UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

Specializarea: Informatică

Proiect la Procesarea Semnalelor

ANALIZA SEMNALELOR RADAR ÎN PREZENȚA SEA CLUTTER

Abordare bazată pe CFAR-STFT și experimente pe date sintetice și reale

Studenți
Ingrid Corobana
Teodora Nae

Coordonator științific
Prof. Dr. Cristian Rusu

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

București, 2026

Rezumat

Acest document prezintă o implementare completă a algoritmului CFAR–STFT, propus de Abratkiewicz (2022) [1], pentru detecția și reconstrucția semnalelor radar în prezența zgomotului și a sea clutter-ului (zgomot de fond produs de suprafața mării). Pornind de la o reprezentare timp–frecvență (STFT), aplicăm un prag adaptiv local (CFAR 2D) pentru a obține o matrice de detecție rară, pe care o stabilizăm prin grupare (DBSCAN) și conectare morfologică (dilatare geodezică), astfel încât reconstrucția să rețină componenta de interes.

Implementarea este validată pe date sintetice (chirp neliniar) și pe date reale (IPIX radar, "sea clutter" (date reale)) [2]. Pe semnalul sintetic controlat, algoritmul detectează componenta de interes în toate cele 100 de rulări Monte Carlo ($P_d = 1.00$). RQF (Reconstruction Quality Factor) crește de la 7.28 dB (SNR = 5 dB) la 29.17 dB (SNR = 30 dB).

Contribuția principală: adaptăm algoritmul pe date reale folosind distribuția K (în loc de Gaussian) [3], o îmbunătățire bazată pe proprietăți fractale (exponentul Hurst) pentru target-uri slabe [4] și DBSCAN cu metrică anizotropică pentru clustering de semnături cvasi-verticale. De asemenea, realizăm o analiză comparativă cu algoritmi alternativi: CA-CFAR + HDBSCAN și separare geometrică prin triangulare Delaunay.

Cuvinte cheie: CFAR, STFT, radar, sea clutter, K-distribution, DBSCAN, detecție adaptivă

Abstract

This document presents a complete implementation of the CFAR-STFT algorithm, proposed by Abratkiewicz (2022) [1], for detection and reconstruction of radar signals in the presence of noise and sea clutter (sea-surface backscatter). Starting from a time–frequency (STFT) representation, we apply a local adaptive threshold (2D CFAR) to obtain a sparse mask, which is then stabilized via grouping (DBSCAN) and morphological linking (geodesic dilation) so that the reconstruction preserves the component of interest.

The implementation is validated on synthetic data (nonlinear chirp) and real data (IPIX radar, "sea clutter" real measurements) [2]. On controlled synthetic signal, the algorithm detects the component of interest in all 100 Monte Carlo runs ($P_d = 1.00$). RQF (Reconstruction Quality Factor) increases from 7.28 dB (SNR = 5 dB) to 29.17 dB (SNR = 30 dB).

Key contribution: we adapt the algorithm to real sea clutter using a K-distribution (instead of Gaussian) [3], a fractal-based enhancement (Hurst exponent) for weak targets [4], and an anisotropic DBSCAN metric for clustering quasi-vertical signatures. We also include a comparative analysis against alternative pipelines: CA-CFAR + HDBSCAN and Delaunay-triangulation-based separation.

Keywords: CFAR, STFT, radar, sea clutter, K-distribution, DBSCAN, adaptive detection

Cuprins

Capitolul 1

Introducere

Problema principală abordată este detecția obiectelor mici în date radar maritime, într-un mediu complex. Mediul acvatic are următoarele caracteristici particulare: (1) statisticile nu sunt bine modelate de o distribuție Gaussiană (valorile extreme sunt frecvente), (2) există corelație temporală (valuri cu tipare structurate), (3) efectele Doppler extind spectrul (valuri în mișcare), (4) apar vârfuri de energie (spikes) vizibile când valurile sunt mari.

Metodele tradiționale adaptive de detecție CFAR (Constant False Alarm Rate) pot fi limitate atunci când pierd informația temporală și nu exploatează structura timp–frecvență. Abratkiewicz (2022)[1] propune o abordare care folosește explicit structura timp–frecvență pentru a îmbunătăți atât detecția, cât și recuperarea/reconstrucția componentelor semnalului.

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

Capitolul 2

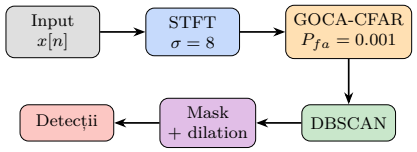
Descrierea completă a algoritmului

2.1 Fundamente teoretice și STFT

Componentă	Model matematic	Rol în algoritm
STFT (Short-Time Fourier Transform)	$X(k,n) = \sum_{m=0}^{N-1} x(m)$ $w(m-nH)\, e^{-j2\pi km/N}$ Param.: $N = 256$, $H = 32$, fereastră Gauss, $\sigma = 8$	Transformă semnalul în plan timp–frecvență; spectrograma $\lvert X(k,n)\rvert^2$ este input-ul pentru CFAR și clustering.
CFAR 2D (GOCA-CFAR)	$H(k,n) = 1\{\lvert X(k,n)\rvert^2 > T(k,n)\}$ $T(k,n) = \alpha\, \hat{Z}$ $\hat{Z} = \max(\mu_1, \mu_2, \mu_3, \mu_4)$ unde α depinde de P_{fa} și N_T	Detectează candidații de target pe baza unui prag adaptiv local, folosind 4 cadrane de training (colțuri); controlează rata de alarmă falsă.
DBSCAN – clustering pe densitate	$d = \sqrt{(s_t\, \Delta t)^2 + (\Delta f / s_f)^2}$ Anizotropic ⇒ scalare diferită pe axe; aici $s_f = 3$, $s_t = 1.5$ Notă: $\Delta f, \Delta t$ sunt măsurate în unități normalizate la rezoluția STFT (bin-uri)	Grupează punctele detectate (k,n) în clustere coerente (target-uri radar) păstrând structura aproape verticală.

Tabela 2.1: Fundamente teoretice: STFT, detecția CFAR 2D și clusteringul DBSCAN în cadrul algoritmului CFAR-STFT.

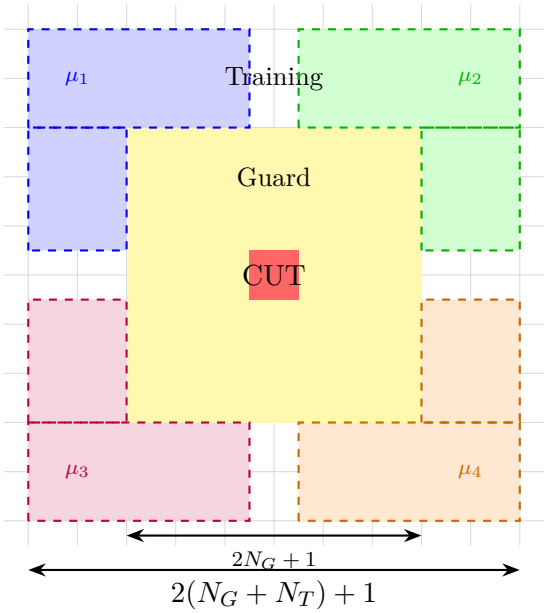
2.2 Pipeline general (5 pași)



Algoritmul complet constă din cinci etape:
(1) STFT: fereastră Gaussiană; $N_{fft} = 256$, $H = 32$, $\sigma = 8$.
(2) GOCA-CFAR 2D:
prag $T = \alpha \cdot \max(\mu_1, \ldots, \mu_4)$;
decizie $\lvert X(k,n)\rvert^2 > T$.
(3) DBSCAN:
distanță anizotropică în unități de bin:
$d = \sqrt{(s_t \Delta t)^2 + (\Delta f / s_f)^2}$; tipic $\varepsilon = 8$, minSamples=5.
(4) Dilatare geodezică:
dilatare geodezică iterativă cu conectivitate 4, limitată de o mască de energie joasă (până la convergență / max. iterații).
(5) Extragere:
$X_{\text{masked}} = X_{\text{STFT}} \odot H_{\text{dil}}$.

Figura 2.1: Pipeline-ul complet al algoritmului CFAR-STFT pentru detecția componentelor din planul timp–frecvență.

2.2.1 Pasul 2: Structura CFAR 2D



GOCA-CFAR:

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

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

Rayleigh (CA, aprox.): $\alpha = N_T(P_{fa}^{-1/N_T} - 1)$

K-dist.: $\alpha = q_K(P_{fa}; a_K, b_K)/b_K$

Decizie:

$P_{\text{CUT}} \geq T \Rightarrow$ Det

$P_{\text{CUT}} < T \Rightarrow$ Zgomot

unde $P_{\text{CUT}} = |X(k,n)|^2$ (power STFT)

Figura 2.2: Structura celulelor GOCA-CFAR 2D: CUT (roșu), Guard (galben), Training (colorat pe 4 cadrane). Mediile $\mu_1, \ldots, \mu_4$ se calculează doar pe celulele de training (excluzând zona de guard și CUT), iar GOCA folosește maximul pentru adaptare locală.

Notă: pentru a evita contaminarea estimării de fond de către „ridge”-uri/sidelobe-uri ale unui target puternic (energie care se scurge pe axele timp/frecvență sau range/Doppler), se exclude o „cruce” de celule pe axele verticală/orizontală dintre cadrane (buffer alb); în implementarea de față aceasta este realizată implicit prin definirea cadranelor în colțuri, separate de zona de guard.

Capitolul 3

Date și surse de validare

3.1 Baza de date IPIX

IPIX (McMaster University) [2]: radar X-band (f_{RF} = 9.39 GHz, PRF=1000 Hz), date complexe I/Q. Target-uri reale (sferă 1m la 2660m): #17, #18, #30, #40.

3.2 Experimente pe semnale sintetice

S-au rulat 100 de simulări Monte Carlo pentru fiecare nivel SNR, folosind un chirp neliniar, cu rată de detecție 100%.

Metrica RQF:

$$\mathrm{RQF} = 10 \log_{10} \left(\frac{\sum_n |x[n]|^2}{\sum_n |x[n] - \hat{x}[n]|^2} \right) \ [\mathrm{dB}] \tag{3.1}$$

Tabela 3.1: Rezultate CFAR-STFT (100 MC)

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

3.3 Analiză comparativă: Algoritmi CFAR și Clustering

3.3.1 Principii fundamentale CFAR

Deși arhitecturile specifice variază, toți algoritmii CFAR (Constant False Alarm Rate) urmează principii comune: procesare prin fereastră glisantă (CUT + guard cells + training cells, iar guard este exclus din estimare pentru a evita „scurgerea" energiei unui target), prag adaptiv $T = \alpha \cdot Z$ (cu α dependent de P_{fa} și N_T) și decizie de detecție dacă $P_{\mathrm{CUT}} \geq T$.

3.3.2 Algoritmi CFAR

CA-CFAR: folosește media aritmetică, robust în zgomot (aproape) omogen. OS-CFAR: folosește o percentilă (în locul mediei), robust la interferențe/outlieri. SOCA-CFAR: alege minimul dintre mediile subregiunilor ferestrei; util la marginile de clutter.

3.3.3 CFAR: Formule și pseudocod

Metodă	Model matematic	Pseudocod simplificat
CA-CFAR	$Z = \frac{1}{N_T} \sum_{i=1}^{N_T} x_i$ $T = \alpha \cdot Z$	1. Extrage celulele de antrenament S_{train} (fără zona de guard) 2. $Z = \text{mean}(S_{train})$ 3. IF $CUT \geq T$ THEN detecție
OS-CFAR	$Z = x_{(k)}$, $T = \alpha \cdot Z$	1. Sortează $S_{train} \rightarrow S_{sorted}$ 2. $Z = S_{sorted}[k]$ (valoarea de rang k) 3. IF $CUT \geq T$ THEN detecție
SOCA-CFAR	$Z = \min(\mu_1, \ldots, \mu_4)$ $T = \alpha \cdot Z$	1. Împarte fereastra în 4 subregiuni 2. Calculează media μ_i pentru fiecare subregiune 3. $Z = \min(\mu_1, \ldots, \mu_4)$ 4. IF $CUT \geq T$ THEN detecție

Tabela 3.2: Formule și pseudocod pentru metodele CFAR

3.3.4 Metode de clustering

Observații: Agglomerative nu elimină zgomotul și poate grupa detecții false CFAR; HDBSCAN folosește Mutual Reachability Distance (MRD) pentru a penaliza zonele cu densitate mică, reducând fenomenul de chaining.

Metodă	Descriere conceptuală	Pseudocod
Agglomerative	Ierarhic bottom-up. Fuzionează succesiv perechile cele mai apropiate.	1. Normalizează punctele (f, t) 2. Fiecare punct = cluster individual 3. WHILE nr_clustere $> K$: Găsește clusterele cu d_{min} Le unește sub aceeași etichetă 4. Reindexează etichetele (0, 1, 2...)
HDBSCAN	Bazat pe densitate ierarhică. Folosește core distance și MST pentru stabilitate.	1. Normalizează punctele (f, t) 2. Calculează core distance (c) pentru fiecare punct 3. Determină MRD: $d_{mrd}(u, v) = \max(c(u), c(v), d(u, v))$ 4. Construiește MST folosind MRD 5. Extrage componente conexe cu prag ε 6. Elimină clusterele sub $min\_samples \rightarrow$ zgomot (-1)
DBSCAN	Bazat pe densitate	Grupează puncte cu distanță $\leq \varepsilon$. (Coordonate normalizate la rezoluția STFT: t/dt, f/df.) Anizotropic: $d = \sqrt{(s_t \Delta t)^2 + (\Delta f / s_f)^2}, \quad s_f = 3,\ s_t = 1.5$

Tabela 3.3: Metode de clustering utilizate

3.3.5 Separare prin triangulare Delaunay

Alternativă geometrică la CFAR + clustering:

Pas	Idee	Pseudocod simplificat
Detectare Vârfuri	Maxime locale în spectrograma STFT peste un prag percentilă ($S = \|X\|^2$)	1. STFT $\rightarrow S = \|X\|^2$ 2. Threshold T = magnitudinea percentilei din S 3. Maxime locale cu $S > T$ 4. Salvează coordonate vârfuri
Triangulare Delaunay	Triangulare pe vârfuri cu criteriul cercului circumscris (Delaunay)	1. T_1: triunghi auxiliar (acoperă punctele) 2. Pentru fiecare p: inserează; reface triangularea local 3. La final: șterge triunghiurile cu vârfuri din T_1
Gruparea triunghiurilor	Triunghiuri vecine conectate după energie medie, dacă au muchie comună	1. Construiește lista de adiacență 2. Leagă triunghiuri cu energie similară ($\Delta < \varepsilon$) 3. Componente conexe (DFS/BFS) 4. Pentru fiecare componentă: centroid + energie

Tabela 3.4: Sinteză a pașilor de separare prin triangulare Delaunay și pseudocod asociat

3.3.6 Experimente pe semnale sintetice controlate

Metodă 1: CA-CFAR + HDBSCAN: STFT Hamming, CA-CFAR, HDBSCAN clustering, reconstrucție componentă, evaluare RQF.

Metodă 2: Triangulare Delaunay (separare geometrică): Se calculează STFT cu fereastră Gaussiană și se formează spectrograma $S = |X|^2$ în plan timp–frecvență. Apoi, în locul unei marginire CFAR, se extrag maxime locale peste un prag percentilă, obținând un set rar de vârfuri candidate. Pe coordonatele acestor vârfuri se construiește triangularea Delaunay, iar triunghiurile rezultate sunt grupate în componente conexe (pe baza adiacenței) și filtrate după energie medie similară. În final, Doppler-ul este estimat din componenta cu energia maximă (de ex. prin centroid/medie pe axa frecvență, rezultând f_D).

Rezultate sintetice: CA-CFAR+HDBSCAN: 100% detecție, RQF stabil SNR 5-30 dB. Triangulare: bună SNR mediu/ridicat, variabilitate mai mare RQF.

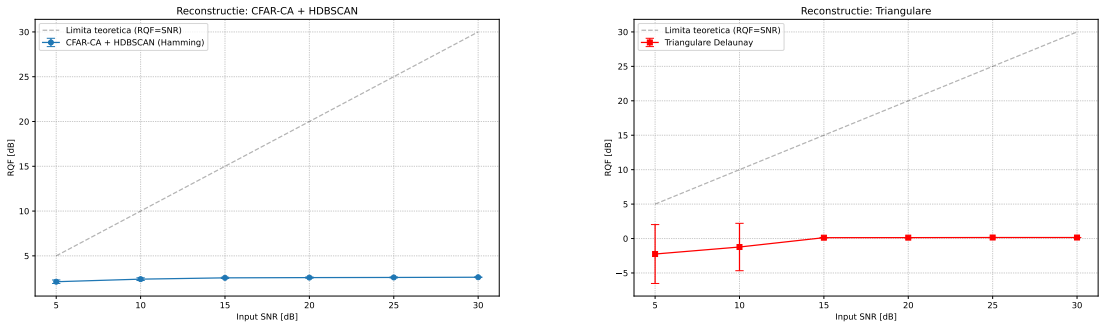


Figura 3.1: Extensie pe semnalul sintetic din lucrarea de referință: RQF vs. SNR pentru (stânga) CA-CFAR + HDBSCAN (fereastră Hamming) și (dreapta) separare geometrică prin triangulare Delaunay.

3.4 Experimente pe IPIX cu target-uri reale

S-au efectuat experimente pe date reale din baza IPIX. Figurile de mai jos arată cadre reprezentative din secvențele de detecție:

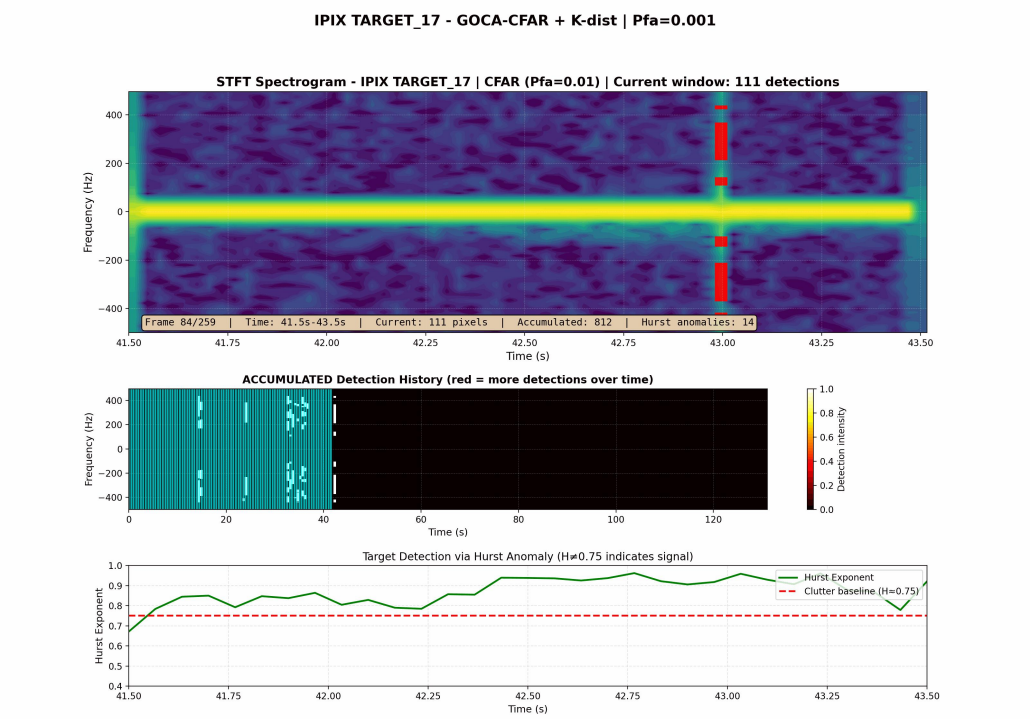


Figura 3.2: GOCA-CFAR pe IPIX Target #17: spectrograma (stânga), heatmap detecții (centru), detecții cadru (dreapta). Linie verticală = target la Doppler pozitiv.

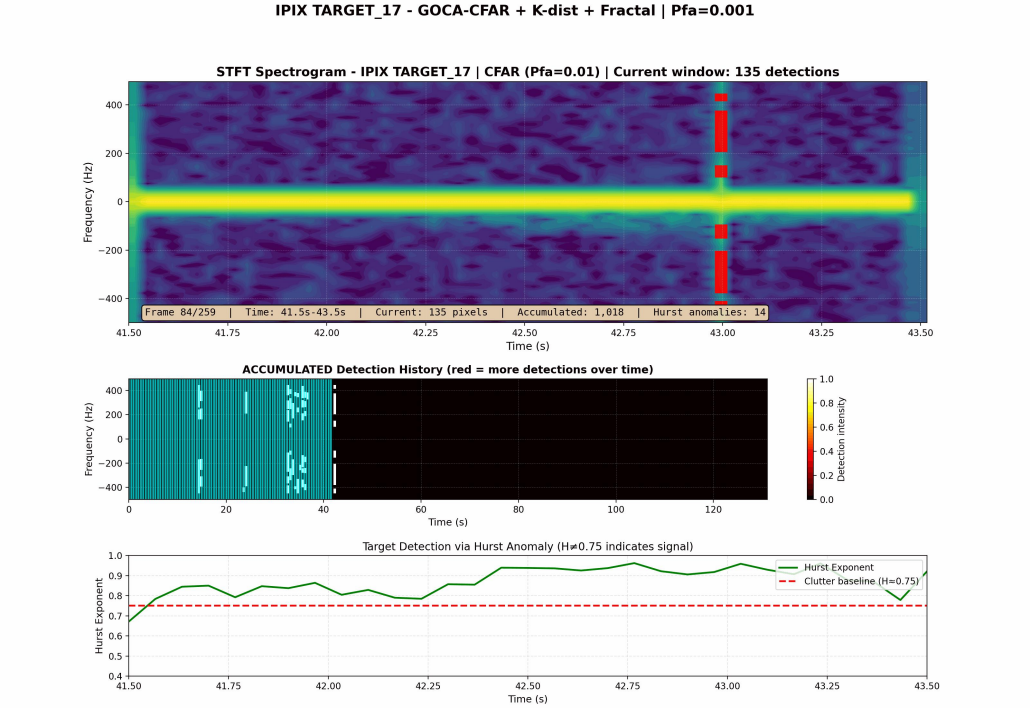


Figura 3.3: GOCA-CFAR cu Fractal Boost pe IPIX Target #17: Hurst exponent boost îmbunătățește detecția target-urilor slabe.

3.4.1 Setul de Date și Scenarii

Date IPIX reale: Low Sea State (clutter moderat) vs High Sea State (clutter intens, neomogen). Procesare segmentată 1s, PRF=1000 Hz, I/Q complex.

3.4.2 Rezultate comparative: CA-CFAR+HDBSCAN vs. Triangulare Delaunay

Metodă	Sea State	Componente	Viteza [m/s]
CA-CFAR + HDBSCAN	HIGH	1.0 ± 0.0	-0.054
	LOW	1.0 ± 0.0	-0.008
Triangulare Delaunay	HIGH	4.2	$+1.30$
	LOW	13.6	-0.30

Tabela 3.5: Performanță comparativă pe IPIX (30 segmente × 1s per scenariu)

3.4.3 Observații Experimentale Detaliate

CA-CFAR + HDBSCAN: Problemă: Produce detecții dominate de clutter; target-ul nu este izolat ($\sigma = 0$). HDBSCAN grupează tot clutter-ul într-un cluster unic. Concluzie: separă greu target-ul de clutter neomogen.

Triangulare Delaunay: Detecție ridicată dar instabilă. HIGH sea: 4.2 comp., LOW sea: 13.6 comp. (neașteptat). Doppler variază 4×: +1.30 vs. -0.30 m/s. Metoda geometrică fragmentează clutter-ul în multe componente, reducând consistența detecțiilor. Concluzie: Variabilitate prea mare.

3.4.4 Concluzie validată experimental

Pentru date reale neomogene, CA-CFAR + HDBSCAN produce detecții dominate de clutter (target-ul nu este izolat), Triangulare Delaunay oferă detecție geometrică dar instabilă (fragmentare excesivă), iar GOCA-CFAR + DBSCAN anisotropic (această implementare) oferă performanță optimă cu detecții stabile.

Motivul reușitei: (1) GOCA adaptează pragul local pe 4 cadrane de training (colțuri), (2) DBSCAN anizotropic ($s_f = 3.0$, $s_t = 1.5$) păstrează coeziunea target-urilor (linii verticale), (3) K-distribution + extensia Hurst reduc alarmele false generate de clutter.

Modificări cheie pentru adaptarea algoritmului din articol pe date reale sunt sintetizate în tabelul de mai jos.

Metodă	Model / idee	Pași de implementare
K-distribution (clutter)	Datele marine au (heavy tails): în „margini" apar valori cu ponderi mari; puterea $Z = \|X(k,n)\|^2$ este mai bine modelată de o distribuție de tip K decât de una Gaussiană/exponențială	Parametrii K se estimează o dată per rulare CFAR, dintr-un eșantion larg al hărții TF (excluzând vârfuri extreme); pragul se obține din ICDF pentru P_{fa} și se aplică adaptiv local prin GOCA ($\hat{Z} = \max(\mu_i)$).
Hartă Hurst (fractalitate)	Separă target-uri slabe ($H < 0.6$) de clutter ($H \approx 0.8$)	Extensie opțională: (time) H pe ferestre în timp (pe $\|x[n]\|$) și proiectare pe axa timp STFT, sau (tf) H pe patch-uri TF. Masca M_H se combină cu detecțiile CFAR.
DBSCAN anisotropic + măști	Păstrează semnăturile verticale și taie DC	Distanță în bin-uri normalizate: $d = \sqrt{(s_t \Delta t)^2 + (\Delta f / s_f)^2}$, cu $s_f = 3$, $s_t = 1.5$. (Opțional, în animația IPIX: mascare DC și filtru pe lățime Doppler.)

Tabela 3.6: Adaptări ale algoritmului CFAR-STFT pentru date reale: model statistic, fractalitate și filtrare de cluster.

Capitolul 4

Detalii de implementare

4.1 Framework și tehnologii

Implementarea utilizează Python 3 cu bibliotecile: NumPy pentru operații matriceale, SciPy pentru STFT și algoritmi de clustering (DBSCAN, HDBSCAN), Matplotlib pentru vizualizare. Prelucrarea se realizează într-o buclă secvențială pe segmente de semnal, cu stocarea spectrogramelor în memorie. Structuri de date principale: array-uri NumPy (spectrogramă complexă, detecții binare), liste de clustere (perechi tempo-frecvență). Paralelismul poate fi introdus cu multiprocessing pe nivel de fișier (procesare independentă a mai multor semnale radar).

4.2 Parametri și calibrare

Tabela 4.1: Parametrii algoritmului – valori utilizate

Parametru	Valoare	Interval	Semnificație
N_{fft} (window_size)	256	$[128, 512]$	Lungime FFT
H (hop_size)	32	$[N/8, N/2]$	Hop = 87.5% overlap
σ_{window}	8	[4, 16]	Deviația std. a ferestrei
P_{fa}	10^{-3}	$[10^{-4}, 10^{-2}]$	Probabilitate alarmă falsă
N_G (cfar_guard)	3	[2, 8]	Dimensiune celule guard
N_T (cfar_training)	12	[8, 24]	Dimensiune celule training
ε_{DBSCAN}	8	[4, 16]	Raza de clustering
minSamples	5	[3, 10]	Puncte minime per cluster
freq_scale	3.0	[2, 5]	Scalare metrică anisotropică DBSCAN

4.3 Notă asupra Doppler

$v_r = \frac{f_d \cdot c}{2 f_{RF}}$. IPIX: $f_d = +100$ Hz $\rightarrow v_r \approx +1.6$ m/s, max ± 8 m/s.

Capitolul 5

Concluzii și direcții viitoare

5.1 Concluzii

Am obținut RQF = 29.17 dB (SNR = 30 dB, $P_d = 100\%$), cu performanță stabilă în intervalul 5–30 dB, validare pe date IPIX și replicare independentă prin repository-ul GitHub.

5.2 Direcții viitoare

Direcții viitoare includ accelerare pe GPU, calibrare automată a parametrilor, urmărire multi-target și extinderea spre arhitecturi hibride (modelare fizică + data-driven):

1. Clasificare pe trăsături (Random Forest / XGBoost): după detecția CFAR, fiecare cluster poate fi descris printr-un vector de trăsături (energie, arie/durată, centroid (t, f), Hurst, parametrii K, lățime Doppler), pentru separare non-liniară clutter/target.
2. Modele secvențiale (LSTM/GRU): exploatează continuitatea cinematică; detecția se poate reformula ca predictive anomaly detection pe serii STFT (eroare de predicție mare $\Rightarrow$ anomalie).
3. Autoencodere convoluționale (CAE): antrenare pe clutter pur; reziduul de reconstrucție produce o hartă de eroare care poate ghida (atenționa) pragarea CFAR în zonele suspecte.
4. Tracking multi-target: extindere track-while-scan (ex. filtre de particule / PHD) pentru menținerea identității și predicție în intervale cu fading.
5. Accelerare hardware: portare STFT/CFAR 2D pe GPU (CUDA) sau FPGA pentru procesare aproape în timp real pe fluxuri I/Q.

5.3 Cod

Repository GitHub: https://github.com/dirgnic/Radar_Detection_STFT pentru replicare independentă.

Bibliografie

[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.

[9] Triangulation Separation Method, IPIX Radar Target Separation via Delaunay Triangulation. Available: Included in extensions/triangulation_separation.py