

Received June 30, 2021, accepted July 26, 2021, date of publication August 13, 2021, date of current version August 24, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3104738

Drone Detection Sensor With Continuous 2.4 GHz ISM Band Coverage Based on Cost-Effective SDR Platform

PRZEMYSŁAW FLAK 

Institute of Automatic Control, Faculty of Automatic Control, Electronics and Computer Science, Ph.D. School, Silesian University of Technology, 44-100 Gliwice, Poland

e-mail: przemyslaw.flak@polsl.pl

This work was supported by the Ministry of Education and Science of Poland under Grant DWD/4/21/2020-00375/003.

ABSTRACT The development of Unmanned Aerial Vehicles (UAVs), commonly referred to as drones, has introduced revolutionary changes in many areas over the past few years. However, aside from opening new possibilities, the usage of drones in an irresponsible and dangerous manner leads to many hazardous incidents. This paper presents a drone detection sensor with a continuous 2.400 GHz-2.483 GHz operational frequency range for detection methods based on passive radio frequency imaging techniques. The implementation based on Software Defined Radio (SDR) and Field Programmable Logic Array (FPGA) hardware that overcomes the 40 MHz real-time bandwidth limit of other popular SDRs is presented utilizing low-cost off-the-shelf components. Furthermore, a hardware realization of the signal processing chain for specific detection algorithms is proposed to minimize the throughput between SDR and the companion computer and offload software computations. The device validation is made in a laboratory and real-life scenario and presented in relation to the sensor used in other works. In addition to the increased real-time bandwidth, the measurements show a 9 dB reduction in detection sensitivity compared to the reference receiver, in line with the analog RF front-end specifications. The final analysis demonstrates the proposed device's relevance as a sensor for obtaining machine learning datasets and as a part of a final anti-drone system.


INDEX TERMS Drones, field programmable gate array, software defined radio, surveillance, unmanned aerial vehicles.

I. INTRODUCTION

Unmanned Aerial Vehicles (UAVs) were initially referred to as military reserved technology restricted for tactical use. However, within a few years, the growing popularity of smartphones, widespread navigation systems availability, technological advancements in communication and imaging established a significant growth of commercial drone market size, which is expected to generate a value of EUR 10 billion per year by 2035 [1]. Although giving new possibilities in various sectors such as precision agriculture [2], energy [3], cinematography [4], construction inspection [5], and even package delivery [6], the rising demand for aerial services affects not only the safety but also the privacy of people. Apart from highly hazardous events like drone incursions into nuclear power plants [7], aerodrome operations holdups [8],

or public service disruptions [9], everyday minor accidents associated with drones also occur. For example, the reputation of drones has been compromised by irresponsible pilots flying them to harass or stalk other people [10]. According to the FAA UAS Sightings Report, incidents involving drones average about 100 a month [11].

The above examples indicate the need to develop drone detection systems not only for highly professional use but also for widespread general applications. The Author's primary motivation was to consider the adaptability of cost-effective devices, based on Commercial-off-the-Shelf (COTS) components, for automatic drone detection. This study concentrates on monitoring the over-the-air communication between the operator and the drone. The aim of the article is to investigate the aspect of the sensor used for this task and propose the solution based on Software Defined Radio (SDR), rather than to evaluate the effectiveness of various detection methods. While other surveys pay attention to algorithms for drone

The associate editor coordinating the review of this manuscript and approving it for publication was Zhenbao Liu .

type identification, this work focuses on the hardware implementation of the signal processing chain to offload software computations.

The paper is organized as follows: the related work is discussed in Section 2. The detection sensor is presented in Section 3, and details of implementation are provided in Section 4. Finally, experimental results are outlined in Section 5, whereas conclusions and future plans are noted in Section 6.

II. RELATED WORK

In the literature, several sensing techniques based on different physical phenomena were proposed for drone detection. They include sound and electromagnetic waves or video image analysis in both visible and infrared spectrums. Each of these methods has its limitations. Therefore systems with heterogeneous sensors and data fusion algorithms were widely presented to meet real-life demands [12]. Based on the drone's propellers sound distinction, the audio approach is affected by noisy environments, especially in urban areas [13]. Video detection, usually used together with machine learning algorithms, requires high-resolution cameras and high-quality optics to work well in poor light conditions, which has a significant cost impact [14]. The radar approach, both passive [15] and active [16], is the most complex and expensive solution, although it can reach long distances. However, ever smaller drone sizes, low altitude operation, low flying speed, and the urban environment highly reduce the radar detection accuracy. Considering these limitations, the Authors in [17] investigate the micro-Doppler signatures induced by the micro-motion of small drones. While effectiveness against drones performing autonomous tasks is a valuable asset, high power active radar beam, needed for long-range detection, usually limits its use to licensed users.

The last method is passive Radio Frequency (RF) sensing of drone communication signals based on characteristic feature distinction [18]. Whereas the performance of this approach suffers from coexistence with WiFi, Bluetooth, or ZigBee transmissions, it has the incontestable advantage of being license-free and ambient environmental conditions independent. This feature allows the use of an almost unlimited number of sensors to build a distributed system. One of the passive RF detection techniques is the Media Access Control (MAC) layer statistical fingerprint analysis. The main system component, in this case, is the WiFi packet sniffing device intercepting over-the-air communication between the drone and the remote controller [19]. Despite good efficiency and vast commercial receivers availability, the method remains outside the scope of this study because of the minor WiFi controlled devices market share and limited range of operation compared to non-WiFi driven drones. Therefore, this work focuses on the other RF physical layer spectrum sensing principle, utilizing SDR platforms. The disadvantage of this technique over MAC is that more detailed information for further drone type classification can be obtained by pocket sniffing. However, the blind scan approach opens extensive

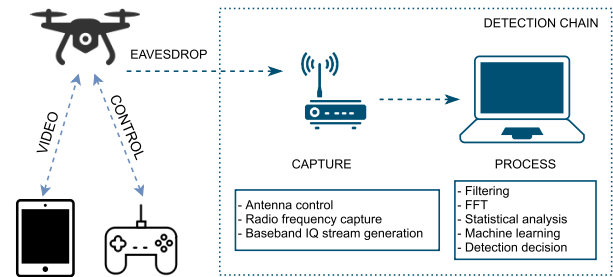


FIGURE 1. Passive radio frequency drone detection scheme. Over-the-air signals between the drone and its controller are intercepted by an RF front-end device, transferred to the PC, and post-processed in the software.

possibilities for the evolution of various detection algorithms. It is generally more important in hazardous scenarios to detect the intrusion itself than precisely classify the drone model in the Author's opinion.

Wireless transmissions of commercially available drones occupy an unlicensed Industrial, Scientific, and Medical (ISM) frequency band. Most models communicate with the remote controller employing the Frequency Hopping Spread Spectrum (FHSS) technique in the 2.400 GHz to 2.483 GHz frequency range. Video signal is transmitted over the same frequency band or in the 5.725 GHz to 5.850 GHz range, using the Orthogonal Frequency Division Multiplexing (OFDM) technique. A receiver with 83 MHz of instantaneous bandwidth is required to provide a continuous 2.4 GHz ISM spectrum overview and identify prominent RF signal features. The basic setup to intercept drone transmission consists of an antenna, RF receiver, and a processing unit. Recorded data are usually post-processed and further analyzed in software, using either a frequency domain or time domain approach. The generic hardware arrangement is presented in Fig. 1.

The RF capture device can be based on high frequency oscilloscope [18], spectrum analyzer [20] or SDR platform [21]–[24]. Although both, oscilloscope and spectrum analyzer, offers high accuracy and fulfills bandwidth requirement, they are more intended for laboratory use. Moreover, high power consumption, high cost, and size exclude the possibility of building a distributed surveillance system based on them. On the other hand, solutions utilizing inexpensive SDR suffer from the limited 40 MHz bandwidth [23], allowing to observe only half of the needed spectrum in real-time. The Authors in [23], [24] used dual SDRs running simultaneously to obtain a machine learning dataset and overcome this limit. Some methods based on statistical signatures like skewness, kurtosis, and standard deviation, can be applied to a small band of the signal in the frequency domain [25]. Nevertheless, the full-band approach enhances spectrogram dataset accuracy and opens up new opportunities for detection algorithms. In addition, the methodology described in [20], based on Short Time Fourier Transform (STFT), cannot be applied without observing the entire 2.400 GHz to 2.483 GHz frequency range continuously. There are some

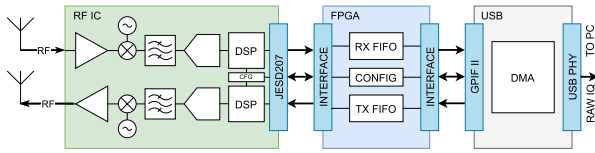


FIGURE 2. Block diagram of the classic SDR configuration with a color marking of individual on-board chips and corresponding interfaces. In this concept, raw IQ baseband signal representation is transferred into PC and further processed in the software to obtain a spectral estimator.

high-performance SDRs on the market that offers 120 MHz instantaneous bandwidth and PCIe or 10 GigE interface [26]. However, they cannot be considered cost-effective, which is one of the main aspects of this work.

III. PROPOSED METHOD

A basic inexpensive SDR hardware architecture is presented in Fig. 2. The RF front-end integrates analog and digital signal chain elements like mixers, filters, ADC, and DAC, converting the RF signal down to the baseband. In addition, Field Programmable Gate Array (FPGA) is responsible for configuration, data buffering, and interface translation between the front-end and USB controller.

In this concept, In-Phase and Quadrature components (IQ) baseband signal representation is transferred into PC and post-processed in the software by the central processing unit (CPU) or graphical processing unit (GPU). The IQ stream transfer rate varies depending on the received signal bandwidth and the number of channels set in SDR to operate. For example, widely used in the articles mentioned in the previous section USRP B210 SDR [27], using a single channel, receive-only mode, generates around 160 MB/s IQ stream. This transfer rate stays below USB 3.0 maximum transfer capability but still states a considerable amount of data to receive and process in real-time. A solution proposed in this paper is developed by upgrading the firmware of the COTS SDR platform containing analog front-end capable of continuous 2.4 GHz ISM band spectrum sensing. Furthermore, some signal processing algorithms are also implemented in the FPGA to offload software computations and further decrease transfer rate, allowing even embedded system processors to analyze data in real-time. The RF front-end [28] that can continuously acquire even over 83 MHz instantaneous bandwidth is utilized by the LimeSDR-USB board [29]. However, the specification defines the maximum operating bandwidth as 61.44 MHz because of the USB controller transfer limit. This value defines the transfer speed as 184 MB/s, calculated for a 12-bit I and Q sample without protocol overhead included. The SDR features and classical usage of signal processing in GNU Radio software were presented in [30]. LimeSDR-USB is a part of the Open Source initiative, thus some approaches to modify the original firmware were already made. For example, the hardware FPGA accelerator was proposed in [31] to increase the SDR possibilities in terms of bandwidth. However, this implementation was created with a Python-to-HDL language converter and did not give space for future improvements due

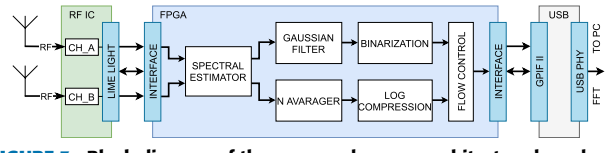


FIGURE 3. Block diagram of the proposed sensor architecture based on classic SDR with extended FPGA contribution. A signal processing chain for the spectral estimator was implemented in the FPGA fabric to minimize detection latency and USB transfer rate.

to limited FPGA resources located on LimeSDR-mini used in a project. The aim of this work is to present a spectrum sensor with extensive signal processing features, especially for drone detection purposes. Therefore, the current implementation was made from scratch for the evaluation board with more extensive FPGA resources.

The simplified architecture of the proposed sensor that provides direct data input for drone detection methods proposed in [20], [22]–[24] is demonstrated in Fig. 3. The minimum data rate of a dual 12-bit I and Q stream to acquire 83 MHz bandwidth can reach around 250 MB/s, which is beyond the initial 184 MB/s limit. Fast Fourier Transform (FFT) calculation module and initial filtering arranged inside FPGA hardware can decrease the transfer rate demand to the appropriate level. Rather than send the separate time domain I and Q components to the PC, the proposed sensor delivers one 12-bit formatted frequency domain sample. A bitstream throughput is reduced by half with this approach, and the produced data can be applied in detection methods based on machine learning or spectrogram features proposed by [22]–[24]. The data rate could be reduced even further, to about 10 MB/s, by employing the adapted binarization module. This mode can be applied in the STFT detection approach presented in [20], where only features related to timing aspects are considered for detection. Finally, mode selection was implemented to stream the spectral estimator in a 12-bit averaged or 1-bit binarized format to satisfy various situations. Still, the protocol provided by the SDR manufacturer was preserved in both cases, thus minimal effort is needed to receive and further process samples in the software of a companion computer. The individual modules are discussed in more detail in the following subsections.

A. RF FRONT-END

The LMS7002M [28] integrated circuit is a software configurable RF transceiver providing a continuous operation frequency range of 100 kHz to 3.8 GHz and up to 96 MHz modulation bandwidth using a digital interface. It integrates 12-bit ADC, DAC, and on-chip microcontroller to further simplify calibration and integration. The device is fabricated using a low-cost CMOS process, thus ideally fulfills this work assumption. Additionally, it features a single power rail supply and the capability of individual block power down. Therefore, some energy savings could be made in the receiver-only application. Although the device supports dual independent receive paths, they cannot be used simultaneously to achieve full real-time bandwidth capability. Instead, dual paths can be

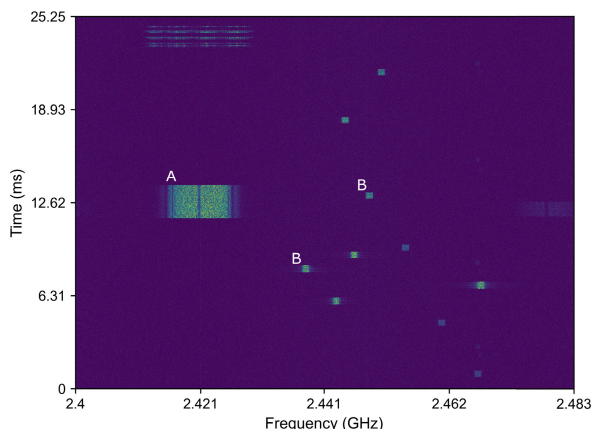


FIGURE 4. Sample spectrogram of 2.4 GHz ISM band including (a) WiFi, and (b) Bluetooth signal. Characteristic signal features such as bandwidth, frame duration, and hopping pattern are visible through full-band real-time sampling.

utilized to connect a directional and omnidirectional antenna simultaneously, and switch between them in the FPGA fabric.

B. SPECTRAL ESTIMATOR

The principal of the proposed sensor is to provide a hardware implementation of a spectrum estimator to accelerate software time-frequency domain analysis of drone communication signals. The spectrogram is obtained by the STFT separating constant input data stream into segments of equal length and computing Fourier Transform on every segment. The discrete form of the STFT can be expressed as:

$$STFT \equiv X(m, \omega) = \sum_{n=-\infty}^{\infty} x[n]w[n - m]e^{-j\omega n}, \quad (1)$$

where $x[n]$ states for the received signal, $w[n]$ indicates the window function, n represents the sample number, and m is the position of the analysis window. An exemplary spectrogram including Bluetooth signal in the presence of WiFi is demonstrated in Fig. 4.

The window length was selected to follow the requirements suggested in [20], [24] as 2048 samples. The characteristics of the window shape were chosen to suit the current situation and purpose. In this scenario, there is a need to accurately measure the duration of the frequency hopping drone communication signal. To accurately detect transient signals such as the start or end of the frame, it was better not to use the spectral windows because many windows attenuate vital information at the edges of the sample frame. According to [32], this is explained by Parseval’s Theorem about the power level produced by the discrete transform of a coherent signal and can be noted as follows:

$$Power(dB) = 20 \log_{10} \frac{SignalSamples}{TotalSamples}. \quad (2)$$

Therefore, a uniform window was used to improve the accuracy of the measurement of time events in return for some spectrogram artifacts. Furthermore, simplicity made it suitable for FPGA architecture.

The 2048 point FFT module, using 12-bit time domain I and Q input samples, produces a dual 24-bit frequency domain output stream due to internal multiplication by twiddles and butterfly computations. Squared magnitude application to calculate energy extended bit-width even further, and square root implementation to reduce it was very resource consuming. Approximate arithmetic can reduce the hardware cost in error-tolerant applications such as digital signal processing, pattern recognition, and machine learning [33], therefore several methods were used in this paper. Apart from the classical energy estimate, the final stage of this module employs the approximation of the complex number modulus for signal magnitude M calculation expressed as:

$$M = \sqrt{a^2 + b^2} \approx \alpha \max(a, b) + \beta \min(a, b). \quad (3)$$

The various α and β options were evaluated and presented in terms of approximation errors in [34]. The hardware-efficient version, which uses the coefficients $\alpha = 15/16$ and $\beta = 15/32$ to replace the multiplication with bit shift and addition operations, provides a mean error of 3.1%. Nevertheless, at the cost of some additional resources, the accuracy can be improved by the dual coefficient approach [35], according to condition:

$$\alpha, \beta = \begin{cases} \alpha = \frac{7}{8}, \beta = \frac{1}{2}, & \min(a, b) > \frac{1}{4} \max(a, b), \\ \alpha = 1, \beta = 0, & otherwise. \end{cases} \quad (4)$$

This algorithm provides a maximum error of 3.0% and a mean error of 0.95%, still without multiplier utilization.

As the FFT module offers digit-reverse mode by default, to prepare data for post-processing, the output data order was rearranged around the center frequency. Finally, to lower the bitrate and reduce the spectrogram variance, a non-overlapping ensemble averaging module was also implemented [36]. The user can optimize the averaging parameter during operation to obtain suitable spectral estimates for different drone detection approaches. A value of 1 means that the module is bypassed, whereas the value of 16 gives the additional output bitrate reduction for USB streaming.

C. LOGARITHM COMPRESSION

In order to preserve the original data format for the embedded post-processing unit, the 24-bit magnitude dynamic range was compressed to 12-bit per sample value. The use of a logarithmic scale representation is common for spectrum sensing purposes. Therefore, several hardware approximation methods were considered to implement a 4-bit integer and 8-bit fractional (4.8) data format. Coordinated rotation digital computer (CORDIC) algorithm and Newton-Raphson approach were rejected due to extensive hardware resource utilization and iterative nature. On the other hand, the Taylor series polynomial approximation requires many series multiplication and addition operations that have to be pipelined to achieve the high-speed design. The Look-Up-Table (LUT) method offers the highest computation speed comparing to the previous ones, in exchange for storage utilization [37].

Science memory requirement increases exponentially with accuracy improvement, Piecewise Linear (PWL) methods that only require linear segments to approximate the target function were considered. In this case, hardware implementation is based on several parallel comparators, single multiply addition operation, storage of slope, intercept, and endpoints for each segment. In [38], with 15 segments, a mean approximation error of 1.2×10^{-4} is achieved to satisfy the 12-bit fractional fixed-point format. However, the available representation in the current design has an 8-bit fractional length that corresponds to 3.9×10^{-3} level accuracy.

A method that combines the piecewise linear approximation and small LUT-based error correction was the most effective technique for the current design due to the versatility of error management regarding the available bit-width. Hardware efficient approach introduced in [39], where the slope coefficients are chosen to be realized using only a bit shift operations, proposes the approximation of the fundamental logarithmic function $L_a = \log_2(1 + x)$ in intervals expressed as:

$$L_a \approx \begin{cases} (2^0 + 2^{-2} + 2^{-5})x + 0.005, & [0, 0.25), \\ (2^0 + 2^{-5})x + 0.0684, & [0.25, 0.5), \\ (2^{-1} + 2^{-1} + 2^{-3})x + 0.1505 & [0.5, 0.75), \\ (2^{-1} + 2^{-2})x + 0.248, & [0.75, 1). \end{cases} \quad (5)$$

This method provides the mean approximation error of 2.5×10^{-3} before and 9.8×10^{-5} after LUT correction. Still, even without correction, the accuracy of the combined method is below the current system ± 1 LSB value. Due to flexibility in case of future accuracy improvements, this approach was selected to represent the spectrum estimator according to the logarithm properties determined as follows:

$$10 \log_{10}(M^2) = \frac{20 \log_2(M)}{\log_2(10)} \approx 6.02 \log_2(M), \quad (6)$$

where M denotes the sample magnitude, squared to obtain the spectral power, thus giving a proportionality factor of 6.02 to be considered in the software of the post-processing unit.

D. NOISE FILTERING

Numerous algorithms have been developed to solve the problem of noise removal [40]. As the pre-processing stage for the thresholding unit, spectrogram smoothing is performed to minimize the false alarm detection rate. Detected signal edge preservation is significant for this purpose in the aspect of losing valuable information about occupied bandwidth and exact signal duration. Various kernel sizes and filter architectures were simulated based on real-life data captured from the previous stage. Comparison of spectrogram thresholding results, using the Otsu’s method [41], after applying different filters is shown in Fig. 5.

An exemplary test spectrogram is presented as a grayscale image, where the x-axis represents the frequency bin, the y-axis denotes time, and the intensity reflects signal magnitude. After thresholding, a simple cv.findContours()

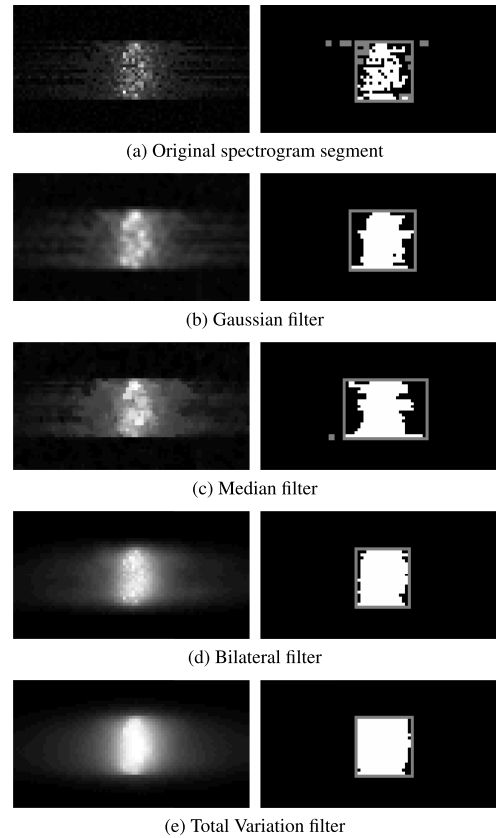


FIGURE 5. Various filtering algorithms applied to the spectrogram segment and the corresponding results of the thresholding process. Minimum bounding rectangle generated in software with the OpenCV library.

method from the OpenCV library [42] was employed in the software of a post-processing unit to test the filtering quality. In this case, finding contours can be explained as defining the rectangle bounding the white object on the black background. The results of these experiments are presented in Table 1.

Total Variation filter is an effective solution for edge preservation and smoothing out noise in uniform regions, even at low signal-to-noise ratios [43]. However, the iterative nature of this filter makes it impractical for the FPGA pipeline architecture. Bilateral filtering, proposed by Tomasi [44] as a non-iterative scheme for edge-preserving smoothing, was also considered. An FPGA implementation based on this fundamental idea was proposed in [45]. However, a simple Gaussian filter performed well in simulations in terms of signal duration detection, and offered higher resource utilization efficiency due to the multiplier-less design possibility. Furthermore, no false detection as with the median filter was observed. Finally, to remove high-frequency noise, minimize edge blur, and keep the hardware implementation simple, a Gaussian filter with a kernel size of 3×3 and Sigma equal to 1.0 was chosen.

E. BINARIZATION

Binarization is generally the process of converting a multi-tone input signal or image into a bi-tonal output. In the

TABLE 1. The OpenCV cv.findContours() detection results for thresholded spectrogram images from Fig.5.

Filter type	False detection	Signal duration	Signal bandwidth
None	10 counts	20 FFT windows	19 FFT bins
Gaussian	0 counts	20 FFT windows	20 FFT bins
Median	1 counts	20 FFT windows	28 FFT bins
Bilateral	20 counts	20 FFT windows	18 FFT bins
TV filter	20 counts	20 FFT windows	19 FFT bins

scope of this work, this is a process of comparing the energy of the received sample, as the squared magnitude of FFT bins, with a threshold to determine the signal presence. In this case, the output data is represented by one bit per frequency bin, thus the overall bitstream is reduced by a factor of 12.

Assuming that H_0 states for energy related to the noise-only region, and H_1 for signal plus noise zone of the spectrum, the sensing decision θ is given by:

$$\theta = \begin{cases} H_1, & \text{for } E \geq \lambda, \\ H_0, & \text{for } E < \lambda, \end{cases} \quad (7)$$

where, if the energy of the frequency bin E is higher than the threshold λ , the observed sample is considered as signal and noise, otherwise classified as noise only.

The general signal thresholding variations analyzed in the literature are fixed and adaptive. Although implementation simplicity, the performance of the fixed approach suffers from fluctuating background noise, especially in the congested ISM band. A fixed threshold is established once in advance and does not recalculate under dynamic spectrum variations. As the proposed sensor should provide signal detection in various environments, the adaptive technique was chosen for implementation. This methodology can be further divided into local, that computes different threshold value for each sample in the spectrum, and global, applied for a single spectral window. A comparative analysis of local and global adaptive threshold estimation is discussed in detail in [46]. It is not the purpose of this paper to evaluate the thresholding methods' properties. Thus, one of the most recent and interesting works, introducing the Non-Parametric Amplitude Quantization Method (NPAQM) [47], was adapted in this paper and altered for FPGA implementation. The approach proposed in the cited work is based on the theoretical basis of the popular Otsu's thresholding method, and further improved with the heuristic algorithm to enhance the performance under noise-only regimes. Threshold calculation simulations, summed up in Fig. 6, align with those presented in [47].

F. NPAQM METHOD

In this section, the principle of the Non-Parametric Amplitude Quantization Method introduced in [47] is presented. The energy sample set obtained by FFT of length F , sorted in an ascending order to fulfill $y_{(1)} < y_{(2)} < \dots < y_{(F)}$, is used as algorithm input and written as:

$$Y_f = \{y_1, y_2, \dots, y_F\}. \quad (8)$$

In order to minimize the calculation effort, the quantization of the magnitude range is proposed to determine the best

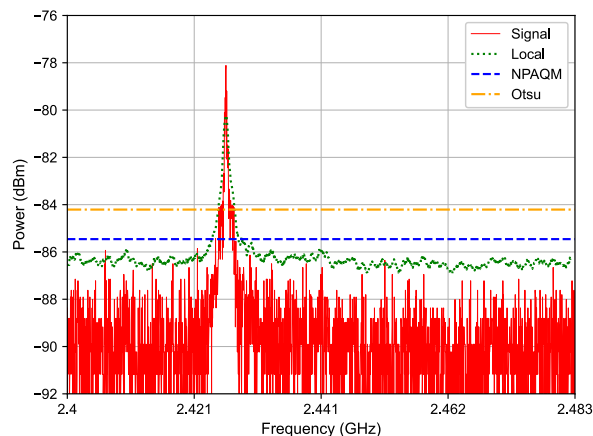


FIGURE 6. Comparison of a different threshold calculation methods. Novel NPAQM algorithm was chosen for implementation due to the best simulation results and hardware adaptability.

choice for the threshold candidate. The quantization level M , using ceiling function $\lceil \cdot \rceil$, is calculated as:

$$M = \lceil 1 + \log_2 F + \log_2(1 + \frac{|g|}{\sigma_g}) \rceil, \quad (9)$$

where $|g|$ refers to the modulus function of distribution skewness obtained as:

$$g = \frac{\frac{1}{F} \sum_{i=1}^F (Y_i - \bar{Y}_f)^3}{(\frac{1}{F} \sum_{i=1}^F (Y_i - \bar{Y}_f)^2)^{\frac{3}{2}}}, \quad (10)$$

and \bar{Y}_f denotes the mean of the input sample set Y_f , where:

$$\sigma_g = \sqrt{\frac{6(F - 2)}{(F + 1)(F + 3)}}. \quad (11)$$

The calculation step q , is expressed as:

$$q = \lceil \frac{y_{(F)} - y_{(1)}}{M} \rceil, \quad (12)$$

where $y_{(F)}$ and $y_{(1)}$ corresponds to $max(Y_f)$ and $min(Y_f)$ respectively, because the input data was sorted previously. At this point, the set of potential threshold candidates can be created as:

$$Y(m) = \{y_{(1)}, y_{(1)} + q, \dots, y_{(1)} + (M - 1)q\}, \quad (13)$$

and written in simplified form as:

$$\gamma_i = \gamma_1, \gamma_2, \dots, \gamma_M. \quad (14)$$

At this stage, according to the algorithm, the data is divided into two subsets: noise only subset $\omega(\gamma_i)$ and the signal plus noise subset $S(\gamma_i)$. Samples with values smaller than the threshold are assumed as noise-only and others as signal-plus-noise elements. The sets of $\omega(\gamma_i)$ and $S(\gamma_i)$ are determined as follows:

$$\omega(\gamma_i) = \{Y_f < \gamma_i\} = \{y_{(1)}, y_{(2)}, \dots, y_{(k)}\}, \quad (15)$$

$$S(\gamma_i) = \{Y_f \geq \gamma_i\} = \{y_{(k+1)}, y_{(k+2)}, \dots, y_{(F)}\}, \quad (16)$$

for all elements in (14), where k is the number of elements smaller than γ_i . To compute the between class variance of

both subsets for each threshold element included in (14) calculation is performed as:

$$\sigma^2(\gamma_i) = P_s(\gamma_i)P_\omega(\gamma_i)[\bar{S}(\gamma_i) - \bar{\omega}(\gamma_i)]^2, \quad (17)$$

where $P_s(\gamma_i)$ is the probability of signal elements and $\bar{S}(\gamma_i)$ is the mean of the subset (16), $P_\omega(\gamma_i)$ is the probability of noise elements and $\bar{\omega}(\gamma_i)$ is the mean of the subset (15). The probabilities can be defined as:

$$P_s(\gamma_i) = \frac{k}{F}, \quad P_\omega(\gamma_i) = 1 - \frac{k}{F}, \quad (18)$$

where k is the number of elements in each subset. Next, the first-order difference function is applied to (17) in range $m = 1, 2, \dots, M - 1$ expressed as:

$$\lambda_m(\gamma_i) = |\sigma_{m+1}^2(\gamma_i) - \sigma_m^2(\gamma_i)|. \quad (19)$$

The value that minimizes (19) indicates the optimal threshold, what can be written as:

$$\lambda(\gamma^{eff}) = \min \lambda(\gamma_i), \text{ for } \gamma_i = \gamma_1, \gamma_2, \dots, \gamma_M. \quad (20)$$

As a complement to this method, a novel heuristic algorithm, employing the degree of closeness between the estimated threshold $\lambda(\gamma^{eff})$ the mean of the entire sample set Y_f , was also proposed in [47] to improve the performance in noise-only regimes. The heuristic indicates that the spectrum set consists of noise-only elements when the threshold ceiling is less or equal to the mean ceiling, and the difference of mean and threshold is smaller than 10% of the entire sample range.

IV. HARDWARE IMPLEMENTATION

As a starting point to perform the FPGA structure modification, open-source code from LimeSDR-USB manufacturer was utilized [48]. All the subsystem parts related to the transmit path were wiped out from the original project to provide as much space for current system features as possible. Additionally, as only LimeLight interface to RF front-end in double data rate mode provides 96 MHz bandwidth operation, other options handlers were also removed. However, a USB controller having a FIFO input interface was left unchanged for seamless integration with post-processing software. The block diagram of the proposed signal processing was shown before in Fig. 3. The hardware implementation of the essential modules is described below.

A. SPECTRAL ESTIMATOR

The spectrogram module implementation is based on the FFT IP core (altera_fft_ii, version 20.1) with pipeline processing using a fixed-point format. The signal chain is presented in Fig. 7. The throughput of this FFT IP core fulfills the sampling speed rate of the RF front-end, thus real-time spectrum with no blind time is obtained, utilizing a single transform module. The 12-bit IQ time domain input stream is converted to zero mean representation after windowing to decrease the impact of the DC component. At the end of the window, the possibility of changing the RF input channel by USB interface to take advantage of the dual antenna system option

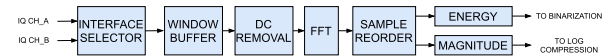


FIGURE 7. Block diagram of the signal processing chain in the spectrogram module. All modules were designed in a pipeline architecture.

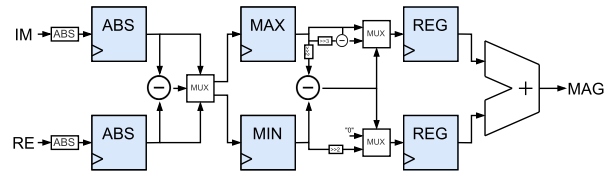


FIGURE 8. Register transfer level diagram of complex number magnitude estimator.

is provided. Followed by magnitude approximation, a 24-bit resolution stream is generated for the logarithm compression path and a 32-bit energy sample stream for the binarization path.

The ‘‘Alpha max plus Beta min’’ algorithm implementation as a three-stage pipeline that processes FFT complex output without maximum operating frequency reduction is presented in Fig. 8. After the absolute value of the complex operands is computed, the sign bit of the subtraction result is used to control the multiplexer, which applies the maximum and minimum to appropriate registers. In the second step, the $\min > \max/4$ condition is checked, again by examining the sign bit of the subtraction result, and coefficient selection is performed. The processed maximum and minimum values are added up at the output to obtain the final magnitude approximation. Until this point, the order of the samples emerging from the FFT IP core was irrelevant. Therefore, the FFT IP configuration was selected in digit reverse output mode to save resources and latency. Moreover, the FFT Core does not offer the most common sample order representation, with the center frequency bin in the middle of the output window, hence it is implemented using internal FPGA memory.

B. LOGARITHM COMPRESSION

A block diagram of a 12-bit binary logarithmic converter with 4-bit integer and 8-bit fractional data format, evaluated in three pipeline stages, is shown in Fig. 9. The implementation of 16-bit Leading-One Detector (LOD), based on the group of four parallel 4-bit LOD circuits, followed by a single 4-bit LOD that determines the group containing leading-one, was introduced in [49]. The second stage LOD, controlling four enable lines to four groups of 4-bit multiplexers, provides a 16-bit binary word with only one bit set to point the leading one. Next, a 16-bit to 4-bit binary encoder is utilized to obtain the logarithm integer part. A barrel shifter, driven by the inverted LOD output, rotates the input to prepare the data for linear piecewise approximation. The determination of segment affiliation is defined by the two most significant bits that are used to address the set of multiplexers. According to (5), the slope coefficients and intercept values are added together in the last pipeline stage to represent the logarithm fractional part. The error LUT, which compensates for the residual error, was omitted in this design due to low bit-width

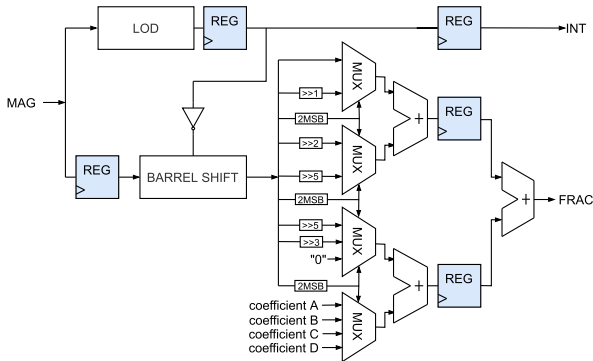


FIGURE 9. Register transfer level diagram of logarithm estimator.

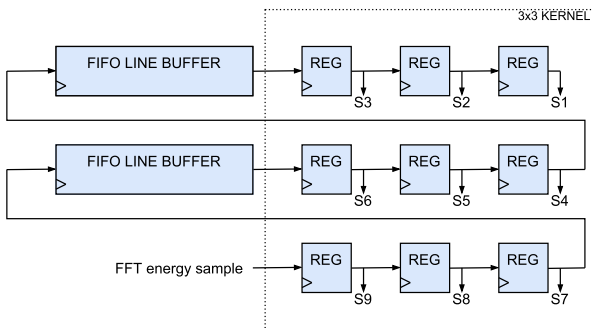


FIGURE 10. Sliding window used for Gaussian filtering.

availability for the fractional part. However, to achieve higher accuracy, it can be easily implemented in the future without the need for extensive architectural alterations.

C. NOISE FILTERING

Gaussian filtering of the spectrogram with parameters based on the previous simulations and 3×3 sliding window was implemented as shown in Fig. 10. In this architecture, using the input data as an energy sample from the FFT module and two 2048-long linear buffers, a filter mask (kernel) is prepared for further calculations. Kernel k can be written as:

$$k = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}. \quad (21)$$

The Gaussian convolution matrix is given by:

$$G(x, y) = S(x, y) \otimes k(x, y), \quad (22)$$

where \otimes denotes convolution, $k(x, y)$ is a Gaussian kernel, $S(x, y)$ is the input energy signal and $G(x, y)$ is the convolved output. As a result the output stream of the weighted sum is generated according to:

$$G5 = (S1 + 2 \cdot S2 + S3 + 2 \cdot S4 + 4 \cdot S5 + 2 \cdot S6 + S7 + 2 \cdot S8 + S9)/16. \quad (23)$$

Selected filter parameters reduce the implementation effort to simple usage of shift operations instead of utilizing hardware multipliers. Some pipeline registers are introduced in the module to achieve high performance, as shown in Fig. 11.

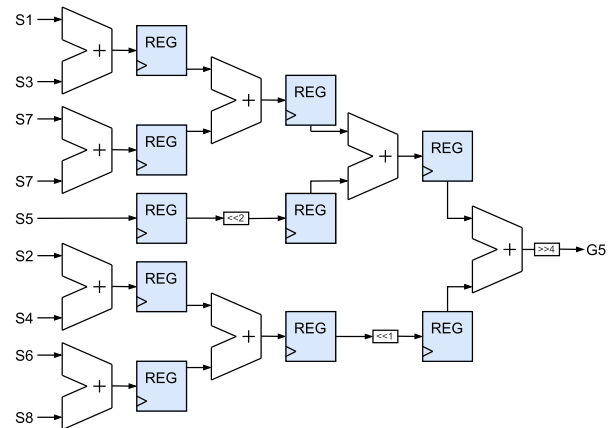


FIGURE 11. Register transfer level diagram of convolution realization.

D. BINARIZATION

The selected technique of binarization is described in section III-E. As the method is meant for software use, some adjustments are considered to adapt the original algorithm for the hardware computation. One of the main advantages of the NPAQM technique is the calculation simplicity in terms of machine instructions, compared to Otsu's method. However, the need for floating-point calculations and input data collection sorting complicates the hardware approach. Adjustments proposed in this article do not influence the mathematical base of the original NPAQM method, but proposes a way to perform the process in hardware. Figure 12 provides an overview of the calculation process. The module latency is a product of the histogram creation from FFT of length F in the first stage, between class variance calculation for M quantization levels.

1) FIRST STAGE-DATA PRE-PROCESSING

The quantization of the input sample magnitude range to narrow down the search area for the threshold value is proposed in the NPAQM method. The idea is to test each quantized value instead of all possible samples to determine the optimal threshold candidate. This approach gives a tremendous advantage for software computation. However, to create a set of optimal quantized values a logarithmic and square root calculations are employed according to (9). In the proposed approach, the quantization level was chosen arbitrarily. The signal latency and calculation time must be constant in the pipelined processing, regardless of the sample values. The quantization value M was chosen as a constant to find the balance between resource utilization, latency, and accuracy, based on various simulations. Besides, as the optimal threshold for signal detection is located just above the noise level, only the first $H = 2048$ values over the signal minimum in the current spectrum frame were considered a potential threshold. This approach opens the space for a second modification, which is eliminating the need to sort input samples. Although sorting is a basic procedure in software, histogram implementation was chosen for FPGA as a more efficient technique. Moreover, in the between class variance

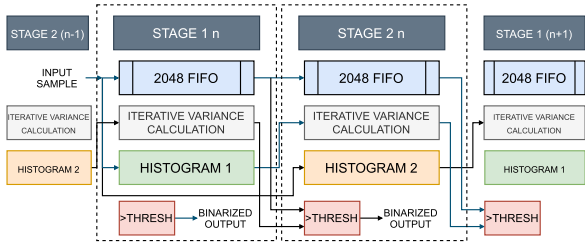


FIGURE 12. Block diagram of pipelined NPAQM hardware implementation. The single phase of the two-stage thresholding process is presented in detail inside a dashed line.

calculation proposed in the NPAQM method, where only the mean and the number of elements from corresponding subsets are significant, the same result is obtained. Thus, a histogram module containing H bins that relate to the selected threshold values was implemented. In this module, the information about the number of elements is stored. The set of bins, as threshold candidates, are created at the input of the first stage by using the information from the previous section about minimum and maximum sample values in the processed spectrum frame. The first bin always matches the minimum value, and if the difference between the minimum and maximum is no greater than the H , every value in the set is considered a potential threshold. Otherwise, the search area is limited, but there is no information loss. This is assured by an additional register implemented to store the cumulative sum of all elements in the spectrum frame needed for calculations in the second stage. The architecture of the histogram containing H bins corresponding to adequate threshold values is shown in Fig. 13. To obtain the histogram, the content of the memory location, addressed by a recent quantized sample value, is incremented. A single clock cycle read-modify-write operation is required in the case of a pipeline scenario, which can be achieved using dual clock edge operation. Due to the simplicity of implementation, a dual memory bank was used instead. The sum of partial results is calculated on the output to get the final histogram. In the native dual-port RAM configuration, one port is intended for histogram calculation, and the second for data transferring to the next stage and clearing memory afterward. This approach does not introduce additional multiplexers and minimizes the critical signal path from the RAM output through the adder to the input. As the histogram data is used in the second stage calculations, alternating histogram modules are employed to prevent overwriting the data.

2) SECOND STAGE-BETWEEN CLASS VARIANCE

A sequential calculation of between class variance is performed according to (17) in the second stage of operation to obtain the optimum value of the threshold. All arithmetic calculations are done using the fixed-point format to save resources and better control the maximum frequency of system implementation. To compute the mean of the two subsets, $\omega(\gamma_i)$ and $S(\gamma_i)$, the i/H value in the range of $0 \leq i \leq H$ was pre-calculated and stored in ROM in 24-bit unsigned fixed-point format. The result of the multiplication

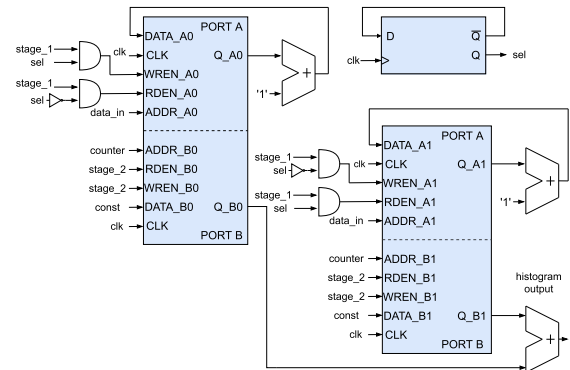


FIGURE 13. Register transfer level diagram of dual memory block histogram module.

of pre-calculated probabilities $P_\omega(\gamma_i)$ and $P_s(\gamma_i)$ was stored accordingly. The sum of elements in $\omega(\gamma_i)$ is taken as a result of the multiplication of the histogram bin address and the value stored under this address. The result is accumulated thru the iterations. The sum of elements in $S(\gamma_i)$ is obtained by subtracting the sum of elements in $\omega(\gamma_i)$ from the sum of all elements in the frame, calculated in the first stage.

Data prepared as described enables to calculate between class variance after a series of pipelined multiplications. Each final result of (17) is stored in the register, and the first-order difference is applied between iterations to find the optimal threshold according to (19). The last step is to examine the noise-only condition as described in the heuristic approach. All the data required for this purpose had already been calculated in the previous stage and had to be recalled from suitable registers to acquire the final threshold. At this moment, the delayed signal samples going out from the FIFO buffer are compared to the threshold, and the binarization process is finished. Every thresholded value is placed in the 12-bit shift register to preserve the 12-bit default output data format. The new output is indicated after every twelfth input sample, thus giving a compression ratio needed for the overall bitrate reduction concept.

V. RESULTS AND DISCUSSION

The implementation presented in the paper was done using VHSLC Hardware Description Language (VHDL), synthesized in Quartus Prime 20.1, and simulated in integrated ModelSim software provided by Intel. The synthesis results of the proposed design for INTEL CYCLONE IV E: EP4CE40F23C8 FPGA located on the LimeSDR-USB board version 1.4s indicates: 112 MHz maximum system frequency and 62% of overall resource utilization. All PC software for system modeling used in this work was developed in the Python 3.6 programming language. The motivation to apply Python instead of MATLAB was that it is a cross-platform approach, and PC-developed software can be fitted to Raspberry Pi after minimal effort. Moreover, the original USB drivers delivered by the LimeSDR-USB manufacturer provide Python support.

The real-life drone signal dataset, created in the initial development stage for experiments and parameters

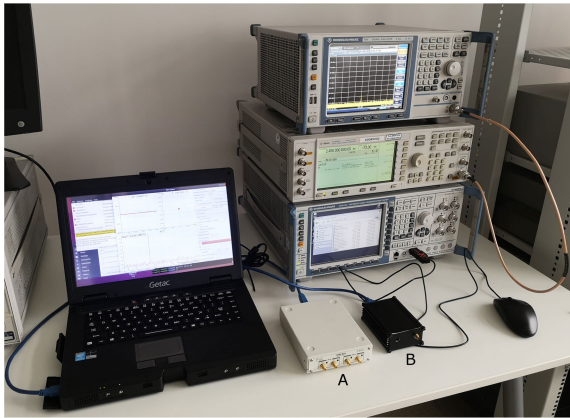


FIGURE 14. Laboratory setup for sensitivity measurements. The Agilent Digital Signal Generator was used as an arbitrary GFSK waveform source for (a) USRP B210, and (b) LimeSDR-USB. The low-loss coaxial cable connection was verified with the Rohde&Schwarz FSV Signal Analyzer.

optimization, was acquired using the original low-cost LimeSDR-USB as RF front-end without firmware modifications, and companion computer for data storage. The idea was to record a clear time domain drone signal without external interference, but still taking into account the channel propagation characteristics. This data was used for the early optimization of the signal processing chain. However, the main goal of this step was to manually tune a set of analog front-end receiver parameters, without GNU Radio support.

After software simulations to find satisfying parameters, including FFT window length, filter type, thresholding algorithm, and investigating approximation quality trade-off, another dataset was obtained using the proposed final implementation. Mavic 2 Zoom (DJI, Shenzhen, China) and Duplex 2.4EX (JETI model, Czech Republic) were utilized as a signal source in the data acquisition process. The product documentation confirmed that the signal of interest is located in the 2.400 GHz–2.483 GHz ISM frequency band. Therefore, the center frequency of the proposed receiver was set to 2.4415 GHz, with the bandwidth adjusted to 83 MHz in all experiments. Contrary to the initial dataset, frequency domain data was collected at this point. The details of the test set configuration are described in the following subsections depending on the experiment. In both cases, the environmental conditions did not change, and there was a clear line of sight between the transmitter and receiver. To further ensure the compliance of the data with real working conditions, the measurements were made in the vicinity of the local aerodrome in Gliwice.

Apart from real-life data, to evaluate the proposed sensor performance, some laboratory measurements were also executed to serve as a reference point for further considerations. The evaluation of the proposed approach based on LimeSDR-USB is discussed in the following subsections with reference to the USRP B210.

A. LABORATORY MEASUREMENTS

The study presented in [18], [50] offers an analysis of the signal-to-noise (SNR) ratio impact on the neural networks

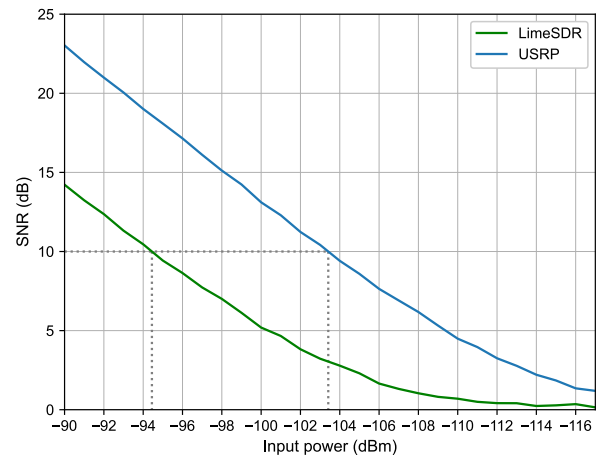


FIGURE 15. Output signal over the noise ratio in terms of given reference input signal power. The 10 dB SNR level taken as a reference point was achievable for -94.5 dBm input power when using LimeSDR, and -103.5 dBm input power using USRP B210.

classification performance employed for drone detection. In stationary scenarios with constant environmental background noise, the SNR ratio of an input signal decreases in function of the distance between the transmitter and the receiver. The RF sensor operation range can be estimated using the free-space propagation model when the required SNR and system gain are known parameters. The laboratory setup, shown in Fig. 14, was arranged to compare the SNR of the proposed and reference sensor in terms of received signal power. In this experiment, the low-loss coaxial cable connection between the E4432B Agilent Digital Signal Generator and SDR receiver was used instead of the antenna to minimize external interference. Both receivers were connected by USB cable to the PC and controlled by the original software provided by the manufacturers. All instruments were turned on for 2 hours beforehand to reach thermal equilibrium as suggested in [51]. To verify the prepared measurement setup, the Rohde&Schwarz FSV Signal Analyzer was connected instead of SDRs to determine the real power at the signal entry point. The reference signal was chosen to represent the generic drone signal between 2.400 GHz – 2.483 GHz frequency range with Gaussian Frequency-Shift Keying (GFSK) modulation, and the receivers were configured accordingly. Both devices operated with maximum available RF gain and bandwidth. In the first step of this experiment, with the reference signal generator turned off, 1000 trails of the test statistic, collecting 2048 samples containing noise only, were captured. Next, with the signal generator turned on, another 1000 trials were performed for signal power between -120 dBm and -90 dBm with 1 dBm step. The SNR in each trial was evaluated, and maximum likelihood estimation was used to obtain the final SNR from all detection trials. Figure 15 shows a considerable difference between the calculated relative signal level over the noise floor for both devices at the same input signal power. The 10 dB level is considered a reference point for comparison according to the analysis of detection effectiveness presented in [18], [50]. The source of

the 9 dB difference is explainable by the maximum available gain of the RF input stage, which is higher for the USRP. From the results, it can be concluded that a limited detection range is achievable using LimeSDR-USB. The USRP is supplied with on-board directly soldered SMA connectors and an original shielding enclosure. Whereas for the Lime SDR board, supplied as a bare PCB by default, a simple shielding enclosure and UFL to SMA adapter was arranged for the purpose of this work. Fortunately, the enclosure does not introduce additional impedance mismatch and significant cable connection losses.

B. STFT DETECTION VERIFICATION

System performance evaluation in terms of hopping time measurement was performed using the Duplex 2.4EX controller as a signal source. The experimental setup was based on off-the-shelf components, including a 3 dBi omnidirectional antenna, a band-pass filter to remove out-of-band interference, and Raspberry Pi 4.0. In this case, the dwell time of the signal is a known parameter similar to considered in [20]. Therefore the comparison with the actual value is made, along with the Normalized Mean Square Error (NMSE), expressed as:

$$NMSE = \frac{1}{N} \sum_{i=1}^N \left(\frac{\hat{t}_i - t}{t} \right)^2, \tag{24}$$

where \hat{t}_i is measured dwell time, and t is the actual value of it. As proposed by the Authors of the detection method in [20], N is assumed to be 22. Since the sensor detection range relies strongly on the system gain, which was not included in the cited work, the results presented in Fig. 16 can not be directly compared in terms of detection range. However, the overall correlation between the distance and the dwell time NMSE is consistent with previous studies. Some differences may be caused by the more advanced thresholding algorithm applied in this work. The error increasing above 150m is mainly caused by the absence of a signal at the expected moment in time and not a false measurement of its duration.

C. SPECTROGRAM FEATURES COMPARISON

Spectrogram features extracted from the public dataset, obtained simultaneously by dual USRP devices observing half of the 2.4 GHz ISM spectrum each, were employed for drone detection in [23], [24] together with machine learning techniques. The dataset contains IQ time domain samples of three drone classes operating in different modes. A recent analysis in [24] revealed the pre-processing steps to eliminate the noise and shrink the data size before calculations. To extract the essential information from the dataset, the Authors calculated the frequency domain transform with zero mean signals and averaged it by 15, corresponding to the proposed sensor architecture. It is hard to directly compare this public dataset with the data obtained from the proposed sensor due to the impossibility of reproducing the measurement conditions. In exchange, the data collected in a

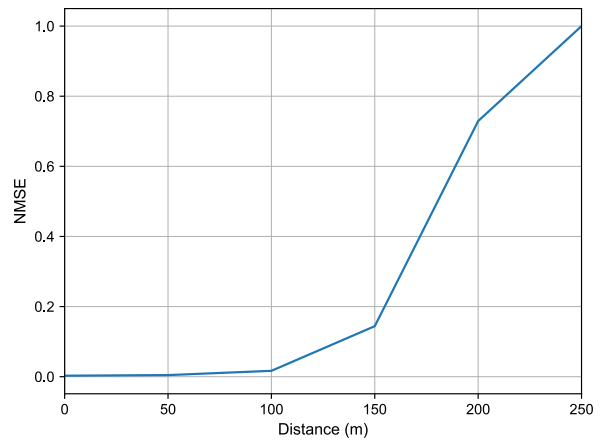


FIGURE 16. NMSE of dwell time measurements in terms of object distance from the proposed sensor. In current measurement scenario, the signal was undetectable over the background noise level over 250m distance.

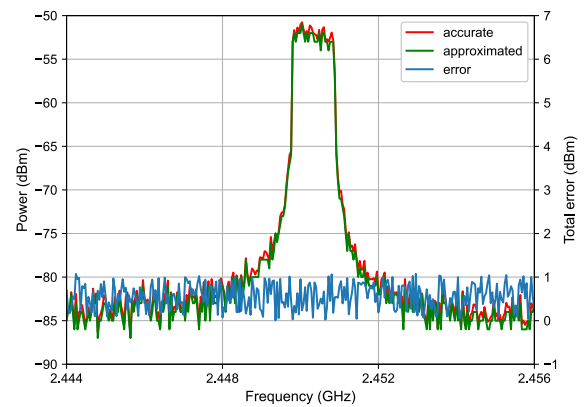


FIGURE 17. Spectral estimate of a single drone data communication frame, averaged by 16 using accurate and approximated techniques, and the approximation total error. About 1 dB of the total error between the exact and approximate spectrum from LimeSDR-USB meets current expectations.

real-life scenario with the reference and the proposed sensor connected in parallel are presented using Mavic 2 Zoom as a signal source. The experimental set with a single antenna and RF splitter was chosen to ensure the same input signal for both receivers. Both devices were connected simultaneously to the same PC. For the data captured with USRP, the FFT and averaging were applied in the software, whereas the spectrum estimate provided by the proposed sensor was obtained directly. About 1 dB of the total error between the exact and approximate spectrum from LimeSDR-USB, shown in Fig. 17, meets the current expectations. This is the contribution of both applied approximation techniques, limited bit-width, and fixed-point averaging by truncated right shifts.

The spectrogram presented in Fig. 18 shows a significant correlation between the data obtained by both devices. The comparison in terms of detection effectiveness using obtained data stays out of the scope of this study because it requires the implementation of other Authors’ algorithms. In summary, these results suggest that it is better to employ the sensor to build a new machine learning dataset rather than apply it

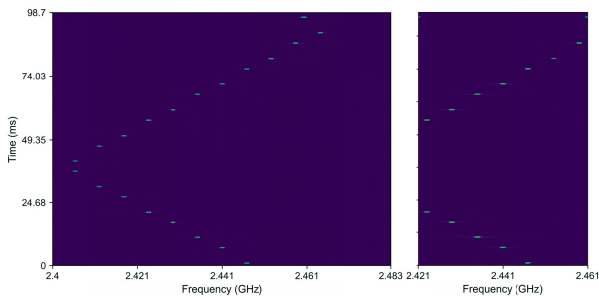


FIGURE 18. Spectrogram of a drone communication signal captured via LimeSDR-USB with 83 MHz bandwidth (left), and USRP B210 with 40 MHz bandwidth (right). Characteristic signal features and full hopping pattern are visible through continuous ISM 2.4GHz RF bandwidth analysis.

to networks already trained on the public dataset. Moreover, with twice the sampling rate offered by LimeSDR-USB, additional features related to the temporal aspects of the drone communication signal and hopping patterns could be considered during the development of new detection algorithms.

VI. CONCLUSION AND FUTURE WORK

In this paper, a cost-effective RF sensor with data pre-processing for commercial drone detection and its hardware implementation was presented. The single receiver was used to deliver a spectrum estimator of the entire 2.4 ISM GHz frequency band by the USB interface instead of raw IQ data. Therefore, the entire processing power of the companion computer software can be utilized for neural network-based or similar calculations without the need to pre-evaluate any time-frequency domain transformation. Besides, after selecting the binarized output mode for the STFT detection approach, the computing power requirement can be reduced even further. The verification results showed no significant impact on the observed signal shape obtained by the adapted approximation techniques over accurate calculations and reference device indications. This leads to the conclusion that the proposed sensor can be used for creating the new drone RF classification dataset. In terms of dwell time measurement, good performance was noted even in low SNR conditions, thanks to the advanced adaptive thresholding. The most relevant limitation of the proposed sensor compared to the USRP B210 is the lower gain of the RF chain inside the LimeSDR-USB, which results in a 9 dB sensitivity reduction.

Future work will focus on a distributed network of drone detection sensors based on the proposed low-cost hardware set for scenarios where the performance of a single central RF sensing element is limited. Development of more advanced filtering, other than a uniform window shape and spectrogram with overlap between frames to increase resolution, will also be investigated. In addition, consideration should be given to expanding the FPGA contribution towards broader data pre-processing and further offloading software algorithms to minimize the detection latency.

Field measurements of the proposed sensor revealed the incontestable impact of the background noise level and local

interference on the quality of drone detection. In the future, care should be taken to develop a specific test methodology allowing the comparison of different counter-drone approaches based on RF detection, regardless of varying external conditions. For the purposes of laboratory tests and software simulations, in addition to the drone signature database, a common database of background noise recorded in different places can be helpful. To ensure reproducible test conditions real-life Wifi, Bluetooth, and drone signals can be mixed and superimposed on the background noise to create common test waveforms. In addition, some MATLAB and Simulink RF toolboxes can be useful to simulate channel propagation characteristics or multi-drone scenarios. This approach will enable better control of the provided signal-to-noise ratio, which is essential for determining the quality of the RF drone detection and classification system. Moreover, using an arbitrary waveform with Vector Signal Generator or other SDR as the transmitter will enable experiments inside the EMC test chamber, where flying a drone is not always possible.

ACKNOWLEDGMENT

The author would like to express his sincere gratitude to Prof. Roman Czyba, Silesian University of Technology, Poland, for professional advice on his research. He would also like to thank Dr. Maciej Surma, Silesian University of Technology, Poland, for his assistance and for providing the infrastructure of the measurement laboratory.

REFERENCES

- [1] SESAR Joint Undertaking and Publication Office of the EU. (Apr. 21, 2017). *European Drones Outlook Study*. [Online]. Available: <https://op.europa.eu/s/oHuM>
- [2] P. K. R. Maddikunta, S. Hakak, M. Alazab, S. Bhattacharya, T. R. Gadekallu, W. Z. Khan, and Q.-V. Pham, "Unmanned aerial vehicles in smart agriculture: Applications, requirements, and challenges," *IEEE Sensors J.*, early access, Jan. 6, 2021, doi: 10.1109/JSEN.2021.3049471.
- [3] M. Novák, V. Novák, and J. Pech, "The use of drones in the distribution of energy," in *Proc. 9th Int. Conf. Adv. Comput. Inf. Technol. (ACIT)*, Jun. 2019, pp. 417–420.
- [4] A. Alcántara, J. Capitán, A. Torres-González, R. Cunha, and A. Ollero, "Autonomous execution of cinematographic shots with multiple drones," *IEEE Access*, vol. 8, pp. 201300–201316, 2020.
- [5] R.-C. Lee and Y.-H. Chen, "Bridge pier inspection system using UAV," in *Proc. IEEE Eurasia Conf. IoT, Commun. Eng. (ECICE)*, Oct. 2020, pp. 32–35.
- [6] J. Grzybowski, K. Latos, and R. Czyba, "Low-cost autonomous UAV-based solutions to package delivery logistics," in *Advanced, Contemporary Control*, A. Bartoszewicz, J. Kabziński, and J. Kacprzyk, Eds. Cham, Switzerland: Springer, 2020, pp. 500–507.
- [7] A. Solodov, A. Williams, S. A. Hanaei, and B. Goddard, "Analyzing the threat of unmanned aerial vehicles (UAV) to nuclear facilities," *Secur. J.*, vol. 31, no. 1, pp. 305–324, Apr. 2017.
- [8] European Union Aviation Safety Agency, Cologne, Germany. (Mar. 8, 2021). *Drone Incident Management at Aerodromes*. [Online]. Available: <https://www.easa.europa.eu/sites/default/files/dfu/>
- [9] D. Djudjic. (Jan. 18, 2021). *Man Facing Prison 250,000 Fine After Smashing his Drone Into a Police Helicopter*. [Online]. Available: <https://www.diyphotography.net/man-facing-prison-and-250000-fine-after-smashing-his-drone-into-a-police-helicopter/>
- [10] B. Gonzalez, "Drones and privacy in the Golden state," *Santa Clara High Tech. L.J.*, vol. 33, no. 2, pp. 288–323, 2017. [Online]. Available: <https://digitalcommons.law.scu.edu/chtj/vol33/iss2/3>
- [11] Federal Aviation Administration. *UAS Sightings Report*. Accessed: Feb. 20, 2021. [Online]. Available: https://www.faa.gov/uas/resources/public_records/uas_sightings_report/

- [12] H. Ryu, I. Wee, T. Kim, and D. H. Shim, "Heterogeneous sensor fusion based omnidirectional object detection," in *Proc. 20th Int. Conf. Control, Autom. Syst. (ICCAS)*, Oct. 2020, pp. 924–927.
- [13] S. Salman, J. Mir, M. T. Farooq, A. N. Malik, and R. Haleemdeen, "Machine learning inspired efficient audio drone detection using acoustic features," in *Proc. Int. Bhurban Conf. Appl. Sci. Technol. (IBCAST)*, Jan. 2021, pp. 335–339.
- [14] D. D. C. Trapal, B. C. C. Leong, H. W. Ng, J. T. G. Zhong, S. Srigrarom, and T. H. Chan, "Improvement of vision-based drone detection and tracking by removing cluttered background, shadow and water reflection with super resolution," in *Proc. 6th Int. Conf. Control Robot. Eng. (ICCARE)*, Apr. 2021, pp. 162–168.
- [15] N. Del-Rey-Maestre, M.-P. Jarabo-Amores, D. Mata-Moya, A. Almodóvar-Hernández, and P.-J. G.-D. Hoyó, "A DVB-T passive radar 3D-detection approach based on non-coherent spatial integration," in *Proc. 17th Eur. Radar Conf. (EuRAD)*, 2021, pp. 362–365.
- [16] V. Semkin, M. Yin, Y. Hu, M. Mezzavilla, and S. Rangan, "Drone detection and classification based on radar cross section signatures," in *Proc. Int. Symp. Antennas Propag. (ISAP)*, Jan. 2021, pp. 223–224.
- [17] J. Gérard, J. Tomasik, C. Morisseau, A. Rimmel, and G. Vieillard, "Micro-Doppler signal representation for drone classification by deep learning," in *Proc. 28th Eur. Signal Process. Conf. (EUSIPCO)*, Jan. 2021, pp. 1561–1565.
- [18] M. Ezuma, F. Erden, C. K. Anjinappa, O. Ozdemir, and I. Guvenc, "Detection and classification of UAVs using RF fingerprints in the presence of Wi-Fi and Bluetooth interference," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 60–76, 2020.
- [19] I. Bisio, C. Garibotto, A. Sciarrone, and F. Lavagetto, "Blind detection: Advanced techniques for WiFi-based drone surveillance," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 938–946, Jan. 2019.
- [20] B. Kaplan, I. Kahraman, A. Görçin, H. A. Çirpan, and A. R. Ekti, "Measurement based FHSS-type drone controller detection at 2.4 GHz: An STFT approach," in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, May 2020, pp. 1–6.
- [21] I. S. Gelman, J. P. Loftus, and A. A. Hassan, "Adversary UAV localization with software defined radio," Worcester Polytech. Inst., Worcester, MA, USA, Tech. Rep. E-project-041719-144214, Apr. 2019.
- [22] C. Xu, B. Chen, Y. Liu, F. He, and H. Song, "RF fingerprint measurement for detecting multiple amateur drones based on STFT and feature reduction," in *Proc. Integr. Commun. Navigat. Surveill. Conf. (ICNS)*, Sep. 2020, pp. 4G1-1–4G1-7.
- [23] M. S. Allahham, M. F. Al-Sa'd, A. Al-Ali, A. Mohamed, T. Khattab, and A. Erbad, "DroneRF dataset: A dataset of drones for RF-based detection, classification and identification," *Data Brief*, vol. 26, Oct. 2019, Art. no. 104313. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352340919306675>
- [24] I. Nemer, T. Sheltami, I. Ahmad, A. U.-H. Yasar, and M. A. R. Abdeen, "RF-based UAV detection and identification using hierarchical learning approach," *Sensors*, vol. 21, no. 6, p. 1947, Mar. 2021. [Online]. Available: <https://www.mdpi.com/1424-8220/21/6/1947>
- [25] S. Yang, H. Qin, X. Liang, and T. Gulliver, "An improved unauthorized unmanned aerial vehicle detection algorithm using radiofrequency-based statistical fingerprint analysis," *Sensors*, vol. 19, no. 2, p. 274, Jan. 2019.
- [26] Ettus Research, *The USRP X310*. Accessed: May 31, 2021. [Online]. Available: <https://www.ettus.com/all-products/x310-kit/>
- [27] *The USRP B210*. Accessed: May 31, 2021. [Online]. Available: <https://www.ettus.com/all-products/ub210-kit/>
- [28] Lime Microsystems. (2014). *FPRF MIMO Transceiver IC With Integrated Microcontroller, Revision 3.2*. [Online]. Available: https://github.com/myriadrf/LMS7002M-docs/blob/master/LMS7002M_Data_Shee_v3.2r00.pdf
- [29] Lime Microsystems. *The LimeSDR*. Accessed: Jan. 10, 2021. [Online]. Available: <https://limemicro.com/products/boards/limesdr/>
- [30] R. Sahu, "Theoretical and practical approach to GNU radio and LimeSDR platform," Tech. Rep., Jan. 2020. Accessed: Jan. 10, 2021. [Online]. Available: https://www.researchgate.net/publication/344675197-Theoretical_and_Practical_Approach_to_GNU_Radio_and_LimeSDR_Platform
- [31] G. Karm. *LimeSDR Mini FPGA-Accelerated Real-Time Spectrogram*. Accessed: Jan. 10, 2021. [Online]. Available: <https://github.com/gasparka/spectrogram>
- [32] Tektronix. *Understanding FFT Overlap Processing Fundamentals*. Accessed: Jan. 10, 2021. [Online]. Available: https://download.tek.com/document/37W_18839_1.pdf
- [33] W. Liu, F. Lombardi, and M. Shulte, "A retrospective and prospective view of approximate computing [point of view]," *Proc. IEEE*, vol. 108, no. 3, pp. 394–399, Mar. 2020.
- [34] M. Allie and R. Lyons, "A root of less evil [digital signal processing]," *IEEE Signal Process. Mag.*, vol. 22, no. 2, pp. 93–96, Mar. 2005.
- [35] J. Brady and W. Adams, "Magnitude approximations for microprocessor implementation," *IEEE Micro*, vol. 3, no. 5, pp. 27–31, Sep. 1983.
- [36] F. J. Harris, "On detecting white space spectra for spectral scavenging in cognitive radios," *Wireless Pers. Commun.*, vol. 45, no. 3, pp. 325–342, May 2008.
- [37] A. M. Mansour, A. M. El-Sawy, M. S. Aziz, and A. T. Sayed, "A new hardware implementation of base 2 logarithm for FPGA," *Int. J. Signal Process. Syst.*, vol. 3, no. 2, pp. 171–181, Jan. 2014.
- [38] H. Dong, M. Wang, Y. Luo, M. Zheng, M. An, Y. Ha, and H. Pan, "PLAC: Piecewise linear approximation computation for all nonlinear unary functions," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 28, no. 9, pp. 2014–2027, Sep. 2020.
- [39] M.-H. Nguyen, "An efficient hardware logarithm generator with modified quasi-symmetrical approach for digital signal processing," *Int. J. Electr. Comput. Eng.*, vol. 10, no. 5, p. 4671, Oct. 2020.
- [40] M. Motwani, M. Gadiya, R. Motwani, and F. Harris, "Survey of image denoising techniques," in *Proc. GSPX*, Santa Clara, CA, USA, 2004, pp. 27–30.
- [41] N. Otsu, "A threshold selection method from gray-level histograms," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-9, no. 1, pp. 62–66, Jan. 1979.
- [42] G. Bradski, "The OpenCV library," *Dr. Dobbs's J. Softw. Tools*, vol. 25, pp. 120–125, Nov. 2000.
- [43] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Phys. D, Nonlinear Phenomena*, vol. 60, nos. 1–4, pp. 259–268, Nov. 1992.
- [44] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Proc. 6th Int. Conf. Comput. Vis.*, 1998, pp. 839–846.
- [45] T. Q. Vinh, J. H. Park, Y.-C. Kim, and S. H. Hong, "FPGA implementation of real-time edge-preserving filter for video noise reduction," in *Proc. Int. Conf. Comput. Electr. Eng.*, Dec. 2008, pp. 611–614.
- [46] A. J. Onumanyi, A. M. Abu-Mahfouz, and G. P. Hancke, "A comparative analysis of local and global adaptive threshold estimation techniques for energy detection in cognitive radio," *Phys. Commun.*, vol. 29, pp. 1–11, Aug. 2018.
- [47] A. J. Onumanyi, A. M. Abu-Mahfouz, and G. P. Hancke, "Amplitude quantization method for autonomous threshold estimation in self-reconfigurable cognitive radio systems," *Phys. Commun.*, vol. 44, Feb. 2021, Art. no. 101256. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1874490720303335>
- [48] Myriad-RF. (2020). *LimeSDR-USB*. [Online]. Available: <https://github.com/myriadrf/LimeSDR-USB>
- [49] K. H. Abed and R. E. Siferd, "CMOS VLSI implementation of a low-power logarithmic converter," *IEEE Trans. Comput.*, vol. 52, no. 11, pp. 1421–1433, Nov. 2003.
- [50] E. Ozturk, F. Erden, and I. Guvenc, "RF-based low-SNR classification of UAVs using convolutional neural networks," 2020, *arXiv:2009.05519*. [Online]. Available: <http://arxiv.org/abs/2009.05519>
- [51] T. Šolc, M. Mohorčič, and C. Fortuna, "A methodology for experimental evaluation of signal detection methods in spectrum sensing," *PLoS ONE*, vol. 13, no. 6, Jun. 2018, Art. no. e0199550.



PRZEMYSŁAW FLAK was born in Katowice, Poland. He received the B.S. and M.S. degrees in electronic engineering from Silesian University of Technology, Gliwice, Poland, in 2010, where he is currently pursuing the Ph.D. degree. Since 2009, he has been with the Flytronic SA, WB Group, Poland, where he is working on research and development topics related to drones, programmable systems, and radio telecommunication. His current research interests include unmanned aerial systems (UAS), counter-UAS systems (C-UAS), software-defined radio (SDR), and field-programmable gate arrays (FPGA). He was a recipient of the 5th Annual Diligent Design Contest Award, in 2009, organized within the Technical University of Cluj-Napoca, Romania, sponsored by Diligent and Xilinx.

• • •