Software-Defined Radio GNSS Instrumentation for Spoofing Mitigation: A Review and a Case Study

Erick Schmidt, *Student Member, IEEE*, Zachary Ruble, *Member, IEEE*, David Akopian, *Senior Member, IEEE*, and Daniel J. Pack, *Senior Member, IEEE*

***Abstract*— Recently, several global navigation satellite systems (GNSS) emerged following the transformative technology impact of the first GNSS—U.S. global positioning system (GPS). The power level of GNSS signals as measured at earth's surface is below the noise floor and is consequently vulnerable against interference. Spoofers are smart GNSS-like interferers, which mislead the receivers into generating false position and time information. While many spoofing mitigation techniques exist, spoofers are continually evolving, producing a cycle of new spoofing attacks and counter-measures against them. Thus, upgradability of receivers becomes an important advantage for maintaining their immunity against spoofing. Software-defined radio implementations of a GPS receiver address such flexibility but are challenged by demanding computational requirements of both GNSS signal processing and spoofing mitigation. Therefore, this paper reviews reported SDRs in the context of instrumentation capabilities for both conventional and spoofing mitigation modes. This separation is necessitated by significantly increased computational loads when in spoofing domain. This is demonstrated by a case study budget analysis.**

***Index Terms*— Global navigation satellite systems (GNSS), global positioning system (GPS), instrumentation review, interference mitigation, software-defined radio (SDR).**

I. Introduction

THE success of the U.S. global position system (GPS) promoted broader deployment of other global navigation satellite systems (GNSS) as well for providing position, velocity, and time (PVT) information for users in civil, commercial, and military applications [1], [2]. After GPS became available to commercial and civil markets, many components of critical infrastructure and broadly used applications started relying on the continuous availability of PVT information. Today, a sudden shutdown of GNSS would have a tremendous impact on systems relying on them for navigation or timing. Examples of GNSS-reliant ecosystems include conventional and emerging autonomous transportation [3], cellular networks [4], and even regulation and measurement of phasors in power systems [5], [6].

With so many existing and new markets depend on GPS, it is essential for GPS receivers to withstand interference, intentional, or otherwise. GPS signals can experience unintentional interference from other radio frequency (RF) emitters or intentional interference due to jamming or spoofing attacks. Spoofers are intelligent jammers that transmit-specific counterfeit GNSS-like signals to force the receiver to compute erroneous positioning and timing [7], [8].

There have been many techniques developed to deal with both intentional and unintentional interference over the years. In particular, arrays of antennas are effective in validating known direction of arrivals (DOAs) of satellite signals at the expense of increasing receiver cost and size [9]–[11], which is undesirable for mass-market applications. There exist many single antenna techniques as well [7], [8], [12]–[16].

Nevertheless, spoofing techniques continuously evolve, and there are no universal mitigation techniques that address all current and future threats. In that light, upgradability of the GNSS receivers becomes an important issue to support current functionality for best achievable overall performance and protection against spoofing attacks. Conventional GNSS receivers rely heavily on hardware (HW) components due to intense computational requirements and are not flexible for essential upgrades. Emerging software-defined radio (SDR) solutions implement most of the critical operations in software (SW) mode and are highly flexible for upgradability. The flexibility introduced by SDRs also makes it preferred research and development instrumentation for fast prototyping and testing of new receiver architectures and algorithms.

Still, state-of-the-art SDR solutions do not match computational power of HW-based receivers [17], [18], and often do not support real-time (RT) operations. For acceleration, SDRs often employ field programmable gate arrays (FPGAs) [19]–[27], digital signal processors (DSPs) [28]–[31] or are fully implemented in SW on a host PC [32]–[54]. Of the mentioned SDR GNSS receivers, only a few address interference mitigation techniques.

Protection against spoofing attacks requires significant additional computational resources, which are not guaranteed by reported general-purpose solutions. With an existing considerable research spectrum related to the general field of GNSS SDRs [43], [45], [55], [56], the specific domain of spoofing mitigation in the context of instrumentation capabilities is not

Manuscript received March 3, 2018; revised June 6, 2018; accepted August 14, 2018. Date of publication October 4, 2018; date of current version July 12, 2019. This work was supported by the National Science Foundation under Grant ECCS-1719043. The Associate Editor coordinating the review process was Dr. Jesús Ureña. *(Corresponding author: Erick Schmidt.)*

E. Schmidt and D. Akopian are with the Department of Electrical and Computer Engineering, The University of Texas at San Antonio, San Antonio, TX 78249 USA (e-mail: erickschmidtt@gmail.com; david.akopian@utsa.edu).

Z. Ruble and D. J. Pack are with the Department of Electrical Engineering, University of Tennessee at Chattanooga, Chattanooga, TN 37403 USA (e-mail: zachary-ruble@utc.edu; daniel-pack@utc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIM.2018.2869261

systematically reviewed to the best of the authors' knowledge, and this paper addresses this existing gap.

Therefore, this paper, first, contributes a general-to-specific SDR review through the lens of interference mitigation and instrumentation budgeting. It includes: 1) an overall GNSS receiver classification; 2) reported performance of SDRs for this application; and 3) a demonstration of a significant surge in complexity between conventional GNSS and advanced mitigation operational modes by using a case study receiver and a cross correlation mitigation algorithm.

Section II presents a state-of-the-art overview of general domain GNSS-SDR solutions. Section III examines reported SDR applications for spoofing mitigation domain in terms of instrumentation capabilities. Section IV presents a case study receiver computational budget of common operations. Section V presents a reduced complexity minimum mean-squared error (MMSE) technique proposed in [12], which is used as a case study integrated with a receiver previously developed by the authors. Section VI examines aforementioned MMSE implementation in a fast prototyping RT SDR testbed, along with a computational complexity budget analysis. Section VII presents simulation and performance results of mitigation algorithms to validate functionality. Section IX finalizes with concluding remarks.

II. GNSS Receiver Design and Overview

SDR emerged with a substantially beneficial purpose: to place the analog-to-digital converter (ADC) as close to the antenna front-end (FE) as possible, so that all samples from the ADC are postprocessed in a reconfigurable SW mode rather than HW. This adds configurability, flexibility, and upgradability [55]. This trend has gained attention in GNSS receivers, especially and recently, in RT operation capabilities.

It was not until the last two decades that general-purpose processors (GPPs) have gained enough processing power to achieve RT operations that previous HW application-specific integrated circuits (ASICs) were performing, such as HW correlators. The main components of a GNSS receiver are an RF block, which consists of an active antenna, a low-noise amplifier, and an FE, followed by three common baseband blocks: acquisition, tracking, and navigation.

Acquisition is a process of coarse synchronization of the received signals with locally generated correlating replicas for estimating their time and frequency misalignment. For acquisition, recent advances use fast Fourier Transform (FFT)-based techniques to either replace correlation operations in the frequency domain (carrier replica) or the time (code replica) domain. Other advanced acquisition algorithms use shared FFT techniques for joint-space searching in both domains for even faster computations [57].

Tracking is a process of fine synchronization where continuous alignment of the incoming signal with the so-called local pseudorandom (PRN) code replicas (time domain) and carrier replicas (frequency domain) need to be attained. This becomes the most challenging task in GPS SDRs since tracking loops conduct continuous processing of an incoming satellite signal by estimating carrier phase and code phase offsets for a stable synchronization lock. This is done for all previously acquired (visible) satellite signals. Code phase estimation is obtained by a delay-locked loop, and carrier phase estimation is obtained by a phase-locked loop and/or a frequency-locked loop [32], [47], [58]. This allows for fine synchronization in both the time and frequency domains, resulting in successful despreading of the incoming GNSS signals for navigation data extraction. During tracking synchronization, correlator outputs from the delay-locked loop are grouped into unique frames to form navigation data and to compute pseudorange measurements [32], [47], [55].

Finally, if navigation data are successfully collected from enough satellites (four is the minimum), then the tracking stage gathers data from all channels, aligns data in a systematic set, and runs navigation algorithms to solve user PVT solutions [47], [58].

GNSS-SDR receivers are designed based on the aforementioned baseband modules and the RF block. Principe *et al.* [55] and Hein *et al.* [56] describe SW radio receivers with various HW/SW configurable components where baseband functionality (i.e., acquisition, tracking, and navigation) can be distributed after the RF block (FE sampler): reconfigurable hardware such as FPGAs, coprocessor units such as graphical processing units (GPUs), embedded specialized GPPs such as DSPs, and general-purpose host PCs. Principe *et al.* [55] have defined three architectures that divide GNSS functionality onto these components: classical HW, hybrid, and fully SW architectures, based on where most of these baseband modules are distributed. Other authors distinguish SDR categories between postprocessing and RT solutions [56], based on SDR being RT or not, and whether it is implemented on a PC or an embedded system.

As an alternate approach, we focus on correlators as being the most computational consuming operation in the receiver. In addition, while some authors associate FPGAs and DSPs as being in the same category [45], [55], other authors distinguish DSPs for their ease of use [59] and for having a non-HW-configurable GPP. Therefore, we categorize FPGAs and DSPs separately. We also isolate DSP-based GPPs from PC GPPs. In addition, by considering aforementioned HW/SW configurable components by their ease of use, we define a category similar to fully SW, which includes an even higher level component recently adopted in SDR receivers: a prototyping SW (P-SW) platform. A P-SW platform is defined as a high-level programming framework working atop the host PC operating system (OS) and has the ability to efficiently manage and optimize PC computational capabilities such as parallelism and multithreading (MT). Such platforms or environments can be LabVIEW (LV), MATLAB + Simulink (M + S), or even open-source solutions such as GNU Radio [60] and Python. Many of these P-SW platforms offer ease of use and fast prototyping by means of built-in block implementations. We then define the following categories:

1) FPGA where all correlations occur, and a host PC with (optional) PVT solutions and preconfigurations;
2) DSPs or embedded GPPs, and a host PC for (optional) PVT solutions and/or output visualizations;

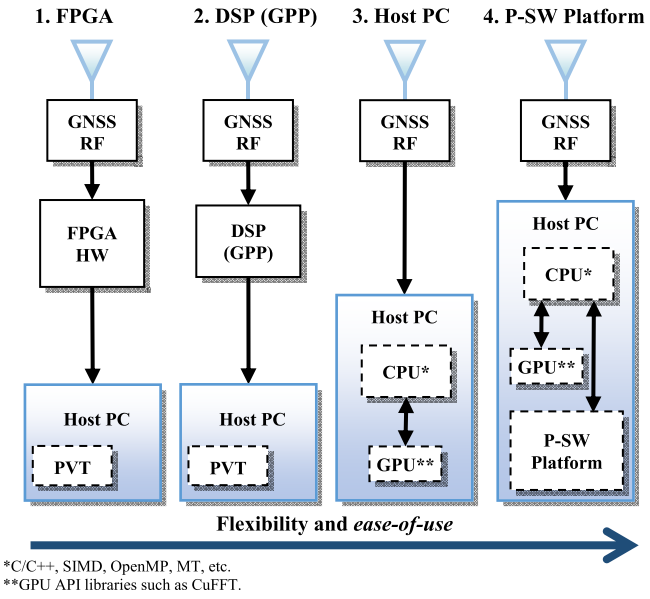

Fig. 1. GNSS SDR categorization based on flexibility and ease of use.

3) host PC where most baseband functions are executed;
4) P-SW platform, for fast prototyping.

This final category is a novelty in this paper and offering the highest flexibility in terms of SW configurability and ease of use. We attempt to categorize currently reported SDR solutions in said categories as shown in Fig. 1, with optional hybrid approaches across. We also avoid ASIC solutions [17], [18] as we focus on purely SDR implementations.

Table I summarizes current reported SDR solutions organized according to listed categories 1)–4) (see Fig. 1). For a given receiver category (row), all receiver references falling into this category are listed under said section. In addition, receiver category references are divided into three other columns: RT capabilities, their overall SW/HW architecture, and leveraged factors for acceleration, if any. These three rightmost columns are independent of each other, and the acceleration factor column can repeat a receiver reference based on implemented acceleration techniques.

Of all these SDR solutions, the combination of RT and ease-of-use implementations is considered the most valuable for research. Many solutions in category 1) implement reconfigurable HW accelerators in correlators [19]–[27] but offer less flexibility and ease of use, as opposed to GPPs found in categories 2)–4) [59]. Category 2) offers low-power embedded solutions such as DSP boards [28]–[31], which are not tailored for optimal speed. The ease of use of this category is considered intermediate to advance since it often requires low-level programming such as bit-wise operations [29], [31]. Category 3) offers more flexible environments, as it can use more generic programming skills and optimized libraries in the form of application programming interfaces (APIs) such as FFTs [32], [35], [38], [39], [41], [43]. Many accelerators in this category are C/C++ based and exploit multicore abilities for parallelism [59], MT [33]–[35], [38], [41], [43], [45], single-input multiple-data (SIMD) instructions [33], [34], [39]–[41], [43], [45],

TABLE I
REPORTED GPS RECEIVERS IN SDR CATEGORIES 1)–4) (SEE FIG. 1)

SDR category	Real-time capability	SW/HW architecture	Acceleration factors
1. FPGA [19]-[27]	Non-real-time [24], real-time [19]-[23], both [25]-[27]	FPGA/DSP board [19]-[21], [25], GPP-based PVT [19]-[23], [25]-[27], host PC visuals [22]-[24], [26]	FPGA accelerators [19]-[27]
2. DSP (GPP) [28]-[31]	Real-time [28]-[31]	DSP board [28]-[31], host PC visuals [28], [30], [31]	DSP accelerators [28]-[31], bit-wise operations [29], [31]
3. Host PC [32]-[45]	Non-real-time [32]-[34], real-time [35]-[43], both [44], [45]	C++-based [32]-[35], [38]-[41], [43], [44] APIs [34], [38], [41], [43], [44] Linux OS [36], [37], [45] GPU libraries [34], [39], [40], [41], [43] GUI [38], [41], [43], [44], [45]	Optimized libraries [32], [35], [38], [39], [41], [43] SIMD [33], [34], [39]-[41], [43], [45] MT [33]-[35], [38], [41], [43], [45] assembler [36], [38], [39], [44], [45] GPU APIs [33], [34], [39]-[41], [43] RT-Linux [36], [37], [45] bit-wise operations [35], [36], [41]-[43] OpenMP API [38], [41], [43]
4. P-SW Platform [46]-[54]	Non-real-time [46]-[51], both [52]-[54]	MATLAB-based [47]-[49], M+S [50], LV-based [46], [53], [54], open-source [51], [52]	FPGA accelerators [46], optimized libraries [52]-[54], MT [52]-[54], SIMD [52]-[54], P-SW accelerators [46], [53], [54]

OpenMP APIs to schedule CPU multicore resources with noticeable speedups on correlators [38], [41], [43], [59], GPU-based APIs for parallel arithmetic operations [33], [34], [39]–[41], [43], RT Linux OS [36], [37], [45], among others (see Table I). Category 3) can be associated with the fully SW option mentioned in [55]. Nevertheless, category 3) receivers often require advanced dedicated SW solutions which have evolved during many years and optimizations using dedicated OS platforms such as RT Linux [43], [44] and assembler instructions [36], [38], [39], [44], [45]. Moreover, API-based SDRs typically do not give access to the source code [41], thus limiting tight integrations with potential research algorithms. Category 4), which uses a P-SW platform for fast prototyping, commonly offers an ease-of-use framework such as in LV based [46], [53], [54] and M + S based [50]. This property enables researchers to test algorithms quickly and generally requires less programming effort and time, e.g., Ng and Gao [51] developed a Python-based vector tracking multireceiver. Nonetheless, SW platform-based SDRs often lack accelerators due to their ease-of-use nature. Therefore, a scarcity of RT receivers in said category proposes a challenge for ongoing research.

Recent research has proposed accelerators for category 4) that offer a combination of RT and fast prototyping [53], [54]. In terms of interference mitigation in the context of instrumentation capabilities, said SDR fits well for research extensions.

III. Review of GNSS SDRs for Interference Mitigation

Introducing algorithms that address interference mitigation generally increase the computational complexity of the receiver; thus, only a few of the many existing GNSS-SDR approaches operate in RT. For this reason, this section separates this category of receivers. Furthermore, we examine reported interference mitigation techniques with their corresponding host receivers.

A. GNSS SDRs With Interference Mitigation

Interference and spoofing of GNSS signals are a vast area of research. As mentioned in [7], certain signal properties of GNSS signals are vulnerable to and exploited by interference and spoofing attacks. An overview of types of interference attacks can be categorized into jamming and spoofing, which can be very similar but should be distinguished from one another. Some forms of jamming include narrowband or wideband continuous wave and RF interference (RFI) often called chirping signals [10], [14]. These techniques aim to make the receiver lose lock by transmitting overpowering signals. For spoofing, the attacks rely on subtlety by trying to take over the receiver's current position or time, rather than blocking its signal altogether. One of the most common spoofing approaches is meaconing, which is based on relaying satellite signals with increased power to introduce a delay on the target receiver and influence PVT outputs. The reader is directed to [7] and [8], for more relevant information on jamming and spoofing attacks.

Table II presents a categorized set of SDR solutions found in the literature that use interference mitigation techniques based on mitigation types described in [8], as well as their reported SDR platform and RT capability. Based on the mitigation categories column, Schmidt *et al.*[8] described these four groups of interference mitigation:

1) signal processing based, which exploit stages of a GNSS receiver on the RF chain, i.e., automatic gain control monitoring, as well as other common GPS stages such as acquisition, tracking, and/or navigation;
2) cryptographic based, which attempt to provide encryption on GNSS signals, specifically on payload navigation data broadcast from the satellites for additional layers of security and authentication;
3) correlation with other GNSS signals, which utilize additional constellations, frequency bands, and/or sensors to monitor and detect counterfeit signals on current GNSS signals;
4) radio spectrum and antenna based, which exploit multiple antenna techniques, such as angle-of-arrival (AOA), to spatially pinpoint the counterfeit signal sources for dissolution from authentic signals.

In addition, the specific technique column lists keywords that describe the overall method utilized for each mitigation category. For each overall method referenced in the specific technique column, an associated SDR platform is also referenced as the solution used, listed in the same order used in the specific technique column. If no SDR was reported for the specific technique, an N/A is used. For the RT column, the reference correlates to the specific technique's reference, and not the SDR platform's reference.

TABLE II
Overview of SDR Solutions With Interference Mitigation Techniques Divided in Categories [8]

Mitigation category	Specific technique	SDR platform	Real-time?
Signal processing	Post-correlators: [61], [62], [63]	[45], [34], [52]	[61], [63]
	wavelet + notch filter [64]	[38]	N/A
	vector-tracking correlators [65]	[34]	N/A
	MLE based on DP [66], and DTE [67]	[51], [51]	N/A
	Power analysis [62]	[34]	N/A
	MMSE + MWF [68], [69]	[53], N/A	[68]
Cryptograph	SCER [70], [72]	[71]	[72]
Correlation w/ other GNSS signals	GPS C/A and P(Y) correlation [73], [74], [75], [76], [77]	[71], [74], [29], [29], [51]	[73], [76]
Radio spectrum and antenna	Space-time correlation [79]	[34]	N/A
	AOA [80], [81]	[80], [71]	[80]
	Antenna arrays [82], [83], [84], [85], [86], [87]	N/A, [34], [34], [52], [52], [52]	[87]

1) Signal Processing Based: For the signal processing mitigation category found in Table II, several authors modify a certain step of the GNSS receiver chain to assess and mitigate interference. Similarly, they employ digital signal techniques for postprocessing and analysis. Falletti *et al.* [61] modified an RT SW receiver for detecting the presence of spoofing based on live postcorrelator outputs from the tracking stage of their GNSS receiver, NGene [45]. These correlator outputs were filtered through a Chi-square statistic to detect anomalies. The NGene receiver successfully attained live spoofing detection by using two additional correlators per channel. Similarly, Broumandan *et al.* [62] modeled correlator outputs with and without spoofing as a Gaussian distribution and attempted to detect spoofing by monitoring the model variance. Their simulations were run on a modified version of the GSNRx receiver [34]. Another postcorrelation application was introduced in [63], where Ranganathan *et al.* tracked vicinity peaks of the acquisition output for possible spoofing signals. This technique was implemented on an RT open-source receiver, GNSS-SDR [52]. Paonni *et al.* [64] evaluated the performance of wavelets for RFI, and notch filtering for continuous wave interference using recorded GPS data on their SW receiver platform IpexSR [38]. Vector tracking correlators reported in [65] rely on joint vector processing of concurrent tracking channels for monitoring of possible attacks and discards pseudorange measurements from suspected nefarious channels

TABLE III
COMPARISON TABLE FOR DIFFERENT SDR PLATFORMS ON INSTRUMENTATION CAPABILITIES FOR INTERFERENCE MITIGATION TECHNIQUES AS REPORTED IN INCLUDED REFERENCES

Group authors and SDR	Mitigation technique	SDR architecture	Real-time?	Sampling rate*	Live channels**	Acceleration factors
Case study receiver [53], [54]	**Signal processing**: MMSE blind detector [12], [68]	**Category (4)**: Host PC, USB FE, C++ DLL & LV platform.	Yes	Up to 25 Msps	Up to 12 live channels** [68]	SIMD, LV parallelizable loops, multi-threading, etc.
Jafarnia-Jahromi et al. *GNSRx* [34]	**Antennas**: space-time correlation [79], arrays [83], [84], **Signal processing**: vector tracking [65], power analysis [62]	**Category (3):** 3 channel NI FE, C++ modular design	N/A	Up to 25 Msps	Up to 22 total signals**	SIMD, multi-threading, GPU accelerators
Humphreys et al. *GRID* [29], *GRID* + spoofer/receiver, [71]	**Cryptography**: SCER [70], [72], **Correlation**: C/A and P(Y) [73]-[76], **Antennas:** AOA [80], [81]	**Category (2)**: GPP DSP in C++-based	Yes [72], [73], [76], [80]	5.7 Msps	Up to 14 live channels** [72]	DSP accelerators, C++ bit-wise parallel operations, LUTs
Gao et al. *PyGNSS* [51]	**Signal processing:** MLE based on DP [66], and DTE [67], **Correlation**: multi-layer multi-receiver [77]	**Category (4)**: Python-based SDR. USRP N210 and GN3S [47] for offline recordings	N/A	Up to 5 Msps	N/A	N/A
Fernandez-Prades et al. *GNSS-SDR* [52]	**Signal processing**: SPREE [63], **Antennas**: [85]-[87]	**Category (4)**: Host PC, *GNU Radio* [60], Linux-based, C++ open source	Yes [63], [87]	Up to 10 Msps [63].	One** [87]	Multi-threading, SIMD, modularity, C++ optimized libraries

*Maximum reported sampling rate per SDR with interference mitigation solution.

**Includes real-time conventional and spoofed channels.

before calculating the PVT solution. This solution used the aforementioned GSNRx platform in offline mode. A non-RT Python-based vector tracking receiver, PyGNSS [51], applied direct GPS positioning (DP) [66] and direct timing estimation (DTE) [67] techniques, which are based on a maximum likelihood estimator (MLE), to discern spoofed GPS parameters in vector processing, such as clock bias. Power analysis and monitoring were reported in [62] with GSNRx in offline mode. Finally, other signal processing techniques used cross correlation properties of PRN codes in optimization techniques, such as MMSE [12], [68] and multistage Weiner filtering, to separate the spoofer from authentic signals [69].

2) Cryptographic Based: For cryptographic-based techniques, Humphreys [70] used an improved GRID receiver [29] with an antispoofing and defender-receiver testbed reported in [71] for cryptographic mitigation techniques based on security code estimation and replay (SCER). The attack obtained an estimate of the security chip used on the received encrypted signals to replay it for authentication. The testbed was able to detect and mitigate SCER attacks in offline mode by using the modified GRID testbed [71]. Humphreys [72] achieved RT mitigation for up to 14 channels with said attacks.

3) Correlation With Other GNSS Signals: In correlation with other GNSS signals, O'Hanlon *et al.* [73], [75], [76], Psiaki [74], and Heng *et al.* [77] proposed using the GPS L1 band, which contains both the civilian C/A and the encrypted military P(Y) signals in the in-phase and quadrature, respectively. Authors correlated code phase and timing relations between both codes to detect possible spoofers. Receivers used were variations of GRID for [73]–[76] and PyGNSS for [77]. For most offline receivers, an existing Texas Spoofing Test Battery (TEXBAT) recording files database [78], which provide common spoofing scenarios for static and dynamic attacks, was used.

4) Radio Spectrum and Antenna Based: Finally, for the radio spectrum and antenna-based category, several techniques, such as space–time correlation introduced in [79], were proposed for multiantenna testbeds. Spoofing and jamming signals were also detected by DOA and AOA of incoming satellite signals in [80]–[86]. However, the cost and additional equipment needed for such a receiver may not be suitable for all applications. Arribas *et al.* [87] implemented an antenna array antispoofing testbed in their RT GNSS-SDR receiver [52].

While many SDR receivers with interference mitigation integration are listed in Table II, only [61], [63], [68], [70], [71], [73], [76], [80], and [87] provided RT mitigation capability. Said capability is considered state of the art because of its increased computational complexity in spoofing domain.

B. Interference Mitigation Solution Comparison

As an attempt to narrow the aspects of instrumentation capabilities on previously discussed SDRs with interference mitigation, in this section, we include four selected popular receivers for a more detailed analysis. Table III provides this summary. It also includes a receiver from [53] and [54], for a demonstration of computational complexity distribution of a case study interference mitigation algorithm (Section VI). Three of five solutions are category 4) P-SW platform with two being RT: LV-based solution [53], [54] and GNU Radio-based solution [52]. Another RT solution is category 2) DSP from Humphreys *et al. GRID* [29], [71]. Several maximum

sampling rates are seen across the reported and cited references, ranging from 5 to 25 Msps (see Table III). As for RT tracking, between 1 and 22 channels have been reported for selected solutions. When using 22 channels, typically half are used for conventional satellite tracking and the other half for interference injection and mitigation [71].

1) Jafarnia-Jahromi et al. (GNSRx): Jafarnia-Jahromi *et al.* have several research papers concerning their *GNSRx* [34] receiver. They implemented signal processing-based interference mitigation techniques, as well as radio spectrum and antenna-based methods (see Table III). Since their SDR receiver is not RT, a Spirent GPS simulator was used to generate signals for testing. The receiver uses a C++ modular design and uses three channels from an NI FE. They successfully implemented the receiver with 22 channels and sampling rates up to 25 Msps, normal and spoofed.

In the receiver, various GNSS signal operations are divided into high, medium, and low rate computational categories, which are performed in the 4–50 MHz, 50–1000 Hz, and 20 Hz or less range, respectively. To make the system adaptable to new algorithms, a modular object-oriented approach is used. The Doppler removal and correlation (DRC) object, used to track a given satellite signal, incurs the largest computational burden. To improve DRC processing, SIMD instructions for x86 processors and a multithreaded architecture are implemented. These are used as C++-based accelerators for faster computations.

In addition, an NVIDIA 8800GTX GPU is used in GSNRx to leverage DRC operations. The high degree of parallelism, as a result of the large number of available GPU threads, provides considerable processing improvements. GPU coprocessors can be divided into small threads. Each thread computes the local code and carrier phase, Doppler removal, and local code multiplication for each sample; it is responsible for and sums the result. It was shown in [34] that the average DRC processing time for 1 ms of data on eight satellites was less than 1 ms when sampled at 25 Msps when using the GPU, providing an RT operation, which was implemented in [40].

In [79], [83], and [84], antenna techniques were used with GSNRx to achieve spoofing detection and immunity. In [79], space–time correlation was used and in [83] and [84]; an array of antennas was implemented to detect and mitigate spoofing attacks. In [62] and [65], they implemented vector tracking and power analysis mitigation techniques, respectively. None of these testbeds were reported as RT since they used recordings from the Spirent simulator and used real, recorded signals to assess their performance.

2) Humphreys et al. (GRID): Humphreys *et al.* [71] developed an augmented version of GRID [29] for RT spoofing and detection of signals. The conventional GRID [29] receiver can track hundreds of GNSS live channels in its latest reported iteration [59]; nonetheless, when spoofing is implemented, this computational power decreases. The augmented system [71] is capable of tracking up to 12 live authentic channels and 12 spoofed channels (see Table III). The proposed SDR uses a DSP, with most code written in C++ for upgradability. Most of the optimizations carried out in GRID are bit-wise parallel operations leveraged from the DSP's architecture by using built-in AND, NOR, and XOR modules. They also use look-up tables stored in memory for fast local carrier and code generation. Their receiver's FE was able to sample at 5.714 MHz with an IF of 1.405 MHz. Its full range capability is a combination receiver and spoofer testbed, which can simultaneously generate 12 spoofer channels and defend them as reported in their latest SCER attack iteration [72].

Psiaki *et al.* [80] and Montgomery *et al.* [81] successfully tested and implemented a dual-antenna receiver using GRID, which detects phase and AOA of authentic and potential spoofing signals. By knowing delta phases calculated using the known antenna location and their separation, the technique has shown the ability to discern between authentic and counterfeit signals.

O'Hanlon *et al.* [73], [75], [76] and Psiaki *et al.* [74] used GRID in a different mitigation approach. They implemented an additional receiver, which observes the military GPS P(Y) signals for possible detection of attacks, assuming that this additional receiver signal from P(Y) is clean. This detection algorithm exploits the known phase-quadrature relationship of the encrypted P(Y) code relative to the C/A code and other properties. Only [73] and [76] were able to run in RT operation based on optimizations that were implemented in the receiver.

Finally, Humphreys [70] and Humphreys *et al.* [72] use cryptographic-based techniques for spoofing detection and mitigation. Assuming navigation data have a cryptographic security code associated with it, it is possible for the spoofer to execute an SCER attack, in which the spoofer attempts to estimate the security code chip value for each GPS signal it intends to attack. The defending receiver attempts to detect possible SCER attacks by means of hypothesis tests related to noise levels and the received signal power. The augmented GRID receiver performed close to RT for these detection techniques since a 2-ms delay was found on the SCER attack in [70], although Humphreys *et al.* [72] did report RT. TEXBAT [78] database was used in all nonlive experiments for this group of authors.

3) Gao et al. (PyGNSS): Ng and Gao [51] developed an advanced receiver based on vector tracking of concurrent GPS channels for added robustness. The concept is called multireceiver vector tracking (MRVT), which can be seen as multiple GPS receivers jointly tracked through shared receiver states and a single navigational filter. The navigation filter is commonly used as a modified, jointly shared, extended Kalman filter, to exploit joint tracking techniques. The receiver is non-RT and is mainly implemented in Python, an open-source interpreted language that lacks optimizations but is highly configurable. The receiver design is highly modular for fast adaptations and prototyping.

As the states of each receiver are shared using MRVT, many receiver optimizations can be achieved. For example, MLE can be used for joint observations of clock bias and clock drift in the navigation solution. In [66], a DP technique based on MLE of raw observations on the MRVT was implemented. In [67], a DTE technique similar to the previous DP solution was also implemented using joint observations for phasor measurement units (PMUs) in power grids. These PMUs share a known location. Thus, fewer unknowns can be exploited for

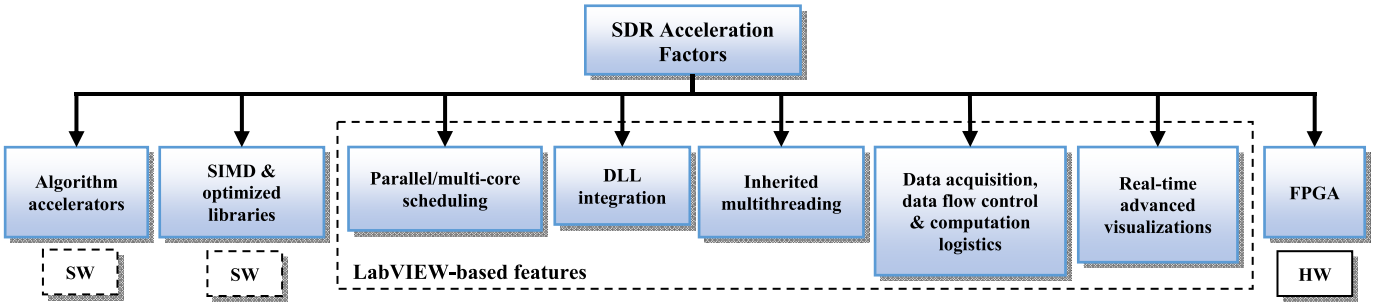

Fig. 2. Software-defined GPS receiver platform accelerator factors for RT operation.

DTE techniques. In [77], a multilayered multireceiver architecture was also proposed, where shared states from the MRVT are used to detect spoofing across channels. Several detection tests were implemented based on power analysis of joint elements between the multireceiver architecture.

4) Fernandez-Prades (GNSS-SDR): GNSS-SDR [52] is an open-source RT GNSS multifrequency receiver. The receiver uses the GNU Radio [60] framework as its core implementation. The receiver is highly configurable, but at the same time has many SW dependencies (including GNU Radio). It is based on a Linux environment and can utilize many FE options, such as USRP units from Ettus [88], GN3S sampler [47], NSL Stereo [89], and IFEN's SX3 [90]. It can also work with recorded binary files. The architecture is based on modularity and MT, meaning that each channel has its own independent acquisition, tracking, and navigation thread. Once the receiver has enough channels, it can then compute PVT solutions. The receiver's performance depends on the host PC's computational capabilities, but still leverages several acceleration factors such as SIMD and vector-based operations in Linux [52].

Due to their robustness, there has been considerable research in the use of antenna arrays for interference detection [85]–[87]. In these works, the authors built and implemented an eight-channel antenna array, and tested it using several scenarios, both during RT operation using the GNSS-SDR receiver and a commercial off-the-shelf GPS simulator, as well as with offline recordings [87]. They were able to run algorithms with enhanced acquisition and detection of spoofers using an antenna array, in RT on a single tracking channel.

Since the receiver is open source, Ranganathan *et al.* [63] implemented RT interference detection and mitigation on the same SW receiver using a signal processing technique. They added a parallel tracking channel for the same PRN code to detect a second correlation peak that might be deviating from the main peak, thus detecting a spoofer attempting to avert the receiver to an erroneous PVT solution. In this paper, Humphreys *et al.* [78] used TEXBAT recordings for offline testing as well as RT operation with GNSS-SDR.

IV. Case Study Receiver Budgeting Without Interference

Before introducing advanced mitigation techniques, a receiver used for the case study is explored. A LV-based GPS L1 single-frequency receiver reported by Schmidt *et al.* [53] and Schmidt and Akopian [54]. It implements baseband functionality in C/C++ units that have been compiled as dynamic link libraries (DLLs). The P-SW platform architecture uses several LV-based acceleration factors such as parallelizable loops, inherent MT, and other non-LV-based enhancements, such as SIMD Intrinsics from Intel [91] to exploit the host CPU, as shown in Fig. 2. For algorithm accelerators, an advanced acquisition module was implemented in SW based on [57]. For optimized libraries, FFTW [92] and Eigen [93] packages are used for FFT and matrix operations, respectively. LV allows FE interfacing and easy control based on automatic parallelization blocks that allow RT operation of the receiver. Advanced visualizations and GUI controls were also used for RT debugging and monitoring. Additional LV-based accelerations can be achieved via FPGA. As shown in Table III, up to 25 Msps can be achieved, and up to 12 live tracking and/or spoofing channels are attained.

For the case study interference mitigation algorithm implemented, a common configuration in the receiver is used: sampling rate at 5 MHz, int8 data type, and I-Q interleaved samples. For further details on SW and HW components, as well as performance tests, the reader is directed to [53] and [54].

In the following, a computational complexity budget breakdown of common operations is examined in the case study receiver. It will be followed in Section VI by a computational complexity budget analysis in interference mitigation mode to demonstrate a noticeable surge in computational load.

To assess instrumentation capabilities of the case study receiver, several offline computations are benchmarked to obtain the aforementioned budget of common GPS baseband operations. These operations, although in offline mode, provide a very close benchmark value to RT execution. In the context of host PC memory allocations and data type handling, previously mentioned common configurations are used for all budget operations. These common configuration parameters become relevant in terms of (linear) scalability of basic arithmetic operations, as well as memory allocation.

Sample data type is also important, as it specifies the number of quantization bits for sample resolution [47], [58]. In GPS SDR solutions, 8-bit samples have been proven to be enough for nominal operation and good precision. Some solutions use even fewer bits for faster computations [41],

TABLE IV
COMPUTATIONAL COMPLEXITY BUDGET FOR CONVENTIONAL SDR OPERATIONS AT 5-MHz SAMPLING RATE

Operation Type	Cycle time (ns)
FFT 8,192 (1 ms)	46,547.2
FFT 32,768 (4 ms)	399,186.2
Full acquisition (single PRN)	5,684,375.5
Single correlator*	1,689.03
Full tracking epoch**	18,635.0

*Single correlator includes code generation, mixing, and integration cycle time, for, e.g., early in-phase.
**Full tracking includes carrier wipe-off, 6 conventional correlators for in-phase and quadrature, for early, prompt and late, respectively, final integration, and discriminator output.

but since P-SW platform SDRs use multipurpose processors, the byte (8 bit) fits naturally.

As an example, using said common configuration, a 1-ms block of data would consist of 10 000 bytes since each sample is of byte size (int8), and 1 ms of raw data collected at 5-MHz sampling rate consists of 5000 samples for both the in-phase and quadrature interleaved components. Many FEs use intermediate frequency (IF) mixers, which output IF in-phase only samples, thus receiving a smaller block of samples but also requiring an extra down conversion step to get the baseband I-Q sample pair. For the case study SDR, the FE uses a direct down-converter, thus providing samples in baseband as an I-Q pair to save on extra operations. In terms of FFT operations, using a next power-of-two FFT size is common practice in SDR receivers. For 5-MHz sampling rate, the next FFT size would be of 8192 for a single 1-ms epoch. For a sampling rate increase to 10 MHz, the numbers would scale up by a factor of two, and a 1-ms block would now be of size 20 000 samples, considering previous logic. This also means that the correlators would need to deal with twice the number of operations, depending on their implementation. The FFT size would also jump to 16 384. The number of these operations provides an idea of computational budget for common SDR operations in RT operation.

Table IV shows a computational budget for the case study SDR. A full acquisition of a single PRN search benchmark is included. As mentioned previously, the acquisition operation executes on 4-ms integration time and a 10-KHz frequency search band, respectively. More on FFT sizes and the algorithm implemented can be seen in [46] and [57]. For common tracking operations, conventional correlators were benchmarked, i.e., in-phase and quadrature early, prompt, and late. This totals six correlators per GPS channel. As for the discriminators, a second-order delay-locked loop and phase-locked loop, and first-order frequency-lock loop were used, respectively [58]. Therefore, a full tracking cycle contains carrier generation and wipe-off, six correlators processing and integration, and common computations, i.e., atan for discriminators. For a single correlator, all these previous operations were considered after the carrier wipe-off. Navigation benchmarks are not included since they account for minimal expenses as PVT solutions run at 2 Hz.

V. CHOSEN INTERFERENCE MITIGATION ALGORITHM FOR RECEIVER BUDGETING

This section expands on a modified interference mitigation algorithm [68], which is a computationally optimized version of [12]. There are many other spoofing methods, nevertheless and without the loss of generality, the following algorithm is chosen because of the noticeable computational surge (times) of receiver operation in mitigation mode.

A. Background

The overall function of a GPS receiver is to synchronize to satellites to obtain navigation data and extract measurements for range estimations. The tracking synchronization is performed by correlating received (spreading) signals with locally generated (despreading) replicas of expected satellite PRN codes to maintain their alignment. A GPS received signal is composed of a superposition of multiple satellite codes contaminated by channel distortions and spoofing interference.

Conventional GPS receivers employ replica codes, which are the same as those sent by satellite transmitters. This ensures acceptable reception in outdoor areas but does not provide sufficient immunity against spoofing. There exist advanced solutions that enhance local (despreading) codes in the receivers to reduce these effects. Decorrelators and MMSE detectors [94] are typically used for mitigating the multiple access interference effects in spread spectrum systems. However, the MMSE approach is computationally intensive due to required autocorrelation matrix inversions. To simplify the processing, some investigators estimate the cross-correlated signal and subtract it from the weak signal channel [58], [95], [96]. This solution is not optimal, computational overhead is still high, and applies only to known jamming signals. An *ad hoc* solution is suggested in [97] using an additional orthogonalization process for better separation of weak and strong channels. There, the authors observed the performance of their method deteriorates for various conditions of available stronger jammers. An optimal solution is presented in [12], where Lacatus *et al.* propose an algorithm for GPS-like interferer mitigation. This method is further optimized computationally next and used as a case study for budgeting the operation complexity of the spoofer-mitigating SDR receiver.

B. Mitigation Algorithm Description

In the following, the signal model described in [12] is explored. Without loss of generality, it is assumed the GPS receiver is already coarsely synchronized with available GPS signals using an acquisition process, and each satellite signal is being tracked by finely aligning the received signal with a corresponding locally generated replica, as conventionally applied in GPS receivers in nominal conditions.

Each satellite is assigned a dedicated processing channel $k \in \{1, \ldots, K\}$. The received sampled signal for each channel, k, after carrier wipe-off and prior to code correlation is denoted as vector $\mathbf{r}_k$ of length N samples for one code period. This signal is a linear additive combination of the synchronized kth PRN satellite signal $\mathbf{s}_k^0$ of one code period length, power p_k, and sinusoidal and code modulated signals from other $K-1$ visible satellites, plus noise. In addition, the kth satellite signal is modulated by a navigation bit-sample b_k such that

$$\mathbf{r}_k = b_k \sqrt{p_k} \mathbf{s}_k^0 + \mathbf{i}_k + \mathbf{n}_k \tag{1}$$

where $\mathbf{i}_k$ is the interference of other $K-1$ satellite signals and $\mathbf{n}_k$ is the noise.

The receiver wipes-offs (despreads) the codes by multiplying to a despreading code $\mathbf{h}_k$ and integrating (i.e., a correlator is used to associate received signal and dispreading replica). The decision variable d_k for the channel k is then determined to be the following:

$$d_k = \mathbf{h}_k^T \mathbf{r}_k \tag{2}$$

where $\mathbf{h}_k$ is a unit norm vector that does not amplify or attenuate the received power during a bit sample. Each receiver channel minimizes its mean-squared error (MSE) cost function, represented as MSE_k to determine an optimal dispreading code [12]. The MSE for each satellite k is as follows:

$$\text{MSE}_k = E\left[\left(b_k - \frac{d_k}{\sqrt{p_k}}\right)^2\right]. \tag{3}$$

This can be interpreted as a normalized MSE in comparison with the definition used in [98]. Applying Karush–Kuhn–Tucker (KKT) conditions, the resulting optimal despreading code solution is given as [12]

$$\mathbf{h}_k = p_k \mathbf{R}_k^{-1} \mathbf{s}_k^0 \tag{4}$$

where $\mathbf{R}_k = E[\mathbf{r}_k \mathbf{r}_k^T]$ is the autocorrelation matrix of the received signal of channel k after carrier synchronization and $\mathbf{R}_k^{-1}$ is its inverse or pseudoinverse for minimum norm solution if the matrix is singular.

Lacatus *et al.* [12] proposed a group-weighting method that could tradeoff complexity versus performance in the code adaptation. Section V-C describes a computationally optimized version of this technique, as shown in [68].

C. Reduced Complexity Despreading Approach

We restrict the decoding sequence to the following format:

$$\mathbf{h}_k = \mathbf{w}_k(.^*)\mathbf{s}_k^0 \tag{5}$$

$$\mathbf{w}_k = [(w_{1,1}, \ldots, w_{1,g}), (w_{2,1}, \ldots, w_{2,g}), \ldots, (w_{M,1}, \ldots, w_{M,g})]^T \tag{6}$$

where $(.^*)$ is the element-by-element multiplication of the vectors and $w_{i,j}$ is the jth element of the ith group of size g, making $\mathbf{w}_k$ a vector of size $M = (N/g)$. As the elements of $\mathbf{s}_k^0$ are ± 1, then $\mathbf{h}_k^T \mathbf{s}_k^0 = g\mathbf{w}_k^T \mathbf{1}_M$. Let us split $\mathbf{h}_k$, $\mathbf{r}_k$, and $\mathbf{s}_k^0$ into M segments

$$\mathbf{h}_k = \begin{bmatrix} \mathbf{h}_{k1} \\ \mathbf{h}_{k2} \\ \vdots \\ \mathbf{h}_{kM} \end{bmatrix}; \quad \mathbf{r}_k = \begin{bmatrix} \mathbf{r}_{k1} \\ \mathbf{r}_{k2} \\ \vdots \\ \mathbf{r}_{kM} \end{bmatrix}; \quad \mathbf{s}_k^0 = \begin{bmatrix} \mathbf{s}_{k1}^0 \\ \mathbf{s}_{k2}^0 \\ \vdots \\ \mathbf{s}_{kM}^0 \end{bmatrix}. \tag{7}$$

The constrained $\mathbf{h}_k$ will be then as follows:

$$\mathbf{h}_k = \begin{bmatrix} \mathbf{h}_{k1} \\ \mathbf{h}_{k2} \\ \vdots \\ \mathbf{h}_{kM} \end{bmatrix} = \begin{bmatrix} w_1 \mathbf{s}_{k1}^0 \\ w_2 \mathbf{s}_{k2}^0 \\ \vdots \\ w_M \mathbf{s}_{kM}^0 \end{bmatrix}. \tag{8}$$

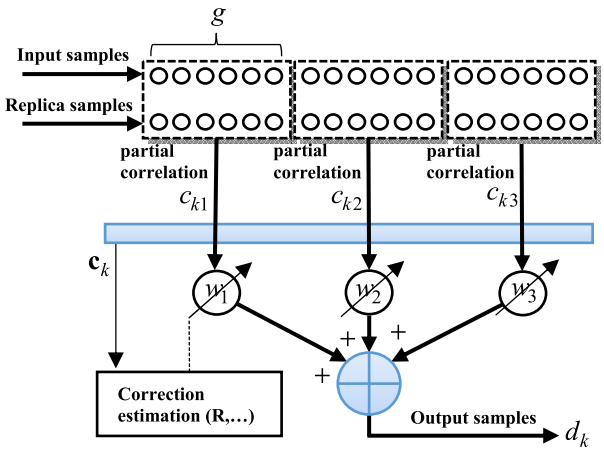

Fig. 3. Group-weighting method partial correlation sample interaction [68].

Denote $c_{kj} = \mathbf{h}_{kj}^T \mathbf{r}_{kj}$ as a partial correlation, and $\mathbf{c}_k = [c_{k1}, c_{k2}, \ldots, c_{kM}]^T$ as a vector of partial correlations. Then

$$d_k = \mathbf{h}_k^T \mathbf{r}_k = \mathbf{w}_k^T \mathbf{c}_k \tag{9}$$

$$\begin{aligned} \text{MSE}_k &= 1 - 2g\mathbf{w}_k^T \mathbf{1}_M + E\left[\frac{(\mathbf{w}_k^T \mathbf{c}_k)(\mathbf{c}_k^T \mathbf{w}_k)}{p_k}\right] \\ &= 1 - 2g\mathbf{w}_k^T \mathbf{1}_M + \frac{1}{p_k} w_k^T E\left[\mathbf{c}_k \mathbf{c}_k^T\right] w_k \\ &= 1 - 2g\mathbf{w}_k^T \mathbf{1}_M + \frac{\mathbf{w}_k^T \mathbf{R}_{ck} \mathbf{w}_k}{p_k} \end{aligned} \tag{10}$$

where $\mathbf{R}_{ck}$ is the autocorrelation of partial correlations. Minimization of MSE_k is achieved by applying KKT conditions, taking the partial derivative with respect to $\mathbf{w}_k$, and equating to zero

$$\frac{2\mathbf{R}_{ck}\mathbf{w}_k}{p_k} - 2g w_k^T \mathbf{1}_M = 0 \tag{11}$$

$$\mathbf{w}_k = g p_k \mathbf{R}_{ck}^{-1} \mathbf{1}_M. \tag{12}$$

These solutions are suboptimal for $g > 1$, in comparison with those provided by the optimal algorithm from (4). This is a result of the group-weighting method decreasing the freedom of designing dispreading code; alternatively, it has the advantage of significantly decreasing the computational complexity. Moreover, the solution presented for partial correlations can also be implemented with a computational complexity of $O(M^3)$. This solution has a computational gain, of order $O(g^3)$, in comparison with the conventional MMSE solution from (4). Fig. 3 shows the interaction of partial correlations $\mathbf{c}_k$ with input and replica samples, generating the optimal output sample d_k. Ultimately, these M weights $\mathbf{w}_k$ are correlated with obtain the maximum SINR solution in decision variable d_k, and are implemented in the tracking correlators for ideal navigation bit extraction.

The quality of a GNSS receiver operation is defined by the accuracy of the PVT solution. The presented algorithm addresses pre-PVT detection and mitigation of the interference and falls in the signal processing-based category. It applies a scalable blind equalizer-like technique to sense and remove

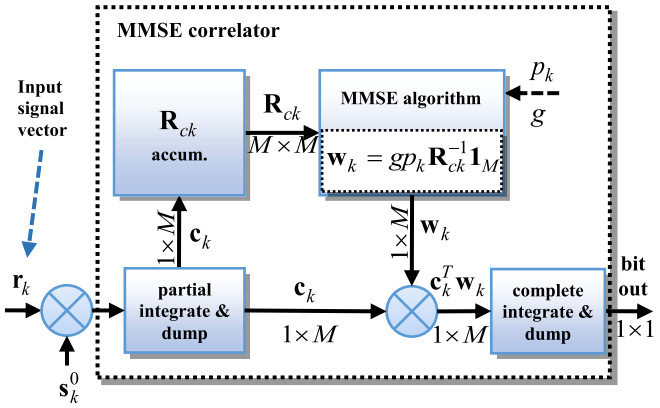

Fig. 4. Single MMSE correlator implementation in SDR tracking loop after carrier wipe-off coming from the IP channel [68].

certain interference per channel. The performance of this is not based on bit recovery but on bit-error rate, as is used to assess its quality.

VI. Case Study Receiver Budgeting With Interference

This chapter describes the implementation of the MMSE algorithm from Section V as an advanced correlator unit in the case study receiver. It further elaborates on the implementation as shown in [68] and addresses complexity through diverse optimization techniques. It describes a computational budget with said interference mitigation algorithm to complement the discussion from Section IV.

The MMSE correlator is presented as a modification to the conventional GPS tracking correlator. For this implementation, the in-phase prompt (IP) channel is adjusted. An essential aspect of the MMSE algorithm is the computation of an autocorrelation matrix seen in (12), which is obtained from a single correlator output. The autocorrelation matrix, obtained from the outputs of the partial correlator, serves as a blind sensing module for possible interferences. It senses cross correlation anomalies in the matrix and adjusts weights accordingly for interference filtering. Some options to control the computational complexity of these updates are the grouping parameter g and window size L, which are detailed next.

A. MMSE Correlator Implementation

Fig. 4 shows the MMSE correlator as an advanced unit, which can be added to the existing tracking loops. Specifically, it modifies a traditional correlator unit so there are partial integrate and dump (I and D) and complete I and D filters instead of common I and D filters to extract samples between said filters. This modified MMSE correlator is used to compute, after carrier wipe-off, the autocorrelation matrix of partial correlations $\mathbf{R}_{ck}$ corresponding to the method in (12). Using optimal weights, the MMSE correlator outputs an integrated value with filtered-out cross correlation interference.

Conventional tracking loops may have variable signal integration lengths (in samples) which depend on Doppler shifts. In the case study SDR, the tracking loop maintains a constant length of 1023 chips, using a preintegration of samples within one chip, to address Doppler shifts, e.g., one may have four or five samples per chip at a sampling rate of 5 MHz depending on the Doppler effect. Initially, the tracking loop collects a fixed size block of data corresponding to 1 ms. Thus, the locally generated carrier is also sampled at 5 MHz, which is equivalent to approximately 5000 samples per epoch, again including Doppler effects. When the carrier is wiped off, the samples should be preintegrated within each chip, thus ending with an array of $N = 1023$ chips. For the group-weighting method, the array is up-sampled to $N = 1024$ so the size is a power of two, and the chips can be grouped using g, as shown in Section V-C.

After chip preintegration and up-sampling, the received vector $\mathbf{r}_k$ is mixed with aligned in-prompt code replica $\mathbf{s}_k^0$ for code alignment. The next step involves a partial I and D block, which integrates a sequence of chips into M partial correlation groups, outputting the $1 \times M$ $\mathbf{c}_k$ vector as shown in Fig. 4.

The $\mathbf{R}_{ck}$ accum. block collects the $\mathbf{c}_k$ vector for statistical matrix estimation on every 1-ms epoch. Once the autocorrelation matrix $\mathbf{R}_{ck}$ has collected enough epoch vectors, thus reaching a suitable statistical significance; the MMSE solution can be linearly computed using (12) for the optimal group-weighting solution. This is done at the MMSE algorithm block. Once the matrix is suitable for inversion, it computes the solution on every epoch iteration. This block then outputs optimal $\mathbf{w}_k$ coefficients needed for interference filtering based on potential cross correlation interference patterns seen in the partial autocorrelation matrix $\mathbf{R}_{ck}$. The $\mathbf{w}_k$ coefficients are of size M, so they are matched to the received signal partial correlation vector $\mathbf{c}_k$ for corrections. Other inputs to the MMSE block are group size g and channel signal power p_k, the latter of which is estimated continuously on conventional tracking loops.

Last, another complete I and D block is used to integrate the result $\mathbf{w}_k \times \mathbf{c}_k$ for an optimal bit-sample MMSE correlator output. Since IP channel is used, this output corresponds to navigation data.

As mentioned, three preintegration steps are involved in the modified MMSE correlator: 1) chip preintegration from samples to chips; 2) partial integration from N to M, based on the grouping parameter g; and 3) complete integration from M to 1 for the navigation bit sample. The first two integrations are done in one step at the partial I and D block shown in Fig. 4 for reduced complexity implementation. Also, when comparing said modifications to a conventional GPS correlator, the MMSE correlator can be seen as a single-unit replacing classical GPS blocks. In addition, this unit can be implemented either as a replacement to a conventional correlator or as an additional unit in the receiver.

B. Recursive Autocorrelation Matrix Computation

The implementation of the $\mathbf{R}_{ck}$ accum. block seen in Fig. 4 is achieved by a recursive statistical method with attainable computational complexity for the current SDR testbed. Fig. 5 shows a detailed implementation of the $\mathbf{R}_{ck}$ accum. block. Consider a sequence of partial correlation vectors

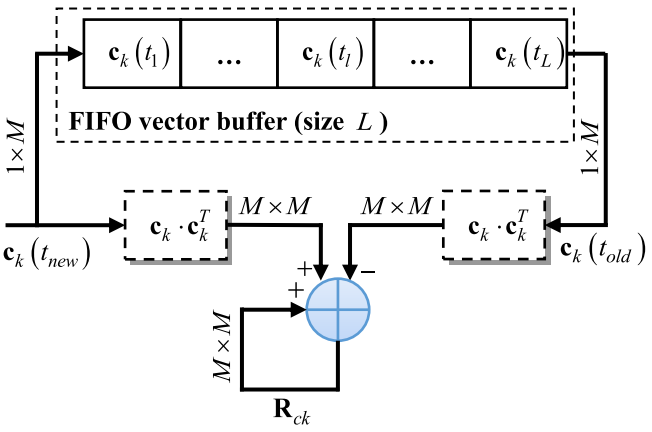

Fig. 5. Sliding window technique for recursive autocorrelation matrix computation and FIFO vector machine.

$\mathbf{c}_k(t_l), l = 1, \ldots, t_L$, that are used to estimate autocorrelation matrix $\mathbf{R}_{c_k}$ given as

$$\mathbf{R}_{ck}(t_L) = \frac{1}{L}\sum_{l=1}^{L}\mathbf{c}_k(t_l)\mathbf{c}_k(t_l)^T \tag{13}$$

where t_l is the time instant of $\mathbf{c}_k(t_l)$ epoch-vector availability and L is the window size, which is fixed during receiver initialization. To address complexity constraints for the case study SDR implementation, the autocorrelation matrix $\mathbf{R}_{ck}$, seen in (12), is computed recursively in a sliding window manner by discarding the oldest $\mathbf{c}_k(t_l)$ entry and adding the newest one as follows:

$$\begin{aligned}\mathbf{R}_{ck}(t_L+1) = \mathbf{R}_{ck}(t_L) &\\ -\frac{1}{L}\mathbf{c}_k(t_1)\mathbf{c}_k(t_1)^T &\\ +\frac{1}{L}\mathbf{c}_k(t_L+1)\mathbf{c}_k(t_L+1)^T.&\end{aligned} \tag{14}$$

This recursively computed matrix is based on a sliding window technique explained as follows. The $\mathbf{R}_{ck}$ accum. block seen in Fig. 4 receives as input vectors of partial correlations $\mathbf{c}_k$ of length M which are fed into a first-input first-output (FIFO) vector buffer of size L vectors and $M \times L$ samples. On each epoch, a new input vector of partial correlations $\mathbf{c}_k(t_{\text{new}})$ is received for a channel k, which is fed into the FIFO vector buffer. At the same time, the oldest value on the FIFO vector buffer is pushed out, thus keeping the buffer at a fixed window size, L. At the same time, the input vector of partial correlations $\mathbf{c}_k(t_{\text{new}})$ is multiplied by its transpose to generate a matrix of size $M \times M$ to be added to the $\mathbf{R}_{ck}$ accumulator seen in Fig. 5. Similarly, the oldest output vector of the buffer $\mathbf{c}_k(t_{\text{old}})$ is similarly subtracted from $\mathbf{R}_{ck}$ and then discarded. These additions and subtraction on the $\mathbf{R}_{ck}$ accum. block occur continuously on every 1-ms integration epoch, as per (14), until the FIFO vector buffer is full. This signifies a statistically sound autocorrelation matrix $\mathbf{R}_{ck}$.

From the C/C++ implementation perspective, both the size of the FIFO vector buffer as well as the $\mathbf{R}_{ck}$ matrix become relevant in terms of memory usage. The FIFO vector buffer and the matrix size are determined by grouping parameter g, vector length M, and sliding window size L. The FIFO vector buffer block and $\mathbf{R}_{ck}$ matrix are both implemented using dynamic memory allocation on variables that are initialized when called from the LV DLL call library function blocks. These memory allocations depend on initial user parameters g and L prior to runtime execution and are not adjustable during runtime execution. To adjust them, the receiver would have to be stopped, reconfigured, and reinitialized. There are calibrated recommendations for the grouping size g and sliding window L pairs: $g = 1$, $L = 1500$; $g = 2$, $L = 1000$; $g = 4$, $L = 1000$; $g = 8$, $L = 1000$; $g = 16$, $L = 500$; $g = 32$, $L = 100$; and $g = 64$, $L = 300$. In this paper, the configuration that achieves the best tradeoff between computation complexity and performance is $g = 64$ and $L = 300$, which allocates a dynamic variable array of 4800 samples. Another recommended configuration pair, but slightly heavier in computation complexity is $g = 64$ and $L = 1200$. This last pair consists of four times the window size for the same g parameter and provides improved blind adaptation response, as the matrix becomes more robust and more sound in statistical terms, but at the cost of memory allocation. This configuration pair will be used as a case study in Section VII-B.

Since testing is done on navigation data mitigation abilities, a parallel MMSE correlator is used to bypass spoofing effects on synchronization loops. Therefore, this MMSE correlator unit is separated from nominal receiver operations to explore its instrumentation and mitigation capabilities at the correlator level and without disturbing common tracking operations and synchronization loops.

C. Autocorrelation Matrix Numerical Estimation

The recursive autocorrelation matrix accumulated on every epoch is assumed to be full rank and real valued. In realistic scenarios, however, it can be affected by numerical estimation errors and become ill-conditioned or singular. This paper follows [99], which applies a full-rank regularization technique based on eigenvalue spread. This regularization technique is a constrained minimum output energy estimator that substitutes $\mathbf{R}_{ck}^1$ in (12) as follows:

$$\mathbf{w}_{\text{CMOE},k} = g p_k \left(\mathbf{R}_{ck} + \nu \mathbf{I}_{\text{MM}}\right)^{-1} \mathbf{1}_M \tag{15}$$

where ν is a positive-valued aid parameter that moderates eigenvalue spread in the autocorrelation matrix, which is common in multiuser detection. This approach was tailored for the MMSE correlator unit by fixing a calibrated value as the aid parameter for invertibility. This technique was implemented internally in the receiver and is used continuously for every epoch computation of the solution in (12).

D. MMSE Correlator Block Operations and Numerical Approach

It is easy to surpass computational capacity when linear algebra floating-point operations are involved, especially if RT operation is required. The operation count for the individual blocks on the MMSE correlator unit (see Fig. 4), including the recursive autocorrelation matrix accumulator (see Fig. 5) and

TABLE V
OPERATION COUNT (FLOPS) FOR SEVERAL STAGES ON MMSE CORRELATOR ON EVERY EPOCH

Stage	Operation Count (FLOPs)
Interference Injection $\mathbf{i}_k$	N
Partial integrate and dump $(N \rightarrow M)$	$M(g-1)$
Old $\mathbf{c}_k \mathbf{c}_k^T$	M^2
New $\mathbf{c}_k \mathbf{c}_k^T$	M^2
Accumulate $\mathbf{R}_{ck}$	$3M^2$
Compute (12) solution	$2M + M^2 + M^3$
Integrate bit-sample $\mathbf{c}_k^T \mathbf{w}_k$	$2M$

the reduced complexity solution (12) computation, is presented in floating-point operations (FLOPs) [100] in Table V. Without the loss of generality, we consider $N = 1024$ and a grouping parameter of $g = 64$ with a window size of $l = 300$, which proves to be a feasible approach on the case study receiver. The resulting size of the matrix is $M = 16 \times 16$. Table V shows an estimated count of FLOPs for each stage involved in the MMSE correlator unit. These operations are for every 1 ms (epoch) of a bit sample. For the first step, which involves interference injection, up to three interferers can be configured, each one accounting for N FLOPs. These interferers inject chip sequences of size N with cross correlation jamming effects. More on interference injection is discussed in Section VII-A. The partial correlations stage involves reducing the received signal vector ($N \rightarrow M$). The matrix accumulation (see Fig. 5) can be seen of order $O(M^2)$. The MMSE solution in (12) is an estimated approach on FLOPs since it contains a matrix inverse operation that requires to be computed on every epoch iteration. This stage accounts for the most operations as it is of order $O(M^3)$.

E. MMSE Unit Integration Budget

Similar to conventional GPS computational complexity budget operations listed in Section IV, this section provides a budget including case study mitigation operations to assess complexity load. It also discretizes common RT operations as individual units for comparison.

Tracking correlators and discriminators are commonly implemented in HW for faster operation since they are updating their discriminators and integrators on every given epoch (1 ms). Other advanced correlators have longer integration times, e.g., 10–20 ms, but are considered for advanced receivers. Thus, this 1-ms measurement becomes the threshold for RT operation in SW. Therefore, all relevant computations, occurring at each epoch including carrier and code generators, correlators, and tracking discriminators, should be computed in less than 1 ms for the SW receiver to be considered RT. If this ensemble of conventional GPS computations can be processed in less than 1 ms of integration time, it leaves additional computational resources for aggregate operations, such as interference injection and mitigation algorithms similar to those seen in Table V.

TABLE VI
REAL-TIME COMPUTATIONAL COMPLEXITY BUDGET IN TIME CYCLE OF 1-ms EPOCH

Operation type	Scope (*corr* or *chan***)	Cycle time (ns)	Share of computational time* (%)
SIMD carrier generation and wipe-off	*chan*	8,500.8	0.85%
SIMD code generation, wipe-off, and integration	*corr*	1,655.7	0.17%
Tracking discriminators	*chan*	200.0	0.02%
SIMD jamming injection	*corr*	5,263.8	0.53%
MMSE pre-integrator block $\mathbf{c}_k$ ***	*corr*	28,683.8	2.87%
Accumulate $\mathbf{R}_{ck}$ (14)	*corr*	26,348.0	2.63%
MMSE solution (12)	*corr*	8,502.2	0.85%
Integrate bit $\mathbf{c}_k^T \mathbf{w}_k$	*corr*	78.8	0.01%

*Computing time period cycle of 100% is equivalent to 1,000,000 nsec or 1 ms.
**Scope of the operation is either *corr* (correlator channel) or *chan* (full GPS channel).
**Pre-integrator modifies received $\mathbf{r}_k$ vector of size, e.g., 5,000 samples at 5 MHz, to $\mathbf{c}_k$ vector of size 16.

Conventional tracking and MMSE correlator unit operations from Table V can be seen in Table VI as a RT computational budget benchmarked from the case study receiver in mitigation mode. This computational budget relates all operation benchmarks to 1 ms, which represents 100% of the shared computations for the time period cycle.

SIMD-based correlators for carrier and code are shown for a single epoch using 1-ms integration time. For code correlation, a single unit benchmark is shown, i.e., IP correlator, as opposed to six correlators [58]. This encompasses code generation, wipe off, and integration. If the full nominal operation of a single GPS channel would be assessed, then six correlator values and a carrier value would need to be accounted for. Moreover, if 12 GPS channels are assumed to be tracked, then this number would be scaled properly. Tracking discriminator computations are also shown, which comprises feedback updates from tracking loops as discussed in Section IV. For interference injection, an SIMD jammer benchmark was added per correlator channel, representing a single interferer's computational expense. The preintegrator of vector $\mathbf{c}_k$ is measured per correlator channel, and integrates from samples to preintegrations, e.g., 5000 to 16, as mentioned in Section VI-A. The accumulation on $\mathbf{R}_{ck}$, as seen in Table V, represents the full computation seen in (14) for a single correlator channel. The MMSE solution in (12) was similarly evaluated for a single correlator channel. Finally, the full integration of optimized coefficients with grouping vector $\mathbf{c}_k$ for the navigation bit sample is measured.

Table VI shows case study receiver tracking operations. It can be seen that a single GPS channel accounts for roughly $\sim$1.9% of the total computational budget. Since the receiver can run up to 12 channels, this would be scaled to $\sim$22.4%, thus leaving more than 70% of the computational budget in conventional mode. Additionally, previous testing on case study SDR showed the LV environment would provide an additional overhead of approximately $\sim$3%, which covers the

FE interface and communication, and data acquisition among other common operations [46]. This ∼3% is included in the total budget computations to approximate live capabilities of the receiver in conventional mode.

Alternatively, a single channel being jammed with one interferer and mitigated spends roughly ∼8.8%, a 6.9% rise from the nonmitigation mode. Therefore, full implementation of the conventional operation, jamming, and mitigation mode on 12 live channels reaches the 100% threshold for RT operation. This shows a 5 times surge in computational complexity in the case study receiver when in mitigation mode for the same common configuration. For additional comparison purposes, the SDR was able to achieve a maximum of eight channels at 10-MHz sampling rate, with similar full implementation in mitigation mode, when contrasting a maximum configuration of 25-MHz sampling rate with 12 live channels in nonmitigation mode. These comparisons demonstrate a computational surge even for a fully optimized mitigation algorithm (see Section V) [68]. Section VII assesses the performance of the algorithm filtering capabilities in terms of bit-error rate (BER) for a single channel when interference is added to real GPS signals.

VII. Simulation Results

In this paper, specific interference is used to represent an idealized perfectly synchronized GPS-like spoofer with malicious intentions that affect cross correlation of despreading codes. The case study method does not specifically detect the interference but adjusts when such phenomena are observed (blind estimator). The interference injected mimics a worst case cross correlation scenario.

The interference injection occurs right before the code wipe off and after the carrier wipe off; therefore, the interference signal does not contain Doppler effects. This is assumed to be the worst case scenario for an attack since a spoofer would have fully modulated power to the intended in-phase channel. This creates the most damage, thus, can be used as a reference scenario. This approach makes the receiver immune to GPS-like synchronized intervention.

The case study SDR with interference injection and mitigation is evaluated by using a hybrid approach of real signals coming from a LabVIEW-based NI GPS simulation toolkit [101] and internally generated synthetic interference. The method in [12] addresses a type of spoofer that can generate an exact same but time-shifted signal per channel using worst case scenario delays, which have the greatest effect. Since code delays and Doppler shifts are known in the interference injection, a potential spoofer can use it to its advantage by filtering signals into the channel via cross correlation interference.

Specifically, interference injection occurs by replicating the kth channel code replica $\mathbf{s}_k^0$ and delaying it a known number of chips. The delayed code replica, $\mathbf{s}_k^\alpha$ (where α is a delay in chips), is based on known cross correlation properties of the PRN sequence for satellite k, thus spoofing the modulated navigation bits coming from GPS satellites. The spoofing power amplifies when at the higher sidelobes of the gold code cross correlation. This causes the correlation to lose orthogonality [102]. Therefore, the most efficient spoofing can occur at these sidelobes with delay α.

The interferer navigation bits are modulated onto the spoofing signal by assigning random bit polarity every 20 bit samples. The authentic and spoofed navigation bits are not necessarily synchronized in terms of bit edges since the spoofing bits can be injected at any time.

As mentioned in Section VI-E, the case study receiver can successfully jam up to 12 live channels. The grouping parameter was set to $g = 64$ with a window size of $l = 300$. As an aggregate experiment, we also tested $g = 64$ and $l = 1200$, which is four times the window size of previous parameters as discussed in Section VI-B.

A. Performance Results

In this section, we assess [12] in a case study receiver for algorithm functionality and RT operation [68] per channel for pre-PVT mitigation. Without the loss of generality, we simulate interference power relative to signal power in dB scale. Interference-to-signal ratio (ISR) noise power can be neglected when compared to the sum of the interference and signal powers after channel synchronization and integration.

We assess testing scenarios for matched filter (MF) correlators [58], and the case study MMSE correlator for a single channel. BER versus ISR performance curves are plotted and evaluated. This is done to study the cleaning capabilities of the algorithm in the presence of strong interference, compared to an MF correlator. Similarly, a worst case scenario of three jamming signals per channel, where all three jammers match in chip and bit alignment as well, is included.

Our goal is to obtain statistically significant BER performance results for functionality purposes in the chosen case study mitigation technique. Thus, we simulate 3 million navigation bits or 60 million bit-samples, corresponding to 1000 min of simulation. We perform the simulations with a 200-s real signal file previously recorded from the NI GPS simulator [101]. The recorded file has strong GPS signals; thus, noise can be neglected as previously mentioned. We proceed to run the file 300 times to obtain our navigation bits quota while injecting interference bits with relative power levels from reported SNR from target channel, and random bit polarity. This is all done on the in-house LV-based SDR [53], [54], which was prototyped for this test as in [68].

A well-known cross correlation immunity of around 26 dB is present in MF correlators based on GPS system design [1]. Nevertheless, after or around this ISR level (26 dB) the channel is completely corrupted. Fig. 6 shows the performance curve for one and three interfering signals, where BER has been assessed for 20 bit samples corresponding to a full navigation bit. The curve compares the performance of MF versus MMSE correlators. It also compares an MMSE correlator with 4 times the proposed window length discussed in Section VI-B. The comparison approach assesses the BER performance at a 30-dB ISR level (see vertical dashed line), which is well above the aforementioned 26-dB immunity level. This 30-dB ISR level would notably corrupt the GPS channel as it can be clearly seen for the cases of MF (1) and MF (3), corresponding

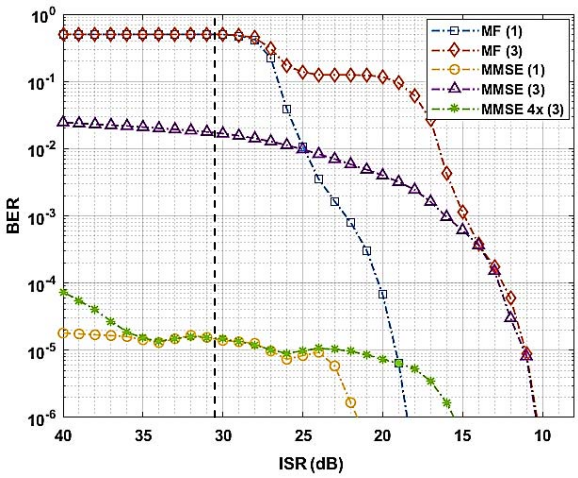

Fig. 6. BER versus ISR performance results for one and three interferers for MF and case study MMSE approach [68].

TABLE VII
RECEIVER BER PERFORMANCE FOR SPECIFIED TARGET ISR = 30 dB TO ASSESS GAINS AFTER GPS IMMUNITY LEVELS

	MF (1)	*MMSE (1)*	*MF (3)*	*MMSE (3)*	*MMSE 4x (3)*
ISR=30dB	$10^{0.3}$	$10^{-4.9}$	$10^{0.3}$	$10^{-1.8}$	$10^{-4.9}$

to MF with 1 interferer, and MF with three interferers present, respectively. For both these cases, the channel is corrupted with a BER of 50%.

When comparing against the MMSE correlator, we can see MMSE (1) shows a BER gain of almost 10^5 against MF (1), for one interferer. *For MMSE (3),* a BER gain of around $10^{1.5}$ can be seen against *MF (3)*, for three interferers. Finally, when increasing the window size four times used in MMSE (3), i.e., using MMSE 4× (3), a BER gain of around 10^5 is seen against MF (3), similar to the gain seen in MMSE (1) versus MF (1), but for three interferers. Table VII summarizes these BER performance results at the 30-dB ISR observation dashed line (see Fig. 6).

VIII. CONCLUSION

This paper provides an overall review and categorization of reported GNSS-SDR receivers for conventional and spoofing domains, in terms of reported instrumentation. Many conventional SDRs are not able to support mitigation modes in RT, which necessitates separate categorization of spoofing mitigation receivers. A case study demonstration of a significant surge in complexity from a conventional to a mitigation receiver is presented. This is done through computational budgeting of an SDR receiver employing a modified MMSE algorithm from [12] optimized for RT operation [68] in an LV-based receiver [53], [54]. The expanded version of [68] is presented, where the (pre-PVT) mitigation is accomplished through a blind equalizer processing. The approach was validated and assessed using BER performance curves, as [12] did not demonstrate operation with real signals and in RT. It is selected due to demanding computational complexity. To analyze said complexity increase the paper tabulates a breakdown of common algorithmic operations for conventional and mitigation mode of the case study receiver. A surge of the computational load from 20% to 100% (5 times) of the case study receiver was seen when in mitigation mode compared to conventional operation. A common configuration of 5-MHz sampling rate, int8 I-Q interleaved samples, and 12 live channels were used. It should be mentioned that the MMSE correlator unit could potentially be incorporated *in lieu* of any of the conventional correlators.

The case study employed an idealized spoofer perfectly synchronized in both carrier frequency and phase with the authentic signal, which allowed concentrating on computational budgeting aspects and not on spoofing-specific effects. At the same time, the selected idealized environment can simulate worst case scenarios as a baseline. Authors currently address spoofing mitigation aspects more comprehensively including sophisticated advancements of [12] and [68], which will be presented in the future dedicated work. This projected work will align various methods with state-of-the-art spoofing mitigation techniques and test with representative signals such as TEXBAT [78].

In summary, when tightly integrating interference mitigation algorithms, especially in RT manner, the complexity surge truly differentiates this class of GNSS-SDR receivers.

ACKNOWLEDGMENT

The authors would like to thank anonymous reviewers for their valuable assessment in improving this paper.

REFERENCES

[1] P. Misra and P. Enge, *Global Positioning System: Signals, Measurements, and Performance*. Lincoln, MA, USA: Ganga-Jamuna Press, 2012.
[2] C. J. Hegarty and E. Chatre, "Evolution of the global navigation satellitesystem (GNSS)," *Proc. IEEE*, vol. 96, no. 12, pp. 1902–1917, Dec. 2008.
[3] I. Skog and P. Handel, "In-car positioning and navigation technologies—A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 10, no. 1, pp. 4–21, Mar. 2009.
[4] K. J. Zou *et al.*, "Network synchronization for dense small cell networks," *IEEE Wireless Commun.*, vol. 22, no. 2, pp. 108–117, Apr. 2015.
[5] P. A. Crossley, H. Guo, and Z. Ma, "Time synchronization for transmission substations using GPS and IEEE 1588," *CSEE J. Power Energy Syst.*, vol. 2, no. 3, pp. 91–99, Sep. 2016.
[6] A. Carta, N. Locci, C. Muscas, and S. Sulis, "A flexible GPS-based system for synchronized phasor measurement in electric distribution networks," *IEEE Trans. Instrum. Meas.*, vol. 57, no. 11, pp. 2450–2456, Nov. 2008.
[7] M. L. Psiaki and T. E. Humphreys, "GNSS spoofing and detection," *Proc. IEEE*, vol. 104, no. 6, pp. 1258–1270, Jun. 2016.
[8] D. Schmidt, K. Radke, S. Camtepe, E. Foo, and M. Ren, "A survey and analysis of the GNSS spoofing threat and countermeasures," *ACM Comput. Surv.*, vol. 48, no. 4, May 2016, Art. no. 64, doi: 10.1145/2897166.
[9] M. G. Amin and W. Sun, "A novel interference suppression scheme for global navigation satellite systems using antenna array," *IEEE J. Sel. Areas Commun.*, vol. 23, no. 5, pp. 999–1012, May 2005.
[10] R. L. Fante and J. J. Vaccaro, "Wideband cancellation of interference in a GPS receive array," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 36, no. 2, pp. 549–564, Apr. 2000.
[11] J. Arribas, C. Fernandez-Prades, and P. Closas, "Antenna array based GNSS signal acquisition for interference mitigation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 1, pp. 223–243, Jan. 2013.

[12] C. Lacatus, D. Akopian, and M. Shadaram, "Reduced complexity crosscorrelation interference mitigation in GPS-enabled collaborative ad-hoc wireless networks–theory," *Comput. Elect. Eng.*, vol. 38, no. 3, pp. 603–615, May 2012.

[13] G. Seco-Granados, J. A. Fernandez-Rubio, and C. Fernandez-Prades, "ML estimator and hybrid beamformer for multipath and interference mitigation in GNSS receivers," *IEEE Trans. Signal Process.*, vol. 53, no. 3, pp. 1194–1208, Mar. 2005.

[14] D. Borio, "GNSS acquisition in the presence of continuous wave interference," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 46, no. 1, pp. 47–60, Jan. 2010.

[15] P. Closas, C. Fernandez-Prades, and J. A. Fernandez-Rubio, "Maximum likelihood estimation of position in GNSS," *IEEE Signal Process. Lett.*, vol. 14, no. 5, pp. 359–362, May 2007.

[16] S. Savasta, L. L. Presti, and M. Rao, "Interference mitigation in GNSS receivers by a time-frequency approach," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 1, pp. 415–438, Jan. 2013.

[17] F. Dominici, P. Mulassano, D. Margaria, and K. Charqane, "SAT-SURF and SAT-SURFER: Novel hardware and software platform for research and education on satellite navigation," in *Proc. Eur. Navigat. Conf. Global Navigat. Satellite Syst. (ENC GNSS)*, Naples, Italy, May 2009. [Online]. Available: http://www.enc-gnss09.it/

[18] P. Fenton, B. Falkenberg, T. Ford, K. Ng, and A. J. Van Dierendonck, "NovAtel's GPS receiver: The high performance OEM sensor of the future," in *Proc. Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Albuquerque, NM, USA, 1991, pp. 49–58.

[19] W. Roberts *et al.*, "PRECISIO—Design considerations for a multi-constellation, multi-frequency software receiver," in *Proc. 5th ESA Workshop Satellite Navigat. Technol. Eur. Workshop GNSS Signals Signal Process. (NAVITEC)*, Noordwijk, The Netherlands, Dec. 2010, pp. 1–5.

[20] T. Paakki, J. Raasakka, F. D. Rosa, H. Hurskainen, and J. Nurmi, "TUTGNSS University based hardware/software GNSS receiver for research purposes," in *Proc. Ubiquitous Positioning Indoor Navigat. Location Based Service (UPINLBS)*, Kirkkonummi, Finland, Oct. 2010, pp. 1–6.

[21] S. Fantinato, L. Foglia, P. Iacone, D. Rovelli, C. Facchinetti, and A. Tuozzi, "PEGASUS–GNSS receiver platform for safety of life user segment," in *Proc. 6th ESA Workshop Satellite Navigat. Technol. Eur. Workshop GNSS Signals Signal Process. (NAVITEC)*, Noordwijk, The Netherlands, Dec. 2012, pp. 1–8.

[22] A. Fridman and S. Semenov, "System-on-chip FPGA-based GNSS receiver," in *Proc. East-West Design Test Symp.*, Rostov-on-Don, Russia, Sep. 2013, pp. 1–7.

[23] G. Kappen and T. G. Noll, "Application specific instruction processor based implementation of a GNSS receiver on an FPGA," in *Proc. Design, Automat. Test Eur. Conf. (DATE)*, Munich, Germany, Mar. 2006, p. 6.

[24] M. S. Meraz, J. M. C. Arvizu, and A. J. A. Cruz, "GNSS receiver based on a SDR architecture using FPGA devices," in *Proc. IEEE Electron., Robot. Automot. Mech. Conf. (CERMA)*, Cuernavaca, Mexico, Nov. 2011, pp. 383–388.

[25] M. Grondin *et al.*, "A new operational low cost GNSS software receiver for microsatellites," in *Proc. 5th ESA Workshop Satellite Navigat. Technol. Eur. Workshop GNSS Signals Signal Process. (NAVITEC)*, Noordwijk, The Netherlands, Dec. 2010, pp. 1–5.

[26] G. Artaud, L. Ries, M. Monnerat, H. Al-Bitar, F. Legrand, and M. Weyer, "Development of a flexible real time GNSS software receiver," in *Proc. 5th ESA Workshop Satellite Navigat. Technol. Eur. Workshop GNSS Signals Signal Process. (NAVITEC)*, Noordwijk, The Netherlands, Dec. 2010, pp. 1–8.

[27] B. Sauriol and R. J. Landry, "FPGA-based architecture for high throughput, flexible and compact real-time GNSS software defined receiver," in *Proc. Nat. Tech. Meeting Inst. Navigat.*, San Diego, CA, USA, Jan. 2007, pp. 708–717.

[28] J. Tian, W. Ye, S. Lin, and Z. Hua, "SDR GNSS receiver design over stand-alone generic TI DSP platform," in *Proc. IEEE 10th Int. Symp. Spread Spectr. Techn. Appl.*, Bologna, Italy, Aug. 2008, pp. 42–47.

[29] T. E. Humphreys, M. L. Psiaki, P. M. Kintner, Jr., and B. M. Ledvina, "GNSS receiver implementation on a DSP: Status, challenges, and prospects," in *Proc. 19th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Fort Worth, TX, USA, Sep. 2006, pp. 2370–2382.

[30] J. Raasakka, H. Hurskainen, T. Paakki, and J. Nurmi, "Modeling multi-core software GNSS receiver with real time SW receiver," in *Proc. 22nd Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Savannah, GA, USA, vol. 1, 2009, pp. 468–473.

[31] E. G. Lightsey, T. E. Humphreys, J. A. Bhatti, A. J. Joplin, B. W. O'Hanlon, and S. P. Powell, "Demonstration of a space capable miniature dual frequency GNSS receiver," *Navigat. J. Inst. Navigat.*, vol. 61, no. 1, pp. 53–64, 2014.

[32] S. Gleason, M. Quigley, and P. Abbeel, "A GPS software receiver," in *GNSS Applications and Methods*, S. Gleason and D. Gebre-Egziabher, Eds. Norwood, MA, USA: Artech House, 2009, ch. 5, pp. 121–146.

[33] S. A. Nik and M. Petovello, "Multichannel dual frequency GLONASS software receiver," in *Proc. 21st Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Sep. 2008, pp. 614–624.

[34] M. G. Petovello, C. O'Driscoll, G. Lachapelle, D. Borio, and H. Murtaza, "Architecture and benefits of an advanced GNSS software receiver," *J. Global Positioning Syst.*, vol. 7, no. 2, pp. 156–168, 2008.

[35] S. Jeon, H. So, H. No, T. Lee, and C. Kee, "Development of real-time software GPS receiver using windows visual C++ and USB RF front-end," in *Proc. Int. Tech. Meeting Inst. Navigat.*, San Diego, CA, USA, 2010, pp. 713–721.

[36] B. M. Ledvina, M. L. Psiaki, D. J. Sheinfeld, A. P. Cerruti, S. P. Powell, and P. M. Kintner, Jr., "A real-time GPS civilian L1/L2 software receiver," in *Proc. 17th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Long Beach, CA, USA, Sep. 2004, pp. 986–1005.

[37] F. Principe, C. Terzi, M. Luise, and M. Casucci, "SOFT-REC: A GPS/EGNOS software receiver," in *Proc. 14th IST Mobile Wireless Commun. Summit*, Dresden, Germany, 2005.

[38] C. Stöber *et al.*, "ipexSR: A real-time multi-frequency software GNSS receiver," in *Proc. ELMAR*, Zadar, Croatia, Sep. 2010, pp. 407–416.

[39] B. Huang, Z. Yao, F. Guo, S. Deng, X. Cui, and M. Lu, "STARx—A GPU based multi-system full-band real-time GNSS software receiver," in *Proc. 26th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Nashville, TN, USA, Sep. 2013, pp. 1549–1559.

[40] A. Knezevic, C. O'Driscoll, and G. Lachapelle, "Co-processor aiding for real-time software GNSS receiver," in *Proc. Int. Tech. Meeting Satellite Division Inst. Navigat. (ITM)*, San Diego, CA, USA, vol. 2, 2010, pp. 667–678.

[41] G. Heinrichs, M. Restle, C. Dreischer, and T. Pany, "NavX–NSR–a novel Galileo/GPS navigation software receiver," in *Proc. 20th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Fort Worth, TX, USA, 2007, pp. 1329–1334.

[42] T. E. Humphreys, M. Murrian, and L. Narula, "Low-cost precise vehicular positioning in urban environments," in *Proc. IEEE/ION PLANS*, Monterey, CA, USA, Apr. 2018, pp. 456–471.

[43] J. Dampf *et al.*, "More than we ever dreamed possible: Processor technology for GNSS software receivers in the year 2015," *Inside GNSS*, vol. 10, no. 4, pp. 62–72, 2015.

[44] P.-L. Normark and C. Ståhlberg, "Hybrid GPS/Galileo real time software receiver," in *Proc. 18th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Long Beach, CA, USA, Sep. 2005, pp. 1906–1913.

[45] L. L. Presti, P. di Torino, E. Falletti, M. Nicola, and M. T. Gamba, "Software defined radio technology for GNSS receivers," in *Proc. Metrol. Aerosp. (MetroAeroSpace)*, Benevento, Italy, May 2014, pp. 314–319.

[46] A. Soghoyan, A. Suleiman, and D. Akopian, "A development and testing instrumentation for GPS software defined radio with fast FPGA prototyping support," *IEEE Trans. Instrum. Meas.*, vol. 63, no. 8, pp. 2001–2012, Aug. 2014.

[47] K. Borre, D. M. Akos, N. Bertelsen, P. Rinder, and S. H. Jensen, *A Software-Defined GPS and Galileo Receiver: A Single-Frequency Approach*. Boston, MA, USA: Birkhäuser Verlag, 2007.

[48] F. Macchi and M. G. Petovello, "Development of a one channel Galileo L1 software receiver and testing using real data," in *Proc. 20th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, FortWorth, TX, USA, vol. 2, 2007, pp. 2256–2269.

[49] D. F. Cristaldi, D. Margaria, and L. L. Presti, "A multifrequency low-cost architecture for GNSS software receivers," in *Proc. Int. Tech. Meeting Satellite Division Inst. Navigat. (ITM)*, San Diego, CA, USA, vol. 2, 2010, pp. 679–687.

[50] J. S. Silva, P. F. Silva, A. Fernandez, J. Diez, and J. F. M. Lorga, "Factored correlator model: A solution for fast, flexible, and realistic GNSS receiver simulations," in *Proc. 20th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Fort Worth, TX, USA, 2007, pp. 2676–2686.

[51] Y. Ng and G. X. Gao, "GNSS multireceiver vector tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 5, pp. 2583–2593, Oct. 2017.

[52] C. Fernández-Prades, J. Arribas, and P. Closas, "GNSS-SDR: An open source tool for researchers and developers," in *Proc. 24th Int. Tech. Meeting Satellite Division Inst. Navigat.*, Portland, OR, USA, Sep. 2011, pp. 780–794.

[53] E. Schmidt, D. Akopian, and D. J. Pack, "Development of a real-time software-defined GPS receiver in a labVIEW-based instrumentation environment," *IEEE Trans. Instrum. Meas.*, vol. 67, no. 9, pp. 2082–2096, Sep. 2018, doi: 10.1109/TIM.2018.2811446.

[54] E. Schmidt and D. Akopian, "Exploiting acceleration features of labVIEW platform for real-time GNSS software receiver optimization," in *Proc. 30th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Portland, OR, USA, Sep. 2017, pp. 3694–3709.

[55] F. Principe, G. Bacci, F. Giannetti, and M. Luise, "Software-defined radio technologies for GNSS receivers: A tutorial approach to a simple design and implementation," *Int. J. Navigat. Observ.*, vol. 2011, Feb. 2011, Art. no. 979815, doi: 10.1155/2011/979815.

[56] J.-H. Won, T. Pany, and G. W. Hein, "GNSS software defined radio: Real receiver or just a tool for experts?" *Inside GNSS*, vol. 1, no. 5, pp. 48–56, Aug. 2006.

[57] D. Akopian, "Fast FFT based GPS satellite acquisition methods," *IEE Proc.-Radar, Sonar Navigat.*, vol. 152, no. 4, pp. 277–286, Aug. 2005.

[58] E. D. Kaplan and C. Hegarty, *Understanding GPS: Principles and Applications*. Boston, MA, USA: Artech House, 1996.

[59] T. E. Humphreys, J. A. Bhatti, T. Pany, B. M. Ledvina, and B. W. O'Hanlon, "Exploiting multicore technology in software-defined GNSS receivers," in *Proc. ION GNSS Meeting*, Savannah, GA, USA, 2009, pp. 326–338.

[60] The GNU Radio Foundation. *GNU Radio*. Accessed: Jan. 2015. [Online]. Available: https://www.gnuradio.org

[61] E. Falletti, B. Motella, and M. T. Gamba, "Post-correlation signal analysis to detect spoofing attacks in GNSS receivers," in *Proc. 24th Eur. Signal Process. Conf. (EUSIPCO)*, Budapest, Hungary, Aug./Sep. 2016, pp. 1048–1052.

[62] A. Broumandan, A. Jafarnia-Jahromi, and G. Lachapelle, "Spoofing detection, classification and cancelation (SDCC) receiver architecture for a moving GNSS receiver," *GPS Solutions*, vol. 19, pp. 475–487, Jul. 2015.

[63] A. Ranganathan, H. Ólafsdóttir, and S. Capkun, "SPREE: A spoofing resistant GPS receiver," in *Proc. 22nd Annu. Int. Conf. Mobile Comput. Netw.*, New York, NY, USA, Oct. 2016, pp. 348–360.

[64] M. Paonni *et al.*, "Innovative interference mitigation approaches: Analytical analysis, implementation and validation," in *Proc. 5th ESA Workshop Satellite Navigat. Technol. Eur. Workshop GNSS Signals Signal Process. (NAVITEC)*, Noordwijk, The Netherlands, Dec. 2010, pp. 1–8.

[65] A. Jafarnia-Jahromi, T. Lin, A. Broumandan, J. Nielsen, and G. Lachapelle, "Detection and mitigation of spoofing attacks on a vector based tracking GPS receiver," in *Proc. Int. Tech. Meeting Inst. Navigat. (ION ITM)*, Newport Beach, CA, USA, Jan. 2012, pp. 790–800.

[66] Y. Ng and G. X. Gao, "Mitigating jamming and meaconing attacks using direct GPS positioning," in *Proc. IEEE/ION Position, Location Navigat. Symp. (PLANS)*, Savannah, GA, USA, Apr. 2016, pp. 1021–1026.

[67] Y. Ng and G. X. Gao, "Robust GPS-based direct time estimation for PMUs," in *Proc. IEEE/ION Position, Location Navigat. Symp. (PLANS)*, Savannah, GA, USA, Apr. 2016, pp. 472–476.

[68] E. Schmidt, Z. A. Ruble, D. Akopian, and D. J. Pack, "A reduced complexity cross-correlation interference mitigation technique on a real-time software-defined radio GPS L1 receiver," in *Proc. IEEE/ION PLANS*, Monterey, CA, USA, Apr. 2018, pp. 931–939.

[69] W. L. Myrick, M. Picciolo, J. S. Goldstein, and V. Joyner, "Multistage anti-spoof GPS interference correlator (MAGIC)," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Tampa, FL, USA, Oct. 2015, pp. 1497–1502.

[70] T. E. Humphreys, "Detection strategy for cryptographic GNSS anti-spoofing," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 2, pp. 1073–1090, Apr. 2013.

[71] T. E. Humphreys, B. M. Ledvina, M. L. Psiaki, B. W. O'Hanlon, and P. M. Kintner, Jr., "Assessing the spoofing threat: Development of a portable GPS civilian spoofer," in *Proc. 21st Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Savannah, GA, USA, Sep. 2008, pp. 2314–2325.

[72] T. E. Humphreys, D. Shepard, J. A. Bhatti, and K. Wesson, "A testbed for developing and evaluating GNSS signal authentication techniques," in *Proc. Int. Symp. Certif. GNSS Syst. Serv. (CERGAL)*, Dresden, Germany, Jul. 2014.

[73] B. W. O'Hanlon, M. L. Psiaki, J. A. Bhatti, D. P. Shepard, and T. E. Humphreys, "Real-time GPS spoofing detection via correlation of encrypted signals," *Navigation*, vol. 60, no. 4, pp. 267–278, 2013.

[74] M. L. Psiaki, B. W. O'Hanlon, J. A. Bhatti, D. P. Shepard, and T. E. Humphreys, "GPS spoofing detection via dual-receiver correlation of military signals," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 49, no. 4, pp. 2250–2267, Oct. 2013.

[75] B. W. O'Hanlon, M. L. Psiaki, T. E. Humphreys, and J. A. Bhatti, "Real-time spoofing detection in a narrow-band civil GPS receiver," in *Proc. 23rd Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Portland, OR, USA, Sep. 2010, pp. 2211–2220.

[76] B. W. O'Hanlon, M. L. Psiaki, T. E. Humphreys, and J. A. Bhatti, "Real-time spoofing detection using correlation between two civil GPS receiver," in *Proc. 25th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Nashville, TN, USA, Sep. 2012, pp. 3584–3590.

[77] L. Heng, J. J. Makela, A. D. Domínguez-García, R. B. Bobba, W. H. Sanders, and G. X. Gao, "Reliable GPS-based timing for power systems: A multi-layered multi-receiver architecture," in *Proc. Power Energy Conf. Illinois (PECI)*, Feb./Mar. 2014, pp. 1–7.

[78] T. Humphreys, J. Bhatti, D. Shepard, and K. Wesson, "The texas spoofing test battery: Toward a standard for evaluating GPS signal authentication techniques," in *Proc. 25th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Nashville, TN, USA, Sep. 2012, pp. 3569–3583.

[79] S. Daneshmand, A. Jafarnia-Jahromi, A. Broumandan, and G. Lachapelle, "GNSS spoofing mitigation in multipath environments using space-time processing," in *Proc. Eur. Navigat. Conf. (ENC)*, Vienna, Austria, Apr. 2013.

[80] M. L. Psiaki, B. W. O'Hanlon, S. P. Powell, J. A. Bhatti, K. D. Wesson, and T. E. Humphreys, "GNSS spoofing detection using two-antenna differential carrier phase," in *Proc. 27th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Tampa, FL, USA, Sep. 2014, pp. 2776–2800.

[81] P. Y. Montgomery, T. E. Humphreys, and B. M. Ledvina, "Receiver-autonomous spoofing detection: Experimental results of a multi-antenna receiver defense against a portable civil GPS spoofer," in *Proc. Int. Tech. Meeting Inst. Navigat.*, Anaheim, CA, USA, Jan. 2009, pp. 2776–2800.

[82] S. Daneshmand, A. Jafarnia-Jahromi, A. Broumandan, and G. Lachapelle, "A GNSS structural interference mitigation technique using antenna array processing," in *Proc. IEEE 8th Sensor Array Multichannel Signal Process. Workshop (SAM)*, Coruna, Spain, Jun. 2014, pp. 109–112.

[83] S. Daneshmand, A. Jafarnia-Jahromi, A. Broumandon, and G. Lachapelle, "A low-complexity GPS anti-spoofing method using a multi-antenna array," in *Proc. 25th Int. Tech. Meeting Satellite Division Inst. Navigat. (ION GNSS)*, Nashville, TN, USA, Sep. 2012, pp. 1233–1243.

[84] A. Jafarnia-Jahromi, A. Broumandan, S. Daneshmand, N. Sokhandan, and G. Lachapelle, "A double antenna approach toward detection classification and mitigation of GNSS structural interference," in *Proc. 7th ESA Workshop Satell. Navigat. Technol. Eur. Workshop GNSS Signals Signal Process. (NAVITEC)*, Noordwijk, The Netherlands, Dec. 2014.

[85] C. Fernández-Prades, J. Arribas, and P. Closas, "Robust GNSS receivers by array signal processing: Theory and implementation," *Proc. IEEE*, vol. 104, no. 6, pp. 1207–1220, Jun. 2016.

[86] J. Arribas, C. Fernández-Prades, and P. Closas, "Multi-antenna techniques for interference mitigation in GNSS signal acquisition," *EURASIP J. Adv. Signal Process*, vol. 2013, no. 1, p. 143, Sep. 2013.

[87] J. Arribas, P. Closas, and C. Fernández-Prades, "Interference mitigation in GNSS receivers by array signal processing: A software radio approach," in *Proc. 8th IEEE Sensor Array Multichannel Signal Process. Workshop*, Coruña, Spain, Jun. 2014, pp. 121–124.

[88] Ettus Research, A National Instruments (NI) Company. *Ettus Research–The Leader in Software Defined Radio (SDR)*. Accessed: Jan. 2015. [Online]. Available: https://www.ettus.com

[89] Nottingham Scientific Ltd—STEREO GNSS FRONTEND. *GNSS SDR Front End and Receiver*. Accessed: Sep. 2018. [Online]. Available: https://www.nsl.eu.com/nsl-jcms/advanced-gnns-hw-sw/2016-04-11-15-01-35

[90] I. F. E. N. GmbH. *SX3 GNSS Software Receiver*. Accessed: Sep. 2018. [Online]. Available: https://www.ifen.com/products/sx3-gnss-software-receiver/

[91] Intel Corporation. *Intel(R) C++ Intrinsic Reference*. Accessed: Jun. 2017. [Online]. Available: https://software.intel.com/sites/default/files/a6/22/18072-347603.pdf
[92] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proc. IEEE*, vol. 93, no. 2, pp. 216–231, Feb. 2005.
[93] Eigen. *Eigen is a C++ Template Library for Linear Algebra: Matrices, Vectors, Numerical Solvers, and Related Algorithms*. Accessed: 2016. [Online]. Available: http://eigen.tuxfamily.org
[94] U. Madhow and M. L. Honig, "MMSE interference suppression for direct-sequence spread-spectrum CDMA," *IEEE Trans. Commun.*, vol. 42, no. 12, pp. 3178–3188, Dec. 1994.
[95] R. Kohno, H. Imai, M. Hatori, and S. Pasupathy, "An adaptive canceller of cochannel interference for spread-spectrum multiple-access communication networks in a power line," *IEEE J. Sel. Areas Commun.*, vol. 8, no. 4, pp. 691–699, May 1990.
[96] G. Lopez-Risueno and G. Seco-Granados, "CN_0 estimation and near-far mitigation for GNSS indoor receivers," in *Proc. 61st Veh. Technol. Conf. (VTC)*, Stockholm, Sweden, May/Jun. 2005, pp. 2624–2628.
[97] A. Dempster and E. Glennon, "Apparatus and method for mitigation of cross correlation in GPS system," U.S. Patent 20070058696 A1, Mar. 15, 2007.
[98] M. Honig, U. Madhow, and S. Verdu, "Blind adaptive multiuser detection," *IEEE Trans. Inf. Theory*, vol. 41, no. 4, pp. 944–960, Jul. 1995.
[99] L. Rugini, P. Banelli, and S. Cacopardi, "A full-rank regularization technique for MMSE detection in multiuser CDMA systems," *IEEE Commun. Lett.*, vol. 9, no. 1, pp. 34–36, Jan. 2005.
[100] G. H. Golub, and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD, USA: Johnson Hopkins Univ. Press, 1996.
[101] National Instruments. *NI Global Navigation Satellite System Toolkits National Instruments*. Accessed: 2016. [Online]. Available: http://sine.ni.com/nips/cds/view/p/lang/en/nid/204980
[102] M. A. Abu-Rgheff, "Pseudo-random code sequences for spread-spectrum systems," in *Introduction to CDMA Wireless Communications*. London, U.K.: Academic, 2007, pp. 203–220.

Erick Schmidt (S'17) received the B.S. degree (Hons.) in electronics and computer engineering from The Monterrey Institute of Technology and Higher Education, Monterrey, Mexico, in 2011, and the M.S. degree from The University of Texas at San Antonio, San Antonio, TX, USA, in 2015, where he is currently pursuing the Ph.D. degree in electrical engineering.

From 2011 to 2013, he was a Systems Engineer with Qualcomm Incorporated, San Diego, CA, USA. His current research interests include software defined radio, indoor navigation, global navigation satellite system, and fast prototyping algorithms and accelerators for baseband communication systems.

Mr. Schmidt is a Student Member of the Institute of Navigation.

Zachary Ruble received the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in electrical engineering from the University of Wyoming, Laramie, WY USA, in 2007, 2009, and 2015, respectively.

From 2010 to 2015, he was a Research Assistant with the Electrical and Computer Engineering Department, University of Wyoming. Since 2015, he has been with the Unmanned Systems Laboratory, University of Tennessee at Chattanooga, Chattanooga, TN, USA, where he is currently a Post-Doctoral Fellow. His current research interests include cooperative unmanned systems, robotics, satellite systems, navigation, and distributed control systems.

David Akopian (M'02–SM'04) received the Ph.D. degree from the Tampere University of Technology, Tampere, Finland.

He was a Senior Research Engineer and a Specialist with Nokia Corporation, Espoo, Finland, from 1999 to 2003. From 1993 to 1999, he was a Researcher and an Instructor with the Tampere University of Technology. He is currently a Professor with The University of Texas at San Antonio, San Antonio, TX, USA. He has authored or co-authored over 140 publications and holds 30 patents. His current research interests include digital signal processing algorithms for communication and navigation receivers, positioning, dedicated hardware architectures and platforms for software-defined radio and communication technologies for healthcare applications.

Dr. Akopian is elected as a fellow of the U.S. National Academy of Inventors in 2016. He served in organizing and program committees of many IEEE conferences and co-chairs an annual conference on Multimedia and Mobile Devices. His research has been supported by National Science Foundation, National Institutes of Health, USAF, U.S. Navy, and Texas foundations.

Daniel J. Pack received the degree in electrical engineering from Purdue University, West Lafayette, IN, USA, in 1995, and the B.S. degree in computer engineering and the M.S. and Ph.D. degrees in electrical engineering from the University of Wyoming, Laramie, WY, USA, in 2007, 2009, and 2015, respectively.

From 2010 to 2015, he was a Research Assistant with the Electrical and Computer Engineering Department, University of Wyoming. Since 2015, he has been with the Unmanned Systems Laboratory, University of Tennessee at Chattanooga, Chattanooga, TN, USA, where he is currently a Post-Doctoral Fellow and the Dean with the College of Engineering and Computer Science. He was a Professor and the Mary Lou Clarke Endowed Chair with the Electrical and Computer Engineering Department, University of Texas at San Antonio, San Antonio, TX, USA, and a Professor Emeritus of electrical and computer engineering with the United States Air Force Academy, Air Force Academy, CO, USA, where he was the founding Director of the Academy Center for Unmanned Aircraft Systems Research, Department of Electrical and Computer Engineering. His current research interests include cooperative unmanned systems, robotics, embedded systems, distributed control systems, unmanned aerial vehicles, intelligent control, automatic target recognition, robotics, and engineering education.