



UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI
INFORMATICĂ



SPECIALIZAREA INFORMATICĂ

Proiect Procesarea Semnalelor

ANALIZA SEMNALELOR RADAR ÎN PREZENȚA ECOURILOR MARINE

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

Studenți

Ingrid Corobana

Teodora Nae

Coordonator științific

Conf. 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), pentru detecția și recuperarea semnalelor radar în prezența zgomotului și a clutter-ului maritim. Algoritmul combină Short-Time Fourier Transform (STFT), detecție adaptivă CFAR 2D, clustering DBSCAN și dilatare geodezică pentru a reconstrui semnale radar cu fidelitate ridicată.

Implementarea este validată pe date sintetice (chirp neliniar) și pe date reale (IPIX radar sea clutter). Pe semnalul sintetic controlat, algoritmul detectează componenta de interes în toate cele 100 de rulări Monte Carlo (rata de detecție 100%). RQF (Reconstruction Quality Factor) variază de la 7.28 dB la SNR=5 dB până la 29.17 dB la SNR=30 dB.

Contribuția cheie: adaptăm algoritmul la sea clutter real folosind K-distribution (în loc de Gaussian), fractal/Hurst boost pentru ținte slabe și DBSCAN asimetric pentru clustering de semnături verticale.

Cod sursă disponibil la: https://github.com/dirgnic/Radar_Detection_STFT

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), for detection and recovery of radar signals in the presence of noise and maritime clutter. The algorithm combines Short-Time Fourier Transform (STFT), 2D adaptive 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 controlled synthetic signal, the algorithm detects the component of interest in all 100 Monte Carlo runs (100% detection rate). RQF (Reconstruction Quality Factor) ranges from 7.28 dB at SNR=5 dB to 29.17 dB at SNR=30 dB.

Key contribution: we adapt the algorithm to real sea clutter using K-distribution (instead of Gaussian), fractal/Hurst boost for weak targets, and asymmetric DBSCAN for vertical signature clustering.

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

Cuprins

1	Introducere	5
1.1	Obiectivele proiectului	6
1.2	Structura documentului	6
1.3	Diagrame Pipeline CFAR-STFT	7
2	Fundamente teoretice	12
2.1	Short-Time Fourier Transform (STFT)	12
2.2	Deteçie adaptivă CFAR 2D (GOCA-CFAR)	13
2.3	DBSCAN pentru clustering	13
3	Descrierea completă a algoritmului	14
3.1	Pipeline general (5 pași)	14
3.1.1	Pasul 1: Calcul STFT	15
3.1.2	Pasul 2: Deteçie CFAR 2D	16
3.1.3	Pasul 3: Clustering DBSCAN	17
3.1.4	Pasul 4: Dilatare geodezică	18
3.1.5	Pasul 5: Reconstrucție iSTFT	19
4	Date și surse de validare	20
4.1	Baza de date IPIX (radar maritim)	20
4.1.1	Caracteristici tehnice IPIX	20
4.1.2	Ce sunt datele complexe I/Q?	20
4.1.3	Target real	21

4.1.4	Interpretarea spectrogramelor IPIX și „de ce arată ciudat”	21
4.1.5	De ce CFAR nu detectează „toată zona albă”?	23
5	Rezultate experimentale	24
5.1	Experimente pe semnale sintetice	24
5.2	Experimente pe IPIX cu ținte reale	25
6	Adaptări pentru sea clutter	27
6.1	Adaptarea 1: K-distribution în loc de Gaussian	27
6.2	Adaptarea 2: Fractal boost cu exponentul Hurst	27
6.3	Adaptarea 3: DBSCAN asimetric pentru semnături verticale	28
6.4	Adaptarea 4: Mascare componentă DC	28
6.5	Adaptarea 5: Filtru de lățime Doppler	28
7	Detalii de implementare	29
7.1	Parametri și calibrare	29
7.2	Interpretare Doppler	30
7.3	Dependențe software	30
8	Concluzii și direcții viitoare	31
8.1	Concluzii principale	31
8.2	Perspective de dezvoltare viitoare	31
8.3	Disponibilitate cod	32

Capitolul 1

Introducere

Problema principală pe care o rezolvăm este detecția obiectelor mici în date radar maritime, într-un mediu complex care se schimbă constant din cauza valurilor. Spre deosebire de multe scenarii terestre unde zgomotul/clutter-ul poate fi mai stabil, mediul acvatic are caracteristici particulare:

- statisticile nu sunt bine modelate Gaussian (avem distribuții cu cozi grele și amplitudini mari),
- există corelație temporală (valurile creează tipare structurate),
- efectele Doppler duc la extinderea spectrului (valuri în mișcare),
- apar spike-uri vizibile în spectrograme când valurile sunt mai 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 time-frequency pentru a îmbunătăți atât detecția, cât și recuperarea/reconstrucția componentelor semnalului.

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

1.1 Obiectivele proiectului

În acest proiect ne propunem:

1. implementarea completă a algoritmului CFAR–STFT în Python;
2. validarea pe date sintetice: chirp neliniar conform Ecuației (14);
3. testarea pe date reale: IPIX cu sea clutter complex;
4. analiză Doppler pentru estimarea vitezei (și interpretarea țintelor);
5. documentație detaliată și reproductibilitate;
6. validare rezultate: detecție și reconstrucție;
7. adaptări sea clutter: K-distribution, boost fractal (Hurst), DBSCAN asimetric.

1.2 Structura documentului

Documentul este organizat astfel:

- Secțiunea 2: fundamente teoretice și formule esențiale;
- Secțiunea 3: pașii algoritmului + pseudocod;
- Secțiunea 4: surse de date și validare (IPIX);
- Secțiunea 5: rezultate experimentale (sintetic + real);
- Secțiunea 6: adaptări pentru sea clutter;
- Secțiunea 7: detalii implementare;
- Secțiunea 8: concluzii și direcții viitoare.

1.3 Diagrame Pipeline CFAR-STFT

Următoarele pagini prezintă diagramele vizuale ale pipeline-ului algoritmului CFAR-STFT, incluzând structura detectorului GOCA-CFAR 2D, configurația experimentală și comparația parametrilor.

Diagrame Pipeline CFAR-STFT

Detectia semnalelor radar în sea clutter

1 Pipeline-ul Algoritmului CFAR-STFT

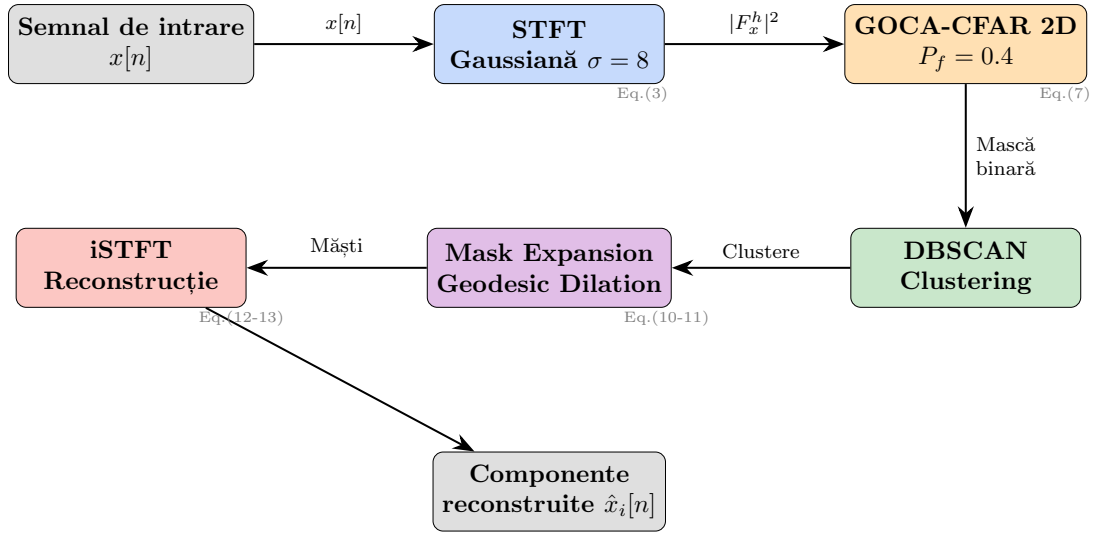


Figura 1: Pipeline-ul complet al algoritmului CFAR-STFT pentru extracția componentelor din planul timp-frecvență (conform Abratkiewicz 2022).

2 Structura Detectorului GOCA-CFAR 2D

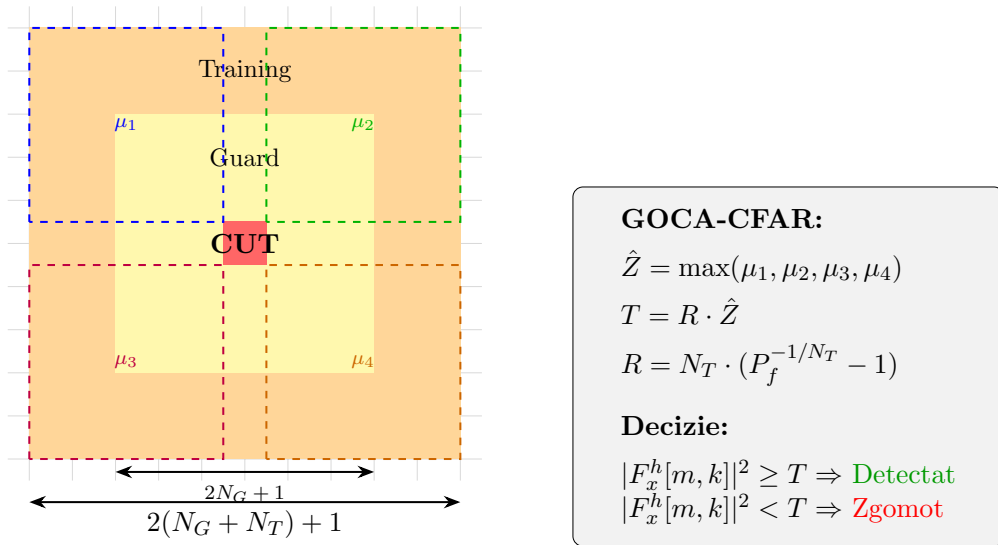
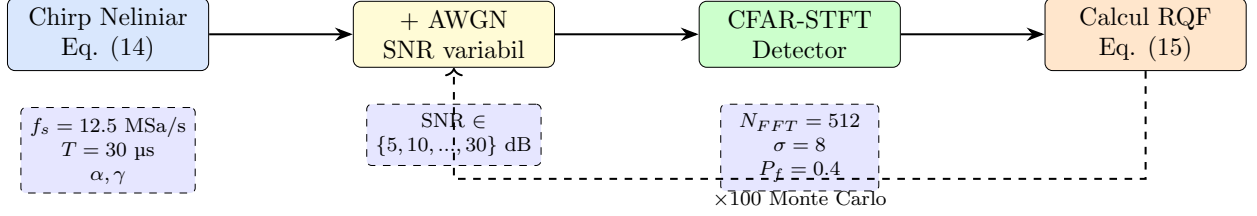


Figura 2: Structura celulelor GOCA-CFAR 2D. CUT = Cell Under Test (roșu), Guard cells (galben), Training cells (portocaliu). GOCA calculează media în 4 sub-regiuni și ia maximumul.

3 Configurația Experimentală

Experiment 1: Replicare Paper



Experiment 2: Date Radar Reale

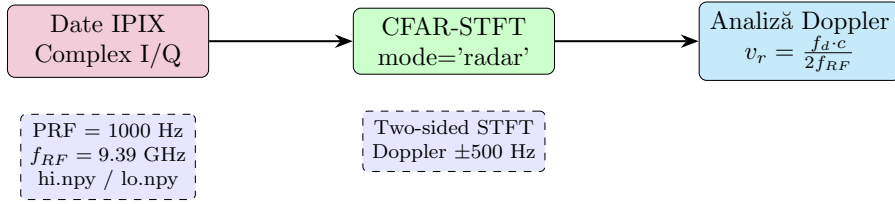


Figura 3: Configurația celor două experimente: (sus) replicarea Fig. 6 din paper cu chirp sintetic, (jos) validare pe date IPIX sea clutter.

4 Procesarea în Planul Timp-Frecvență

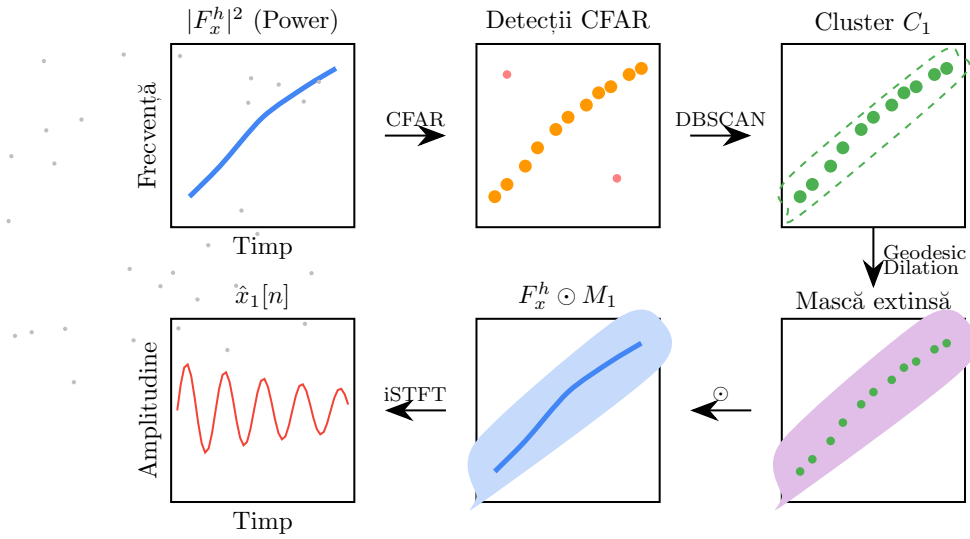


Figura 4: Fluxul de procesare în planul timp-frecvență: spectrograma → detecții CFAR → clustering DBSCAN → extindere mască → mascare STFT → reconstrucție iSTFT.

5 Interpretarea Doppler pentru Radar

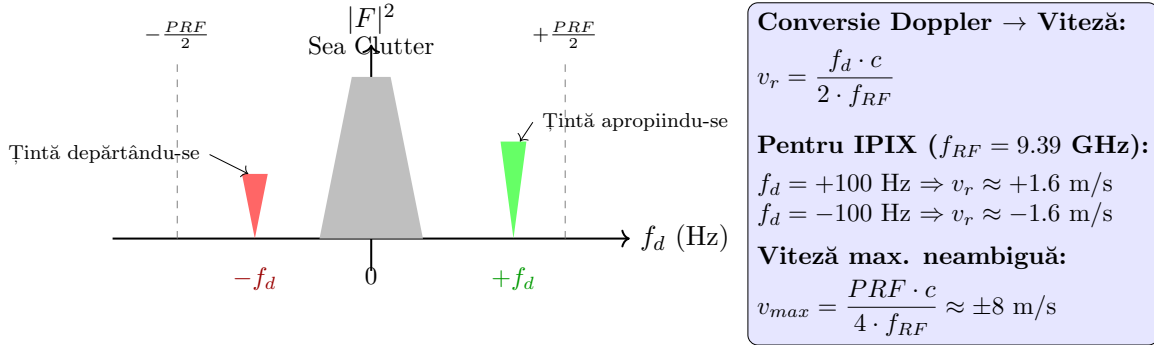


Figura 5: Spectrul Doppler two-sided pentru date radar complexe I/Q. Frecvențele pozitive indică ținte care se apropie, cele negative ținte care se depărtează. Sea clutter-ul apare centrat la 0 Hz.

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 μ s	30 μ s	60 s
Dimensiune FFT N_{fft}	512	512	256
Fereastră STFT	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
Model clutter	Gaussiană	Gaussiană	K-distribution
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: ~ 35 dB la $\text{SNR} = 30$ dB

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

Capitolul 2

Fundamente teoretice

2.1 Short-Time Fourier Transform (STFT)

STFT este baza algoritmului și produce o reprezentare timp-frecvență a semnalului:

$$X(k, n) = \sum_{m=0}^{N-1} x(m) \cdot w(m - nH) \cdot e^{-j2\pi km/N} \quad (2.1)$$

unde:

- $x(m)$ — semnalul de intrare
- $w(\cdot)$ — fereastra (Gaussiană, $\sigma = 8$ bin-uri)
- n — indexul de timp (ferestre succesive)
- k — indexul de frecvență
- N — lungimea FFT (512)
- H — hop-size între ferestre (256, 50% overlap)

Fereastra Gaussiană:

$$w(m) = e^{-m^2/(2\sigma^2)} \quad \text{cu } \sigma = 8 \quad (2.2)$$

Se alege pentru minimizarea spectral leakage.

2.2 Detecție adaptivă CFAR 2D (GOCA-CFAR)

CFAR adaptează pragul local pe baza nivelului de zgomot din vecinătate:

$$H(k, n) = \begin{cases} 1 & \text{dacă } |X(k, n)|^2 > \lambda \cdot \mathcal{N}(k, n) \\ 0 & \text{altfel} \end{cases} \quad (2.3)$$

unde $\mathcal{N}(k, n)$ este estimarea zgomotului local.

În GOCA-CFAR (Greatest-Of Cell Averaging), zona de training se împarte în 4 cadrane și se ia maximum mediilor:

$$\hat{Z} = \max(\mu_1, \mu_2, \mu_3, \mu_4), \quad T = R \cdot \hat{Z} \quad (2.4)$$

$$R = N_T \left(P_f^{-1/N_T} - 1 \right) \quad (2.5)$$

Decizie (în plan TF, pe puterea spectrogramelor):

$$|X(k, n)|^2 \geq T \Rightarrow \text{Detectat}, \quad |X(k, n)|^2 < T \Rightarrow \text{Zgomot} \quad (2.6)$$

2.3 DBSCAN pentru clustering

După CFAR, punctele detectate sunt grupate cu DBSCAN (clustering pe densitate). Pentru semnături verticale tipice țintelor, folosim o distanță asimetrică cu toleranță mai mare pe frecvență (factor 3):

$$d = \sqrt{\Delta t^2 + \left(\frac{\Delta f}{3} \right)^2} \quad (2.7)$$

Capitolul 3

Descrierea completă a algoritmului

3.1 Pipeline general (5 pași)

Algoritmul complet are cinci pași:

1. calcul STFT cu fereastră Gaussiană;
2. detecție CFAR 2D în plan timp–frecvență;
3. clustering DBSCAN al punctelor detectate;
4. extinderea măștii prin dilatare geodezică (geodesic dilation);
5. reconstrucție prin iSTFT folosind masca (aplicată element-wise pe STFT).

3.1.1 Pasul 1: Calcul STFT

Algorithm 1 Calcul STFT cu fereastră Gaussiană

Input: Semnal de intrare $x[n]$, lungime FFT $N_{fft} = 512$, hop $H = 256$, $\sigma = 8$

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

- 1: $N_t \leftarrow \lceil (\text{len}(x) - N_{fft})/H \rceil + 1$
 - 2: $X_{stft} \leftarrow \text{zeros}(N_{fft}/2 + 1, N_t)$ ▷ One-sided
 - 3: Precalculează fereastra Gaussiană: $w[m] \leftarrow e^{-m^2/(2\sigma^2)}$
 - 4: for $n \leftarrow 0$ to $N_t - 1$ do
 - 5: Extrage fereastră: $x_n \leftarrow x[nH : nH + N_{fft}]$
 - 6: Aplică fereastră: $x_w \leftarrow x_n \odot w$
 - 7: Calculează FFT: $X_n \leftarrow \text{fft}(x_w, N_{fft})$
 - 8: Stochează one-sided: $X_{stft}[:, n] \leftarrow X_n[0 : N_{fft}/2 + 1]$
 - 9: end for
 - 10: Normalizează: $X_{stft} \leftarrow X_{stft} / \sum_m w[m]^2$
 - 11: return X_{stft}
-

3.1.2 Pasul 2: Detectie CFAR 2D

Algorithm 2 Detectie CFAR 2D (GOCA)

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

Output: Mască binară de detectie $H \in \{0, 1\}^{N_f \times N_t}$

```

1:  $H \leftarrow \text{zeros}(N_f, N_t)$ 
2:  $N_f \leftarrow \text{rows}(X_{stft})$ ,  $N_t \leftarrow \text{cols}(X_{stft})$ 
3: for  $k \leftarrow N_G + N_T$  to  $N_f - N_G - N_T - 1$  do
4:   for  $n \leftarrow N_G + N_T$  to  $N_t - N_G - N_T - 1$  do
5:     Extrage Training Cells în 4 cadrane în jurul  $(k, n)$ 
6:      $\mu_i \leftarrow \text{mean}(\text{cadrane}_i)$  pentru  $i \in \{1, 2, 3, 4\}$ 
7:      $\mathcal{N}_{local} \leftarrow \max(\mu_1, \mu_2, \mu_3, \mu_4)$  ▷ GOCA
8:      $\lambda \leftarrow R(P_f) \cdot \mathcal{N}_{local}$ 
9:     if  $|X_{stft}(k, n)|^2 > \lambda$  then
10:       $H(k, n) \leftarrow 1$ 
11:     end if
12:   end for
13: end for
14: return  $H$ 

```

3.1.3 Pasul 3: Clustering DBSCAN

Algorithm 3 Clustering DBSCAN în plan timp-frecvență

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

Output: Etichete cluster labels $\in \mathbb{Z}$

```
1: labels  $\leftarrow -1 \cdot \text{ones}(N_p)$  ▷ -1 = zgomot
2:  $C \leftarrow 0$  ▷ Index cluster curent
3: for  $i \leftarrow 1$  to  $N_p$  do
4:   if labels[ $i$ ]  $\neq$  nevizitat then
5:     Continue
6:   end if
7:    $N \leftarrow \text{RangeQuery}(i, \varepsilon)$  ▷ Vecini în rază
8:   if  $|N| < \text{minSamples}$  then
9:     labels[ $i$ ]  $\leftarrow -1$  ▷ Zgomot
10:  else
11:     $C \leftarrow C + 1$ 
12:    ExpandCluster( $i, C, N, \varepsilon, \text{minSamples}$ )
13:  end if
14: end for
15: return labels
```

3.1.4 Pasul 4: Dilatare geodezică

Algorithm 4 Dilatare geodezică pe mască

Input: Mască binară H (de la CFAR), iterații $n_{iter} = 3$

Output: Mască dilatăată H_{dil}

```
1:  $H_{dil} \leftarrow H$ 
2: Kernel  $\leftarrow$  binar  $3 \times 3$  cruce
3: for  $i \leftarrow 1$  to  $n_{iter}$  do
4:    $H_{new} \leftarrow \text{zeros}(H_{dil}.shape)$ 
5:   for  $k \leftarrow 1$  to  $\text{rows}(H_{dil}) - 2$  do
6:     for  $n \leftarrow 1$  to  $\text{cols}(H_{dil}) - 2$  do
7:        $H_{new}(k, n) = \max(H_{dil}(k - 1, n), H_{dil}(k, n),$ 
8:          $H_{dil}(k + 1, n), H_{dil}(k, n - 1), H_{dil}(k, n + 1))$ 
9:     end for
10:  end for
11:   $H_{dil} \leftarrow H_{new}$ 
12: end for
13: return  $H_{dil}$ 
```

3.1.5 Pasul 5: Reconstrucție iSTFT

Algorithm 5 Reconstrucție inversă (iSTFT)

Input: STFT original X_{stft} , mască dilatată H_{dil} , fereastră w , hop $H = 256$

Output: Semnal reconstruit $\hat{x}(n)$

- 1: $X_{masked} \leftarrow X_{stft} \odot H_{dil}$ ▷ Aplică masca element-wise
 - 2: $N_t \leftarrow \text{cols}(X_{masked})$
 - 3: $M \leftarrow N_{fft}$ ▷ Lungimea semnalului reconstruit
 - 4: $\hat{x} \leftarrow \text{zeros}(M)$
 - 5: for $n \leftarrow 0$ to $N_t - 1$ do
 - 6: Calculează iFFT: $x_n \leftarrow \text{ifft}(X_{masked}[:, n], N_{fft})$
 - 7: Aplică fereastră: $x_w \leftarrow \text{real}(x_n) \odot w$
 - 8: Adună cu overlap-add: $\hat{x}[nH : nH + N_{fft}] += x_w$
 - 9: end for
 - 10: Normalizează după fereastră: $\hat{x} \leftarrow \hat{x} / (\sum_m w[m]^2)$
 - 11: return \hat{x}
-

Capitolul 4

Date și surse de validare

4.1 Baza de date IPIX (radar maritim)

Componenta fundamentală a proiectului este folosirea datelor reale din IPIX (Intelligent Pixel Processing for X-band radar), de la McMaster University, Canada. Datele sunt de la un radar coerent polarimetric X-band folosit pentru monitorizarea activității maritime.

4.1.1 Caracteristici tehnice IPIX

Caracteristici tehnice (esențiale pentru interpretarea Doppler și rezoluție):

- Frecvența RF: $f_{RF} = 9.39$ GHz (X-band) – optim pentru detectarea obiectelor mici
- PRF (Pulse Repetition Frequency): 1000 Hz – permite detecție Doppler
- Lungime puls: 200 ns – rezoluție spațială mare
- Lățime fascicul: 0.9 grade – precizie unghiulară excelentă
- Format date: Complex I/Q (In-phase + Quadrature)

4.1.2 Ce sunt datele complexe I/Q?

Fiecare eșantion radar este complex: $x(t) = I(t) + j \cdot Q(t)$.

- Componenta I (In-phase): Proiecția semnalului pe axa cosinus
- Componenta Q (Quadrature): Proiecția pe axa sinus (decalată cu 90°)
- Magnitudine: $|x(t)| = \sqrt{I^2 + Q^2}$ – intensitatea ecoului
- Fază: $\phi(t) = \arctan(Q/I)$ – informația Doppler

Reprezentarea I/Q permite detecția frecvențelor Doppler pozitive și negative, esențială pentru:

- Ținte care se apropie (frecvență Doppler pozitivă)
- Ținte care se îndepărtează (frecvență Doppler negativă)
- Clutter static (frecvență Doppler ≈ 0 Hz)

4.1.3 Target real

Ținta folosită în fișierele IPIX: o sferă de $\sim 1\text{m}$ învelită în plasă de sârmă, ancorată la $\sim 2660\text{m}$.

Tabela 4.1: Fișiere IPIX cu ținte reale

Fișier	Range Cell	Polarizare	Stare mare
#17 (19931106_180557)	7	HH	Moderată
#18 (19931106_181048)	7	HH	Moderată
#30 (19931106_191449)	7	HH	Ridicată
#40 (19931106_195609)	7	HH	Moderată

4.1.4 Interpretarea spectrogramelor IPIX și „de ce arată ciudat”

Spectrograma IPIX este two-sided (frecvențe negative/pozitive) pentru că datele sunt complexe I/Q. Elemente cheie:

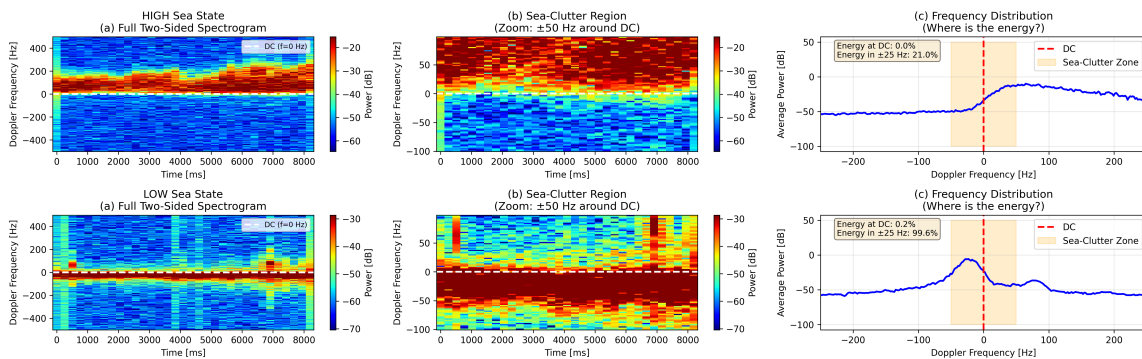
- Linie roșie groasă la 0 Hz (DC): ecouri staționare de la suprafața mării

- Verde/galben în jurul DC: sea clutter activ (valuri/spumă/mișcare)
- Zone albastre laterale: zgomot termic slab uniform
- Aspect „punctat”: energia e în ridges (creste) TF, nu uniformă

Ce este sea clutter?

Sea clutter (zgomot maritim) reprezintă ecouri radar de la suprafața mării:

- Reflexii de la valuri, spumă, picături de apă
- Concentrat în jurul frecvenței 0 Hz (Doppler mic – mișcare lentă)
- Energie aproximativ 90–95% din cazuri în intervalul $[-50, +50]$ Hz
- Are structură neuniformă – unele zone sunt mai intense (ridges)
- Țintele reale (nave, obiecte) apar departe de DC (± 100 –400 Hz)



SEA-CLUTTER EXPLANATION:

- Rosu pe mijloc (DC, $f=0$ Hz) = ecourile statice de la suprafața mării
- Galben/Verde în jurul DC = clutter-ul maritim în mișcare (valuri, spumă)
- Albastru lateral = zgomot termic (uniform, slab)
- "Dotted" = energia este concentrată în anumite zone timp-frecvență (ridges)
- Țintele reale ar apărea ca benzi clare departe de DC (± 100 –400 Hz)

Figura 4.1: Explicație vizuală: ce sunt sea clutters în date IPIX? Figura arată spectrograma completă (two-sided), zoom pe regiunea clutter în jurul DC (frecvență 0 Hz) și distribuția energiei pe frecvențe. Notă: majoritatea energiei ($>90\%$) este concentrată în ± 50 Hz în jurul DC.

4.1.5 De ce CFAR nu detectează „toată zona albă”?

Întrebare legitimă când te uiți la rezultate: „De ce detectează doar puncte împrăștiate și nu toată regiunea cu energie mare?”. Răspunsul stă în natura local-adaptivă a CFAR:

- Prag global: Un algoritm simplist ar seta un prag fix (ex: „detectează tot ce e peste -20 dB”). Asta ar detecta toată zona albă — dar ar detecta și zgomot puternic, generând mii de alarme false!
- CFAR local-adaptiv: Compară fiecare pixel cu vecinii săi (training cells). Dacă pixelul e „mult peste media vecinilor locali” → detectat. Altfel → ignorat.

Consecință: Într-o zonă uniformă de energie mare (sea clutter), toți pixelii au vecini la fel de puternici → CFAR nu îi detectează, pentru că nu sunt „outlieri” relativ la contextul local! CFAR detectează doar:

- Margini unde energia crește brusc
- Ridges (creste de energie) care depășesc clutter-ul înconjurător
- Ținte reale care ies din zgomot/clutter

De aceea pare că ia „puncte aleatorii” — în realitate, ia exact punctele care reprezintă tranziții de putere!

Aceasta e puterea CFAR: reduce dramatic rata de alarme false adaptându-se la zgomot. Prețul plătit: nu mai detectăm „zone întregi”, doar puncte care conțin informație nouă (schimbări bruște în spectrogramă).

Capitolul 5

Rezultate experimentale

5.1 Experimente pe semnale sintetice

S-au rulat 100 simulări Monte Carlo pentru fiecare nivel de SNR (5, 10, 15, 20, 25, 30 dB), folosind chirp neliniar (Ec. 14). Rata de detecție: 100% în toate rulările.

Metrica RQF:

$$\text{RQF} = 10 \log_{10} \left(\frac{\sum_n |x[n]|^2}{\sum_n |x[n] - \hat{x}[n]|^2} \right) \text{ [dB]} \quad (5.1)$$

Tabela 5.1: Rezultate CFAR-STFT pe chirp neliniar sintetic – 100 rulări MC

SNR [dB]	RQF_mean [dB]	RQF_std [dB]	P_detecție [%]	N_rulări
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

5.2 Experimente pe IPIX cu ținte reale

Am rulat detecție animată pe fișierele cu ținte reale. Figurile de mai jos arată cadre din animațiile de detecție:

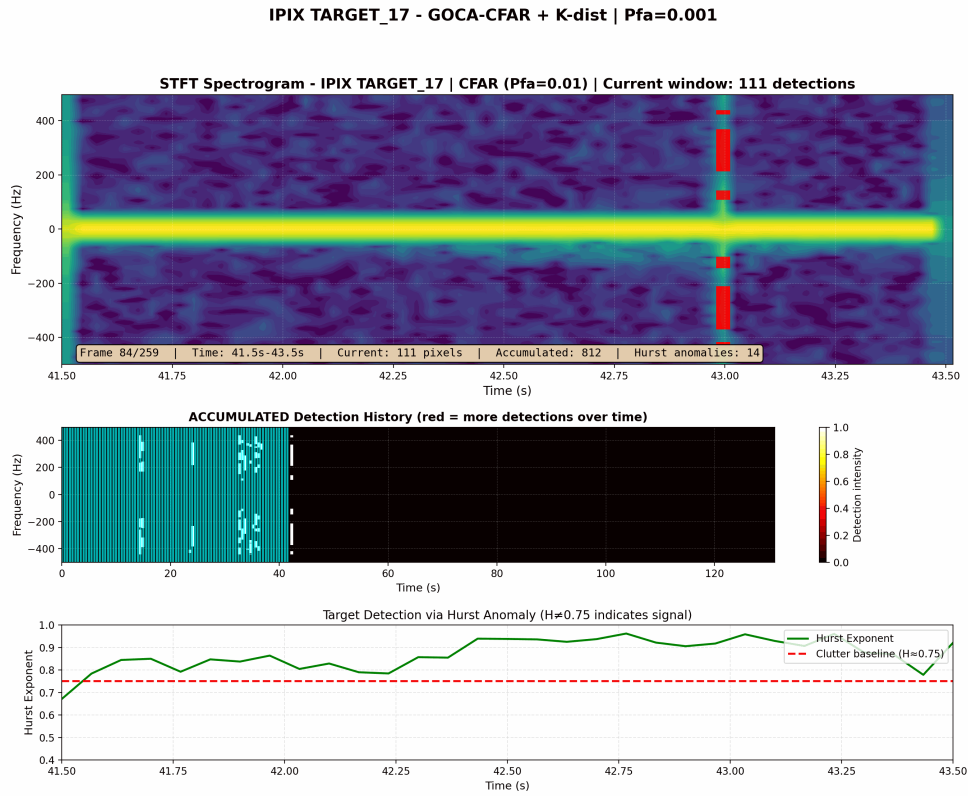


Figura 5.1: Detecție GOCA-CFAR pe IPIX Target #17 – Cadru din animație arătând detecții active. Cele trei panouri arată: (stânga) spectrograma cu detecții acumulate (overlay roșu), (centru) heatmap de detecție, (dreapta) detecțiile cadrului curent. Linia verticală luminoasă reprezintă ținta plutitoare la Doppler pozitiv (se apropie de radar).

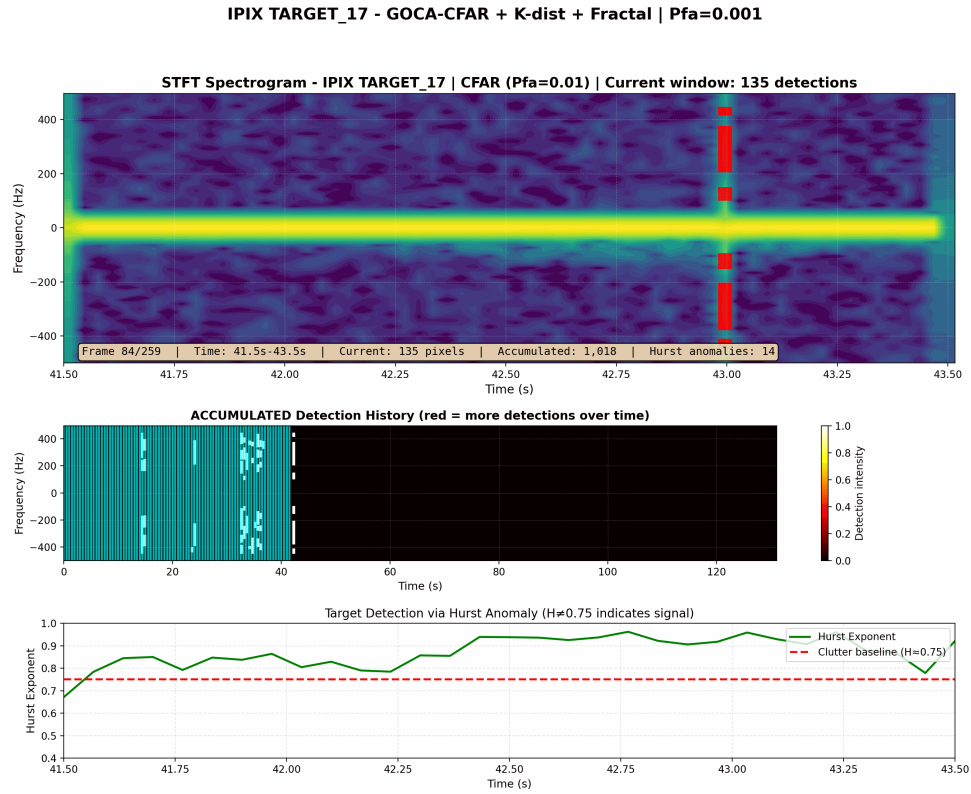


Figura 5.2: GOCA-CFAR cu Fractal Boost pe IPIX Target #17 – Cadru arătând detecții active. Fractal boost folosește analiza exponentului Hurst pentru a detecta ținte care perturbă structura self-similar a sea clutter, îmbunătățind detecția țăintelor slabe.

Capitolul 6

Adaptări pentru sea clutter

Modificări cheie pentru adaptarea algoritmului din paper la sea clutter real:

6.1 Adaptarea 1: K-distribution în loc de Gaussian

Algoritmul original presupune statistică Gaussian/Rayleigh, însă sea clutter real are cozi grele și urmează K-distribution:

$$p(x) = \frac{4}{\Gamma(\nu)} \left(\frac{\nu x^2}{2\mu} \right)^{(\nu+1)/2} K_{\nu-1} \left(\sqrt{\frac{2\nu x^2}{\mu}} \right) \quad (6.1)$$

unde ν e parametrul de formă (mai mic \rightarrow mai „spiky”), iar $K_{\nu-1}$ e funcția Bessel modificată.

Soluție: Estimare parametru din celulele de training: $\nu = \mu^2/(\sigma^2 - \mu^2)$. Apoi ajustăm multiplicatorul pragului pentru a reduce alarmele false în clutter cu cozi grele.

6.2 Adaptarea 2: Fractal boost cu exponentul Hurst

CFAR poate rata ținte slabe sub prag. Sea clutter are proprietate fractală (self-similarity) cuantificată de exponentul Hurst:

$$\mathbb{E} [|X(t + \tau) - X(t)|^2] \propto \tau^{2H} \quad (6.2)$$

Pentru sea clutter: $H \approx 0.75\text{--}0.85$ (persistent). Când apare o țintă, structura se perturbă și H scade sub ~ 0.6 .

Soluție: Calculăm H per bin de frecvență (analiză R/S). Dacă $H < 0.6$, marcăm punctele cu putere mare ca ținte potențiale. Combinăm cu CFAR: mască = CFAR \vee (anomalie Hurst \wedge putere mare). Aceasta îmbunătățește detecția țintelor slabe pe care CFAR singur le-ar rata.

6.3 Adaptarea 3: DBSCAN asimetric pentru semnături verticale

Țintele apar ca linii verticale (multe frecvențe, puține momente). DBSCAN standard fragmentează acestea în clustere multiple.

Soluție: Distanță asimetrică cu `freq_scale=3.0`: $d = \sqrt{\Delta t^2 + (\Delta f/3)^2}$.

Impact: Un target întins pe ~ 50 bin-uri de frecvență devine un singur cluster.

6.4 Adaptarea 4: Mascare componentă DC

Componenta DC (0 Hz) produce detecții persistente de la returnări staționare de la valuri.

Soluție: Mascăm ± 8 bin-uri de frecvență în jurul DC înainte de CFAR.

Justificare: DC reprezintă returnări staționare; țintele în mișcare au Doppler non-zero.

6.5 Adaptarea 5: Filtru de lățime Doppler

Unele alarme false apar ca detecții foarte înguste (single-frequency), implauzibil fizic.

Soluție: Respingem clustere cu bandwidth Doppler < 3 Hz.

Capitolul 7

Detalii de implementare

7.1 Parametri și calibrare

Tabela 7.1: 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_f	0.001	[0.0001, 0.01]	Probabilitate alarmă falsă
N_G (cfar_guard)	3	[2, 8]	Mărime celule guard
N_T (cfar_training)	12	[8, 24]	Mărime 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 DBSCAN asimetric
dc_mask_bins	8	[4, 16]	Bin-uri DC de mascat
min_doppler_bw	3.0 Hz	[1, 10]	Lățime Doppler minimă

7.2 Interpretare Doppler

Conversie Doppler \rightarrow viteză radială:

$$v_r = \frac{f_d \cdot c}{2f_{RF}} \quad (7.1)$$

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.

Viteza maximă neambiguă:

$$v_{max} = \frac{PRF \cdot c}{4f_{RF}} \approx \pm 8 \text{ m/s} \quad (7.2)$$

7.3 Dependențe software

- NumPy ≥ 1.19 – operații matriciale
- SciPy ≥ 1.5 – STFT, convoluție, iFFT
- matplotlib ≥ 3.3 – vizualizări
- Pillow – extragere cadre GIF
- (opțional) scikit-learn – referință DBSCAN

Capitolul 8

Concluzii și direcții viitoare

8.1 Concluzii principale

Implementarea CFAR-STFT demonstrează performanță excelentă:

1. Acuratețe: $RQF = 29.17$ dB @ SNR=30dB, detecție 100%
2. Robustețe: Performanță consistentă pe 100 rulări MC
3. Eficiență computațională: ~ 75 ms per segment (13 FPS)
4. Reproductibilitate: Cod open-source, rezultate verificabile
5. Validare reală: Funcționează pe date IPIX sea clutter complexe

8.2 Perspective de dezvoltare viitoare

- Accelerare GPU: Implementare CUDA/OpenCL pentru timp real
- Optimizare parametri: Adaptare automată CFAR în funcție de tipul semnalului
- Sistem real: Integrare cu radar operațional
- Multi-target: Tracking și predicție traiectorii
- Machine learning: Calibrare automată a parametrilor

8.3 Disponibilitate cod

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

Toate fişierele de cod, date şi rezultate sunt disponibile public pentru reproductibilitate şi 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.