Real-Time Variants of Vertical Synchrosqueezing: Application to Radar Remote Sensing

Karol Abratkiewicz, *Student Member, IEEE*, and Jacek Gambrych, *Member, IEEE*

***Abstract*—This article presents thorough research on using high-order vertical synchrosqueezing (VSS) in different radar remote sensing applications. The method well established in the literature is examined and compared to the novel form of third-order VSS and first-order VSS using an enhanced estimator of the instantaneous frequency, both proposed by the authors. An investigation shows that the two introduced variants of VSS are characterized by preserved capabilities (understood as the possibility to concentrate the time-frequency distribution and its reconstruction) with significantly reduced computation cost. The research shows that in practical radar remote sensing applications, high-order VSS can be successfully replaced by the approach proposed in this article with a lower computational burden. Furthermore, the methods are validated under numerical experiments, both simulated and real-life, which showed the efficiency of the proposed methods in radar signal processing, particular component extraction, and signal decomposition. Moreover, the authors developed the real-time graphical-processing-unit-based implementation of the proposed techniques and presented its efficiency in practical conditions.**

***Index Terms*—Compute unified device architecture (CUDA), radar remote sensing, radar signal processing, time–frequency (TF) analysis, vertical synchrosqueezing (VSS).**

I. Introduction

RADAR remote sensing encompasses many facets that may include not only detection of the range and velocity of a target, but also estimation of its instantaneous kinematic parameters, clutter, and jamming modeling, as well as techniques that serve to avoid being interfered with. This requires radar systems to be agile and needs extraction of data from various signals, such as linear/nonlinear frequency modulated (NLFM) pulses, micro-Doppler signatures, stochastic noise, or digitally modulated commercial signals widely used in passive radars [1]–[16].

Among many techniques whose objective is to find a given signal parameter, one of the most used is time-frequency (TF) processing due to several advantages, such as assessing time and frequency dependencies of the distribution simultaneously and invertibility into the time domain. However, linear transforms, e.g., the short-time Fourier transform (STFT), usually suffer from the limited resolution of the distribution defined by the Heisenberg–Gabor uncertainty principle [17]. It means that a narrow window leads to a poor frequency resolution, and a wide window results in a poor time resolution, thereby preventing unequivocal interpretation of the transform. In the literature, one can find several attempts to improve the readability of the STFT (and other transforms) pioneered by Kodera *et al.* [17]. Such techniques as TF reassignment [18] and vertical or horizontal synchrosqueezing [19] amount to the relocation of the transform values from a given point to another one expected to be closer to the true instantaneous frequency (IF) ridge. Even though these are not perfect and super-resolution methods, their popularity in the radar community has been increasing over recent years [20]. Moreover, TF reassignment favors sharpening over invertibility, while synchrosqueezing allows for the reconstruction of the signal at the cost of worse signal concentration on the TF plane [21].

In the literature, techniques whose aim is to concentrate the distribution isotropically simultaneously, allowing further reconstruction of the signal, can be found. One of the most recognizable techniques is second-order vertical synchrosqueezing (VSS2) proposed by Oberlin *et al.* [22]. The method uses an extended IF estimator considering the instantaneous frequency rate [chirp rate (CR)], which perfectly localizes linear chirps and guarantees reconstruction. However, the problem occurs when analyzing nonlinear or fast oscillating components. Then, the IF estimator is biased, and its bias is proportional to dynamically modulated terms (for example, for polynomial phase signals, the bias may be defined by the third- or higher-order component of the phase). In such a case, the continuity of the TF ridge can be broken, creating potential problems with mode extraction. Another significant improvement of the energy relocation was proposed by Fourer *et al.* [23], who used the concept of the complex phase of the STFT and a chirp signal model to deduce the IF estimators for each (t, ω) coordinate with reduced computational complexity compared to [22]. However, the existing impediments had to be resolved, namely, continuity of the concentrated transform ridge with simultaneous invertibility for nonlinear terms. Pham and Meignen [24] addressed this problem. They proposed an efficient algorithm allowing third- and fourth-order vertical synchrosqueezing (VSS3 and VSS4) to be obtained, and the properties of the method favor analysis of fast oscillating modulation. The method was successfully applied in, e.g., voice signal processing [25], seismic analysis [26], fault diagnosis for rolling bears [27], and others [28], [29]. This article, therefore, is based on the existing methods and presents an alternative approach, preserving usability of higher-order

Manuscript received November 5, 2021; revised December 13, 2021; accepted January 12, 2022. Date of publication January 25, 2022; date of current version February 22, 2022. *(Corresponding author: Karol Abratkiewicz.)*

The authors are with the Institute of Electronic Systems, Faculty of Electronics and Information Technology, Warsaw University of Technology, 00-665 Warsaw, Poland (e-mail: karol.abratkiewicz@pw.edu.pl; jacek.gambrych@pw.edu.pl).

Digital Object Identifier 10.1109/JSTARS.2022.3145085

VSS known from the literature but with reduced computational cost allowing for real-time computation.

Furthermore, highly optimized software using the graphical processing unit (GPU) was created for this article to check whether it is possible to use VSS in the real-time radar applications. Several factors dictated the choice of this computational platform. In recent years, GPUs have become more and more popular in the field of digital signal processing. This is mainly due to the enormous computational power they provide. As a result, it is possible to perform demanding calculations that would usually be performed offline in real-time in many cases. The relatively simple programming model, especially in contrast to field-programmable gate array devices, also contributes to the growing popularity of general-purpose computing on graphical processing units (GPGPUs). These features are closely related to the compute unified device architecture (CUDA) introduced by NVIDIA in 2007. Currently, the CUDA is one of the most popular technologies that enable GPGPUs. One of the reasons is that it is a complete technology. In addition to the hardware platform, the CUDA also includes an application programming interface and numerous libraries dedicated, among others, to digital signal processing, such as cuFFT. Such a combination means that the CUDA technology provides very high computational power and provides tools to make use of this power with a reasonable development effort. This feature made the choice of the GPU and CUDA technology as a platform for real-time computing described in this article.

Therefore, the novelty proposed in this article is threefold.

1) A novel form of third-order VSS and enhanced first-order VSS are proposed and compared to VSS2, VSS3, and VSS4 giving comparable outcomes but in reduced time.
2) High-order VSS is applied to different radar remote sensing applications. For the first time in the literature, this approach is used to concentrate and decompose multicomponent radar signals with strongly oscillating and nonlinear frequency modulation, which shows the technique's huge potential.
3) All variants of VSS considered in this article were implemented on a GPU platform, allowing real-time processing. To the authors' knowledge, this is the first real-time realization of VSS in the literature.

The rest of this article is organized as follows. Section II presents the background on high-order VSS known from the literature. In Section III, the authors introduce two fast modifications of the IF estimators. Section IV is devoted to numerical experiments using simulated signals and presents possibilities and limitations of all techniques considered in this article. GPU-based implementation of real-time VSS is described in Section V, and the real-life data analysis is covered in Section VI. Finally, Section VII concludes this article. Additionally, the authors have provided an appendix with derivations of the proposed IF estimators.

II. Higher-Order Vertical Synchrosqueezing

As the only algorithm of higher-order VSS for the STFT, the one proposed in [24] is the reference technique for the investigation conducted in the following. The multicomponent amplitude and frequency modulated (AM-FM) signal is defined as

$$x(t) = \sum_{k=1}^{K} x_k(t), \quad \text{with} \quad x_k(t) = A_k(t)\mathrm{e}^{j\phi_k(t)} \tag{1}$$

where $A(t)$ is the amplitude, $\phi(t)$ is the phase, and $j^2 = -1$. For nonstationary signals, a natural language of their processing is TF analysis, and the STFT is the method considered in this article. For a signal $x(t)$ and a differentiable window $h(t)$, the STFT is defined as

$$\begin{aligned} F_x^h(t,\omega) &= \int_{\mathbb{R}} x(\tau)h^*(\tau - t)\mathrm{e}^{-j\omega(\tau - t)}d\tau \\ &= \int_{\mathbb{R}} x(\tau + t)h^*(\tau)\mathrm{e}^{-j\omega\tau}d\tau \end{aligned} \tag{2}$$

where ω is the angular frequency and z^* is the complex conjugate of z. Such a TF distribution using the STFT suffers from the limited resolution defined by the Heisenberg–Gabor uncertainty principle [17]. The final resolution is a tradeoff between time and frequency separability, which reduces its readability. A recent attempt to improve component separation proposed in [24] consists of the relocation of transform values along the frequency axis. This operation may significantly improve the separability, preserving reconstruction capabilities. VSS is defined as

$$S_x^h(t,\omega) = \int_{\mathbb{R}} F_x^h(t,\omega')e^{j\omega'(t-t_0)}\delta\left(\omega - \hat{\omega}_x(t,\omega')\right)d\omega' \tag{3}$$

which is a modified version of classical VSS, given by

$$S_x^h(t,\omega) = \int_{\mathbb{R}} F_x^h(t,\omega')\delta\left(\omega - \hat{\omega}_x(t,\omega')\right)d\omega' \tag{4}$$

but simplifies the reconstruction formula to the following form [23]:

$$\hat{x}(t - t_0) = \frac{1}{2\pi h^*(t_0)}\int_{\mathbb{R}} S_x^h(t,\omega)d\omega \tag{5}$$

where $h(t)$ is non-zero and continuous at any t_0 (usually $t_0 = 0$), $\delta(\cdot)$ denotes the Dirac delta function, and $\hat{\omega}_x(t,\omega)$ is the frequency reassignment operator based on the IF estimator. At this point, it has to be mentioned that the IF estimator has a crucial influence on concentration capabilities. Initially, the frequency reassignment operator in the TF domain was defined as $\hat{\omega}_x(t,\omega) = \Im(\widehat{\omega}_x(t,\omega))$, where

$$\widehat{\omega}_x(t,\omega) = j\omega - \frac{F_x^{\mathcal{D}h}(t,\omega)}{F_x^h(t,\omega)} \tag{6}$$

where $\Im$ is the imaginary part of the complex number, $\mathcal{D}^n h = \frac{d^n h(t)}{dt^n}$ is the modification of the original window (in general nth order), and $\Im(\frac{F_x^{\mathcal{D}h}(t,\omega)}{F_x^h(t,\omega)})$ is the local instantaneous frequency estimate.

The expression $\hat{\omega}_x(t,\omega) = \Im(\widehat{\omega}_x(t,\omega))$ used in (3) results in a poor concentration over the energy ridge for rapid frequency modulation. To improve the concentration of the distribution, the IF estimator was first extended by Oberlin *et al.* [22] to the second-order form and next to the third- and fourth-order

by Pham and Meignen [24]. For the latter case, the amplitude and the phase of the signal were approximated using the Taylor expansion of (1) for τ close to t

$$x(\tau) = \exp\left(\sum_{k=0}^{N} \frac{[\log(A)]^{(k)}(t) + j\phi^{(k)}(t)}{k!}(\tau - t)^k\right) \quad (7)$$

where $X^{(k)}$ denotes the kth-order derivative of X with respect to time. Substituting (7) into (2) yields

$$F_x^h(t,\omega) = \int_{\mathbb{R}} \exp\left(\sum_{k=0}^{N} \frac{[\log(A)]^{(k)}(t) + j\phi^{(k)}(t)}{k!}\tau^k\right) h^*(\tau)\mathrm{e}^{-j\omega\tau}d\tau. \quad (8)$$

Consequently, one can introduce the higher-order IF estimator as the partial derivative of (8) with respect to time and dividing by $F_x^h(t,\omega)$ obtaining

$$\tilde{\omega}_x(t,\omega) = \sum_{k=1}^{N} r_k(t)\frac{F_x^{\mathcal{T}^{k-1}h}(t,\omega)}{F_x^h(t,\omega)} = [\log(A)]'(t) + j\phi'(t) + \sum_{k=2}^{N} r_k(t)\frac{F_x^{\mathcal{T}^{k-1}h}(t,\omega)}{F_x^h(t,\omega)} \quad (9)$$

where $r_k(t) = \frac{1}{(k-1)!}([\log(A)]^{(k)}(t) + j\phi^{(k)}(t))$ and $\mathcal{T}^n h = t^n h(t)$. In order to extract the precise IF estimate, one has to subtract $\Im(\sum_{k=2}^{N} r_k(t)\frac{F_x^{\mathcal{T}^{k-1}h}(t,\omega)}{F_x^h(t,\omega)})$ to compute $\Im(\tilde{\omega}_x(t,\omega))$, and for this purpose, modulation operators were derived such that $\tilde{q}_x^{[k,N]}(t,\omega) = r_k(t)$. They can be understood as polynomial phase components being approximated by the Nth-order Taylor expansion, as, e.g., $\phi''(t) \to \tilde{q}_x^{[2,4]}(t,\omega)$, $\phi'''(t) \to \tilde{q}_x^{[3,4]}(t,\omega)$, etc., assuming $N = 4$ order of the Taylor expansion. Thus, the Nth-order IF estimate can be defined as follows [24]:

$$\tilde{\omega}_x^{[N]}(t,\omega) = \begin{cases} \widehat{\omega}_x(t,\omega) + \sum_{k=2}^{N} \tilde{q}_x^{[k,N]}(t,\omega)(-f_{k,1}(t,\omega)), \\ \text{if } F_x^h(t,\omega) \neq 0, \text{and } \frac{\partial}{\partial_\omega} f_{j,j-1}(t,\omega) \neq 0 \\ 2 \leq j \leq N \\ \widehat{\omega}_x(t,\omega), \quad \text{otherwise} \end{cases} \quad (10)$$

where $f_{k,1}(t,\omega) = \frac{F_x^{\mathcal{T}^{k-1}h}(t,\omega)}{F_x^h(t,\omega)}$. The final frequency reassignment operator is given as $\hat{\omega}_x^{[N]}(t,\omega) = \Im(\tilde{\omega}_x^{[N]}(t,\omega))$ and can be directly substituted into (3) in order to obtain sharpened STFT distribution.

Higher-order VSS handles the problem of various AM-FM signals, including rapid and oscillating frequency modulation. However, the overall procedure of the computation of the IF estimate (and consequently the frequency reassignment operators) may be complicated. The method has the following drawbacks.

1) The computational complexity is relatively high, namely, it requires 14 STFTs each having computational complexity $\mathcal{O}(NK\log_2(N))$, where N is the number of points in the fast Fourier transform (FFT) and K is the number of time instants for which the FFT has to be applied, and the method requires additional matrix operations.
2) For computing higher-order IF estimators, one has to solve the matrix problem consisting of multiple operations on complex numbers because higher-order IF estimators are not given directly and need lower-order estimators to be computed.

These impediments prompted the authors to conduct intense research in this matter to obtain a novel IF estimator preserving high concentration capabilities with reduced computation time. The proposed method is described in the next section.

III. Simplification

In this section, the authors propose the two alternative forms of VSS: the novel form of the frequency reassignment operator based on the third-order IF estimator dedicated to canonical third-order VSS (CVSS3), and the second form based on the first-order IF estimator dealing with fast and nonlinearly modulated signals used in enhanced first-order synchrosqueezing (EVSS1) in the following.

A. Canonical Third-Order Vertical Synchrosqueezing

The approach proposed in this article extends second-order VSS by the use of the third-order phase component in the IF estimator [30]. In other words, the method is deduced from [22], which aims to define the local signal phase as

$$\phi_2(t) = \varphi_x + \omega_x t + \alpha_x t^2/2 \quad (11)$$

where φ_x is the initial phase, ω_x stands for the angular frequency, and α_x is the CR. Its first-order derivative leads to the instantaneous frequency estimate given as the local approximation

$$\phi_2'(t) = f(t) \approx f_0 + \alpha t \quad (12)$$

which in the TF domain can be rewritten as [22]

$$\phi_2'(t) = \phi'\left(\hat{\tau}_x(t,\omega)\right) + \phi''\left(\hat{\tau}_x(t,\omega)\right)\left(t - \hat{\tau}_x(t,\omega)\right) \quad (13)$$

and leads to the direct form of the frequency reassignment operator based on the second-order IF estimator as

$$\hat{\omega}_x^{[2]}(t,\omega) = \Im\left(\widehat{\omega}_x(t,\omega) + \widehat{q}_x(t,\omega)\left(t - \widehat{t}_x(t,\omega)\right)\right) \quad (14)$$

where $\widehat{t}_x(t,\omega) = t + \Re(\frac{F_x^{\mathcal{T}h}(t,\omega)}{F_x^h(t,\omega)})$, $\Re$ is the real part of the complex number, and $\widehat{q}_x(t,\omega)$ is the modulation operator unbiased when the amplitude varies and can be expressed by one of the following forms:

$$\widehat{q}_x^{\mathfrak{D}}(t,\omega) = \frac{F_x^{\mathcal{D}^2 h}(t,\omega)F_x^h(t,\omega) - F_x^{\mathcal{D}h}(t,\omega)^2}{F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) - F_x^{\mathcal{T}\mathcal{D}h}(t,\omega)F_x^h(t,\omega)} \quad (15)$$

and

$$\widehat{q}_x^{\mathfrak{F}}{}_{(t,\omega)} = \frac{F_x^{\mathcal{T}\mathcal{D}h}(t,\omega)F_x^h(t,\omega) + F_x^h(t,\omega)^2 - F_x^{\mathcal{T}h}(t,\omega)F_x^{\mathcal{D}h}(t,\omega)}{F_x^{\mathcal{T}h}(t,\omega)^2 - F_x^{\mathcal{T}^2 h}(t,\omega)F_x^h(t,\omega)}. \quad (16)$$

As can be seen in (14), the estimate assumes the signal instantaneous frequency rate to be linear. Local linearity cannot always be assumed; thus, extending the phase by the additional

factor β_x being the angular jerk yields

$$\phi_x(t) = \varphi_x + \omega_x t + \alpha_x t^2/2 + \beta_x t^3/3. \tag{17}$$

Analogously, extending (14) by consecutive terms can be done if higher-order modulation operators are known. Abratkiewicz [30] deduced the third-order modulation operator β_x in the TF domain given by

$$\widehat{p}_x(t,\omega) = \frac{\widehat{p}_x^N(t,\omega)}{\widehat{p}_x^D(t,\omega)} \tag{18}$$

where $\widehat{p}_x^N(t,\omega)$ and $\widehat{p}_x^D(t,\omega)$ are given by (19) and (20), (shown at the bottom of this page), respectively.

The authors of this article, therefore, propose the extension of (14) by (18), and the frequency reassignment operator based on the third-order IF can be deduced as the local approximation

$$\phi_3'(t) = f(t) \approx \omega_x + \alpha_x t + \beta_x t^2 \tag{21}$$

which in the TF domain can be derived as [22]

$$\begin{aligned}\phi_3'(t) = {} & \phi'\left(\hat{\tau}_x(t,\omega)\right) + \phi''\left(\hat{\tau}_x(t,\omega)\right)\left(t - \hat{\tau}_x(t,\omega)\right) \\ & + \phi'''\left(\hat{\tau}_x(t,\omega)\right)\left(t - \hat{\tau}_x(t,\omega)\right)^2.\end{aligned} \tag{22}$$

As revealed by Oberlin *et al.* [22], $\phi'(\hat{\tau}_x(t,\omega))$ and $\phi''(\hat{\tau}_x(t,\omega))$ can be expressed as (6) and (15) [equivalently (16)], respectively. Expression $\phi'''(\hat{\tau}_x(t,\omega))$ is the modulation operator given by (18), which leads to the novel frequency reassignment operator dedicated to CVSS3

$$\begin{aligned}\hat{\omega}_x^{[3']}(t,\omega) = \Im\Big(\widehat{\omega}_x(t,\omega) + \widehat{q}_x(t,\omega)\left(t - \widehat{t}_x(t,\omega)\right) \\ + \widehat{p}_x(t,\omega)\left(t - \widehat{t}_x(t,\omega)\right)^2\Big)\end{aligned} \tag{23}$$

where all the symbols are as established before. Equation (23) is valid whenever any numerator in (15), (16), and (18) is nonzero; otherwise, the IF is given by (6). Finally, CVSS3 (23) can be directly substituted into (3) giving a concentrated distribution even for oscillating terms. Estimator (23) can also be deduced from (8) (assuming the third-order signal phase) and brings significant benefits over (10). First, it does not require recursive computation resulting from the back-substitution algorithm used in [24]; second, the computational cost of (10) is lower than for (23) and only requires ten STFTs to be performed. A comparison of the computation time and the reconstruction capability is presented in the following.

B. Enhanced First-Order Synchrosqueezing

In [30], besides the estimator of β_x whose properties were used in (23) to CVSS3, the author proposed the enhanced frequency reassignment operator based on the first-order complex IF estimator, whose definition is given by

$$\widehat{\omega}_x^{[1]}(t,\omega) = j\omega - \frac{\widehat{\omega}_x^N(t,\omega)}{\widehat{\omega}_x^D(t,\omega)} \tag{24}$$

where $\widehat{\omega}_x^N(t,\omega)$ and $\widehat{\omega}_x^D(t,\omega)$ are given by (25) and (26), (shown at the bottam of the next page) respectively, and its imaginary part such that $\hat{\omega}_x^{[1]}(t,\omega) = \Im(\widehat{\omega}_x^{[1]}(t,\omega))$ leads to the enhanced frequency reassignment operator dedicated for VSS. It has to be noted that the assumption on the nonlinearity of the signal phase allows one to concentrate nonlinear (including oscillating) terms. Thanks to this, the final synchrosqueezed STFT distribution is sharp even for dynamic frequency modulation. Although the computational complexity is the same as in (23), the overall processing requires less complex multiplications, making the approach faster than for all other methods investigated in this article. The reader is encouraged to see more details regarding the estimator and mathematical fundamentals in the appendix and [30].

EVSS1 using (24) leads to comparable outcomes as when using (23), but the nature of these IF estimators is different. The frequency reassignment operator dedicated to CVSS3 is composed of signal parameter estimates assuming the signal phase to be a third-order polynomial at least locally. Consecutive estimates of the signal terms are computed by means of individual estimators and construct (23). The frequency reassignment operator (24) dedicated to EVSS1 results from direct estimation of the nonlinear instantaneous frequency in the TF domain, ignoring the individual factors resulted from the polynomial phase [30]. There is no clear and direct answer on which method should be used in a particular case. When the signal character is completely unknown, one can use the Rényi entropy (defined in Section IV) to find the estimator that gives the best energy concentration. When one can make some assumptions, it is possible to select the estimator with the best precision or the lowest processing time (which depends on the application). However, as shown in this article, both proposed frequency reassignment operators may be successfully applied for concentrating a signal with fast oscillating instantaneous frequency and nonlinear modulation, which enables mode extraction in a shortened time compared to the method known from the literature. Efficiency, the measurement of energy gathering properties, and the ability

$$\begin{aligned}\widehat{p}_x^N(t,\omega) = {} & F_x^{\mathcal{T}h}(t,\omega)\left[F_x^{\mathcal{D}^2h}(t,\omega)\right]^2 - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{D}^2h}(t,\omega)F_x^{\mathcal{D}h}(t,\omega) - F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega)F_x^{h}(t,\omega)F_x^{\mathcal{D}^2h}(t,\omega) \\ & + F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega)\left[F_x^{\mathcal{D}h}(t,\omega)\right]^2 - F_x^{\mathcal{D}^3h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega)F_x^{\mathcal{D}h}(t,\omega) + F_x^{\mathcal{D}^3h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{h}(t,\omega)\end{aligned} \tag{19}$$

$$\begin{aligned}\widehat{p}_x^D(t,\omega) = {} & F_x^{\mathcal{D}^2h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{T}^2h}(t,\omega) - F_x^{\mathcal{D}^2h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) - F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{T}^2h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) \\ & + F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega)F_x^{h}(t,\omega) + F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega)F_x^{h}(t,\omega)\end{aligned} \tag{20}$$

to reconstruct the signal and extracted modes are presented in the further part of this article.

C. Discrete Implementation

When analyzing nonstationary signals in the TF domain, one has to consider its discrete-time representation. The discretization process based on the rectangle method yields the approximation for each STFT $F_x^h[n,m] \approx F_x^h(nT_s, \frac{2\pi m}{MT_s})$, where $T_s = \frac{1}{f_s}$ is the sampling period, $n \in \mathbb{Z}$ expresses time indices, and $m \in [-M/2; +M/2]$ are the discrete frequency bins. Thereby, each vertical slice of the discrete-time STFT $F_x^h[n,m]$ can be efficiently computed using the FFT algorithm.

Additionally, readable distribution is obtainable when the analyzing window is characterized by appropriately adjusted width [31]. This issue is clear for linear chirps, for which the frequency slope is directly related to the time spread of the Gaussian window [32]. For nonstationary and multicomponent signals, this assumption is broken; thus, in this article, the authors use the Rényi entropy to choose such window width, for which the entropy attains minimum.

IV. Experiments

In order to quantitatively compare the method known from the literature to the proposed CVSS3 and the investigated EVSS1, the authors performed a numerical experiment. In both central processing unit (CPU) and GPU implementation in CVSS3, the authors use (16) as a second-order modulation operator. The synthetic signal was processed. Its sample rate was $f_s = 512$ Sa/s, and the definition was as follows:

$$x(t) = A(t)\exp\left(j2\pi\phi(t)\right) \tag{27}$$

where $A(t)$ and $\phi(t)$ are the amplitude and phase defined in $t \in [0,1]$, respectively

$$A(t) = 1 + 5t^2 + 7(1-t)^6 \tag{28}$$

$$\phi(t) = 240t - 2\exp\left(-2(t-0.2)\right)\sin(14\pi(t-0.2)). \tag{29}$$

The signal was processed using the technique proposed by Pham and Meignen [24] giving VSS2, VSS3, and VSS4. The proposed CVSS3 and the enhanced EVSS1 were used as well. The results are illustrated in Fig. 1.

As can be seen, apart from VSS2, all techniques gave comparable outcomes, and the distributions are strongly concentrated near the energy ridge. For $S_x^{h[2]}$, the continuity of the frequency ridge is broken, creating potential problems with

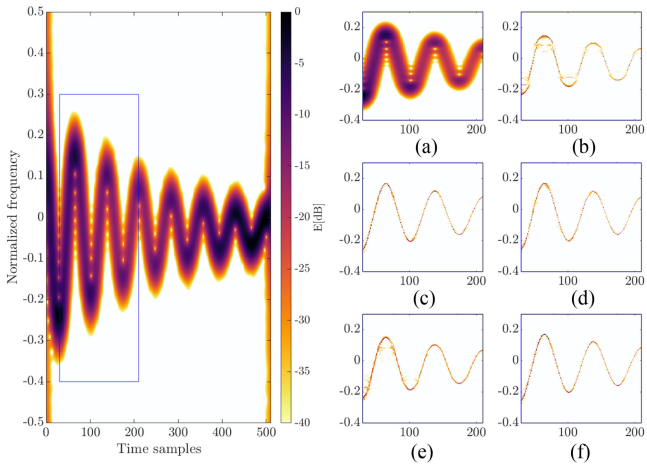

Fig. 1. Comparison of the synchrosqueezed STFT for different IF estimators: (a) STFT; (b) $S_x^{h[2]}$ – VSS2; (c) $S_x^{h[3]}$ – VSS3; (d) $S_x^{h[4]}$ – VSS4; (e) $S_x^{h[3']}$ – CVSS3; (f) $S_x^{h[1]}$ – EVSS1.

mode extraction. Oscillating and rapidly arising terms are sharp and continuous for $S_x^{h[3]}$, $S_x^{h[4]}$, $S_x^{h[3']}$, and $S_x^{h[1]}$, and as a factor of transform sharpening, the authors use the Rényi entropy—a classical measure of the TF concentration [33]. The γ-order Rényi entropy of any TF signal representation $\mathrm{TF}_x(t,\omega)$ is expressed in bits and given as

$$H_R^{\gamma}\left(\mathrm{TF}_x\right) = \frac{1}{1-\gamma}\log_2\left(\frac{\iint_{\mathbb{R}^2}\left(\mathrm{TF}_x(t,\omega)\right)^{\gamma}dtd\omega}{\iint_{\mathbb{R}^2}\left(\mathrm{TF}_x(t,\omega)\right)dtd\omega}\right). \tag{30}$$

In this article, for the computation of the Rényi entropy, the authors use its absolute value such that $|\mathrm{TF}_x(t,\omega)|$. For the signal from Fig. 1, the third-order Rényi entropy ($\gamma = 3$) and the processing time are listed in Table I.[1]The results clearly show that VSS3 and VSS4 are precise but inefficient from the perspective of real-time applications, such as radar and sonar. The proposed CVSS3 estimator allowed for the obtaining of comparable energy concentration (expressed by the Rényi entropy) approximately seven times faster than the method known from the literature. In contrast, for the enhanced first-order IF estimator, the authors obtained acceleration of approximately nine times while obtaining low Rényi entropy. The outcomes show the possibility of applying the proposed solution in real-time platforms using a simple CPU chip, as illustrated in Section VI

[1]For computation, the MATLAB 2021a environment was used: CPU Intel i7-9750H 2.6 GHz, 32-GB DDR4, SSD drive, and Windows 10.

$$\hat{\omega}_x^N(t,\omega) = F_x^{\mathcal{D}^3 h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{T}^2 h}(t,\omega) - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}^2 h}(t,\omega) - F_x^{\mathcal{D}^2 h}(t,\omega)F_x^{\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) - F_x^{\mathcal{D}^3 h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) + F_x^{\mathcal{D}^2 h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) + F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) \tag{25}$$

$$\hat{\omega}_x^D(t,\omega) = F_x^{\mathcal{D}^2 h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{T}^2 h}(t,\omega) - F_x^{\mathcal{D}^2 h}(t,\omega)F_x^{\mathcal{D}\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) - F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) + F_x^{\mathcal{D}h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{T}h}(t,\omega) - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}^2 h}(t,\omega)F_x^{h}(t,\omega) + F_x^{\mathcal{D}\mathcal{T}^2 h}(t,\omega)F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega)F_x^{h}(t,\omega) \tag{26}$$

TABLE I
THIRD-ORDER RÉNYI ENTROPY AND THE PROCESSING TIME FOR THE SIMULATED SIGNAL

TF_x	**STFT**	**VSS2**	**VSS3**	**VSS4**	**CVSS3**	**EVSS1**
$H_R^3(\lvert\mathrm{TF}_x\rvert)$	15.5240	11.2242	11.3535	11.1657	11.4636	11.3389
t_p [ms]	13.55	85.52	577.99	745.134	103.71	83.55

for example. On the other hand, more demanding systems may take advantage of the GPU technology allowing fast and precise operation even for critical infrastructure, as shown in the further part of this article.

The following experiment covered the possibility to reconstruct extracted modes for two components with a strongly oscillating frequency. For this purpose, the authors used the technique initially proposed in [34] and implemented in [35]. The method is based on the local minimum computation of the following formula:

$$E_x(\psi_1,\ldots,\psi_K) = \sum_{k=1}^{K} -\int_{\mathbb{R}} |\mathrm{TF}_x(t,\psi_k(t)|^2 + \lambda\psi_k'(t)^2 + \varepsilon\psi_k''(t)^2 dt \tag{31}$$

where K stands for the known number of modes and TF_x is the TF representation whose modes $\psi_k(t)$ are extracted. As shown in [28], λ and ε can be neglected in the analysis due to their irrelevant influence on ridge detection; thus, in this research, they were set to 0. As a quantitative measure of the reconstruction signal quality, the authors used the reconstruction quality factor (RQF) given as [36]

$$\mathrm{RQF} = 10\log_{10}\frac{\sum_{n=0}^{N-1}|x[n]|^2}{\sum_{n=0}^{N-1}|x[n]-\hat{x}[n]|^2} \tag{32}$$

where $x[n]$ is the reference waveform and $\hat{x}[n]$ is its reconstructed version (both of the length N); hence, RQF is the ratio (expressed in decibel) of the energy of the signal to the energy of the estimation error. A flowchart of the overall algorithm is shown in Fig. 2 to facilitate their understanding. The methodology was the same for both simulated and real-life signals.

Verification of the reconstruction quality was performed using a signal composed of two modes with an oscillating frequency. The signal of the following definition was processed:

$$x(t) = x_1(t) + x_2(t) = A_1(t)\exp(j2\pi\phi_1(t)) + A_2(t)\exp(j2\pi\phi_2(t)) \tag{33}$$

where

$$A_1(t) = A_2(t) = 1.7 - \exp\left(-\frac{t^2}{2\cdot 1.6}\right) \tag{34}$$

$$\phi_1(t) = 125t + 5\sin(6\pi t - \pi/4) \tag{35}$$

$$\phi_2(t) = -125t + 3\cos(10\pi t + \pi/4). \tag{36}$$

The results of mode extraction are presented in Fig. 3. As can be seen, the RQF arises with the signal-to-noise ratio (SNR) and is slightly smaller for the fast-oscillating component (the second mode). The quality of the extracted components is comparable for all methods. The investigation shows that the proposed CVSS3 and EVSS1 preserve the possibility to precisely reconstruct fast-oscillating components in reduced time. The additional experiment consisted of GPU-based implementation, and validation of all considered methods was performed and is described in the following sections.

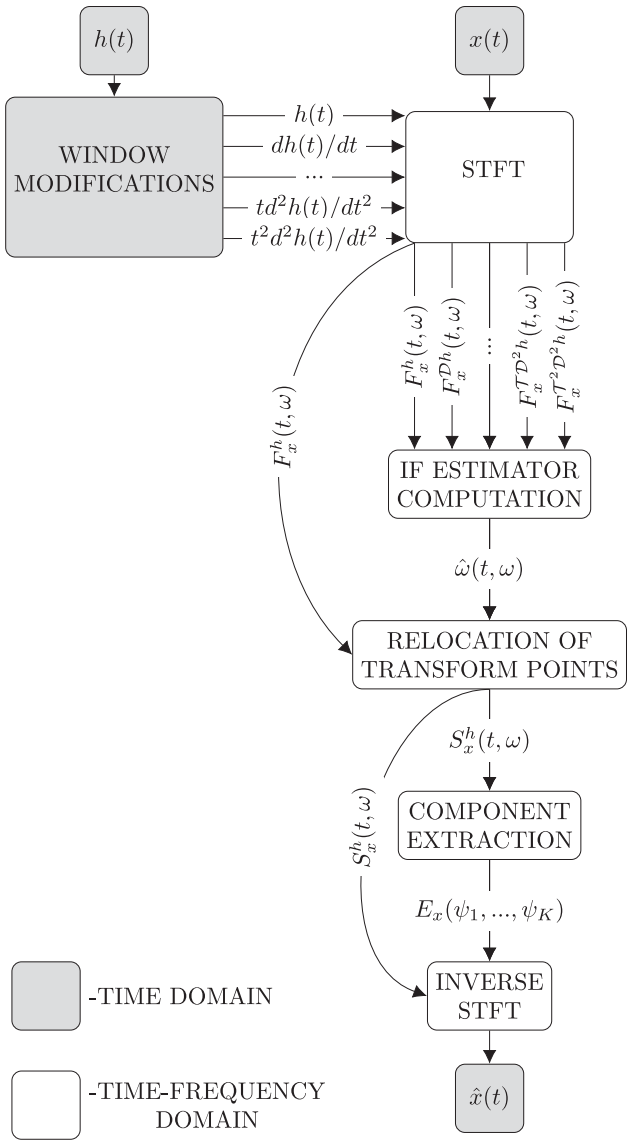

Fig. 2. Flowchart of signal decomposition using CVSS3 and EVSS1.

In the literature, one can find other signal separation methods that allow component disentangling when the signal nature is unknown and many terms can occur. The technique presented in this section can be compared to the representative one proposed

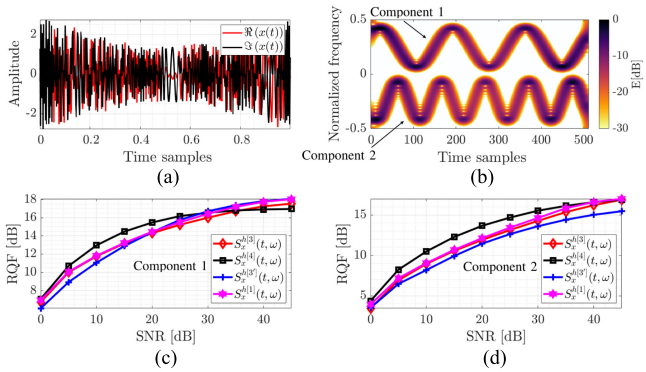

Fig. 3. Comparison of the RQF for the investigated methods for fast oscillating modes. (a) Time-domain signal. (b) Spectrogram. (c) RQF for the first component. (d) RQF for the second component.

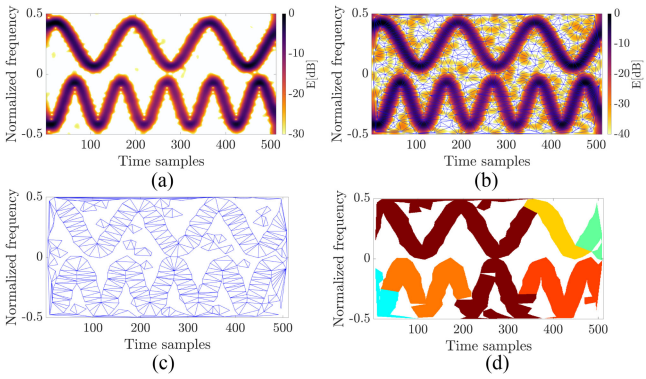

Fig. 4. Signal decomposition using the method proposed in [37]. (a) Signal spectrogram for SNR $= 20$ dB. (b) Energy distribution with superimposed triangles connecting spectrogram zeros. (c) Selected triangles fulfilling assumption on the edge length for which the component is considered as valid. (d) Extracted components—each component in a different color.

in [37]. In short, the procedure is based on connecting spectrogram zeros into triangles using the Delaunay triangulation. For noise, edges have a length from a known range. When the signal appears on a plane, the distance between the spectrogram zeros elongates around the component. Then, the triangles with at least one edge with a length out of the range known for the noise are connected, representing a single signal component. The mask composed of the entangled triangles is created and applied to the STFT (1—detected component, 0—otherwise), allowing the extraction of a particular component using the inverse STFT. More mathematical and technical details are presented in [37], and the radar application for a simple and single-component radar waveform is illustrated in [38]. Herein, the authors show the superiority of the VSS method for multicomponent signals with closely located modes. The triangulation of spectrogram zeros can fail in such a case, leading to incorrect signal separation. The rationale for such a statement is illustrated in Fig. 4.

As can be seen, even for high SNR (20 dB), the components are incorrectly separated. Due to local fluctuations resulting from noise and the proximity of components, the signal decomposition was faulty. The method detected multiple modes, as shown in different colors in Fig. 4(d). This example clarifies why the

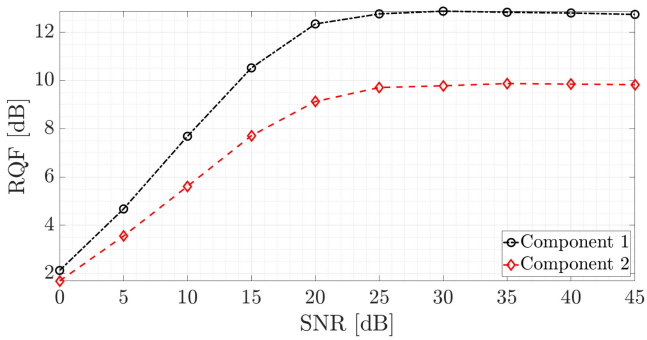

Fig. 5. RQF computed using the triangulation-based method for two components of (33). Processing for each mode was conducted separately.

precise concentration of transform points is essential in signal decomposition. In addition, the higher the oscillation rate, the worse the extraction capabilities. The main problem occurs for the second component with greater oscillating frequency, and in this case, the algorithm returned separate modes instead of one. Moreover, a part of one of the components is attached to the second one, which dramatically spoils reconstruction quality. Since the algorithm could not extract two defined modes, a more straightforward test was performed, where two components of (33) were processed separately. The RQF of each component was computed individually, averaged over 100 realizations, and the results are illustrated in Fig. 5. The experiments were carried out for 512 points of the FFT, and the arbitrary edge length for noise was assumed in the range $[0.2, 2.2]$, which is the default value proposed in [37].

The results clearly show that the method gives worse results even for separate analyses for two modes. The use of VSS allows for both separation of multicomponent signal components and obtaining higher reconstruction quality, especially for fast oscillating frequency modulation. In addition, the processing time is dramatically higher for the triangulation-based method. For example, extraction of a single component from (33) took over 6 s, while VSS allowed the reconstruction of two components in ca. 0.25 ms. This justifies using VSS and mode extraction to analyze multicomponent radar signals embedded in noise.

V. GPU Implementation

To calculate the synchrosqueezed STFT for VSS3, VSS4, CVSS3, and EVSS1, in the first place, it is necessary to compute all modified STFTs, namely, several transforms with modified analyzing windows. This, in turn, requires all mandatory modified windows given by $\mathcal{D}^n h = \frac{d^n h(t)}{dt^n}$ and $\mathcal{T}^n h = t^n \cdot h(t)$ to be calculated. Since the window samples are computed only once for a given window width and these computations are not crucial for software performance, it was decided that they would be performed in the prologue on a CPU. Then, the calculated coefficients are copied to the GPU memory.

Next, the input signal is transferred to the GPU. The need to transfer data between the CPU and the GPU is one of the fundamental drawbacks of GPGPUs. Despite the very high throughput provided by the latest generations of the peripheral component interconnect (PCI) express bus, the enormous computing power

of GPUs is partially lost on data transfer. Fortunately, in this case, the input signal is a one-dimensional vector of complex samples with a size typically not exceeding tens of kB. As a result, data transfer between the CPU and the GPU takes relatively little time in relation to the computation time, and its negative impact is very limited in this case.

All subsequent calculations are carried out on the GPU. One of the most important issues in GPGPU programming is to structure the algorithm in such a way as to emphasize its parallelism as much as possible. Then, such an algorithm needs to be efficiently mapped to the GPU architecture [39].

The code of the kernel function[2] can be divided into several blocks. The first and most computationally demanding is responsible for calculating the samples of all modified STFTs that appear in formulas (6), (15), (16), (19), (20), (25), and (26). The second block of the kernel function code is responsible for computing the IF estimates given by (23) or (24) depending on the chosen VSS variant. It should be emphasized that each thread only performs calculations for a single (t, ω) coordinate; thus, calculations performed by different threads are almost independent of each other and highly computationally intensive. These features enable efficient parallelization [40]. Moreover, both variants of the presented VSS allow a wide range of optimization techniques typical for CUDA technology to be used. These include the possibility of optimal use of many types of memory available in the CUDA and the loop unrolling or removing conditional statements techniques. The first one makes memory access operations very efficient. In turn, the second and third techniques minimize the instruction overhead [41], [42]. All these features cause the presented algorithms to map very well to the CUDA. Thus, the expected effects of implementing both presented VSS variants in the CUDA technology are promising.

In the last block of the kernel function, the VSS defined in (3) is computed. In practice, it comes down to adding the product $F_x^h(t, \omega')e^{j\omega'(t-t_0)}$ to the output array element that corresponds to the coordinates t and ω'.

The last stage depends on the adopted program execution path. If the data are further processed on the GPU, other kernel or library functions are called here. If the data are processed on the CPU, it must first be copied from the GPU to the CPU. Since the output data size is proportional not only to the signal length but also to the number of frequency bins, transfer operation may have a negative impact on the overall software performance.

Fortunately, in CUDA technology, there is a so-called CUDA stream mechanism that partially mitigates this disadvantage. This optimization technique introduces additional parallelism [39]. It enables concurrent calculations and data transfers between the GPU and the CPU in both directions for different data blocks. As a result, the CUDA stream mechanism allows one to copy one block of data from the GPU to the CPU, process another block and transfer one more block of data from the CPU to the GPU simultaneously. Moreover, it is worth paying attention to the fact that not only data transfers and computations may overlap, but also the computations themselves can be performed in parallel in different streams [43]. Thus, a situation may arise where the overall computation time (including data transfers between the CPU and the GPU) divided by the number of processed blocks may be shorter than the execution time of a single kernel function. The results of implementing both proposed forms of VSS, based on real-life signal processing, are presented in the next section.

[2]A kernel function is a parallel subroutine executed on the GPU by all threads according to the single-instruction multiple-thread architecture [39].

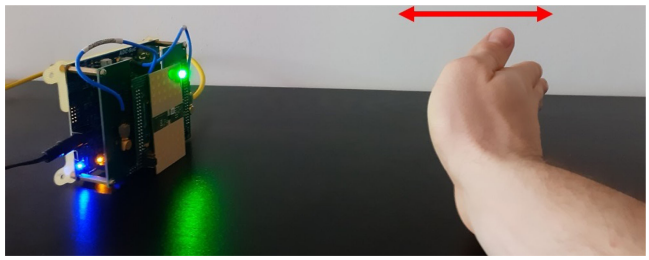

Fig. 6. Measurement geometry. A hand was moving toward and away from the radar front end, which allowed the cosine phase to be obtained.

VI. Real-Life Data Analysis

The CUDA software development and tests were carried out on the Supermicro SuperWorkstation 7047GR-TPRF running under the Linux Debian 10.0 operating system equipped with the RTX 2070 SUPER GPU and CUDA version 11.2. During the tests, three tools were used to measure the time and validate the measurements: the CUDA event mechanism, NVIDIA Visual Profiler, and a high-resolution CPU timer from the real-time extensions library. The computations were called repeatedly. Then, the shortest execution time was chosen to become independent from the instantaneous load changes of both the CPU and the GPU. Total time measurements for the application using CUDA streams consisted of performing calculations in three streams on 16 data blocks. Then, the measured time was divided by the number of processed blocks.

The tests were performed on two real-life signals. Unfortunately, in both cases, the ideal pattern of the signal was unknown. Thus, the RQF cannot be computed. The first signal under consideration is a micro-Doppler radar signature from a moving hand. The window width was optimized, and its normalized standard deviation was computed as $\sigma = 7.72$. The method of obtaining the signal is shown in Fig. 6. The radar sensor used in the experiment was the XY-DemoRad system developed by XY-Sensing Ltd. [44]. The echo was recorded by the radar using a signal bandwidth $B = 1$ GHz corresponding to 15 cm range resolution, carrier frequency $f_c = 24$ GHz, and a sweep repetition rate $f_{\mathrm{PRF}} = 1$ kHz. The waving hand, imitating a gesture used, for example, in the Google Soli radar [45], was located ca. 50 cm from the antenna aperture. The signal length was $N = 2048$ samples, and the same was the Fourier transform size. Such a scenario was intentionally selected to verify the efficiency of extracting oscillating modes using high-order synchrosqueezing. As can be seen in Fig. 7, all considered methods allowed the gesture's echo to be distinguished. Despite frequency oscillations, the energy was well gathered near the instantaneous frequency ridge without gaps (excluding local

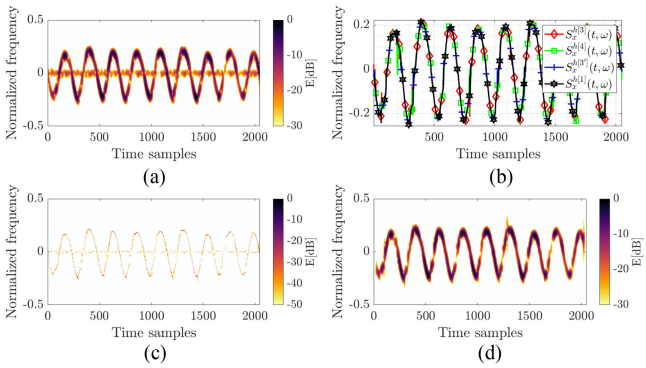

Fig. 7. Results of the real-life signal processing. Hand gesture example: (a) input signal; (b) modes extracted using the considered methods; (c) exemplary concentrated distribution (using EVSS1); and (d) spectrogram of the extracted mode. For each method, the FFT size was 2048 points.

TABLE II
PROCESSING TIME USING CPU AND GPU FOR THE GESTURE'S MICRO-DOPPLER SIGNAL

Method	CPU	GPU_{proc}	$\text{GPU}_{\text{total}}$
VSS3	6.32 s	6.61 ms	7.97 ms
VSS4	8.10 s	9.77 ms	10.52 ms
CVSS3	1.70 s	6.21 ms	7.82 ms
EVSS1	1.29 s	6.06 ms	7.78 ms

fluctuations of the signal amplitude), which is illustrated in Fig. 7(c) for EVSS1.

Table II presents the processing time for the techniques considered in this article. The GPU_{proc} refers to the execution time of the kernel function only and the $\text{GPU}_{\text{total}}$ to the total time (computation and data transfers between the CPU and the GPU) for software that utilizes CUDA streams. As can be seen, for the CPU platform, the processing time is the shortest for EVSS1 and is ~ 6.3 times shorter than for VSS4, which shows the significant superiority of the proposed approach. For such a case, the approaches proposed in this article work in real-time even for the CPU platform, since the processing time is shorter than the signal acquisition, which is unobtainable for VSS3 and VSS4. Processing time reduction is possible thanks to the lower computational cost of the proposed IF estimators. Thanks to fewer complex operations, the authors were able to process the signal from the high-resolution radar system using a middle-class computing machine, which shows the efficiency of the introduced VSS variants.

In addition, in the case of CUDA implementation, both methods described in this article (EVSS1 and CVSS3) are the fastest, but the differences from the reference methods (VSS3 and VSS4) are much lower than in the case of CPU versions. This is quite a typical feature of CUDA computations that, due to the specific memory architecture, kernel function overhead, and enormous computing power of modern GPUs, it is often possible to hide differences in the performance of serial versions of algorithms.

As a result, it is clearly visible that all the methods implemented in CUDA technology can operate in real-time. The assumption is still valid when considering the total GPU processing time. For such a case, the total processing time takes below 0.6% of the signal duration, which also allows the real-time operation to be performed.

It is worth mentioning here that the processing time of about 6 ms for the described signal and processing parameters corresponds to a computational performance of 5 trillion floating-point operations per second (TFLOPS). This coincides with almost 55% of the theoretical computational performance of the GPU used in the research. Due to the numerous limitations of the CUDA architecture in memory access, such a high value is challenging to achieve. Only algorithms with high computational intensity and appropriate memory access patterns can ensure such a high GPU utilization. This fact proves that all of the presented VSS variants are very well suited for implementation in CUDA technology.

The parameters of the extracted gesture signal were estimated. The signal model was assumed as follows:

$$x(t) = A(t)\exp\left(ja\cos\left(2\pi ft + \phi_0\right)\right) \tag{37}$$

where $A(t)$ is the amplitude of the signal (assumed to be constant and unitary), a stands for the oscillation of frequency in the TF domain resulting from the Doppler effect, f is the frequency of this oscillation, and ϕ_0 is its initial phase. Approximating the extracted micro-Doppler signature as a pure cosine frequency-modulated signal, the initial phase ϕ_0 and the frequency of the oscillation f were estimated using the matched filtering principle [46] similarly to the approach proposed in [47]. In general, the method computes the output of the input signal (extracted gesture signature) with a matched filter of all parameters for a given range. The final estimate is the maximum value from this operation. The results are illustrated in Fig. 8.

The estimated parameters of the oscillating component are as follows: $\hat{f} = 4.4803$ Hz and $\hat{\phi_0} = 0.883888$ rad [see Fig. 8(a)]. Of course, the analyzed signal is composed of many components and does not represent a pure tone; thus, in the vicinity of the maximum peak [see Fig. 8(b)], there are local disturbances that manifest themselves as local maxima. In fact, the actual parameters of the signal are unknown, but for visualization, the estimated instantaneous frequency curve is superimposed on the spectrogram, which is shown in Fig. 9. The results clearly show a convergence of the real-life signal and its estimate.

The second test covered analysis of an NLFM real-life radar pulse originating from the S-band air traffic control (ATC) radar located at the Warsaw Chopin Airport. The data collection was done using a National Instruments PCI eXtensions for Instrumentation vector signal analyzer, ensuring good signal dynamics and easy parameter adjustment. A receiving antenna pointed directly toward the transmitter was ca. 7 km away from the ATC radar and connected to a power amplifier and then to the signal downconverter and digitizer. Raw signal samples were stored and processed offline. The nonlinearity of the signal favors verifying the correct operation of the algorithms discussed, and a time regime resulting from the need for pulse-to-pulse processing makes this case appropriate for validation. Additionally, for the experiment, a low SNR signal was selected to test the quality of the pulse reconstruction under challenging conditions, as the

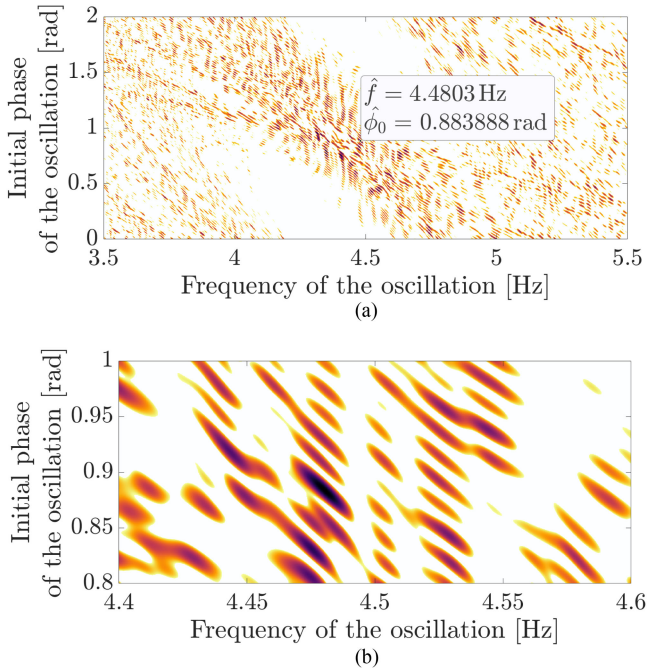

Fig. 8. Estimated parameters of the extracted gesture signal. (a) Whole map. (b) Closeup on the region with the maximum value of matched filtering.

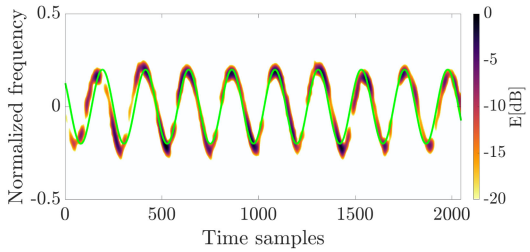

Fig. 9. Spectrogram of the extracted micro-Doppler signature with superimposed estimated instantaneous frequency curve (green).

assumption of high SNR cannot always be made. The signal was sampled with a rate $f_s = 40$ MSa/s, decimated, and then processed offline for 512 points of the Fourier transform and the exact size of the signal vector. In this case, due to solid nonlinearity at the ends of the pulse, the window width was chosen manually because, in the automatic process, the Rényi entropy minimization resulted in the concentration of only the linear part of the chirp while smearing its nonlinear parts. Thus, the window width was selected to allow approximately constant resolution for the entire chirp to be obtained, and its normalized standard deviation was $\sigma = 6.5$.

Fig. 10 presents the outcomes in the form of a spectrogram, exemplary synchrosqueezed spectrogram (in this case using CVSS3), extracted modes, and the reconstructed pulse. Particularly, in Fig. 10(a), the multipath effect is apparent, which manifests by the coexistence of the delayed pulse disturbing the direct one. The extracted modes are largely similar, and the main differences are visible at the ends of the pulse, where the instantaneous frequency arises almost vertically. The reconstructed pulse precisely maps the waveform without noise and interference. In the TF domain, the result is illustrated only for

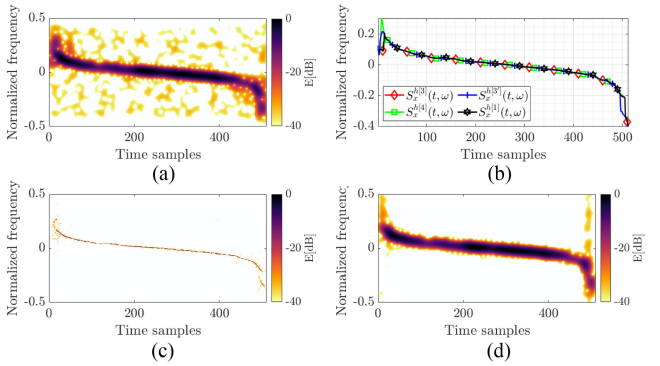

Fig. 10. Results of the real-life signal processing. NLFM radar pulse example: (a) input signal; (b) modes extracted using the considered methods; (c) exemplary concentrated distribution (using CVSS3); and (d) spectrogram of the extracted mode.

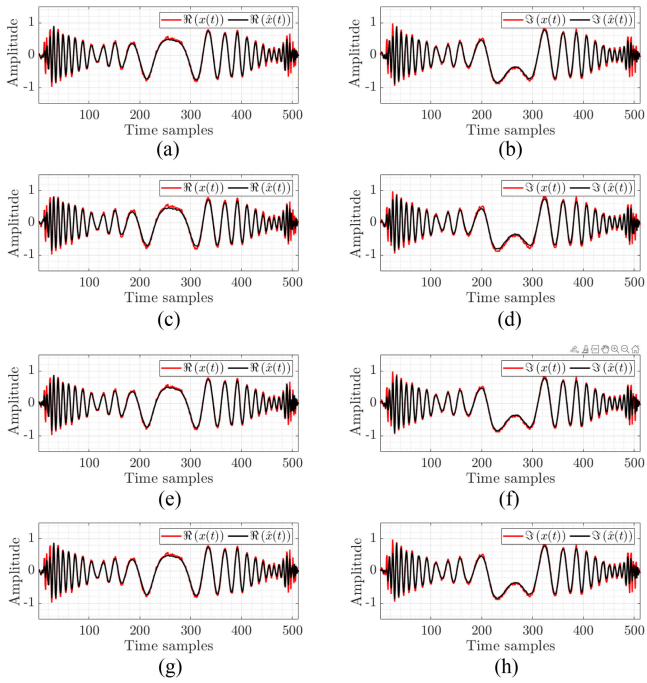

Fig. 11. Results of the real-life signal processing. NLFM radar pulse example and the input ($x(t)$) and the reconstructed ($\hat{x}(t)$) signals: (a) and (b) real and imaginary parts for VSS3; (c) and (d) real and imaginary parts for VSS4; (e) and (f) real and imaginary parts for the CVSS3; and (g) and (h) real and imaginary parts for EVSS1.

CVSS3 as an example. A comparison of the received pulse with superimposed reconstructions for each high-order VSS considered in this article is depicted in Fig. 11. Additionally, the residuals defined as the difference between the input signal $x(t)$ and its reconstruction $\hat{x}(t)$ are presented in Fig. 12. As can be seen, the residuals are composed mainly of a weak multipath copy of the pulse and the noise. The effect at the ends of the pulse was aforementioned and resulted from the inability to concentrate vertical terms by the methods in question. However, this effect partially comes from the spectrum leakage and has negligible influence on the pulse reconstruction quality.

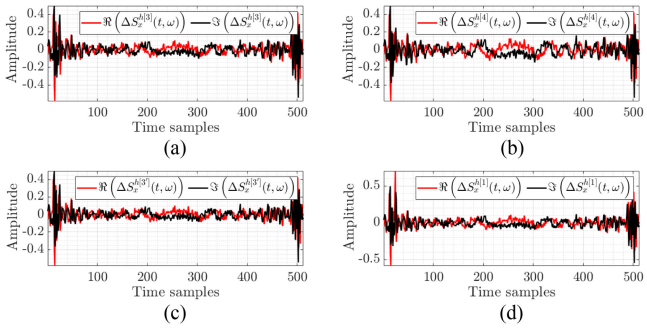

Fig. 12. Results of the real-life signal processing. NLFM radar pulse example and the residuals of the input ($x(t)$) and the reconstructed ($\hat{x}(t)$) signals: (a) VSS3; (b) VSS4; (c) CVSS3; and (d) EVSS1.

TABLE III
PROCESSING TIME USING CPU AND GPU FOR THE NLFM RADAR PULSE SIGNAL

Method	CPU	GPU_{proc}	$\text{GPU}_{\text{total}}$
VSS3	519.98 ms	0.741 ms	0.761 ms
VSS4	630.73 ms	1.125 ms	1.233 ms
CVSS3	95.74 ms	0.411 ms	0.353 ms
EVSS1	60.99 ms	0.413 ms	0.355 ms

The processing time for the CPU and GPU platforms is listed in Table III. When comparing VSS4 and EVSS1, the authors obtained over tenfold acceleration in the CPU processing, while the extracted modes are almost identical. The pulse duration was ca. $T \approx 10\ \mu$s, and for this particular ATC radar, the pulse repetition frequency (PRF) was in the range $f_{\text{PRF}} \in (500, 1000)$ Hz (typically 825 Hz). Thus, assuming the highest PRF, the processing time should be below 1 ms to obtain real-time operation and signature extraction in pulse repetition interval (PRI). In such a case, a CPU platform is insufficient to perform the processing in real-time, and one may need GPU technology to accomplish the system requirements.

As can be seen, the total GPU processing time of the two methods proposed in this article is shorter than 0.36 ms, which means that it takes less than 36% of the signal PRI. Thus, it is possible to perform all computations in real-time. On the other hand, CUDA implementation for both of the reference methods works noticeably slower in this case (VSS3 more than twice and VSS4 about 3.5 times slower). It is worth mentioning that such a significant change in relation to the previously analyzed signal results from different processing parameters, including a different FFT size and a different window size. In the case of VSS3, the GPU processing time is still shorter than the signal PRI, but the margin is much smaller in this case, while for VSS4, real-time computation is no longer possible.

Fig. 13 presents how GPU implementations of both proposed variants of VSS behave for various processing parameters. In order to make the presented results independent of the signal length, it was decided that on both graphs, the Y-axis would correspond to the processing time of one signal sample. Thanks to this, to estimate the time needed to process a given signal, it is enough to multiply the value read from the graph by the signal length. It should be noted here that presented timings apply to the kernel execution time only and do not take into account data transfers between the GPU and the CPU. Fig. 13(a) depicts the processing time of one signal sample against the window width for different numbers of frequency bins. On the other hand, Fig. 13(b) shows the processing time of one signal sample against the number of frequency bins for different window widths.

As can be seen, both described variants of VSS operate virtually identically. It is mainly because the most time-consuming (around 60%) is the computation of the modified STFTs, which for both variants are the same. Additionally, for both variants, the number of accesses to global and shared memories is the same. The only difference is in the number of floating-point operations necessary to calculate IF estimates, but it is small enough to be hidden by the enormous computational power of the GPU. Moreover, with the linear growth of the window width and the number of frequency bins, the processing time also increases linearly. It means that the CUDA implementations of both presented variants of VSS work very predictably and are stable and highly efficient.

VII. CONCLUSION

This aricle presented two novel IF estimators with application to real-time VSS. The research supported by simulated and real-life signals showed the efficiency and precision of the developed frequency reassignment operators in the context of signal concentration in the TF domain. Compared to the techniques known from the literature, the proposed ones allowed for the obtaining of comparable outcomes almost ten times faster using a simple CPU platform. In addition, for exacting solutions, the GPU implementation can be used, which was presented in this article using a real-life NLFM radar pulse. The main conclusion from the research presented in this article is that the enhanced first-order frequency reassignment operator (with the lowest computational complexity) deduced from the nonlinear signal phase allows intense signal concentration after the VSS process in reduced time compared to high-order frequency reassignment operators. Similar outcomes were obtained for CVSS3, for which the processing time and the precision were similar as in EVSS1. For practical and demanding applications, the use of first-order VSS is more efficient and comparably accurate. Thus, from the practical perspective of radar remote sensing, high-order VSS is valuable and important; however, it can be replaced by faster methods like those proposed in this article. An additional conclusion from the research is that VSS algorithms map very well to the CUDA, which allows for very efficient calculations that can be performed in real-time.

Further research covers finding new applications for CVSS3 and EVSS1 and their implementation on embedded systems able to work in various scenarios, such as real-time signal decomposition on small radar platforms like the one used in the experiment. As an extension of the GPU implementation, the authors want to implement classification processing to extract

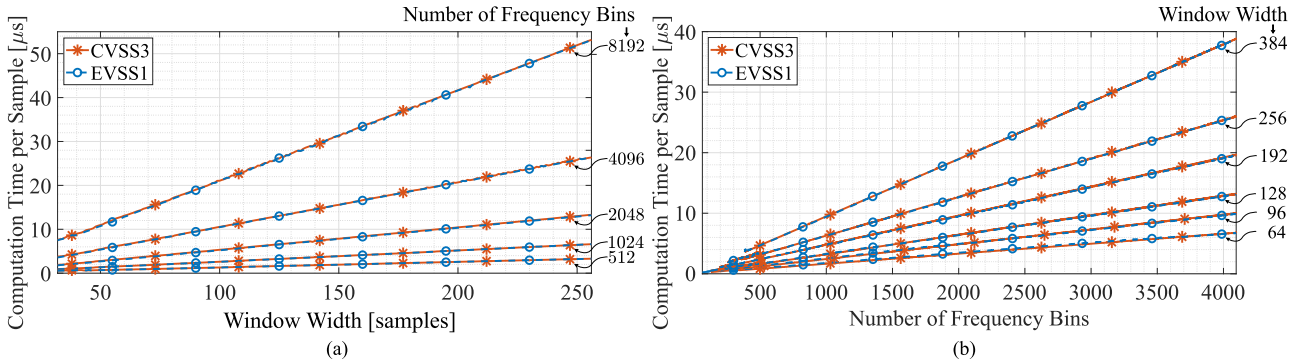

Fig. 13. Processing time for different STFT parameters: (a) window width; (b) number of frequency bins.

the signal and then classify, e.g., radar pulses, gestures, and vehicles.

Appendix
Derivations for the Angular Jerk, CR, and Instantaneous Frequency Estimators

In this appendix, derivations of the angular jerk, CR, and the third-order instantaneous frequency estimator are provided. For more details regarding numerical stability, see [30].

Signal model

$$x(t) = A_x(t)\exp\left(j\phi_x(t)\right) \tag{38}$$

where the amplitude is given as

$$A_x(t) = \mathcal{A}_x \cdot \exp\left(-\frac{(t-t_x)^2}{2T_x^2} - \Delta_x\frac{(t-t_x)^3}{3T_x^2}\right) \tag{39}$$

and the phase is defined by

$$\phi_x(t) = \varphi_x + \omega_x t + \alpha_x t^2/2 + \beta_x t^3/3 \tag{40}$$

where

$\mathcal{A}_x$ constant envelope;
$-\frac{(t-t_x)^2}{2T_x^2}$ second-order amplitude modulation component;
t_x time shift;
T_x duration;
$-\Delta_x\frac{(t-t_x)^3}{3T_x^2}$ third-order modulation of the amplitude;
$j^2 = -1$;
φ_x initial phase;
ω_x angular frequency;
α_x CR;
β_x angular jerk.

The first-order derivative of the signal with respect to time gives

$$\frac{dx(t)}{dt} = \left(a_x t^2 + b_x t + c_x\right)x(t) \tag{41}$$

where

$$\begin{aligned} a_x &= \frac{-\Delta_x}{T} + j\beta_x \\ b_x &= -\frac{1-2\Delta_x t_x}{T_x^2} + j\alpha_x \\ c_x &= \frac{t_x - \Delta_x t_x^2}{T_x^2} + j\omega_x. \end{aligned} \tag{42}$$

The STFT definition using the analysis window $h(t)$

$$\begin{aligned} F_x^h(t,\omega) &= \int_{\mathbb{R}} x(\tau)h^*(\tau-t)e^{-j\omega\tau}d\tau \\ &= e^{-j\omega t}\int_{\mathbb{R}} x(\tau-t)h^*(\tau)e^{j\omega\tau}d\tau \end{aligned} \tag{43}$$

where $(\cdot)^*$ – complex conjugate. Differentiating the STFT with respect to time once leads to

$$\begin{aligned} F_x^{\mathcal{D}h}(t,\omega) &= \frac{\partial F_x^h}{\partial t}(t,\omega) = \underbrace{\int_{\mathbb{R}} x(\tau)\frac{dh^*}{dt}(\tau-t)e^{-j\omega\tau}d\tau}_{\text{STFT with the differentiated window}} \\ &= -j\omega F_x^h(t,\omega) - e^{-j\omega t}\int_{\mathbb{R}}\frac{dx}{dt}(\tau-t)h^*(\tau)e^{j\omega\tau}d\tau. \end{aligned} \tag{44}$$

At this point, one can introduce modifications of the STFT. $F_x^{\mathcal{D}h}(t,\omega)$ is the STFT computed using the differentiated window $\mathcal{D}^n h = \frac{d^n h(t)}{dt^n}$. Analogously, the window multiplied by the time ramp can also be used for the STFT. Assuming $\mathcal{T}^n h = t^n \cdot h(t)$, one can substitute differentiated signal (41) to the right-hand form of (44), obtaining

$$\begin{aligned} F_x^{\mathcal{D}h}(t,\omega) = &-a_x F_x^{\mathcal{T}^2 h}(t,\omega) - (2a_x t + b_x)F_x^{\mathcal{T}h}(t,\omega) \\ &- (a_x t^2 + b_x t + c_x + j\omega)F_x^h(t,\omega) \end{aligned} \tag{45}$$

where a_x, b_x, and c_x are given by (42). Because our interest is to obtain the third-order phase derivative, (45) can be differentiated with respect to time once and then twice, giving respectively (the following equations can be interpreted as the STFT with the window differentiated twice and three times – see the superscript,

e.g., $\mathcal{D}^2h$ or $\mathcal{D}^3h$ and their combinations)

$$\begin{aligned} F_x^{\mathcal{D}^2h}(t,\omega) = & -a_x F_x^{\mathcal{T}^2\mathcal{D}h}(t,\omega) - (2a_x t + b_x) F_x^{\mathcal{T}\mathcal{D}h}(t,\omega) \\ & - (a_x t^2 + b_x t + c_x + j\omega) F_x^{\mathcal{D}h}(t,\omega) \end{aligned} \tag{46}$$

$$\begin{aligned} F_x^{\mathcal{D}^3h}(t,\omega) = & -a_x F_x^{\mathcal{T}^2\mathcal{D}^2h}(t,\omega) - (2a_x t + b_x) F_x^{\mathcal{T}\mathcal{D}^2h}(t,\omega) \\ & - (a_x t^2 + b_x t + c_x + j\omega) F_x^{\mathcal{D}^2h}(t,\omega). \end{aligned} \tag{47}$$

Combining (45)–(47) into a linear system with three variables p_x, q_x, and r_x yields

$$\begin{cases} F_x^{\mathcal{D}h}(t,\omega) = r_x F_x^{\mathcal{T}^2h}(t,\omega) + q_x F_x^{\mathcal{T}h}(t,\omega) + p_x F_x^{h}(t,\omega) \\ F_x^{\mathcal{D}^2h}(t,\omega) = r_x F_x^{\mathcal{T}^2\mathcal{D}h}(t,\omega) + q_x F_x^{\mathcal{T}\mathcal{D}h}(t,\omega) \\ \qquad\qquad +p_x F_x^{\mathcal{D}h}(t,\omega) \\ F_x^{\mathcal{D}^3h}(t,\omega) = r_x F_x^{\mathcal{T}^2\mathcal{D}^2h}(t,\omega) + q_x F_x^{\mathcal{T}\mathcal{D}^2h}(t,\omega) \\ \qquad\qquad +p_x F_x^{\mathcal{D}^2h}(t,\omega) \end{cases}$$

where $p_x = -a_x t^2 - b_x t - c_x - j\omega$, $q_x = -(2a_x t + b_x)$, and $r_x = -a_x$ are the unknown variables. Let us return to the definition of a_x, b_x, and c_x

$$\begin{aligned} a_x &= \frac{-\Delta_x}{T} + j\beta_x \\ b_x &= -\frac{1 - 2\Delta_x t_x}{T_x^2} + j\alpha_x \\ c_x &= \frac{t_x - \Delta_x t_x^2}{T_x^2} + j\omega_x. \end{aligned} \tag{48}$$

As can be seen, the phase signal parameters are attached to the imaginary part of these variables. If we solve the system and compute the imaginary part of its solution, we obtain the following.

1) Angular jerk estimator

$$\hat{\beta}_x(t,\omega) = \Im\left[\frac{\hat{\beta}_x^N(t,\omega)}{\hat{\beta}_x^D(t,\omega)}\right] \tag{49}$$

where

$$\begin{aligned} \hat{\beta}_x^N(t,\omega) = & F_x^{\mathcal{T}h}(t,\omega)\left[F_x^{\mathcal{D}^2h}(t,\omega)\right]^2 \\ & - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}h}(t,\omega) \\ & - F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) F_x^{h}(t,\omega) F_x^{\mathcal{D}^2h}(t,\omega) \\ & + F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega)\left[F_x^{\mathcal{D}h}(t,\omega)\right]^2 \\ & - F_x^{\mathcal{D}^3h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) F_x^{\mathcal{D}h}(t,\omega) \\ & + F_x^{\mathcal{D}^3h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{h}(t,\omega), \end{aligned} \tag{50}$$

$$\begin{aligned} \hat{\beta}_x^D(t,\omega) = & F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) \\ & - F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) \\ & - F_x^{\mathcal{D}h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) \\ & + F_x^{\mathcal{D}h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) \\ & - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) F_x^{h}(t,\omega) \\ & + F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) F_x^{h}(t,\omega). \end{aligned} \tag{51}$$

2) CR estimator

$$\hat{\alpha}_x(t,\omega) = \Im\left[\frac{\hat{\alpha}_x^N(t,\omega)}{\hat{\alpha}_x^D(t,\omega)}\right] \tag{52}$$

where

$$\begin{aligned} \hat{\alpha}_x^N(t,\omega) = & -F_x^{\mathcal{T}^2h}(t,\omega)\left[F_x^{\mathcal{D}^2h}(t,\omega)\right]^2 \\ & + F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}h}(t,\omega) \\ & + F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) F_x^{h}(t,\omega) F_x^{\mathcal{D}^2h}(t,\omega) \\ & - F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega)\left[F_x^{\mathcal{D}h}(t,\omega)\right]^2 \\ & + F_x^{\mathcal{D}^3h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}h}(t,\omega) \\ & - F_x^{\mathcal{D}^3h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{h}(t,\omega) \end{aligned} \tag{53}$$

$$\begin{aligned} \hat{\alpha}_x^D(t,\omega) = & F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) \\ & - F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) \\ & - F_x^{\mathcal{D}h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) \\ & + F_x^{\mathcal{D}h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) \\ & - F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) F_x^{h}(t,\omega) \\ & + F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) F_x^{h}(t,\omega). \end{aligned} \tag{54}$$

3) Instantaneous frequency estimator

$$\hat{\omega}_x(t,\omega) = \Im\left[\frac{\hat{\omega}_x^N(t,\omega)}{\hat{\omega}_x^D(t,\omega)}\right] \tag{55}$$

where

$$\begin{aligned} \hat{\omega}_x^N(t,\omega) = & -F_x^{\mathcal{D}^3h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) \\ & + F_x^{\mathcal{D}\mathcal{T}h}(t,\omega) F_x^{\mathcal{D}h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) \\ & + F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) \\ & + F_x^{\mathcal{D}^3h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) \\ & - F_x^{\mathcal{D}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{T}h}(t,\omega) \\ & - F_x^{\mathcal{D}h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}^2h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}h}(t,\omega) \end{aligned} \tag{56}$$

$$\begin{aligned}\hat{\omega}_x^D(t,\omega) = {} & F_x^{\mathcal{D}^2 h}(t,\omega) F_x^{\mathcal{D}\mathcal{T} h}(t,\omega) F_x^{\mathcal{T}^2 h}(t,\omega) \\ & - F_x^{\mathcal{D}^2 h}(t,\omega) F_x^{\mathcal{D}\mathcal{T}^2 h}(t,\omega) F_x^{\mathcal{T} h}(t,\omega) \\ & - F_x^{\mathcal{D} h}(t,\omega) F_x^{\mathcal{T}^2 h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T} h}(t,\omega) \\ & + F_x^{\mathcal{D} h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2 h}(t,\omega) F_x^{\mathcal{T} h}(t,\omega) \\ & - F_x^{\mathcal{D}\mathcal{T} h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T}^2 h}(t,\omega) F_x^{h}(t,\omega) \\ & + F_x^{\mathcal{D}\mathcal{T}^2 h}(t,\omega) F_x^{\mathcal{D}^2\mathcal{T} h}(t,\omega) F_x^{h}(t,\omega). \quad (57)\end{aligned}$$

ACKNOWLEDGMENT

The authors would like to thank Prof. Piotr Samczyński for consultations that helped prepare the article to its current form.

REFERENCES

[1] M. Malanowski, K. Kulpa, M. Baczyk, and Ł. Maślikowski, "Noise vs. deterministic waveform radar—Possibilities and limitations," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 35, no. 10, pp. 8–19, Oct. 2020.

[2] Y. Zhao and Y. Su, "Synchrosqueezing phase analysis on micro-Doppler parameters for small UAVs identification with multichannel radar," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 3, pp. 411–415, Mar. 2020.

[3] C. Karabacak, S. Z. Gurbuz, A. C. Gurbuz, M. B. Guldogan, G. Hendeby, and F. Gustafsson, "Knowledge exploitation for human micro-Doppler classification," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 10, pp. 2125–2129, Oct. 2015.

[4] Y. Ding and J. Tang, "Micro-Doppler trajectory estimation of pedestrians using a continuous-wave radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 9, pp. 5807–5819, Sep. 2014.

[5] R. J. Javier and Y. Kim, "Application of linear predictive coding for human activity classification based on micro-Doppler signatures," *IEEE Geosci. Remote Sens. Lett.*, vol. 11, no. 10, pp. 1831–1834, Oct. 2014.

[6] Y. Kim, S. Ha, and J. Kwon, "Human detection using Doppler radar based on physical characteristics of targets," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 2, pp. 289–293, Feb. 2015.

[7] M. Gustafsson, Å. Andersson, T. Johansson, S. Nilsson, A. Sume, and A. Örbom, "Extraction of human micro-Doppler signature in an urban environment using a 'sensing-behind-the-corner' radar," *IEEE Geosci. Remote Sens. Lett.*, vol. 13, no. 2, pp. 187–191, Feb. 2016.

[8] F. Fioranelli, M. Ritchie, and H. Griffiths, "Classification of unarmed/armed personnel using the NetRAD multistatic radar for micro-Doppler and singular value decomposition features," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 9, pp. 1933–1937, Sep. 2015.

[9] S. S. Ram, C. Christianson, Y. Kim, and H. Ling, "Simulation and analysis of human micro-Dopplers in through-wall environments," *IEEE Trans. Geosci. Remote Sens.*, vol. 48, no. 4, pp. 2015–2023, Apr. 2010.

[10] J. M. Munoz-Ferreras, F. Perez-Martinez, J. Calvo-Gallego, A. Asensio-Lopez, B. P. Dorta-Naranjo, and A. Blanco-del Campo, "Traffic surveillance system based on a high-resolution radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 46, no. 6, pp. 1624–1633, Jun. 2008.

[11] S. Wang, Z. Li, Y. Zhang, B. Cheong, and L. Li, "Implementation of adaptive pulse compression in solid-state radars: Practical considerations," *IEEE Geosci. Remote Sens. Lett.*, vol. 12, no. 10, pp. 2170–2174, Oct. 2015.

[12] M. Malanowski, M. Żywek, M. Płotka, and K. Kulpa, "Passive bistatic radar detection performance prediction considering antenna patterns and propagation effects," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–16, 2022, Art no. 5101516, doi: 10.1109/TGRS.2021.3069636.

[13] Z. Wu, C. Wang, Y. Li, and Z. Zhou, "Extended target estimation and recognition based on multimodel approach and waveform diversity for cognitive radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 60, pp. 1–14, 2022, Art. no. 5101014, doi: 10.1109/TGRS.2021.3065335.

[14] Z. Cao, J. Li, C. Song, Z. Xu, and X. Wang, "Compressed sensing-based multitarget CFAR detection algorithm for FMCW radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 59, no. 11, pp. 9160–9172, Nov. 2021.

[15] J. Mei, Z. Li, C. Yi, J. Zhang, Y. Wang, and M. Sun, "Terahertz radar with high resolution range profile," in *Proc. 46th Int. Conf. Infrared, Millimeter THz Waves*, 2021, pp. 1–2.

[16] X. Li, Z. Du, Y. Huang, and Z. Tan, "A deep translation (GAN) based change detection network for optical and SAR remote sensing images," *ISPRS J. Photogramm. Remote Sens.*, vol. 179, pp. 14–34, 2021.

[17] K. Kodera, C. De Villedary, and R. Gendrin, "A new method for the numerical analysis of non-stationary signals," *Phys. Earth Planet. Interiors*, vol. 12, no. 2, pp. 142–150, 1976.

[18] F. Auger and P. Flandrin, "Improving the readability of time-frequency and time-scale representations by the reassignment method," *IEEE Trans. Signal Process.*, vol. 43, no. 5, pp. 1068–1089, May 1995.

[19] F. Auger *et al.*, "Time-frequency reassignment and synchrosqueezing: An overview," *IEEE Signal Process. Mag.*, vol. 30, no. 6, pp. 32–41, Jun. 2013.

[20] M. Płotka, K. Abratkiewicz, M. Malanowski, P. Samczyński, and K. Kulpa, "The use of the reassignment technique in the time-frequency analysis applied in VHF-based passive forward scattering radar," *Sensors*, vol. 20, no. 12, 2020, Art. no. 3434.

[21] P. Flandrin, *Explorations in Time-Frequency Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 2018.

[22] T. Oberlin, S. Meignen, and V. Perrier, "Second-order synchrosqueezing transform or invertible reassignment? Towards ideal time-frequency representations," *IEEE Trans. Signal Process.*, vol. 63, no. 5, pp. 1335–1344, May 2015.

[23] D. Fourer, F. Auger, K. Czarnecki, S. Meignen, and P. Flandrin, "Chirp rate and instantaneous frequency estimation: Application to recursive vertical synchrosqueezing," *IEEE Signal Process. Lett.*, vol. 24, no. 11, pp. 1724–1728, Nov. 2017.

[24] D. Pham and S. Meignen, "High-order synchrosqueezing transform for multicomponent signals analysis-with an application to gravitational-wave signal," *IEEE Trans. Signal Process.*, vol. 65, no. 12, pp. 3168–3178, Dec. 2017.

[25] J. M. Miramont, M. A. Colominas, and G. Schlotthauer, "Voice jitter estimation using high-order synchrosqueezing operators," *IEEE/ACM Trans. Audio, Speech, Lang. Process.*, vol. 29, pp. 527–536, 2021, doi: 10.1109/TASLP.2020.3045265.

[26] W. Liu, S. Cao, Z. Wang, K. Jiang, Q. Zhang, and Y. Chen, "A novel approach for seismic time-frequency analysis based on high-order synchrosqueezing transform," *IEEE Geosci. Remote Sens. Lett.*, vol. 15, no. 8, pp. 1159–1163, Aug. 2018.

[27] W. Liu, W. Chen, and Z. Zhang, "A novel fault diagnosis approach for rolling bearing based on high-order synchrosqueezing transform and detrended fluctuation analysis," *IEEE Access*, vol. 8, pp. 12533–12541, 2020.

[28] S. Meignen, D. Pham, and S. McLaughlin, "On demodulation, ridge detection, and synchrosqueezing for multicomponent signals," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 2093–2103, Aug. 2017.

[29] S. Meignen, T. Oberlin, and D. Pham, "Synchrosqueezing transforms: From low- to high-frequency modulations and perspectives," *Comptes Rendus Phys.*, vol. 20, no. 5, pp. 449–460, Jul. 2019.

[30] K. Abratkiewicz, "On the instantaneous angular jerk estimation in the time-frequency domain," *IEEE Signal Process. Lett.*, vol. 28, pp. 798–802, 2021, doi: 10.1109/LSP.2021.3068714.

[31] K. Abratkiewicz, "Double-adaptive chirplet transform for radar signature extraction," *IET Radar, Sonar Navig.*, vol. 14, no. 11, pp. 1463–1474, Oct. 2020.

[32] S. Meignen, M. Colominas, and D. Pham, "On the use of Rényi entropy for optimal window size computation in the short-time Fourier transform," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2020, pp. 5830–5834.

[33] R. G. Baraniuk, P. Flandrin, A. J. E. M. Janssen, and O. J. J. Michel, "Measuring time-frequency information content using the Rényi entropies," *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1391–1409, Apr. 2001.

[34] R. A. Carmona, W. L. Hwang, and B. Torresani, "Characterization of signals by the ridges of their wavelet transforms," *IEEE Trans. Signal Process.*, vol. 45, no. 10, pp. 2586–2590, Oct. 1997.

[35] G. Thakur, E. Brevdo, N. S. Fučkar, and H. Wu, "The synchrosqueezing algorithm for time-varying spectral analysis: Robustness properties and new paleoclimate applications," *Signal Process.*, vol. 93, no. 5, pp. 1079–1094, 2013.

[36] D. Fourer, F. Auger, and P. Flandrin, "Recursive versions of the Levenberg-Marquardt reassigned spectrogram and of the synchrosqueezed STFT," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process.*, 2016, pp. 4880–4884.

[37] P. Flandrin, "Time-frequency filtering based on spectrogram zeros," *IEEE Signal Process. Lett.*, vol. 22, no. 11, pp. 2137–2141, Nov. 2015.

[38] K. Abratkiewicz, P. J. Samczyński, R. Rytel-Andrianik, and Z. Gajo, "Multipath interference removal in receivers of linear frequency modulated radar pulses," *IEEE Sens. J.*, vol. 21, no. 17, pp. 19000–19012, Sep. 2021.
[39] *CUDA C Programming Guide, NVIDIA Corp.*, Santa Clara, CA, USA, 2021.
[40] S. Cook, *CUDA Programming: A Developer's Guide to Parallel Computing With GPUs*. San Mateo, CA, USA: Morgan Kaufmann, 2012.
[41] *CUDA C Best Practices Guide, NVIDIA Corp.*, Santa Clara, CA, USA, 2021.
[42] D. Kirk and W. Hwu, *Programming Massively Parallel Processors: A Hands-On Approach*. San Mateo, CA, USA: Morgan Kaufmann, 2013.
[43] H. Li, D. Yu, A. Kumar, and Y.-C. Tu, "Performance modeling in CUDA streams—A means for high-throughput data processing," in *Proc. IEEE Int. Conf. Big Data*, 2014, pp. 301–310.
[44] P. Samczynski, K. Stasiak, D. Gromek, K. Kulpa, and J. Misiurewicz, "XY-demorad—Novel K- and mm-band radar demo kit for educational and commercial applications," in *Proc. 20th Int. Radar Symp.*, 2019, pp. 1–11.
[45] S. Trotta *et al.*, "2.3 SOLI: A tiny device for a new human machine interface," in *Proc. IEEE Int. Solid- State Circuits Conf.*, vol. 64, 2021, pp. 42–44.
[46] P. Samczyński, "Extended generalized chirp transform for signal parameter estimation in bistatic passive pulse radars," in *Proc. 14th Int. Radar Symp.*, 2013, pp. 155–160.
[47] Y. D. Zhang, X. Xiang, Y. Li, and G. Chen, "Enhanced micro-Doppler feature analysis for drone detection," in *Proc. IEEE Radar Conf.*, 2021, pp. 1–4.

Karol Abratkiewicz (Student Member, IEEE) was born in Brodnica, Poland, in 1993. He received the B.Sc. (Hons.) and M.Sc. degrees in electronics and telecommunications from the Gdańsk University of Technology (GUT), Gdańsk, Poland, in 2016 and 2017, respectively. He is currently working toward the Ph.D. degree with the Radar Research Group, Warsaw University of Technology, Warsaw, Poland.

His specialization was in microwave engineering and antenna design, whereas his M.Sc. thesis covered signal processing techniques. In 2014, he had an internship with Przemysłowy Instytut Telekomunikacji S.A., Gdańsk, Poland, and from 2015 to 2018, he was with GUT and took part in international R&D projects, providing wireless and embedded infrastructure for the Hi-Tech industry. In the summer of 2018, he had an internship with Fraunhofer FHR, Wachtberg, Germany, in the Department of Passive Radar and Antijamming Techniques. His main research interests include radar signal processing (especially time–frequency analysis), signal parameter estimation, and linear and nonlinear pulse reconstruction.

Mr. Abratkiewicz has been a Student Member of the IEEE Aerospace and Electronic Systems Society, the IEEE Signal Processing Society, and the IEEE Geoscience and Remote Sensing Society since 2018.

Jacek Gambrych (Member, IEEE) received the B.Sc. and M.Sc. degrees in electronics from the Warsaw University of Technology, Warsaw, Poland, in 2009 and 2011, respectively.

He has been with the Institute of Electronic Systems, Warsaw University of Technology, since 2012. His research interests include radar signal processing and parallel computing.