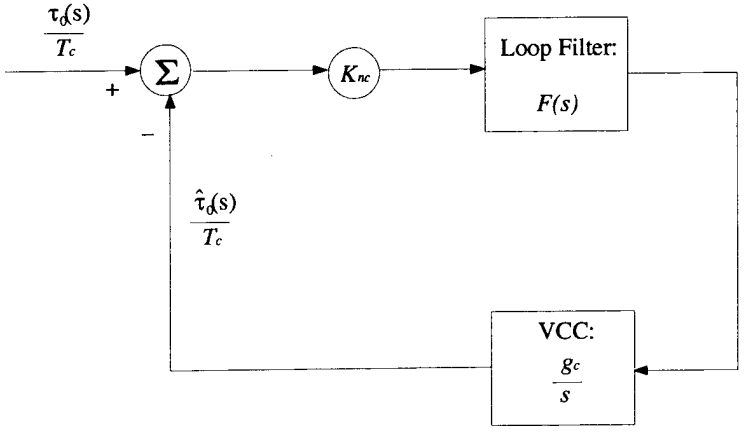

Figure 9. Linearized NCDLL Laplace transform.

2.2.5 Loop dynamics and tracking performance. Examination of Figure 9 allows us to write the NCDLL closed-loop transfer function as

$$H(s) \triangleq \frac{\widehat{\tau_0}(s)}{\tau_0(s)} = \frac{K_{nc} g_c F(s)}{s + K_{nc} g_c F(s)} \tag{23}$$

In this equation, $F(s)$ is the s-domain transfer function of the loop filter, which is typically chosen to be an active lead-lag filter of the form

$$F(s) = \frac{1 + \tau_2 s}{\tau_1 s} \ . \tag{24}$$

The presence of the single power of s in the denominator of 24 allows the loop filter to act as an integrator. In this case, the NCDLL is called a *type 2* loop because it contains two integrators (the VCC being the second integrator). The advantage of a type 2 loop is that it can track a signal in the presence of Doppler shift (i.e., the introduction of a phase ramp) without incurring a steady-state phase error (2).

The closed-loop transfer function can now be rewritten as (2)

$$H(s) = \frac{2\zeta\omega_n s + \omega_n^2}{s^2 + 2\zeta\omega_n s + \omega_n^2} \ , \tag{25}$$

where ω_n is the loop natural frequency in rad/sec given as

$$\omega_n = \sqrt{\frac{K_{nc} g_c}{\tau_1}} \ , \tag{26}$$

and ζ is the loop damping factor given as

$$\zeta = \frac{\tau_2}{2}\omega_n \ . \tag{27}$$

The single-sided noise equivalent bandwidth of the loop, B_L (in Hz), can be shown to be

$$\begin{aligned} B_L &\triangleq \frac{1}{2}\int_{-\infty}^{\infty} |H(j2\pi f)|^2 \, df \\ &= \frac{1}{2}\omega_n \left(\zeta + \frac{1}{4\zeta}\right) \ . \end{aligned} \tag{28}$$

The natural frequency and damping ratio determine tracking performance of the NCDLL in the presence of signal dynamics such as code Doppler shifts; the higher the natural frequency, the faster the dynamic response. The noise bandwidth determines the amount of noise power at the loop input; the smaller the bandwidth, the greater the loop signal-to-noise ratio (SNR). Equations 26 to 28 demonstrate the fundamental tradeoff in tracking loop design: better dynamic response is achieved only at the expense of increased noise sensitivity.

Another important NCDLL performance parameter is the variance of the steady-state tracking error, σ_δ^2, also called *tracking jitter*. The power spectrum of the tracking jitter, $G_\delta(f)$, is given by (2)

$$G_\delta(f) = |H(j2\pi f)|^2 \, G_{n\epsilon'}(f) \tag{29}$$

where $G_{n\epsilon'}(f) = G_{n\epsilon}(f)/K_{nc}^2$ is the noise PSD at the input of the NCDLL linear model in Figure 8. This noise PSD is approximately flat over the relatively small loop bandwidth; therefore, the tracking jitter can be written as

$$\begin{aligned} \sigma_\delta^2 &= \int_{-\infty}^{\infty} G_\delta(f) df \\ &= \int_{-\infty}^{\infty} G_{n\epsilon'}(f) \, |H(j2\pi f)|^2 \, df \\ &= \frac{G_{n\epsilon}(0)}{K_{nc}^2} B_L \end{aligned} \tag{30}$$

where $G_{n\epsilon}(0)$ is given in Equation 18.

2.3 NCDLL operation in the presence of multipath

2.3.1 Additional assumptions. This section investigates the operation of the NCDLL in the presence of multipath interference. Particular emphasis is placed on the effects of the multipath signal on NCDLL steady-state tracking performance. In addition to the assumptions stated at the beginning of the chapter, two additional assumptions are necessary:

1. The purpose of this analysis is to determine the effects of multipath interference on NCDLL operation. Therefore, in order to isolate multipath effects, received signal noise is neglected.
2. The received multipath signal consists of a direct-path component with a single reflection.

2.3.2 The received multipath signal. The received multipath signal (neglecting data modulation) is given by

$$r_{mp}(t) = \sqrt{2P}\sum_{i=0}^{1} a_i c\left(t - \tau_0 - \alpha_i T_c\right)\cos\left(\omega_c t + \theta_i\right) + n(t) \tag{31}$$

where a_i is an attenuation coefficient and θ_i is the received signal carrier phase in radians such that $\theta_i = -\omega_c\left(\tau_0 + \alpha_i T_c\right)$. The parameter, α_i , is the delay coefficient which is defined such that $\alpha_0 = 0$ and α_1 is any real number greater than zero; from this point forward, let $\alpha_1 = \alpha$. We are concerned only with $0 < \alpha < 1.5$ because relative multipath delays outside this region have little effect on GPS code phase tracking (11). As before, the received noise, $n(t)$, is modeled as bandpass, zero-mean, AWGN according to Equation 12.

2.3.3 NCDLL discriminator output in the presence of multipath. Neglecting noise and applying the signal component of Equation 31 to the NCDLL input, the correlator outputs now become

$$\begin{aligned} y_1(t) &= \sqrt{2P}\left\{a_0 R_c\left(\delta + \frac{1}{2}\right)\cos\left(\omega_c t + \theta_0\right) + a_1 R_c\left(\delta + \alpha + \frac{1}{2}\right)\cos\left(\omega_c t + \theta_1\right)\right\} \\ y_2(t) &= \sqrt{2P}\left\{a_0 R_c\left(\delta - \frac{1}{2}\right)\cos\left(\omega_c t + \theta_0\right) + a_1 R_c\left(\delta + \alpha - \frac{1}{2}\right)\cos\left(\omega_c t + \theta_1\right)\right\} . \end{aligned}$$

Therefore, the discriminator output is

$$\begin{aligned} \epsilon\left(t, \delta\right) &= \left[y_2^2(t) - y_1^2(t)\right]_{LP} \\ &= P\left\{a_0^2 S\left(\delta\right) + a_1^2 S(\delta + \alpha) + 2a_0 a_1 \cos\left(\alpha\omega_c T_c\right) S_{cross}(\delta, \alpha)\right\} \end{aligned} \tag{32}$$

where $S_{cross}(\delta, \alpha)$ results from the presence of cross-terms and is given by

$$S_{cross}(\delta, \alpha) = R_c\left(\delta - \frac{1}{2}\right) R_c\left(\delta + \alpha - \frac{1}{2}\right) - R_c\left(\delta + \frac{1}{2}\right) R_c\left(\delta + \alpha + \frac{1}{2}\right) . \tag{33}$$

2.3.4 Effect of multipath on NCDLL tracking performance. Equation 32 is illustrated in Figure 10. Note that the NCDLL discriminator output no longer consists of a single S-curve centered about $\delta = 0$. Instead, the output is now the summation of three different S-curves, each attenuated by a different amount and shifted by a different amount along the δ-axis. The net result, shown in Figure 11, is a tracking point that is no longer at $\delta = 0$; this results in a steady-state code phase tracking error that depends on the signal parameters, a_0, a_1, θ_0, θ_1, and α. Figure 12 shows the predicted steady-state tracking error for different values of α when $a_0 = 1$, $a_1 = 0.5$, and the multipath component is in carrier phase with the direct-path component (i.e., $\theta_0 - \theta_1 = \alpha 2\pi f_c T_c = 2n\pi$ for n an integer). This plot was obtained by setting Equation 32 equal to zero for a given α and solving for δ.

Let's examine the effect of the phase difference, $\theta_0 - \theta_1 = \alpha 2\pi f_c T_c$, on the NCDLL multipath tracking error. It can be seen that $\cos(\alpha 2\pi f_c T_c)$ in the cross-term of Equation 32 is periodic in α with period $n/f_c T_c$ (n an integer). In (12), Van Nee states that, for a given set of attenuation coefficients, a_0 and a_1, NCDLL code phase errors are maximum when the reflected signal is in carrier phase or 180 degrees out of phase with the direct-path signal. To demonstrate this, Figure 13 plots the steady-state tracking error and the phase difference versus α over approximately one period of $\cos(\theta_0 - \theta_1)$ for $a_0 = 1$, $a_1 = 0.5$, and $f_c T_c = 10$; note that the peaks corresponding to maximum phase error occur when $\theta_0 - \theta_1 = 0$ and $\theta_0 - \theta_1 = \pm\pi$.

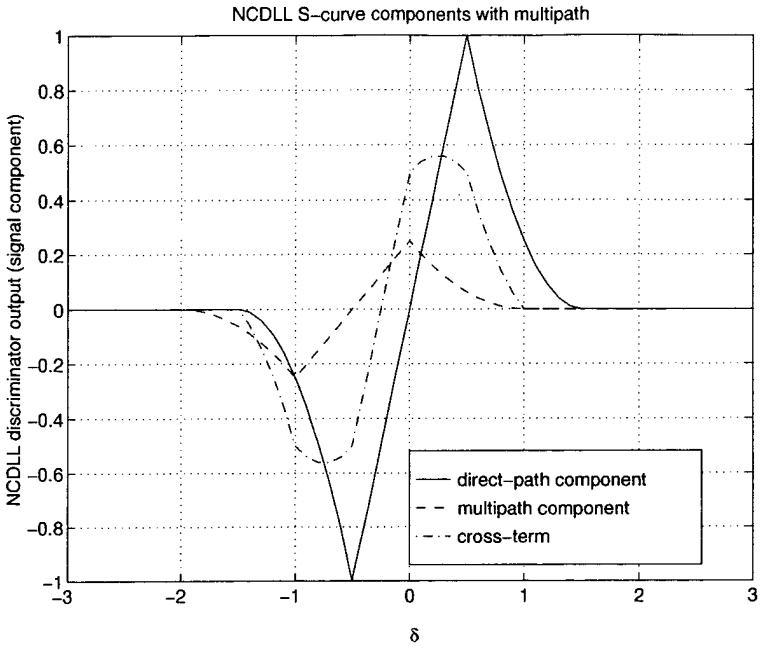

Figure 10. Typical NCDLL discriminator output components in the presence of multipath.

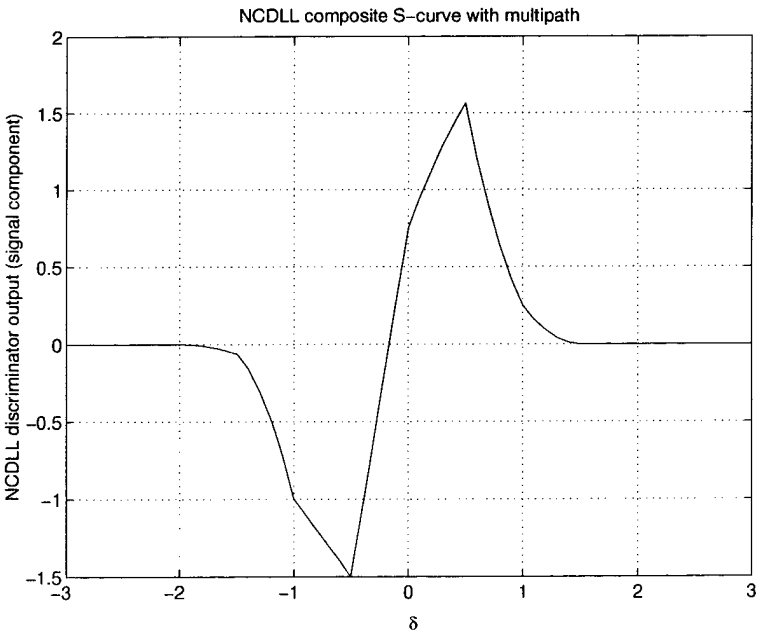

Figure 11. Typical NCDLL discriminator output in the presence of multipath.

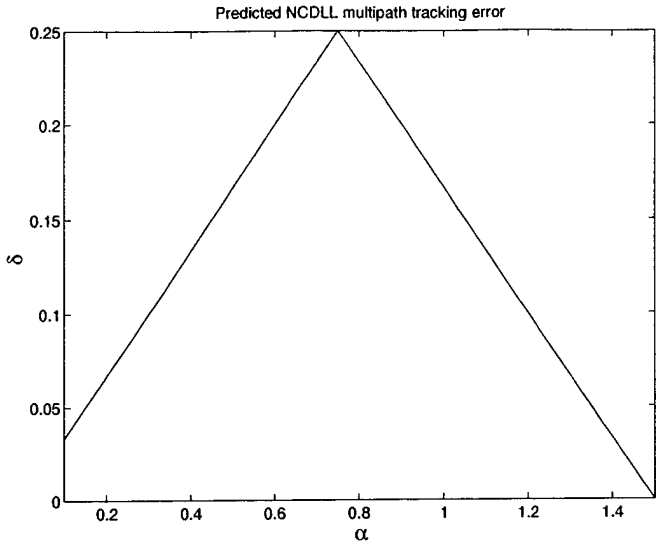

Figure 12. Predicted NCDLL multipath tracking error for $a_0 = 1$, $a_1 = 0.5$, and $\theta_0 - \theta_1 = 2n\pi$ (for n an integer).

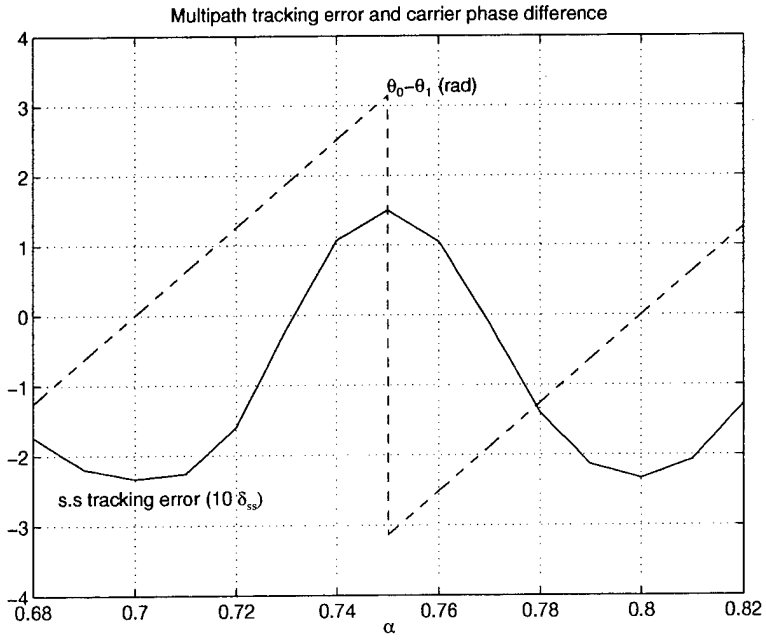

Figure 13. Relationship between NCDLL steady-state tracking error and multipath carrier phase difference for $a_0 = 1$, $a_1 = 0.5$, and $f_c T_c = 10$.

III. MRDLL analysis

3.1 Overview

This chapter investigates the operation of the modified RAKE delay-lock loop (MRDLL). The first section describes the received GPS signal and its conversion to baseband as it enters the MRDLL. The last three sections describe the operation of the MRDLL's three main components: the multiple-correlator tracking loop (MCTL), the adaptive loop controller (ALC), and the multiple-correlator estimation unit (MCEU). In the analysis that follows, the assumptions presented at the beginning of Chapter II still apply.

3.2 Received signal model

The MRDLL block diagram is shown in Figure 14. As in Chapter II, the received GPS multipath signal is modeled as having a direct-path component and a single reflection according to Equation 31. After conversion to baseband as depicted in Figure 14, the signal into the MRDLL becomes

$$r(t) = x_0 c(t - \tau_0) + x_1 c(t - \tau_0 - \alpha T_c) + n'(t) \tag{34}$$

where the coefficients, x_0 and x_1, are defined as (θ_3 is the locally-generated carrier phase)

$$\begin{aligned} x_0 &= \sqrt{2P} a_0 \cos(\theta_0 - \theta_3) \\ x_1 &= \sqrt{2P} a_1 \cos(\theta_1 - \theta_3) \ , \end{aligned} \tag{35}$$

and $n'(t)$ is a lowpass, zero-mean, AWGN process given as (see Equation 12)

$$n'(t) = \sqrt{2}\left[n_I(t)\cos(\theta_3) + n_Q(t)\sin(\theta_3)\right] \tag{36}$$

which has a two-sided PSD of N_0 W/Hz over the LPF bandwidth, B_{LPF} Hz.

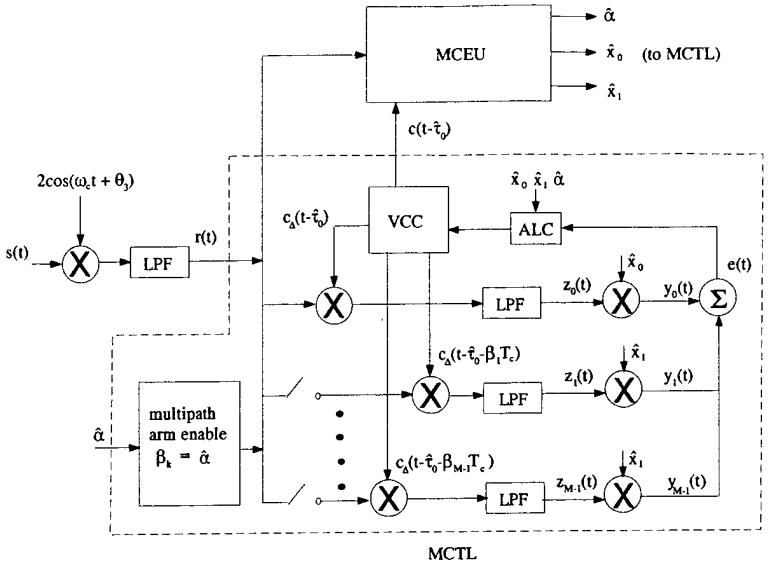

Figure 14. Modified RAKE delay lock loop (MRDLL) block diagram.

As shown in Figure 15, the local carrier is generated using a code-correlator and phase-locked loop (PLL) in combination (4). To derive an expression for the local carrier phase, θ_3, we assume negligible code phase tracking error (i.e., $\widehat{\tau}_0 = \tau_0$) and neglect the effects of noise on the PLL (i.e., $\theta_3 = \theta_0$ when no multipath is present). The signal, $r'(t)$, into the PLL can be written as

$$r'(t) = \sqrt{2P}a_0 \cos(\omega_c t + \theta_0) + \sqrt{2P}a_1 R_c(\alpha) \cos(\omega_c t + \theta_1) \tag{37}$$

where $\theta_0 - \theta_1 = 2\pi f_c \alpha T_c$. Using trigonometry, Equation 37 can be rewritten as

$$r'(t) = C \cos(\omega_c t + \theta_0 + \phi_e) \tag{38}$$

where the phase error, ϕ_e, due to the reflected signal is a function of α and a_0/a_1 and is described by

$$\phi_e = \tan^{-1}\left[\frac{-R_c(\alpha)\sin(\theta_0 - \theta_1)}{a_0/a_1 + R_c(\alpha)\cos(\theta_0 - \theta_1)}\right] = \tan^{-1}\left[\frac{-R_c(\alpha)\sin(2\pi f_c \alpha T_c)}{a_0/a_1 + R_c(\alpha)\cos(2\pi f_c \alpha T_c)}\right]. \tag{39}$$

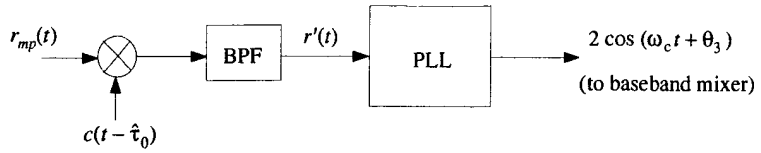

Figure 15. Typical carrier phase recovery scheme for GPS.

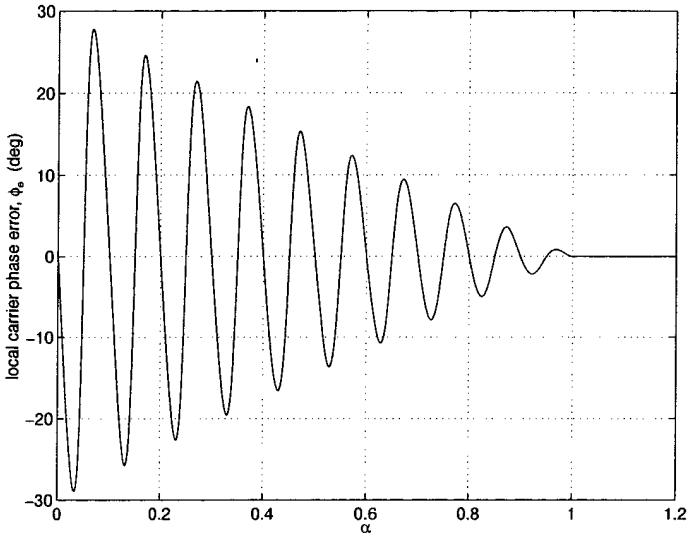

Figure 16. Local carrier phase error, ϕ_e, for $a_0/a_1 = 2$ and $f_c T_c = 10$.

Therefore, the phase tracked by the PLL (and hence, the phase of the locally-generated carrier) is $\theta_3 \triangleq \theta_0 + \phi_e$.

Figure 16 plots ϕ_e versus α for $f_c T_c = 10$ and $a_0/a_1 = 2$; note that the local carrier is in phase with the direct-path carrier (i.e., $\phi_e = 0$) whenever α takes on values for which the reflected signal is in phase or 180 degrees out of phase with the direct path component. Also, note from Figure 16 that the phase error envelope follows the code autocorrelation function, $R_c(\alpha)$, and the local carrier phase error is zero for all $\alpha \geq 1$; this is due to the fact that the code correlation preceding the PLL effectively removes the reflected signal component (i.e., $R_c(\alpha) = 0$ for $\alpha \geq 1$).

To see the effect of ϕ_e on the signal parameters, x_0 and x_1, we can use trigonometric identities and Equation 35 to write the ratio, x_0/x_1 as

$$\frac{x_0}{x_1} = \frac{a_0}{a_1}\rho \tag{40}$$

where

$$\rho \triangleq \left[\frac{1}{\cos(\omega_c \alpha T_c) - \sin(\omega_c \alpha T_c)\tan\phi_e}\right] . \tag{41}$$

Note that for zero phase error, ϕ_e, the ratio in Equation 40 reduces to $\pm a_0/a_1$ depending on whether the reflected signal is in-phase (i.e, $2\pi f_c \alpha T_c = 2n\pi$) or out of phase (i.e., $2\pi f_c \alpha T_c = (2n+1)\pi$) with the direct-path signal for n an integer.

3.3 MCTL operation

3.3.1 Overview. The MCTL is responsible for tracking the direct-path code phase of the incoming multipath DS/SS code. Like the NCDLL, the MCTL uses early-late code correlation (without the squaring operation) to perform tracking. However, unlike the NCDLL, the MCTL relies on signal parameter estimates to accurately track the code phase in the presence of multipath. Therefore, the MCEU has been designed to provide the MCTL with maximum-likelihood (ML) estimates of the multipath delay coefficient, $\widehat{\alpha}$, and of the signal parameters $\widehat{x}_0$ and $\widehat{x}_1$. As will be seen later, these estimates enable the MCTL to remove the tracking error introduced by the reflected signal.

As shown in Figure 14, the MCTL consists of M correlator arms. In the kth arm of the MCTL, the received signal is correlated with shifted early and late replicas of the DS/SS code. For an early-late spacing of one chip length, these shifted replicas are defined as

$$c_\Delta(t - \widehat{\tau}_0 - \beta_k T_c) \triangleq c(t - \widehat{\tau}_0 - \beta_k T_c - T_c/2) - c(t - \widehat{\tau}_0 - \beta_k T_c + T_c/2) \tag{42}$$

where β_k is the *delay spacing coefficient* of the kth loop arm and $\widehat{\tau}_0$ is the MCTL's estimate of the direct-path delay. The arm corresponding to $k = 0$ is referred to as the *direct-path arm* ($\beta_0 = 0$) and the remaining arms as *multipath arms.* During tracking, the MCTL enables both the direct-path arm and the multipath arm corresponding to $\beta_k = \widehat{\alpha}$ (where $\widehat{\alpha}$ is the MCEU's estimate of the multipath delay coefficient).

In the analysis that follows, assume that the the MCEU is providing perfect estimates to the MCTL and that the multipath arm corresponding to $k = 1$ is enabled. In other words,

$$\beta_1 = \widehat{\alpha} = \alpha, \ \widehat{x}_0 = x_0, \text{ and } \widehat{x}_1 = x_1 \ . \tag{43}$$

3.3.2 Early-late correlator outputs. The baseband signal undergoes early-late correlation in each of the enabled arms. Under our assumption of exact MCEU estimates, the lowpass components of the correlator outputs are

$$\begin{aligned} z_0(t,\delta) &= x_0 D(\delta) + x_1 D(\delta + \alpha) + \eta_0(t) \\ z_1(t,\delta) &= x_0 D(\delta - \alpha) + x_1 D(\delta) + \eta_1(t) \end{aligned} \tag{44}$$

where the MCTL *D-curve*, $D(\delta)$, is defined as

$$D(\delta) \triangleq R_c\left(\delta - \frac{1}{2}\right) - R_c\left(\delta + \frac{1}{2}\right) \ . \tag{45}$$

Note that this curve is almost identical (except for the squaring operation) to the NCDLL S-curve of Equation 17. The noise components, $\eta_0(t)$ and $\eta_1(t)$, are described by

$$\eta_k(t) = \sqrt{2}\left[\eta_k^I(t)\cos(\theta_3) + \eta_k^Q(t)\sin(\theta_3)\right] \tag{46}$$

$$\begin{aligned}\eta_k^I(t) &\triangleq n_I(t)c_\Delta(t-\widehat{\tau}_0-\beta_k T_c) * h_{lpf}(t)\\ \eta_k^Q(t) &\triangleq n_Q(t)c_\Delta(t-\widehat{\tau}_0-\beta_k T_c) * h_{lpf}(t)\end{aligned}$$

where $*$ denotes convolution and $h_{lpf}(t)$ is the impulse response of the LPFs. It can be shown that, for code period $N \gg 1$, the two-sided PSD, $G_{\eta_k'}$, of $\eta_k^I(t)$ and $\eta_k^Q(t)$ is approximately (5)

$$G_{\eta_k'}(f) = N_0 \tag{47}$$

in units of W/Hz. Therefore, the output noise of the kth LPF given by Equation 46 has a two-sided PSD of

$$G_{\eta_k}(f) = \begin{cases} 2N_0 & |f| \le \frac{B_{LPF}}{2} \\ \\ 0 & elsewhere \end{cases} . \tag{48}$$

3.3.3 Discriminator output. After early-late correlation, the signal enters the *gain-phase correlators.* Here, the signal is mixed with the appropriate MCEU signal parameter estimate, $\widehat{x}_0$ or $\widehat{x}_1$, as shown in Figure 14. Assuming perfect estimation, the outputs of the gain-phase correlators are

$$\begin{aligned}y_0(t,\delta) &= x_0^2 D(\delta) + x_0 x_1 D(\delta+\alpha) + x_0\eta_0(t)\\ y_1(t,\delta) &= x_0 x_1 D(\delta-\alpha) + x_1^2 D(\delta) + x_1\eta_1(t)\,.\end{aligned} \tag{49}$$

These correlator outputs are summed to produce the MRDLL discriminator output, $e(t)$, given by

$$e(t,\delta) = y_0(t,\delta) + y_1(t,\delta) = S(\delta) + n_e(t) \; ; \tag{50}$$

the MRDLL S-curve is given as

$$S(\delta) = \left(x_0^2 + x_1^2\right) D(\delta) + x_0 x_1 \left[D(\delta + \alpha) + D(\delta - \alpha)\right] \; , \tag{51}$$

and the discriminator output noise, $n_e(t)$, is given by

$$n_e(t) = x_0 \eta_0(t) + x_1 \eta_1(t) \; . \tag{52}$$

Using the result of Equation 48, the discriminator output noise PSD becomes

$$G_{ne}(f) = \begin{cases} \left(x_0^2 + x_1^2\right)(2N_0) & |f| \leq \frac{B_{LPF}}{2} \\ 0 & elsewhere \end{cases} \; . \tag{53}$$

3.3.4 MRDLL steady-state tracking error due to multipath. To determine MRDLL steady-state tracking error due to multipath (neglecting noise effects), consider the representative MCTL S-curve shown in Figure 17. As was the case for the NCDLL in the presence of multipath, the MCTL S-curve is the summation of three separate curves multiplied and shifted by different amounts. One D-curve is centered at $\delta = 0$; the other two D-curves are shifted by α in opposite directions along the δ-axis. However, because of the way we chose to implement the gain-phase correlation in Equation 49, the two shifted D-curves are each multiplied by the same amount. As shown in Figure 18, this implementation ensures that the loop tracking point remains at $\delta = 0$. Therefore, unlike the NCDLL, the MRDLL exhibits zero steady-state code phase tracking error in the presence of multipath interference (neglecting noise effects).

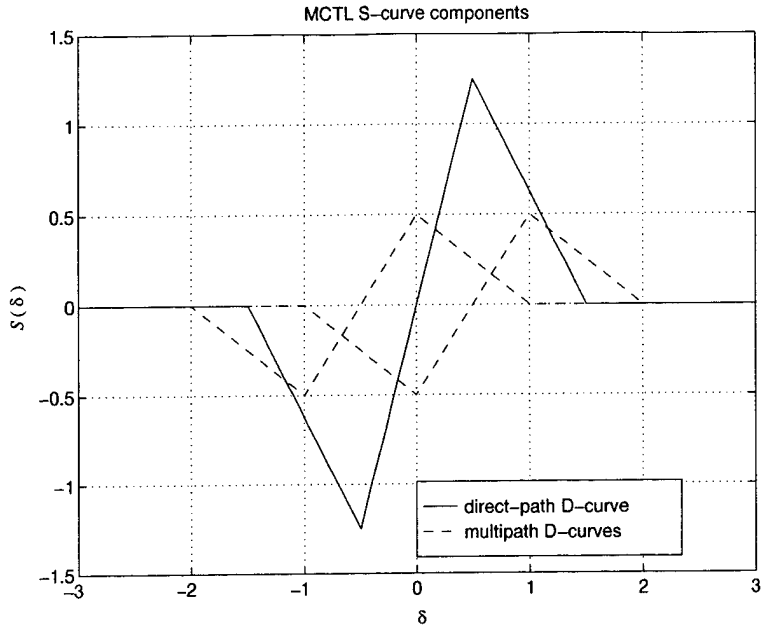

Figure 17. Typical MRDLL S-curve components in the presence of multipath interference.

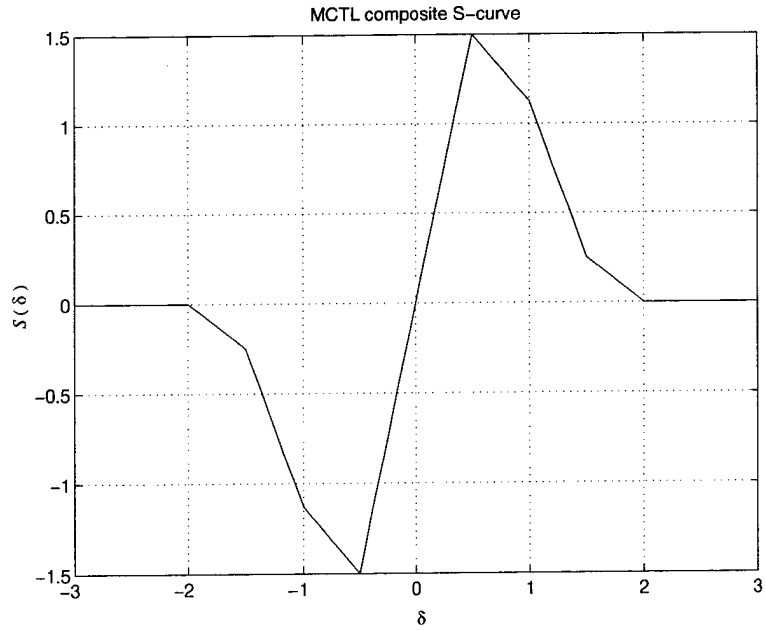

Figure 18. Typical composite MRDLL S-curve in the presence of multipath interference.

3.3.5 MCTL linear model. The MCTL linear model for small tracking errors ($\delta \approx 0$) is formed by first considering the operation of the VCC, described by

$$\frac{\widehat{\tau}_0(t)}{T_c} = g_c \int_0^t e(\lambda, \delta) * f(\lambda)\, d\lambda \ , \tag{54}$$

where $*$ denotes convolution, g_c is the VCC constant in Hz/volt, and $f(t)$ is the impulse response of the MCTL loop filter (contained in the ALC). For small tracking errors, the linear region of the S-curve can be expressed as

$$S(\delta) = A\delta = A\left(\frac{\tau_0 - \widehat{\tau}_0}{T_c}\right) \ , \tag{55}$$

where A is defined as the slope of the linear operating region for small δ. As will be seen in Section 3.4, this slope depends on the signal parameters, x_0, x_1, and α. Applying Equation 55 to Equation 50, the VCC equation can be rewritten as

$$\frac{\widehat{\tau}_0(t)}{T_c} = Ag_c \int_0^t \left[\left(\frac{\tau_0(\lambda) - \widehat{\tau}_0(\lambda)}{T_c}\right) + \frac{n_e(\lambda)}{A}\right] * f(\lambda)\, d\lambda \ . \tag{56}$$

Note that Equation 56 is almost identical to Equation 22 in Chapter II; therefore, the linear equivalent circuit and the Laplace transform linear model for the MCTL can be represented by Figures 8 and 9, respectively, with K_{nc} replaced by A, and $n_\epsilon(t)$ replaced by $n_e(t)$.

3.3.6 MCTL dynamics and tracking performance. The MCTL closed-loop transfer function can be written as

$$H(s) \triangleq \frac{\widehat{\tau}_0(s)}{\tau_0(s)} = \frac{Ag_cF(s)}{s + Ag_cF(s)} \tag{57}$$

where $F(s)$ is the transfer function of the loop filter. The MCTL is chosen to be a type 2 loop with the active lead-lag loop filter transfer function described by Equation 24; the loop natural frequency in rad/sec is given as

$$\omega_n = \sqrt{\frac{Ag_c}{\tau_1}} \ , \tag{58}$$

and the damping ratio and noise bandwidth are described by Equations 27 and 28.

The power spectrum of the tracking jitter is given by (2)

$$G_\delta(f) = |H(j2\pi f)|^2 \, G_{ne'}(f) \tag{59}$$

where $G_{ne'}(f) = G_{ne}(f)/A^2$ is the noise PSD at the input of the MCTL linear model. This noise PSD is approximately flat over the relatively small loop bandwidth; therefore, the tracking jitter can be approximated as

$$\begin{aligned} \sigma_\delta^2 &= \int_{-\infty}^{\infty} G_\delta(f)df = \int_{-\infty}^{\infty} G_{ne'}(f)\,|H(j2\pi f)|^2\,df = \frac{G_{ne}(0)}{A^2}B_L \\ &= \frac{4(x_0^2 + x_1^2)N_0}{A^2}B_L \end{aligned} \tag{60}$$

where $G_{ne}(0)$ was determined from Equation 53.

3.4 Adaptive loop controller (ALC) operation

The block diagram for the ALC is shown in Figure 19. The ALC (assuming a second order loop) consists of a loop filter defined by

$$F_A(s) = \frac{1 + \tau_2 s}{s} \tag{61}$$

preceded by a variable gain, $K_A = 1/\tau_1$; this effectively gives a loop filter transfer function identical to Equation 24 (i.e., $F(s) = K_A F_A(s)$). As will be seen, the variable gain allows the loop designer to fix the dynamic performance of the MCTL about the linear operating region (i.e., assuming small δ). The ALC feature is unique to the MRDLL and was not present in the RDLL design in (7).

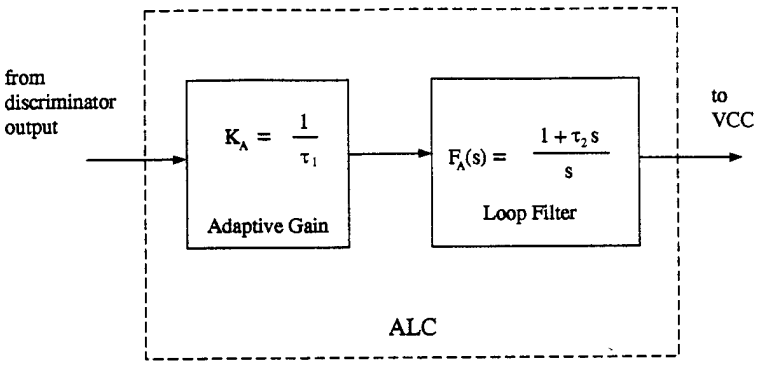

Figure 19. Adaptive loop controller (ALC) block diagram.

Adaptation to the multipath environment enables consistent dynamic code tracking performance in the presence of code Doppler shift. Such a Doppler shift is naturally created by satellite motion and can be intensified by aircraft maneuvers. The need for adaptive control is embodied in Equation 58. This Equation shows that the MCTL natural frequency (and, therefore, the damping ratio and noise bandwidth) is dependent upon the value of the S-curve slope, A, which in turn depends on the signal parameters, x_0, x_1, and α; the MCTL linear operating region is defined in Table 2.

Table 2. MCTL linear operating region.

Gain = $(A/2)$	**Multipath Delay Range**	**Tracking Error Range**
$x_0^2 + x_1^2 + 2x_0x_1$	$0 \leq \alpha < 0.5$	$-0.5 + \alpha \leq \delta \leq 0.5 - \alpha$
$x_0^2 + x_1^2 + 0.5x_0x_1$	$\alpha = 0.5$	$-0.5 \leq \delta \leq 0.5$
$x_0^2 + x_1^2 - x_0x_1$	$0.5 < \alpha \leq 1$	$0.5 - \alpha \leq \delta \leq -0.5 + \alpha$
$x_0^2 + x_1^2 - x_0x_1$	$1 \leq \alpha < 1.5$	$-1.5 + \alpha \leq \delta \leq 1.5 - \alpha$
$x_0^2 + x_1^2 - 0.5x_0x_1$	$\alpha = 1.5$	$-0.5 \leq \delta \leq 0.5$

Table 3. MCTL affine tracking region.

Slope	**Multipath Delay Range**	**Tracking Error Range**
$x_0^2 + x_1^2 + 0.5x_0x_1$	$0 \leq \alpha < 0.5$	$\lvert\delta\rvert > 0.5 - \alpha$
$x_0^2 + x_1^2 + 0.5x_0x_1$	$0.5 < \alpha < 1$	$\lvert\delta\rvert > -0.5 + \alpha$
$x_0^2 + x_1^2 - 0.5x_0x_1$	$1 \leq \alpha < 1.5$	$\lvert\delta\rvert > 1.5 - \alpha$

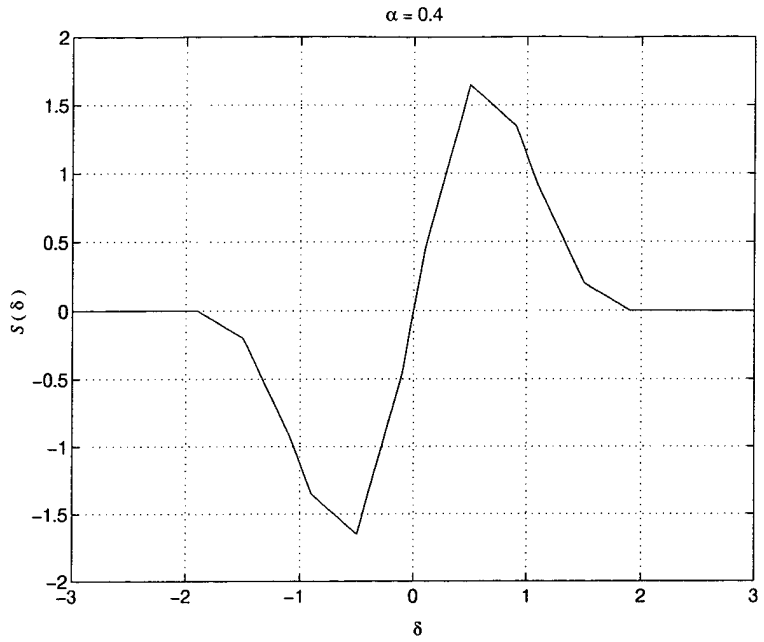

Figure 20. MRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 0.4$.

Figures 20 through 24 show typical MCTL S-curves for different values of α with $x_0 = 1$ and $x_1 = 0.5$; note the change in slope in the linear region about the origin. Also note (for $\alpha \neq 0.5,\ 1.5$) that outside this linear region (into the *affine* region defined in Table 3), the S-curve slope slightly changes, but the overall S-curve for $|\delta| \leq 0.5$ still remains approximately linear; therefore, loop dynamic performance will vary in the affine tracking region, but only slightly from that of the linear region. Both the figures and Table 2 demonstrate a significant difference between the MRDLL and the RDLL. In the RDLL, the equation governing the linear operating region does not change with integer chip delays and is always defined over $-0.5 \leq \delta \leq 0.5$ (7).

To maintain a fixed natural frequency, ω_n, (and therefore a fixed damping ratio and noise bandwidth), the ALC adapts the value of τ_1 via estimates from the MCEU; this approach is based on the certainty equivalence principle (1) wherein the controller is designed using estimates of unknown parameters. The order of operation for the ALC is

1. Receive signal estimates ($\widehat{x}_0$, $\widehat{x}_1$, and $\widehat{\alpha}$) from the MCEU

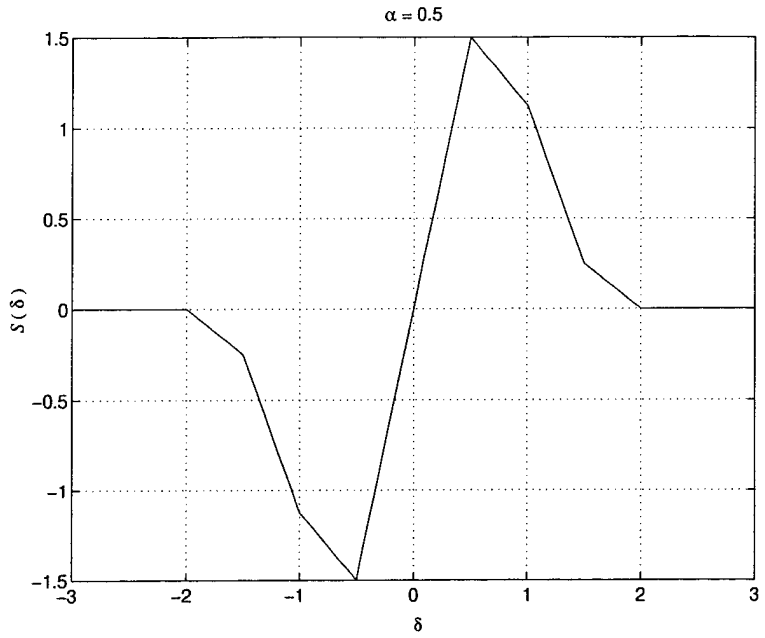

Figure 21. MRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 0.5$.

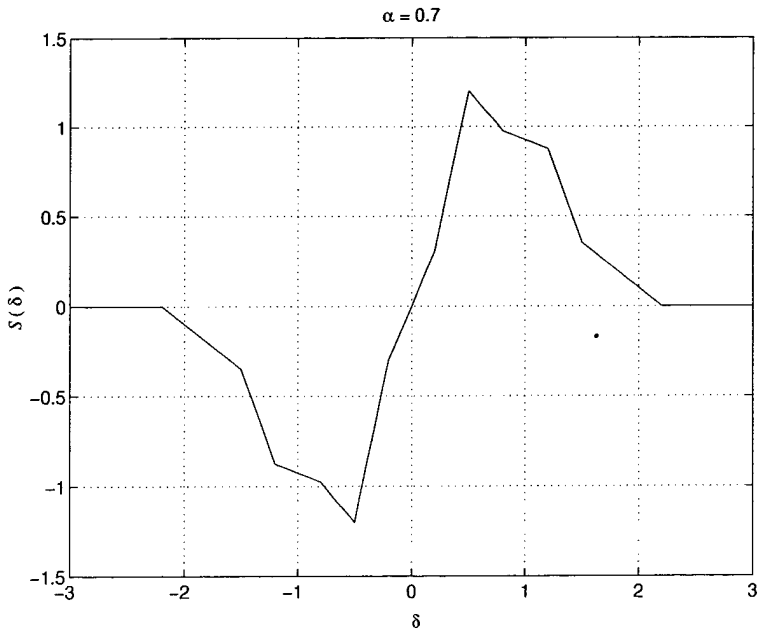

Figure 22. MRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 0.7$.

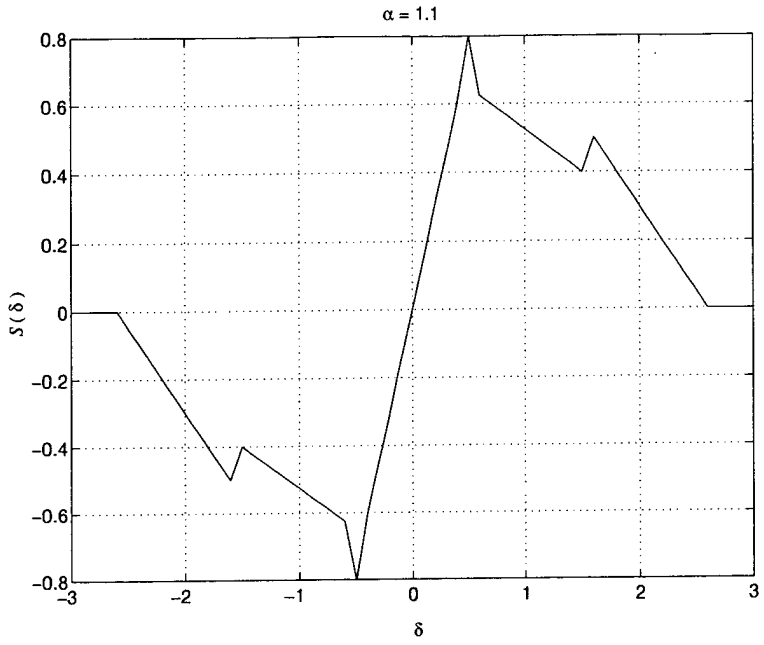

Figure 23. MRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 1.1$.

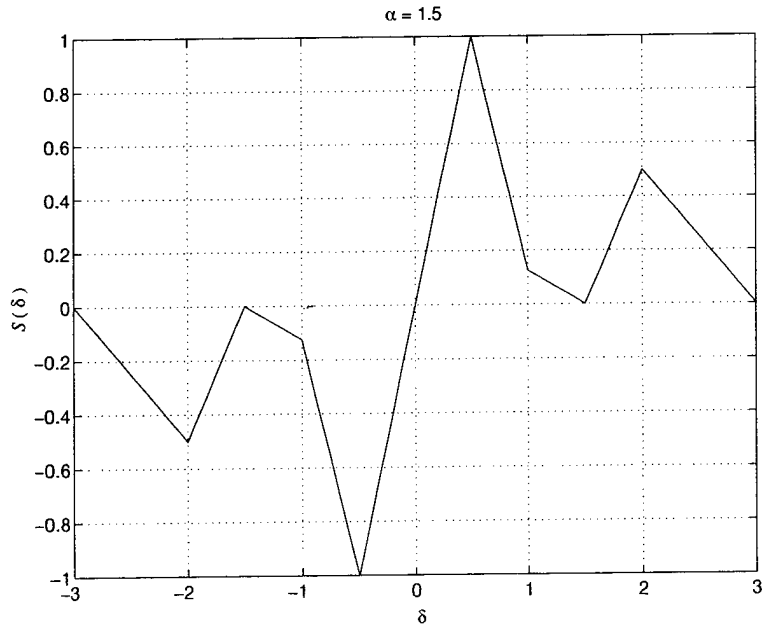

Figure 24. MRDLL S-curve with $x_0 = 1$, $x_1 = 0.5$, and $\alpha = 1.5$.

2. Use the MCEU estimates to compute an estimate, $\widehat{A}$, according to Table 2

3. Use $\widehat{A}$ to adjust the gain, K_A, according to

$$K_A = \frac{1}{\tau_1} = \frac{\omega_n^2}{\widehat{A} g_c} \tag{62}$$

which was obtained by rearranging Equation 58 and substituting $\widehat{A}$ for A.

One final design consideration needs to be made. For a second-order loop to operate correctly, the relationship

$$A g_c \gg \omega_n \tag{63}$$

must be satisfied (2). This condition can be restated by applying Equation 58 to give

$$K_A = \frac{1}{\tau_1} \ll \omega_n \, . \tag{64}$$

Therefore, the ALC must be designed so that the variable gain, K_A, does not exceed some threshold. If we represent this gain threshold as $K_{A\max}$, then

$$K_A \leq K_{A\max} \tag{65}$$

must be satisfied for the MCTL to operate correctly. The value of $K_{A\max}$ is chosen by the loop designer to optimize MCTL performance.

3.5 MCEU Analysis

3.5.1 Overview. We have already seen that if the MCEU provides perfect estimates to the MCTL, then we are able to remove steady-state tracking error due to a reflected signal; furthermore, these estimates allow the ALC to maintain consistent dynamic performance in the

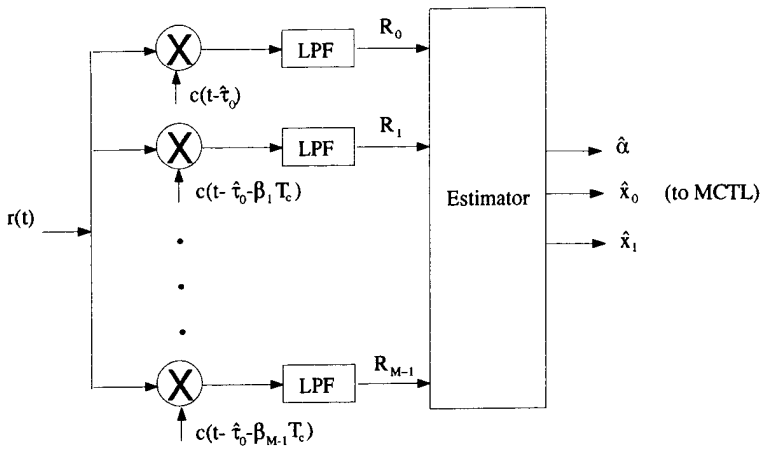

Figure 25. Multiple-correlator estimation unit (MCEU) block diagram.

MCTL (assuming linear operation). Now, we will examine the operation of the MCEU and the theory behind it.

3.5.2 Sampling the correlation function. The MCEU block diagram is shown in Figure 25. The MCEU consists of two components: a *correlator bank* and an *estimator*. The correlator bank consists of M arms in which the kth arm is correlated with a replica code given by $c(t-\widehat{\tau}_0-\beta_k T_c)$. Assuming that loop tracking errors are negligible (i.e., $\widehat{\tau}_0 = \tau_0$), the output of the kth correlator into the estimator is given by

$$R_k(t) = x_0 R_c(\beta_k) + x_1 R_c(\alpha - \beta_k) + v_k(t) \; . \tag{66}$$

The correlator output noise, $v_k(t)$, assumed to be zero-mean, lowpass AWGN, is given by

$$v_k(t) = n'(t)c(t-\widehat{\tau}_0-\beta_k T_c) * h_{lpf}(t) \tag{67}$$

where $n'(t)$ is the received baseband signal noise defined in Equation 36. This noise is assumed to have a two-sided PSD of N_0 W/Hz over B_{LPF} Hz.

Examining the signal component of the correlator output in Equation 66; this expression shows that, in effect, the kth correlator arm 'samples' both the direct path component and the corresponding multipath component of the code correlation function, $R_c(\Omega)$, at $\Omega = \beta_k$. The estimator then uses the correlator bank outputs as discrete-time measurements to estimate the unknown parameters, x_0, x_1, and α. This concept of sampling the correlation function is presented in (9); however, the analysis in (9) does not specify the estimation scheme employed. For our estimator, we have chosen to use the ML estimation scheme presented in the next section.

3.5.3 The MCEU estimator. To begin the estimator analysis, let one sample of the unknown signal parameters be represented in vector form, $\mathbf{x}$, as

$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} . \tag{68}$$

This allows us to represent the sampled correlator outputs, at a given time instant, in vector form as

$$\mathbf{R} = \begin{bmatrix} R_0 & R_1 & \cdots & R_{M-1} \end{bmatrix}^T = \mathbf{H}(\alpha)\mathbf{x} + \mathbf{v} \tag{69}$$

where $\mathbf{R}$ is an $M \times 1$ vector of the correlator outputs, R_k, and $\mathbf{H}(\alpha)$ is an $M \times 2$ regressor matrix given by

$$\mathbf{H}(\alpha) = \begin{bmatrix} R_c(\beta_0) & R_c(\alpha - \beta_0) \\ R_c(\beta_1) & R_c(\alpha - \beta_1) \\ \vdots & \vdots \\ R_c(\beta_{M-1}) & R_c(\alpha - \beta_{M-1}) \end{bmatrix} . \tag{70}$$

The $M \times 1$ noise vector, $\mathbf{v} = [v_0\ v_1\ \cdots\ v_{M-1}]^T$, consists of the correlator noise output at the time the measurements, R_k, are taken and is distributed as $N : [\mathbf{0}, \mathbf{C_v}]$, where $\mathbf{C}_v$ is the noise covariance matrix. To determine an expression for $\mathbf{C_v}$, we treat the spreading code as a random binary process and assume it is independent of the noise; this gives the continuous-time cross-correlation between

the ith and jth correlator noise outputs as

$$\begin{aligned} R_{ij}(\tau) &= E\left[v_i(t)v_j(t+\tau)\right] \\ &= E\left\{\left[n'(t)c(t-\widehat{\tau}_0-\beta_i T_c)*h_{lpf}(t)\right]\left[n'(t+\tau)c(t+\tau-\widehat{\tau}_0-\beta_j T_c)*h_{lpf}(t+\tau)\right]\right\} \\ &= E\left[c(t-\widehat{\tau}_0-\beta_i T_c)c(t+\tau-\widehat{\tau}_0-\beta_j T_c)\right]E\left[n''(t)n''(t+\tau)\right] \\ &= R_c\left(\frac{\tau}{T_c}+\Delta\beta_{ij}\right)R_{n''}(\tau) \end{aligned} \tag{71}$$

where

$$n''(t) = n'(t) * h_{lpf}(t) \tag{72}$$

and

$$\Delta\beta_{ij} = \beta_i - \beta_j \ . \tag{73}$$

Because the spreading code bandwidth is much greater than bandwidth of the lowpass noise, $R_{n''}(\tau)$ is approximately constant over significant values of $R_c\left(\frac{\tau}{T_c}+\Delta\beta_{ij}\right)$, the cross-correlation can be approximated by

$$R_{ij}(\tau) \approx R_c(\tau+\Delta\beta_{ij})R_{n''}(0) = R_c(\tau+\Delta\beta_{ij})\sigma_v^2 \tag{74}$$

where σ_v^2 is the variance of each the correlator noise outputs. This result gives the discrete-time noise covariance matrix, $\mathbf{C_v}$, as

$$\mathbf{C_v} = \sigma_{\mathbf{v}}^{\mathbf{2}}\mathbf{C} \tag{75}$$

where $\mathbf{C}$ is the $M \times M$ *correlation* matrix

$$\mathbf{C} = \begin{bmatrix} 1 & R_c(\beta_0-\beta_1) & \cdots & R_c(\beta_0-\beta_{M-1}) \\ R_c(\beta_1-\beta_0) & 1 & \ddots & \vdots \\ \vdots & \ddots & \ddots & R_c(\beta_{M-2}-\beta_{M-1}) \\ R_c(\beta_{M-1}-\beta_0) & \cdots & R_c(\beta_{M-1}-\beta_{M-2}) & 1 \end{bmatrix} \tag{76}$$

which is symmetric because $R_c(\beta_i - \beta_j) = R_c(\beta_j - \beta_i)$ and Toeplitz since $R_c(\beta_{i+k} - \beta_i) = R_c(\beta_k - \beta_0)$ for all i, k.

Assuming that a reflected signal is present (i.e., $\alpha \neq 0$), the maximum-likelihood (ML) estimate of $\mathbf{x}$ for a given value of α is given by (6)

$$\widehat{\mathbf{x}} = \mathbf{P}(\alpha)\mathbf{R} \tag{77}$$

where

$$\mathbf{P}(\alpha) \triangleq \left[\mathbf{H^T}(\alpha)\mathbf{C^{-1}}\mathbf{H}(\alpha)\right]^{-1}\mathbf{H^T}(\alpha)\mathbf{C^{-1}}\ ; \tag{78}$$

observe that knowledge of σ_v^2 is not needed to compute the estimate, $\widehat{\mathbf{x}}$, of Equation 77.

It is not feasible to compute $\widehat{\mathbf{x}}$ per Equation 77 for the continuum of $\alpha \in [0,\ 1.5]$. We alternatively choose to consider the estimates, $\widehat{\mathbf{x}}_k$, generated by assuming that $\alpha = \beta_k$ for $k = (0,\ 1,\ 2,\ ...,\ M-1)$. We are, in effect, considering a finite collection of uniformly spaced possibilities for α. To compute the estimate, $\widehat{\mathbf{x}}_k$ corresponding to the event $\alpha = \beta_k$, the MCEU forms $P(\beta_k)$ according to Equation 78. Equation 77 is then, for $k = (1,\ 2,\ ...,\ M-1)$, used to compute $\widehat{\mathbf{x}}_k$ as

$$\widehat{\mathbf{x}}_k = \mathbf{P}(\beta_k)\mathbf{R} \quad (k \neq 0)\ . \tag{79}$$

This computation is simplified by realizing that $\mathbf{P}(\beta_\mathbf{k})$ is a 2×16 matrix with non-zero entries only in the first and $(k+1)$st columns (zeros elsewhere); furthermore, the elements in the $(k+1)$st column are simply the reverse of the elements in the first column; therefore, the kth estimate can be rewritten as

$$\widehat{\mathbf{x}}_k = \begin{bmatrix} p_{11} & p_{21} \\ p_{21} & p_{11} \end{bmatrix} \begin{bmatrix} R_0 \\ R_k \end{bmatrix} = \mathbf{P}_k\mathbf{R}_k \quad (k \neq 0) \tag{80}$$

where $\mathbf{P_k}$ is a symmetric 2×2 matrix with the same first column as $\mathbf{P}(\beta_\mathbf{k})$ and $\mathbf{R_k}$ is a 2×1 column vector with elements corresponding to the correlator measurements R_0 and R_k.

To compute the direct-path only estimate corresponding to $k = 0$ (the event $\alpha = 0$), we must first realize that $\mathbf{H^T}(\alpha)\mathbf{C^{-1}}\mathbf{H}(\alpha)$ in Equation 78 is singular when $\alpha = 0$; this means $\mathbf{P}(\beta_0) = \mathbf{P(0)}$ cannot be evaluated as in Equation 79. Since we have assumed no multipath, we can also assume that $\widehat{x}_1 = 0$ (see Equation 35); therefore, we only need to estimate the direct-path signal parameter, x_0. This direct-path estimate is calculated by defining a constant 16×1 vector, $\mathbf{H}(\beta_0)$, as

$$\mathbf{H}(\beta_0) \triangleq \begin{bmatrix} R_c(\beta_0) \\ R_c(\beta_1) \\ \vdots \\ R_c(\beta_{M-1}) \end{bmatrix} \tag{81}$$

and substituting this vector for $\mathbf{H}(\alpha)$ in Equation 78 to form

$$\mathbf{P}(\beta_0) \triangleq \left[\mathbf{H}(\beta_0)^{\mathbf{T}}\mathbf{C^{-1}}\mathbf{H}(\beta_0)\right]^{-1}\mathbf{H}(\beta_0)^{\mathbf{T}}\mathbf{C^{-1}}\ ; \tag{82}$$

this gives the $k = 0$th estimate as

$$\widehat{\mathbf{x}}_0 \triangleq \begin{bmatrix} \widehat{x}_0 \\ \widehat{x}_1 \end{bmatrix}_0 = \begin{bmatrix} \mathbf{P}(\beta_0)\mathbf{R} \\ 0 \end{bmatrix}. \tag{83}$$

As for the multipath case, this computation can be simplified by realizing that $\mathbf{P}(\beta_0)$ is a fixed 1×16 vector with a 1 in the first element and zeros in all the remaining elements; therefore, the direct path signal estimate is simply the 0th correlator output (i.e., $\widehat{x}_0 = R_0$) because, as seen in Equation 66, $R_0 = x_0 R_c(0)$ and, from Equation 5, $R_c(0) = 1$. In summary, the estimation vector for the direct-path only case becomes

$$\widehat{\mathbf{x}}_0 = \begin{bmatrix} R_0 \\ 0 \end{bmatrix}. \tag{84}$$

Once all estimates have been computed, the MCEU must determine which estimate, $\widehat{\mathbf{x}}_k$, most likely gave rise to the correlator measurement matrix, $\mathbf{R}$; the corresponding β_k becomes the ML estimate for α (i.e., $\widehat{\alpha} = \beta_k$). To accomplish this, the MCEU forms a matrix of predicted observations, $\widehat{\mathbf{R}}_k$, for each of the M correlator arms as

$$\widehat{\mathbf{R}}_k = \mathbf{H}(\widehat{\alpha})\widehat{\mathbf{x}}_\mathbf{k} = \mathbf{H}(\beta_\mathbf{k})\widehat{\mathbf{x}}_\mathbf{k} \ . \tag{85}$$

The prediction error vector, $\mathbf{e}_k$, is then computed according to

$$\mathbf{e_k} = \mathbf{R} - \widehat{\mathbf{R}}_\mathbf{k} \ . \tag{86}$$

Consider the case when the estimates equal the actual values; define this event to be

$$\Psi = \{(\alpha = \widehat{\alpha} = \beta_k) \cap (x_0 = \widehat{x}_0) \cap (x_1 = \widehat{x}_1)\} . \tag{87}$$

We want to choose the most likely value of $\mathbf{e_k}$ given that Ψ is true; in other words, we want to maximize the likelihood function, $p(\mathbf{e_k}|\Psi)$.

Applying Equations 69 and 85, and our definition of $\mathbf{e_k}$ in Equation 86, we get

$$\mathbf{e_k}|\Psi = \mathbf{R} - \widehat{\mathbf{R}}_\mathbf{k} = \mathbf{H}(\alpha)\mathbf{x} - \mathbf{H}(\widehat{\alpha})\widehat{\mathbf{x}}_\mathbf{k} + \mathbf{v} = \mathbf{v} \ , \tag{88}$$

because, under our assumed condition, $\mathbf{H}(\alpha)\mathbf{x} = \mathbf{H}(\widehat{\alpha})\widehat{\mathbf{x}}_\mathbf{k}$. Therefore, $(\mathbf{e_k}|\Psi)$ is distributed as $N : [\mathbf{0}, \mathbf{C_v}]$ and this gives the likelihood function as (6)

$$p(\mathbf{e_k}|\Psi) = \frac{1}{(2\pi)^{M/2}\sqrt{\det(\mathbf{C_v})}} \exp\left\{-\frac{1}{2}\mathbf{e_k^T}\mathbf{C_v^{-1}}\mathbf{e_k}\right\} \ . \tag{89}$$

For a given noise variance, σ_v^2, it can be seen from Equation 89 that the likelihood function is maximized when $\mathbf{e_k^T C^{-1} e_k}$ is minimum; therefore, the MCEU computes $E_k = \mathbf{e_k^T C^{-1} e_k}$ and chooses the estimates corresponding to the minimum value of E_k (for $k = 0,\ 1,\ ...,\ M-1$). These estimates are sent to the MCTL as the ML estimates, $\widehat{\mathbf{x}} = \widehat{\mathbf{x}}_k$ and $\widehat{\alpha} = \beta_k$.

IV. Computer Simulation Results

4.1 Overview

This chapter presents the results of digital computer simulations of the NCDLL and MRDLL. Computer simulations were performed to verify the predicted theoretical results presented in Chapters I through III, and to investigate actual loop performance in the absence of simplifying assumptions such as perfect estimation and linear operation. Specifically, four different simulations were performed:

1. MCEU estimator performance assuming MCTL perfect code phase estimation in the presence of AWGN (preliminary simulation using MATLAB)
2. NCDLL vs MRDLL steady-state tracking performance in the presence of multipath without AWGN
3. NCDLL vs MRDLL steady-state tracking performance in a direct-path only environment without AWGN
4. MRDLL steady-state tracking performance in the presence of multipath with AWGN

Unless otherwise stated, all simulations were performed with the Signal Processing Workstation (SPW) software package, and the resulting data was analyzed using MATLAB. All SPW computer models and MATLAB m-files used for simulation and/or data processing are contained in Appendices A and B.

The remainder of this chapter is divided into six sections. Section 4.2 presents the main simulation parameters and discusses how and why the parameters were chosen. Next, section 4.3 describes the data processing methods that were used. The remaining four sections present the objectives, methods, and results of each of the four simulation scenarios outlined above.

4.2 Simulation Parameters

4.2.1 Simulation parameters versus actual GPS parameters. We desired simulation parameters that reflected (as best as possible) actual GPS signal and tracking loop specifications; however, because SPW processing time was relatively slow, simulation times and data file size became limiting factors. Because of this, we were constrained to choose parameters with the intent of demonstrating general loop behavior and tendencies as opposed to attempting to simulate actual performance in the GPS environment; despite these constraints, simulation results still provided valuable insight into the application of the MRDLL to the GPS multipath mitigation problem. In this section, we look at several important simulation parameters, how these parameters relate to actual GPS values, and each parameter's potential effect on simulation results.

4.2.2 Sampling frequency. Three main factors were considered in choosing the SPW simulation sampling frequency, f_s (in Hz): code phase timing error resolution, the number of relative multipath delay samples, and simulation time. Estimating the code phase timing error involved comparing leading edges of the loop on-time replica and the received direct-path code. The relative time delay (or advance) between any two leading edges had to be estimated with adequate resolution; therefore, a sampling frequency corresponding to 100 samples/chip was chosen. The second constraint on f_s was that it had to be chosen so that the value of the relative multipath delay, αT_c, corresponded to an integer number of samples (i.e., $\alpha T_c f_s$ had to be a positive integer); in our simulations, α took on values equal to integer multiples of 0.1 between 0 and 1.5. Taking all these considerations into account, along with simulation time and data file size, led to a choice of $f_s = 1000$ Hz; this corresponded to a chip rate of $W_c = 1/T_c = 10$ Hz (to meet the 100 samples/chip requirement).

4.2.3 Carrier phase parameter. Since the NCDLL code phase error due to multipath is maximum whenever the direct-path signal is in carrier phase with the reflected signal (see Chapter II), simulations were run with $\theta_0 = \theta_1$; this allowed us to demonstrate the MRDLL's ability to

reduce multipath tracking errors even in a multipath environment that would otherwise cause significant tracking errors in the standard NCDLL. Note that the in-phase condition also implies that the MRDLL local carrier phase error, ϕ_e, described in Chapter III, is equal to zero; note, however, this does not imply a best-case scenario for the MRDLL since *any* local carrier phase error is accounted for by the signal estimates, $\widehat{x}_0$ and $\widehat{x}_1$ (see Equation 35).

4.2.4 Carrier frequency. Simulations were run independent of choosing a carrier frequency parameter, ω_c. This was possible because the complex envelope model described in Appendix A is independent of carrier frequency except for the carrier phases, θ_i, which depend on α and ω_c. Since we are interested in *relative* multipath delays, we chose the direct-path delay as $\tau_0 = 0$, which meant $\theta_0 = 0$ and $\theta_1 = -\omega_c \alpha T_c = -2\pi f_c \alpha T_c$. For our simulation, we assumed the product, $f_c T_c$, was large enough so that $\alpha f_c T_c$ was always an integer for our chosen values of $\alpha = 0.1k$ and k an integer between 0 and 15; for comparison, recall that for GPS, $f_c T_c$ is a relatively large number (e.g., $f_c T_c = 1540$ for C/A code) (see Table 1). Therefore, under our assumption of large $f_c T_c$, our simulated reflected signal carrier phase, θ_1, was always an integer multiple of 2π. This was equivalent to our setting $\theta_0 = \theta_1$ during simulation.

4.2.5 Filter bandwidths. Recall from Chapters II and III that the NCDLL and MCTL BPFs and LPFs have a one-sided noise bandwidth, B Hz, such that $B \ll 2W_c$; however, choosing too small a value of B increased the response time of the code tracking loops so as to make simulation time unacceptable. Taking these factors into consideration, we chose our simulation models to have filters with a bandwidth such that $2W_c/B = 100$ (i.e., $B = 0.2$ Hz). Recall from chapter II that a typical ratio for a GPS C/A code tracking loop is $2W_c/B \approx 1000$. The bandwidths of the LPFs in the MCEU's code correlator bank were also chosen with bandwidth $B = 0.2$ Hz.

4.2.6 Code period parameter. Examining Equation 3 in Chapter I, one can see that the code correlation function, $R_c(\Omega)$, can be implemented by a code multiplier followed by a BPF or

LPF with bandwidth $B \leq 2/NT_c$ Hz, where N is the code period in chips (the filter acts as an integrator). As we have already seen, our simulation model used a chip rate of $1/T_c = 10$ Hz with filters of bandwidth $B = 0.2$ Hz; this corresponded to a maximum code period of $N = 2^m - 1 = 63$ chips (for m an integer), where $m = 6$ is the code generator's shift register order described in Appendix A. Choosing $N = 63$ for our simulations still allowed us to use the large N code correlation approximation of Equation 5 with reasonable accuracy; however, the approximation is not as accurate in our case as it is for actual GPS code period parameters where N is much greater (see Table 1).

4.2.7 Carrier-to-noise density ratio and loop SNR. When considering GPS code tracking loop performance in AWGN, it is important to consider the received direct-path carrier power to noise density ratio (C/N_0 in dB-Hz) and the tracking loop SNR, ρ_L (in dB). In our case, the received carrier power, C, equaled the transmitted power, P, since $C \triangleq Pa_0^2$ and we chose $a_0 = 1$. The code tracking loop's SNR is determined by the two-sided PSD of the lowpass noise input to the linear loop model (Figure 8) and the two-sided loop noise bandwidth, $2B_L$ Hz, where B_L is defined in Equation 28.

For the NCDLL, if we assume $N \gg 1$ and an early-late offset of $\Delta = 1$, the loop SNR is given by (5)

$$\rho_L = \frac{P}{N_0}\left(\frac{1}{2B_L}\right) \text{ (for NCDLL).} \tag{90}$$

For GPS, typical values of P/N_0 range from approximately 30 to 45 dB-Hz while a typical NCDLL loop bandwidth is $B_L = 10$ Hz (4) (14); this gives a loop SNR between 17 to 32 dB.

For the MCTL, the input noise PSD for the linear model is defined over the LPF bandwidth, B, and can be expressed as (see Chapter III)

$$G_{ne'}(f) = \frac{2N_0(x_0^2 + x_1^2)}{A^2}\,, \tag{91}$$

where A is the slope of the MCTL S-curve defined in Table 2 for different signal parameters x_0, x_1, and α. Therefore, the MCTL loop SNR can be written as

$$\rho_L = \frac{P}{G_{ne'} 2B_L} = G_L \frac{P}{N_0 2B_L} \text{ (for MCTL)} \tag{92}$$

where the *loop SNR gain*, G_L, depends on the multipath signal parameters and is defined as

$$G_L \triangleq A^2 / \left(x_0^2 + x_1^2\right) \ . \tag{93}$$

For all simulations, we chose $P = 1/2$ W and the attenuation coefficients as $a_0 = 1$ and $a_1 = 0.5$; under our in-phase condition described in section 4.2.3, this gave the multipath signal parameters as $x_0 = 1$ and $x_1 = 0.5$ (see Equation 35). Therefore, G_L was defined for different values of α as shown in Table 4.

Table 4. MRDLL loop SNR gain ($x_0 = 1,\ x_1 = 0.5$).

α	$\mathbf{G_L}$ (dB)
$0 \leq \alpha < 0.5$	12
$\alpha = 0.5$	8.6
$0.5 < \alpha \leq 1$	2.6
$1 \leq \alpha < 1.5$	2.6
$\alpha = 1.5$	5

In practical code tracking loops, the loop noise bandwidth is usually much less than the BPF or LPF bandwidths (i.e., $2B_L \ll B$); however, since our choice of B was constrained to be relatively small, satisfying $2B_L \ll B$ required extremely small values of the loop natural frequency, ω_n (since, from Equation 28, B_L is proportional to ω_n^2). These low values of ω_n made MCTL response slow and resulted in excessive simulation times and data file sizes; therefore, $\omega_n = 6\pi$ rad/sec was selected for the MCTL to keep simulations within reasonable limits while maintaining loop stability (the same consideration led to choosing $\omega_n = 4\pi$ rad/sec for the NCDLL as $\omega_n = 6\pi$ rad/sec made the loop unstable). However, this choice of $\omega_n = 6\pi$ rad/sec resulted in $2B_L \approx 20$

Hz (obtained from Equation 28 for a damping ratio, $\zeta = \sqrt{2}/2$), which was 100 times greater than the MCTL LPF bandwidth, $B = 0.2$ Hz; since the input noise PSD for the MCTL linear model is defined over B, this LPF bandwidth became the effective loop bandwidth. Therefore, the MCTL loop SNR for our simulations was

$$\rho_{eff} = G_L \frac{P}{N_0 B} \text{ (for MRDLL simulation).} \tag{94}$$

For the simulations involving MRDLL performance in AWGN, values of ρ_{eff} ranged from approximately 20 to 40 dB.

4.2.8 Simulation parameter summary. A summary list of all simulation parameters is presented below; any parameters not specifically covered in sections 4.2.2 through 4.2.7 were chosen based on designer preference and/or simulation time and data file size constraints:

- Sampling frequency: $f_s = 1000$ Hz
- Transmitted signal power: $P = 1/2$ W
- Attenuation coefficients: $a_0 = 1$; $a_1 = 0.5$
- Direct-path propagation delay: $\tau_0 = 0$ sec
- Multipath delay coefficient: $\alpha = 0.1k$ (for k an integer between 0 and 15)
- Carrier phase (direct-path and reflected signals): $\theta_0 = \theta_1 = 0$ rad
- Local carrier phase error (MRDLL baseband conversion): $\phi_e = 0$ rad
- MRDLL signal parameters (see Equation 35): $x_0 = 1$; $x_1 = 0.5$ ($x_1 = 0$ for direct-path only)
- Chip rate: $1/T_c = 10$ Hz (i.e., 1 chip per 100 samples)
- DS/SS code period: $N = 63$ chips/period
- Code Doppler: negligible

- BPF bandwidth (at IF), LPF bandwidth (two-sided), and MCEU correlator bank LPF bandwidth: $B = 0.2$ Hz
- NCDLL and MCTL loop order: Second-order with active lead-lag filter as described by Equation 24.
- NCDLL natural frequency: $\omega_n = 4\pi$ rad/sec (from Equation 28)
- MCTL natural frequency: $\omega_n = 6\pi$ rad/sec (from Equation 28)
- NCDLL and MCTL damping ratio: $\zeta = \sqrt{2}/2 \approx 0.707$
- MCTL effective loop SNR: $20 \leq \rho_{eff} \leq 40$ dB
- ALC gain threshold (MRDLL): $K_{A\,\max} = \omega_n/50$
- Early-late correlator spacing (MRDLL and NCDLL): $\Delta = 1$ chip
- Delay spacing coefficients (MRDLL): $\beta_k = 0.1k$ (for k an integer between 0 and 15)

4.3 Data processing

All data was processed in MATLAB using m-files contained in Appendix B; for the SPW simulations, data was first converted from SPW format to MATLAB *mat-file* format via SPW's MATLAB SIGNAL SINK block. Data processing focused mainly on two measurements: normalized root-mean-square (rms) code phase tracking error, and the MCEU estimator means and variances.

For each loop simulation run, the normalized rms tracking error, σ_{rms} (in chips), was estimated by comparing each leading edge of the received direct-path DS/SS code to the corresponding leading edge of the 'on-time' replica code generated by the code tracking loop; by representing the number of samples between the ith pair of leading edges as d_i, we computed the normalized rms tracking error estimate as

$$\sigma_{rms} = \frac{1}{T_c f_s}\sqrt{\frac{1}{L}\sum_{i=1}^{L}(d_i^2)} \quad \text{chips}, \tag{95}$$

where L was the number of leading edge pairs present over the given sample size. The algorithm described by Equation 95 was implemented using the *leadedge.m* m-file.

To measure MCEU estimator performance, the MCEU estimator outputs ($\widehat{x}_0$, $\widehat{x}_1$, and $\widehat{\alpha}$) were sampled for a given MRDLL or MCEU simulation run. The outputs were plotted versus time using the *mrdlest.m* m-file; this m-file also returned the sample means and variances of $\widehat{x}_0$, $\widehat{x}_1$, and $\widehat{\alpha}$ using MATLAB's *mean* and *cov* functions.

4.4 Simulation #1: MCEU estimator performance with perfect code phase estimation (MATLAB simulation)

4.4.1 Objective. As seen in Chapter III, the MRDLL's ability to track out multipath errors greatly depends on the ability of the MCEU to provide accurate signal parameter estimates to the MCTL; therefore, investigating MCEU estimator behavior was essential to evaluating the the MRDLL design for possible GPS multipath applications. The objective of this simulation was to investigate the MCEU estimator performance assuming zero MCTL code phase tracking error (i.e., $\widehat{\tau}_0 = \tau_0$); by making this assumption, we were able perform the simulation in MATLAB without running the full SPW MRDLL simulation model (which was relatively slow). The advantage of using MATLAB to simulate the MCEU estimator was that MATLAB is optimized to perform the matrix operations needed to generate the estimates provided to the MCTL.

4.4.2 Method. Using the *mceu.m* m-file, simulations were run for different values of α and P/N_0B; each iteration of *mceu.m* represented one sample of the MCEU estimator output. A total of 100 iterations were performed for each value of α and P/N_0B; then, for each of these 100-sample outputs, we estimated the sample mean and sample variance of the ML estimates $\widehat{x}_0$, $\widehat{x}_1$, and $\widehat{\alpha}$. In each trial, $x_0 = 1$ and $x_1 = 0.5$ were the true parameters except at $\alpha = 0$, wherein $x_1 = 0$ was required (no multipath present).

4.4.3 Results. In Figures 26 through 33, the mean values (taken over 100 samples) of the MCEU ML estimates are plotted versus α for four different values of P/N_0B; these plots yield the following observations:

- As expected, the mean values of $\widehat{x}_0$ and $\widehat{x}_1$ approached the actual signal parameter values as P/N_0B increased. On the other hand, $\widehat{\alpha}$ provided an accurate estimate of multipath delay only when multipath was actually present; for the direct-path only case (i.e., $\alpha = 0$ and $x_1 = 0$), $\widehat{\alpha}$ exhibited a significant bias.

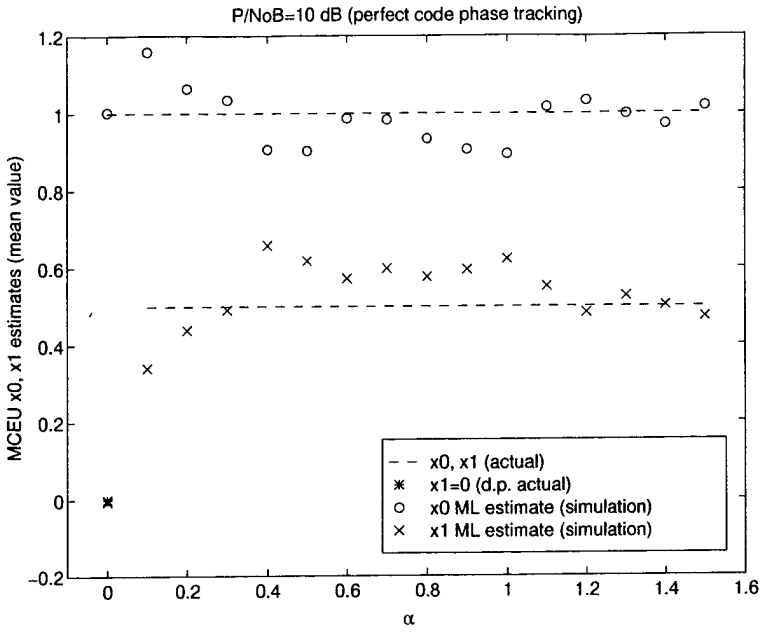

Figure 26. Simulation results showing the mean value of MCEU ML estimate, $\widehat{\mathbf{x}}$, at $P/N_0 =$ 10 dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

- The bias on $\widehat{\alpha}$ at $\alpha = 0$ implied that, when no multipath was present, the MCEU was consistently choosing the wrong $\widehat{x}_k$ as its ML estimate and would therefore erroneously enable one of the MCTL's multipath arms. However, the plots also show that the bias on $\widehat{x}_0$ and $\widehat{x}_1$ at $\alpha = 0$ was minimal despite the large bias on $\widehat{\alpha}$; this seemingly contradictory result implied that the effect on code tracking would be minimal, because the signal in the multipath arm would be multiplied by $\widehat{x}_1 \approx 0$ (refer to Figure 14).

- The ML estimates when the relative multipath delay was small (i.e., $\alpha = 0.1$) were generally worse than the ML estimates at greater relative multipath delays.

- Correlation between the $\widehat{x}_0$ and $\widehat{x}_1$ ML estimates is apparent (see Figure 26).

These observations are explained theoretically by examining the statistical properties of the 16 signal parameter estimates, $\widehat{\mathbf{x}}_k$, from which the MCEU chooses its ML estimate, $\widehat{\mathbf{x}}$. The mean

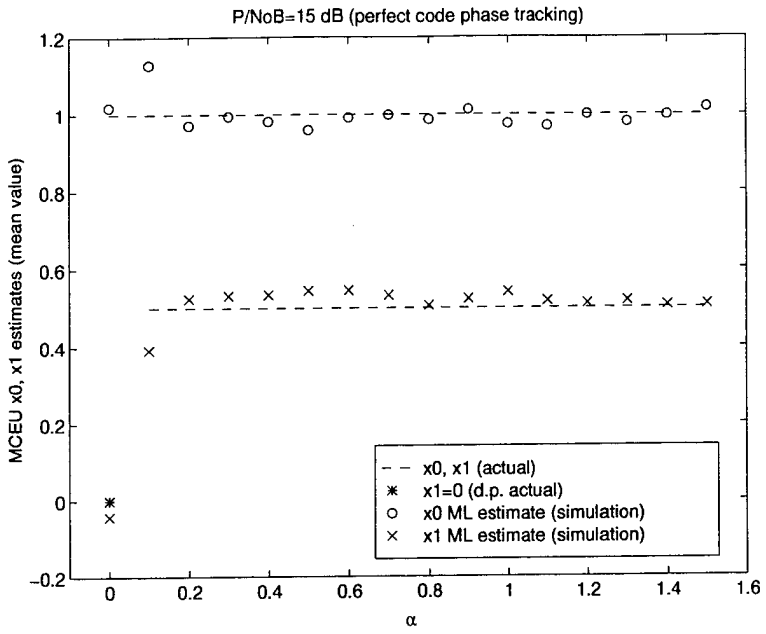

Figure 27. Simulation results showing the mean value of MCEU ML estimate, $\hat{\mathbf{x}}$, at $P/N_0 = 15$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

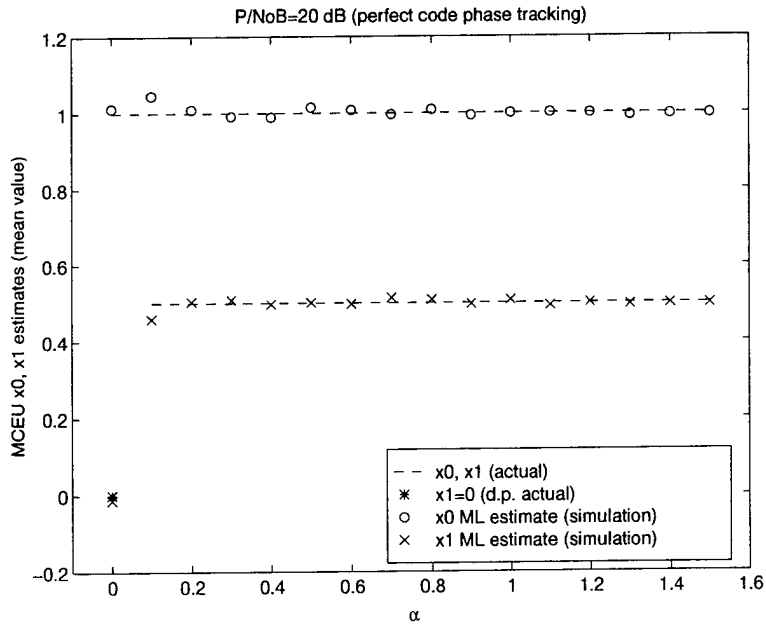

Figure 28. Simulation results showing the mean value of MCEU ML estimate, $\hat{\mathbf{x}}$, at $P/N_0 = 20$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

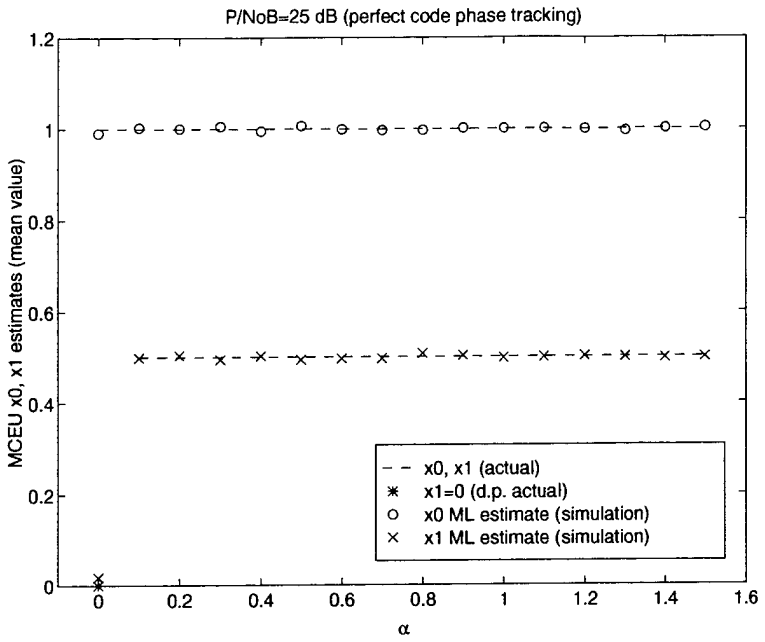

Figure 29. Simulation results showing the mean value of MCEU ML estimate, $\hat{\mathbf{x}}$, at $P/N_0 =$ 25 dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

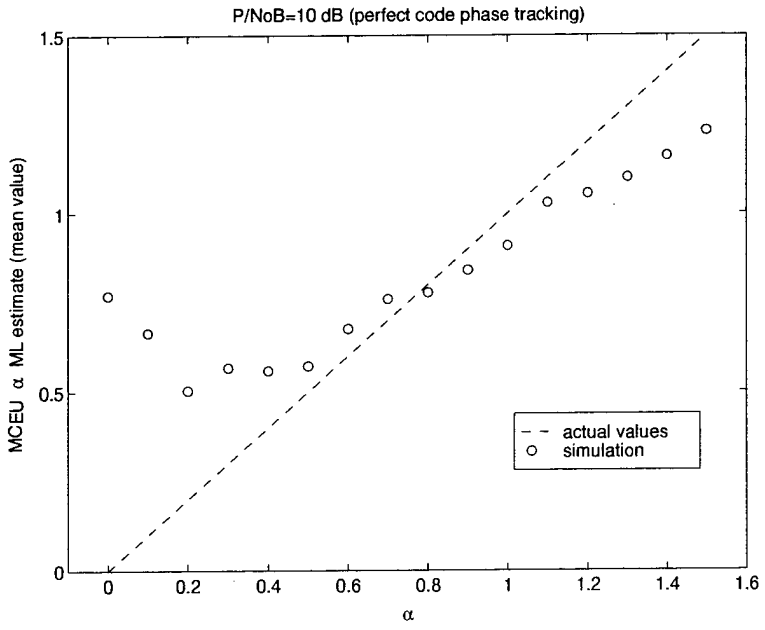

Figure 30. Simulation results showing the mean value of MCEU ML estimate, $\hat{\alpha}$, at $P/N_0 =$ 10 dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

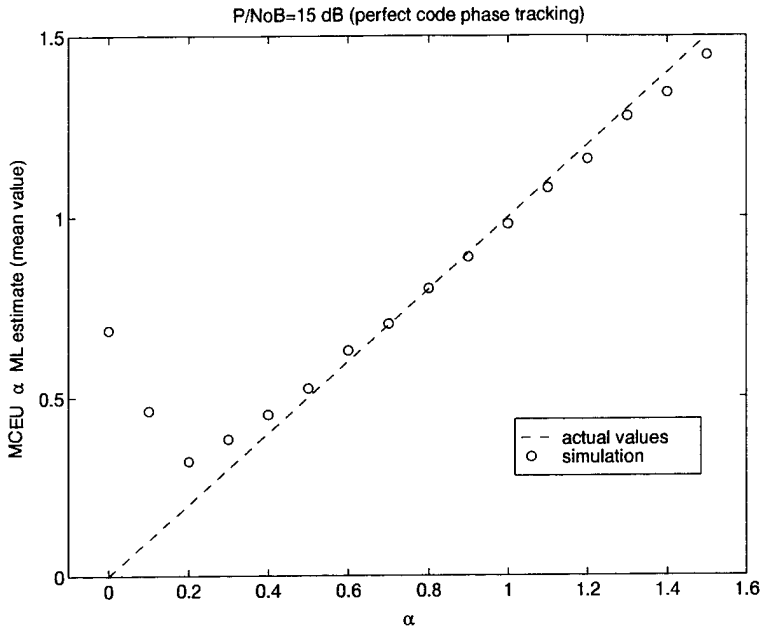

Figure 31. Simulation results showing the mean value of MCEU ML estimate, $\widehat{\alpha}$, at $P/N_0 =$ 15 dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

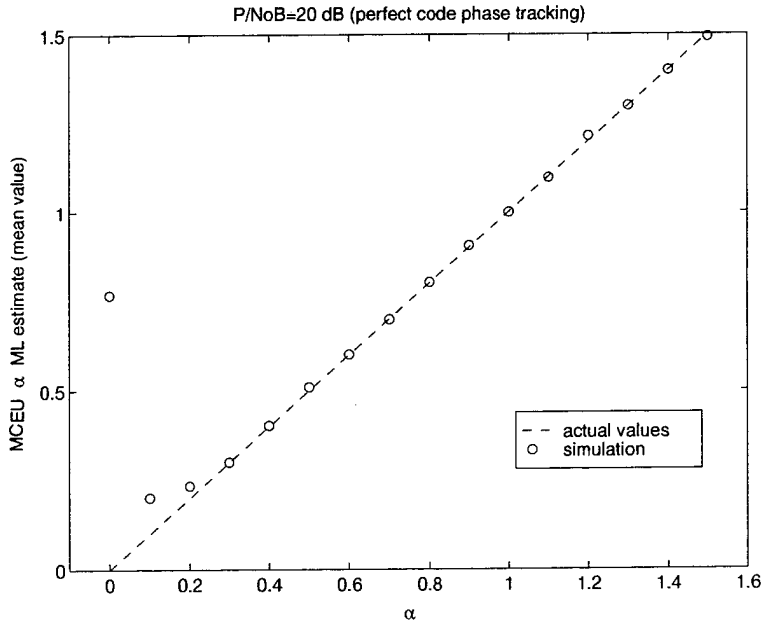

Figure 32. Simulation results showing the mean value of MCEU ML estimate, $\widehat{\alpha}$, at $P/N_0 =$ 20 dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

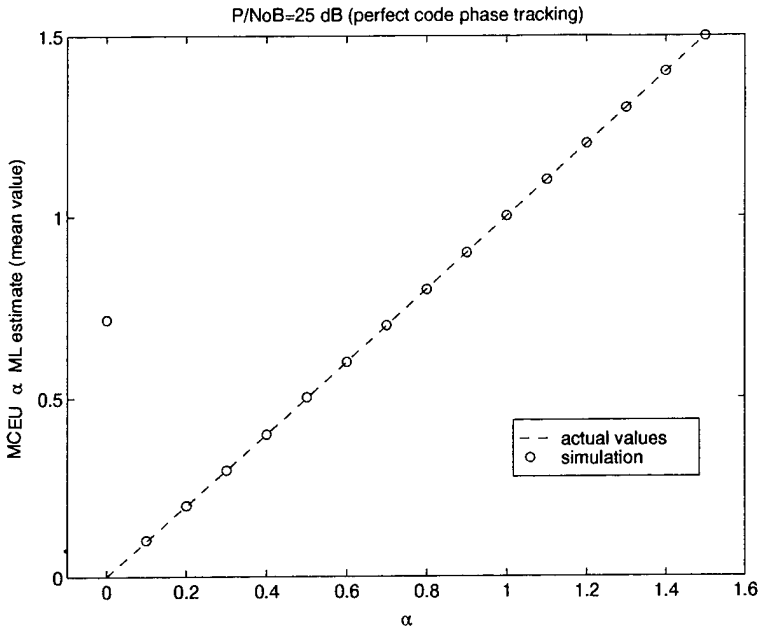

Figure 33. Simulation results showing the mean value of MCEU ML estimate, $\widehat{\alpha}$, at $P/N_0 =$ 25 dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

of $\widehat{\mathbf{x}}_k$ for $k \neq 0$ is given by (see Chapter III)

$$E\{\widehat{\mathbf{x}}_k\} = E\{\mathbf{P}(\beta_k)\mathbf{R}\} = E\{\mathbf{P}(\beta_k)[\mathbf{H}(\alpha)\mathbf{x} + \mathbf{v}]\} = \mathbf{P}(\beta_k)\mathbf{H}(\alpha)\mathbf{x}\,, \tag{96}$$

while the mean of the $k = 0$th estimate is

$$E\{\widehat{\mathbf{x}}_0\} = \begin{bmatrix} \mathbf{P}(\beta_0)\mathbf{H}(\alpha)\mathbf{x} \\ \\ 0 \end{bmatrix}. \tag{97}$$

When multipath is present, and because $\mathbf{P}(\beta_k)\mathbf{H}(\beta_k) = \mathbf{I}$ for $k \neq 0$, any given estimator for $k \neq 0$ is unbiased (i.e., $E\{\widehat{\mathbf{x}}_k\} = \mathbf{x}$) when $\alpha = \beta_k$; the estimator for $k = 0$, on the other hand, is always biased in the presence of multipath. For the direct-path only case, the estimator for $k = 0$ is unbiased because $\mathbf{P}(\beta_0)\mathbf{H}(\beta_0)\mathbf{x} = x_0$; however, each of the remaining estimators for $k \neq 0$ are also

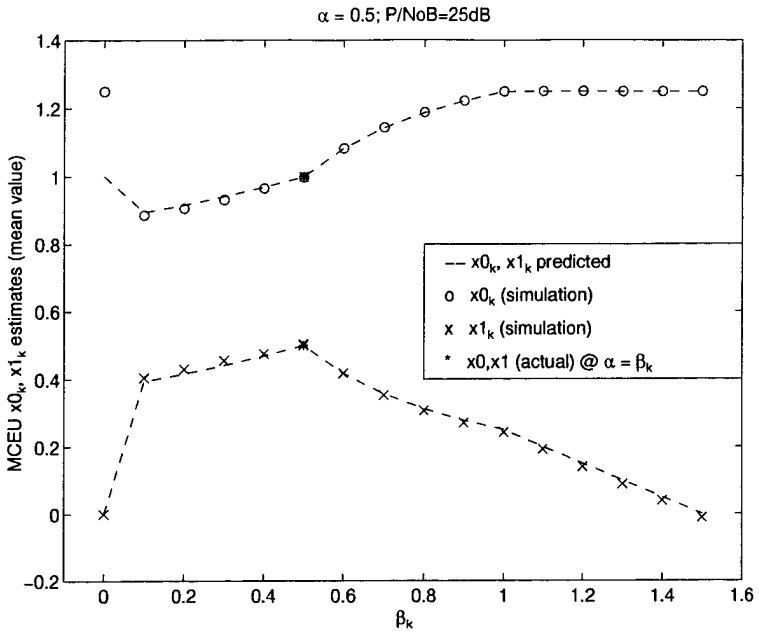

Figure 34. Simulation results showing the mean values of the MCEU estimates, $\widehat{\mathbf{x}}_k$, at $P/N_0 =$ 25 dB when $\alpha = 0.5$. (Assumes perfect MCTL code phase tracking).

unbiased since

$$\mathbf{P}(\beta_k)\mathbf{H}(0)\mathbf{x} = \begin{bmatrix} 1 & m_{12} \\ 0 & m_{22} \end{bmatrix} \begin{bmatrix} x_0 \\ 0 \end{bmatrix} = \begin{bmatrix} x_0 \\ 0 \end{bmatrix} \tag{98}$$

where m_{12} and m_{22} denote arbitrary matrix elements. The fact that the first column of $\mathbf{P}(\beta_k)\mathbf{H}(0)$ is always $[1\ 0]^T$ is due to the fact that $\mathbf{P}(\beta_k)\mathbf{H}(\beta_k) = \mathbf{I}$ and that the first column of $\mathbf{H}(\alpha)$ (see Equation 70) is the same regardless of the value of α. The preceding discussion illustrates an important property of the MCEU estimator: *when multipath is present, only the* $\widehat{\mathbf{x}}_k$ *corresponding to* $\beta_k = \alpha$ *is an unbiased estimate; when no multipath is present, all of the* $\widehat{\mathbf{x}}_k$*'s are unbiased estimates*; both cases are illustrated in Figures 34 through 35.

To predict the effect of the estimates, $\widehat{\mathbf{x}}_k$, upon the MCEU ML estimate, $\widehat{\mathbf{x}}$, we calculate the mean value of the kth error vector, $\mathbf{e}_k$, as

$$E\{\mathbf{e}_k\} = \mathbf{H}(\alpha)\mathbf{x} - \mathbf{H}(\beta_k)\, E\{\widehat{\mathbf{x}}_k\}\ . \tag{99}$$

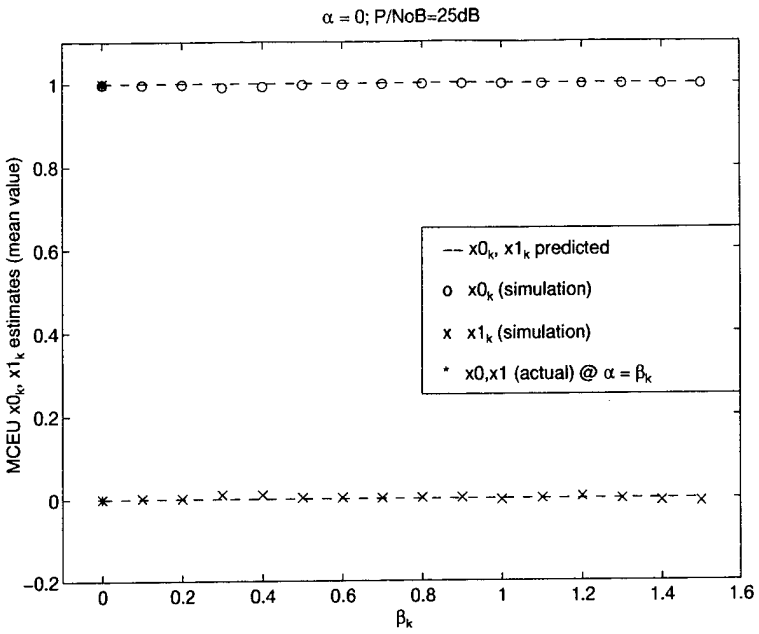

Figure 35. Simulation results showing the mean values of the MCEU estimates, $\widehat{\mathbf{x}}_k$, at $P/N_0 =$ 25 dB when $\alpha = 0$ and $x_1 = 0$ (i.e., no multipath). (Assumes perfect MCTL code phase tracking).

From the preceding paragraph we see that, when no multipath is present, $E\{\mathbf{e}_k\} = \mathbf{0}$ when $\beta_k = \alpha$. Therefore, since $E_k = \mathbf{e}_k^T \mathbf{C}^{-1} \mathbf{e}_k = 0$ maximizes the likelihood function in Equation 89, the MCEU will tend to choose the estimates corresponding to $\beta_k = \alpha$ as its ML estimates (note, however, that $E\{E_k\} \neq 0$). For the direct-path only case, $E\{\mathbf{e}_k\} = \mathbf{0}$ for *all* values of k, which makes any choice of β_k acceptable as far as the MCEU is concerned; this explains why the $\widehat{\alpha}$ estimate can exhibit a large bias while the $\widehat{x}_0$ and $\widehat{x}_1$ estimates remain accurate (as demonstrated in the simulation results). Therefore, the MRDLL can function in both the direct-path *and* multipath environments.

Now, we will examine the covariance matrix, $\mathbf{W}$, for $\widehat{\mathbf{x}}_k$, which can be shown to be (6)

$$\mathbf{W} = \sigma_{\mathbf{v}}^{\mathbf{2}} \left[\mathbf{H^T} \left(\beta_{\mathbf{k}} \right) \mathbf{C^{-1}} \mathbf{H} \left(\beta_{\mathbf{k}} \right) \right]^{\mathbf{-1}} \tag{100}$$

and is 2×2 symmetric. The $\widehat{\mathbf{x}}_k$ estimator variance (taken from the diagonal terms of $\mathbf{W}$) is plotted for different values of $P/N_0 B$ in Figures 36 through 43 along with the sample variances of the

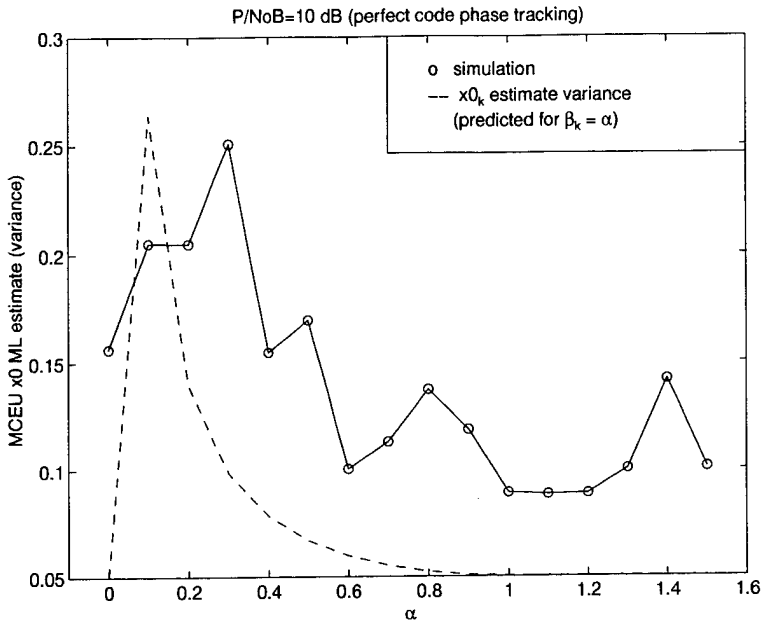

Figure 36. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_0$, at $P/N_0 = 10$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

simulated ML estimates. Note that the variance of the estimates at $\alpha = 0.1$ is significantly greater than that for the other estimates; this explains why the estimates at small relative multipath delay ($\alpha = 0.1$) were generally not as accurate as those for other values of delay. In addition, calculation of $\mathbf{W}$ for different values of β_k reveals the presence of non-zero diagonal terms for $0 < \beta_k < 1$, which implies a cross-correlation exists between $\widehat{x}_0 k$ and $\widehat{x}_1 k$; this explains the correlation observed in Figure 26.

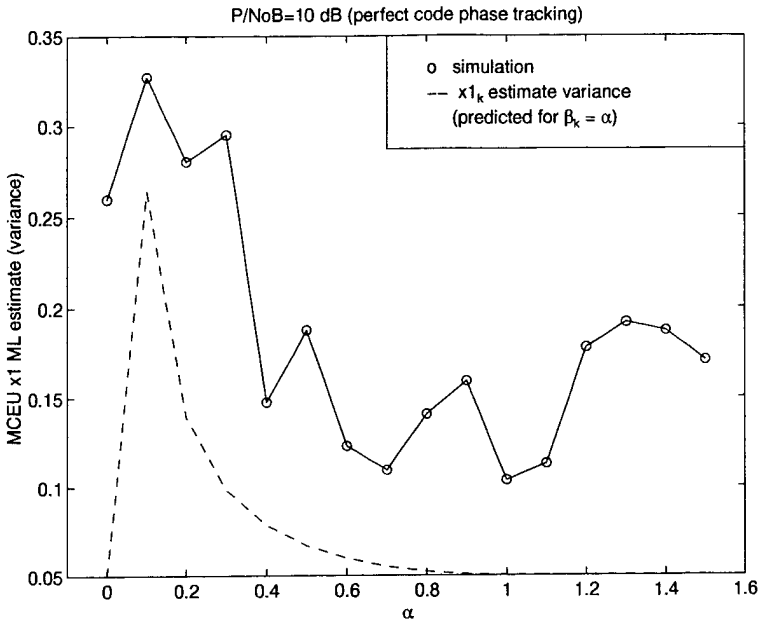

Figure 37. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_1$, at $P/N_0 = 10$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

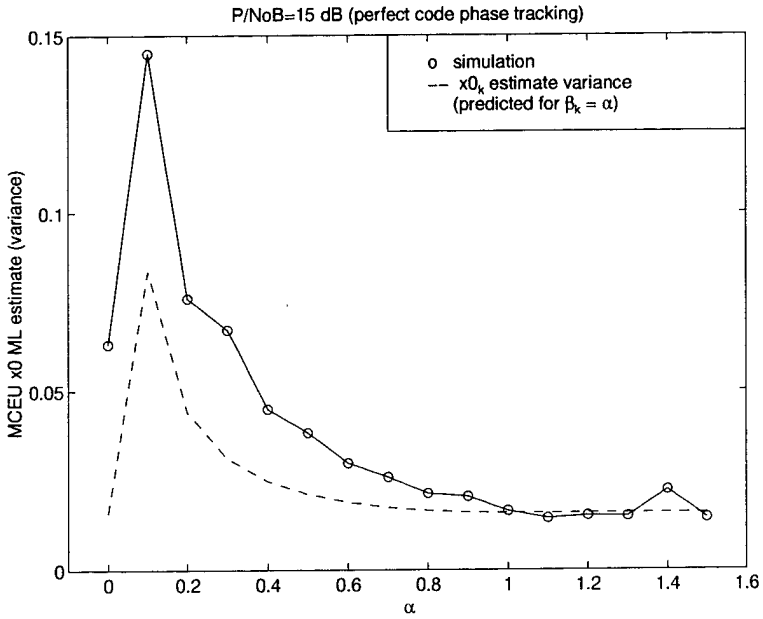

Figure 38. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_0$, at $P/N_0 = 15$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

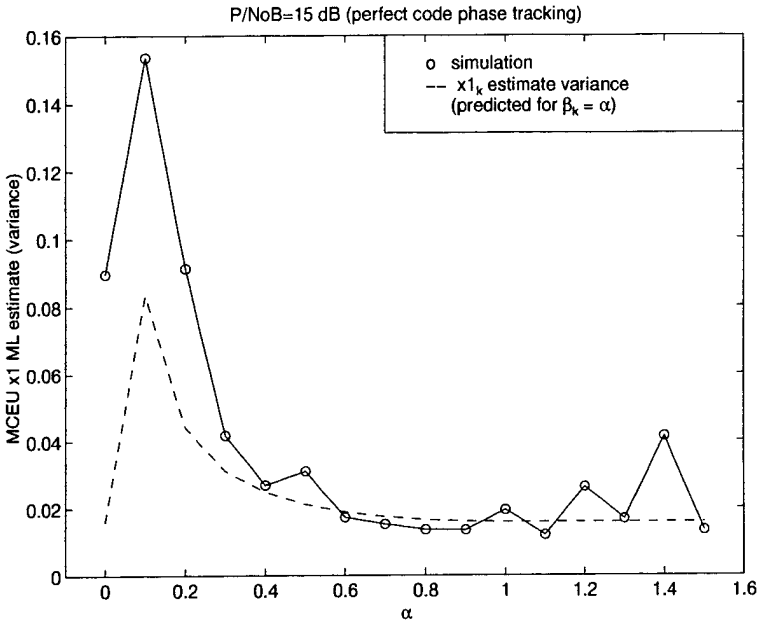

Figure 39. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_1$, at $P/N_0 = 15$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

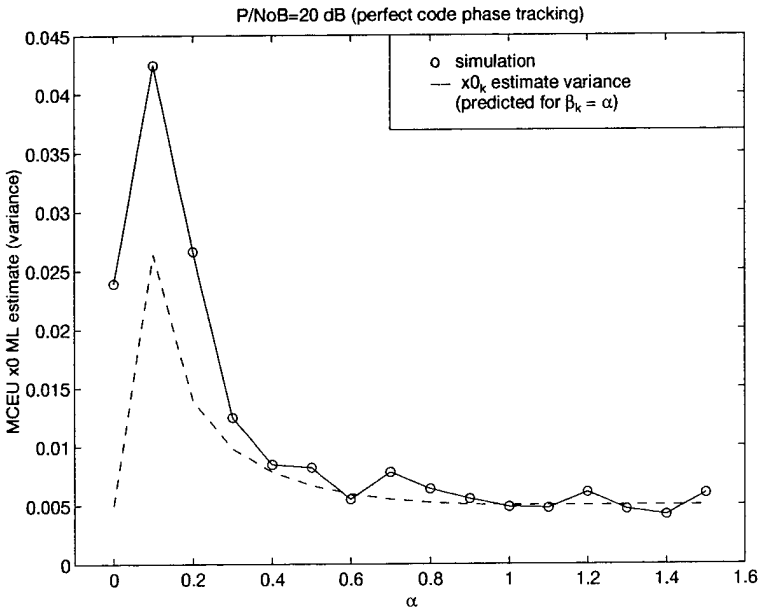

Figure 40. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_0$, at $P/N_0 = 20$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

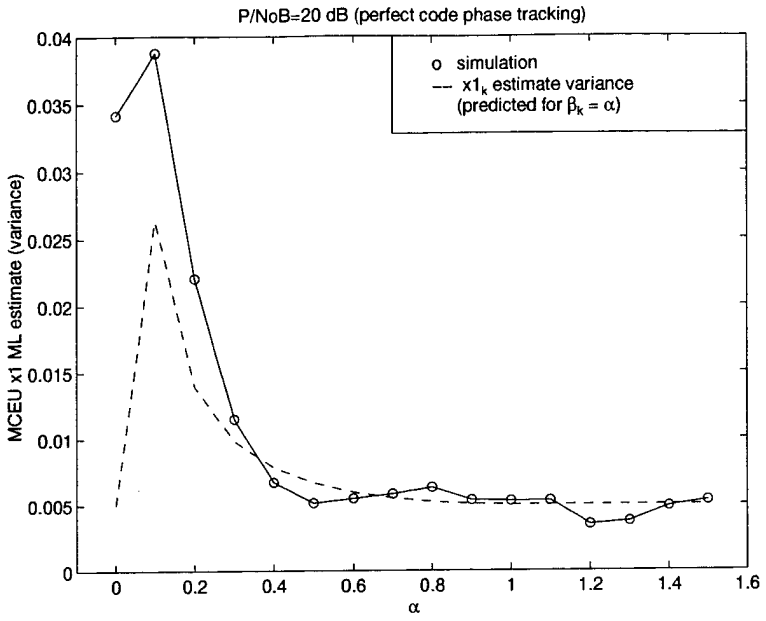

Figure 41. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_1$, at $P/N_0 = 20$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

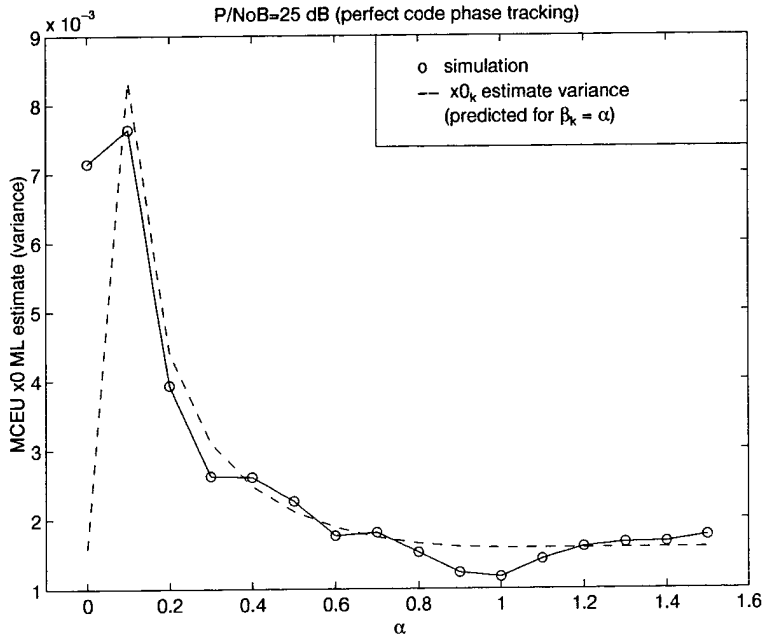

Figure 42. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_0$, at $P/N_0 = 25$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

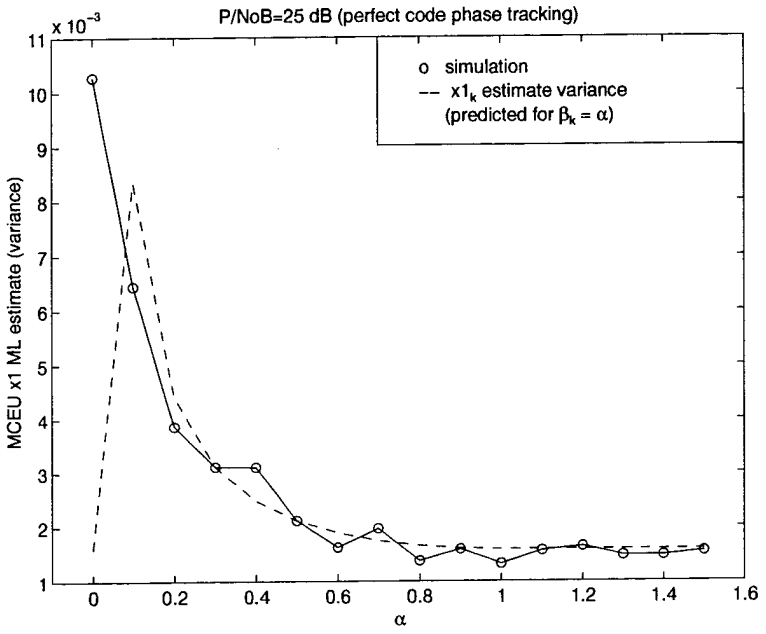

Figure 43. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_1$, at $P/N_0 = 25$ dB, for different values of relative multipath delay, α. (Assumes perfect MCTL code phase tracking).

4.5 Simulation #2: NCDLL vs MRDLL steady-state tracking performance in the presence of multipath without AWGN

4.5.1 Objectives. The effect of multipath interference on the NCDLL (neglecting noise) was discussed in Chapter II; we saw that the reflected signal introduced a steady-state code phase tracking error in the NCDLL. Then, in Chapter III we claimed that the MCTL could remove this tracking error by effectively removing the reflected signal's contribution to the loop's S-curve. However, the analysis in Chapter III assumed that the MCTL received perfect signal estimates from the MCEU; furthermore, the MCEU simulation in section 4.4 assumed zero code phase tracking error. Therefore, in order to view MRDLL behavior under conditions where these ideal conditions no longer apply, computer simulation was necessary; the objectives of this simulation were to

1. Investigate the effects of multipath interference on the MCTL steady-state tracking error with imperfect MCEU signal parameter estimation (neglecting noise effects).
2. Investigate the effects of multipath interference on the MCEU estimator when code phase timing errors are present (without AWGN).
3. Demonstrate that the MRDLL exhibits reduced steady-state code phase tracking error when compared to the NCDLL in the presence of multipath without AWGN.

4.5.2 Method. Simulations were run for both loops at different values of $\alpha > 0$ (the direct-path only simulation at $\alpha = 0$ is presented in the next section). Since we were interested in loop performance *after* acquisition, both loops were allowed to reach zero steady-state tracking error before introduction of the multipath signal; in the MRDLL, this was accomplished by feeding the MCTL perfect estimates at the start of each run, then switching to the MCEU estimates. After the introduction of the multipath signal, each loop was allowed to return to steady-state; the normalized rms steady-state timing error was then estimated over the last 100,000 output samples

using the method described in section 4.3. The sample mean and sample variance of the MCEU estimates at steady-state were also calculated over the last 100,000 output samples.

4.5.3 Results. The normalized rms steady-state tracking error resulting from the NCDLL and MRDLL simulations is plotted versus α in Figure 44. This plot shows that:

- The MRDLL exhibits a smaller tracking error than the NCDLL for all values of α; also note that the improvement is most significant at values of α which introduce the greatest error in the NCDLL.
- Since we no longer have perfect MCEU estimation, the MRDLL exhibits a non-zero steady-state error.
- The MRDLL steady-state error peaks at $\alpha = 0.6$, much in the same way the NCDLL error peaks between $\alpha = 0.7$ and $\alpha = 0.8$.
- The NCDLL simulation results agree almost exactly with the multipath tracking error predicted by Equation 32.

MCEU estimator performance is plotted in Figures 45 through 49; in addition, several plots showing the MCEU output over time for selected values of α are presented in Figures 50 through 55. Plots showing the MCTL initial transient response after introduction of the multipath signal are also shown in Figures 56 through 63. (Note that the time scaling on the MCEU plots is based on the simulation parameter of $T_c = 0.1$ seconds; keep in mind that actual GPS time would scale our time axis by a factor on the order of 10^{-5} seconds since, assuming C/A code, the actual GPS code chip period is approx 10^{-6}). From the simulation results, observe that:

- As expected, the imperfect code phase estimation introduced a bias into the MCEU ML signal estimates, $\widehat{x}_0$ and $\widehat{x}_1$.
- For the $\widehat{x}_1$ estimate, the bias was relatively large at $\alpha = 0.6$, corresponding to the peak steady-state timing error observed in Figure 44.

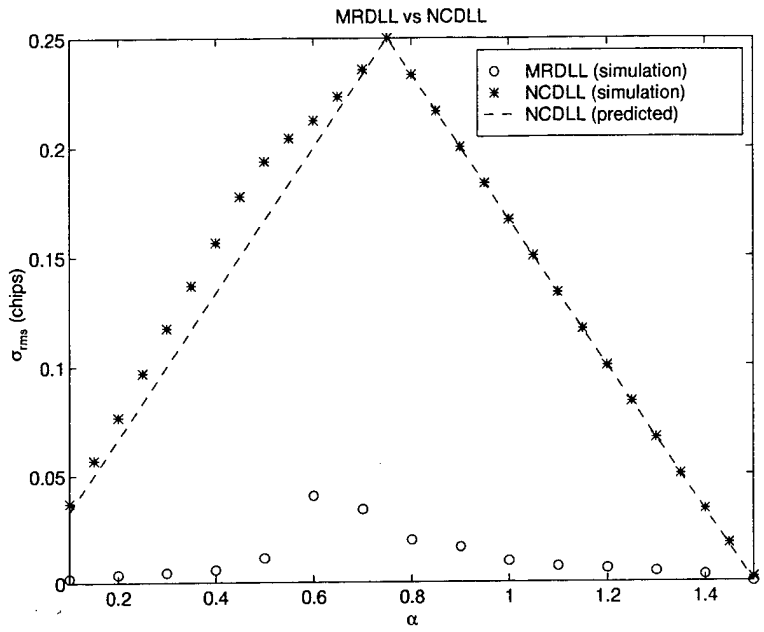

Figure 44. Simulation results comparing MRDLL and NCDLL rms steady-state tracking error performance for different values of relative multipath delay, α.

- The variance of the $\widehat{x}_0$ and $\widehat{x}_1$ ML estimates tended to increase with increasing α; this is the opposite of what was observed in Simulation #1.
- There were significant increases at $\alpha = 0.6$ in the variances of $\widehat{x}_0$ and $\widehat{\alpha}$; in fact the $\widehat{\alpha}$ ML estimate when $\alpha = 0.6$ was the only $\widehat{\alpha}$ with a non-zero variance at steady-state.
- The MCEU outputs display an oscillatory behavior at steady-state, which is likely due to the presence of adaptation in the MCTL loop filter (via the ALC).

The most significant result from these simulations is that, despite a non-zero steady-state tracking error, *the MRDLL demonstrates a dramatic improvement over the NCDLL in code phase tracking error.* The MRDLL non-zero steady state error exists because of the interdependence between the MCEU estimates and the MCTL code phase estimate; the code phase error, δ, affects the accuracy of the $\widehat{\mathbf{x}}$ estimate and vice-versa, which is evident in the oscillatory nature of the MCEU output. Furthermore, the peaks in the steady-state error at $\alpha = 0.6$ may be due to the fact that the magnitude of the peak transient error varied with α, and was greatest at $\alpha = 0.6$; perhaps

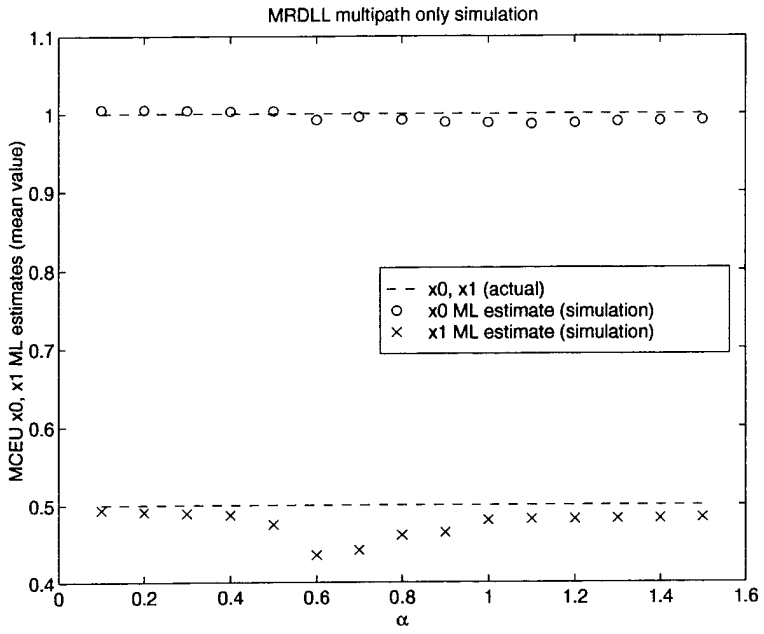

Figure 45. Simulation results showing the mean value of MCEU ML estimate, $\widehat{\mathbf{x}}$ for different values of relative multipath delay, α (neglects noise effects).

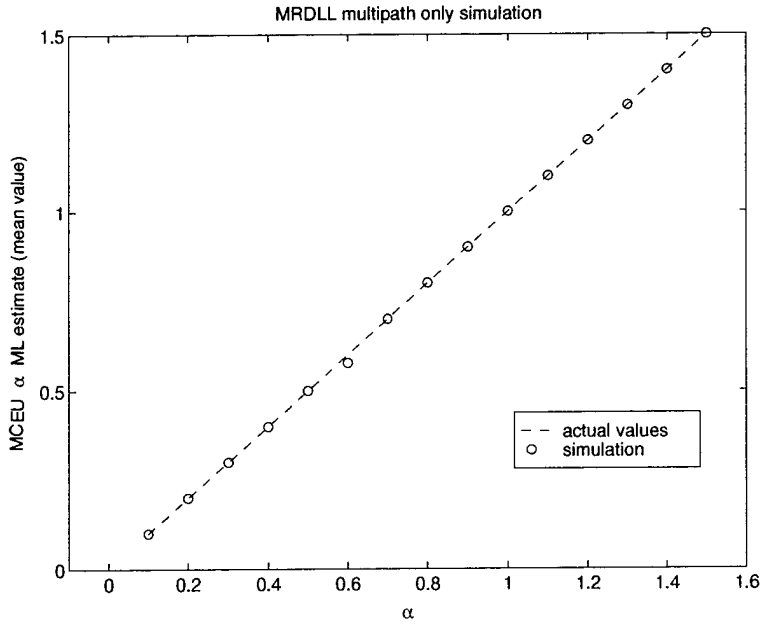

Figure 46. Simulation results showing the mean value of MCEU ML estimate, $\widehat{\alpha}$, for different values of relative multipath delay, α (neglects noise effects).

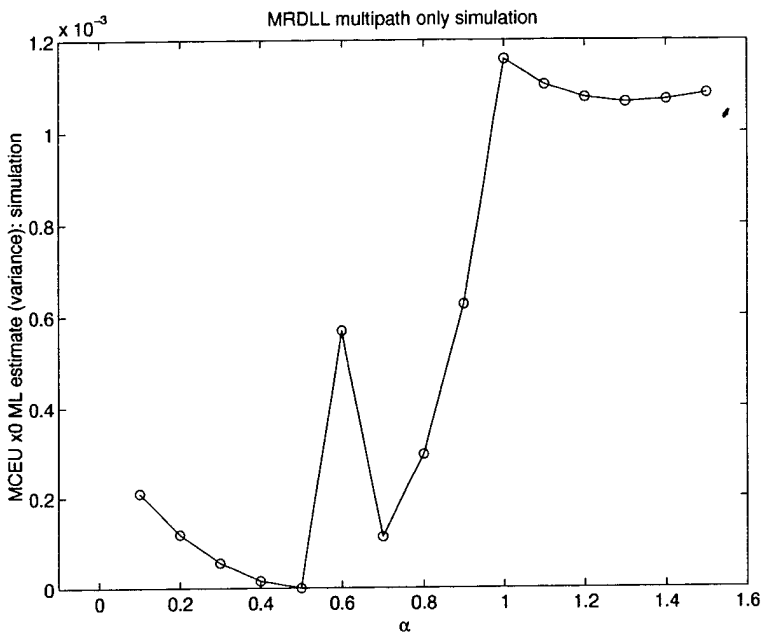

Figure 47. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_0$, for different values of relative multipath delay, α (neglects noise effects).

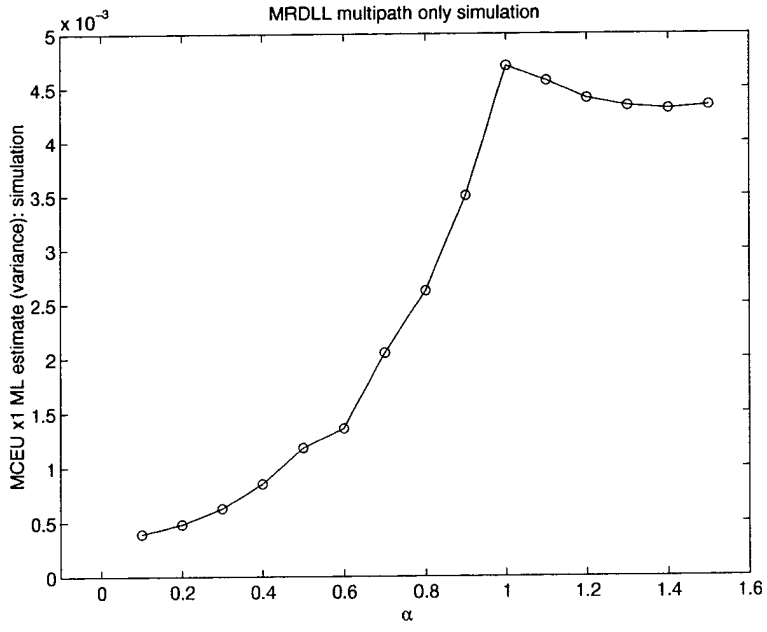

Figure 48. Simulation results showing the variance of MCEU ML estimate, $\widehat{x}_1$, for different values of relative multipath delay, α (neglects noise effects).

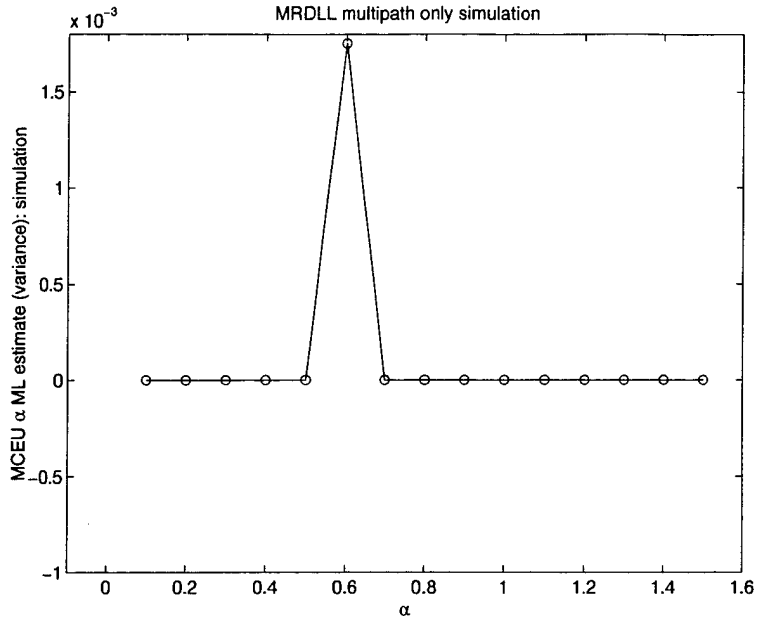

Figure 49. Simulation results showing the variance of MCEU ML estimate, $\widehat{\alpha}$, for different values of relative multipath delay, α (neglects noise effects).

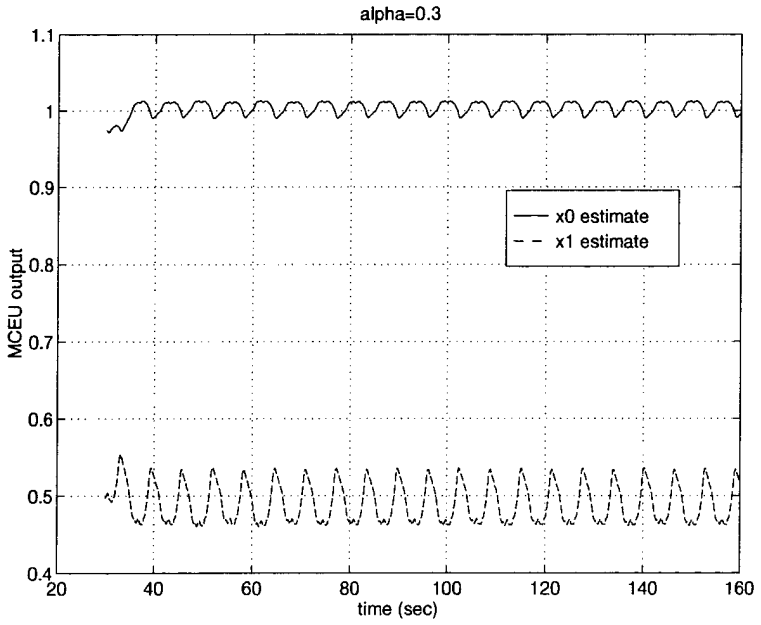

Figure 50. Simulation results showing MCEU estimator output, $\widehat{\mathbf{x}}$, over time for $\alpha = 0.3$ (no noise).

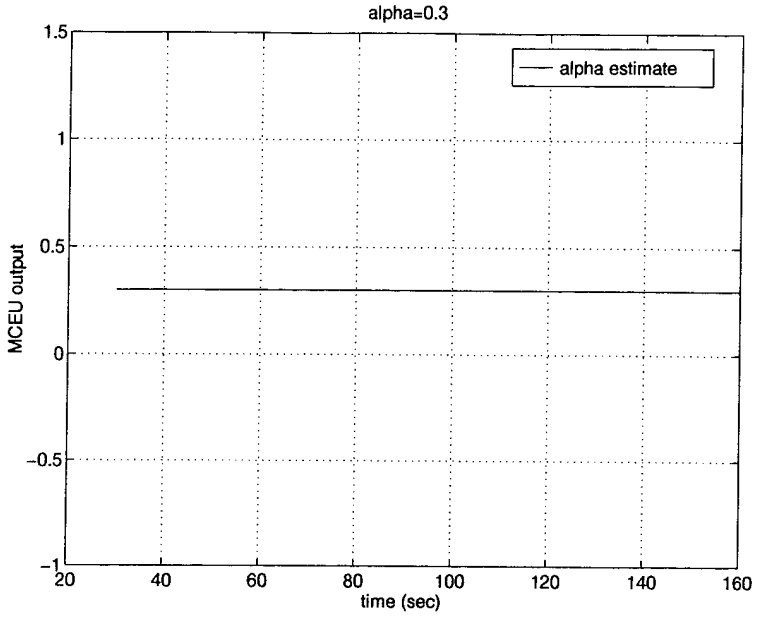

Figure 51. Simulation results showing MCEU estimator output, $\widehat{\alpha}$, over time for $\alpha = 0.3$ (no noise).

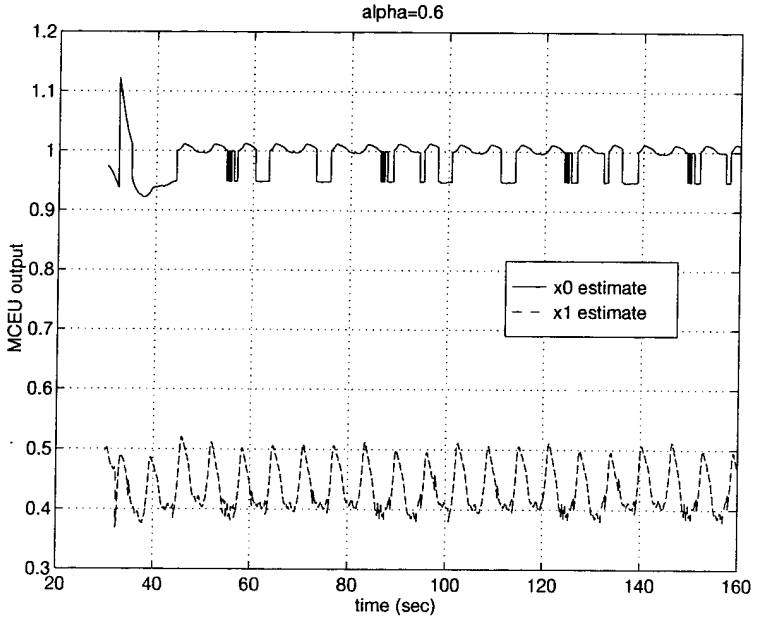

Figure 52. Simulation results showing MCEU estimator output, $\widehat{\mathbf{x}}$, over time for $\alpha = 0.6$ (no noise).

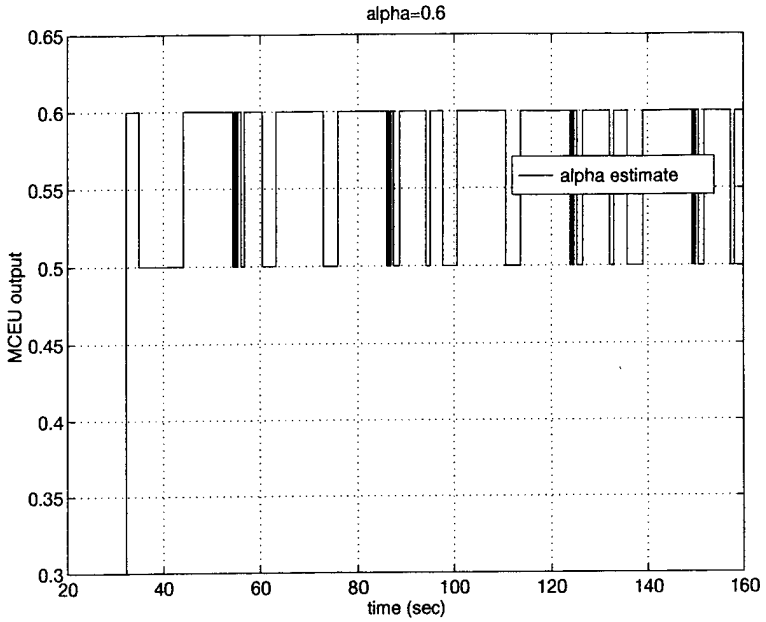

Figure 53. Simulation results showing MCEU estimator output, $\widehat{\alpha}$, over time for $\alpha = 0.6$ (no noise).

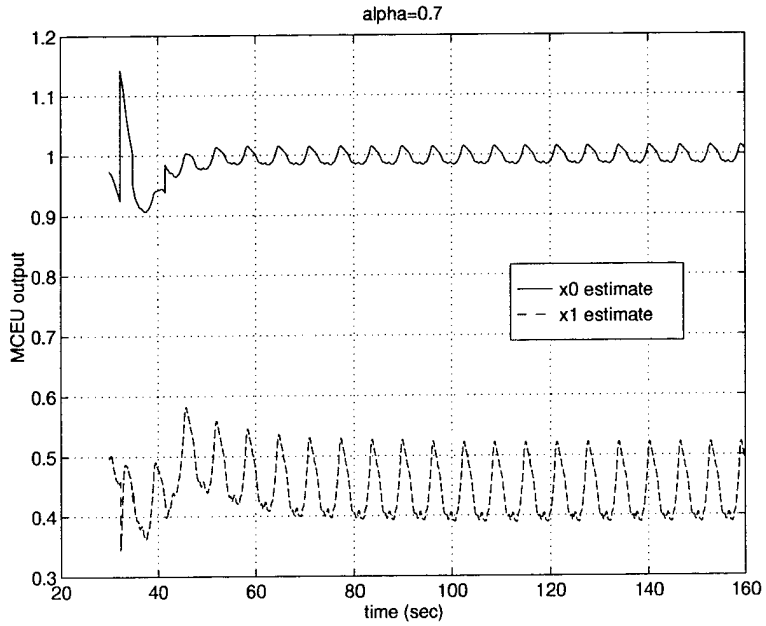

Figure 54. Simulation results showing MCEU estimator output, $\widehat{\mathbf{x}}$, over time for $\alpha = 0.7$ (no noise).

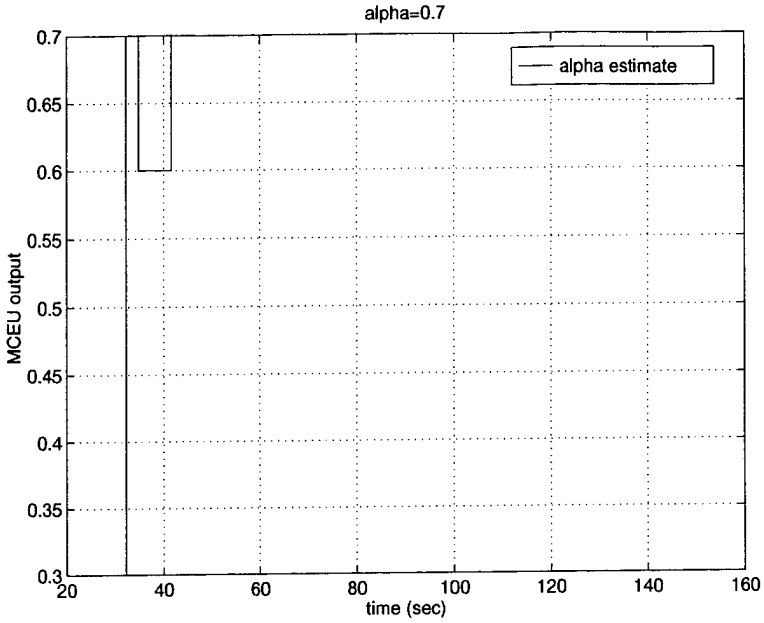

Figure 55. Simulation results showing MCEU estimator output, $\widehat{\alpha}$, over time for $\alpha = 0.7$ (no noise).

in this case, an error threshold was exceeded that prevented the code phase and signal estimates from attaining more accurate steady state values (compare the MCEU outputs at steady state for $\alpha = 0.6$ to those for $\alpha = 0.3$ and $\alpha = 0.7$).

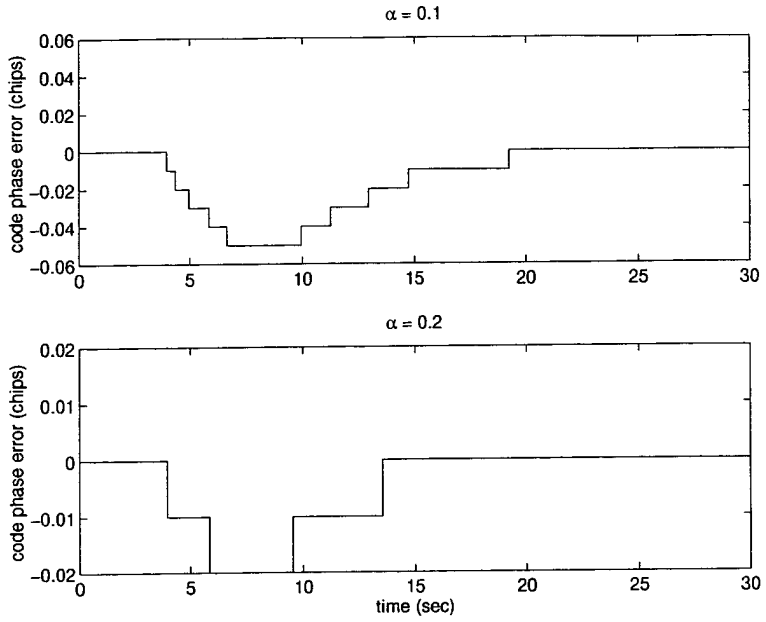

Figure 56. Simulation results showing MRDLL transient code phase timing error for $\alpha = 0.1$ and $\alpha = 0.2$ (no noise).

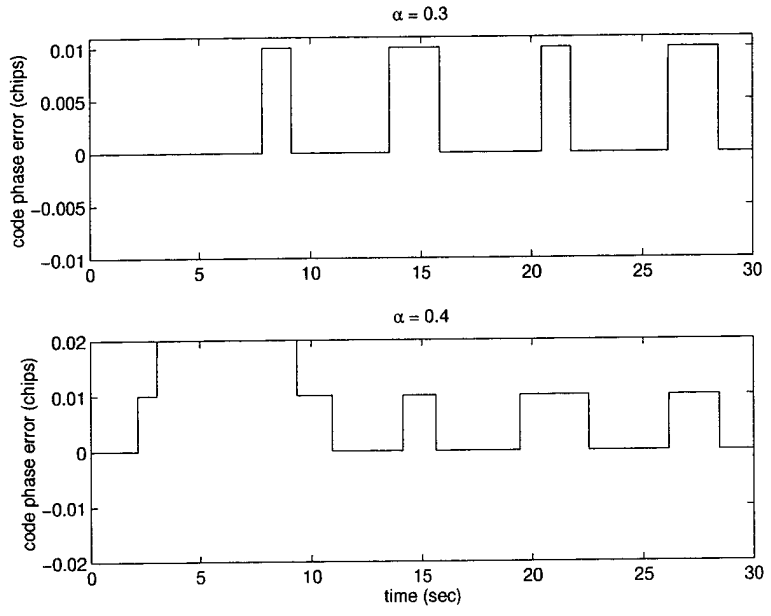

Figure 57. Simulation results showing MRDLL transient code phase timing error for $\alpha = 0.3$ and $\alpha = 0.4$ (no noise).

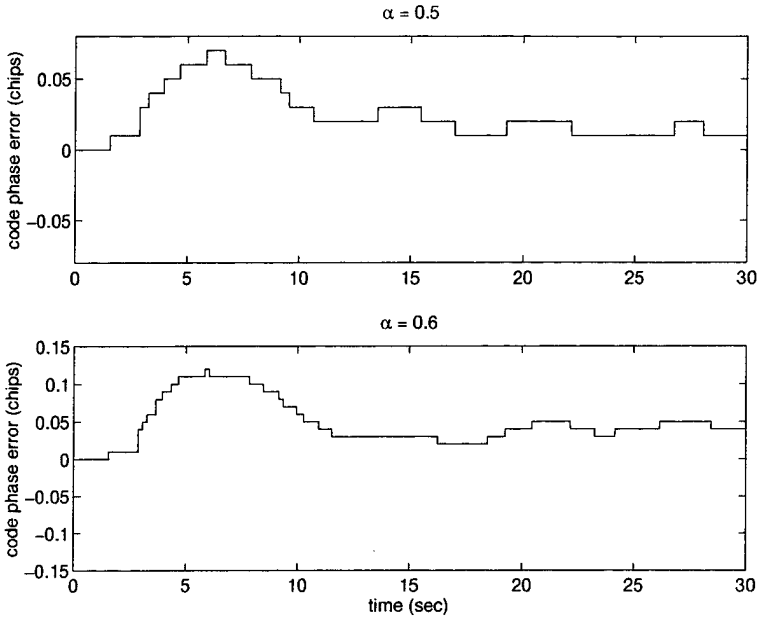

Figure 58. Simulation results showing MRDLL transient code phase timing error for $\alpha = 0.5$ and $\alpha = 0.6$ (no noise).

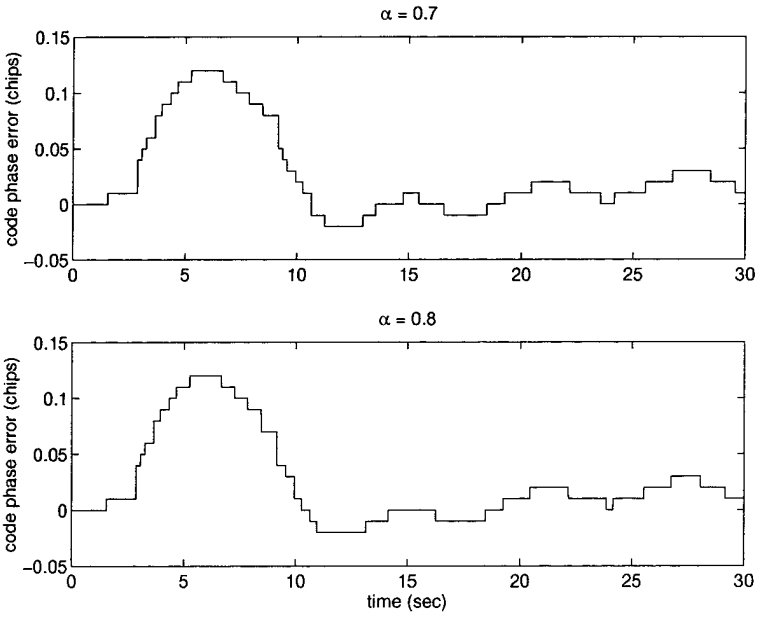

Figure 59. Simulation results showing MRDLL transient code phase timing error for $\alpha = 0.7$ and $\alpha = 0.8$ (no noise).

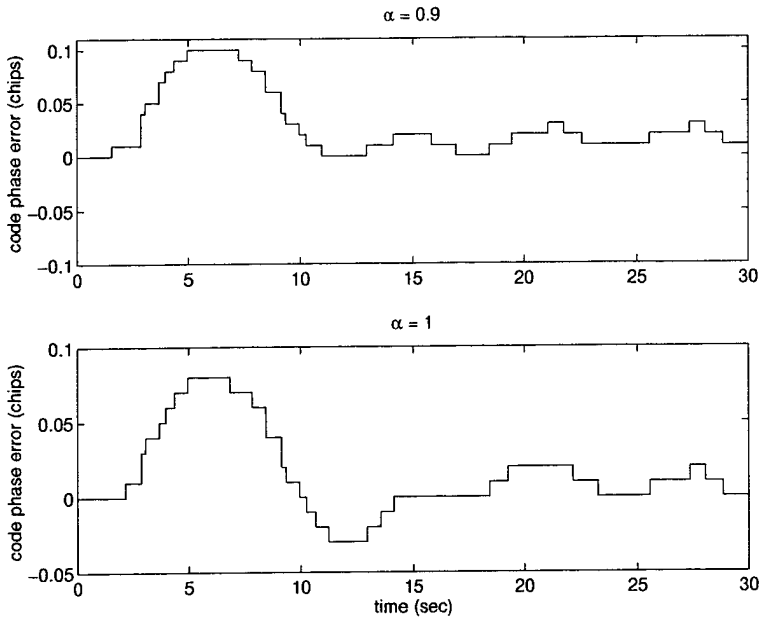

Figure 60. Simulation results showing MRDLL transient code phase timing error for $\alpha = 0.9$ and $\alpha = 1.0$ (no noise).

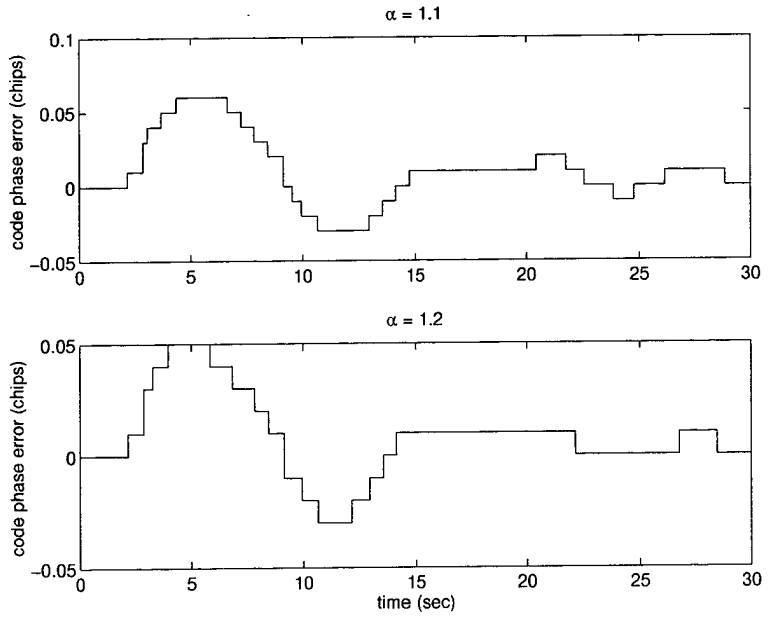

Figure 61. Simulation results showing MRDLL transient code phase timing error for $\alpha = 1.1$ and $\alpha = 1.2$ (no noise).

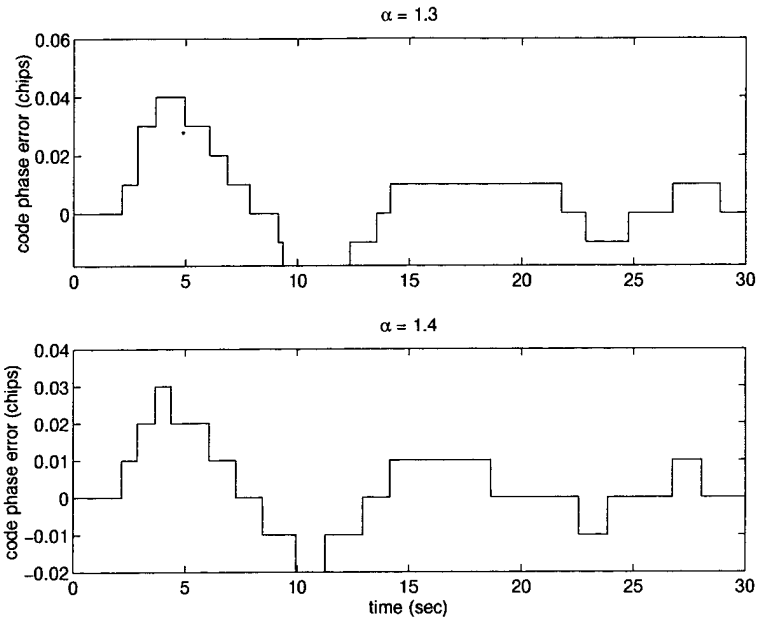

Figure 62. Simulation results showing MRDLL transient code phase timing error for $\alpha = 1.3$ and $\alpha = 1.4$ (no noise).

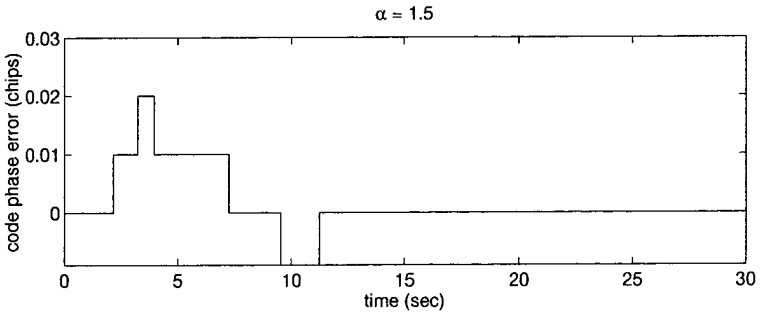

Figure 63. Simulation results showing MRDLL transient code phase timing error for $\alpha = 1.5$ (no noise).

4.6 Simulation #3: MRDLL vs NCDLL steady-state tracking performance in a direct-path only environment without AWGN.

4.6.1 Objective. As we saw from the previous simulation, the MRDLL exhibits a significant improvement in multipath code-phase tracking error over the NCDLL; however, the MRDLL must also be able to track the GPS code phase when no reflected signal is present to have any practical application. The objective of this simulation was to examine MRDLL steady-state tracking error when no multipath is present, and compare it to that of the NCDLL.

4.6.2 Method. This simulation was identical to Simulation #2; the only difference was that a direct-path signal (i.e., $\alpha = 0$ and $x_1 = 0$) was introduced after each loop entered the initial steady-state. The rms tracking error and MCEU estimator means and variances were calculated over 100,000 samples as in Simulation #2.

4.6.3 Results. Simulation results are presented in Table 5. Note from the table that the MRDLL exhibits a small but non-zero steady-state code tracking error while for the NCDLL, $\sigma_{rms} = 0$; therefore, *the NCDLL is a better choice for code phase tracking in the direct-path only environment.* The reason for this is simple: the MRDLL must estimate the amplitude (or existence) of the reflected signal, whereas the NCDLL design already assumes that no multipath is present.

Observing the MCEU ML estimates, note that despite the introduction of code phase tracking error, $\widehat{x}_0 \approx 1$ and $\widehat{x}_1 \approx 0$; however, the $\widehat{\alpha}$ estimate shows a significant bias and a relatively high variance. These results are similar to the direct-path results of the MATLAB MCEU simulation in section 4.4.

Table 5. MRDLL vs NCDLL in direct-path only environment ($x_0 = 1$, $x_1 = 0$, $\alpha = 0$): simulation results.

Parameter	MRDLL	NCDLL
σ_{rms} (chips)	0.0036	0.0000
$\widehat{x}_0$ (mean)	0.9871	N/A
$\widehat{x}_1$ (mean)	-0.0158	N/A
$\widehat{\alpha}$ (mean)	1.2376	N/A
$\widehat{x}_0$ (variance)	2.6432×10^{-6}	N/A
$\widehat{x}_1$ (variance)	.0056	N/A
$\widehat{\alpha}$ (variance)	.0634	N/A

4.7 Simulation #4: MRDLL steady-state tracking performance in the presence of multipath with AWGN

4.7.1 Objectives. Now that we have demonstrated the MRDLL's ability to significantly reduce code phase tracking errors due to multipath, we desire to investigate the effects of AWGN on MRDLL tracking performance in the multipath environment. Recall that the noise analysis and models in Chapter III assumed perfect MCEU estimation and MCTL code phase tracking in the S-curve's linear operating region. In practical tracking loops, linear tracking occurs only when the SNR exceeds a certain threshold; computer simulation is a useful tool for both demonstrating this threshold effect as well as confirming theoretical predictions based on the assumption of linear tracking and perfect estimation. The objectives of this simulation were to:

1. Investigate MRDLL non-linear behavior in the presence of multipath with AWGN, demonstrating the SNR threshold effect.
2. Compare MRDLL simulated linear tracking performance in AWGN without perfect MCEU estimation to that predicted by Equation 61 for perfect estimation.
3. Evaluate the effects of non-linear tracking on the MCEU estimator.

4.7.2 Method. Simulations were run for different values of MRDLL loop SNR, ρ_{eff} (in dB) and five different values of α = (0.4, 0.5, 0.7, 1.1, 1.5); since we were investigating linear versus non-linear behavior and since the MCTL linear region varied with α, we chose one value of α from each of the five linear regions described in Table 2. As in the previous loop simulations, the MRDLL was allowed to reach steady-state prior to introduction of the multipath signal; once the MCTL returned to steady-state, σ_{rms} and MCEU estimator mean and variance were calculated over the last 20,000 output samples.

4.7.3 Results. Simulation results plotting σ_{rms} versus ρ_{eff} are shown in Figures 64 through 68; this data yields the following observations:

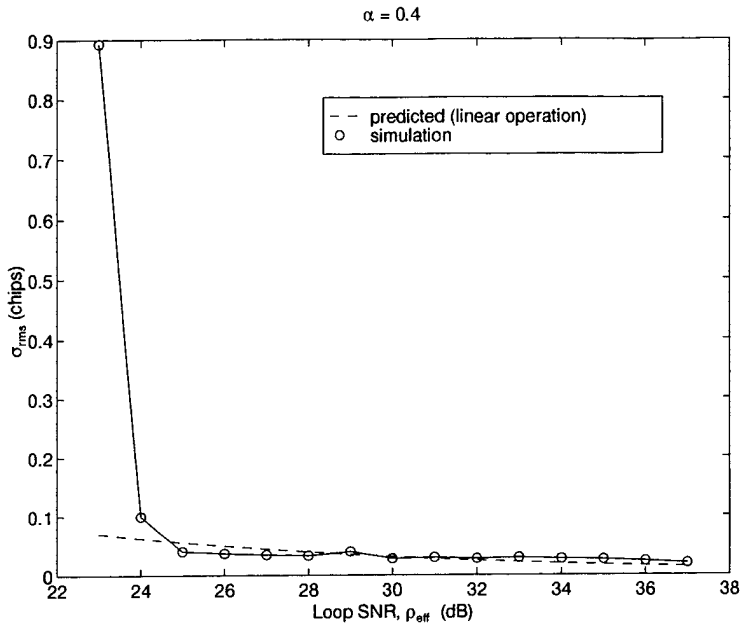

Figure 64. Simulation results showing estimated steady-state tracking error, σ_{rms} (in chips), for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.4$.

- For each different value of α, there there is an obvious loop SNR threshold (in dB) at which the loop tends toward linear tracking behavior (denoted by the dashed line).
- The SNR threshold is not constant, but depends on the value of α.
- Simulation results in the linear region closely agree with those predicted by Equation 61.

Figures 69 through 78 plot the means of the MCEU ML estimates, along with the results from Simulation #1 for comparison; (the variances of the ML estimates are included in Appendix C). Also, Figures 79 through 82 represent typical MCEU outputs versus time for both low and high values of ρ_{eff}. From the MCEU estimator plots, observe that:

- The ML estimates exhibit a significant bias in the non-linear tracking region, and the point at which linear tracking begins is clearly evident from the plots for $\widehat{x}_0$ and $\widehat{x}_1$; for a given α, note how this point corresponds to the SNR threshold in Figures 64 through 68.

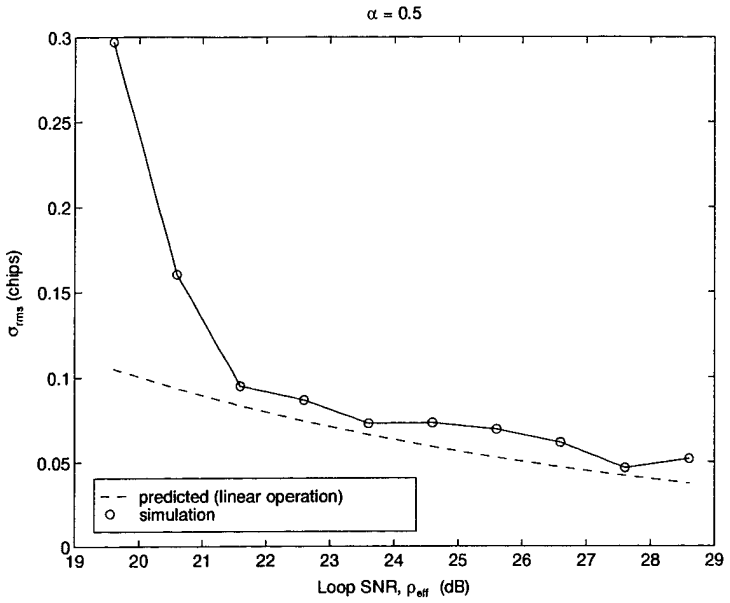

Figure 65. Simulation results showing estimated steady-state tracking error, σ_{rms} (in chips), for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.5$.

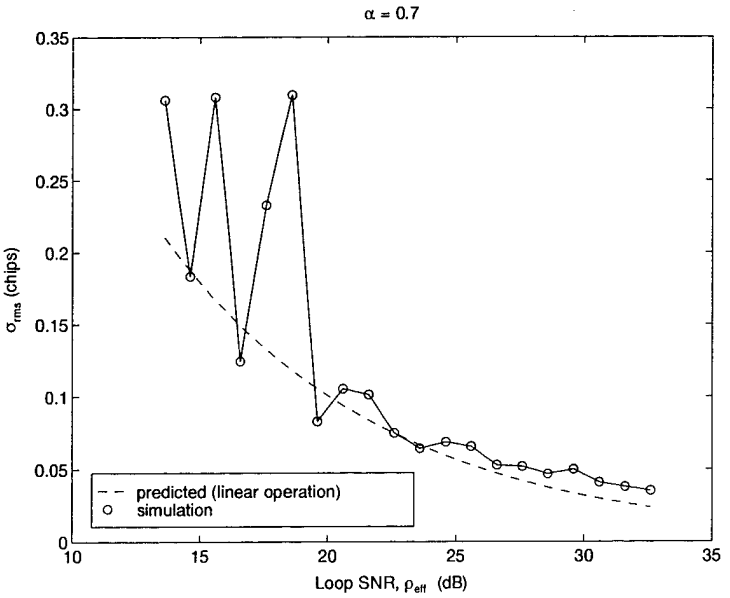

Figure 66. Simulation results showing estimated steady-state tracking error, σ_{rms} (in chips), for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.7$.

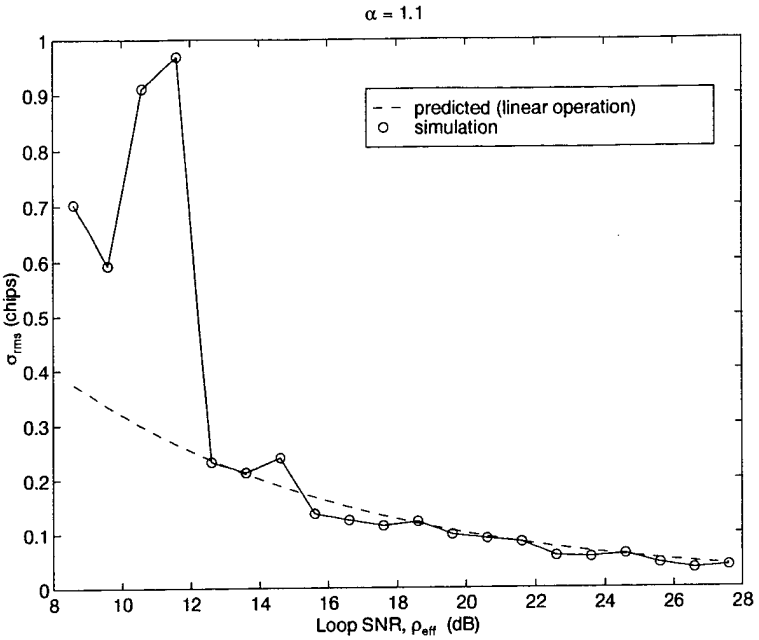

Figure 67. Simulation results showing estimated steady-state tracking error, σ_{rms} (in chips), for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.1$.

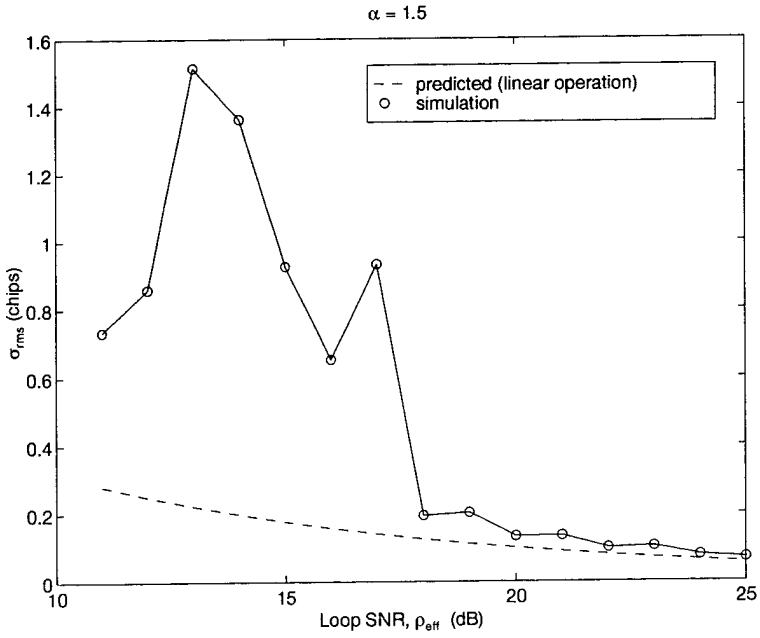

Figure 68. Simulation results showing estimated steady-state tracking error, σ_{rms} (in chips), for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.5$.

- Above the SNR threshold, the code phase tracking error maintained linear performance even when signal estimates were relatively poor.
- As expected, code phase estimation errors generally introduce an additional bias into the mean ML estimates (in addition to that introduced by AWGN); this is seen by comparing the data points corresponding to perfect code phase estimation with AWGN (Simulation #1).

Simulation results show that code phase tracking in the linear regions followed theoretical predictions very closely; the non-linear behavior of the MRDLL, however, is difficult to predict by theory, but one can hypothesize that the fact that the loop SNR threshold varied with α follows from the fact that the MCTL linear operating region varies with α as well (although the complete tracking region of positive slope, termed the affine region, does not depend upon α and is always $|\delta| \leq 0.5$). For instance, note that the threshold for $\alpha = 0.4$ entered the linear tracking region at a loop SNR approximately 5 dB higher than that for $\alpha = 0.5$; this corresponds to the fact that the MCTL's linear tracking region when $\alpha = 0.4$ (in terms of normalized tracking error, δ) is one-fifth that of $\alpha = 0.5$ (see Table 2). Also note that the simulations at $\alpha = 1.1$ and $\alpha = 1.5$ generally demonstrated a lower SNR threshold than the other three values of α; this is most likely due to the fact that the MCTL at these values of α has a relatively large linear operating region ($\delta = \pm 0.4$ chips and $\delta = \pm 0.5$ chips,respectively) and the fact that transient and steady-state tracking errors are lower (as seen in Simulation #2). Finally, the fact that linear phase tracking performance (above the SNR threshold) was achieved even when signal estimates were poor, shows that SNR and not estimator performance is the limiting factor for linear loop performance. This particular result demonstrates that this simulation would be an excellent simulation to perform with real GPS signal parameters, as it would help determine limits on the MRDLL operational environment in terms of received signal SNR.

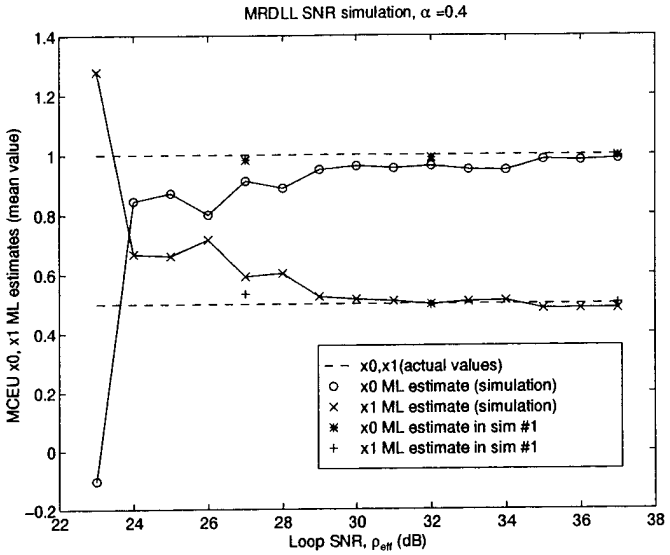

Figure 69. Simulation results showing MCEU ML estimate, $\widehat{\mathbf{x}}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.4$.

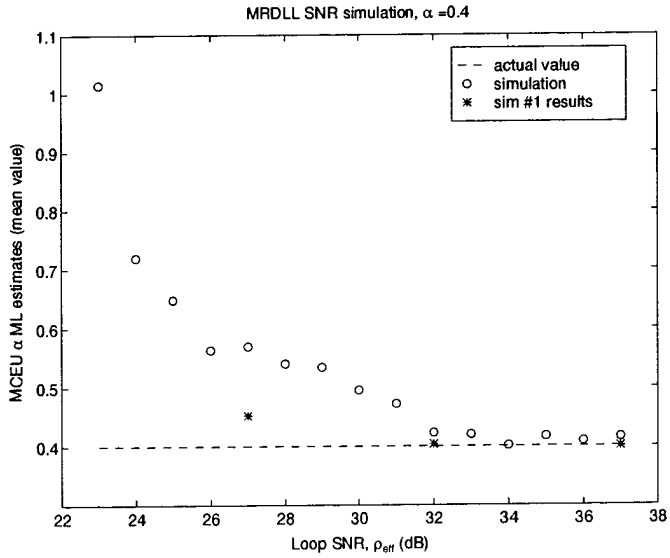

Figure 70. Simulation results showing MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.4$.

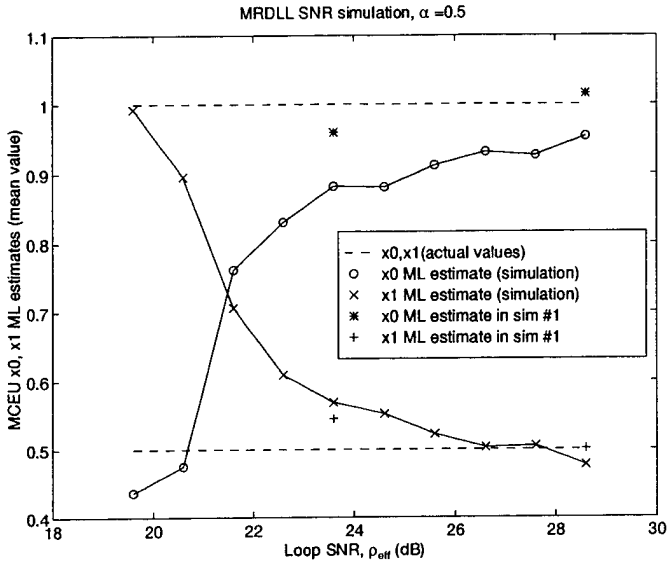

Figure 71. Simulation results showing MCEU ML estimate, $\widehat{\mathbf{x}}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.5$.

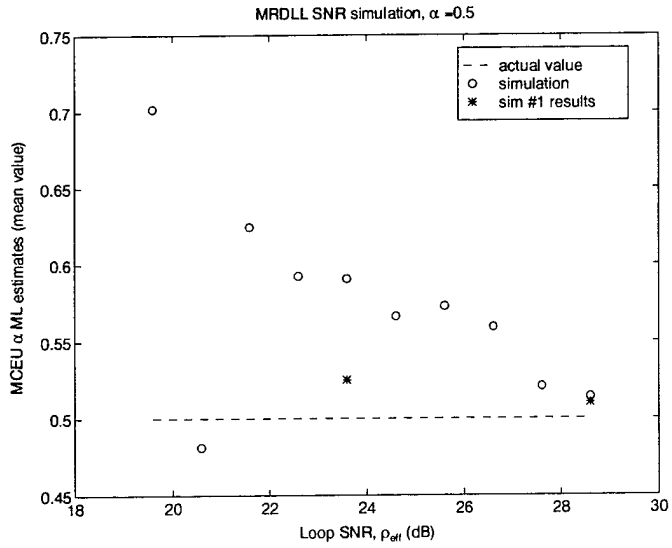

Figure 72. Simulation results showing MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.5$.

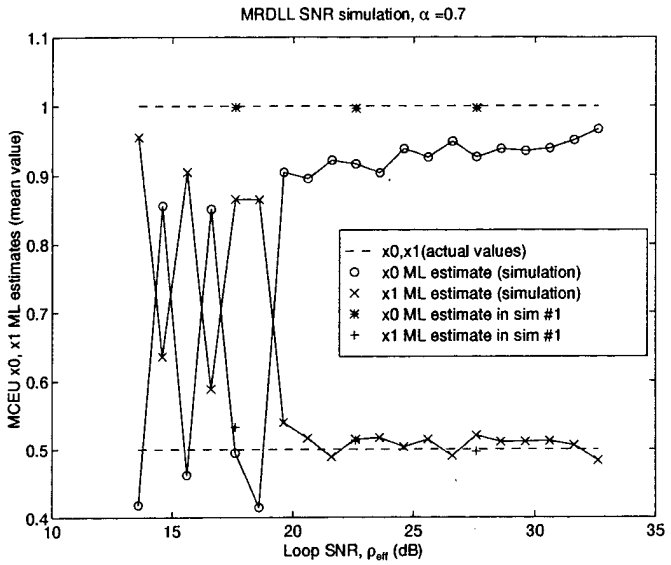

Figure 73. Simulation results showing MCEU ML estimate, $\widehat{\mathbf{x}}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.7$.

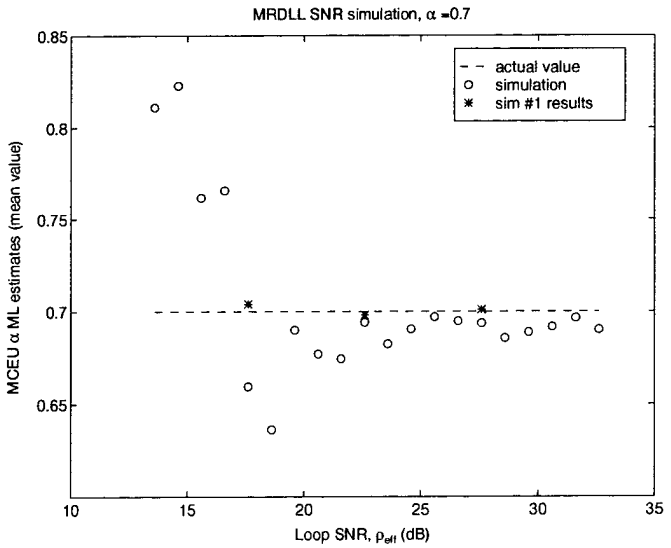

Figure 74. Simulation results showing MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.7$.

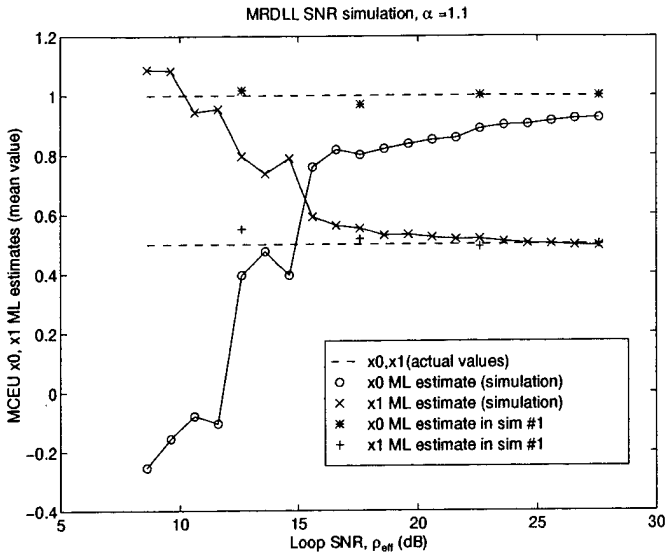

Figure 75. Simulation results showing MCEU ML estimate, $\widehat{\mathbf{x}}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.1$.

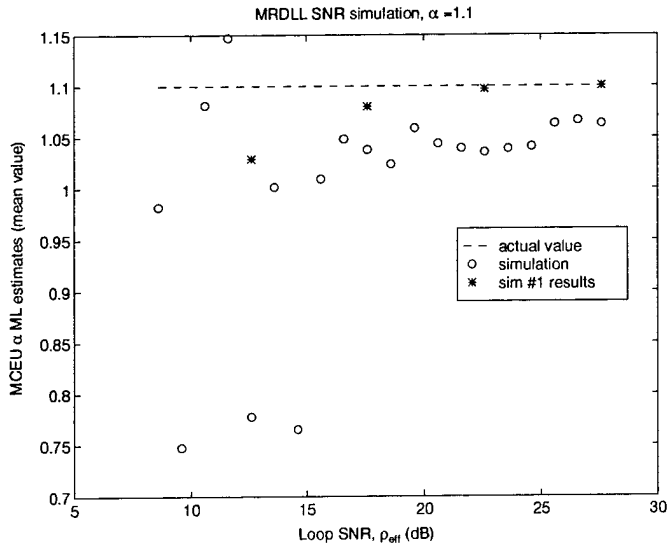

Figure 76. Simulation results showing MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.1$.

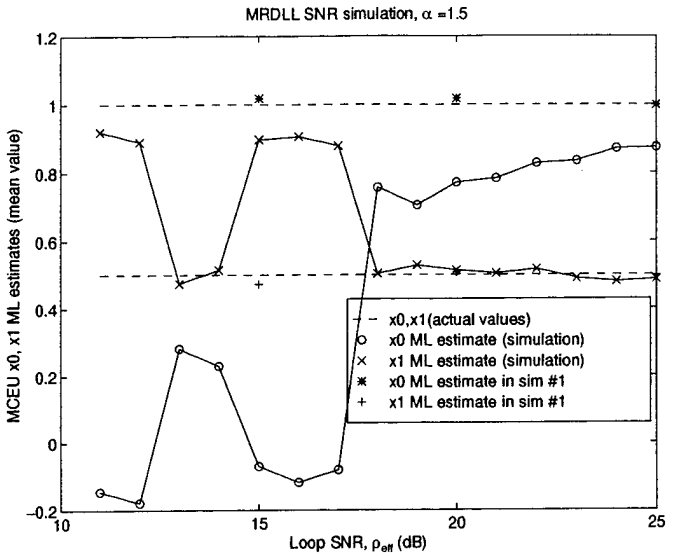

Figure 77. Simulation results showing MCEU ML estimate, $\widehat{\mathbf{x}}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.5$.

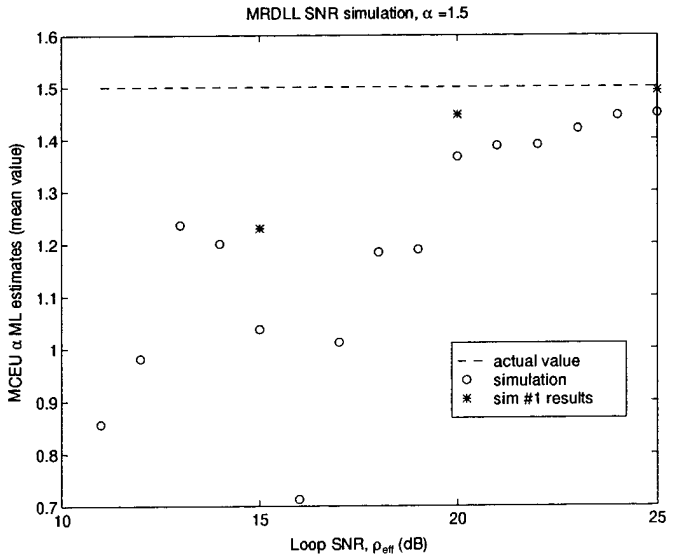

Figure 78. Simulation results showing MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.5$.

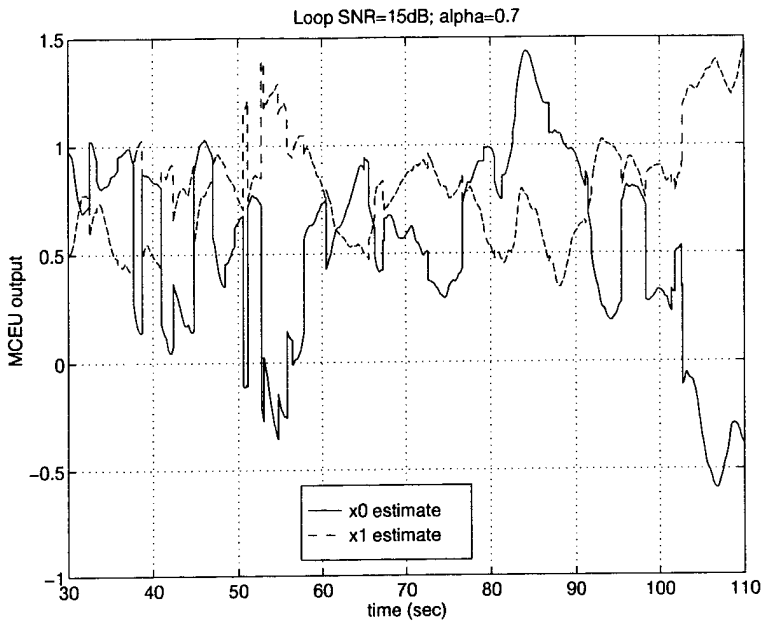

Figure 79. Simulation results showing the MCEU ML estimate output, $\widehat{\mathbf{x}}$, for $P/N_0B = 15$ dB when $\alpha = 0.7$.

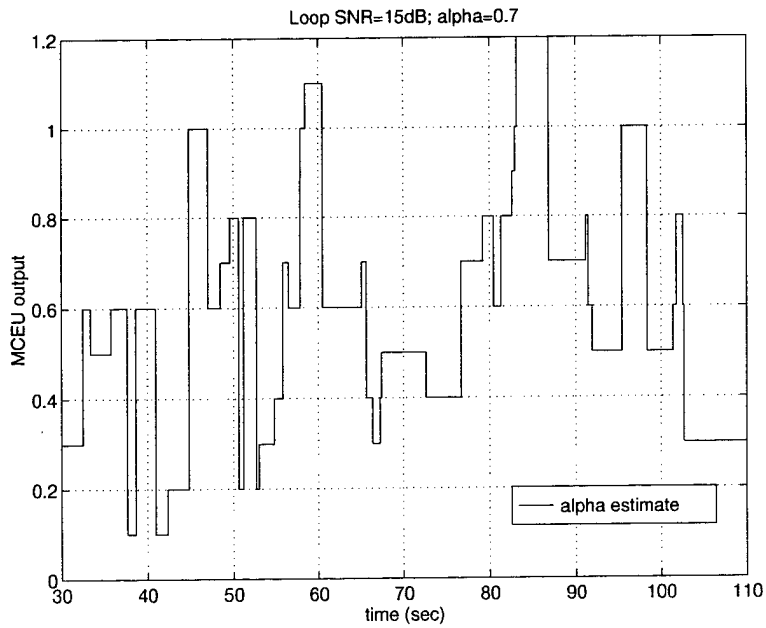

Figure 80. Simulation results showing the MCEU ML estimate output, $\widehat{\alpha}$, for $P/N_0B = 15$ dB when $\alpha = 0.7$.

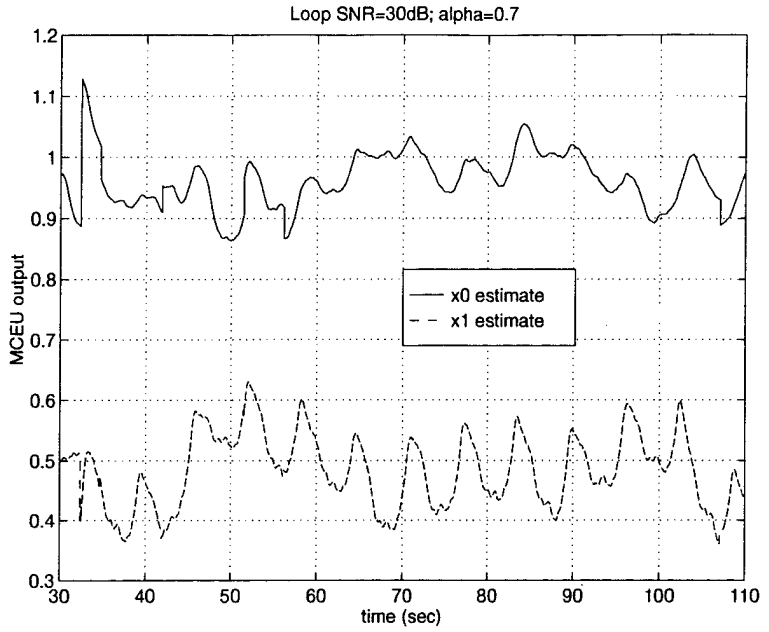

Figure 81. Simulation results showing the MCEU ML estimate output, $\widehat{\mathbf{x}}$, for $P/N_0B = 30$ dB when $\alpha = 0.7$.

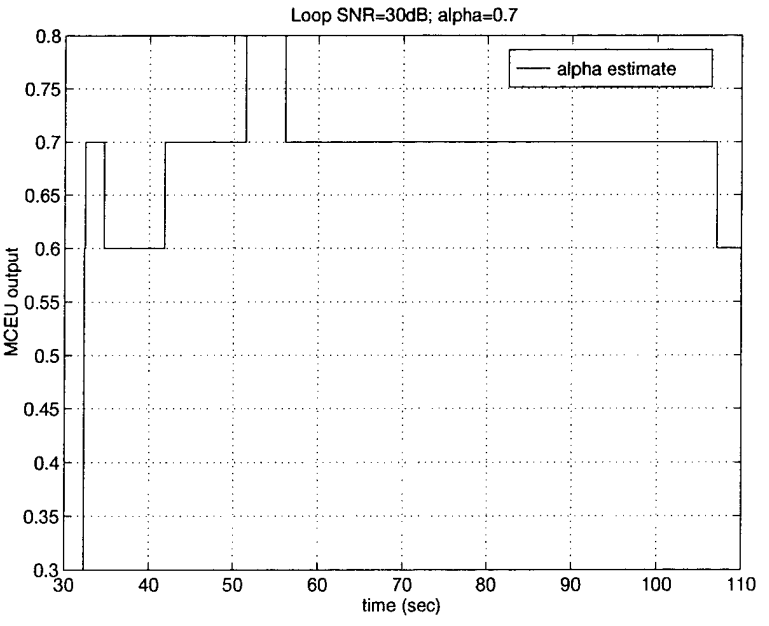

Figure 82. Simulation results showing the MCEU ML estimate output, $\widehat{\alpha}$, for $P/N_0B = 30$ dB when $\alpha = 0.7$.

V. Conclusions and Recommendations

5.1 Overview

This thesis proposed a new code tracking loop, the modified RAKE delay-lock loop (MRDLL), for GPS multipath mitigation. Prior to introducing the proposed loop, the GPS multipath problem was examined by observing the effects of multipath interference on the the standard noncoherent delay-lock loop (NCDLL), which is typically used by GPS receivers. Once the effects of the multipath problem were understood, we proposed a solution in the form of the MRDLL; this was done by performing a theoretical analysis of the MRDLL's three main components: the multiple-correlator tracking loop (MCTL), the adaptive loop controller (ALC), and the multiple-correlator estimation unit (MCEU). Finally, after theoretical predictions indicated the MRDLL's ability to mitigate multipath effects on code tracking, computer simulations were performed to verify theoretical predictions and investigate loop behavior in the absence of simplifying assumptions. This chapter provides a summary of the research results, and includes recommendations for further research.

5.2 NCDLL results

We began the NCDLL analysis by examining the equations that govern the loop's behavior; in doing so, we demonstrated the ability of the NCDLL to track the code phase of the incoming GPS spreading code signal in a direct-path only environment. The analysis showed that under direct-path conditions, the NCDLL tended to operate at the point on its discriminator output curve, or 'S-curve', where the steady-state code phase error was zero. The NCDLL linear model was also developed; this model became a valuable tool in the subsequent MRDLL analysis and simulation.

Next, the effect of multipath interference on GPS code tracking via its influence on the NCDLL steady-state tracking point was investigated; the significance of code tracking error was emphasized by pointing out that a seemingly small error in code phase estimation can translate into substantial

GPS ranging error. The NCDLL multipath analysis, performed by neglecting noise effects and focusing on the effects of a single reflected signal, showed that the reflected signal introduced a bias in the NCDLL's steady-state tracking point; this was caused by the introduction of additional cross-terms and delay shifts, which formed a composite S-curve no longer centered about the zero-phase tracking point. In addition, this composite S-curve was demonstrated to be a function of the received multipath signal parameters (i.e., amplitudes of and relative delay between direct-path and reflection); this implied that, for a GPS multipath mitigation scheme to be effective, accurate gain and phase estimates would be required to remove the contribution of the reflected signal. The analysis also revealed that multipath errors were greatest when the reflected signal was in carrier phase with the direct-path signal.

5.3 MRDLL results

5.3.1 Theoretical analysis. After demonstrating the NCDLL code phase tracking degradation caused by multipath interference, we proposed a solution to the problem in the form of the MRDLL; the subsequent analysis showed that the MRDLL had the capability to remove the contribution of a reflected signal and track the code phase with zero steady-state error in the presence of multipath. The MRDLL was a modified version of a code-division multiple access (CDMA) tracking loop (the RAKE delay-lock loop) introduced in (7); the main difference was that we designed the MRDLL to mitigate the effects of small relative multipath delays (less than 1.5 code chips), whereas the RDLL was designed for delays corresponding to integer multiples of the code chip period.

The MRDLL's tracking loop component, the MCTL, is a design similar to that of the NCDLL; both loops rely on early-late code correlation to track the received signal's code phase and provide the receiver with an 'on-time' replica code. However, the MCTL does not perform a squaring operation on the signal since phase information is needed to perform signal estimation; this signal

estimation was shown to be crucial to the MRDLL's ability to determine the contribution of the reflected signal and remove its effects. Another key difference between the NCDLL and MRDLL is that the MRDLL consists of multiple tracking arms: one corresponding to the direct-path signal component and the others arms to candidate delays of a reflected signal.

Assuming perfect signal parameter estimates from the MCEU, analysis showed that the MCTL's linear tracking point was formed (as in the NCDLL multipath case) from a summation of delay-shifted and attenuated operating curves; however, unlike the NCDLL, the operating curves were combined (using signal estimates from the MCEU), to yield a composite S-curve possessing a steady-state tracking point of zero. Therefore, the MCTL (given perfect signal estimation), is able to successfully track the GPS code phase in a multipath environment.

Analysis of the MCTL's S-curve and the resulting linear model demonstrated a very interesting property of the MCTL: the MRDLL's linear operating region varies with the multipath signal signal parameters. This realization led to the development of the ALC, which was introduced to allow the loop designer to fix the linearized dynamic parameter specifications of the loop such as natural frequency, damping ratio, and noise bandwidth. The ALC was designed as a loop filter preceded by an adaptive gain which adjusted its value according to estimates received from the MCEU, once again demonstrating the importance of signal estimation to optimal tracking loop performance. The ALC design was not present in the RDLL and was unique to the MRDLL design.

The MCEU analysis was performed by assuming perfect code phase estimates at the MCEU's 16 correlator bank inputs; these correlators, in effect, sample the code correlation function at 16 different values of relative multipath delay. The MCEU estimator uses these correlator measurements and knowledge of the noise output correlation matrix to generate 16 candidate maximum-likelihood (ML) signal estimates, each determined assuming a particular value of relative multipath delay; then, the MCEU chooses the most-likely of these 16 estimates via examination of the resulting

prediction error vectors. As already discussed, these estimates are used by the MCTL to remove the tracking error introduced by the reflected signal.

5.3.2 Computer simulations. Computer simulations were performed to verify the results predicted by theoretical analysis, as well as to examine MRDLL behavior in the absence of simplifying assumptions such as perfect signal estimation and linear operation. The initial intent of the simulations was to simulate (as close as possible) real-world GPS conditions; however, the Signal Processing Workstation (SPW) software simulation package proved to be prohibitively slow in processing closed loop simulations of the MRDLL. Therefore, the focus of the simulations became to demonstrate general MRDLL behavior and tendencies in a variety of multipath and direct-path environments; the results provided valuable insight into the application of the MRDLL to the GPS multipath problem. It should be mentioned that the MATLAB software performed extremely well (in terms of simulation time) in the MCEU simulation assuming perfect MCTL code phase estimates; this was because MATLAB is optimized for matrix operations that are used by the MCEU estimator (SPW's matrix algorithms are not even remotely as efficient).

The first simulation (using MATLAB) investigated MCEU performance in AWGN assuming perfect code phase estimation. The results demonstrated the ability of the MCEU to accurately estimate the received signal parameters in both direct-path and multipath environments. It was seen that, when multipath was present, the MCEU estimate corresponding to the received relative multipath delay was the only unbiased estimate out of the 16 possible; this meant the MCEU tended to send this estimate as its ML estimate to the MCTL. For the direct-path only environment, however, it was observed that the MCEU tended to choose the estimate that did *not* correspond to zero relative delay, but still provided accurate signal amplitude estimates; this result was due to the fact that, when no reflection was present, *all* of the 16 possible estimates were unbiased estimators. The simulation also revealed that the estimator performance was worse when the reflected signal

exhibited a small relative multipath delay; this was because the estimator variance depended on the delay value and was greatest when the relative delay was small.

The second and third simulations used the SPW environment to compare MRDLL vs NCDLL steady-state code phase tracking performance in the presence of multipath interference. The results showed that, as predicted by the theoretical analysis, the MRDLL exhibited a significant improvement in steady-state tracking error over the NCDLL *when multipath was present*; the presence of a non-zero steady-state error, however, indicated the effects of imperfect estimation on the MRDLL. In a direct-path only environment, it was the NCDLL that demonstrated the best tracking performance as it had zero steady-state error compared to the MRDLL's finite tracking error; this was because the NCDLL is designed *assuming* no multipath, while the MRDLL must *estimate* the received signal parameters to determine the strength of the reflected signal or if the reflected signal is present at all.

The final simulation investigated MRDLL performance in AWGN. Results indicated that tracking performance predicted by linear theory was applicable only when the received SNR exceeded a given threshold; this result is common to all tracking loop analysis. What made the MRDLL SNR results unique was that fact that, because the linear tracking region depends on the relative multipath delay, so does the SNR threshold; as one might expect, the greater the linear operating region, the lower the required SNR threshold. Results demonstrated that this type of simulation would be very useful in determining the MRDLL's operating specifications for different signal parameters (should actual GPS parameters be used in the simulation).

5.4 Summary and Recommendations

This thesis has shown that The MRDLL has tremendous potential for GPS application where multipath mitigation is a concern. This was demonstrated by both simulation and analysis in which the MRDLL exhibited significant improvement over the NCDLL in terms of steady-state code-phase

tracking error with a single reflection present. The following is a list of suggested follow-on work for MRDLL research (and research on GPS multipath mitigation in general):

1. Develop a MATLAB (or another alternative) simulation model of the MRDLL. This would make simulations more efficient than they were in SPW, and perhaps allow the simulations to run at GPS parameters.
2. Design a detection/estimation scheme that detects the presence of multipath, and allows the GPS receiver to switch between the NCDLL and the MRDLL depending on the received signal environment. This could optimize code phase tracking performance since the NCDLL is still the optimum choice for GPS code phase tracking when multipath is not present.
3. Change the MCEU estimation scheme to allow $\widehat{\alpha}$ to be any element of the interval (0, 1.5]. To be useful, the MCTL would also have to be modified to allow mitigation of any of a continuum of possible delays.
4. Generalize the mitigation task for a multiple-reflection scenario.
5. Develop a carrier loop simulation model to run in tandem with the MRDLL model.
6. Investigate the effects of code and frequency Doppler shift on the MRDLL in the presence of multipath; compare both ALC and non-ALC loop designs.
7. Investigate MRDLL performance when random data modulation is included on the received GPS signal. The data can then be demodulated and a bit error rate analysis conducted.
8. Further investigate the minimum mean-square error estimation scheme suggested by Weill (14).
9. Consider narrower correlator spacings ($\Delta < 1$) in the MCTL (MRDLL).

Appendix A. Computer Simulation Models

A.1 Received GPS signal - complex envelope representation

Each of the simulation models presented in this appendix are based on the *complex low-pass envelope* of the received GPS multipath signal. The complex low-pass approach is a useful simulation tool because it decreases the required sampling frequency and simulation time; the approach is valid as long as the carrier frequency is much greater than the bandwidth of the bandpass signal (as is the case for GPS) (3). To determine the complex envelope representation, the GPS multipath signal (neglecting data modulation and noise) is written as

$$\begin{aligned} r(t) &= \sqrt{2P}\left\{a_0 c\left(t-\tau_0\right)\cos(\omega_c t+\theta_0)+a_1 c\left(t-\tau_0-\alpha T_c\right)\cos(\omega_c t+\theta_1)\right\} \\ &= \mathrm{Re}\left\{\sqrt{2P}\left[a_0 c(t-\tau_0)e^{j(\omega_c t+\theta_0)}+a_1 c(t-\tau_0-\alpha T_c)e^{j(\omega_c t+\theta_1)}\right]\right\} \\ &= \mathrm{Re}\left\{\widetilde{r}(t)e^{j\omega_c t}\right\} \end{aligned} \tag{101}$$

where $\widetilde{r}(t)$ is the complex envelope of the signal given by

$$\widetilde{r}(t)=\sqrt{2P}\left[a_0 c(t-\tau_0)e^{j\theta_0}+a_1 c(t-\tau_0-\alpha T_c)e^{j\theta_1}\right]\ . \tag{102}$$

Therefore, as shown in Figure 83, the signal can be modeled as a DS/SS code passing through a tapped delay line with complex tap weights (7).

For the MRDLL, the received signal must be converted to baseband. Recall from Chapter III that this conversion is accomplished by mixing the received signal (neglecting data and noise) with a local carrier to give

$$\begin{aligned} r_{bb}(t) &= x_0 c\left(t-\tau_0\right)+x_1 c\left(t-\tau_0-\alpha T_c\right) \\ &= \sqrt{2P}\left\{a_0\cos(\theta_0-\theta_3)c\left(t-\tau_0\right)+a_1\cos(\theta_1-\theta_3)c\left(t-\tau_0-\alpha T_c\right)\right\} \end{aligned} \tag{103}$$

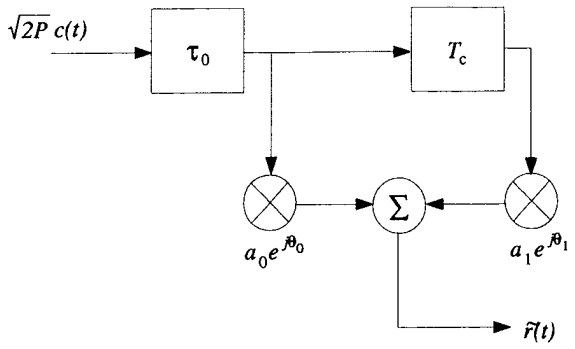

Figure 83. Multipath channel tapped delay line model.

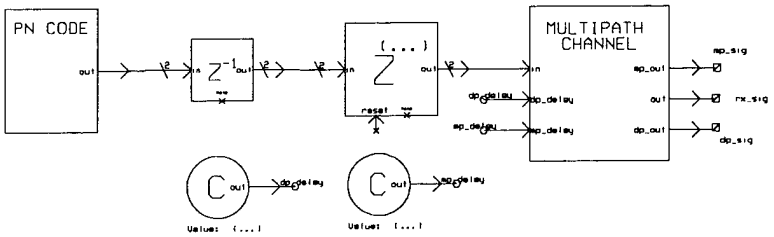

Figure 84. SPW GPS Multipath Signal block detail.

where θ_3 is an arbitrary phase error. Therefore, the baseband conversion is modeled by multiplying $\tilde{r}(t)$ in Equation 102 by $e^{-j\theta_3}$ and taking the real part to give Equation 103.

A.2 GPS Multipath Signal model

The GPS MULTIPATH SIGNAL block detail is shown in Figure 84. This block generates the received GPS signal according to Equation 102 and the channel model in Figure 83; the direct-path and multipath components of the signal are monitored separately via the 'dp_out' and 'mp_out' ports, respectively. The GPS MULTIPATH SIGNAL block consists of two main components: the PN CODE GENERATOR and the MULTIPATH CHANNEL. Two COMPLEX DELAY blocks (marked 'Z' in the diagram) are also included; the purpose of these blocks will be explained in Section A.3.1.

The PN CODE GENERATOR block detail, shown in Figure 85, generates a DS/SS code at a specified chip rate ($1/T_c$) and code period, N. The desired chip rate is achieved by setting

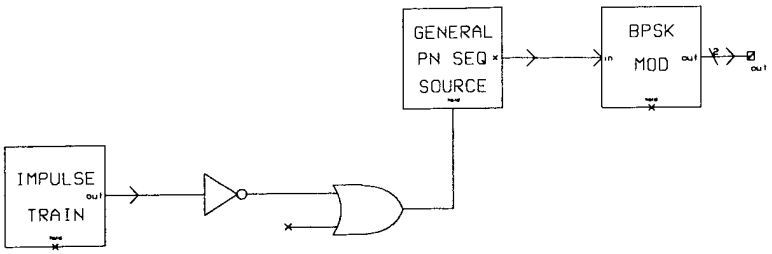

Figure 85. SPW PN Code Generator block detail.

the IMPULSE TRAIN frequency equal to $1/T_c$; the impulse train acts as a control input to the GENERAL PN SEQ SOURCE which generates a maximal-length PN code of 1s and 0s. The PN code period is determined by setting the block's *shift register order* according to

$$N = 2^m - 1 \tag{104}$$

where m is the shift register order. The BPSK MOD block then converts the PN code to a binary waveform of 1s and -1s to give the DS/SS code, $c(t)$.

The MULTIPATH CHANNEL block, shown in Figure 86, implements the tapped delay line model shown in Figure 83. The COMPLEX VARIABLE DELAY blocks allow the designer to vary the values of τ_0 and α, while the G/P MULT blocks, representing the complex tap weights, allow the designer to vary the attenuation coefficients, a_i, and the carrier phases, θ_i.

A.3 NCDLL model

The NCDLL block detail, shown in Figure 87, is based on the NCDLL block diagram and analysis presented in Chapter II. Upon receiving the multipath signal, the NCDLL block returns two outputs: the *on-time code* and the loop filter signal. The on-time code ('on_time' in Figure 87) is a replica of the incoming direct-path DS/SS code and has a code phase equal to the NCDLL's estimate of the direct-path propagation delay. The loop filter signal ('loop_flt' in Figure 87) is the

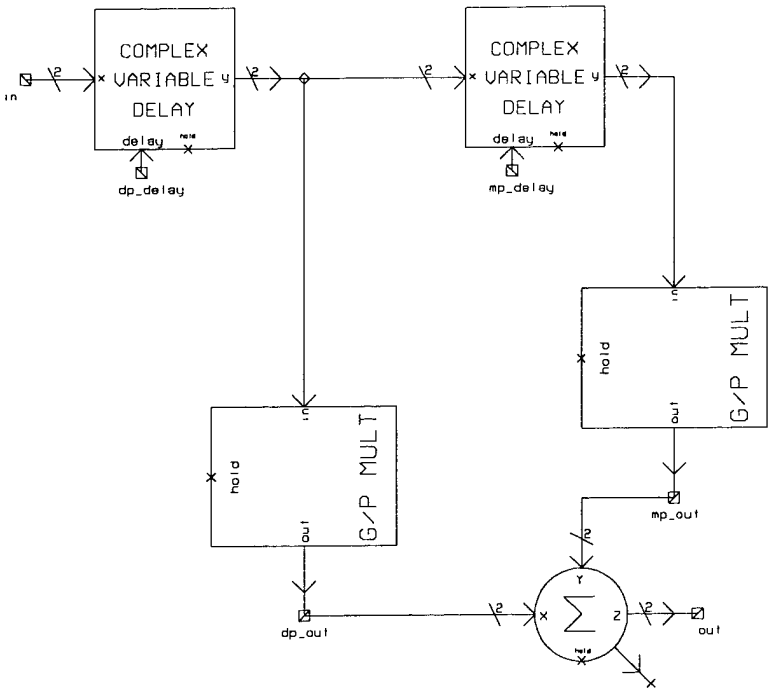

Figure 86. SPW Multipath Channel block detail.

lowpass output of the NCDLL loop filter which drives the VCC; this signal is monitored to ensure the loop remains stable and to determine when the loop reaches steady-state.

A.3.1 Early/late correlator model. Now, we will examine more closely the individual components of the NCDLL block, starting with the early/late correlators. The correlators are modeled as multipliers which mix the incoming GPS signal with shifted early and late versions of the on-time replica code generated by the CODE GENERATOR block. The delay block preceding the 'late' correlator ensures that the signal into the multiplier lags the signal into the 'early' multiplier by one chip period, T_c; this corresponds to an early/late offset of $\Delta = 1$.

The early/late correlator implementation just described dictates that there be one half-chip ($T_c/2$) delay between the CODE GENERATOR block and the 'on-time' output; this delay is accomplished by placing a delay block just before the NCDLL block's 'on_time' output. The additional unit delay block immediately after the CODE GENERATOR is necessary because SPW

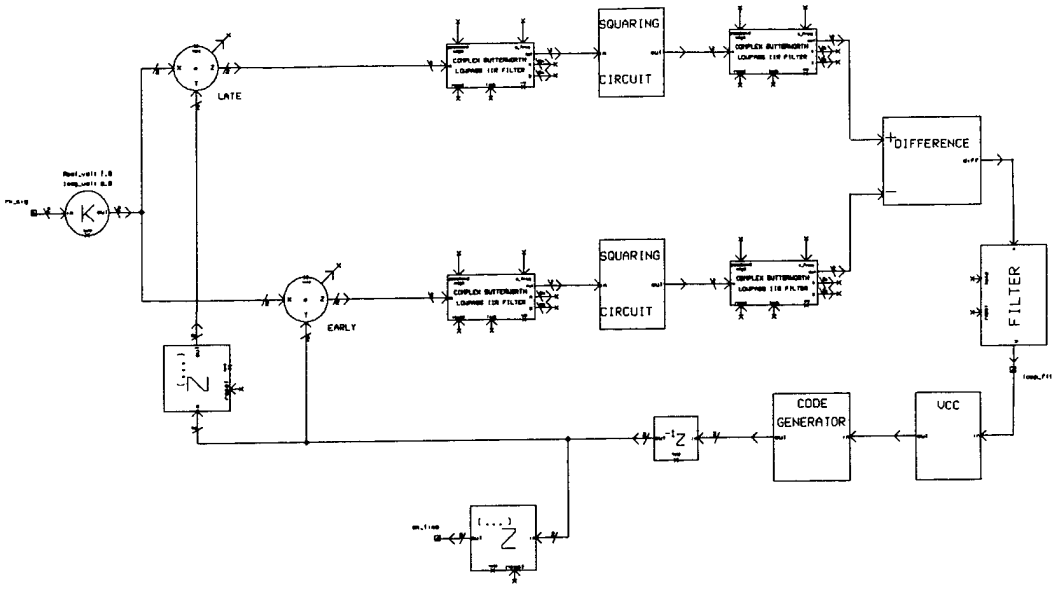

Figure 87. SPW NCDLL block detail.

requires at least one sample of delay in feedback loops. The presence of these two delay blocks is accounted for by the two delay blocks in the GPS MULTIPATH SIGNAL block (Figure 84).

A.3.2 Bandpass and lowpass filter models. The NCDLL's BPFs are modeled as LPFs since we are using complex lowpass signals. Therefore, these filters (along with the LPFs following the squaring operation) are modeled using SPW's COMPLEX BUTTERWORTH LOWPASS IIR FILTER block; this block allows the designer to specify the filter passband edge (in Hz) and the amount of attenuation (in dB) at the passband edge.

A.3.3 Squaring circuit model. The NCDLL's squaring operation is modeled by the SQUARING CIRCUIT block. Because we are using complex lowpass signals, this block cannot simply return the square of the input signal. Referring back to the NCDLL multipath analysis in Chapter II, the NCDLL early/late correlator output, $y(t)$, into the squaring circuit is given by

$$y(t) = \sqrt{2P}\left\{a_0 R_c\left(\delta \pm \frac{1}{2}\right)\cos\left(\omega_c t + \theta_0\right) + a_1 R_c\left(\delta + \alpha \pm \frac{1}{2}\right)\cos\left(\omega_c t + \theta_1\right)\right\}, \qquad (105)$$

and the lowpass component of the squaring circuit output (into the adder circuit) is

$$y^2(t)\big|_{LP} = P\left\{a_0^2 R_c^2\left(\delta \pm \frac{1}{2}\right) + a_1^2 R_c^2\left(\delta + \alpha \pm \frac{1}{2}\right) + 2a_0 a_1 \cos(\theta_0 - \theta_1) R_c\left(\delta \pm \frac{1}{2}\right) R_c\left(\delta + \alpha \pm \frac{1}{2}\right)\right\} . \quad (106)$$

The complex lowpass envelope of $y(t)$ into the squaring circuit is

$$\widetilde{y}(t) = \sqrt{2P}\left\{a_0 R_c\left(\delta \pm \frac{1}{2}\right) e^{j\theta_0} + a_1 R_c\left(\delta + \alpha \pm \frac{1}{2}\right) e^{j\theta_1}\right\} . \quad (107)$$

Multiplying by the complex conjugate, $\widetilde{y}^*(t)$ and dividing by two, we see that

$$\frac{1}{2}\widetilde{y}(t)\widetilde{y}^*(t) = y^2(t)\big|_{LP} . \quad (108)$$

Therefore, in order to accurately model the NCDLL squaring operation, the SQUARING CIRCUIT block multiplies the incoming complex lowpass signal by its complex conjugate and divides the result by two.

A.3.4 Loop filter model. The loop filter is modeled by SPW's FILTER block which defines a filter according to its difference equation coefficients. As seen in (3), the digital realization of the active lead-lag filter described in Equation 24 corresponds to the difference equation

$$y[n] = A_0 x[n] + A_1 x[n-1] - B_1 y[n-1] \quad (109)$$

where $x[n]$ and $y[n]$ are the discrete-time representations of the filter input and output, respectively. The difference equation coefficients are given by (3)

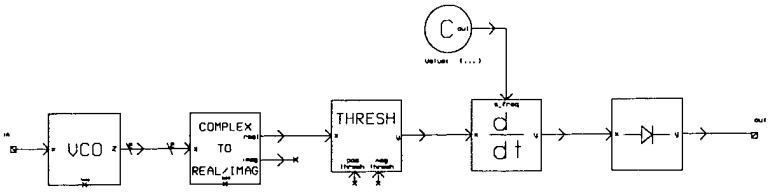

Figure 88. SPW VCC block detail.

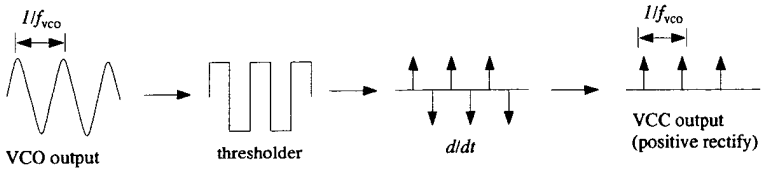

Figure 89. Illustration of VCC operation in SPW.

$$\begin{aligned} A_0 &= \frac{T_s + 2\tau_2}{2\tau_1}; \\ A_1 &= \frac{T_s - 2\tau_2}{2\tau_1}; \\ B_1 &= -1 \end{aligned} \tag{110}$$

where T_s is the sampling period in seconds; the FILTER block allows the designer to specify these coefficients according for the desired values of τ_1 and τ_2.

A.3.5 VCC model. The VCC block adjusts the frequency of the on-time code so that its phase matches the NCDLL's estimate of direct-path phase. The VCC block detail is shown in Figure 88 and its operation is illustrated in Figure 89; this implementation is based on a design suggested in (3). The VCC output serves the same purpose in the CODE GENERATOR block as the IMPULSE TRAIN does in the PN CODE block (see Figure 85); the difference is that the VCC's impulse train output changes in frequency according to the VCO output as shown in Figure 89.

The VCC block allows the designer to specify the VCO *quiescent frequency* in Hz and the VCO constant in Hz/volt. The quiescent frequency of the VCO (i.e., the output frequency with a zero-volt input) is set equal to the chip rate, $1/T_c$ Hz. Therefore (with no noise and/or multipath), if the on-time replica code is in phase with the received signal code, the early/late correlator outputs will be equal. This will result in a loop filter output equal to zero and will cause the replica code to remain at $1/T_c$ Hz. However, if the received code lags the on-time code, the received code will more closely correlate with the late replica; therefore, the loop filter output will be positive. Similarly, if the received code leads the on-time replica, the loop filter output will be negative. Therefore, the VCO constant is specified in such a way that the VCC frequency increases with a negative input voltage and decreases with a positive input voltage. The actual value of the VCO constant is chosen according to the desired dynamic performance of the loop.

A.4 MCTL model

The MCTL block detail, shown in Figure 90, is based on the MRDLL block diagram and MCTL analysis presented in Chapter III. Like the NCDLL block, the MCTL block generates an on-time replica of the direct-path code and a loop filter signal for monitoring purposes. Unlike the NCDLL block, however, the MCTL block receives signal parameter estimates from the MCEU block in addition to the received baseband signal.

The MCTL model has two main arms, each consisting of an EARLY-LATE CORRELATOR block followed by a gain/phase multiplier. The top arm represents the MCTL's direct-path arm, and the lower arm represents the enabled multipath arm corresponding to the MCEU's estimate of the multipath delay coefficient, $\widehat{\alpha}$. Both arms' EARLY-LATE CORRELATOR blocks (see Figure 91) operate in the same manner as the NCDLL's early/late correlator arms (except for the squaring operation). In the multipath arm, a delay block delays the MCTL's on-time replica by $\widehat{\alpha}T_c$ seconds prior to early/late correlation. Therefore, each arm's early/late correlator output is the correlation

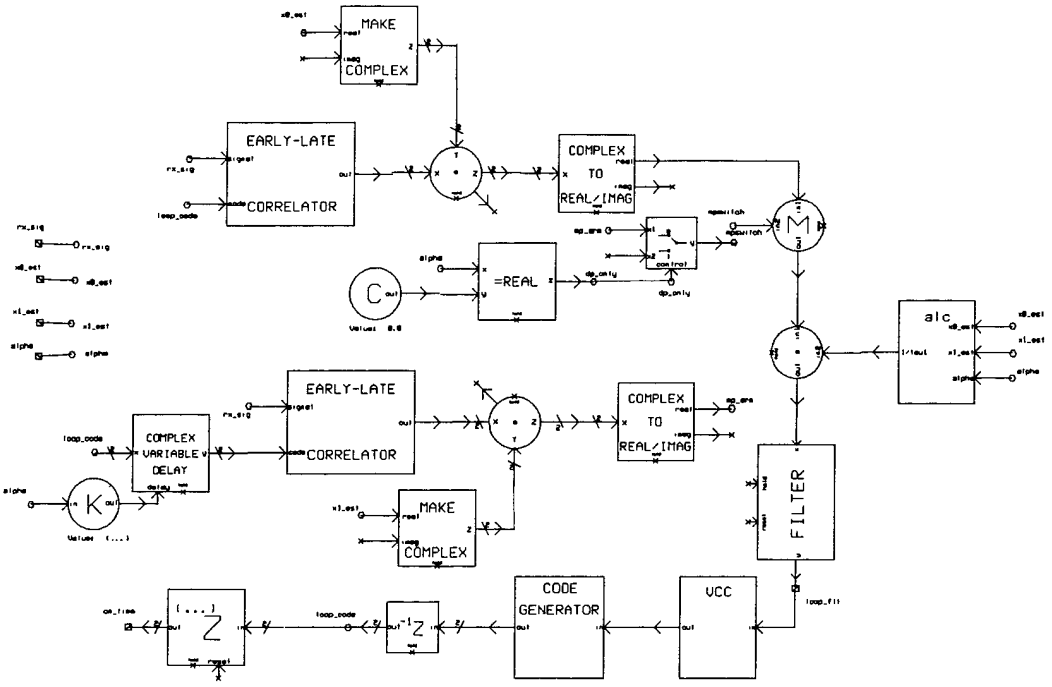

Figure 90. SPW MCTL block detail.

between the received signal and a shifted replica code given by

$$c_{\Delta}(t)|_{dp} = c(t - \widehat{\tau}_0 - T_c/2) - c(t - \widehat{\tau}_0 + T_c/2) \tag{111}$$

for the direct-path arm and

$$c_{\Delta}(t)|_{mp} = c(t - \widehat{\tau}_0 - \widehat{\alpha}T_c - T_c/2) - c(t - \widehat{\tau}_0 - \widehat{\alpha}T_c + T_c/2) \tag{112}$$

for the enabled multipath arm. The early/late correlator outputs then enter the gain/phase multipliers, where they are mixed with the appropriate MCEU signal estimates. If the MCEU output is $\widehat{\alpha} = 0$ (i.e., no multipath present), a switch disables the lower arm.

The ALC block operates in conjunction with the SPW FILTER block to comprise the overall ALC model (refer to Figure 19 for the ALC block diagram). The ALC block receives signal estimates

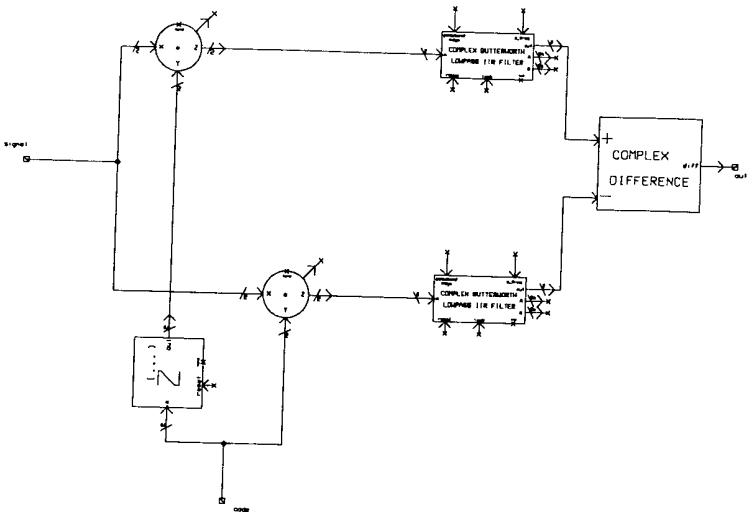

Figure 91. SPW Early-Late Correlator block detail.

from the MCEU block and adjusts the loop filter gain, K_A, as described in Chapter III; the adaptive gain threshold, $K_{A\max}$, is also set within this block. In the FILTER block, the difference equation coefficients in Equation 110 are chosen so that the loop filter transfer function is as described in Equation 61

A.5 MCEU model (MATLAB)

A.5.1 Description. As seen in Chapter III, the MCEU relies on matrix operations to compute the multipath signal estimates sent to the MCTL. The MATLAB software package is optimized for matrix operations; therefore, a MATLAB model of the MCEU was developed to perform preliminary simulation and analysis prior to SPW implementation.

The MCEU MATLAB simulation model is based on the MCEU block diagram in Figure 25 and the analysis presented in Chapter III. The MATLAB code for this simulation model is contained in the *mceu.m* m-file in Section A.5.2. Running *mceu.m* represents one sample of the MCEU's M signal parameter estimates ($\widehat{\mathbf{x}}_k$, and $\widehat{\alpha}_k$) and their corresponding error estimates E_k. The model assumes a received baseband signal as described by Equation 103, zero tracking error

(i.e., $\tau_0 = \widehat{\tau}_0$), and large code period, N, so that the code autocorrelation approximation in Equation 5 holds. Also, we choose the number of correlator arms as $M = 16$ with replica codes spaced $0.1T_c$ seconds apart (i.e., $\beta_k = 0.1k$).

Care must be taken when modeling the correlator noise outputs in MATLAB. Recall that the correlator output noise samples are not uncorrelated; instead, they are represented as a 16×1 random vector, $\mathbf{v}$, distributed as $N : [\mathbf{0}, \sigma_{\mathbf{v}}^2 \mathbf{C}_{\mathbf{v}}]$, where σ_v^2 is the variance of $\mathbf{v}$ and $\mathbf{C}_{\mathbf{v}}$ is the noise covariance matrix defined in Equation 75. MATLAB can generate a similar random vector, $\mathbf{w}$, but with distribution $N : [\mathbf{0}, \mathbf{I}]$, where $\mathbf{I}$ is the identity matrix. Therefore, it is necessary to transform the MATLAB-generated white noise vector, $\mathbf{w}$, to the colored noise vector, $\mathbf{v}$. The *whit2col.m* m-file (Section A.5.3) accomplishes this transformation according to (6)

$$\mathbf{v} = \sigma_{\mathbf{v}} \mathbf{L}^{\mathbf{T}} \mathbf{w} \tag{113}$$

where σ_v is the noise standard deviation and $\mathbf{L}$ is the Cholesky factorization of $\mathbf{C}_{\mathbf{v}}$.

A.5.2 The mceu M-file.

```
%MCEU
%Multiple-Correlator Estimation Unit (MCEU) estimator output
%
%[E, X, ALPHA_ML, X_ML] = MCEU(X0, X1, ALPHA, CNo, B) returns the MCEU estimator
output
%for received signal parameters, X0, X1, and multipath delay coefficient, alpha.
%The remaining two input arguments are the received direct-path carrier-to-noise
density
%ratio (CNo in dB-Hz) calculated for received carrier power P=1/2, and the MCEU
%integrator one-sided lowpass bandwidth (B in Hz) (setting B=0 gives no-noise
response).
%
%The output arguments are defined as follows:
%
%E = a vector whos elements are the 16 candidate error measurements, E_k; (the
MCEU
%chooses as its ML estimate the candidate signal parameter estimate
corresponding to
%the minimum E_k).
%
%X = a 2 X 16 matrix whos columns correspond to the 16 candidate signal
parameter
%and rows correspond to x0 (1st row) and x1 (2nd row)
%
```

```
%ALPHA_ML = The MCEU's ML alpha estimate
%
%X_ML = The MCEU's ML signal parameter estimate vector
%
%This function uses the WHIT2COL m-file to generate the correlator output noise
%vector, and the CODECORR m-file to compute the code autocorrelation function.
%
%Written by Mark C. Laxton, 1 May 96. Last updated: 15 Nov 96.

function [E,X,alpha_ml,x_ml]=mceu(x0,x1,alpha,CNo,B)

beta=0:0.1:1.5; %Set up delay-spacing coefficients, beta
alpha_k=beta; %Set alpha estimates, alpha_k, equal to beta

v=whit2col(beta,CNo,B); %Generate correlator noise vector, v
v=v';

%Define correlation matrix, C
w=codecorr(beta-beta(1));
C=toeplitz(w);

%Form correlator measurement vector, R
R=codecorr(beta)*x0+codecorr(alpha-beta)*x1+v;

%Form candidate estimates for x0 and x1 according to MCEU estimation algorithm
for k=0:length(alpha_k)-1
```

```
if k==0
H=(codecorr(beta))';
x0_0=inv(H'*inv(C)*H)*H'*inv(C)*R';
x1_0=0;
X(:,k+1)=[x0_0 x1_0]';
  else
H(:,1)=(codecorr(beta))';
H(:,2)=(codecorr(alpha_k(k+1)-beta))';
  X(:,k+1)=inv(H'*inv(C)*H)*H'*inv(C)*R';
end
end

%Choose ML estimate by computing and choosing estimates corresponding to minimum
E.
for k=0:length(beta)-1
H(:,1)=(codecorr(beta))';
H(:,2)=(codecorr(alpha_k(k+1)-beta))';
  R_est(:,k+1)=H*X(:,k+1);
end
R_chkmx=R'*ones(1,length(beta));
e=R_est-R_chkmx;
for k=0:length(beta)-1
E(k+1)=e(:,k+1)'*inv(C)*e(:,k+1);
end

m=find(E==min(E));
```

```
alpha_ml=alpha_k(m);

x_ml=X(:,m);
```

A.5.3 The whit2col M-file.

```
%WHIT2COL
%Transform white noise to colored noise.
%
%(This function was written to be used in conjunction with
%the MCEU m-file).
%
%Z = WHIT2COL(BETA,CNo,B) transforms a white noise vector distributed as
%N[0,var*I] to a noise vector, Z, distributed as N[0,var*R] where R is the
%correlator matrix corresponding to the noise outputs of the MCEU correlators
%with delay spacing, BETA.  The integrator bandwidth of the BPFs is B (in Hz).
%and the received carrier-to-noise density ration is CNo dB-Hz (assumes signal
%power, P=1/2).
%
%Written by Mark C. Laxton, 1 May 1996. Last updated: 15 Nov 96


function z=whit2col(beta,CNo,B)


%Create white noise vector, w
k=length(beta);
w=randn(k,1);


%define correlation matrix, R
v=codecorr(beta-beta(1));
R=toeplitz(v);
```

```
%Use Cholesky factorization to determine transformation matrix A
A=chol(R);


%transform w to output noise vector, z
z=A'*w;


%multiply by sqrt of variance, sig2
No=1/(2*10^(CNo/10));
sig2=2*B*No;
z=sqrt(sig2)*z;
```

A.5.4 The codecorr M-file.

```
%CODECORR
%Compute DS/SS code autocorrelation function for large code period
%
%Rc = CODECORR(x) returns the autocorrelation function, Rc, of a DS/SS
%code having a code period, N >> 1. The value of Rc is approximated by
%Rc(x) = 1-|x| for |x|<=1 and Rc(x)=0 elsewhere.
%
%Written by Mark C. Laxton, 1 May 1996. Last updated: 15 Nov 96

function Rc=codecorr(x)


for k=1:length(x)
if abs(x(k))<=1
Rc(k)=1-abs(x(k));
else
Rc(k)=0;
end
end
```

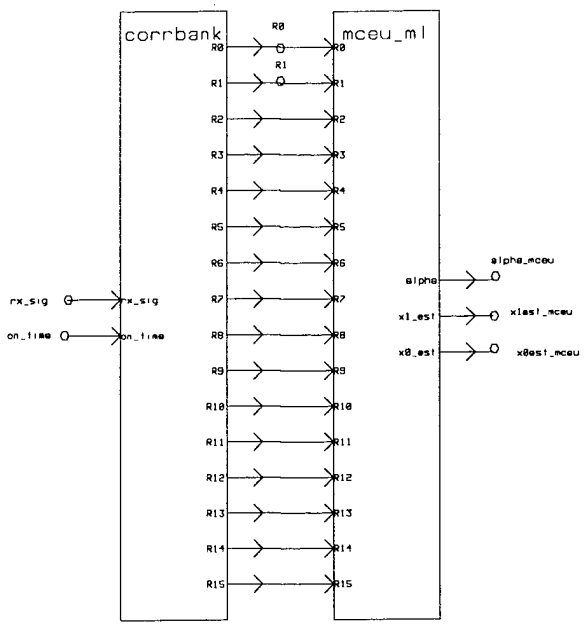

Figure 92. SPW MCEU model.

A.6 MCEU model (SPW)

The SPW MCEU model is shown in Figure 92 and consists of two main components: the multiple correlator bank (CORRBANK) block and the maximum-likelihood estimator (MCEU_ML) block. The CORRBANK block generates $M = 16$ correlator measurements that are fed into the MCEU_ML block; each correlator is modeled using the variable-delay correlator (VDC) block shown in Figure 93. Each VDC correlates the received baseband signal with a delayed version of the on-time replica code from the MCTL block; the kth VDC has a code replica delay spacing coefficient, $\beta_k = 0.1k$. The MCEU_ML block receives the CORRBANK correlator measurements, computes the $M = 16$ signal parameter estimates, and sends the ML estimate to the MCTL block as described in Chapter III.

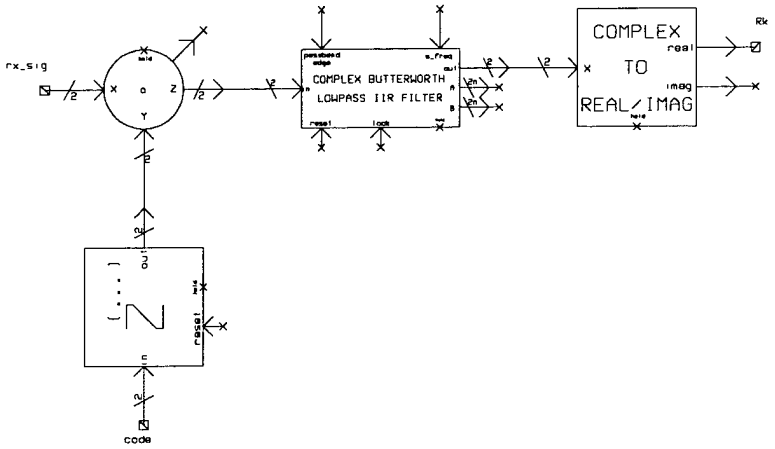

Figure 93. SPW Variable-Delay Correlator (VDC) block detail.

Appendix B. Data Processing M-files

B.1 The mrdlest M-file

```
%MRDLEST Compute and plot means and variances of MCEU ML output estimates
%
%EST = MRDLEST(START,STOP,X0FNM,X1FNM,ALPHAFNM,S_FREQ) computes the sample means
%and variances of three vectors ('x0', 'x1', and 'alpha') which contain samples
%of the corresponding MCEU ML estimator outputs; the sample frequency is defined
%by the S_FREQ input argument. The three vectors are contained in mat-files named,
%'X0FNM', 'X1FNM', and 'ALPHAFNM'. Each mat-file is assumed to have come from
%an SPW MATLAB SINK which has assigned the output data to a vector, 'Y', and
%has also returned an erronneous 'sampfreq' value (based on SPW v.3.5, 1996). This
%m-file renames each Y vector to x0, x1, or alpha as appropriate, and deletes the
%erroneous sampfreq variable. The range of data samples used to compute the sample
%means and variances are defined by the START and STOP input arguments; these values
%specify the first and last indices of the x0, x1, and alpha vectors used in the
%mean and variance computation. The output matrix, EST, is a 2 X 3 matrix whose
%columns correspond to x0, x1, and alpha (in that order), and whose 1st row is the
%sample mean and 2nd row is the sample variance.
%
%EST = MRDLEST(...,LABEL, XFIG, AFIG) plots the MCEU outputs vs time and sends the
%results to a print file in eps format. The LABEL argument is a string containing
%the title of the plot; XFIG and AFIG are strings which correspond to the print
%filenames of each plot.
%
%Written by Mark C. Laxton, 1 May 96. Last updated: 15 Nov 96.
```

```
function EST=mrdlest(start,stop,x0fnm,x1fnm,alphafnm,s_freq,label,xfig,afig)


%load matfiles, rename Y vectors, clear sampfreq value

eval(['load ' x0fnm ' -mat']);

clear sampfreq

x0=Y;

eval(['load ' x1fnm ' -mat']);

clear sampfreq

x1=Y;

eval(['load ' alphafnm ' -mat']);

clear sampfreq

alpha=Y;

clear Y


%if plot options are selected, plot vs time and send to printfile

if nargin>6

t=0:length(x0)-1;

time=t/s_freq;

figure,plot(time,x0)

hold on

plot(time,x1,'--')

hold off

xlabel('time (sec)')

ylabel('MCEU output')

grid
```

```
s=num2str(label);
title(s)
h=legend('-','x0 estimate','--','x1 estimate',0);
axes(h)
eval(['print -deps -epsi' xfig]);
figure,plot(time,alpha)
grid
xlabel('time (sec)')
ylabel('MCEU output')
title(s)
h=legend('alpha estimate');
axes(h)
eval(['print -deps -epsi' afig]);
end

%determine mean and variance of estimates
x0mn=mean(x0(start:stop));
x1mn=mean(x1(start:stop));
alphamn=mean(alpha(start:stop));
x0var=cov(x0(start:stop));
x1var=cov(x1(start:stop));
alphavar=cov(alpha(start:stop));
EST=[x0mn x1mn alphamn;x0var x1var alphavar];
```

B.2 The leadedge M-file

```
%LEADEDGE
%Normalized rms code phase tracking error estimate.
%
% RMS_ERR = LEADEDGE(DP, OT, CHP_RATE, S_FREQ) calculates the rms
% code phase timing error (in chips) between a direct-path version of a
% BPSK-modulated (1's and -1's) direct-sequence spread-spectrum (DS/SS) code
% and and'on-time' replica generated by a code tracking loop. This is done by
% comparing the leading edges between the two codes. The direct-path and
% on-time codes are input as mat-file filenames ('DP' and 'OT') for a
% specified code chip rate (CHP_RATE) in Hz and simulation sampling freq
% ('S_FREQ') in Hz.
%
% RMS_ERR = LEADEDGE(DP, OT, CHP_RATE, S_FREQ, LOCK) will display a
% 'loss of lock' message if the code phase error between any pair of leading
% edges exceeds the threshold (in chips) specified by LOCK.
%
% RMS_ERR = LEADEDGE(DP, OT, CHP_RATE, S_FREQ, LOCK, LIN) will display an
% additional 'out of linear region' message if the code phase error between
% any pair of leading edges exceeds the threshold (in chips) specified by LIN.
%
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% CAUTION: Care must be taken to ensure that the first pair of leading edges
% correspond to the same point in the DS/SS code sequence.
% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Written by Mark C. Laxton, 1 May 96. Last updated: 15 Nov 96.
```

```
function rms_err=leadedge(dp,ot,chp_rate,s_freq,lock,lin)

%Find samples where each code equals -1; put these samples into two vectors,
%'fdp' and 'fot'.
fdp=find(dp==-1);
fot=find(ot==-1);

%Determine which samples correspond to leading edges (-1 to 1 transitions)
m=1;
p=1;
for k=1:length(fdp)-1
if fdp(k)+1~=fdp(k+1) %Check if each sample number in 'fdp' is followed by the next
%highest integer. If not, that sample is a leading edge.

ledp(m)=fdp(k); %Form vector 'ledp' of sample numbers corresponding to the
%leading edges of the direct-path code.
m=m+1;
end
end

for k=1:length(fot)-1 %Repeat for on-time code, forming 'leot' vector.
if fot(k)+1~=fot(k+1)
leot(p)=fot(k);
p=p+1;
end
end
```

```
%Make the number of leading edges for both codes the same.
if length(ledp)>length(leot)
ledp=ledp(1:length(ledp)-1);
elseif length(ledp)<length(leot)
leot=leot(1:length(leot)-1);
end

%Determine number of samples between each pair of leading edges, place these values into
%a vector, 'sampdiff'.
sampdiff=leot-ledp;

%Determine 'loss of lock', and 'out of linear region' outputs.
if nargin>4
loselock=find(abs(sampdiff)>(lock*s_freq/chp_rate)); %Determine the samples where the
%'loss of lock' threshold is
%is exceeded.
if loselock~=[]
disp(['            ';'loss of lock']) %If the threshold has been exceeded at any two pair
%of leading edges, display 'loss of lock'.
end
end

%Repeat similar steps for 'out of linear region'
if nargin>5
out_lin=find(abs(sampdiff)>(lin*s_freq/chp_rate));
```

```
if out_lin~=[]
disp(['                    ';'out of linear region'])
end
end

%Calculate rms error (in chips)
rms_err=chp_rate*sqrt(mean(sampdiff.^2))/s_freq;
```

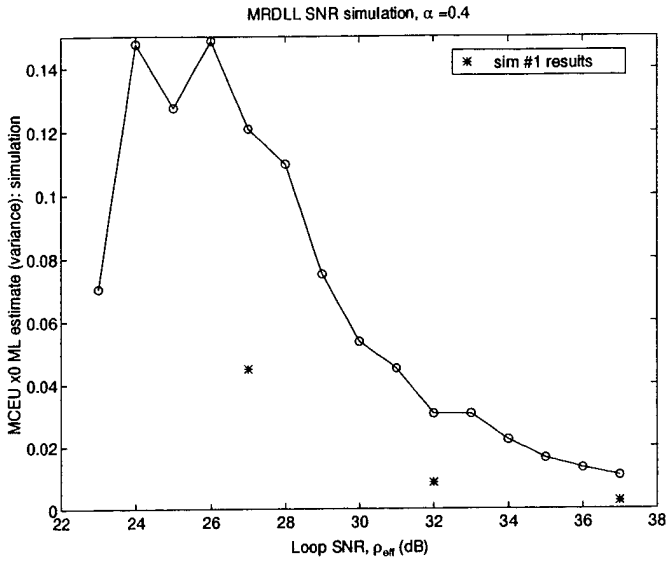

Figure 94. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_0$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.4$.

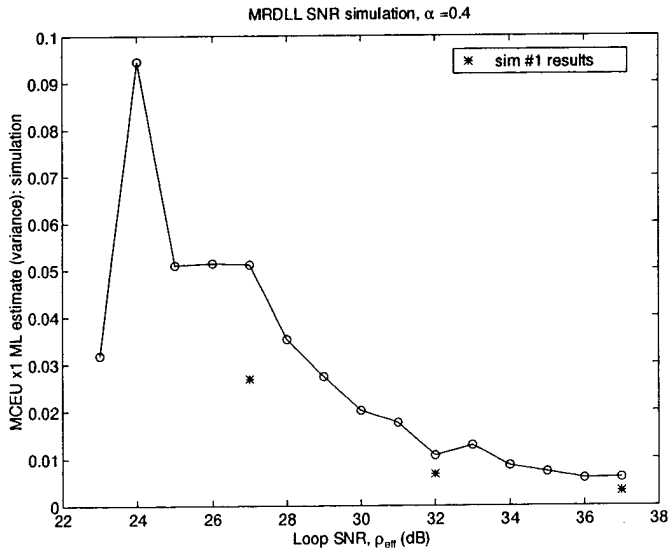

Figure 95. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_1$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.4$.

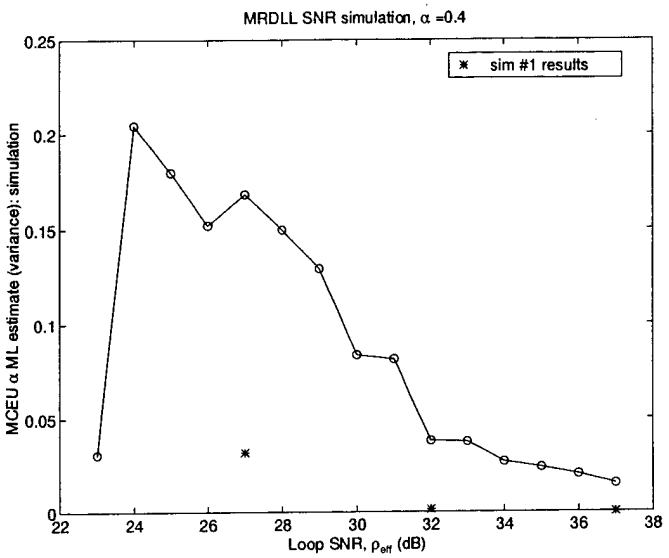

Figure 96. Simulation results showing the variance of the MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.4$.

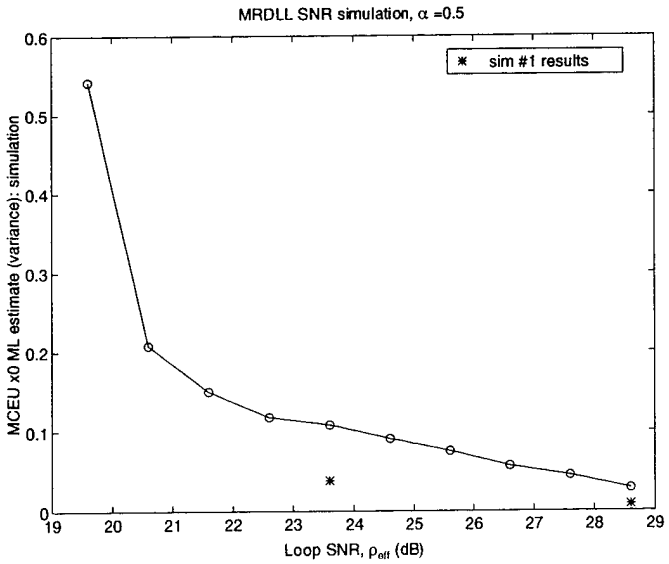

Figure 97. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_0$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.5$.

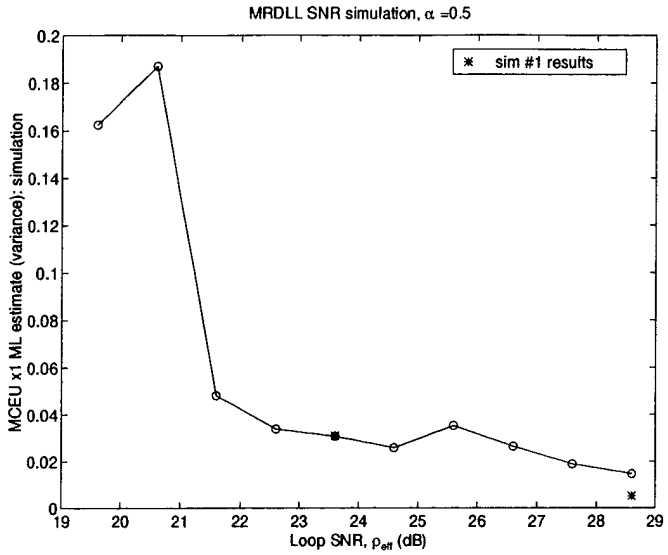

Figure 98. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_1$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.5$.

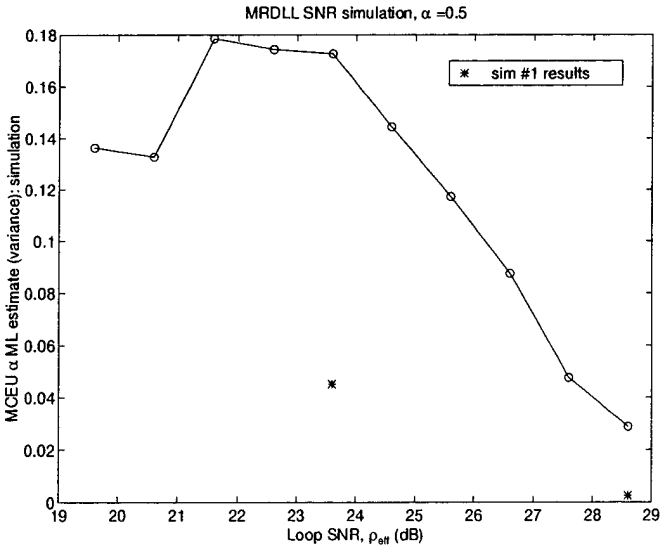

Figure 99. Simulation results showing the variance of the MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.5$.

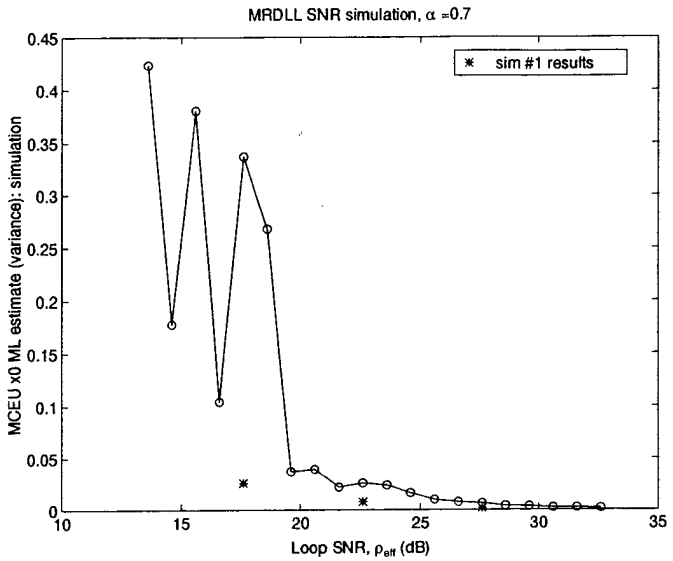

Figure 100. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_0$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.7$.

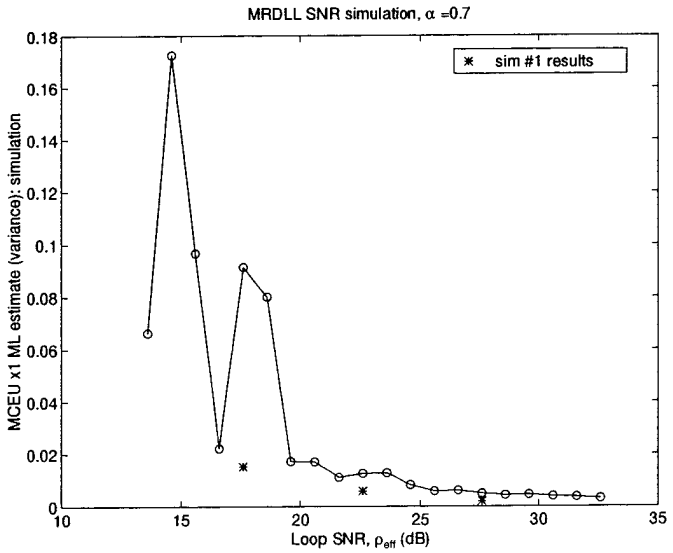

Figure 101. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_1$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.7$.

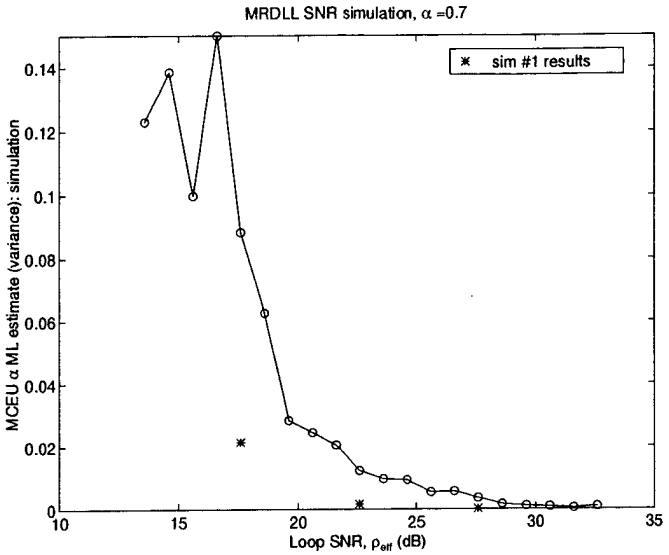

Figure 102. Simulation results showing the variance of the MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 0.7$.

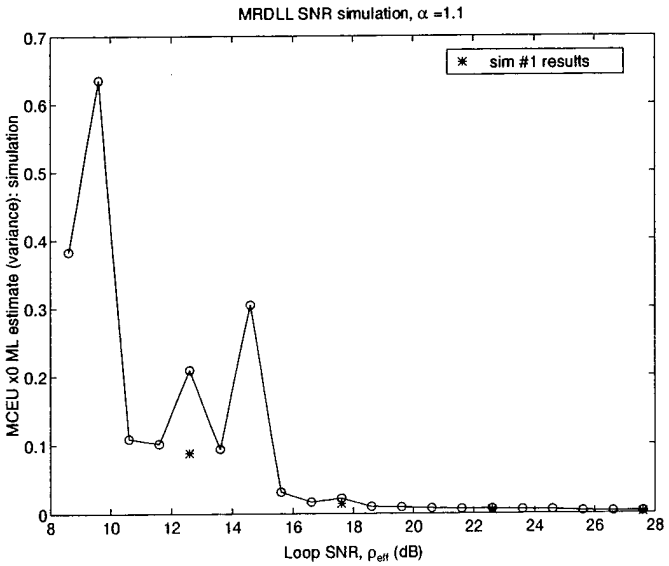

Figure 103. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_0$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.1$.

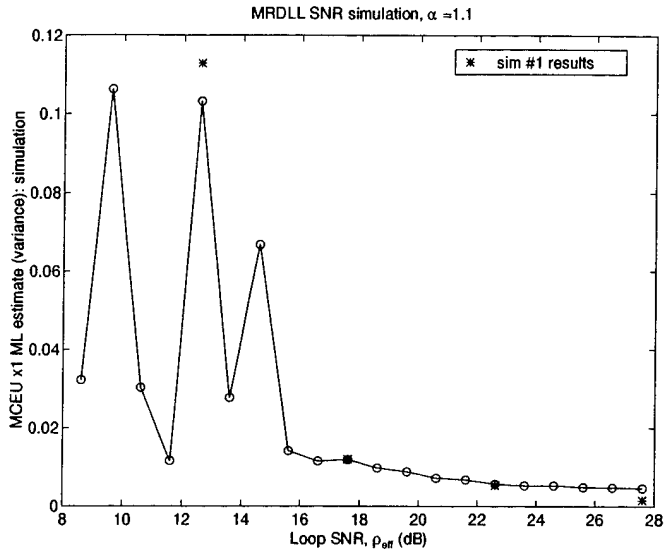

Figure 104. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_1$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.1$.

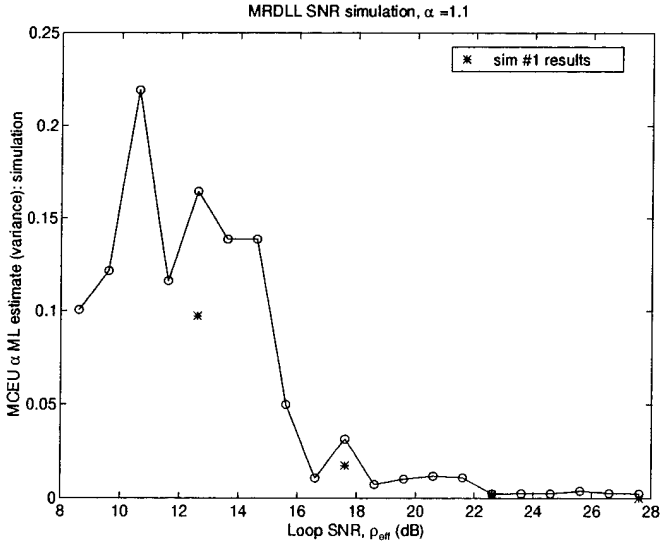

Figure 105. Simulation results showing the variance of the MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.1$.

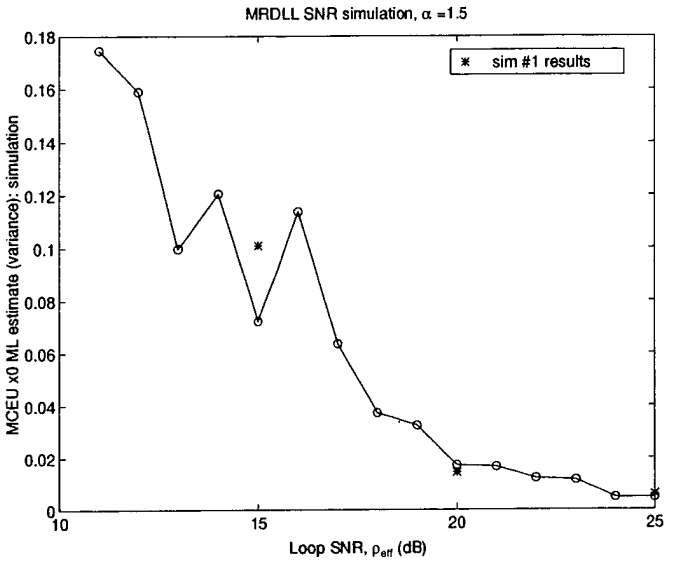

Figure 106. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_0$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.5$.

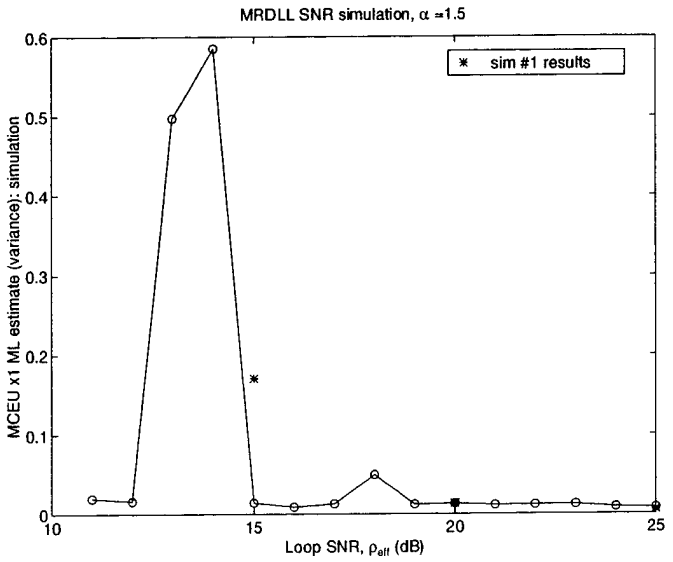

Figure 107. Simulation results showing the variance of the MCEU ML estimate, $\widehat{x}_1$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.5$.

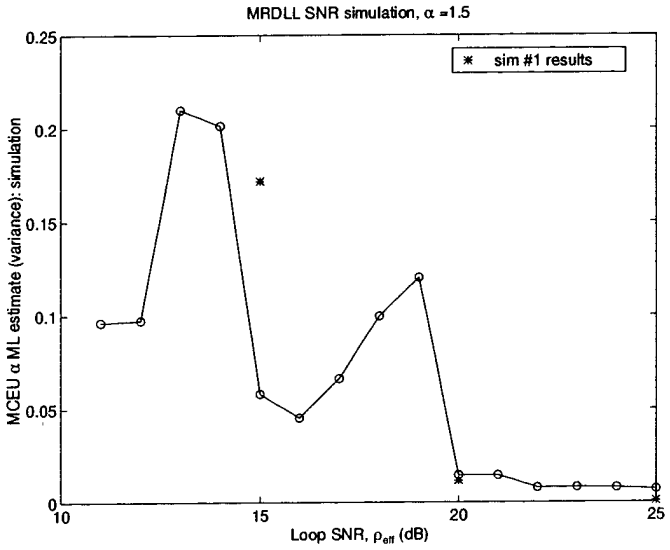

Figure 108. Simulation results showing the variance of the MCEU ML estimate, $\widehat{\alpha}$ for different values of effective MCTL loop SNR, ρ_{eff} (in dB) when $\alpha = 1.5$.

Bibliography

1. Åström, Karl and Bjorn Wittenmark. *Adaptive Control*. New York: Addison-Wesley, 1989.
2. Gardner, Floyd M. *Phaselock Techniques*. New York: John Wiley and Sons, 1979.
3. Jeruchim, Michel C. and others. *Simulation of Communication Systems*. New York: Plenum Press, 1992.
4. J.J. Spilker, Jr. "GPS Signal Structure and Performance Characteristics," *Global Positioning System: Papers Published in NAVIGATION*, *1* (1980).
5. Peterson, Roger L. and others. *Introduction to Spread Spectrum Communications*. Englewood Cliffs, New Jersey: Prentice-Hall, 1995.
6. Scharf, Louis L. *Statistical Signal Processing*. New York: Addison-Wesley, 1991.
7. Sheen, Wern-Ho and Gordon L. Stuber. "A New 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.
8. Stuber, Gordon L. and others. "A Coherent Tracking Loop for Direct Sequence CDMA Systems." *First European Personal and Mobile Communications Conference*. 221–226. November 1995.
9. Townshend, Brian R. and others. "Performance Evaluation of the Multipath Estimating Delay Lock Loop," *NAVIGATION: Journal of the Institute of Navigation*, *42*(3):503–514 (Fall 1995).
10. 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).
11. VanNee, Richard. "Multipath effects on GPS code phase measurements," *NAVIGATION: Journal of the Institute of Navigation*, *39*(2):177–190 (Summer 1992).
12. VanNee, Richard. "Spread-Spectrum Code and Carrier Synchronization Errors Caused by Multipath and Interference," *IEEE Transactions on Aerospace and Electronic Systems*, *29*(4):1359–1365 (October 1993).
13. 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.
14. Weill, Lawrence. "Achieving Theoretical Accuracy Limits for Pseudoranging in the Presence of Multipath." *Proceedings of the ION GPS-95 8th International Technical Meeting*. 1521–1530. September 1995.

Vita

Captain Mark C. Laxton [redacted] He graduated from University High School in 1986 and went on to attend Washington State University. In December 1990, he graduated WSU with a Bachelor of Science Degree in Electrical Engineering, where he also received his ROTC commission as an Air Force Second Lieutenant. Captain Laxton reported to the 2872d Test Squadron at Hill AFB, Utah in September 1991 and flew as a software/munitions test engineer for the F-16A/B fighter aircraft. In 1995, he was assigned to the Air Force Institute of Technology to earn his Master of Science Degree in Electrical Engineering.

[redacted]

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	December 1996	Master's Thesis

4. TITLE AND SUBTITLE
Analysis and Simulation of a New Code Tracking Loop for GPS Multipath Mitigation

5. FUNDING NUMBERS

6. AUTHOR(S)
Mark C. Laxton, Capt, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GE/ENG/96D-10

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Captain Tony Romano
WL/AACN
Wright Laboratory
WPAFB OH, 45433

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This thesis proposes a new direct-sequence spread spectrum (DS/SS) code phase tracking loop which mitigates the effects of multipath interference on code phase tracking error; such errors can translate to significant range measurement errors in DS/SS ranging systems such as Global Positioning System (GPS). The new code tracking loop, called the modified RAKE delay-lock loop (MRDLL), uses maximum-likelihood (ML) signal parameter estimation to determine the amplitude, carrier phase, and relative propagation delay of both a direct-path and a reflected signal; a multiple-correlator code phase tracking loop then exploits these ML signal estimates to remove the tracking error introduced by the reflection. A preliminary analysis showed that the MRDLL's linear tracking region varied with the reflected signal parameters; therefore, an adaptive loop controller (ALC) was introduced to allow the loop designer to fix dynamic specifications such as loop natural frequency. Analysis and computer simulations demonstrated that, when multipath was present, the MRDLL exhibited a significantly lower steady-state code phase tracking error than that of the standard non-coherent delay-lock loop (NCDLL), which is typically used in GPS receivers. In an ideal multipath-free environment, the NCDLL is still the best choice for code phase tracking.

14. SUBJECT TERMS	15. NUMBER OF PAGES
GPS, spread-spectrum, multipath, maximum-likelihood estimation, delay-lock loop, code tracking, synchronization, adaptive control	161
	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	UL