Low Computational Cost GNSS GPS/GLONASS Maximum Likelihood Bit Synchronization Method

Gerardo Ludovico Puga, Pedro Agustin Roncagliolo, and Javier Gonzalo García

Laboratorio de Electrónica Industrial, Control e Instrumentación (LEICI),
Departamento de Electrotecnia,
Facultad de Ingeniería, Universidad Nacional de la Plata,
La Plata, Argentina
{gerardo.puga,agustinr,jgarcia}@ing.unlp.edu.ar

Abstract. A maximum likelihood bit synchronization method is proposed for GNSS signals that works on multiple millisecond integrations. Since the amount of samples that need to be processed in order to achieve bit edge synchronization is decreased, the peak processor load is reduced as well. This method also presents lower error rate than other bit edge synchronization algorithms, while improving error rate performance under low signal-to-noise ratios. The method can be used with both GPS and GLONASS data bit streams with few modifications. Finally, the method proposed is simple enough as to be implemented on real-time embedded GNSS receivers.

Keywords: GNSS, GPS, GLONASS, bit edge synchronization

1 Introduction

Global Navigation Satellite Systems (GNSS for short) work by using a constellation of satellites placed in a series of orbits around the earth. Each satellite transmits radio signals that can be decoded and processed by a receiver. Using the signals of several satellites (four or more) the receiver can calculate a navigation solution: position, speed, time and local clock drift.

There are several different GNSS Systems. GPS is the GNSS System maintained by the United States Department of Defense (DoD) and has been active since the 1980's [1], [2], [3]. GLONASS is an independent navigation system developed by Russia that has been completely operational since 1995 [4]. The European Union has been preparing another system called Galileo, but this system will not reach operational status before 2015, and complete deployment is scheduled for 2019. China has been using Beidou, a national navigation system that only has coverage over it's own national territory, since year 2000; a newer navigation system called Compass is being deployed by China that will offer

This work was supported by ANPCyT PICT 2011-0909 and by UNLP I-166.

global coverage in the same way the other GNSS systems do, but full operational status will only be reached by 2020.

Currently the only operative GNSS systems are GPS and GLONASS. While lack of funding and a series technical problems have had a detrimental effect on the operational status of the GLONASS system in past years, during the second half of the previous decade there's been a strong push to recover the system back to full operational status. As of May 2012 both GLONASS and GPS are in full operative status with global coverage.

A combined GPS/GLONASS receiver can use the signals of both systems to obtain a single navigation solution. This presents its own set of challenges because different GNSS systems transmit using different frequencies and modulations, and use different orbital data, with little or no regard for inter-compatibility. The advantages of using more than one system are greater availability of satellites, greater coverage, improved accuracy and independence of a single national authority. This is one of the efforts the GNSS group in the Facultad de Ingeniería of the Universidad Nacional de La Plata is currently working on.

The principles of operation of both GPS and GLONASS are very similar. Each satellite has a very precise on-board time reference. All the GPS satellites on-board time references are synchronized within a few microseconds to a single time reference called the GPS System Time, which is the time as indicated by an atomic clock maintained in one of the system's ground control stations. Similarly, all the GLONASS satellites are bound to a single time reference called th GLONASS Time located in the main control station located in Moscow. The GNSS signal transmitted by the satellites (either GPS or GLONASS) encodes the on-board time at the moment of transmission. A GNSS receiver decodes this information and processes it in order to obtain a navigation solution.

Before being able to decode the timing information embedded within the GNSS signal the receiver must perform synchronization with the signal at several levels: phase of the Direct Sequence Spread Spectrum (DS-SS) codification of the signal, data bit edge position, bit number within a data frame, and frame number.

A very important part of this synchronization is the millisecond ambiguity resolution, during which the receiver probes the signal in order to detect the position of data bit edges. Satellite data bits are sent at 50bps, and the spreading code sequences of both GPS and GLONASS are periodic sequences with 1 ms periods. The DS-SS sequence used to spread a signal is aligned with the bit edges of the signal in such way that a data bit edge always concurs with the start of new period of the DS-SS sequence. However, since there are 20 periods of the spreading sequence within each bit interval, once the receiver has managed to synchronize with the DS-SS sequence there are still 20 possible data bit alignments. This ambiguity needs to be solved before being able to extract the timing information present in the GNSS signal.

The classic algorithm for GPS millisecond ambiguity resolution the the histogram method [2], [8]. This method searches for sign changes in consecutive 1 ms correlation results. Performance is adequate when signal to noise ratios are

high enough, but degrades quickly for C/No values under 30 dB. Because of its simplicity this method has been extensively used for general purpose receivers.

There are other more complex methods that present higher sensitivity, allowing receivers to perform millisecond ambiguity resolution under extremely low C/No condition, such as those endured by GPS receivers for indoor applications and street level car receivers. These algorithms work by finding the bit edge candidate position that maximizes the recovered average bit energy [6], [7], [10], which is equivalent to choosing the maximum likelihood candidate [6]. These methods can work with C/No levels down to 12 dB [7].

Slightly modified versions of these algorithms can be used for GLONASS. Because of the presence of the meander code in the GLONASS data signal [4] most ambiguity resolution algorithms will perform significantly better.

Both the histogram and the energy maximization algorithms are designed to use 1 ms integration results. This is not desirable for embedded receivers that must work on real-time data: the processor must be interrupted every millisecond to either process the correlation results or store them for later processing. This increases the processor average load while the receiver is performing the millisecond ambiguity resolution of a recently found satellite signal. This increase must be accounted for when estimating the worst-case processor load. This peak processor load is a very important parameter when determining if the receiver will be able to perform in real-time ([5]). Since this peak load scenario accounts for a very small fraction of time of the receiver operation, the ratio between the average and the peak processor load values will be small.

While the histogram can be extended to work on multiple millisecond correlation times, this causes the reliability of the algorithm to decrease. This happens because if N millisecond samples are being used, the extended histogram method must not only decide whether or not there was a data bit sign transition, but it must also decide which bin to increase among N possible candidates. Thus the noise variance grows faster than the distance between the decision thresholds used to decide which histogram bin to increase, if any.

This paper proposes a new algorithm that can work on multi-millisecond integrations and still resolve the millisecond ambiguity with millisecond resolution. Given a sequence of correlation results, the method finds the position of the bit edges based on a Maximum Likelihood criterion. The computational cost of the method is kept low by prunning the decision tree early and using a trellis-like structure that at any given moment keeps only the most probable bit sequence candidates (including their edge positions) under consideration.

In this work we focus on 3 ms integrations. This number was chosen because in practical applications this algorithm is coupled with carrier and code phase tracking loops that work concurrently with the synchronization algorithm. While longer integration times would decrease considerably the processor load during the process of millisecond ambiguity resolution, any sign change during the integration time would cause a decrease in the correlation samples signal-to-noise ratio and introduce perturbations in the tracking loops. Moreover, if the integration time length is an even number of milliseconds there is a chance that data

bit sign changes at the bit edges (or mid-bit changes in GLONASS) might cause the signal component to be completely canceled leaving a pure noise sample. For these reasons $T_{int} = 3$ ms was chosen as a compromise between processor load reduction and synchronization tracking loops restrictions: it is large enough so that processor load can be decreased roughly threefold, while at the same time not large enough as let the tracking loops be affected by partially canceled samples due to data bit sign changes. This choice has one more advantage: 3 ms is an odd number, which means sign changes cannot completely cancel the signal energy component in the correlation samples.

It is important to note that while this work focuses on 3 ms samples it is almost trivial to extend this algorithm to other integration time lengths.

2 GNSS Signals

Both GPS and GLONASS satellites are able to transmit several different types of signal.

GPS satellites transmit both a civilian or coarse-acquisition (often referred as C/A) signal and an encrypted high-precision signal called P(Y). The P(Y) signal use is only possible for authorized users with access to the decryption keys. For the rest of this paper we will deal exclusively with the C/A signal data bit edge synchronization. Details of the signal structure can be found in [3].

GLONASS satellites also transmit two types of signals: a signal for civilian use, and another with higher precision for authorized users only. As in the case of GPS, we will deal exclusively with the civilian GLONASS signal. The signal structure can be found in reference [4].

2.1 GPS Signal

The C/A signal transmitted by all the GPS satellites consists of a 50 bps data stream modulated using BPSK on a 1575.42 MHz (L1 band) carrier. The signal is spread before transmission using a pseudo-random code. The spreading code is a periodically repeating 1023 chips long sequence that progresses at a rate of 1.023×10^6 chips/sec. Each satellite is assigned its own spreading code. This Direct Sequence Spread Spectrum (DS-SS) codification allows the receiver to separate the signals of different satellites arriving on the same frequency band, and the same time provides the fine (sub-millisecond) part of the transmission time encoded in the signal.

The spreading code sequence has a period of 1 ms, and is aligned with the data bit edges so that a bit edge always concurs with the start of a spreading code period. 20 spreading code periods fit in one data bit time interval. While the phase of the spreading code can be found by correlation because the spreading code used by each GPS satellite is known, the position of the bit edges cannot be determined this way because the bit sequence itself is unknown.

2.2 GLONASS Signal

The signal structure of GLONASS is similar to that of GPS: the civilian signal is made of a 50 bps data stream transmitted over a BPSK modulated carrier, and a DS-SS code is also used to spread the spectrum of the signal. In the case of GLONASS, though, all the satellites share the same spreading sequence. Multiple access in this case is achieved assigning each satellite a different carrier nominal frequency in such a way that the signal spectrum of different satellites do not overlap; each of these nominal carrier values is called a channel. Civilian GLONASS channels are in L1 band, at about 1602 MHz. In order to save frequency spectrum, more than one satellite can be assigned to the same channel if their position in the constellation is such that no user on earth will ever have direct line of sight to more than one of them at any given time. The spreading code is a 511 chips long periodically repeating sequence that advances at a rate of 511×10^3 chips/sec.

Just as in the case of GPS, each code period is 1 ms long and is aligned with data bit edges. In one bit time interval there are exactly 20 spreading code periods.

The presence of a meander code in GLONASS causes a forced sign transition mid-period during a bit time interval. This transition warrants that there is at least one data sign change in each data bit interval. This is important because the more data sign changes that occur during an observation interval, the better most bit edge synchronization algorithms will perform.

3 Signal Model

It is assumed that this algorithm works concurrently with the carrier and code phase tracking loops. This ensures that at any given moment the phase error in the estimates of carrier and code is negligible, and thus data bit power can be completely recovered in the in-phase correlation samples $I_p[n]$. Integrations are 3 ms long, and are aligned with the start of the spreading code period so as to ensure that all integration edges differ from all data bit edges by an integer number of milliseconds. The correlation samples will be modeled as

$$I_p[i] = S[i] + n[i] \tag{1}$$

where $S[i]$ is the area of the data bit sequence during the correlation time. Noise $n[i]$ is Additive White Gaussian Noise (AWGN) with zero mean and variance σ^2. $I_p[i]$ samples have been normalized so that the data bit area $S[i]$ is ± 3 when there are no sign changes during the integration interval.

$I_p[i]$ is a gaussian random sequence of independent samples with variance σ^2 and whose mean can be ± 3 if there are no sign transitions during the integration interval, or ± 1 if a sign change caused a partial cancellation of data bit area (and a decrease of the data signal energy present in the sample).

4 Naive Approach

Suppose for a while a noise-free signal, so that $I_p[i] = S[i]$; the sequence of correlation values $S[i]$ immediately identify the both the data bit edge positions and the data bit sequence. For example, given the following sequence of 3 ms samples obtained from a GPS signal

$$+3, +3, +1, -3, -3, -3, -3, -3, -3, +1, +3, +3, +3$$

it is possible to tell that there was a sign change 7 ms after the start of the first correlation, and then again 20 ms later. It is also possible to affirm that the data bits were $+1, -1, +1$.

Given a signal type (GPS or GLONASS), a data bit sequence and an alignment between correlations and bit edges, then the sequence of correlation samples $S^{(n)}[i]$[1] that will be obtained is completely determined. This is also true the other way around[2], so given a sequence of N correlation samples $S[i]$, if we find the sequence $S^{(n)}[i]$ that matches it then we have found both the bit sequence and the location of all the possible bit edges in the observation interval.

This is deceptively simple: there are 20 possible alignments of the integration interval within a data bit period, and given that the observation interval of N samples includes B bit edges, then there are 2^B data bit sequences. In order to find the position of the data bit edges, 20×2^B candidates $S^n[i]$ of N samples each need to be synthesized and matched against the observed sequence $S[i]$. Even for moderately short observation intervals (i.e., one second) $B \approx 50$, which means that the search space will be huge.

Because of this, the search process needs to be implemented in such a way that the number of candidates that need to be checked is reduced. Besides, in practice samples will be affected by noise, and thus matching should be done in such a way as to be resilient to noise-induced errors. These two problems can be dealt with separately: pattern matching and search space exploration.

5 Pattern Matching

Given a sequence of N observed samples $I_p[i]$ we want to measure the resemblance of this observation to a set of candidate hypothesis $S^{(n)}[i]$ and to determine the member of the set that best matches it.

Employing the Maximum Likelihood criterion it is easy to prove that under the assumptions of gaussianity and independence of the samples, the candidate

[1] The super-index (n) maps a set of sequence parameters to a given sequence that can will be synthesized for such parameters.

[2] Note that when processing a GPS signal, the *all-zeros* and *all-ones* data bit sequences provide no information about the location of bit edges, and thus cannot be disambiguated. GLONASS presents similar problems. These atypical sequences are easy to detect, though.

$S^{(n)}[i]$ that minimizes the following log-likelihood index

$$J(I_p, S^{(n)}) = \sum_{i=1}^{N} (I_p[i] - S^{(n)}[i])^2 \tag{2}$$

is the most likely match for the observed sequence. This provides us with a simple criterion to chose a hypothesis from the set of candidates that constitute the search space: find the hypothesis $S^{(n)}$ such that

$$\min_n J(I_p, S^{(n)}) \tag{3}$$

Better yet, this criterion suggests an easy way to prune the search space in order to reduce the computational complexity of the search process. See next.

6 Search Space Exploration

Given a candidate $S^{(n)}[i]$ the process of calculating $J(I_p, S^{(n)})$ in (2) can be thought as an iterative process where given the set of parameters of the hypothesis sequence $S^{(n)}$ (GNSS system, initial alignment, and data bit sequence) a state machine synthesizes each new sample $S^{(n)}[i]$, compares it against the observation $I_p[i]$, accumulates the quadratic error between the two, and then moves on to the next sample $i + 1$. After processing the last sample the total accumulated quadratic error will be $J(I_p, S^{(n)})$.

There are 20 possible initial alignments of a 3 ms integration within a 20 ms bit interval, and two possible data bit signs for the first candidate sample $S^{(n)}[0]$. The first sample of any candidate sequence will always be in one of these 40 initial (alignment, sign) pairs. We can start 40 state machines simultaneously, one for each pair, and process each sample in parallel in all of them. Each machine will progress generating a candidate sequence as it advances and will calculate the accumulated quadratic error of that sequence against the input $I_p[i]$ samples. If we allow each state machine to fork into two identical copies of itself whenever the future of the sequence can follow two possible paths depending on whether there is a data bit sign inversion or not, then after having processed the N-th sample all the possible candidate sequences will have been generated and tested, and the values of $J(I_p, S^{(n)})$ for all n will have been calculated. While this spanning-tree approach would save a great deal of work by avoiding the need to repeat calculations that are common to many candidates, it still requires large amounts of memory and processing time even for relatively small values of N.

We can save a lot more work if we trim all the branches in the tree that have no chances of producing the ML candidate: given two or more candidate sequences that are equal from certain point on, the one with the higher partial accumulated error up to that point can be safely ignored since its log-likelihood index will always turn to be higher than the one of at least one other candidate. If we do this, the ever-growing spanning-tree can be replaced with a fixed-size 40 states trellis. This allows for a great economy of resources: only the 40 best

candidates up to any point are kept, memory count is kept low and processor load is kept constant during processing because the number of state machines does not grow (for every time a state machine needs to fork, another one is destroyed). After processing the N-th observed sample, the ML candidate is the one with the smallest log-likelihood index among the 40 remaining candidates still present in the trellis.

6.1 Trellis

In Fig. 2 the trellis representation of the possible state transitions can be seen. Notice that there are only 6 states that can fork, only 6 merge points, and all merges are done between two possible paths. This allows for a very efficient implementation of this trellis.

Since the state naming is the same for both GPS and GLONASS, and since both systems have the same bit duration, the transition pattern for both trellis is the same. However, because of the differences in bit coding (because of the presence of the meander code in GLONASS) the sequence of expected $S[i]$ 3 ms integration values associated to each transition between states differs for each system.

Each transition in the trellis increases the accumulated quadratic error by an amount that that is the squared error between the input sample $I_p[i]$ and the expected $S[i]$ value for that transition, which will always be one of these four increments

$$\begin{aligned} ep3 &= (I_p[i] - 3)^2 \\ ep1 &= (I_p[i] - 1)^2 \\ em1 &= (I_p[i] + 1)^2 \\ em3 &= (I_p[i] + 3)^2 \end{aligned}$$

6.2 State Machine Definition

The states to be used in the trellis are defined as each possible combination of alignment of the 3 ms integration within the data bit interval, and data bit sign at the start of the integration. As said before, since there are 20 possible alignments and 2 initial signs, 40 states are needed. Each state will be referred by a number, see Fig. 1 for the naming of the states.

6.3 Implementation

Despite the number of states in the trellis, the transitions follow a very regular pattern and their number is limited. This allows a very efficient implementation. In fact, except for the six merge points, the trellis can be updated by circularly rotating the table of quadratic accumulated errors after having increased each accumulator with the error term associated to the transition; this can be done very efficiently by simply renaming states.

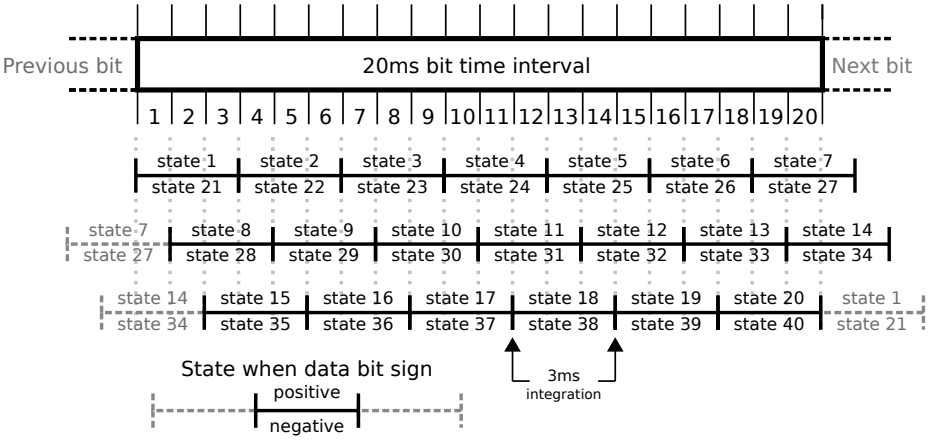

Fig. 1. Possible 3 ms integration positions within a 20 ms bit time interval, and state name assignments

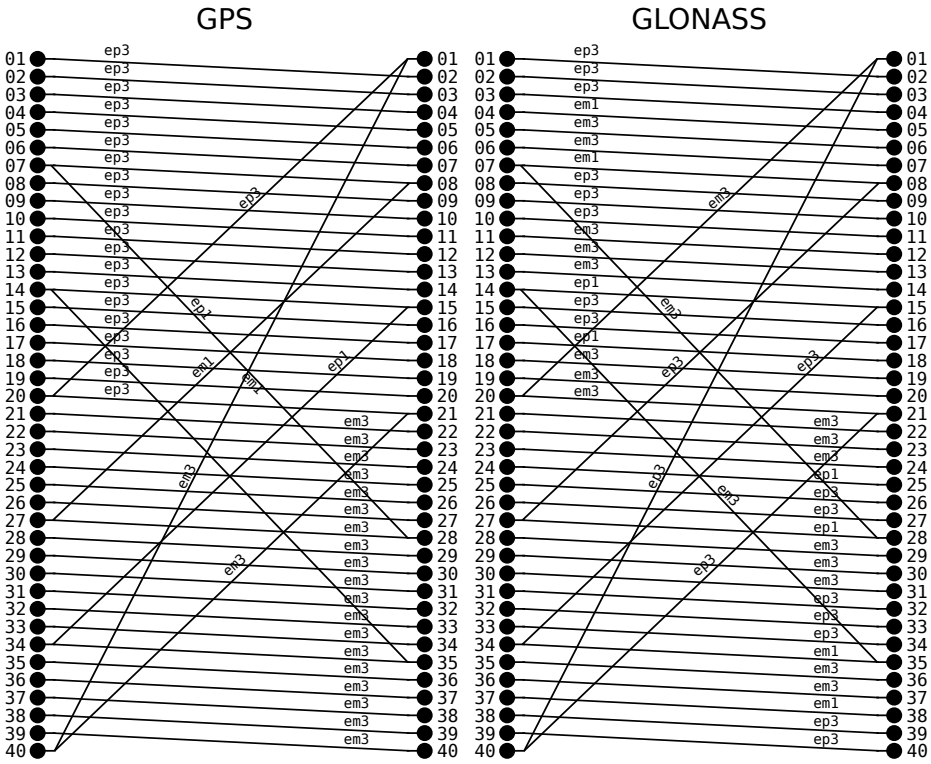

Fig. 2. Resulting trellis diagrams for both GPS and GLONASS. Notice that while the transition pattern is the same in both cases, the expected correlation results associated to the transitions (and thus the quadratic error increments) differ

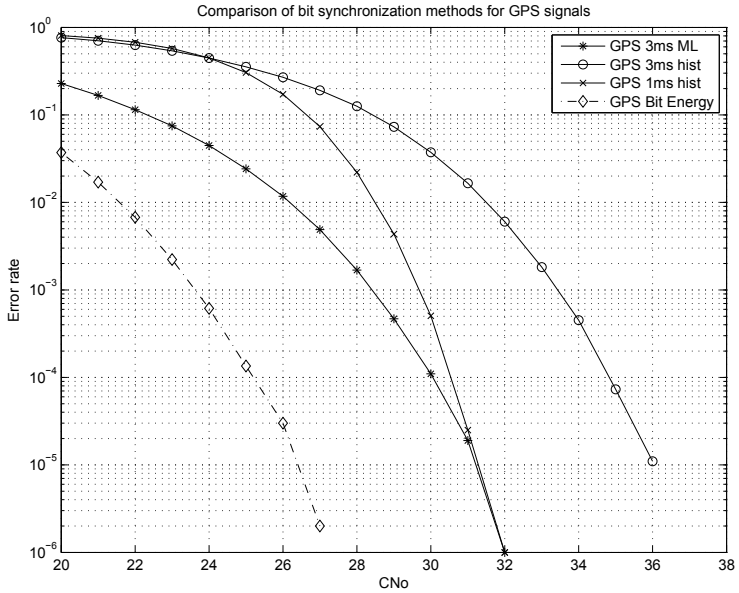

Fig. 3. Comparison of our algorithm to other three synchronization algorithms for simulated GPS signals: 1 ms histogram, 3 ms histogram and maximum bit energy

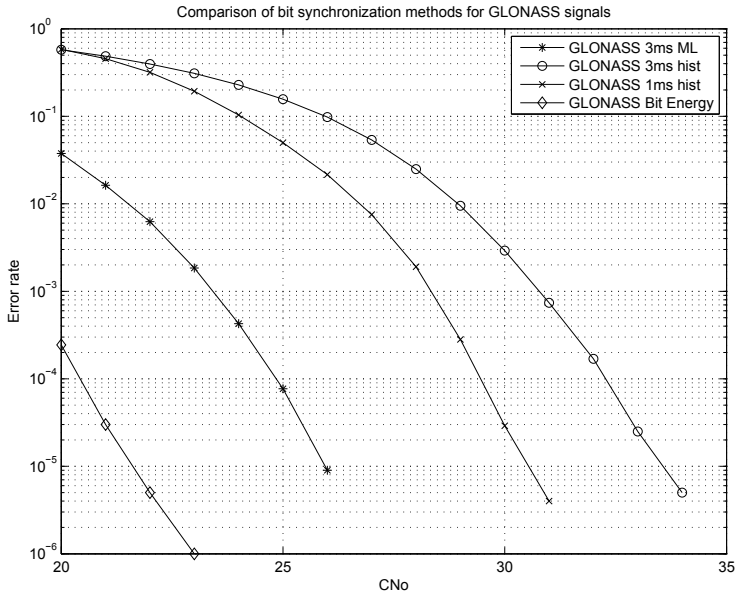

Fig. 4. Comparison of our algorithm to other three synchronization algorithms for simulated GLONASS signals: 1 ms histogram, 3 ms histogram and maximum bit energy

The only operations involved are additions and multiplications. There only four possible increments which can be pre-calculated before performing each trellis update. The amount of memory required to store the table accumulators is small by the standards of today's embedded systems.

7 Simulations

In Figs. 3 and 4 the results of simulating this algorithm can be seen for GPS and GLONASS signals. The graphs show the probability of error (misdetecting the position of the bit edges) as a function of C/No ratio. Each figure shows the error rates of four different methods: the 3 ms ML method herein proposed, 1 ms histogram, 3 ms histogram and maximum bit energy as proposed in [10].

Each point in the graph is the average error after 1 million trials. Simulated bit sequences were 50 bits long, independent and with probability of data bit change at each edge $P_b = 0.5$. For the sake of fairness in the comparison, the 1 ms histogram method was modified so as to make a hard decision after 50 bits.

The 3 ms histogram in these simulations is a simple extension of the 1 ms histogram. Based on the normalized value of a 3 ms correlation sample the algorithm decides whether there was a data bit change during the integration. If such sign change is detected, based on the sign of the previous sample and the normalized modulus of the current sample a hard decision is made to decide which of the three histogram bins covered by the sample must be incremented.

Except for the 3 ms ML algorithm proposed in this paper, none of the other methods were originally developed to work with GLONASS signals. In order to compare their performance to that of our algorithm when processing GLONASS signals these algorithms had to be modified. In the case of the bit energy signal, the meander code was removed before calculating the total bit energy for each possible displacement. Both variants of the histogram method were changed so that they would perform detection of the mid bit transition instead of the actual bit edge; this is because since that sign change happens at least once during every bit, the corresponding histogram bin is incremented twice as many times in average compared to the true bit edge.

The presence of the meander code in the GLONASS signal caused a significant improvement in the performance of all the methods that were simulated compared to the same simulations for GPS signals. However, relative performance between them remained invariant on either system. It can be seen that there is a severe degradation in the performance of the 3 ms histogram algorithm compared to the original 1 ms algorithm. This is because of the drop of the signal-to-noise ratio in the samples when a data sign inversion happens and the data bit energy is partially canceled, which is something that cannot happen with 1 ms samples. The bit energy displayed excellent performance even at really low C/No values, which makes it ideal for indoor applications and software receivers, but just like the classic histogram algorithm it works only on 1ms correlation samples and as such it requires greater processing power than other algorithms that function on multi-millisecond samples.

8 Conclusions

A new low complexity, multi-millisecond integration time maximum likelihood bit synchronization method for GLONASS and GPS was developed. Our algorithm showed improved noise resilience and lower error rate, at a complexity level not much higher than the histogram algorithms. Since it naturally works on 3 ms samples the average processor load during the synchronization is expected to drop.

While the algorithm was developed to be used on 3 ms correlation samples it can easily be extended to work with different integration time lengths, given that the method used to track carrier and code phase of the signal is able to perform reliably using those longer integration times in the presence of data bit signs inversions.

This algorithm was developed in the context of the development of a mixed system multiantenna GPS/GLONASS embedded receiver. Implementation of the algorithm is under way in order to reduce the required processor load of the system while at the same time improving performance under low C/No ratio conditions.

References

1. Kaplan, E. : Understanding GPS: Principles & Applications. Artech House, 1996.
2. Parkinson, B. W. y Spilker, J. J. Jr. (eds): Global Positioning System: Theory and Applications. Volume I. American Institute of Aeronautics and Astronautics (AIAA). Washington, 1996.
3. NAVSTAR Global Positioning System - Interface Specification IS-GPS-200. NAVSTAR GPS Joint Program Office. 7 March 2006
4. Global Navigation Satellite System - GLONASS - Interface Control Document (Edition 5.1). Russian Institute of Space Device Engineering. Moscow 2008
5. Liu, C. L. y Layland, J.: Scheduling algorithms for multiprogramming in a hard real-time environment. Journal of the ACM 20, 1973.
6. Kokkonen, M., Pietila, S. : A New Bit Synchronization Method for a GPS Receiver. In: Position Location and Navigation Symposium, 2002 IEEE , vol., no., pp. 85- 90, 2002
7. Tao Zhang; Gannan Yuan : A New Bit Synchronization Method for an Ultra-Tightly Integrated GPS Receiver. In: Information and Computing Science, 2009. ICIC '09. Second International Conference on , vol.1, no., pp.239-242, 21-22 May 2009
8. Sichao Li; Jinhai Sun; Jinhai Li; Yuepeng Yan : A modified histogram bit synchronization algorithm for GNSS receivers. In: Information Science and Engineering (ICISE), 2010 2nd International Conference on , vol., no., pp.1720-1723, 4-6 Dec. 2010
9. Anghileri, M.; Pany, T.; Jong-Hoon Won; Hein, G. W. : An Algorithm for Bit Synchronization and Signal Tracking in Software GNSS Receivers. In: Proceedings of the ION GNSS 2006 Conference, September 26-29, Fort Worth, Texas, USA
10. Ziedan, N.I., Garrison, J.L. : Bit Synchronization and Doppler Frequency Removal at Very Low Carrier to Noise Ratio Using a Combination of the Viterbi Algorithm with an Extended Kalman Filter. In: Proceedings of the 16th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS/GNSS 2003), Portland, OR, September 2003, pp. 616-627.