Research Article

The Effect of Intermittent Signal on the Performance of Code Tracking Loop in GNSS Receivers

Chung-Liang Chang

Department of Biomechatronics Engineering, National Pingtung University of Science and Technology, No. 1 Shuefu Road, Neipu, Pingtung 91201, Taiwan

Correspondence should be addressed to Chung-Liang Chang, chungliang@mail.npust.edu.tw

Received 17 March 2011; Revised 11 June 2011; Accepted 21 June 2011

Academic Editor: Tzi-Dar Chiueh

Copyright © 2011 Chung-Liang Chang. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper analyzes the code tracking performance in the presence of signal blanking in Global Navigation Satellite System (GNSS). The blanking effect is usually caused by buildings that obscure the signal in either a periodic or random manner. In some cases, ideal blanking is used to remove random or periodic interference. Nevertheless, the effect of temporary discontinuity of signal often leads to the tracking and position error. To analyze this problem, three types of blanking model are considered: no blanking, periodic blanking, and random blanking of the signals input into the code tracking loop. The mean time to lose lock (MTLL) is to assess the performance of code tracking system under signal blanking. Finally, the effect of steady-state tracking errors on the performance of tracking loop resulting from blanking environment is also discussed.

1. Introduction

The Global Navigation Satellite System (GNSS) provides information of position, velocity, and time by means of spread spectrum techniques [1, 2]. An essential way to predict the performance of GNSS system is the ability of the code tracking loop to maintain “lock”. When the signal is blanked, most of the GNSS positioning methods suffer from degradation in accuracy and increase in the receiver processing time [3]. The blanking effect is usually caused by buildings, wall, trees, rotation of vehicle, or random manners that obscure the signal periodically [4]. Besides the above factors, solar radio bursts caused by sunspots also cause the receiver to receive blank signal for a long time [5]. Of course, the use of ideal blanking method can mitigate the effect of pulse interference on positioning performance of receiver [6, 7]. With regards to the improvement methods for blanking effect, studies in early research only determine and deal with the phenomenon prior to signal correlation [8]. When the signal suffers from blanking effect, literature review on the postprocessing of code tracking is relatively few.

In this paper, the performance of code tracking loop in the presence of noise and blanking is assessed. The code tracking loop considered here is a second-order delay lock loop (DLL) with an early-late gating of 1/2 chip. Three blanking models are presented: no blanking, periodic blanking, and random blanking. The effect of random blanking on the loops is more emphasized because two blanks may appear contiguously, which causes the loops to be without signal for a longer time. It inevitably influences the signal tracking performance. Therefore, the mean time to lose lock (MTLL) is shown to evaluate the performance of code tracking loop under the presence of signal blanking.

The remainder of this paper is organized as follows: Section 2 gives a description of signal blanking model and the diagram of code tracking loop. In Section 3, the comparison of loop performance is discussed. Then, survey of code tracking technique is also presented in this section. At last, a brief summary concludes in Section 4.
2. Problem Formulation

2.1. Signal Model and System Description. The major goal of this paper is to analyze the blanking effect on the performance of code tracking loop. The Doppler uncertainties and data modulation are ignored. Based on these assumptions, the received mathematical model in the baseband is given by

\[ r(t) = \sqrt{P}m(t)\tilde{z}_0(t - t_0)\cos(\omega_0t + \phi_0) + w(t), \]  

(1)

where \( P \) is the incoming signal power, \( \tilde{z}_0(\cdot) \) is the filtered maximum length pseudorandom noise (PRN) code sequence with a delay of \( t_0 \) seconds and chip period of \( T_c \) seconds, \( \omega_0 \) is the signal frequency, and \( \phi_0 \) is the phase of direct signal. \( w(t) \) is low-pass filtered noise components that are independent, zero mean with power spectral density \( N_0/2 \). The blanking model \( m(t) \) is used to interrupt the signal so that the receiver is unable to receive consistent signal. The purpose of code tracking is to perform and maintain fine synchronization. A common fine synchronization strategy is to depict a code tracking loop which can track the code delay in the presence of a small frequency error. After the correct code phase is acquired by the code tracking loop, a phase locked loop (PLL) can be used to track the carrier frequency and phase.

Figure 1(a) demonstrates the coherent DLL (c-DLL) model for code phase measurement. The coarse/acquire (C/A) signal is generated and controlled by a blanking switch that can randomly or periodically “blank” the signal. The blanked signal combines white Gaussian noise and then is fed into the delay locked loop. Suppose perfect carrier demodulation is employed in the system. The error of discriminator output is then passed through the loop filter and the output is fed to the digital controlled oscillator (DCO) that steers the clock of the local PRN code generator. It is designed not only to reduce the tracking error but also to produce an accurate estimate of the original signal at its output. The model for the noncoherent DLL (nc-DLL) is shown in Figure 1(b), except for each low pass filter (LPF) followed by a square-law envelope detector. The difference between the choice of an nc-DLL or a c-DLL depends on whether the carrier frequency and phase are known or not. If both of them are known a priori, the c-DLL can be adopted. When neither of them is known, the nc-DLL must be employed to track the received signal.

2.2. Blanking Model. The defined process \( m(t) \) in (1) is a collection or ensemble of time waveforms that change from 1 to 0, from 0 to 1, or remain the same at possible transition time point that is equally spaced a distance \( T \) apart along the time axis. It is worth noting that when \( m(t) = 0 \) for all \( t \), the receiver simply receives noise term. Suppose that the no blanking model is given by

\[ \bar{m}(t) = 1 \quad \forall t \in [0, T]. \]  

(2)

The second is periodic blanking model that is modeled to have a \( T_a \) ms duration in a period of \( T_r \) ms for a duty cycle of \( T_p = T_a/T_r \) and is described on each subinterval for \( i \)th period of the time axis depicted as follows:

\[ \bar{m}(t) = \begin{cases} 1, & \text{for } iT_r + T_a \leq t \leq (i + 1)T_r, \\ 0, & \text{for } iT_r \leq t \leq iT_r + T_a \end{cases} \]  

(3)

for \( i = 0, 1, 2, \ldots, M \),

where \( M = T/T_r \) is the total times of blanking period.

The last model is random blanking. A source generates a sequence of two symbols, say \( u \) and \( v \) (blanking is “u”; without blanking is “v”). The \( u \) and \( v \) can denote the outcome of flipping an honest coin. Assume that the probability of getting a head on a single trial is 1/2 and that tosses at different times are independent. The sample space consists of all possible double infinite sequences of heads and tails:

\[ \mathcal{S} = \{ \xi : \xi = (\ldots, \delta_{-1}, \delta_0, \delta_1, \ldots) \text{ for } \delta_n \in \{u, v\} \}. \]  

(4)

Suppose there are \( N \) segments among fixed time \( T \) and each segment equals \( T_a \) second. The random blanking process, \( m(t) \), using 0 and 1 as values, is described on each subinterval of the time axis as:

\[ m(t, \xi) = \begin{cases} 1, & \text{if } \delta_n = u, \\ 0, & \text{if } \delta_n = v \end{cases} \]  

(5)

for \( nT_a \leq t \leq (n + 1)T_a \), \( n = 0, 1, \ldots, N \),

where \( N = T/T_a \) is the total blanking times. It is impossible for us to write a single expression in terms of random variables for the process. Nevertheless, it is possible to characterize the process by specifying certain statistical properties. The coin assumption shows that the probability of getting either a head or a tail (1 or 0 for the process during each interval) is 1/2. Therefore, the mean, \( E[m(t, \xi)] \), of the process can be found directly from the first-order density by [9]

\[ \eta_m(t) = E[m(t)] = \int_{-\infty}^{\infty} x f_m(x)dx = \frac{1}{2} + \frac{1}{2}(0) = 0.5. \]  

(6)

Suppose \( p \) times of blanking take place within \( N \) times, the probability of blanking is \( \Pr(u) = p/N \) and no-blanking probability is \( 1 - \Pr(u) \). The mean value can be calculated using (6). Figure 2 shows the no-blanking, periodic blanking, and random blanking model.

3. Performance Analysis

3.1. Coherent DLL. This section evaluates the effect of blanking on the performance of code tracking loop. The loop stabilizes when the discriminator output maintains zero. A c-DLL attempts to nullify the discriminator output by adjusting the estimated code delay. The outputs of correlators
The correlated signal with early and late correlators can be approximated as follows:

\[ \Lambda_E^c(t) \approx \sqrt{P}m(t)R\left(t_0 - \hat{t}_0 + \frac{\Delta}{2}T_c\right) \cos(\phi_0 - \hat{\phi}) + w_E, \]

\[ \Lambda_L^c(t) \approx \sqrt{P}m(t)R\left(t_0 - \hat{t}_0 - \frac{\Delta}{2}T_c\right) \cos(\phi_0 - \hat{\phi}) + w_L, \]

where \( w_E \) and \( w_L \) are total noise components, \( R(\cdot) \) denotes autocorrelation function of the PRN sequence and is defined as follows:

\[ R(t_0) = \frac{1}{T_c} \int_{0}^{T_c} \tilde{c}_0(t)\tilde{c}_0(t-t_0)dt = \begin{cases} 1 - \frac{|t_0|}{T_c}, & |t_0| \leq T_c, \\ 0, & |t_0| > T_c. \end{cases} \]

Figure 2: Three blanking models: (a) no-blanking, (b) periodic blanking, and (c) random blanking.
Under the assumption of wide front-end bandwidth, and the estimated phase error $|\varepsilon_c| \leq \Delta/2$, the normalized slope $K_c$ is defined by

$$K_c = \frac{dS^c_n(\varepsilon_c)}{d\varepsilon_c} \bigg|_{\varepsilon_c=0} = 2(2-\Delta).$$

(11)

Then the error signal is expressed as

$$e_c = K_c \sqrt{P} \varepsilon_c + \nu_c = 2(2-\Delta)\sqrt{P} \varepsilon_c + \nu_c.$$  

(12)

An approximation to the effect of blanking can be obtained by looking at the average power after blanking. Assume that the inverse blanking duration is much greater than the loop bandwidth, (12) can be modified as

$$e_c = 2(2-\Delta)\alpha \sqrt{P} \varepsilon_c + \nu_c,$$  

(13)

where $\alpha$ can be considered as the average power reduction factor owing to blanking effect.

Figure 3 depicts the block diagram of the linearized code tracking loop in which $1/s$ is the s-transform operator of the DCO, $F(s)$ is the transfer function of loop filter, and $\alpha K_c \sqrt{P}$ is the sensitivity of the discriminator. The closed-loop transfer function $H_c(s)$ is

$$H_c(s) = \frac{(2\alpha(2-\Delta)\sqrt{P}/T_c)F(s)(1/s)}{1 + (2\alpha(2-\Delta)\sqrt{P}/T_c)F(s)(1/s)}.$$  

(14)

For the second order, the $F(s) = a(1 + b/s)$. Where $a$ and $b$ are constant which are related to natural frequency, damping ratio, and closed loop equivalent noise bandwidth. In simulation system, second-order DLL is adopted to assess the performance of tracking system.

In particular, let $B_n$ be the equivalent noise bandwidth: the variance on code delay error estimation due to noise in the closed-loop operation is shown as follows:

$$E[\varepsilon_c^2] = \sigma_c^2$$

$$= \frac{1}{(2\alpha(2-\Delta)\sqrt{P})^2} \int_{-\infty}^{\infty} |H_c(f)|^2 S_n(f) df$$  

(15)

$$= \frac{N_v B_n}{4\alpha^2(2-\Delta)^2 P'},$$

where $S_n(f)$ represents the power spectral density of noise $v_n$, $E[\cdot]$ denotes the ensemble average operator, and the noise density $N_v$ (after low-pass filtering to smooth the transients) has a value at low frequency of $K_c^2 N_v / 2 T_c$. Thus, the variance of noise in the delay error can be expressed as

$$\sigma_c^2 = \frac{B_n \Delta}{2\alpha^2(2-\Delta)(P/N_0)}$$  

(16)

where $P/N_0$ is the carrier to noise density ratio.

3.2. Noncoherent DLL. The nc-DLL simply adds a square-law envelope detector after the low-pass filter in Figure 1(b).

Similar formulations can also be made for the noncoherent case. Based on this premise, the error signal driving the loop filter is given by

$$e_{nc} = aP \left( R^2 \left( e_{nc} + \frac{\Delta}{2} T_c \right) - R^2 \left( e_{nc} - \frac{\Delta}{2} T_c \right) \right) + \nu_{nc}$$

(17)

where $S^c_n(e_{nc})$ is the discriminator function for nc-DLL case. The definitions of $\nu_{nc}$ and $e_{nc}$ are the same as those in (9) and (12), except for the slight difference of nc-DLL scenario discussed here. For the linearized discriminator gain $K_{nc}$ is expressed as

$$K_{nc} = \frac{dS^c_n(e_{nc})}{d e_{nc}} \bigg|_{e_{nc}=0} = 2(2-\Delta), \quad \text{for} \quad |e_{nc}| \leq \Delta/2.$$  

(18)

The variance of noise $\nu_{nc}$ can be described as follows:

$$\sigma_{nc}^2 = aPN_v B_n + 2N_v^2 B_n^2.$$  

(19)

For the numerical simulations, the error signal $e_{nc}$ given in (17) is normalized to the input noise power $P_n = N_v B_n/2$. Thus, the normalized signal power and loop noise are described by

$$P' = \frac{P}{P_n} = \frac{2(P/N_0)}{B_n},$$

$$\nu'_{nc} = \frac{\sigma_{nc}}{P_n} = \frac{\sqrt{8 + \frac{4\alpha P/N_0}{B_n}}}{B_n}.$$  

(20)

Therefore, the normalized error signal can be expressed in the form of

$$e'_{nc} = aP' S^c_n(e_{nc}) + \nu'_{nc} = a \left( \frac{2P/N_0}{B_n} \right) K_{nc} e_{nc} + \sqrt{8 + \frac{4\alpha P/N_0}{B_n}}.$$  

(21)

Substituting (18) into (21), the error signal can be expressed as

$$e_{nc}' = \frac{4\alpha P/N_0(2-\Delta) e_{nc}}{B_n} + \sqrt{8 + \frac{4\alpha P/N_0}{B_n}}.$$  

(22)
Finally, the closed-loop transfer function $H_{nc}(s)$ is given by

$$H_{nc}(s) = \frac{(2\alpha(2 - \Delta)P'/Tc)F(s)(1/s)}{1 + (2\alpha(2 - \Delta)P'/Tc)F(s)(1/s)}. \quad (23)$$

The noise variance in the delay error can be computed and is shown as follows:

$$E[e_{nc}^2] = \sigma^2_{nc} = \frac{1}{(2\alpha(2 - \Delta)P'/Tc)^2} \int_{-\infty}^{\infty} |H_{nc}(f)|^2 S_{\nu_{nc}}(f) df. \quad (24)$$

Now approximate $S_{\nu_{nc}}(f)$ with $S_{\nu_{nc}}(0)$ and assume that the noise spectral density is essentially flat across the loop bandwidth [10]. Thus (24) can be modified as

$$\sigma^2_{nc} = \frac{2B_{n}S_{\nu_{nc}}(0)}{(2\alpha(2 - \Delta)P'/Tc)^2} \quad \equiv \frac{B_n\Delta}{2\alpha(P/N_0)} \left(1 + \frac{2}{\alpha(2 - \Delta)(P/N_0)T_c}\right). \quad (25)$$

3.3. $\alpha$ Factor. When the time delay estimation error is small, and the phase can be correctly estimated, the variance of code delay error due to noise in the closed-loop operation is obtained which uses (16) and (25). However, the $\alpha$ value influences the tracking performance in the whole loop. The $\alpha$ value is set as one under no-blanking. Under the model of periodic blanking, the value of duty cycle $T_P$ affects the reception duration of desired signals. That is, the larger the duty cycle, the higher probability the tracking loop may lose lock of the signals and also the larger the variance of error. This is because the correlation loss caused by signal discontinuity leads to the reduction of reception power during correlation process [4]. Thus, in the case of periodic blanking, the reduction factor for c-DLL is depicted as $\alpha = 1 - T_u/T_r$. In addition, regarding random blanking case, $\alpha$ is set as $Pr(u)$. Substitute $\alpha$ into (16) to find the variance of code delay error, which is given by

$$\sigma^2_{c} = \begin{cases} \frac{B_n\Delta}{2(2 - \Delta)(1 - (T_u/T_r))^2 P/N_0} & \text{for periodic blanking,} \\ \frac{B_n\Delta}{2(2 - \Delta)(1 - Pr(u))^2 (P/N_0)} & \text{for random blanking.} \end{cases} \quad (26)$$

With respect to nc-DLL case, given that $B_nT_c \ll 1$, the variance of code delay error owing to noise in the close loop operation can be given by (25). Similarly, blanking also affects tracking performance in nc-DLL. Thus, substitute $\alpha$ into (25) to find the variance of code delay error, which is expressed as

$$\sigma^2_{nc} = \begin{cases} \frac{B_n\Delta}{2(1 - (T_u/T_r))(P/N_0)} \times \left(1 + \frac{2}{(2 - \Delta)(1 - (T_u/T_r))(P/N_0)T_c}\right) & \text{for periodic blanking,} \\ \frac{B_n\Delta}{2(1 - Pr(u))P/N_0} \times \left(1 + \frac{2}{(2 - \Delta)(1 - Pr(u))(P/N_0)T_c}\right) & \text{for random blanking.} \end{cases} \quad (27)$$

4. Numerical Results

4.1. Simulation Procedure. To effectively verify the effect of different blanking scenarios on the performance of code tracking loop, this chapter establishes a simulation system and presents results for the delay lock loop. The simulations are conducted using Matlab software. The following will describe the simulation procedure.
(1) Firstly, one digital IF data with desired signal is generated under the scenario of no-blanking effect and synchronized frequency shift. This data is then input to c-DLL and nc-DLL, respectively, to proceed with code tracking and estimate the carrier to noise density ratio ($C/N_0$) as well as a stable estimate of code delay error variance.

(2) The duty cycle of periodic blanking and probability of random blanking are set, respectively; both of which conduct blanking with previous digital IF data. The data after the process of blanking is then input to c-DLL and nc-DLL to estimate $C/N_0$ and stable estimate of code delay error variance.
(3) Digital IF data with different power of satellite signals is generated, respectively. The procedures of (1) and (2) are repeated to analyze the variance of code delay error under the blanking with different $C/N_0$.  

(4) Simulation of the relation between MTLL and different blanking models.  

Through the above simulation process, we can accurately analyze the standard derivation of the code phase error and loss of lock under different blanking scenarios.

### 4.2 MTLL Computation

MTLL is utilized to assess the performance of code tracking system and it also characterizes the mean time that the DLL stays in its tracking range ($\pm 3\Delta/2$ chip spacing) [11, 12]. One model for the MTLL is derived through solving the Fokker-Planck equation. The MTLL, $T_L^\Delta$, for the first-order DLL with $\Delta = 1$ chip spacing is obtained as depicted in the following:

$$2B_n T_L^\Delta = \frac{1}{2\sigma^2} \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} \exp \left[ -\frac{G(x)}{\sigma^2} \right] \exp \left[ \frac{G(y)}{\sigma^2} \right] dxdy,$$

(28)

where $\sigma^2$ and $\epsilon$ denote variance of code delay error and delay error, respectively. $G(x)$ is the integral of the discriminator function $G(x) = \int_{-\infty}^{+\infty} S_{\Delta}(\epsilon)d\epsilon$, which is an even function of $x$. Substituting the variance of code delay error in (26), (27), respectively, into (28), the $T_L^\Delta$ value under different scenarios of blanking is derived. Equation (27) is applicable to first-order and second-order DLL. In fact, the result of (27) is valid for any closed loop, with a discriminator function given by $S_{\Delta}(\epsilon)$. According to Holmes, the MTLL for the second-order DLL can be derived from (28) by decreasing $\sigma$ by 1 dB [13].

### 4.3 Performance Analysis

In all simulations, assume the reception time of digital IF data is 30 seconds, the noise bandwidth of c-DLL and nc-DLL is $B_n = 10$ Hz, respectively, the integration time is 2 ms, and the signal blanking time is 20 ms every unit. Generally, the smaller the $B_n$, the longer the computation time to lock code phase shift in code tracking loop. On the contrary, the larger the $B_n$, the larger the error of code phase. Typically, the value is set between 0.1 Hz and 10 Hz. Suppose that in periodic blanking, the duty cycle is 5% (every 400 ms) and the probability of random blanking is also 0.05. The major reason for such a sameness is to analyze the effect of the two blanking models on the performance of code tracking loop. Both analytical and simulation results are demonstrated in Figure 4. The analytical result for the c-DLL and nc-DLL under blanking is computed via the formula (6)–(26) in Section 3. In contrast, the simulation results are derived by averaging a series of 50 Monte Carlo closed-loop simulation results. When the front-end bandwidth is fixed, the tracking error is controlled, mainly, by the correlator spacing for an early minus late discriminator.

Figure 4(a) illustrates the duration of duty cycle in periodic blanking versus the result of tracking error under the scenario of $C/N_0 = 14$ dB-Hz. The threshold is set to $|\epsilon| \leq 2/3$ chips for $\Delta = 1$ chip case. The time for one sample of the delay error $\epsilon$ to exceed this threshold is the time to lose lock, since at either of these points, the tracking error $S$-curves essentially goes to zero.

The figure presents that when the duty cycle is above 15% (blanking time as 300 ms, duty cycle as 2000 ms), the tracking loop loses lock. Figure 4(b) demonstrates the relation between the probability of random blanking and the result of tracking error. It is shown that when the blanking probability is above 0.085, the loop loses lock (blanking time is 20 ms every unit).

Figure 5 indicates that MTLL is computed by averaging the time to lose lock in all 50 runs for different blanking cases with $B_n = 10$ Hz, and $T_c = 2$ ms, over a range of $C/N_0$. In Figure 5(a), it is observed that nc-DLL with random blanking drops to lose lock slightly earlier than the periodic and the no-blanking case. The same result is shown in Figure 5(b). Figure 6 illustrates and compares the standard deviation of tracking error (in the unit of chip) under the scenario of different blanking effects and two DLL structures. It can also be observed that under the scenario of lower $C/N_0$, random blanking has greater influence on the performance of two models of DLL than periodic blanking. In the case of higher $C/N_0$ ($C/N_0 > 30$ dB-Hz), the effects of periodic blanking and random blanking are pretty close. The results show a little degradation in code tracking performance under the small probability of random blanking and small duty cycle of periodic blanking. When the probability of random blanking is above 0.085 and the duty cycle of periodic blanking is above 15%, the signal loses lock.

### 5. Conclusion

The effect of blanking on the performance of c-DLL and nc-DLL has been presented in previous sections. The performance evaluation is the standard deviation of the phase error and loss of lock. Numerous blanking values are simulated. The effect of blanking in c-DLL degrades the receiver power by $-\log(a^2)$. Assume the duty cycle of blanking is smaller than inverse tracking loop bandwidth, the received power degradation is slightly proportional to the duty factor of blanking in the case of periodic blanking under the scenario of nc-DLL. Under the same prerequisite for random and periodic blanking, the simulation results show that the random blanking degrades the DLLs (c-DLL and nc-DLL) performance more severely than the other blanking cases. In addition, the effect of ideal periodic blanking on the performance of tracking loop serves as the reference for the future use of ideal blanking to mitigate pulse interference.

### Acknowledgments

Many thanks are due to editor and reviewers for their precious comments to improve the content of this paper. In addition, the author would like to thank the mechatronics laboratory, National Cheng Kung University for providing expensive instruments to conduct simulation and experiment tests and National Science Council in Taiwan for supporting this research under Grant NSC 99-2221-E-020-036.
References
