A software receiver phase lock loop analysis and design to implement adaptive phase tracking using a finite impulse response loop filter

Marco Rao, Università di Palermo
Letizia Lo Presti, Politecnico di Torino
Maurizio Fantino, Istituto Superiore Mario Boella
Giovanni Garbo Università di Palermo

BIOGRAPHY

Marco Rao was born in Palermo, Italy, in 1984. He received the summa cum laude Master Degree in Telecommunication Engineering from the Politecnico of Torino, Italy, in 2007. Since January 2008 he is a Ph.D. student in telecommunications at the University of Palermo. His main research interests lie in the area of GNSS navigation, with a particular emphasis on inertial aided navigation and receiver tracking loops.

Letizia Lo Presti is a Full Professor at Politecnico di Torino, Turin, Italy. She is the Head of the NavSAS Research Group. Her research activities cover the field of digital signal processing, simulation of telecommunication systems, array processing for adaptive antennas and the technology of navigation and positioning systems. She is author of three books written in Italian. She is the Scientific Coordinator of the Master on Navigation and Related Applications held by Politecnico di Torino. She actively cooperates with the officers of the United Nations—Office for Outer Space Affair (UN-OOSA), Vienna, Austria, in the framework of the UN/Italy fellowship program, with the aim of keeping the Master program permanently aligned with the UN needs and suggestions. She is involved in several international research projects and is WP leader in several projects related to Galileo funded by the GJU.

Maurizio Fantino was born in Cuneo (CN), Italy, in 1976. He received the degree in communication engineering and the Ph.D. degree in electronics and communications engineering from the Politecnico di Torino, Turin, Italy, in 2001 and 2006, respectively. He is a member of the Navigation Signal Analysis and Simulation (NavSAS) Group, Electronics Department, Politecnico di Torino, and his research interests cover the field of localization, navigation, and communication. In particular, his work is mainly focused on innovative architectures for high performance GPS and Galileo receivers. His interests cover the field of signal quality monitoring and the study of the optimization of the new GPS and Galileo signals.

Giovanni Garbo was born in Palermo, Italy, in 1960. He received the cum laude Master Degree in Electronic Engineering and the Ph.D. degree in Electronics and Computer Science from the University of Palermo, Italy, in 1985 and 1990 respectively. Since April 2006 he is a full Professor of telecommunications at the University of Palermo. He has authored several publications on international journals and contributions to international conferences. His main research interests lie in the broad area of wireless communications, with particular emphasis on Continuous Phase Modulation, Group Codes, Turbo Codes, Code-Division Multiple Access, OFDM and OFDM/OQAM Systems.

ABSTRACT

In this paper we analyze the design of a digital PLL for a GNSS software receiver. Even if the topic of phase tracking has been widely studied, we found it useful to provide a short walkthrough for the design of a PLL in a real software receiver, avoiding to introduce the common theoretical phase-based model and aiming to the practical implementation of a system that deals with frequencies. Nonetheless, our analysis and design will grow away from the canonical approach, in the way that we will not resort to analog filter theory. The result is a PLL which is more reactive than the ones usually found in literature and that implements a simple method to make its bandwidth adaptive with respect to the noise that affects the signal.
INTRODUCTION
Starting from the early 90’s, SDR technology has gained a growing attention in the telecommunication community. The reason for this popularity has to be found in the flexibility of this technology in the emulation of a wide range of radio devices.

Thanks to the continuous advance in the general purpose processors design, today, using a commercially available personal computer, it is possible to implement a real time fully software GNSS receiver (Borre et al., 2007).

Such a receiver is a useful tool in a research environment, because it allows to gain access to every single part of the receiver, giving the user the chance to remove a single functional block and to substitute it with a new block, in order to implement a different algorithm or just to patch the old algorithm.

In this paper we will consider one of the most important functional blocks of a GNSS receiver, the phase lock loop Phase Lock Loop (PLL). This topic has already been analyzed in literature (Kroupa 2003, Lindsey and Chie 1981, Roncagliolo and Garcia 2007, Stephens and Thomas 1995) and it is possible to identify mainly two criteria to the second order digital PLL design.

The first criterion is based upon analog PLL theory: analysis and design are led in the s domain, like in Kroupa (2003), while the z domain filter parameters are obtained by means of the bilinear transform. The main drawback of such a method is represented by the use of the bilinear transform, which is a good approximation at low frequencies, but it gets less precise at high frequencies.

The second criterion leads analysis and design in the z domain but the determination of the filter coefficients is derived in order to obtain the same impulse response of a second order underdamped analog system, like in Stephens and Thomas (1995).

Our main goal is to provide an original approach to the design of a second order adaptive PLL, without resorting to analog PLL theory.

Our design method allows to develop a PLL fitted to GNSS requirements. The choice of the Finite Impulse Response (FIR) loop filter aims to obtain a more responsive system, that shows an impulse response which is shorter with respect to the one of the canonical PLL with an Infinite Impulse Response (IIR) loop filter, the bandwidth being equal. The only constraints considered are the bandwidth of the filter and the transient duration, that has to be minimum.

As a further goal, we worked out a simple way to make our PLL adaptive. We implemented a look up table method in order to change the system bandwidth dynamically and accordingly to the system noise, keeping the transient minimum.

Signal model and tracking architecture
The input of our system is an IF digital signal given at the output of the frontend. We can write it in the following form (Hegarty and Kaplan, 2006)

\[ r[n] = r(nT_s) = \sum_{i=1}^{L_s} y_{IF,i}(nT_s) + n_{IF}(nT_s) \]  

(1)

where \( L_s \) is the number of satellites in view, \( T_s \) is the sampling interval, \( n_{IF} \) is the noisy term at IF and \( y_{IF,i}[n] \) are the samples of the signal

\[ y_{IF,i}(t) = \sqrt{2C_i c_i(t - \tau_i)d_i(t - \tau_i) \cos (2\pi (f_{IF} + f_{dop, i})t + \phi_i)} \]  

(2)

transmitted by the \( i \)-th satellite and received at the front-end output. \( C_i \) and \( c_i(t) \) are the received power and the spreading code of the \( i \)-th satellite, \( d_i(t) \) is the bit stream of the navigation message, \( f_{IF} \) is the IF carrier and \( \phi_i \) is the received carrier phase, which is random. The code and the navigation message feature the same delay, namely \( \tau_i \), while the term \( f_{dop, i} \) is the doppler shift due to the relative motion between the user and the \( i \)-th satellite. The \( t \) used in the superscript highlights the time dependency of the two parameters. The signal is BPSK modulated and the duration of a single bit is equal to 20 ms. Each bit transition is associated to a \( \pi \) phase rotation. We also notice that the GPS signal is a CDMA one, so we have to remove the \( i \)-th code to process the \( i \)-th satellite signal. In order to perform this task, we need to evaluate \( \tau_i \). Nonetheless we have to recover the \( i \)-th satellite carrier, because the nominal frequency is altered by the Doppler shift \( f_{dop, i} \). Once these values are correctly estimated, it is possible to extract the navigation message.

In a GNSS receiver there are three different loops demanded to evaluate the two parameters we need to perform demodulation. These are the Delay Lock Loop (DLL), the Frequency Lock Loop (PLL) and the PLL. In this section we will discuss shortly the first two loops, to make the reader conscious about the whole receiver structure. The PLL is analyzed in the next section. In the following we will refer to Figure 1.

A DLL is a loop demanded to evaluate the code delay \( \tau_i \), in order to align the local code with the transmitted code. Once the DLL has estimated the code delay within an uncertainty range of \( \pm 1 \) chip, it is possible to correlate the received signal with the local code, to obtain a pure (BPSK modulated) sine wave.
The frequency of the sine wave would be deterministic if the user relative velocity with respect to the considered satellite were zero, but it is not the case. The relative motion between the user and the reference is the source of the Doppler shift \( f_{\text{Dop}} \) that must be estimated in two steps. A first step is performed by the FLL, which is identical to the PLL but uses a different discriminator. The FLL provides a coarse estimated frequency that is in the lock-in range of the PLL, which is the loop involved in the subsequent fine frequency estimation.

### PLL architecture and operation

A PLL is a negative feedback control system and its aim is to track the total phase \( \theta[n] \) of a carrier, i.e. the frequency \( f_0 \) and the instantaneous phase \( \phi_n \) of a sinusoid. It is made of three main blocks, namely the PD, the LPF and the NCO. In the following we will analyze these blocks, referring to Figure 2.

The output signal of the PD has to be related to the difference between the input phase \( \theta[n] \) and the estimated phase \( \hat{\theta}[n] \). At first we compute the input signal correlation with a local sinusoidal carrier and its \( \pi/2 \) back shifted version, getting the in-phase and quadrature components, respectively. Using Werner formulas, we can get for the quadrature branch

\[
\sin \theta[n] \cos \hat{\theta}[n] = \frac{1}{2} \left[ \sin(\theta[n] - \hat{\theta}[n]) + \sin(\theta[n] + \hat{\theta}[n]) \right] = \frac{1}{2} \left[ \sin(2\pi(f_n - \hat{f}_n)nT_s + \phi_n - \hat{\phi}_n) - \sin(2\pi(f_n + \hat{f}_n)nT_s + \phi_n + \hat{\phi}_n) \right]
\]

and similarly for the in-phase branch

\[
\sin \theta[n] \sin \hat{\theta}[n] = \frac{1}{2} \left[ \cos(\theta[n] - \hat{\theta}[n]) - \cos(\theta[n] + \hat{\theta}[n]) \right] = \frac{1}{2} \left[ \cos(2\pi(f_n - \hat{f}_n)nT_s + \phi_n - \hat{\phi}_n) - \cos(2\pi(f_n + \hat{f}_n)nT_s + \phi_n + \hat{\phi}_n) \right]
\]

We notice that each branch has a low frequency component and a high frequency component, given by the difference and the sum of the input and local carrier total phases. These values are accumulated in the quadrature and in-phase integrators for a whole integration window \( T_I = NT_o \), where \( N \) is the number of samples included in an integration interval. At this stage the outputted values are low pass filtered due to the action of the integrators. If we consider a tight gap between the estimated frequency and the actual frequency and if the high frequency component has been filtered out by the integrator, we can describe the quadrature and in-phase components ad follows

\[
Q[k] \approx N^2 \cdot 2\pi \Delta f[k] T_s \frac{T_s}{2}
\]

\[
I[k] \approx N
\]

where \( \Delta f[k] = f_n - \hat{f}_n \), with \( k = \frac{n}{N} \), and \( \Delta \phi_n \approx 0 \). This notation implies that \( f_n - \hat{f}_n \) is fairly constant in the integration window \( T_I \). The computed correlation values are the inputs of the PD, that has to output the error signal

\[
e[k] = \tan^{-1}\frac{Q}{I}
\]

which is a function of the phase difference between the quadrature and the in-phase components. In the PLL we implemented for our software receiver, we decided to use the arc tangent as the discriminator function, because it can be shown that it is the ML estimator (Roncagliolo and García, 2007). Nonetheless, there are other choices besides the arc tangent discriminator, but we will not consider them in the following.

The LPF is then fed with the error signal \( e[k] \). Its role is to suppress the high frequency component in the error signal due to noise and to output the signal \( g[k] \), which is the control signal for the NCO.

The NCO is demanded to tune the oscillator phase to be synchronous with the input carrier phase. The z domain model for the NCO is the one of an accumulator

\[
H(z) = \frac{z^{-1}}{1 - z^{-1}}
\]
so that the output will be \( y[k] = y[k-1] + g[k-1] \). \( H(z) \) has a delay \( z^{-1} \) for the system to be causal, so the output \( y[k] \) is the phase difference that has to be compensated at time \( k \) and that has been computed using the date of the integration interval at time \( k - 1 \). To perform the frequency correction, the frequency shift around the oscillator reference frequency has been computed as follows

\[
\Delta f_a[k] = y[k] \cdot \frac{f_s}{\pi N} \tag{9}
\]

where \( f_s = 1/T_s \) is the sampling frequency and \( \Delta f_a[k] \) is the accumulated frequency shift. In fact, if we consider the recursive formula for the NCO output, we obtain

\[
y[k] \approx y[k-1] + \frac{Q[k-1]}{f[k-1]} = y[k-1] + \pi \Delta f_a[k-1] T_s N \tag{10}
\]

**Low pass filter analysis and design**

In our software receiver we chose to implement a FIR loop filter. This choice is justified by the system impulse response we manage to obtain, which has a shorter transient with respect to the one of the filter shown in many papers like Lindsey and Chie (1981) or Roncagliolo and Garcia (2007). Such a quick response has an obvious inconvenient in the shape of the frequency response, but it will be clear from the results shown in the simulation section that the obtained system has filtering properties that are good enough to deal with GNSS signals.

The filter we implemented shows two degrees of freedom, due to the two filter coefficients \( a \) and \( b \) that have to be chosen. The filter is shown in Figure 3 and it is followed by the \( z \)-domain representation of the NCO.

![Block diagram for a LPF and a NCO](image)

**Figure 3 - Block diagram for a LPF and a NCO**

There are two requirements to satisfy to determine \( a \) and \( b \). The first one is represented by the noise equivalent bandwidth of the closed loop transfer function. The second one is given by the accumulated partial energy and it is related to the impulse response transient time. In the following we will analyze these constraints.

The equivalent bandwidth of a system that is characterized by a closed loop transfer function \( H_c(z) \) can be expressed in terms of its transfer function as follows (Roncagliolo and Garcia, 2007)

\[
B_{eq} = \frac{1}{2\pi} \int_{P_{z} \rightarrow 1} H_c(z) H_c(z^{-1}) \frac{dz}{z} \tag{11}
\]

which is easy to compute, by means of Laurent series and residues. In the case of our filter, we can easily determine \( H_c(z^{-1}) \) so that

\[
H_c(z^{-1}) = \frac{(a - bz^{-1})z^{-1}}{1 - (1 - a)z^{-1} - bz^{-2}} \tag{12}
\]

We must notice that \( H_c(z^{-1}) \) shows two distinct real poles, because the discriminator \( \Delta \) of the equation obtained putting the denominator of \( H_c(z^{-1}) \) equal to zero is

\[
\Delta = (1 - a)^2 + 4b > 0, \ \forall \ a > 0, \ b > 0 \tag{13}
\]

where \( a \) and \( b \) are kept positive in order to simplify the analysis.

Resolving (11) and computing the residues inside the unitary circumference, it can be easily shown that

\[
B_{eq} = \frac{(-1 + b)b + (1 + a)(1 + b)}{(1 + b)(2 - (a + b))} \tag{14}
\]

In Figure 4 we show the equivalent digital bandwidth values varying \( a \) and \( b \).

**Figure 4 - Equivalent digital bandwidth as a function of \( a \) and \( b \).**

The second constraint is computed upon the impulse response \( h[n] \), which can be obtained computing the inverse \( z \)-transform of \( H(z^{-1}) \) and which is equal to

\[
h[n] = ((1 - a + c)n(a - a^2 - 2b + ac) + (1 - c)n(a^2 + 2b + a(-1 + c))) \\
\cdot c^{-1} \cdot 2^{-1-n}, \ \forall n \geq 0 \tag{15}
\]

where \( c = \sqrt{1 - 2a + a^2 + 4b} \).
Considering the impulse response and defining a threshold \( 0 < t \leq 1 \), it is possible to define the response time \( p \in \mathbb{N} \) as the minimum number of samples needed to accumulate the \( t \cdot 100\% \) of the total energy of the impulse response, i.e. the minimum \( p \) that verifies the following disequation

\[
\sum_{i=0}^{p} (h[i])^2 \geq t \sum_{i=1} (h[i])^2
\]  
(16)

The parameter \( t \) is an index of the time needed to consider the transient of the system exhausted.

We can now proceed to determine the filter parameters. In fact, given a bandwidth, one parameter between \( a \) and \( b \) is constrained by the selected value. Then, it is possible to evaluate the response time for a set of values for the unconstrained coefficient and to choose the value related to the minimum response time.

The coefficients \( a \) and \( b \) are to be found in an acceptability range, given by the two following statements:

- \( a \) and \( b \) must be positive;
- the closed loop poles modulus must be not greater than one.

The first assertion is to keep the analysis simple, because we could actually study the system for \( a \) positive and \( b \) negative. If both the coefficients were negative, the system would result unstable. The second condition is given in order to keep the system stability. These conditions determine the follow four disequations linear system

\[
\begin{align*}
    a &> 0 \\
    b &> 0 \\
    a + b &< 2 \\
    a - b &> 0
\end{align*}
\]  
(17)

and it describes a triangle of vertices \((0,0)\), \((2,0)\) and \((1,1)\) in the \((a, b)\) plane.

We show the system response time in Figure 5, varying \( a \) and \( b \).

Adaptive phase lock loop

Filter coefficients can be evaluated off-line and for different values or ranges of equivalent bandwidth. If more than one couple of coefficients satisfies the bandwidth requirements, the couple that keeps the transient time minimum will be chosen. In this way, it is possible to store the filter coefficients in a data structure and to recall them upon the system bandwidth request, optimizing the response time. A look up table is in our opinion a good way to store and recall the filter settings while the filter is running. Results are shown in the following section.

Experimental results

In our experiments we compared the performance of our PLL with respect to the one that implements an IIR loop filter. We will not provide any implementation detail for the IIR loop filter, but we suggest the reader to find a reference in [7].

The \( z \)-transform of the FIR loop filter impulse response is the following

\[
H(z) = a - b z^{-1}
\]  
(18)

The \( z \)-transform of the IIR loop filter impulse response is the following

\[
H(z) = \frac{a + b z^{-1}}{1 - z^{-1}}
\]  
(19)

In the following figures we considered a project equivalent bandwidth equal to 20 Hz, even if any other reasonable value would lead to the same conclusions. The correlator integration time is 1ms. These bandwidth and integration time values imply the following parameters for the FIR loop filter

\[
\begin{align*}
    a &= 0.1893 \\
    b &= 0.1775
\end{align*}
\]  
(20)

and the following parameters for the IIR loop filter, which is designed to have the required bandwidth and a damping factor equal to \( 1/\sqrt{2} \)

\[
\begin{align*}
    a &= 0.0532 \\
    b &= 0.0519
\end{align*}
\]  
(21)

The simple experiment we propose is the tracking of a sinusoidal wave with unitary amplitude affected by unitary variance zero mean white noise. This is not a realistic GNSS environment but it is useful for the comparison of the two different loop filters. The frequency of the sinusoid grows linearly by 0.1Hz each 1ms. The PLL are initialized with a frequency error of -15 Hz, which is supposed to be the frequency estimation error related to the FLL and which is in the

![Figure 5 - Response time as a function of a and b.](image)
pull in range of the PLL. Each time instant in the $x$ axis is equivalent to 1ms, which is equal to the integration time.

In the following figures, every graphs that shows a logarithmic scale error, consider the modulus of the error.

In Figure 6 we show the frequency ramp and the evolution of the frequencies estimated by the two different PLL. The analysis of the errors and system reactivity can be easily done considering the errors plot in Figure 7. The figure shows the linear error in the left part and the logarithm of the error modulus on the right, to allow the reader to better appreciate the transitory dynamics, on the left, and the steady state behaviour, on the right.

![Figure 6 - PLL tracking a frequency ramp.](image)

![Figure 7 - Errors while tracking a frequency ramp.](image)

Looking to the left part of Figure 7, we notice that the two PLL start from the same point, which is distant 15Hz from the actual frequency value. Our PLL corrects the estimated frequency with a steeper slope than the IIR loop filter. In the right part of Figure 7, we can easily notice that our filter manages to achieve its steady state in less than 50ms, while the other PLL takes more than 100ms to reach the same goal. Our PLL manage to achieve quickly an error of about 1Hz, which is more than enough for a software receiver to run smoothly. The IIR loop filter is characterized by a less noisy steady state. Actually, this is due to the fact that the PLL aims to recover the total phase. The frequency is subject to bigger variations in order to keep the total phase variance low.

Moreover, in Figure 8 we notice that a quicker response implies a smaller total phase error, because the sinusoid estimated by our PLL manages to run synchronously with the actual sinusoid in a short time, loosing less cycles than the IIR loop filter counterpart.

The different performance of the two designs can be justified by an analysis of their impulse and frequency responses.

![Figure 8 - Total phase residual errors.](image)

![Figure 9 - Impulse responses comparison](image)

In Figure 9 we show that the closed loop impulse response of our filter allows to gather the impulse response energy in the first samples, while the traditional PLL leans toward a distribution of the energy typical of an underdamped system.

In Figure 10 we analyse the closed loop frequency response of the two PLL. We can easily notice that both filter have a low-pass behaviour, but our filter has a narrower low frequency response and it attenuates high frequency less than the IIR filter counterpart. This feature is evident if we analyze again Figure 8, where the phase error of our filter is disturbed by high frequency noise, which is anyway negligible.
In the following we will show another experiment we made to test the adaptive algorithm we implemented. The setting is almost the same, because our goal is to track a frequency ramp. This time, once the PLL is aware of being near the solution, due to a decreasing in the variation of the correction, it changes its parameters in order to make its bandwidth narrower. This change will result in the achievement of a less noisy estimated frequency. The coefficients to be used are stored in a structure accessible by the software.

Our strategy allows us to recover a bigger frequency gap than the one we considered before. In the following example, the PLL is provided with a -80Hz shifted initial frequency. Such an error could not be recovered by a 20Hz equivalent bandwidth PLL. The PLL will be initialized with a 60Hz equivalent bandwidth and will change it to 40Hz, 20Hz and 5Hz. The strategy implemented might be more complicated than this one, in order to track more realistic frequency variations.

In Figure 11 the evolution of the error made by the adaptive filter is represented. We must notice that a constant bandwidth filter could not perform so well in terms of error variance or could not recover such a big initial frequency error. In fact, the initial settings of a 60Hz bandwidth allows to recover the big initial gap but, if it were not changed, it would imply a $1.8722\text{Hz}^2$ error variance, against the $0.0745\text{Hz}^2$ error variance of the adaptive filter. These variances were computed considering a longer simulation and evaluating the error variance once the steady state had been reached. The algorithm that allows the PLL to increase or decrease its bandwidth is based upon discriminator output covariance and the method and its optimization will be the object of future work.

Conclusions

In this paper we provided a guide and the fundamentals to understand and implement a good performing PLL. We did not resort to analog PLL and filter theory. The only constraint considered is the equivalent bandwidth requirements and the whole design is lead in the digital $z$ domain. This system is characterized by a simple design and architecture, a fast convergence, good robustness to noise and is flexible in the way the loop filter can be turned into an adaptive filter. We showed the pros and the cons of our system with respect to the PLL described in classical literature, showing results about a typical case of study of second order PLL, i.e. the frequency linear ramp. Nonetheless, we showed how our adaptive filter can recover a huge frequency error of the PLL, keeping a lower estimated frequency error covariance than a constant bandwidth PLL. Future work will show the advantages of the implementation of such a filter in the presence of a real GNSS signal and may aim to the design of a higher order PLL, to track faster variation of the carrier frequency.

REFERENCES


