Interference Mitigation in GNSS Receivers by Array Signal Processing: A Software Radio Approach

Javier Arribas, Pau Closas, and Carles Fernández-Prades
Centre Tecnològic de Telecomunicacions de Catalunya (CTTC)
Carl Friedrich Gauss, 7, 08860 Castelldefels, Spain
Email: {jarribas, pclosas, cfernandez}@cttc.cat

Abstract—The fact that most of high-accuracy positioning and distributed timing services, including safety-critical operations, rely on Global Navigation Satellite Systems (GNSS) has raised the concern of possible denial-of-service situations. Complementary to time- and frequency-domain mitigation techniques, it is well known that antenna-array based receivers can benefit from spatial domain processing and effectively mitigate unintentional and intentional Radio Frequency Interferences (RFIs). In this work, we propose a software-based GNSS receiver architecture that implements a real-time, array-based signal acquisition algorithm based on the Generalized Likelihood Ratio Test, combined with a null-steering spatial filter for signal tracking, showing its RFI rejection capabilities.

I. INTRODUCTION

The use of Global Navigation Satellite System (GNSS) technology in safety- and mission-critical services has raised the concern in recent times about possible GNSS Denial of Service (DoS) situations [1]. Examples could be the power distribution grid, synchronized using the Global Positioning System (GPS) signals, or the integration of GNSS into civil aviation, which demands specific quality of service to be assured in a variety of conditions, including harsh environments [2]. The GNSS vulnerability to interferences could become a real threat to the entire service integrity.

From the variety of solutions available to the interference problem, in this paper we explore those based on equipping the GNSS receiver with an array of antenna elements. Digital array signal processing is a powerful and attractive approach to exploit spatial diversity, as well as to counteract the aforementioned threats. We propose a software receiver architecture and array-based algorithms that operate in acquisition and tracking modes. We detail the algorithms and provide relevant references in Section III. The solution has been implemented in a custom, array-based GNSS prototype, whose architecture is sketched in Section IV. The prototype features real-time operation, including the connection of the antenna array and radio-frequency front-end to a GNSS software receiver.

II. ARRAY SIGNAL MODEL

We consider the discrete baseband signal received from a single GNSS satellite with an unstructured $N$-element antenna array:

$$X(t) = \mathbf{h}d(t, f_d, \tau) + \mathbf{N}(t),$$

where $X(t) = [x(t) (K - 1)T_s \ldots x(t)] \in \mathbb{C}^{N \times K}$ is the space-time data matrix, with $x(t) = [x_1(t) \ldots x_N(t)]^T$ defined as the array baseband snapshot, each row corresponding to one antenna, and $K$ is the number of captured snapshots. The acquisition time window duration is defined as $T_{acq} = KT_s$ where $F_s = 1/T_s$ is the sampling frequency. Vector $\mathbf{h} = [h_1 \ldots h_N]^T \in \mathbb{C}^{N \times 1}$ is a non-structured vector which models the signal power and both the channel and the array response. It assumes the role of the spatial signature, that is considered constant during $T_{acq}$. This channel vector is not only parametrized by the signal DOA and the antenna locations, but also includes other unmodeled phenomena like the front-end amplitude/phase responses and the electromagnetic mutual coupling between array elements and front-end channels. $d(t, f_d, \tau) = [s(t - (K - 1)T_s - \tau) e^{j(2\pi f_d(t - (K - 1)T_s))} \ldots s(t - \tau) e^{j(2\pi f_d t)}] \in \mathbb{C}^{1 \times K}$ is a GNSS baseband Direct Sequence Spread Spectrum (DSSS) signal with normalized power and known structure $s(t)$, received with a propagation delay $\tau$ and a Doppler frequency $f_d$, considered constant during $T_{acq}$. The general DSSS signal structure can be written as $s(t) = \sum_{k=-\infty}^{\infty} c_k g_k(t - kT_c)$, where $c_k$ is the spreading code sequence, $g_k$ is the receiving pulse, and $T_c$ is the code chip period of a Pseudorandom Noise (PRN) sequence with period $T_{code}$. This signal model can be particularized to describe a Galileo E1 signal [3], among other GNSS signals. Finally, $\mathbf{N}(t) = [n(t) (K - 1)T_s \ldots n(t)] \in \mathbb{C}^{N \times K}$ is a complex, circularly symmetric Gaussian vector process with a zero-mean, temporally white and spatially colored, modeled with an arbitrary (also unknown) spatial covariance matrix $\mathbf{Q} \in \mathbb{C}^{N \times N}$ defined as $E\{n(t) n^H(t')\} = \sigma^2 \mathbf{I} + \mathbf{\Sigma}$, where $\mathbf{E}\{\cdot\}$ stands for the expectation operator. The noise is assumed with double-sided spectral density $\sigma^2 = \frac{N_0}{2}$ W/Hz, $\mathbf{I}$ is the identity matrix, $\mathbf{\Sigma}$ models the covariance matrix of interferences, and $\delta_{\tau_m, \tau_n}$ stands for the Kronecker delta.

III. ALGORITHMS FOR INTERFERENCE MITIGATION

In this work, we address the GNSS interference problem at two stages of the receiver operation. Namely in the acquisition and tracking phases. In both cases we exploit the spatial diversity provided by array-based receivers. Next subsections are devoted to briefly show the formulation of both algorithms.
A. Interference mitigation in acquisition

We protect the signal acquisition stage by using an algorithm based on the Generalized Likelihood Ratio Test (GLRT) detector operating directly on the array snapshots. The selected algorithm outperforms conventional single-antenna acquisition algorithms, specially in the presence of uncorrelated wideband interferences, even if the array is moderately uncalibrated. In general, it is recognized that the signal acquisition has the lowest sensitivity of the whole receiver operation and, consequently, it could become the performance bottleneck. Therefore, it is important to address interference mitigation at this stage before being able to operate on the tracking mode.

The GLRT algorithm takes into account the Probability Density Function of $X(t)$ to obtain a test statistic which is able to detect GNSS signals by maximizing the probability of detection subject to a given false alarm probability [4]. The application of GLRT for GNSS array-based receivers was derived and analyzed by the authors in [5], for the sake of simplicity we show here the final test statistics expression:

$$ T_{GL}(X) = \max_{f_d, \tau} \left\{ \hat{r}_{xx}(f_d, \tau) \hat{R}_{dd}^{H} \hat{r}_{xd}(f_d, \tau) \right\} \geq \gamma, \quad (2) $$

where $\hat{R}_{xx} = \frac{1}{N} X X^H$ is an estimation of the spatial autocorrelation matrix of the array snapshots, $\hat{r}_{xd} = \frac{1}{N} X d^H$ is an estimation of the cross-correlation vector between the received array snapshots and a local satellite signal replica, $\hat{R}_{dd} = \frac{1}{N} d d^H$ is an estimation of the satellite signal autocorrelation, and $\gamma$ is the detection threshold. We consider that $d$ has normalized power ($\hat{R}_{dd} \simeq 1$). Equation (2) can be solved by an exhaustive grid search in the entire $(f_d, \tau)$ parameter space. We will see in Section IV-B how to accelerate the code delay search.

B. Interference mitigation in tracking

The tracking stage is protected using a blind nullsteering beamformer, also known as the power minimization beamformer in the GNSS literature. The nullsteering beamforming algorithm has been widely implemented in hardware-based receivers, see, e.g. [6] among others.

If we assume that the array attitude is not available and there is some amplitude and/or phase differences between array channels responses, which is a realistic assumption (see, e.g. [7]), it is not suitable to use a direction-of-arrival dependent beamformer. However, it is possible to filter out the interferences using a blind null-steering beamformer. This approach presupposes that the GNSS signals are well below the noise floor, and thus their contribution to the array covariance matrix is negligible. The algorithm minimizes the beamformer output power by nulling the directions where interfering signals are present. This beamformer is a particular case of the Minimum Variance Distortion-less Response (MVDR) beamformer, where the steering vector is defined as $h_0 = h_{ref} = [1 \ldots 0]^T \in \mathbb{C}^{N \times 1}$. The beamweights expression can be written as:

$$ \mathbf{w} = \frac{\hat{R}_{xx}^{-1} h_0}{h_0^H \hat{R}_{xx}^{-1} h_0}. \quad (3) $$

where the snapshots autocorrelation matrix estimation was already defined. Finally, the beamformer output can be obtained as:

$$ y = \mathbf{w}^H \mathbf{X}, \quad (4) $$

where $y$ is the spatial-filtered signal output feed to the tracking loops.

IV. SDR REAL-TIME IMPLEMENTATION

The work towards the completion of a flexible GNSS array platform was started by the authors in 2009, presenting a preliminary hardware-based implementation of the GLRT acquisition algorithm using a Field Programmable Gate Array (FPGA) device [8]. The prototype was evolved centering the efforts in the design of an eight channel phase-coherent front-end tuned in the GPS L1 / Galileo E1 band plus an upgrade of the FPGA platform [1]. Fig 1 shows a picture of the antenna array prototype. In parallel, in 2011, authors presented in [9] an open source GNSS software receiver, written in C++, that is able to work either from raw signal samples stored in a file, or in real-time with a radio-frequency front-end as signal source.

GNSS-SDR is an application that defines a software receiver framework and provides working implementations of different algorithms, starting from raw sample reading and conditioning and passing through concurrent blocks performing acquisition, code and phase tracking, lock detectors, demodulation of the navigation message, observable computation and Position-Velocity-Time (PVT) solution. Users can populate the framework with their own algorithms, allowing to put the focus on the signal processing implementation without worrying about how to embed that algorithm in a whole GNSS receiver. Its modularity allows fair performance benchmarks using real GNSS signals, and its open source license allows free downloading, use and code inspection. As a result of such modularity, it is easy to add access to a wide range of radio-frequency front-ends for its use in real-time mode.

This paper presents an array-based GNSS receiver that exploits both the interference mitigation capability of the antenna arrays and the flexibility of a complete Software Defined Radio implementation.
the detection of buffer overflows at the host computer.

the last byte of the payload is a frame counter (id) to allow

32

imaginary part (thus forming a
signed little-endian integer format both for the real and for the
contain the snapshots using a standard two-complement
means of a dedicated GbE bus. A special type of packet
in real–time to the host computer running GNSS-SDR by
tracking beamforming module and to each of the acquisition
modules of the satellite channels. The beamforming is a
null-steering implementation shared by all the satellite
modules of the satellite channels. The beamforming is a
critical operation in the real-time null-steering beam-
currently implemented using GNU Radio’s Vector-Optimized Library
of Kernels (VOLK) [12], composed of a set of pre-defined,
high-efficient implementations of common vector operations.

A. GNSS-SDR array receiver architecture

Fig. 2 shows the GNSS-SDR high level block diagram of the
acquisition and tracking modules adapted to the array
receiver architecture. From left to right, the 8-elements antenna
array front-end downconverts the satellite signals and performs
the analog-to-digital conversion of the baseband snapshots
signal. At this point, and FPGA generates in real-time custom
Gigabit Ethernet (GbE) frames an efficient transport. The GbE
bus is connected to a commodity PC running GNSS-SDR.
There, the snapshot stream is connected both to the global tracking
beamforming module and to each of the acquisition
modules of the satellite channels. The beamforming is a
single nullsteering implementation shared by all the satellite
channels, thus alleviating the computational load. On the
other hand, the acquisition module, particular to each satellite
channel, is connected directly to the array snapshots stream.
The sample covariance matrix estimation is shared between
the beamformer and all the satellite acquisitions by using a
thread-safe concurrent-access structure.

The eight-channel, complex baseband digital signal is sent
in real–time to the host computer running GNSS-SDR by
means of a dedicated GbE bus. A special type of packet
was designed to minimize the overhead imposed by network
protocols, and only the Medium Access Control (MAC) IEEE
802.3 protocol was used.

Table I shows the complete GbE frame, including both the
MAC fields (shadowed) and the payload custom fields. From
left to right, the first byte of the payload indicates the number
of available array channels (N), next two bytes indicate the
number of snapshots contained in the frame (K), next bytes
contain the snapshots using a standard two-complement 16 bits
signed little-endian integer format both for the real and for the
imaginary part (thus forming a 32 bits complex number), and
the last byte of the payload is a frame counter (id) to allow the
detection of buffer overflows at the host computer.

B. FFT-based array-based GLRT acquisition implementation

In order to alleviate the computational load required by the
acquisition search grid, we adapted the well-known Fast
Fourier Transform (FFT) acquisition algorithm [10] to the
array-based GLRT algorithm. The circular cross-correlation
properties and the FFT operator can be exploited to compute
cross-correlation vector \( \hat{R}_{xd}(f_d, \tau) \). The target is to parallelize the
code delay search for an specific Doppler frequency bin
\( f_d \) by defining the following cross-correlation matrix:

\[
\hat{R}_{xd}(f_d) = \frac{1}{K} \begin{bmatrix}
\mathcal{F}^{-1}(\mathcal{F}(x_1 e^{-j2\pi f_d t}) \circ \mathcal{F}(d)^*) \\
\mathcal{F}^{-1}(\mathcal{F}(x_2 e^{-j2\pi f_d t}) \circ \mathcal{F}(d)^*) \\
\vdots \\
\mathcal{F}^{-1}(\mathcal{F}(x_N e^{-j2\pi f_d t}) \circ \mathcal{F}(d)^*)
\end{bmatrix},
\]

where \( \hat{R}_{xd}(f_d) \in \mathbb{C}^{N \times K} \) is the parallel code delay correlation
for the \( f_d \) Doppler, \( \mathcal{F}(\cdot) \) stands for the direct FFT operation
and \( \mathcal{F}^{-1}(\cdot) \) stands for the inverse FFT operation, \( t \) is the
discrete time vector required to generate the carrier wipe-off
signal and \( x_i \in \mathbb{C}^1 \times k \) is the \( i \)-th antenna element input vector.
The element-wise product (Hadamard product) is represented
by the \( \circ \) operator. Finally, the parallel code delay GLRT test
statistic can be written as:

\[
T_{GL}(f_d) = \max_{f_d} \left\{ \text{diag} \left( \hat{R}_{xd}(f_d) \hat{R}_{dd}^{-1} \hat{R}_{xx}^{-1} \hat{R}_{xd}(f_d) \right) \right\}.
\]

The global test statistic maximum in the entire code delay and
Doppler search space can be obtained as:

\[
T_{GL}(f_d) = \max_{f_d} \left\{ \max \left( T_{GL}(f_d) \right) \right\}.
\]

Direct and inverse FFT operations in Eq. 5 were implemented
using the open-source FFTW library [11]. FFTW computes the discrete
Fourier transform in one or more dimensions, of arbitrary input size using Single Instruction Multiple
Data (SIMD) processor instructions, thus, reducing the execu-
tion time. In addition, element-wise vector products were
implemented using GNU Radio’s Vector-Optimized Library
of Kernels (VOLK) [12], composed of a set of pre-defined,
high-efficient implementations of common vector operations.

C. Null-steering beamforming weight update

A critical operation in the real-time null-steering beam-
former is the antenna elements weight update. GNSS-SDR
has two parallel threads to perform the operation: on the one
hand, there is a sample covariance matrix estimation block
implementation that takes a set of snapshots from the array,
computes the autocorrelation using the VOLK library, and
executes the weight computation algorithm described in Eq.
3. On the other hand, the beamformer thread produces the
spatial-filtered output described in Eq.4. Both threads share the
weights memory locations using a thread-safe map structure.
Weights are updated at a lower rate compared to the sampling
frequency.
V. EXPERIMENTAL RESULTS

The performance of the complete array receiver was tested in harsh interference environment conditions. We defined an open-sky scenario where a strong uncorrelated in-band jammer impinged into the array with different DOA. The interference was transmitted using an auxiliary directive antenna with an approximated DOA of $\theta = 45^\circ$ and $\psi = 45^\circ$, which emulated a moderate elevation jammer or a signal coming from nearby communication tower. The satellite signal power and the interference (or jammer) power are given in terms of Carrier-to-Noise density ratio ($C/N_0$) and Interference-to-Noise density ratio ($I/N_0$), respectively. The $C/N_0$ was measured using the tracking $C/N_0$ estimator and the $I/N_0$ was measured at the IF output of the front-end. The AGC function was turned off and the front-end was configured at maximum gain. During the experiment we acquired and tracked a near-zenithal GPS satellite, in the presence of the jammer.

We tested the interference protection offered by the implemented array acquisition and tracking algorithms by performing a sweep of the interference power while trying to acquire and track a visible GPS satellite signal. The interference $I/N_0$ power sweep started at 0 dB-Hz (no interference) and ended at 180 dB-Hz which represents a high power jammer nearby the GNSS receiver. We ran GNSS-SDR in real time and captured 10 seconds of the tracking $C/N_0$ estimations. Since GNSS-SDR produces an $C/N_0$ output each millisecond we recorded 10000 observations for each interference power level. Fig. 3 shows the results.

The single antenna receiver, without any time or frequency domain interference protection, was able to track the satellite signal until the interference power overtook the despreading gain. This situation happened at $I/N_0 = 50$ dB-Hz. In contrast, the nullsteering spatial filter connected to a conventional acquisition and tracking showed an excellent interference protection, and the receiver was able to acquire and track the satellite signal until the interference power reached 140 dB-Hz. Finally, enabling GLRT acquisition, we obtained an extra protection of 10 dB due to the ability of the algorithm to point the array towards the satellite by computing $\hat{r}_{xd}$.

VI. CONCLUSIONS

Interference mitigation is a key aspect in providing integrity to GNSS receivers. In this paper we have addressed the problem using array signal processing, which allows the use of spatial diversity to mitigate intentional/unintentional interfering sources. Many previous work has been focused on using this technology when the receiver is in tracking mode. However, acquisition is known to be the bottleneck and recent work address the issue of acquiring satellites in the presence of interferences using array-based receivers. Here, we propose the general architecture of a GNSS receiver equipped with multiple antenna elements that is exploiting this diversity in both acquisition and tracking modes. Moreover, we present an experimental setup with an array-prototype that feeds a software-defined radio receiver which is in charge of processing the conformed data stream in real-time. Experimental results confirm the flexibility and enhanced capabilities of such approach with respect to conventional single-antenna solutions.

ACKNOWLEDGEMENTS

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness project TEC2012-39143 (SOSRAD), by the European Commission in the Network of Excellence in Wireless COMMunications NEWCOM$^+$ (contract n. 318306)

REFERENCES


Fig. 3. Tracking satellite $C/N_0$ vs. interference $I/N_0$ for three different GNSS receiver implementations.