Real-Time Spoofing Detection in a Narrow-Band Civil GPS Receiver

Brady W. O'Hanlon, Mark L. Psiaki, *Cornell University, Ithaca, NY*
Todd E. Humphreys, Jahshan A. Bhatti, *The University of Texas at Austin, Austin, TX*

BIOGRAPHY

Brady W. O'Hanlon is a graduate student in the School of Electrical and Computer Engineering at Cornell University. He received a B.S. in Electrical and Computer Engineering from Cornell University. His interests are in the areas of GNSS technology and applications, GNSS security, and space weather.

Mark L. Psiaki is a Professor in the Sibley School of Mechanical and Aerospace Engineering. He received a B.A. in Physics and M.A. and Ph.D. degrees in Mechanical and Aerospace Engineering from Princeton University. His research interests are in the areas of estimation and filtering, spacecraft attitude and orbit determination, and GNSS technology and applications.

Todd E. Humphreys is an assistant professor in the department of Aerospace Engineering and Engineering Mechanics at the University of Texas at Austin. He received a B.S. and M.S. in Electrical and Computer Engineering from Utah State University and a Ph.D. in Aerospace Engineering from Cornell University. His research interests are in estimation and filtering, GNSS technology, GNSS-based study of the ionosphere and neutral atmosphere, and GNSS security and integrity.

Jahshan A. Bhatti is pursuing a Ph.D. in the Department of Aerospace Engineering and Engineering Mechanics at the University of Texas at Austin, where he also received his B.S. His research interests are in the development of small satellites, software-defined radio applications, and GNSS technologies.

ABSTRACT

A real-time method for detecting GPS spoofing in a narrow-bandwidth civilian GPS receiver has been implemented and tested, both in the absence of and in the presence of spoofing. The system was implemented as a software-defined radio system on a personal computer, using a pair of narrow-bandwidth radio front-ends that were geographically separated, with data transmitted between the two over the Internet.

The presence of a spoofing signal is determined by mixing and accumulating the base-band quadrature channel samples from the two receivers, with the aim of cross-correlating the P(Y) code that should be present in both signals in the absence of spoofing.

Cross-correlation of spurious signals and undesired autocorrelation of C/A codes precluded the reliable detection of a spoofing signal in the real-time system, though further post-processing in MATLAB resolved these issues.

I. INTRODUCTION

As the reliance of the civilian community on GPS signals for timing and positioning in mission-critical applications grows, so does the vulnerability to and potential cost of an attack via signal spoofing. GPS signal spoofing is a type of attack whereby a GPS receiver is fooled into tracking counterfeit signals, generally with the intention of misleading the receiver with regards to position, time, or both. In 2001 the U.S. Department of Transportation warned of the vulnerability of civilian GPS receivers to attacks such as spoofing[1], and such attacks have since been demonstrated by a variety of parties[2,3].

Given the potential damage a successful spoofing attack could inflict, detecting this kind of attack is of paramount importance. The spoofing detection method implemented here was proposed by Lo et al. [4], and is based on the presumed security of the encrypted P(Y) code. This paper seeks to examine the efficacy of this method in the context of a narrow-bandwidth real-time software receiver using only those components that would normally be used in a civilian receiver (that is, using only a patch antenna rather than a high-gain antenna and no additional timing hardware).

The Lo method assumes that a spoofer can only spoof the C/A code, and in doing so thereby changes the relationship between the C/A code and P(Y) for a particular spoofed signal. Let us assume that there exists a "reference" receiver that is trusted (that is, the signals are believed genuine), and a "user equipment" receiver which may or may not be under a spoofing attack. If one can isolate that portion of the signal that should contain the P(Y) code from the reference receiver, a cross-correlation of this data and similar data from the user equipment receiver can be carried out. Only if the user equipment receiver is not being spoofed should there be a large correlation value due to the cross-correlation of the P(Y) code from both sets of data. Properly executing this cross-correlation requires isolating the portion of the signal that should contain the P(Y) code and temporal alignment of the two data streams.

A good discussion on the probable efficacy of several other spoofing detection methods can be found in the paper by Humphreys.

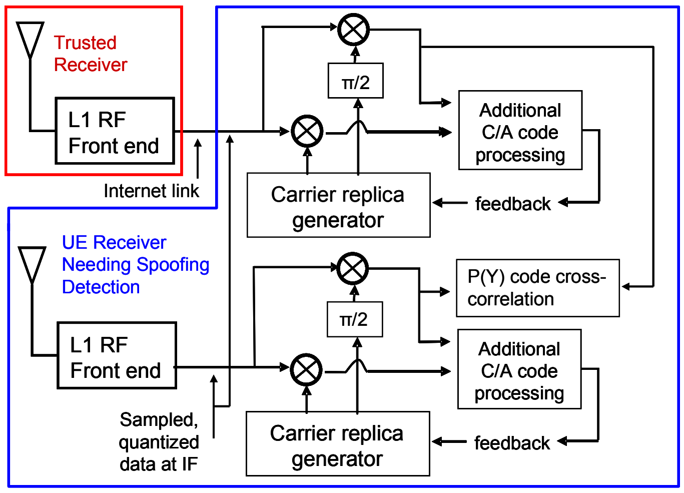
Section II of this paper contains a description of the hardware used in this work. Section III is an overview of the software used in implementing this spoofing detection method including the algorithm in general terms, derivation and analysis of the spoofing detection statistic threshold, and peculiarities that are necessary due to the particular software receiver used. Section VI contains initial results from testing the algorithm under a variety of conditions including when the UE receiver is being subjected to a spoofing attack. Section V contains a discussion of the results, including an in-depth analysis of the algorithm shortcomings and challenges (done in MATLAB), the possibility of spoofing of the reference receiver, and one possible method of C/A code interference, and Section VI contains conclusions.

II. HARDWARE OVERVIEW

Data for this experiment was collected using custom designed radio-frequency front-ends (RFEs) paired with data acquisition units. The RFEs used have intermediate frequency filters with a bandwidth of 1.9 MHz, and produce 2-bit quantized data at a sampling frequency of approximately 5.7 MHz. The quantized data is recorded to a personal computer using a data acquisition peripheral and then transmitted over the internet from the reference receiver to the user equipment receiver, where all processing is done. As the sampling rate is only 5.7 MHz and the data are sampled with 2-bit quantization, this means the data link between the reference and user equipment receivers need only support rates of 11.4 megabits per second, which is well within the capabilities of standard internet connectivity.

All processing was done on a personal computer with a quad-core Intel i7 930 CPU, and only standard hemispherical patch antennas were used at both the reference and user equipment receivers.

A block diagram of the system architecture is shown in Fig. 1.



Trusted Receiver
L1 RF Front end
Internet link
π/2
Additional C/A code processing
Carrier replica generator
feedback
UE Receiver Needing Spoofing Detection
P(Y) code cross-correlation
π/2
L1 RF Front end
Additional C/A code processing
Sampled, quantized data at IF
Carrier replica generator
feedback

Fig. 1. Spoofing detection receiver architecture.

III. SOFTWARE OVERVIEW

In this section we will examine the general theory behind the spoofing detection method implemented here, how the spoofing detection statistic threshold was calculated, and general algorithms required due to the particular software receiver that was used.

A. Spoofing detection algorithm

In this implementation of the Lo spoofing detection method, temporal alignment of the two data streams is first step taken. Rather than time-stamping the data streams using additional equipment, the embedded navigation message data is used. To do this, both data streams are tracked until the time of week (TOW) has been decoded in both. Using this information, the latency between the two streams can be determined. Whichever stream lags is then tracked while the other stream is buffered, until the receiver is processing the exact same C/A code period on both data streams. The estimated start time of the n^{th} C/A code period for the reference receiver is defined as $t_{ref}(n)$. Similarly, the estimated start time of the n^{th} C/A code period for the user equipment receiver is defined as $t_{\text{ue}}(n)$. Given that we know $t_{ref}(n)$ and $t_{ue}(n)$ from the normal, continuous tracking of the C/A signal, and given that any group delay between the C/A and P(Y) codes is determined by the transmitter and common to both the reference and UE receivers, the P(Y) code phase in the reference receiver data stream at $t_{ref}(n)$ should be the same as the P(Y) code phase in the UE receiver data stream at time $t_{ue}(n)$.

Due to low sampling rate (5.7 MHz) of the receivers used in this work as compared to the chipping rate of the P(Y) code (10.23 MHz) there is the question of sub-sample alignment as well as the coarse alignment described above. That is, to achieve a large cross-correlation value, we must temporally align the P(Y) codes to within a fraction of a chip. The P(Y) chip period of about 97 ns and the data sampling period of 175 ns means that if alignment is done only to the nearest sample it could be off by as much as 0.55 chips, leading to significant correlation loss. However, as the sampling period is not a multiple of the P(Y) code chipping period, this error will vary over the course of each accumulation, sometimes being close to zero. The computational resources that would be required to interpolate the data on a sample-by-sample basis to the estimated start times of the P(Y) code chips was deemed prohibitively expensive, so instead we have elected to simply choose the sample nearest the estimated P(Y) code chip start time, updating it every millisecond based on the estimated C/A code start time.

The GPS C/A and P(Y) codes are both transmitted on the L1 frequency, with the C/A code being transmitted ninety degrees out of phase with respect to the P(Y) code. As the P(Y) code is encrypted and generally unavailable to civilians, it is necessary to track the C/A code in such a way that the phase is known.

A phase-locked loop (PLL) is used to accurately measure the phase of the desired C/A code signal. The PLL discriminator requires mixing of the signal with both an in-phase and a quadrature carrier replica, as illustrated in Fig. 1. The PLL is formulated to steer the carrier such that the C/A code power lies entirely in the in-phase channel after carrier wipe-off. As the P(Y) code is in quadrature with the C/A code, all that is required to isolate the portion of the signal that should contain the P(Y) code is to save a replica of the data after mixing with the quadrature carrier replica. It should be noted that it is assumed that the receiver is tracking many satellites (say, ten) concurrently. Due to memory constraints, it was desirous to avoid storing a replica of the data after carrier mixing for every satellite, so instead the only the data along with a copy of the frequency and phase of the carrier replica used for mixing is retained. This is further discussed in section III.C below.

Once the portion of the signal containing P(Y) code has been isolated in both the reference receiver and the user equipment receiver, it only remains to multiply the two data streams on a sample-by-sample basis and accumulate the result. If the receiver is not being spoofed, one is essentially computing the autocorrelation of the P(Y) code as modified by the receiver front-end and with the inclusion of noise.

B. Detection Threshold Calculation and Analysis

It is necessary to analyze the cross-correlation spoofing detection statistic in order to determine how much integration time would be required in order to achieve a reasonably small probability of false alarm and, at the same time, a reasonably large probability of detecting an actual spoofing attack. This analysis is particularly important given the unusual approach used here, one which relies on a heavily filtered version of the P(Y) code that retains only the central 1.9 MHz of its 20 MHz bandwidth.

The analysis begins with mathematical models of the quadrature base-band mixed versions of the two signals for the GPS satellite in question. One is from the reference receiver, Receiver A, and the other is from the UE receiver that seeks to detect a possible spoofing attack, Receiver B. Normalized models of these two signals take the form:

$$y_{Ai} = \sqrt{\Delta t (C/N_0)_A}\, P_F(t_i) + n_{Ai} \tag{1a}$$

$$y_{Bi} = \sqrt{\Delta t (C/N_0)_B}\, P_F(t_i) + n_{Bi} \tag{1b}$$

where y_{Ai} is the quadrature base-band-mixed signal from Receiver A that is sampled at time t_i, and y_{Bi} is the quadrature base-band-mixed signal from Receiver B sampled at the same time. The sample period of the RF front-end is $\Delta t = t_{i+1} - t_i$. The quantities $(C/N_0)_A$ and $(C/N_0)_B$ are the two P(Y) code signals' received carrier-to-noise ratios in absolute Hz units. These quantities include the attenuation effects of the narrow-band RF front-end filter, about 7 dB of P(Y) code attenuation in the case of a 1.9 MHz filter. The signal $P_F(t)$ is the distorted version of the P(Y) code that comes out of the narrow-band RF front-end filter, but re-normalized to have unit power. This re-normalization is consistent with lumping all of the power loss into the received carrier-to-noise ratios $(C/N_0)_A$ and $(C/N_0)_B$.

The terms n_{Ai} and n_{Bi} are the base-band-mixed receiver noise terms, which are assumed to be Gaussian, zero-mean, uncorrelated from sample to sample, and uncorrelated between the two receivers. The normalization used to derive these equations assumes a unit variance for the Gaussian noise in each receiver's raw RF front-end samples. This normalization implies that n_{Ai} and n_{Bi} both have standard deviations equal to $1/\sqrt{2}$. The derivation of this normalized model assumes the use of a unit-amplitude sinusoidal in order to implement the quadrature base-band mixing that generates y_{Ai} and y_{Bi}.

The formulas in Eqs. (1a) and (1b) can be used to analyze the following cross-correlation spoofing detection statistic

$$\begin{aligned}\tilde{\gamma} &= \sum_{i=1}^{M} y_{Ai} y_{Bi} \\ &\cong \Delta t M \sqrt{(C/N_0)_A (C/N_0)_B} \\ &\quad + \sum_{i=1}^{M} \Big[n_{Ai} n_{Bi} + \sqrt{\Delta t (C/N_0)_A}\, P_F(t_i) n_{Bi} \\ &\quad + \sqrt{\Delta t (C/N_0)_B}\, P_F(t_i) n_{Ai} \Big]\end{aligned} \tag{2}$$

where M is the number of quadrature base-band-mixed RF samples used to compute the statistic. This number is related to the correlation statistic's accumulation interval as follows: $T_{corr} = M\Delta t$.

The mean and variance of this detection statistic are

$$\bar{\tilde{\gamma}} = E\{\tilde{\gamma}\} = \Delta t M \sqrt{(C/N_0)_A (C/N_0)_B} \tag{3a}$$

$$\sigma_{\tilde{\gamma}}^2 = E\{\tilde{\gamma}^2\} - (\bar{\tilde{\gamma}})^2 = \frac{M}{4}\left\{1 + 2\Delta t\left[(C/N_0)_A + (C/N_0)_B\right]\right\} \tag{3b}$$

These are the mean and variance of the test statistic under the assumption of no spoofing at Receiver B. Note that it is always assumed that there is no spoofing at reference Receiver A.

If Receiver B is being spoofed, then the mean and variance of the test statistic will change because the P(Y) code will disappear from Receiver B. The modified mean and variance can be computed by setting $(C/N_0)_B = 0$ in both equations. The results are $\bar{\tilde{\gamma}}_{spoofed} = 0$ and

$$\sigma_{\tilde{\gamma}spoofed}^2 = \frac{M}{4}\left\{1 + 2\Delta t (C/N_0)_A\right\} \tag{4}$$

Given this spoofed variance for $\tilde{\gamma}$, it is helpful to re-normalize the detection statistic through division by the value $\sigma_{\tilde{\gamma}spoofed}$. This re-normalized detection statistic is

$$\gamma = \frac{\tilde{\gamma}}{\sigma_{\tilde{\gamma}spoofed}} = \frac{\sum_{i=1}^{M} y_{Ai} y_{Bi}}{\sqrt{\frac{M}{4}\{1 + 2\Delta t (C/N_0)_A\}}} \tag{5}$$

This statistic has a spoofed mean of 0 and a spoofed variance of 1. Its un-spoofed mean and variance are

$$\bar{\gamma} = E\{\gamma\} = 2\Delta t \sqrt{\frac{M (C/N_0)_A (C/N_0)_B}{1 + 2\Delta t (C/N_0)_A}}$$

$$= 2\sqrt{\frac{T_{corr} \Delta t (C/N_0)_A (C/N_0)_B}{1 + 2\Delta t (C/N_0)_A}} \tag{6a}$$

$$\sigma_\gamma^2 = E\{\gamma^2\} - (\bar{\gamma})^2 = \sqrt{\frac{1 + 2\Delta t \left[(C/N_0)_A + (C/N_0)_B\right]}{1 + 2\Delta t (C/N_0)_A}} \tag{6b}$$

Suppose that the probability density function of the detection statistic γ under the null hypothesis of no spoofing is $p(\gamma|H_0)$, and suppose that its probability density function is $p(\gamma|H_1)$ under the hypothesis of spoofing. Both of these probability densities involve a sum of products of Gaussian random variables: the noise product on the third line of Eq. (2). Due to the central limit theorem, however, it is reasonable to approximate both of these distributions as being Gaussian because they are the result of summing many small random components, typically thousands to millions of them. Therefore, these two distributions can be approximated as:

$$p(\gamma | H_0) = \mathcal{N}(\gamma; \bar{\gamma}, \sigma_\gamma) = \frac{1}{\sqrt{2\pi}\sigma_\gamma} \exp\{-\frac{(\gamma - \bar{\gamma})^2}{2\sigma_\gamma^2}\} \tag{7a}$$

$$p(\gamma | H_1) = \mathcal{N}(\gamma; 0,1) = \frac{1}{\sqrt{2\pi}} \exp\{-0.5\gamma^2\} \tag{7b}$$

As is obvious from Eqs. (7a) and (7b), $\mathcal{N}(x;\mu,\sigma)$ refers to a normal distribution in x with mean μ and variance σ^2.

The probability density function in Eq. (7a) can be used to compute a spoofing detection threshold γ_{th} that has the false alarm probability α. It is computed by solving the following integral equation:

$$\alpha = \int_{-\infty}^{\gamma_{th}} p(\gamma | H_0) d\gamma = \int_{-\infty}^{\gamma_{th}} \mathcal{N}(\gamma; \bar{\gamma}, \sigma_\gamma) d\gamma \tag{8}$$

This equation can be solved using the MATLAB function norminv.m: γ_{th} = norminv($\alpha, \bar{\gamma}, \sigma_\gamma$). This detection statistic is applied as a minimum value. A spoofing attack has been detected if $\gamma < \gamma_{th}$.

Given the spoofing detection threshold γ_{th}, the probability density function in Eq. (7b) can be used to compute the probability of detection. It is

$$P_{detect} = \int_{-\infty}^{\gamma_{th}} p(\gamma | H_1) d\gamma = \int_{-\infty}^{\gamma_{th}} \mathcal{N}(\gamma; 0,1) d\gamma \tag{9}$$

This probability can be computed using the MATLAB function normcdf.m: P_{detect} = normcdf(γ_{th},0,1).

A real implementation of the spoofing test $\gamma < \gamma_{th}$ requires an ability to compute the properly normalized statistic γ and the corresponding value of γ_{th}. The necessary computations assume knowledge of the actual noise power (i.e., noise variance) in the receiver's raw RF front-end samples along with knowledge of the carrier-to-noise ratios of the two received P(Y) code signals. The noise power normally can be determined based on an understanding of the RF front-end's analog automatic gain control (AGC) unit. An alternate, perhaps superior, noise power estimate can be developed by considering the variance of the in-phase and quadrature accumulations that are produced by the C/A code tracking loops. The received carrier-to-noise ratios of the P(Y) code can be inferred from those of the tracked C/A code. This inference uses the fact that received P(Y) code carrier-to-noise ratios are typically 2 to 3 dB lower than those of the C/A code on L1 even for a wide-band RF front end. This inference also factors in the loss of P(Y) code power in the RF front-end's narrow-band filter. The needed normalization calculations are developed in the following paragraphs.

The normalization calculations start by computing the mean and variance of the squared magnitudes of each receiver's C/A-code prompt in-phase/quadrature accumulation vector:

$$\bar{z}_{c/a} = E\{I_{c/ak}^2 + Q_{c/ak}^2\} \tag{10a}$$

$$\sigma_{zc/a}^2 = E\{[I_{c/ak}^2 + Q_{c/ak}^2]^2\} - \bar{z}_{c/a}^2 \tag{10b}$$

where $I_{c/ak}$ and $Q_{c/ak}$ are, respectively, the receiver's prompt in-phase and quadrature C/A-code accumulations for the kth accumulation interval. The two expectation operations can be carried out using time averages over many accumulations. These two statistics can be used to compute the $I_{c/ak}$ and $Q_{c/ak}$ accumulations' equivalent noise-free power and their noise variance:

$$A_{IQ}^2 = \sqrt{\bar{z}_{c/a}^2 - \sigma_{zc/a}^2} \tag{11a}$$

$$\sigma_{IQ}^2 = 0.5(\bar{z}_{c/a} - \sqrt{\bar{z}_{c/a}^2 - \sigma_{zc/a}^2}) \tag{11b}$$

where A_{IQ} is the estimated magnitude of the noise-free [$I_{c/ak}$; $Q_{c/ak}$] vector and A_{IQ}^2 is its power. The accumulations' noise variance can be used to estimate the noise variance of the raw RF front-end samples:

$$\sigma_{RF}^2 = \frac{2\Delta t}{T_{accum}} \sigma_{IQ}^2 \tag{12}$$

where T_{accum} is the accumulation interval that the receiver has used to compute the $I_{c/ak}$ and $Q_{c/ak}$ accumulations. This estimate is independent of any knowledge of the receiver's AGC unit. This calculation assumes that unit-amplitude sinusoids have been used to perform the base-band mixing of the raw RF samples in preparation for calculation of the $I_{c/ak}$ and $Q_{c/ak}$ accumulations.

The C/A-code carrier-to-noise ratio in absolute Hz units is computed using the results of Eqs. (11a) and (11b):

$$(C/N_0)_{c/a} = \frac{A_{IQ}^2}{2\sigma_{IQ}^2 T_{accum}} \qquad (13)$$

The results in Eqs. (12) and (13) can be computed for Receivers A and B in order to yield the quantities σ_{RFA}, σ_{RFB}, $(C/N_0)_{c/aA}$, and $(C/N_0)_{c/aB}$. The latter two quantities can be used to estimate the P(Y) code received carrier-to-noise ratios as follows:

$$(C/N_0)_A = (C/N_0)_{c/aA} 10^{-0.25} L_F \qquad (14a)$$

$$(C/N_0)_B = (C/N_0)_{c/aB} 10^{-0.25} L_F \qquad (14b)$$

where the $10^{-0.25}$ factors implement the assumption that the unfiltered received P(Y) code has 2.5 dB less power than the received C/A code. The loss factor L_F accounts for P(Y) code power losses in the two RF front-ends' narrow-band filters. $L_F = 0.2039$ (i.e., a 6.9 dB loss) if both receivers use a 1.9 MHz wide RF front-end filter. This value has been determined by numerically integrating the usual sinc^2 power spectral density of the P(Y) code over the narrow-band filter's bandwidth. This calculation has assumed a "brick-wall" filter roll-off curve. The correct L_F value associated with a different filter bandwidth can be calculated using similar techniques.

Note that the assumption of equal filtering losses in both receivers is consistent with the assumption of this method that both receivers' RF front-ends use similar filters. This assumption is important to the validity of the model in Eqs. (1a) and (1b). Without this condition, the $P_F(t)$ function would not be identical in the two receivers' signal model equations, and the differing distorted versions of the P(Y) code might not correlate as well with each other as is assumed in the present developments.

Given the estimates of the received P(Y) code carrier-to-noise ratios from Eqs. (14a) and (14b), one can compute the expected mean and variance of the non-spoofed γ detection statistic by using Eqs. (6a) and (6b). These values, along with the desired probability of false alarm, can be used in Eq. (8) in order to compute the spoofing threshold γ_{th}. This threshold value is then compared to the normalized spoofing statistic, which is calculated as follows:

$$\gamma = \frac{\sum_{i=1}^{M} y_{rawAi} y_{rawBi}}{\sigma_{RFA}\sigma_{RFB}\sqrt{\frac{M}{4}\{1 + 2\Delta t (C/N_0)_A\}}} \qquad (15)$$

where y_{rawAi} and y_{rawBi} are the un-normalized base-band quadrature RF samples. They will have been computed by mixing the raw RF samples to base-band using a unit-amplitude version of the quadrature base-band-mixing sinusoid.

In order to better appreciate the power of this spoofing detection test, consider the following example: Suppose that the P(Y) codes have carrier-to-noise ratios of 45 dB-Hz before they pass through each receiver's 1.9-MHz-wide RF front-end filter. Then their filtered carrier-to-noise ratios are $(C/N_0)_A = (C/N_0)_B = 10^{3.809}$ Hz (i.e. 38.09 dB-Hz). Suppose that the RF sample interval is $\Delta t = 175\text{x}10^{-9}$ sec and that the correlation statistics are computed by summing over intervals of length $T_{corr} = 1.2$ sec, i.e. by summing over $M = 6{,}857{,}143$ samples. Suppose, also, that the desired probability of a false spoofing alarm is 0.13 %, i.e., $\alpha = 0.0013$. Then the computed mean and standard deviation of the detection statistic are, respectively, $\bar{\gamma} = 5.9029$ and $\sigma_\gamma = 1.0011$. The detection threshold from Eq. (8) is $\gamma_{th} = 2.8881$, and the probability of detection from Eq. (9) is $P_{detect} = 0.998062$ (99.8062 %). Thus, a 1.2-second correlation interval can produce acceptable levels of false-alarm probability and spoofing detection probability even when using the greatly attenuated and distorted version of the P(Y) code that comes out of a 1.9 MHz wide RF front-end filter.

C. Implementation-specific issues

The code for this work was based on a previously existing software GPS receiver[5] written in the C and C++ programming languages. This receiver implements several techniques with regards to carrier mixing and data storage that required the development of additional algorithms.

Phase errors due to non-continuous carrier base-band mixing phases across accumulation boundaries. Carrier replicas in this receiver are pre-computed on a grid of Doppler frequencies and with an initial phase of zero and stored in memory as a way to reduce computational load. Carrier mixing is done over a 1 millisecond period that is defined to be the sub-accumulation period. This leads to an average phase error over the sub-accumulation period that is defined as $\Delta\varphi$. After code and carrier wipe-off, the resultant (I,Q) vector is rotated by $\Delta\varphi$ prior to processing by the PLL to obtain the true phase. Consider now a single sample of the data from one receiver after carrier-wipe off as a complex sample (A+jB). As previously stated, the goal is to mix the quadrature component of the base-band mixed samples from the reference and UE receivers, but each sample after carrier wipe-off has the aforementioned phase error $\Delta\varphi$. One way to resolve this would be to rotate each sample by $\Delta\varphi$, then mix the resultant quadrature portions of the data. Let this post-rotation quadrature accumulation be denoted Q_{rot}. It can be written as follows:

$$Q_{rot} = \sum_{i=0}^{N} \mathrm{Im}[(A_{1i} + jB_{1i}) * \exp(-j\Delta\varphi_1)] * \mathrm{Im}[(A_{2i} + jB_{2i}) * \exp(-j\Delta\varphi_2)] \tag{16}$$

Where the subscript 1 or 2 denoted which receiver the sample is from, the subscript i denotes a time index, and the A and B are the data samples after carrier wipe-off with in-phase and quadrature carrier replicas, respectively. Rearranging this expression and evaluating the sum leads to

$$\begin{aligned} Q_{rot} &= Q_1Q_2\cos(\Delta\varphi_1)\cos(\Delta\varphi_2) + I_1I_2\sin(\Delta\varphi_1)\sin(\Delta\varphi_2) \\ &\quad - I_1Q_2\sin(\Delta\varphi_1)\cos(\Delta\varphi_2) - Q_1I_2\cos(\Delta\varphi_1)\sin(\Delta\varphi_2) \end{aligned} \tag{17}$$

Where Q_1Q_2 indicates the cross-correlation of the quadrature base-band mixed data from receivers 1 and 2, I_1I_2 is the in-phase base-band mixed cross-correlation, and the other terms are cross-terms. Thus the cross-correlation rotation can be done after mixing and accumulation of the two data streams rather than on a sample-by-sample basis, but at the cost of having to carry out four times the number of cross-correlations.

Bit-wise parallel algorithms. Bit-wise parallel algorithms as described in Ref. 6 were implemented as an optimization. In this bit-wise approach, the data are stored as 32-bit integers. The data are quantized to two bits, with the sign bits from one set of 32 samples stored in one integer, and the associated magnitude bits in another. The carrier replicas are similarly packed into integers with sign and magnitude being two separate words. As stated in section A above, carrier wipe-off has not yet actually been done; we have only a copy of the data and the parameters used for carrier mixing. Thus to execute a cross-correlation, me must multiply and accumulate four things: the carrier replicas from both the reference and UE receivers, and the associated data from the reference and UE receivers. To enable a look-up table implementation all of the above inputs were split into 4-bit chunks. The sign bits are all logically exclusive-or'ed together, leaving only the data magnitude and carrier replica bits from each receiver. The 4 bits chunks of each of the above elements (data$_{\text{ref}}$, data$_{\text{ue}}$, carrier$_{\text{ref}}$, carrier$_{\text{ue}}$, sign) are combined into a 20 bit word and then used as an index into a pre-computed look-up table, where the value at that index is the result of multiplying and accumulating the two base-band mixed data streams. It was determined that the largest possible accumulation value could be stored in two bytes, so the resultant table size was 2^{20} *2 bytes = 2MB.

IV. RESULTS

Several different tests were conducted using this algorithm. The first such test was a using data from two receivers located in Ithaca, New York (42.44 E, 76.48 W), spaced about 1 kilometer apart. Neither of the receivers were being spoofed. The cross-correlation statistic versus time for this test is shown in Fig. 2. In the interest of clarity only three channels are shown, though there were a total 9 satellites in view of both receivers.

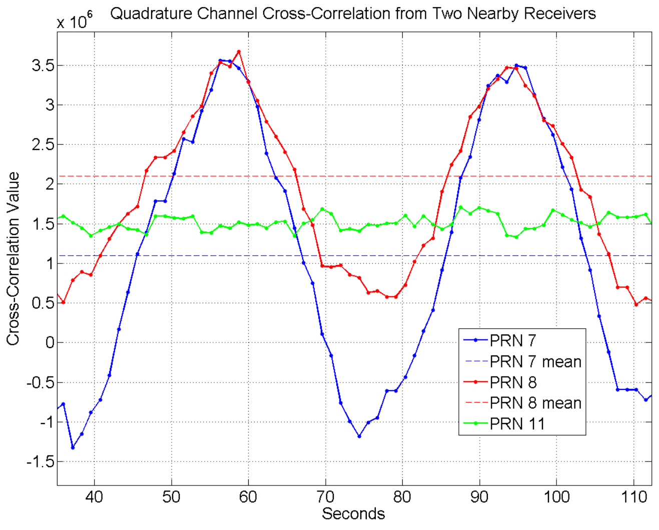


Fig. 2. Cross-correlation statistic vs. time for two nearby receivers, neither of which is being spoofed.

Rather than the expected large positive cross-correlation power, there is a large amplitude oscillation present on two of the signals, though their mean value is large and positive. Other signals from this data set also showed oscillations.

The second test used data from one receiver located in Ithaca, New York, and a second receiver located in Austin, Texas (30.33 N, 97.68 W), with neither of the receivers being spoofed. Cross-correlation statistics versus time for this test are shown in Fig. 3.

Similarly to the result from test 1, there is a large amplitude oscillation present on one of the signals, though two others show a relatively large and positive cross-correlation value over the length of the test.

The third test again used data from one receiver in Ithaca, New York, and one receiver in Austin, Texas. For this test neither receiver was spoofed for the first 65 seconds, after which time spoofing was turned on at the Austin receiver. For the next 65 seconds, although the signal was being spoofed, the spoofed signal was held as closely as possible to the true signal (that is, the spoofed signals had the same pseudorange and time as the true signal). After this point, the pseudoranges were modified to make it appear that the receiver was moving in the ECEF y direction with a velocity of 2 m/s for the remainder of the test. Cross-correlation statistic time histories for this test are shown in Fig. 4. Some of these time histories differ qualitatively from those in Fig. 3, but these results do not show a clear drop of all detection statistics nearly to zero after spoofing has commenced.

The result from test 1 was duplicated, to a large extent, using an independently developed MATLAB-based software receiver (discussed in depth in section V.A). This fact implies that Fig. 2's anomalous results are not caused solely by errors in the processing.

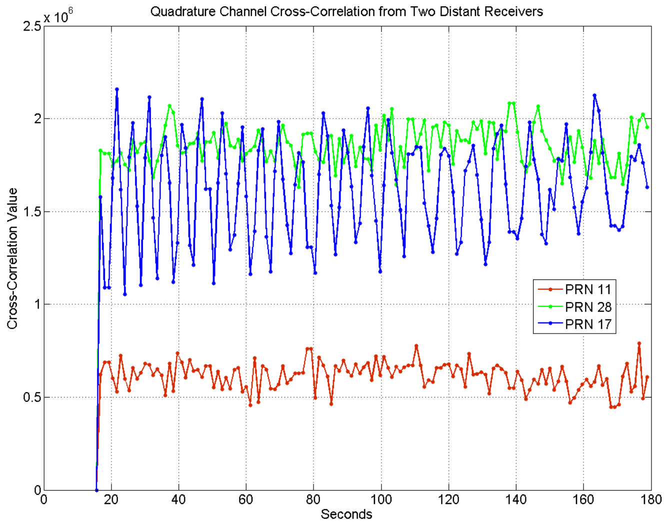


Fig. 3. Cross-correlation statistic versus time for two widely separated receivers, neither of which being spoofed.

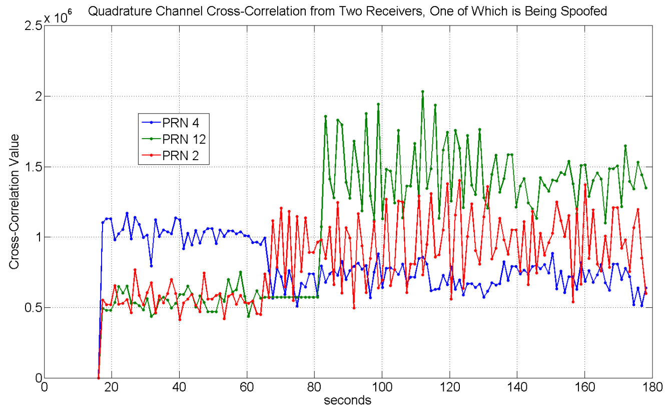


Fig. 4. Cross-correlation statistic vs. time for two widely separated receivers, one of which is spoofed starting at 65 seconds.

V. DISCUSSION

A. MATLAB Analysis

The anomalous initial results from the spoofing detection process represent a significant concern. As is reasonable for a complicated receiver software development task, a candidate explanation of these results is that the receiver software might be incorrect. Therefore, an independent MATLAB software receiver has been used to process the data from two receivers in order to check whether it yields similar or different cross correlation results. It is an off-line software receiver that works in a post-processing mode.

The MATLAB software receiver has confirmed the results of the C-code software receiver. Consider Fig. 5, which plots the cross-correlation for PRN08 for the case with the two receivers located in Ithaca. This cross-correlation time history is plotted as the green dashed curve in Fig. 5. Its values have been computed using correlation intervals of T_{corr} = 1.2 sec. These are normalized detection statistic values, γ as defined in Eq. (15), which is why their scale differs from the corresponding cross-correlation plot for the C-code receiver (red solid curve of Fig. 2), the latter being given in raw, un-normalized units. This cross-correlation curve exhibits similar oscillations to those of the C-code receiver. Unfortunately, there are unexplained differences. The MATLAB plots pass to negative values, while the C-code values do not. Given the reliability of the MATLAB software receiver, this result implies that the C-code software receiver calculations need modification.

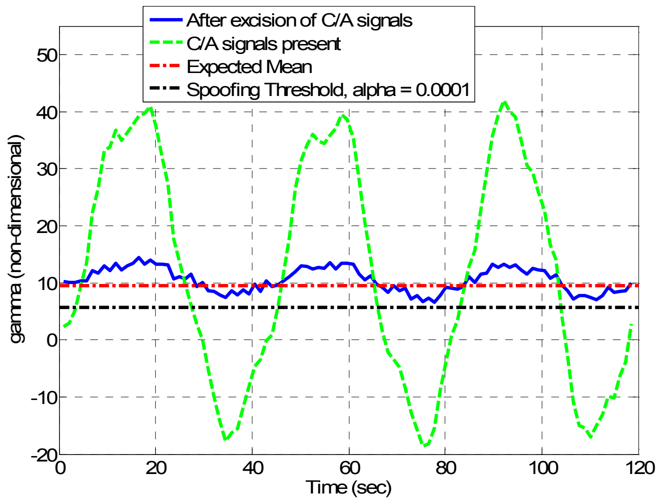


Fig. 5. The γ detection statistic with and without prior excision of the C/A code signals for two receivers near each other (PRN08, T_{corr} = 1.2 sec).

Additional calculations have been used to determine the expected mean value of this curve, $\bar{\gamma}$ from Eq. (6a), and the spoofing detection threshold for a probability of missed detection $\alpha = 10^{-4}$, γ_{th} from Eq. (8). They are also plotted on Fig. 5 as, respectively, the red dash-dotted horizontal line at $\bar{\gamma}$ = 9.42 and the black dash-dotted horizontal line at γ_{th} = 5.70. These values correspond to received P(Y) code carrier-to-noise ratios of $(C/N_0)_A = 10^{4.31}$ Hz (43.1 dB-Hz) and $(C/N_0)_B = 10^{3.71}$ Hz (37.1 dB-Hz) as inferred from the corresponding tracked C/A code carrier-to-noise ratios by using Eqs. (14a) and (14b). The small false-alarm probability still allows a large probability of detection, P_{detect} = 0.99999999397 (99.999999397 %), because of the two signals' relatively high carrier-to-noise ratios.

As is evident from the green dashed curve of Fig. 5, false spoofing alarms occur in the ranges t = 27.5 to 45.4 sec, t = 66.0 to 83.1 sec, and t above 104.0 sec. These are the times when the green dashed curve lies below the black dash-dotted horizontal line that indicates γ_{th}. Something is obviously wrong with this spoofing detection test.

One conjecture about the problem with this spoofing detection statistic is that it is affected by the other C/A codes that are present. A second C/A code could produce these results if its differential C/A code start/stop relative to that of PRN08 were the same for both receivers and if its differential carrier Doppler shift relative to PRN08 were also the same in the two receivers, as will be discussed in further detail below. In such a situation, the second signal's C/A code would look almost identical between the two receivers after base-band mixing in quadrature with the C/A code for PRN08. The second C/A code would likely not be at base band itself after this operation, but its time variations would be nearly identical in the two receivers. When cross-correlated between the two receivers, these nearly identical time variations would produce significant power.

This conjecture has been tested by re-computing the inter-receiver quadrature cross-correlation for PRN08 using a modified algorithm. The modified algorithm first removes all C/A code signals from the raw RF samples of both receivers before mixing to base-band in quadrature with the C/A code of PRN08. Removal of the C/A code signals is a straight-forward, though computationally intensive, task after their carrier and code phases have been successfully acquired and tracked using a PLL and a DLL, as was done in Ref. 7. Given these modified base-band-mixed quadrature signals, the remaining calculations of the modified cross-correlation algorithm are identical to those that culminate in Eq. (15). The result for PRN08 is the blue solid curve in Fig. 5.

The modified cross-correlation curve in Fig. 5 has about the same mean value as the green dashed curve, but its oscillations are greatly reduced. This reduction provides a clear indication that the other C/A code signals played a significant role in producing the large oscillations of the green dashed spoofing detection statistic. Their removal produces a much more reasonable curve, one whose mean value is relatively close to the expected mean shown in the red dash-dotted flat line and whose variations never produce a spoofing false alarm: Note how the blue solid curve never drops below the black dash-dotted spoofing alarm threshold. The spoofing-detection cross-correlations of additional signals have been re-calculated using C/A-code excision, and similar improvements have been observed.

The results shown in Fig. 5's blue solid curve provide strong evidence that the filtered P(Y) code is present in the signal in sufficient strength to be used for spoofing detection. Otherwise, the mean value of the solid blue curve would not have been near its expected mean value, the level of the red dash-dotted line. The residual oscillations of the blue solid curve, however, are still considered to be anomalous, i.e., to differ from what is expected based on a simple analysis of spoofing detection. It is believed that these oscillations are caused by the filtered P(Y) codes of the other signals. These filtered P(Y) codes have a correlation length of 150 m due to their 1.9 MHz filtered bandwidths. Therefore, it is believable that the other signals' filtered P(Y) codes could have had similar offsets relative to the filtered P(Y) code of PRN08 in the two receivers, similar at the 150 m level. Variations of their offsets relative to the PRN08 filtered P(Y) code could cause the observed oscillations in the blue solid curve.

An additional analysis has been performed in order to assess whether the encouraging results of the blue solid curve of Fig. 5 are caused by the presence of filtered P(Y) code. This test computes the correlations of the base-band-mixed PRN08 quadrature signals for various time offsets of the two receivers' signals relative to each other, as defined by first lining up their corresponding PRN08 C/A codes. The resulting cross-correlation vs. delay plot is the blue solid curve in Fig. 6. It applies to the 2nd 1.2-sec cross-correlation interval associated with Fig. 5. This curve should have a shape that is consistent with the autocorrelation function of the filtered P(Y) code. For comparison purposes, that shape is also plotted in Fig. 6 as the red dash-dotted curve. This latter curve assumes a "brick-wall" filter in the RF front-end with a bandwidth of 1.98 MHz. This bandwidth has been "tweaked" up from the advertised 1.9 MHz 1dB bandwidth of the RF front-end in order to better match the blue solid curve. As can be seen, these two curves match relatively well. This close match further supports the conjecture that the good cross-correlation results of the blue solid curve in Fig. 5 are caused by filtered P(Y) code on the PRN08 quadrature signal.

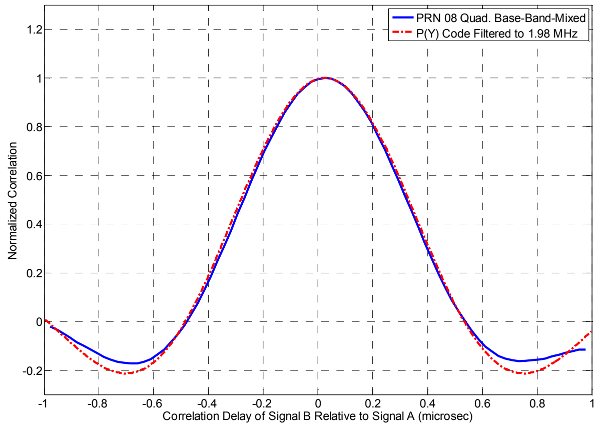


Fig. 6. Filtered P(Y)-code cross-correlation as a function of delay between Signals A and B, experimental result for PRN08 with nearby receivers (blue solid) and theoretical result based on 1.98 MHz RF front-end filter (red dash-dotted).

Note that some of the corresponding correlation curves at different time offsets show multiple correlation peaks of similar magnitude. These alternate correlation curves are the equivalents of the blue solid curve of Fig. 6, but taken from correlation time intervals other than the 2nd interval associated with Fig. 5. The presence of multiple peaks provides an indication that P(Y) codes of other satellites may be playing a role in the generation of the blue solid curve in Fig. 5. They may constitute the cause of that curve's oscillations. Of course, with distant receivers, such effects would likely diminish to below the noise level.

Three additional MATLAB analyses are planned. The first analysis will apply the C/A-code excision technique to data from distant receivers, one in Ithaca, NY and one in Austin, TX. Spoofed and unspoofed cases will be considered. The analysis goal will be to determine whether the presence of other C/A codes caused any of the problems that have been noted for the C-code spoofing detection algorithm when applied to distant receivers.

The second analysis will attempt to detect the filtered P(Y) code in each signal by using semi-codeless techniques, as discussed in Refs. 8, 9, and 10. These techniques will need to be modified in order to account for the effects of the narrow-band filter on the P(Y) code. The principle effect is to broaden the square pulse function of a chip of the P(Y) code into a sinc-like function. The goal of this analysis will be to verify the presence of the filtered P(Y) code in the RF front-end's output and to check its characteristics.

The third analysis will study the effects of applying semi-codeless P(Y) techniques to the problem of spoofing detection. This will be a generalization of the semi-codeless techniques of Refs. 8, 9, and 10 to the spoofing detection cross-correlation calculation. It is hoped that such a method will eliminate the oscillations that appear in the blue solid curve of Fig. 5. This is expected to happen because the semi-codeless technique makes use of the known P code and the known timing, relative to the P code, of the W anti-spoofing bits that transform P code into P(Y) code [8,9,10]. The low cross-correlations of the P codes of different PRN numbers, if suitably exploited by a semi-codeless technique, are expected to eliminate extraneous signal correlations between the two receivers. This holds true even when the reference and UE receivers are close to each other. There is an additional expected benefit of using a semi-codeless spoofing detection technique. A good technique should significantly reduce the required cross-correlation interval T_{corr} for a given false alarm probability α and probability of detection P_{detect}.

B. Reference Station Spoofing

One might suspect that spoofing of the reference receiver would not be a problem for the proposed architecture. The reference receiver knows its location, and therefore, it might be able to use this knowledge in order to detect a spoofing attack. In fact, a spoofer could attack the reference receiver using the method of Ref. 2 in a way that does not try to spoof its position or its receiver clock time. Rather, this "auxiliary" spoofer would have another, more subtle goal in its spoofing attack. It would seek to spoof the reference receiver about what is the proper P(Y) code signal that is in phase quadrature with each received C/A code signal. Given such spoofing, the reference receiver would not detect any error in its position or even in its receiver clock. It would, however, transmit an erroneous base-band-mixed quadrature signal to the UE receivers that it was supposed to aid in detecting spoofing. If another spoofer, the main spoofer, then attacked the UE receivers using the same false P(Y) code in phase quadrature with each spoofed C/A code, then such an attack would defeat the present method and, indeed, the method of Ref. 4.

C. C/A Code Interference

In this section we examine one of the possible mechanisms by which the C/A code could contribute to the quadrature channel cross-correlation result, a mechanism that has already been suggested in the MATLAB analysis section.

To illustrate this mechanism, imagine that both the reference receiver and the UE receiver are only observing two GPS satellites, which we shall denote SV_A and SV_B. As discussed in section III, cross-correlation of the P(Y) code from $\mathrm{SV_A}$ involves mixing the data from the reference and UE receivers to baseband using the estimated phase, Doppler shift, and intermediate frequency of the signal. This will produce signals of the form

$$S_{REF} = P(Y)_A + \cos[(\omega_{A-REF} - \omega_{B-REF})\ t + \alpha]CA_B \quad (18a)$$

$$S_{UE} = P(Y)_A + \cos[(\omega_{A-UE} - \omega_{B-UE})\ t + \beta]CA_B \quad (18b)$$

where S_{REF} and S_{UE} are the base-band mixed signals from the reference and UE receivers, respectively. The ω terms here are the Doppler shift of the signal from either SV_A or SV_B as observed at either the reference of UE receivers, α and β are arbitrary phases, $P(Y)_A$ is the P(Y) code from SV_A, and CA_B is the coarse/acquisition code from SV_B We are neglecting noise, data bit modulation, and signal amplitude terms here.

Once these two signals are mixed to base-band, we multiply them together and accumulate, noting that the result contains the following term:

$$\sum_{i=1}^{N}\cos[\{(\omega_{A-REF} - \omega_{B-REF}) - (\omega_{A-UE} - \omega_{B-UE})\}t_i + \alpha + \beta]CA_B(i) * CA_B(i) \quad (19)$$

Where the sum is over N samples, i is a sample index, and other terms remain as before. Several cross-terms have been omitted here. We see that if the observed difference in Doppler shift between SV_A and SV_B is close to the same for both the reference and UE receiver, this term will possibly have a large magnitude due to the auto-correlation of the C/A code from SV_B, and will be amplitude modulated as determined by this Doppler shift double difference. Of course it is also required that the C/A code from SV_B have the same relationship in terms of code phase to the C/A code from SV_A on both receivers (i.e., the pseudorange double difference must also be small modulo one C/A code period), otherwise the auto-correlation properties of the C/A codes means there will not be a large correlation value.

Referring again to Fig. 2, it should be noted that the observed difference in Doppler shift between PRN 7 and PRN 8 was nearly identical at both the reference and UE receivers, presumably leading to the observed large-amplitude oscillation. PRN 11 had no such small Doppler double difference, thus the lack of a large oscillation. This result is more expected when the reference receiver and the UE receiver are very close together, but is simply a function of the geometry of the system and can happen with widely spaced receivers.

VI. CONCLUSIONS

In summary, an attempt has been made to apply a recently proposed spoofing detection method within a software receiver. This method seeks to verify the absence of spoofing by looking for strong cross correlations between two receivers, one a reference receiver and the other the potential spoofing victim, of the portion of the P(Y) code that passes through each receiver's narrow-band RF front-end. This heavily filtered P(Y) code should be present in phase quadrature with the C/A codes of both receivers if neither is being spoofed. Lack of a strong cross-correlation should indicate a spoofing attack.

The real-time version of this system performs poorly at present. In at least one case, that of nearby receivers, an offline software receiver analysis has determined the cause of much of this poor performance: It results from additional cross-correlation power that arises from other C/A codes. This analysis, however, also confirms that the narrow-band-filtered remnants of the P(Y) are present in sufficient strength to develop a reasonable spoofing detection statistic. One strategy for synthesizing a successful test statistic is to excise the C/A codes from the signals before cross-correlation between receivers. Another possible technique might resort to semi-codeless cross correlation, though the practicality and efficacy of such an approach has yet to be demonstrated.

This paper's negative result for the original simple-minded version of this spoofing detector is at odds with another published work on this subject. It is possible that this discrepancy has been caused by the present algorithm's use of a narrow-band RF front-end. The other work used a wide-band front-end, one that captures most of the P(Y) power. It is possible, however, that alternate, stronger explanations for this discrepancy may be found.

The only certainty at present is that significant further study is required of this concept and of the challenges of implementing it successfully. Its ability to tolerate a narrow-band filter in the RF front-end may make this type of spoofing detection system practical for low-cost, low-power receivers, but this approach may be difficult to implement because of the significant attenuation and distortion of the encrypted P(Y) code signal that constitutes its basis for spoofing detection.

REFERENCES

[1] "Vulnerability assessment of the transportation infrastructure relying on the Global Positioning System," Tech. rep., John A. Volpe National Transportation Systems Center, 2001.

[2] Humphreys, T.E., Ledvina, B.M., Psiaki, M.L., O'Hanlon, B., and Kintner, P.M. Jr., "Assessing the Spoofing Threat: Development of a Portable GPS Civilian Spoofer," *Proceedings of the ION GNSS 2008*, Sept. 16-19, 2008, Savannah, GA, pp. 2314-2325.

[3] Warner, J.S. and Johnston, R.G., "A Simple Demonstration That the Global Positioning System (GPS) is Vulnerable to Spoofing," Journal of Security Administration, 2003.

[4] Lo, S., De Lorenzo, D., Enge, P., Akos, D., and Bradley, P., "Signal Authentication, A Secure Civil GNSS for Today," *Inside GNSS*, Vol. 4, No. 5, Sept./Oct. 2009, pp. 30-39.

[5] Humphreys, T.E., Psiaki, M.L., Kintner, P.M. Jr., Ledvina, B.M., "GNSS Receiver Implementation on a DSP: Status, Challenges, and Prospects," Proc. 2006 ION GNSS Conf., Institute of Navigation, Fort Worth TX, pp. 237002382.

[6] Ledvina, B.M., Psiaki, M.L., Powell, S.P., and Kintner, P.M. Jr., "Bit-Wise Parallel Algorithms for Efficient Software Correlation Applied to a GPS Software Receiver," IEEE Transactions on Wireless Communications, Vol.3, No.5, September 2004.

[7] Psiaki, M.L., Humphreys, T.E., Mohiuddin, S., Powell, S.P., Cerruti, A.P., and Kintner, P.M. Jr., "Searching for Galileo," *Proceedings of the ION GNSS 2006*, Sept. 26-29, 2006, Fort Worth, TX, pp. 1567-1575.

[8] Woo, K.T., "Optimum Semicodeless Carrier-Phase Tracking of L2," *Navigation*, 47(2), 2000, pp. 82-99.

[9] Psiaki, M.L., Powell, S.P., Jung, H., and Kintner, P.M., "Design and Practical Implementation of Multifrequency RF Front Ends Using Direct RF Sampling," *Proc. ION GPS/GNSS 2003*, Sept. 9-12, 2003, Portland, OR, pp. 90-102.

[10] Jung, H., Psiaki, M.L., and Powell, S.P., "Kalman-Filter-Based Semi-Codeless Tracking of Weak Dual-Frequency GPS Signals," *Proceedings of the ION GPS 2003*, Sept. 9-12, 2003, Portland, OR.