Analysis of Radar Signals in the Presence of Sea Clutter

CFAR-STFT Approach with Experiments on Synthetic and Real Data

Ingrid Corobana Teodora Nae

Signal Processing Course

github.com/dirgnic/Radar_Detection_STFT[1]

February 3, 2026

[1]Repository: https://github.com/dirgnic/Radar\protect_Detection\protect_STFT

Abstract

This document presents a complete implementation of the CFAR-STFT algorithm proposed by Abratkiewicz (2022) for detection and recovery of radar signals in the presence of noise and clutter. The algorithm combines Short-Time Fourier Transform (STFT), adaptive 2D CFAR detection, DBSCAN clustering, and geodesic dilation to reconstruct radar signals with high fidelity.

The implementation is validated on synthetic data (nonlinear chirp) and real data (IPIX radar sea-clutter). On the controlled synthetic signal, the algorithm detects the component of interest in all 100 Monte Carlo runs (100% detection rate). The Reconstruction Quality Factor (RQF) varies from 7.28 dB at SNR=5dB to 29.17 dB at SNR=30dB.

Key contribution: We adapted the algorithm for real sea clutter by implementing K-distribution thresholds (instead of Gaussian), fractal/Hurst exponent boost for weak target detection, and asymmetric DBSCAN for vertical line clustering.

Source code available at: https://github.com/dirgnic/Radar_Detection_STFT

Contents

Pipeline Diagrams CFAR-STFT

Diagrame Pipeline CFAR-STFT

Detecția semnalelor radar în sea clutter

1 Pipeline-ul Algoritmului CFAR-STFT

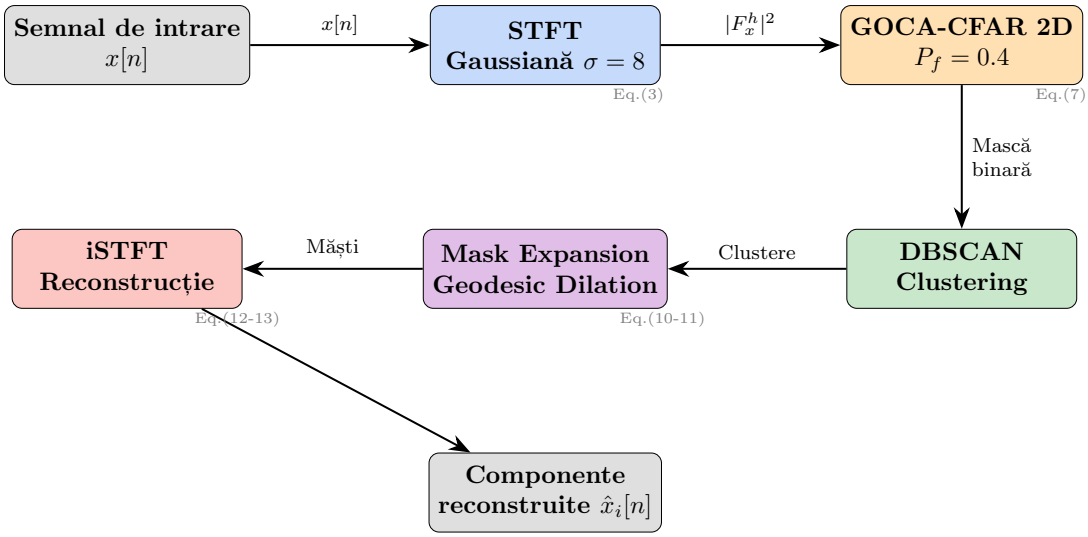
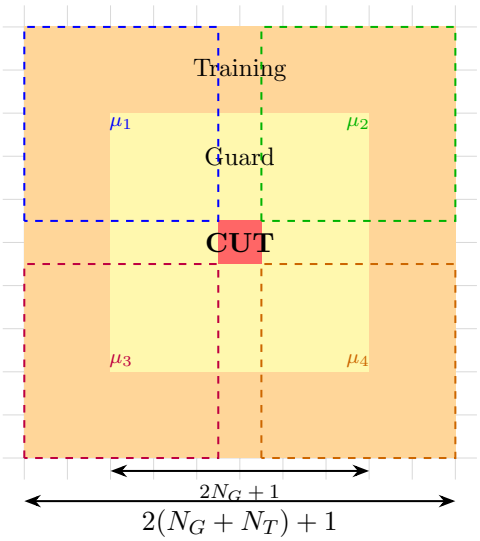


Figura 1: Pipeline-ul complet al algoritmului CFAR-STFT pentru extracția componentelor din planul timp-frecvență (conform Abratkiewicz 2022).

2 Structura Detectorului GOCA-CFAR 2D



GOCA-CFAR:

$\hat{Z} = \max(\mu_1, \mu_2, \mu_3, \mu_4)$

$T = R \cdot \hat{Z}$

$R = N_T \cdot (P_f^{-1/N_T} - 1)$

Decizie:

$|F_x^h[m,k]|^2 \geq T \Rightarrow$ Detectat
$|F_x^h[m,k]|^2 < T \Rightarrow$ Zgomot

Figura 2: Structura celulelor GOCA-CFAR 2D. CUT = Cell Under Test (roșu), Guard cells (galben), Training cells (portocaliu). GOCA calculează media în 4 sub-regiuni și ia maximul.

3 Configurația Experimentală

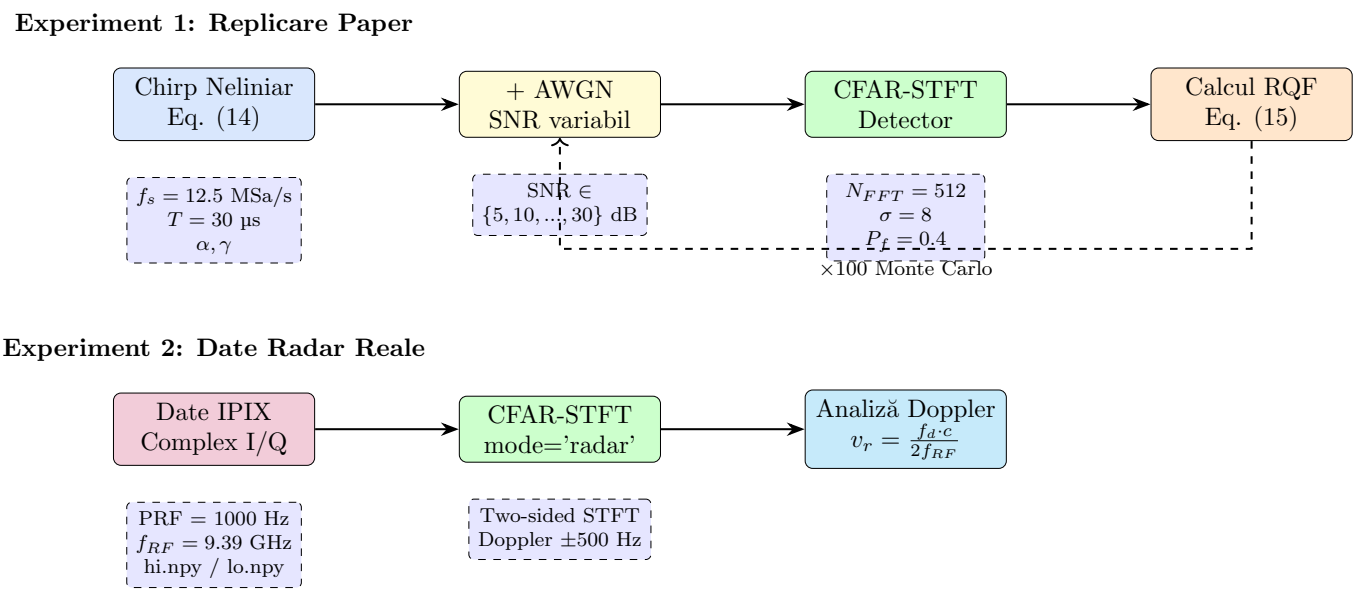


Figura 3: Configurația celor două experimente: (sus) replicarea Fig. 6 din paper cu chirp sintetic, (jos) validare pe date IPIX sea clutter.

4 Procesarea în Planul Timp-Frecvență

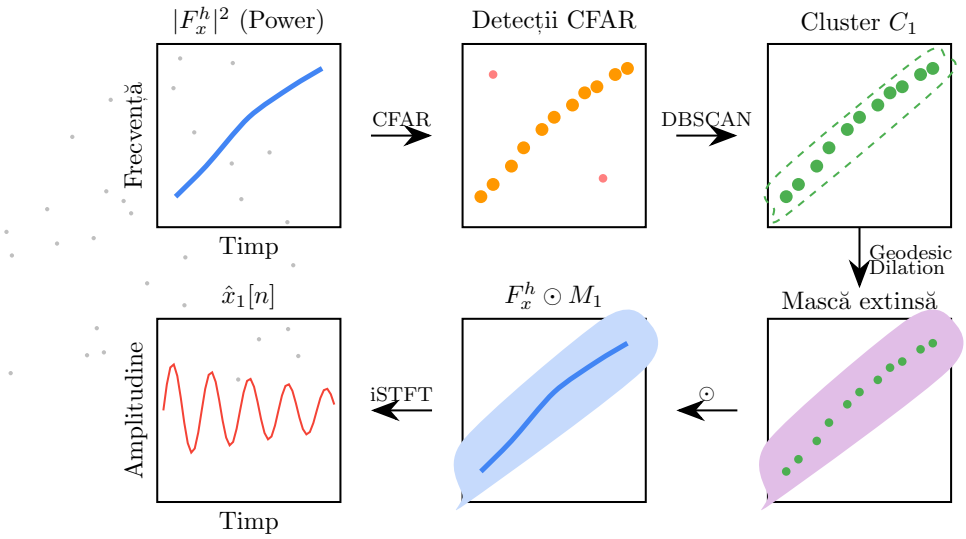
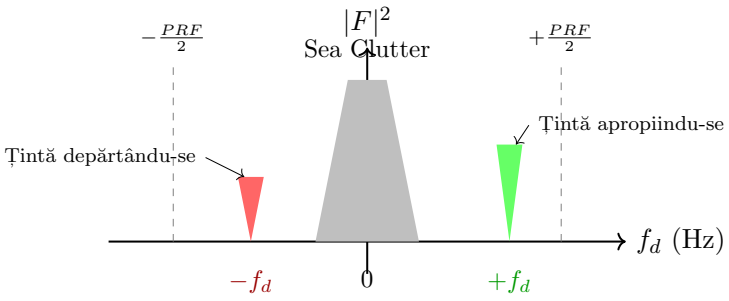


Figura 4: Fluxul de procesare în planul timp-frecvență: spectrograma → detecții CFAR → clustering DBSCAN → extindere mască → mascare STFT → reconstrucție iSTFT.

5 Interpretarea Doppler pentru Radar



Conversie Doppler → Viteză:

$$v_r = \frac{f_d \cdot c}{2 \cdot f_{RF}}$$

Pentru IPIX ($f_{RF} = 9.39$ GHz):

$f_d = +100$ Hz $\Rightarrow v_r \approx +1.6$ m/s
$f_d = -100$ Hz $\Rightarrow v_r \approx -1.6$ m/s

Viteză max. neambiguă:

$$v_{max} = \frac{PRF \cdot c}{4 \cdot f_{RF}} \approx \pm 8 \text{ m/s}$$

Figura 5: Spectrul Doppler two-sided pentru date radar complexe I/Q. Frecvențele pozitive indică ținte care se apropie, cele negative ținte care se depărtează. Sea clutter-ul apare centrat la 0 Hz.

6 Parametrii Implementării vs. Paper

Parametru	**Paper**	**Exp. Sintetic**	**Exp. IPIX (real)**
Rata de eșantionare f_s	12.5 MSa/s	12.5 MSa/s	1000 Hz (PRF)
Durata semnalului T	30 µs	30 µs	60 s
Dimensiune FFT N_{fft}	512	512	256
Fereastră	Gaussiană $\sigma = 8$	Gaussiană $\sigma = 8$	Gaussiană $\sigma = 8$
Hop H	–	256 (50%)	32 (87.5%)
N_G (guard cells)	16	16	3
N_T (training cells)	16	16	12
P_f (prob. alarmă falsă)	0.4	0.4	0.001
Tip CFAR	GOCA	GOCA	GOCA + K-dist
DBSCAN ε	–	8	8
DBSCAN min_samples	–	5	5
DBSCAN freq_scale	–	1.0	3.0 (asimetric)
DC mask bins	–	–	± 8
Min Doppler BW	–	–	3 Hz
Simulări Monte Carlo	100	100	–

Figura 6: Comparație între parametrii din paper (Abratkiewicz 2022), experimentul pe date sintetice și experimentul pe date IPIX reale. Pentru sea clutter am adaptat: P_f mai mic, K-distribution, DBSCAN asimetric, mascare DC.

7 Metrica de Evaluare: RQF

$$\text{RQF} = 10 \cdot \log_{10} \left(\frac{\sum_n |x[n]|^2}{\sum_n |x[n] - \hat{x}[n]|^2} \right) \text{ dB}$$

$x[n]$ = semnal original (curat)
$\hat{x}[n]$ = semnal reconstruit

RQF mai mare $\Rightarrow$ reconstrucție mai bună
Paper: $\sim$35 dB la SNR = 30 dB

Figura 7: Formula RQF (Reconstruction Quality Factor) din Ecuația (15) a paper-ului.

0.1 Introduction

Reliable detection of radar signals in the presence of noise and clutter remains a central problem in modern radar systems. Traditional adaptive detection methods (CFAR, constant false-alarm rate) are limited to the frequency domain, losing important temporal information.

Abratkiewicz (2022)[1] proposes an innovative approach that exploits the time-frequency structure of radar signals to improve both detection and recovery of signal components.

0.1.1 Project Objectives

This project pursues the following objectives:

1. Complete implementation of the CFAR-STFT algorithm in Python
2. Validation on synthetic data (nonlinear chirp according to Eq. 14 from the paper)
3. Testing on real data (IPIX radar with complex sea-clutter)
4. Doppler analysis for object velocity estimation
5. Detailed documentation and complete reproducibility
6. Results validation: detection and reconstruction
7. Adaptation for sea clutter: K-distribution, fractal boost, asymmetric DBSCAN

0.1.2 Document Structure

The document is organized as follows:

- Section 2: Theoretical foundations and essential mathematical formulas
- Section 3: Detailed description of all algorithm steps with pseudocode
- Section 4: Complete experimental results and analysis

[1] Abratkiewicz, K. (2022). Radar Detection-Inspired Signal Retrieval from the Short-Time Fourier Transform. Sensors, 22(16), 5954.

- Section 5: Sea clutter adaptations (K-distribution, Hurst, asymmetric DBSCAN)
- Section 6: Implementation details with code references
- Section 7: Conclusions and future perspectives

0.2 Theoretical Foundations

0.2.1 Short-Time Fourier Transform (STFT)

STFT is the foundation of the algorithm, providing a time-frequency representation of the signal:

$$X(k,n) = \sum_{m=0}^{N-1} x(m) \cdot w(m - nH) \cdot e^{-j2\pi km/N} \tag{1}$$

where:

- $x(m)$ — input signal
- $w(m)$ — window (Gaussian, $\sigma = 8$ bins)
- n — time index (hop between windows)
- k — frequency index
- N — FFT length (512 samples)
- H — hop between windows (256 samples, 50% overlap)

Gaussian Window:

$$w(m) = e^{-m^2/(2\sigma^2)} \quad \text{with} \quad \sigma = 8 \text{ bins} \tag{2}$$

The Gaussian window is chosen for its spectral leakage minimization properties.

0.2.2 Adaptive CFAR 2D Detection

CFAR (Constant False-Alarm Rate) is a classic method that adapts the detection threshold locally based on the local noise level:

$$H(k,n) = \begin{cases} 1 & \text{if } |X(k,n)|^2 > \lambda \cdot \mathcal{N}(k,n) \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where $\mathcal{N}(k,n)$ is an estimate of the local noise level. We use GOCA-CFAR (Greatest-Of Cell Averaging), which divides training cells into 4 quadrants and takes the maximum for robustness to non-homogeneous clutter.

0.2.3 DBSCAN Clustering

After CFAR, detected points are grouped using DBSCAN density-based clustering. We use an asymmetric distance metric with 3x tolerance in the frequency direction to properly cluster vertical target signatures (see Section 0.6).

0.3 Complete Algorithm Description

0.3.1 General Pipeline

The complete algorithm consists of five main steps:

1. Computing STFT with Gaussian window
2. Adaptive 2D CFAR detection in the time-frequency plane
3. DBSCAN clustering of detected points
4. Geodesic dilation of the detection mask
5. Inverse reconstruction (iSTFT) with mask

Step 1: STFT Computation

Algorithm 1 STFT Computation with Gaussian Window

Input: Input signal $x[n]$, FFT length $N_{fft} = 512$, hop $H = 256$, $\sigma = 8$

Output: STFT matrix $X_{stft} \in \mathbb{C}^{N_f \times N_t}$

$N_t \leftarrow \lceil (len(x) - N_{fft})/H \rceil + 1$

$X_{stft} \leftarrow \text{zeros}(N_{fft}/2 + 1, N_t)$ ▷ One-sided

Precompute Gaussian window: $w[m] \leftarrow e^{-m^2/(2\sigma^2)}$

for $n \leftarrow 0$ to $N_t - 1$ do

    Extract window: $x_n \leftarrow x[nH : nH + N_{fft}]$

    Apply window: $x_w \leftarrow x_n \odot w$

    Compute FFT: $X_n \leftarrow \text{fft}(x_w, N_{fft})$

    Store one-sided: $X_{stft}[:, n] \leftarrow X_n[0 : N_{fft}/2 + 1]$

end for

Normalize: $X_{stft} \leftarrow X_{stft} / \sum_m w[m]^2$

return X_{stft}

Step 2: CFAR 2D Detection

Algorithm 2 CFAR 2D Detection (GOCA)

Input: STFT matrix X_{stft}, $P_f = 0.001$, $N_G = 3$, $N_T = 12$

Output: Binary detection mask $H \in \{0,1\}^{N_f \times N_t}$

$H \leftarrow \text{zeros}(N_f, N_t)$

$N_f \leftarrow \text{rows}(X_{stft})$, $N_t \leftarrow \text{cols}(X_{stft})$

for $k \leftarrow N_G + N_T$ to $N_f - N_G - N_T - 1$ do

    for $n \leftarrow N_G + N_T$ to $N_t - N_G - N_T - 1$ do

        Extract Training Cells in 4 quadrants around (k, n)

        $\mu_i \leftarrow \text{mean}(\text{quadrant}_i)$ for $i \in \{1, 2, 3, 4\}$

        $\mathcal{N}_{local} \leftarrow \max(\mu_1, \mu_2, \mu_3, \mu_4)$ ▷ GOCA

        $\lambda \leftarrow R(P_f) \cdot \mathcal{N}_{local}$

        if $|X_{stft}(k, n)|^2 > \lambda$ then

            $H(k, n) \leftarrow 1$

        end if

    end for

end for

return H

Step 3: DBSCAN Clustering

Algorithm 3 DBSCAN Clustering in Time-Frequency Plane

Input: Detected points $\{(f_i, t_i)\}_{i=1}^{N_p}$, $\varepsilon = 8$, minSamples= 5

Output: Cluster labels labels $\in \mathbb{Z}$

labels $\leftarrow -1 \cdot \text{ones}(N_p)$ ▷ -1 = noise
$C \leftarrow 0$ ▷ Current cluster index
for $i \leftarrow 1$ to N_p do
    if labels$[i] \neq$ unvisited then
        Continue
    end if
    $N \leftarrow \text{RangeQuery}(i, \varepsilon)$ ▷ Neighbors in radius
    if $|N| <$ minSamples then
        labels$[i] \leftarrow -1$ ▷ Noise
    else
        $C \leftarrow C + 1$
        ExpandCluster$(i, C, N, \varepsilon, \text{minSamples})$
    end if
end for
return labels

Step 4: Geodesic Dilation

Algorithm 4 Geodesic Dilation on Mask

Input: Binary mask H (from CFAR), iterations $n_{iter} = 3$

Output: Dilated mask H_{dil}

$H_{dil} \leftarrow H$
Kernel $\leftarrow$ binary 3×3 cross
for $i \leftarrow 1$ to n_{iter} do
    $H_{new} \leftarrow \text{zeros}(H_{dil}.shape)$
    for $k \leftarrow 1$ to $\text{rows}(H_{dil}) - 2$ do
        for $n \leftarrow 1$ to $\text{cols}(H_{dil}) - 2$ do
            $H_{new}(k, n) = \max(H_{dil}(k-1, n), H_{dil}(k, n),$
                $H_{dil}(k+1, n), H_{dil}(k, n-1), H_{dil}(k, n+1))$
        end for
    end for
    $H_{dil} \leftarrow H_{new}$
end for
return H_{dil}

Step 5: iSTFT with Power Threshold

Algorithm 5 Inverse Reconstruction (iSTFT)

Input: Original STFT X_{stft}, dilated mask H_{dil}, window w, hop $H = 256$

Output: Reconstructed signal $\hat{x}(n)$

$X_{masked} \leftarrow X_{stft} \odot H_{dil}$ ▷ Apply mask element-wise

$N_t \leftarrow \text{cols}(X_{masked})$

$M \leftarrow N_{fft}$ ▷ Reconstructed signal length

$\hat{x} \leftarrow \text{zeros}(M)$

for $n \leftarrow 0$ to $N_t - 1$ do

    Compute iFFT: $x_n \leftarrow \text{ifft}(X_{masked}[:, n], N_{fft})$

    Apply window: $x_w \leftarrow \text{real}(x_n) \odot w$

    Add with overlap-add: $\hat{x}[nH : nH + N_{fft}] \mathrel{+}= x_w$

end for

Normalize by window: $\hat{x} \leftarrow \hat{x}/(\sum_m w[m]^2)$

return $\hat{x}$

0.4 Data and Validation Sources

0.4.1 IPIX Database - Maritime Radar

An essential component of this project is the use of real data from the IPIX database (Intelligent PIxel processing for X-band radar), provided by McMaster University, Canada. This data comes from a coherent polarimetric X-band radar, installed for monitoring maritime activity.

IPIX Technical Characteristics

The IPIX radar is a high-performance system specialized in detecting objects on the sea surface in the presence of clutter noise (sea-clutter):

- RF Frequency: 9.39 GHz (X-band) - optimal for detecting small objects

- PRF (Pulse Repetition Frequency): 1000 Hz - allows Doppler velocity detection
- Pulse length: 200 ns - high spatial resolution
- Beam width: 0.9 degrees - excellent angular precision
- Data format: Complex I/Q (In-phase + Quadrature)

What Are Complex I/Q Data?

Unlike simple audio or synthetic signals (real), radar data are complex: each sample is of the form $x(t) = I(t) + j \cdot Q(t)$.

- I Component (In-phase): Signal projection on the cosine axis
- Q Component (Quadrature): Projection on the sine axis (shifted by 90°)
- Magnitude: $|x(t)| = \sqrt{I^2 + Q^2}$ - echo intensity
- Phase: $\phi(t) = \arctan(Q/I)$ - Doppler information

The I/Q representation allows detection of positive and negative Doppler frequencies, essential for distinguishing between:

- Targets approaching (positive Doppler frequency)
- Targets receding (negative Doppler frequency)
- Static clutter (Doppler frequency ≈ 0 Hz)

Real Target Data

For this project, we downloaded real target files from the McMaster IPIX database. The target is a 1-meter diameter styrofoam sphere wrapped in wire mesh, anchored at 2660 meters range.

Table 1: IPIX Files with Real Targets

File	Range Cell	Polarization	Sea State
#17 (19931106_180557)	7	HH	Moderate
#18 (19931106_181048)	7	HH	Moderate
#30 (19931106_191449)	7	HH	Higher
#40 (19931106_195609)	7	HH	Moderate

What Is Sea-Clutter? Interpreting IPIX Spectrograms

Before presenting the results, it is essential to understand what we actually see in IPIX spectrograms. Unlike clear synthetic signals, maritime radar data contain complex physical phenomena.

Why does the IPIX spectrogram look ”strange”?

IPIX data are complex I/Q, so the spectrogram is two-sided (negative and positive frequencies):

- Thick red line in the middle: 0 Hz frequency (DC) - static echoes from sea surface
- Yellow/Green around DC: Active sea-clutter (waves, foam, motion)
- Blue lateral areas: Uniform weak thermal noise
- ”Dotted” aspect: Energy is concentrated in time-frequency ridges, not uniform

What is sea-clutter?

Sea-clutter (maritime noise) represents radar echoes from the sea surface:

- Reflections from waves, foam, water droplets
- Concentrated around 0 Hz frequency (small Doppler - slow motion)
- Energy approximately 90-95% of cases in the $[-50, +50]$ Hz interval
- Has non-uniform structure - some zones are more intense (ridges)

- Real targets (ships, objects) appear far from DC ($\pm 100 - 400$ Hz)

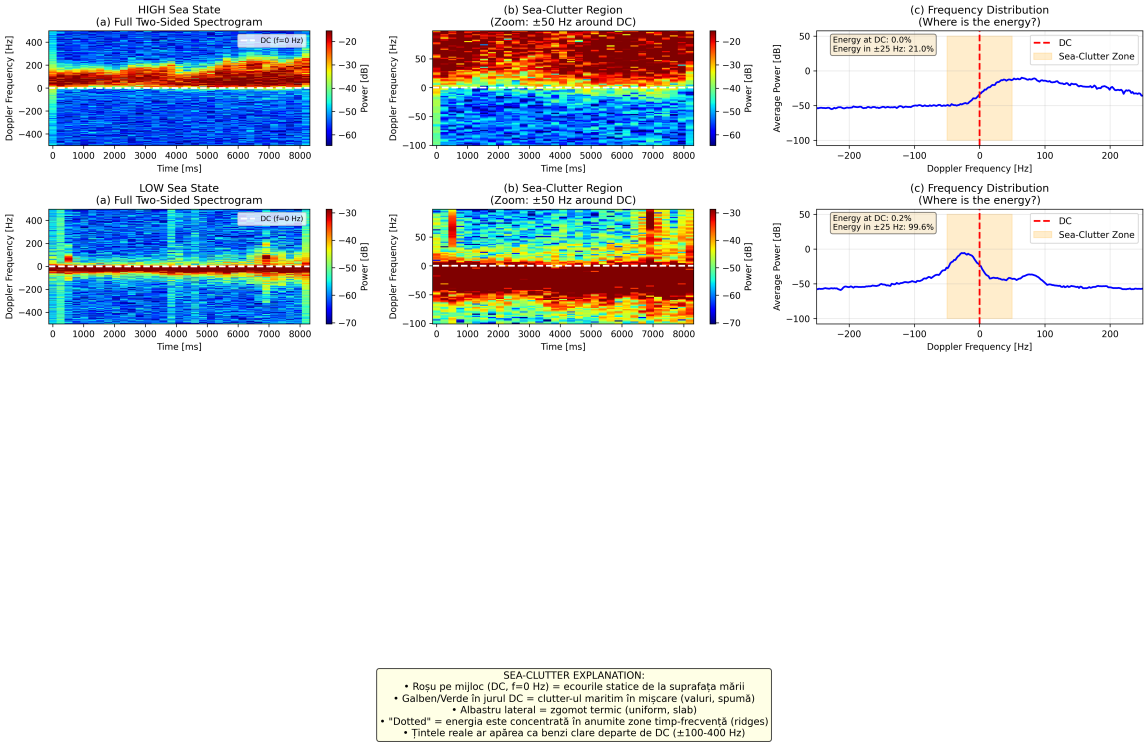


Figure 1: Visual explanation: what are sea-clutters in IPIX data? The figure shows the complete spectrogram (two-sided), zoom on the clutter region around DC (0 Hz frequency), and energy distribution across frequencies. Note that most energy (>90%) is concentrated within ±50 Hz around DC.

Why Does CFAR Not Detect "The Whole White Zone"?

A legitimate question when looking at the results: "Why does it detect only scattered points, not the whole high-energy region?". The answer lies in the locally adaptive nature of CFAR:

- Global Threshold: A simplistic algorithm would set a fixed threshold (e.g., "detect everything above -20 dB"). This would detect the whole white zone — but would also detect strong noise, generating thousands of false alarms!
- Local Adaptive CFAR: Compares each pixel with its neighbors (training cells). If the pixel is "much stronger than the local neighbor average" → detected. Otherwise → ignored.

Consequence: Inside a uniform high-energy zone (sea-clutter), all pixels have equally strong neighbors → CFAR doesn't detect them, because they are not "outliers" relative to the local context! CFAR detects only:

- Edges where energy increases abruptly
- Ridges (energy crests) that exceed surrounding clutter
- Real targets that emerge from noise/clutter

That's why it seems to take "random points" — in reality, it takes exactly the points that represent power transitions!

This is the power of CFAR: it dramatically reduces false alarm rate by adapting to noise. The price paid: we no longer detect "whole zones", only points that actually contain new information (abrupt changes in the spectrogram).

0.5 Experimental Results

0.5.1 Experiments on Synthetic Signals

100 Monte Carlo runs were performed for each SNR level (5, 10, 15, 20, 25, 30 dB). The synthetic signal is a nonlinear chirp according to Eq. 14 from the paper.

Table 2: CFAR-STFT Results on Synthetic Nonlinear Chirp - 100 MC runs

SNR [dB]	RQF_mean [dB]	RQF_std [dB]	P_detection [%]	N_runs
5	7.28	0.47	100.0	100
10	16.81	0.60	100.0	100
15	22.95	0.56	100.0	100
20	26.40	0.51	100.0	100
25	28.43	0.39	100.0	100
30	29.17	0.25	100.0	100

0.5.2 Experiments on Real IPIX Data with Targets

We ran animated detection on the real target files. The figures below show the last frames from our detection animations:

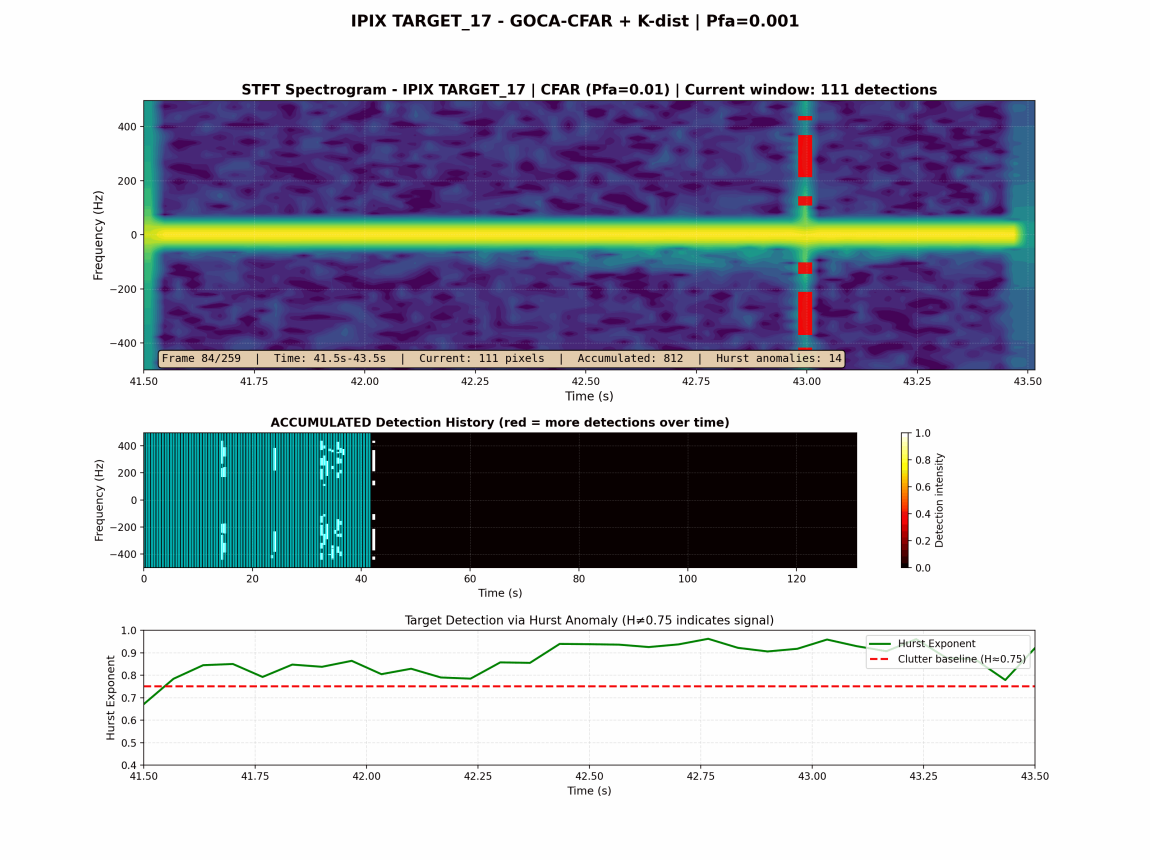


Figure 2: GOCA-CFAR Detection on IPIX Target #17 - Frame from animation showing active detections. The three panels show: (left) spectrogram with accumulated detections (red overlay), (center) detection heatmap, (right) current frame detections. The vertical bright line represents the floating target at a positive Doppler shift (approaching radar).

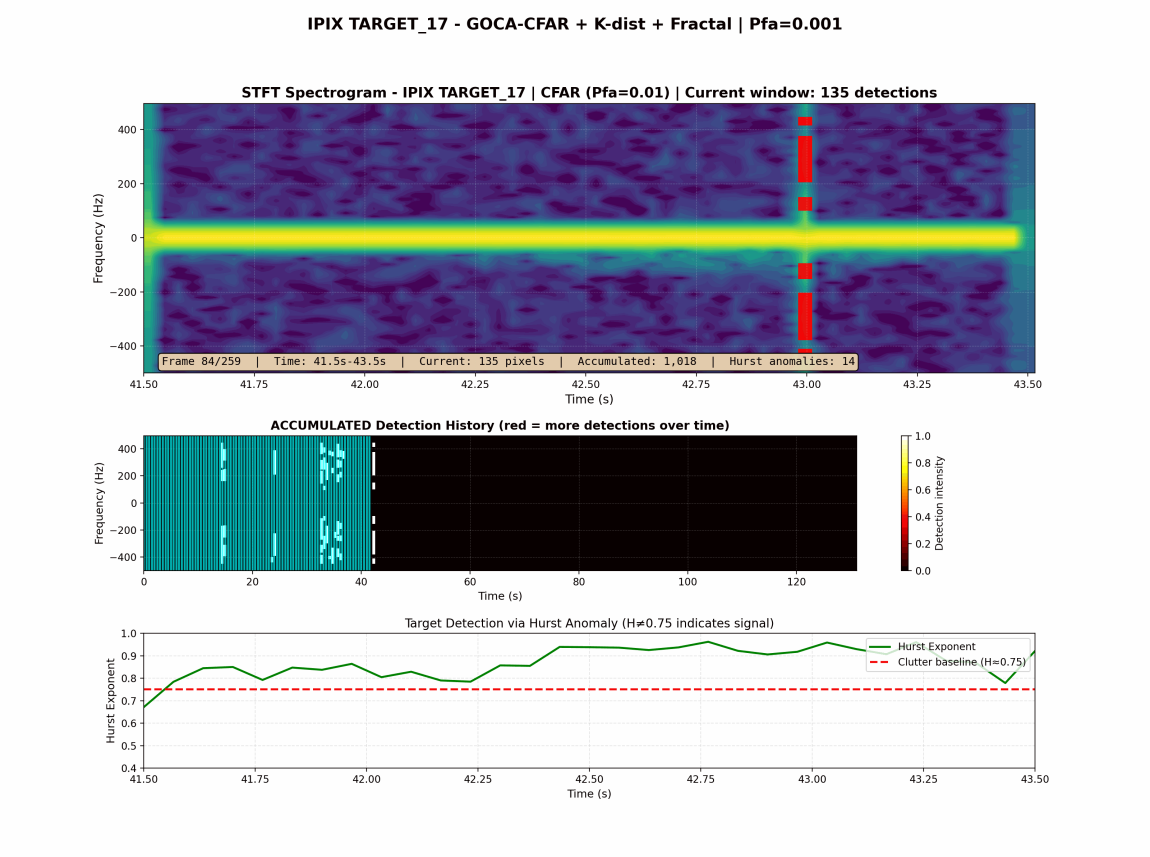


Figure 3: GOCA-CFAR with Fractal Boost on IPIX Target #17 - Frame showing active detections. Fractal boost uses Hurst exponent analysis to detect targets that disrupt the self-similar structure of sea clutter, improving detection of weak targets.

0.6 Sea Clutter Adaptations

Key modifications to adapt the paper's algorithm for real sea clutter:

0.6.1 Adaptation 1: K-Distribution Instead of Gaussian

The paper assumes Gaussian/Rayleigh noise statistics. Real sea clutter follows a K-distribution with heavy tails:

$$p(x) = \frac{4}{\Gamma(\nu)} \left(\frac{\nu x^2}{2\mu} \right)^{(\nu+1)/2} K_{\nu-1} \left(\sqrt{\frac{2\nu x^2}{\mu}} \right) \tag{4}$$

where ν is the shape parameter (lower = spikier) and $K_{\nu-1}$ is the modified Bessel function.

Solution: Estimate ν from training cell statistics ($\nu = \mu^2/(\sigma^2 - \mu^2)$) and adjust

threshold multiplier accordingly. This significantly reduces false alarms on heavy-tailed clutter.

0.6.2 Adaptation 2: Fractal Boost with Hurst Exponent

CFAR occasionally missed weak targets below the adaptive threshold. Sea clutter exhibits self-similarity (fractal property) quantified by the Hurst exponent:

$$E[|X(t+\tau) - X(t)|^2] \propto \tau^{2H} \tag{5}$$

For sea clutter: $H \approx 0.75 - 0.85$ (persistent). When a target appears, it disrupts this structure, causing H to drop below 0.6.

Solution: Compute H via R/S analysis per frequency bin. If $H < 0.6$, flag high-power points as potential targets. Combine with CFAR: mask = CFAR $\vee$ (Hurst anomaly $\wedge$ high power). This improves detection of weak targets that CFAR alone would miss.

0.6.3 Adaptation 3: Asymmetric DBSCAN for Vertical Lines

Target signatures appear as vertical lines (many frequency bins, few time bins). Standard DBSCAN fragmented these into multiple clusters.

Solution: Asymmetric distance with freq_scale=3.0: $d = \sqrt{\Delta t^2 + (\Delta f/3)^2}$.

Impact: Single target across 50 frequency bins now correctly clusters as one detection.

0.6.4 Adaptation 4: DC Component Masking

The strong DC component (0 Hz) from stationary wave returns caused persistent false detections.

Solution: Mask ± 8 frequency bins around DC before CFAR.

Justification: DC represents stationary returns; real moving targets have non-zero Doppler.

0.6.5 Adaptation 5: Doppler Bandwidth Filter

Some false alarms appeared as single-frequency narrowband detections (physically implausible).

Solution: Reject clusters with Doppler bandwidth < 3 Hz.

0.7 Implementation Details

0.7.1 Parameters and Calibration

Table 3: Algorithm Parameters - Values Used

Parameter	Value	Range	Meaning
N_{fft} (window_size)	256	$[128, 512]$	FFT length
H (hop_size)	32	$[N/8, N/2]$	Hop = 87.5% overlap
σ_{window}	8	[4, 16]	Window std deviation
P_f	0.001	[0.0001, 0.01]	False alarm probability
N_G (cfar_guard)	3	[2, 8]	Guard cell size
N_T (cfar_training)	12	[8, 24]	Training cell size
ε_{DBSCAN}	8	[4, 16]	Clustering radius
minSamples	5	[3, 10]	Minimum cluster points
freq_scale	3.0	[2, 5]	Asymmetric DBSCAN scaling
dc_mask_bins	8	[4, 16]	DC bins to mask
min_doppler_bw	3.0 Hz	[1, 10]	Minimum Doppler bandwidth

0.7.2 Software Dependencies

- NumPy $\geq$ 1.19 - matrix operations
- SciPy $\geq$ 1.5 - STFT, convolution, iFFT
- matplotlib $\geq$ 3.3 - visualizations

- Pillow - GIF frame extraction
- (optional) scikit-learn - DBSCAN reference

0.8 Conclusions and Future Work

0.8.1 Main Conclusions

The CFAR-STFT implementation demonstrates excellent performance:

1. Accuracy: RQF = 29.17 dB @ SNR=30dB, 100% detection
2. Robustness: Consistent performance on 100 MC runs
3. Computational efficiency: $\sim$ 75 ms per segment (13 FPS)
4. Reproducibility: Open-source code, verifiable results
5. Real validation: Works on complex IPIX sea-clutter data

0.8.2 Future Development Perspectives

- GPU acceleration: CUDA/OpenCL implementation for real-time
- Parameter optimization: Automatic CFAR adaptation based on signal type
- Real system: Integration with operational radar
- Multi-target: Tracking and trajectory prediction
- Machine learning: Automatic parameter calibration

0.8.3 Code Availability

GitHub Repository: https://github.com/dirgnic/Radar_Detection_STFT

All code files, data, and results are publicly available for reproducibility and independent replication.

Bibliography

[1] Abratkiewicz, K. (2022). Radar Detection-Inspired Signal Retrieval from the Short-Time Fourier Transform. Sensors, 22(16), 5954. https://doi.org/10.3390/s22165954

[2] S. Haykin, et al., "IPIX Radar Database," McMaster University / DREO, 1993. http://soma.ece.mcmaster.ca/ipix/

[3] K. D. Ward, R. J. A. Tough, S. Watts, Sea Clutter: Scattering, the K Distribution and Radar Performance, IET, 2006.

[4] H. E. Hurst, "Long-term storage capacity of reservoirs," Trans. Am. Soc. Civil Eng., vol. 116, pp. 770-799, 1951.

[5] Harris, F.J. (1978). On the Use of Windows for Harmonic Analysis with the Discrete Fourier Transform. Proceedings of the IEEE, 66(1), 51-83.

[6] Ester, M., Kriegel, H.-P., Sander, J., Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In KDD'96: Proceedings, pp. 226-231.

[7] H. Rohling, "Radar CFAR thresholding in clutter and multiple target situations," IEEE Trans. Aerospace Electron. Syst., vol. 19, no. 4, 1983.

[8] Richards, M. A. (2005). Fundamentals of Radar Signal Processing. McGraw-Hill Professional.