UNIVERSITATEA TEHNICĂ DIN CLUJ-NAPOCA

FACULTATEA DE ELECTRONICĂ,
TELECOMUNICAȚII ȘI TEHNOLOGIA INFORMAȚIEI

Inginerie Electronică, Telecomunicații și Tehnologii Informaționale

Proiect Radiocomunicații
„Interferenta intre radarele FMCW”
an III sem. II

Student,
Mathe-Leszay Erik-Robert
Tiselita David

Profesor coordonator,
Cristian Codau

2025

Cuprins

I.Pagina principala

Crearea GUI-ului principal

```
function main_gui
    % Creează fereastra principală
    fig = uifigure('Name', 'PROIECT RC', 'Position', [100 200 1350 550]);
```

Setare rezoluție radar (range resolution)

```
global rangeRes
   rangeRes = 1; % Valoare inițială
```

rangeRes este rezoluția în distanță a radarului – adică cât de aproape pot fi două obiecte și totuși să fie detectate separat.

Formula rezoluției radarului FMCW:

$$\text{Rezoluție} = \frac{c}{2 \cdot B}$$

unde:

- c = viteza luminii ≈ 3e8 m/s
- B = lățimea de bandă a semnalului chirp

Dacă vrei o rezoluție mai bună (mai mică), trebuie o **bandă mai mare**.

Afișare imagini în GUI

```
ax = uiaxes(fig, ...
        'Position', [370 -100 500 550]);  % poziționează unde vrei în GUI
img = imread('1.png');
    imshow(img, 'Parent', ax);
ax = uiaxes(fig, ...
        'Position', [870 -100 500 1050]);  % poziționează unde vrei în GUI
img = imread('3.png');
    imshow(img, 'Parent', ax);
ax = uiaxes(fig, ...
        'Position', [870 -100 500 550]);  % poziționează unde vrei în GUI
img = imread('4.png');
    imshow(img, 'Parent', ax);
```

Etichete cu text explicativ (uilabel)

```
    lbl = uilabel(fig, ...
        'Text', 'Rezoluția radarului (m):', ...
        'Position', [100 160 200 22]);
lbl2 = uilabel(fig, ...
        'Text', ' Interferenta FMCW(unda continuu modulate prin frecventa)', ...
        'Position', [400 460 400 122]);
lbl3 = uilabel(fig, ...
        'Text', '  intre radarele automotive', ...
        'Position', [500 460 400 102]);
lbl4 = uilabel(fig, ...
        'Text', ' Radarele astea sunt folosite în mașinile moderne pentru chestii ca:', ...
        'Position', [400 460 400 52]);
lbl5 = uilabel(fig, ...
        'Text', ' --asistență la frânare', ...
        'Position', [400 460 400 32]);
lbl6 = uilabel(fig, ...
        'Text', ' --detectare obstacole, pietoni, alte mașini etc.', ...
        'Position', [400 360 400 202]);
lbl7 = uilabel(fig, ...
        'Text', ' În loc să trimită „pulse” scurte de semnal (ca unele radare militare)', .
        'Position', [400 360 400 162]);
lbl8 = uilabel(fig, ...
        'Text', ', radarul FMCW trimite un semnal continuu care își schimbă frecvența ', ..
        'Position', [400 360 400 142]);
lbl9 = uilabel(fig, ...
        'Text', ' în timp (crește sau scade – asta se numește „chirp”).', ...
        'Position', [400 360 400 122]);
lbl9 = uilabel(fig, ...
        'Text', 'Imaginează-ți două mașini una lângă alta, fiecare cu radarul ei.', ...
        'Position', [400 360 400 82]);
```

```
lbl7 = uilabel(fig, ...
        'Text', ' În loc să trimită „pulse” scurte de semnal (ca unele radare militare)', ...
        'Position', [400 360 400 162]);
lbl8 = uilabel(fig, ...
        'Text', ', radarul FMCW trimite un semnal continuu care își schimbă frecvența ', ...
        'Position', [400 360 400 142]);
lbl9 = uilabel(fig, ...
        'Text', ' în timp (crește sau scade – asta se numește „chirp”).', ...
        'Position', [400 360 400 122]);
lbl9 = uilabel(fig, ...
        'Text', 'Imaginează-ți două mașini una lângă alta, fiecare cu radarul ei.', ...
        'Position', [400 360 400 82]);
lbl9 = uilabel(fig, ...
        'Text', '  Ambele emit semnale FMCW.Problema e că semnalele se pot amesteca.', ...
        'Position', [400 360 400 62]);
lbl9 = uilabel(fig, ...
        'Text', ' Radarul tău poate primi semnalul emis de altă mașină, crezând ', ...
        'Position', [400 360 400 42]);
lbl9 = uilabel(fig, ...
        'Text', 'că e al lui întors de la un obstacol.', ...
        'Position', [400 360 400 22]);

lbl9 = uilabel(fig, ...
        'Text', 'Producătorii de radare și mașini încearcă tot felul de soluții, gen:', ...
        'Position', [400 260 400 182]);
lbl9 = uilabel(fig, ...
        'Text', 'Frecvențe ușor diferite pentru fiecare radar – ca să nu se calce', ...
        'Position', [400 260 400 162]);
lbl9 = uilabel(fig, ...
        'Text', ' pe bătătură.Rezolutia radarului mai mica.Numarul de sweepuri Doppler', ...
        'Position', [400 260 400 142]);
lbl9 = uilabel(fig, ...
        'Text', ' mai multe', ...
        'Position', [400 260 400 122]);
```

Slider pentru rezoluția radarului

```
sld = uislider(fig, ...
    'Limits', [1 2], ...
    'Value', rangeRes, ...
    'Position', [100 140 200 3], ...
    'ValueChangedFcn', @(sld,event) updateRangeRes(sld));
```

Slider între 1 și 2 metri — permite utilizatorului să schimbe rezoluția radarului dinamic.

Doppler sweeps – detectare viteze

```
global dopplerSweeps
dopplerSweeps = 192; % Valoare default
```

În radar, pentru a estima viteza unei ținte se folosește efectul Doppler:

$$f_d = \frac{2v}{\lambda}$$

unde:

- f_d = frecvența Doppler
- v = viteza țintei
- λ = lungimea de undă

🔹 Cu cât ai mai multe sweep-uri, cu atât ai o rezoluție Doppler mai bună, adică poți distinge viteze apropiate.

```
uilabel(fig, ...
    'Text', 'Număr sweep-uri Doppler:', ...
    'Position', [100 420 200 22]);

sldDoppler = uislider(fig, ...
    'Limits', [64 512], ...
    'MajorTicks', [64, 128, 192, 256, 384, 512], ...
    'Value', dopplerSweeps, ...
    'Position', [100 400 200 3], ...
    'ValueChangedFcn', @(sldDoppler, event) updateDopplerSweeps(sldDoppler));
```

Offset de frecvență între radare (interferență)

```
% Slider pentru offset de frecvență între radare
   global freqOffsetFactor
   freqOffsetFactor = 1; % valoare inițială = 1x (adică fără offset)
```

$$f_2 = f_1 \cdot \text{offset}$$

Buton pentru a porni simularea

```
    % Creează butonul
    btn = uibutton(fig, ...
        'Text', 'Start Simulare FMCW radar', ...
        'Position', [100 60 250 40], ...
        'ButtonPushedFcn', @(btn,event) start_project(fig));
end
```

II.Funcții de actualizare pentru slideri

Funcțiile

```
function updateRangeRes(sld)
    global rangeRes
   rangeRes = round(sld.Value, 1);

end
function updateFreqOffset(sldOffset)
    global freqOffsetFactor
    freqOffsetFactor = sldOffset.Value;
end
function updateDopplerSweeps(sldDoppler)
    global dopplerSweeps
    dopplerSweeps = round(sldDoppler.Value);
end
```

Bibliografie pentru I si II

Skolnik, M. I. – Introduction to Radar Systems, McGraw-Hill, 3rd Edition, 2001.
-Clasic în teoria radarului: formulele de rezoluție, efect Doppler, structura semnalelor FMCW.

Richards, M. A. – Fundamentals of Radar Signal Processing, McGraw-Hill, 2nd Edition, 2014.
-Excelent pentru partea de procesare semnal radar, Doppler FFT, sweepuri, interferență.

MathWorks Documentation – Radar Toolbox™ Documentation – MATLAB
-Exemple oficiale cu simulare radar FMCW, interferență, GUI-uri MATLAB, uiaxes, uilabel, uislider.

Wikipedia (pentru definiții și explicații simple):

-Frequency-modulated continuous-wave radar

-Radar signal processing

-Doppler effect

III.Backend al proiectului
(ce se intampla dupa ce dam run la simulare)

Inițializarea parametrilor de bază pentru radarul victimă

```
function start_project(mainFig)
%Define Victim Baseband FMCW Radar Waveform.
%For illustration purposes, in this example, configure the radar to a maximum range of 150 m. The victim
% radar operates at a 77 GHz frequency. The radar is required to resolve objects that are at least 1 meter
% apart. Because this is a forward-facing radar application, the maximum relative speed of vehicles on highway
% is specified as 230 km/h.

%Radarul Victima Colaterala

%FMCW-Frequency-Modulated Continuous Wave(modulare in frecventa unda-
%-continua.

rng(2023);             % cand folosesc functii random asta face ca functia mereu sa dea acelasi rezultat
fc = 77e9;    % frecventa centrala
c = physconst('LightSpeed');             % viteza luminii (m/s)
global lambda
lambda = freq2wavelen(fc,c);             % lambda/lungimea undei (m)
 global rangeMax
rangeMax = 150;        % distanta maxima la care poate radaru sa vada (m)
global rangeRes  % distanta minima la care obiectele trebuie sa fie separate ca radarul sa le rezolve (m)

vMax = 230*1000/3600;  % viteza maxima relativa a masinilor (m/s)
global fmcwwav1;
fmcwwav1 = helperFMCWWaveform(fc,rangeMax,rangeRes,vMax);% call la functia helperFMCWWaveform
global sig;
sig = fmcwwav1(); %stocheaza in sig
Ns = numel(sig);%cate sampleluri avem
```

Definirea semnalului radarului care interferează (al doilea radar FMCW)

```
%Define Interfering Baseband FMCW Radar Waveform

%Radarul care interfereaza

global freqOffsetFactor
fcRdr2 = 77e9 + (freqOffsetFactor - 1)*200e6;  % între 77.0 GHz și 77.2 GHz          % frecv centrala (Hz)
lambdaRdr2 = freq2wavelen(fcRdr2,c);              % lungimea de unda (m)
rangeMaxRdr2 = 100;     %  distanta maxima la care poate radaru sa vada(m)
rangeResRdr2 = 0.8;     % distanta minima la care obiectele trebuie sa fie separate ca radarul sa le rezolve (m)
vMaxRdr2 = vMax; % viteza maxima relativa a masinilor  (m/s)
global sigRdr2;
global fmcwwav2;

 fmcwwav2 = helperFMCWWaveform(fcRdr2,rangeMaxRdr2,rangeResRdr2,vMaxRdr2);% call la functia helperFMCWWaveform
sigRdr2 = fmcwwav2();%stocheaza in sig
```

Configurarea radarului victimă (transceiver MIMO)

```
% Modelul transceiverelor radar MIMO: victimă și interferent
% Se consideră că ambele radare utilizează un array liniar uniform (ULA) atât pentru transmisie,
% cât și pentru recepția undelor radar.

% Utilizarea unui array de recepție permite radarului să estimeze direcția azimutală a energiei
% reflectate de la ținte potențiale.

% Utilizarea unui array de transmisie permite radarului să formeze un array virtual mare la recepție,
% îmbunătățind astfel rezoluția unghiului azimutal.

% Pentru a recepționa semnalele de la țintă și zgomotul la radarul victimă, modelați transceiverul
% monostatic al radarului victimă folosind radarTransceiver și specificând array-urile ULA de transmisie
% și recepție ale radarului victimă în proprietățile sale.



% Setez parametrii de baza ai transceiverului radar
antAperture = 6.06e-4;                          % Deschiderea antenei (m^2)
antGain = aperture2gain(antAperture,lambda);    % Castigul antenei (dB)
txPkPower = db2pow(13)*1e-3;                    % Puterea de varf a emitătorului (W)
rxNF = 4.5;                                     % Figura de zgomot a receptorului (dB)

% Construiește array-ul de recepție pentru radarul victima
Nvr =  16;     % Numarul de elemente de recepție pentru radarul victima
global vrxEleSpacing
vrxEleSpacing = lambda/2;        % Distanța intre elementele de receptie pentru radarul victima
antElmnt = phased.IsotropicAntennaElement('BackBaffled',false); %Creează un element de antenă izotrop.
% O antenă izotropă este o antenă ideală care radiază în mod uniform în toate direcțiile. În realitate,
% astfel de antene nu există, dar ele sunt utile în modelarea teoretică a radiației antenei.
vrxArray = phased.ULA('Element',antElmnt,'NumElements',Nvr,...
    'ElementSpacing',vrxEleSpacing); % Creează un Uniform Linear Array (ULA), adică o aranjare liniară
% de antene. Un ULA este o configurație de antene în linie dreaptă, folosită frecvent în radare și sisteme
```

```
% de comunicații pentru a capta semnale din diferite direcții.

% Construiește array-ul de transmisie pentru radarul victimă
Nvt = 2;     % Numărul de elemente de transmisie pentru radarul victimă
vtxEleSpacing = Nvr*vrxEleSpacing;     % Distanța între elementele de transmisie pentru radarul victimă
vtxArray = phased.ULA('Element',antElmnt,'NumElements',Nvt,...
    'ElementSpacing',vtxEleSpacing);% Creează un Uniform Linear Array (ULA), adică o aranjare liniară de
% antene. Un ULA este o configurație de antene în linie dreaptă, folosită frecvent în radare și sisteme de
% comunicații pentru a capta semnale din diferite direcții.

% Modelează transceiverul monostatic al radarului victimă pentru recepționarea semnalelor de țintă și zgomot
vradar = radarTransceiver("Waveform",fmcwwav1,'ElectronicScanMode','Custom'); %Creează un obiect radarTransceiver
% care reprezintă un transceiver radar.  'ElectronicScanMode', Configurează modul de scanare electronică al radarului
% ca fiind personalizat, indicând că radarul poate să-și modifice directivitatea antenei pentru a urmări ținta.
vradar.Transmitter = phased.Transmitter('PeakPower',txPkPower,'Gain',antGain);%Creează obiectul Transmitter al radarului,
% care reprezintă emițătorul radarului.
vradar.TransmitAntenna = phased.Radiator('Sensor',vtxArray,'OperatingFrequency',fc,'WeightsInputPort',true);%Creează obiectul
% TransmitAntenna care reprezintă antena de transmisie a radarului. 'Sensor', Asociază antena de transmisie cu array-ul de
% antene de transmisie vtxArray, care a fost definit anterior. 'WeightsInputPort', true Permite aplicarea de cântare pentru
% elementele array-ului antenei, care sunt necesare pentru a direcționa semnalul într-o anumită direcție.
vradar.ReceiveAntenna = phased.Collector('Sensor',vrxArray,'OperatingFrequency',fc); % Creează obiectul ReceiveAntenna, care
% reprezintă antena de recepție a radarului. 'Sensor',Asociază antena de transmisie cu array-ul de antene de transmisie vtxArray,
% care a fost definit anterior.
vradar.Receiver = phased.ReceiverPreamp('Gain',antGain,'NoiseFigure',rxNF,'SampleRate',fmcwwav1.SampleRate);%Creează obiectul
% Receiver, care reprezintă receptorul radarului. 'NoiseFigure' Setează figura de zgomot a receptorului, care indică nivelul de
% zgomot pe care receptorul îl introduce în semnalul de recepție. rxNF este o valoare dată (în dB).
```

Configurarea radarului care interferează (transceiver)

```
%În diferit față de semnalele de țintă, interferența este transmisă de la ULA-ul de transmisie al radarului
% de interferență și recepționată de ULA-ul de recepție al radarului victimă. Modelează transceiverul de interferență
% folosind funcția helperInterferenceTransceiver, care include ULA-ul de transmisie al radarului de interferență și
% ULA-ul de recepție al radarului victimă ca intrări.


% Construiește array-ul de transmisie pentru radarul de interferență
Nit = 3;     % Numărul de elemente de transmisie pentru radarul de interferență
Nir = 4;     % Numărul de elemente de recepție pentru radarul de interferență
irxEleSpacing = lambda/2;        % Distanța între elementele de recepție pentru radarul de interferență
itxEleSpacing = Nir*irxEleSpacing;     % Distanța între elementele de transmisie pentru radarul de interferență
itxArray = phased.ULA('Element',antElmnt,'NumElements',Nit,...
    'ElementSpacing',itxEleSpacing);%Creează un Uniform Linear Array (ULA), adică o configurație de antene plasate
% pe o linie dreaptă. Un ULA este folosit pentru a direcționa semnalele de transmisie sau recepție într-o anumită direcție
% și pentru a colecta semnalele de la diverse unghiuri. În acest caz, ULA-ul va fi utilizat pentru transmisia semnalului
% de interferență. 'Element' Acesta specifică tipul de elemente de antenă folosite în array. antElmnt este un obiect definit
% anterior în cod (probabil un element de antenă izotropă sau o altă configurație de antenă).

% Modelează transceiverul de interferență pentru recepționarea semnalului de interferență la radarul victimă
iradar = helperInterferenceTransceiver(fmcwwav2,txPkPower,antGain,fc,itxArray,vrxArray);
```

Formule și concepte-cheie în spate

Lungimea undei:

$$\lambda = \frac{c}{f}$$

Unde:

- c = 3×10^8 m/s (viteza luminii)
- f = frecvența purtătoare (ex: 77 GHz)

Rezoluția radarului:

$$R_{res} = \frac{c}{2B}$$

Unde:

- B = banda de frecvență
- R_{res} = rezoluția în distanță

Puterea în Watt din dBm:

$$P(W) = 10^{\frac{P(dBm)-30}{10}}$$

Câștigul antenei:

$$G = \frac{4\pi A}{\lambda^2}$$

Crearea scenariului de conducere autonomă și inițializarea vehiculului ego

```
% Create driving scenario
global scenario
global egoCar
[scenario, egoCar] = helperAutoDrivingScenario;
```

Obținerea poziției și vitezei țintelor în raport cu radarul

```
% Inițializează pozițiile țintelor în cadrul de referință al vehiculului ego
tgtPoses = targetPoses(egoCar);

% Distanța și unghiul vehiculului țintă relativ la radarul victimă
[tarRange, tarAngle] = rangeangle(tgtPoses(1).Position');

% Viteza radială a vehiculului țintă relativ la radarul victimă (semnul negativ este din cauza axelor de
% referință diferite între radialspeed și drivingScenarioDesigner)
tarVelocity = -radialspeed(tgtPoses(1).Position', tgtPoses(1).Velocity');

% Distanța și unghiul vehiculului de interferență relativ la radarul victimă
[intRange, intAngle] = rangeangle(tgtPoses(2).Position');

% Viteza radială a vehiculului de interferență relativ la radarul victimă
intVelocity = -radialspeed(tgtPoses(2).Position', tgtPoses(2).Velocity');
```

Range (distanță): Distanța directă între radar și țintă.

Angle (unghi): Unghiul dintre axa radarului și direcția către țintă.

Viteză radială: Componenta vitezei țintei pe direcția de propagare a undei radarului.

Distanța (range):

$$R = \sqrt{(x_t - x_r)^2 + (y_t - y_r)^2}$$

Unde:

- x_t, y_t: coordonatele țintei
- x_r, y_r: coordonatele radarului

Viteză radială (proiecția vitezei pe linia de vizare):

$$v_r = \frac{\vec{v} \cdot \vec{r}}{|\vec{r}|}$$

- $\vec{v}$: vectorul vitezei țintei
- $\vec{r}$: vectorul poziției față de radar
- „$\cdot$" este produs scalar

Obținerea cuburilor de date cu și fără interferență

```
%% Obtain Interference-free and Interfered Data Cubes
% Definește eșantioanele de timp rapid și timp lent
Nft = round(fmcwwav1.SweepTime*fmcwwav1.SampleRate);  % Numărul de eșantioane de timp rapid pentru radarul victimă
global dopplerSweeps
    Nsweep = dopplerSweeps;                                    % Numărul de eșantioane de timp lent pentru radarul victimă
Nift = round(fmcwwav2.SweepTime*fmcwwav2.SampleRate); % Numărul de eșantioane de timp rapid pentru radarul de interferență
Nisweep = ceil(Nft*Nsweep/Nift);                      % Numărul de eșantioane de timp lent pentru radarul de interferență

% Generează codul TDM-MIMO pentru elementele antenei de interferență
wi = helperTDMMIMOEncoder(Nit, Nisweep);

% Asamblează trenul de pulsiuni de interferență la timpul de scenariul radarului de interferență
rxIntTrain = zeros(Nift*Nisweep, Nvr);

% Inițializează timpul de scenariul
time = scenario.SimulationTime;

% Inițializează profilele actorilor
actProf = actorProfiles(scenario);

% Obține trenul de pulsiuni de interferență la timpul de scenariul radarului de interferență
for l = 1:Nisweep
    % Generează toate drumurile către radarul victimă
    ipaths = helperGenerateIntPaths(tgtPoses, actProf, lambdaRdr2);

    % Obține semnalul primit doar cu interferență
    rxInt = iradar(ipaths, time, wi(:, l));
    rxIntTrain((l-1)*Nift+1:l*Nift, :) = rxInt;
```

```
    % Obține timpul curent al scenariului
    time = time + fmcwwav2.SweepTime;

    % Mută țintele înainte în timp pentru următorul sweep
    tgtPoses(1).Position = [tgtPoses(1).Position] + [tgtPoses(1).Velocity] * fmcwwav2.SweepTime;
    tgtPoses(2).Position = [tgtPoses(2).Position] + [tgtPoses(2).Velocity] * fmcwwav2.SweepTime;
end

% Trunchiază trenul de pulsiuni de interferență la lungimea trenului de pulsiuni de țintă
rxIntTrain = rxIntTrain(1:Nft*Nsweep, :);

% Asamblează trenul de pulsiuni de interferență la timpul de scenariul radarului victimă
rxIntvTrain = permute(reshape(rxIntTrain, Nft, Nsweep, Nvr), [1, 3, 2]);
```

Transceiver: Dispozitiv care trimite (transmite) și primește (recepționează) semnal radar.

FMCW Radar: Radar cu unde continue și frecvență modulată liniar.

Semnal chirp (transmis de radar):

$$s(t) = \cos\left(2\pi\left(f_0 t + \frac{B}{2T}t^2\right)\right)$$

- f_0: frecvență de pornire
- B: lățimea de bandă
- T: durata chirpului

Frecvența de bătăi (beat frequency) – după dechirpare:

$$f_b = \frac{2RB}{cT}$$

- R: distanța până la țintă
- c: viteza luminii
- T: durata chirpului
- B: lățimea de bandă

Dechirparea semnalelor (procesarea semnalului radar)

```
% Dechirpează semnalul țintei la radarul victimă pentru a obține cubul de date fără interferență,
% care este folosit pentru a trasa răspunsul țintei ca referință ideală de performanță.
% De asemenea, dechirpează semnalul combinat de țintă și interferență pentru a obține cubul de date cu interferență,
% care este folosit pentru a trasa efectele interferenței asupra detectării țintei.

% Generează codul TDM-MIMO pentru elementele de antenă ale radarului victimă
wv = helperTDMMIMOEncoder(Nvt, Nsweep);

% Asamblează cubul de date fără interferență
XcubeTgt = zeros(Nft, Nvr, Nsweep);

% Asamblează cubul de date cu interferență
Xcube = zeros(Nft, Nvr, Nsweep);

% Inițializează timpul de scenariul
time = scenario.SimulationTime;

% Inițializează pozițiile țintelor în cadrul de referință al vehiculului ego
tgtPoses = targetPoses(egoCar);

% Inițializează profilele actorilor
actProf = actorProfiles(scenario);

% Obține cubul de date la timpul de scenariul radarului victimă
for l = 1:Nsweep
    % Generează drumurile țintelor către radarul victimă
    vpaths = helperGenerateTgtPaths(tgtPoses, actProf, lambda);

    % Obține semnalul primit fără interferență
    rxVictim = vradar(vpaths, time, wv(:, l));

    % Dechirpează semnalul țintei fără interferență
    rxVsig = dechirp(rxVictim, sig);
```

```
    % Salvează sweep-ul în cubul de date
    XcubeTgt(:,:,l) = rxVsig;

    % Obține semnalul de interferență primit
    rxInt = rxIntvTrain(:,:,l);

    % Dechirpează semnalul cu interferență
    rx = dechirp(rxInt + rxVictim, sig);
    Xcube(:,:,l) = rx;

    % Obține timpul curent al scenariului
    time = time + fmcwwav1.SweepTime;

    % Mută țintele înainte în timp pentru următorul sweep
    tgtPoses(1).Position = [tgtPoses(1).Position] + [tgtPoses(1).Velocity] * fmcwwav1.SweepTime;
    tgtPoses(2).Position = [tgtPoses(2).Position] + [tgtPoses(2).Velocity] * fmcwwav1.SweepTime;
end
```

Dechirpare: Multiplicarea semnalului recepționat cu conjugatul semnalului transmis – folosită pentru a extrage frecvența de bătăi, care conține info despre distanță și viteză.

Interferență: Semnal nedorit de la alt radar, care distorsionează semnalul radarului victimă.

Dechirpare (proces matematic):

$$s_{\text{dechirp}}(t) = s_{\text{rx}}(t) \cdot s_{\text{tx}}^*(t)$$

- $s_{\text{rx}}(t)$: semnalul recepționat
- $s_{\text{tx}}^*(t)$: conjugatul complex al semnalului transmis

Distanța din frecvența de bătăi (din nou):

$$R = \frac{c \cdot f_b \cdot T}{2B}$$

Calcularea numărului de eșantioane pentru distanță (range samples)

```
%%
% Calculate number of range samples
global Nrange
Nrange = 2^nextpow2(Nft);

% Define range response

global rngresp

rngresp = phased.RangeResponse('RangeMethod','FFT', ...
    'SweepSlope',fmcwwav1.SweepBandwidth/fmcwwav1.SweepTime, ...
    'RangeFFTLengthSource','Property','RangeFFTLength',Nrange, ...
    'RangeWindow','Hann','SampleRate',fmcwwav1.SampleRate);

% Calculate the range response of interference-free data cube
XrngTgt = rngresp(XcubeTgt);
```

Range (distanță): măsoară cât de departe este o țintă față de radar.

FFT (Fast Fourier Transform): transformă semnalul din domeniul timpului în domeniul frecvenței → ajută la detectarea distanței.

Nrange: numărul de eșantioane pe axa de distanță în cubul radar.

Această secțiune determină dimensiunea optimă (putere a lui 2) pentru FFT, care se aplică pe axa de distanță. Este o practică uzuală pentru eficiență în procesarea semnalului radar.

$$N_{\text{range}} = 2^{\lceil \log_2(N_{\text{ft}}) \rceil}$$

Apoi se definește obiectul RangeResponse, care face FFT pentru a extrage distanțele pe baza pantei chirpului (frecvență vs. timp).

Formulă pentru panta chirpului:

$$\text{SweepSlope} = \frac{B}{T}$$

Unde:

- B = lățime de bandă,
- T = timp de sweep.

Decodificarea semnalului TDM-MIMO și pregătirea pentru FFT Doppler

```
%%
% Decode TDM-MIMO waveform
[XdecTgt,Nsweep] = helperTDMMIMODecoder(XrngTgt,wv);

% Number of Doppler samples
global Ndoppler
Ndoppler = 2^nextpow2(Nsweep);

% Size of virtual array
Nv = Nvr*Nvt;

% Doppler FFT with Hann window
global XrngdopTgt


XrngdopTgt = helperVelocityResp(XdecTgt,Nrange,Nv,Ndoppler);
```

TDM-MIMO este o tehnologie radar în care antenele emit alternativ (în timp). Pentru a putea procesa corect semnalele, acestea trebuie decodate și sortate corespunzător.

După decodare, se pregătește FFT-ul Doppler pentru a extrage informația despre viteza țintei.

Formulă importantă pentru viteza Doppler:

$$v_r = \frac{f_d \cdot \lambda}{2}$$

Ce înseamnă fiecare variabilă:

v_r – **viteza radială** a țintei (componenta vitezei pe direcția radarului)

f_d – **frecvența Doppler**, adică schimbarea frecvenței semnalului reflectat cauzată de mișcare

λ – **lungimea de undă** a semnalului radar:

$$\lambda = \frac{c}{f_c}$$

Calculul hărții Doppler (viteză radială)

```
%% dopler map

% Pulse repetition interval for TDM-MIMO radar
global tpri
 tpri = fmcwwav1.SweepTime*Nvt;
global vmaxunambg
% Maximum unambiguous velocity
vmaxunambg = lambda/4/tpri;
```

Această parte calculează intervalul dintre pulsuri (PRI) și viteza maximă care poate fi detectată fără aliasing (aliasingul apare când două viteze diferite par identice).

Formulă pentru PRI (pulse repetition interval):

$$T_{\text{PRI}} = T_{\text{sweep}} \cdot N_{vt}$$

T_{PRI} = **Pulse Repetition Interval**

➤ Reprezintă timpul total între două transmiteri consecutive de unde radar complete pentru un ciclu TDM-MIMO.

T_{sweep} = **timpul unui chirp** (sau „frecvență sweep")

➤ Adică timpul necesar pentru ca semnalul să urce de la frecvența de start la cea de sfârșit.

N_{vt} = **numărul de transmițători virtuali (Tx)** în arhitectura TDM-MIMO

➤ Pentru că fiecare transmițător transmite pe rând în TDM (time-division), timpul total PRI trebuie să includă toți transmițătorii.

Viteză maximă neambiguă (fără aliasing):

$$v_{\max} = \frac{\lambda}{4 \cdot T_{\text{PRI}}}$$

$v_{\max}$ – **viteza maximă** pe care o poate măsura radarul fără confuzii (aliasing)

λ – lungimea de undă (ca mai sus)

T_{PRI} – **Pulse Repetition Interval**, adică cât timp trece între două transmiteri consecutive

Calculul hărții de unghiuri (direcția țintei)

```
%% angle map

global XrngangdopTgt

global Nangle
global tarDopplerBin
% Number of angle samples
Nangle = 2^nextpow2(Nv);

% Angle FFT with Hann window
XrngangdopTgt = helperAngleResp(XrngdopTgt,Nrange,Nangle,Ndoppler);

% Target Doppler bin
tarDopplerBin = ceil(tarVelocity/lambda*2*tpri*Ndoppler+Ndoppler/2);
```

Această secțiune aplică FFT în domeniul spațial (rețeaua virtuală de antene) pentru a obține unghiul de sosire al semnalului.

Formula utilizată:

$$\text{DopplerBin} = \left\lceil \frac{v_{\text{target}}}{\lambda} \cdot 2 \cdot t_{\text{PRI}} \cdot N_{\text{doppler}} + \frac{N_{\text{doppler}}}{2} \right\rceil$$

v_{target} – viteza reală a țintei (calculată anterior)

λ – lungimea de undă

T_{PRI} – pulse repetition interval (deja definit mai sus)

N_{doppler} – numărul de puncte FFT pe axa Doppler (viteze)

$\frac{N_{\text{doppler}}}{2}$ – offsetul de la mijlocul spectrului FFT (deoarece FFT returnează viteze de la negative la pozitive)

Bibliografia pentru III

Wikipedia - Beamforming
Wikipedia - Short Time Fourier Transform
Wikipedia - Radar Signal Processing
Wikipedia - Pulse Repetition Frequency (PRF)
Wikipedia - Doppler Radar
MathWorks Example – FMCW Radar Simulation
-https://www.mathworks.com/help/radar/ug/fmcw-radar-signal-simulation.html

IV.Pagina secundara

Schimbarea ferestrei principale cu una nouă de tip meniu (GUI)

```
%%
    % Închide fereastra principală
    close(mainFig);

    % Deschide fereastra secundară
    secondFig = uifigure('Name', 'Meniu Principal', 'Position', [200 200 450 350]);

    % Buton 1
    btn1 = uibutton(secondFig, ...
        'Text', 'Radarul Victima', ...
        'Position', [125 230 100 40], ...
        'ButtonPushedFcn', @(btn,event) functie1());

    % Buton 2
    btn2 = uibutton(secondFig, ...
        'Text', 'Radarul care interfereaza', ...
        'Position', [125 180 150 40], ...
        'ButtonPushedFcn', @(btn,event) functie2());

    % Buton 3
    btn3 = uibutton(secondFig, ...
        'Text', 'Simulare Scenariu Trafic', ...
        'Position', [125 130 150 40], ...
        'ButtonPushedFcn', @(btn,event) functie3());

    % Buton 4
    btn4 = uibutton(secondFig, ...
        'Text', 'Doppler Map', ...
        'Position', [125 70 100 40], ...
        'ButtonPushedFcn', @(btn,event) functie4());
```

```
% Buton 5
    btn5 = uibutton(secondFig, ...
        'Text', 'Angle Map', ...
        'Position', [125 20 100 40], ...
        'ButtonPushedFcn', @(btn,event) functie5());
btn6 = uibutton(secondFig, ...
        'Text', 'INF:AmbeleRadare', ...
        'Position', [325 130 100 40], ...
        'ButtonPushedFcn', @(btn,event) functie6());
btn7 = uibutton(secondFig, ...
        'Text', 'INF:DrivingScenario', ...
        'Position', [325 70 100 40], ...
        'ButtonPushedFcn', @(btn,event) functie7());
btn8 = uibutton(secondFig, ...
        'Text', 'INF:AngleDoplerMap', ...
        'Position', [325 20 100 40], ...
        'ButtonPushedFcn', @(btn,event) functie8());
```

Închide fereastra principală și creează o nouă interfață grafică cu mai multe butoane. Acest meniu permite accesul la diferite funcții pentru radar și simulare trafic.

Afișează spectrograma pentru semnalul radarului victimă

```
function functie1()
global sig;
global fmcwwav1;
   figure %deschide o fereastra noua pentru a pune chestii pe ea
pspectrum(repmat(sig,3,1),fmcwwav1.SampleRate,'spectrogram', ...
    'Reassign',true,'FrequencyResolution',10e6)
axis([0 15 -80 80]); title('Victim Radar'); colorbar off
end
```

Se folosește funcția pspectrum pentru a analiza în frecvență semnalul radarului.
Reprezintă în spectrogramă distribuția frecvenței în timp.

Afișează spectrograma pentru radarul care produce interferență

```
function functie2()

global sigRdr2;
global fmcwwav1;
figure %deschide o fereastra noua pentru a pune chestii pe ea
pspectrum(repmat(sigRdr2,3,1),fmcwwav1.SampleRate,'spectrogram',...
    'Reassign',true,'FrequencyResolution',10e6,'MinThreshold',-13)
axis([0 15 -80 80]); title('Interfering Radar'); colorbar off;
end
```

Deschide simularea scenariului de trafic

```
function functie3()
   %%
%Simulate Driving Scenario

global scenario
global egoCar


%activare/afisare
drivingScenarioDesigner(scenario)
end
```

Afișează harta Doppler (Range-Doppler Map)

```
function functie4()
global tpri
global fmcwwav1
global rangeMax
global vmaxunambg
global Nrange
global Ndoppler
global lambda
global XrngdopTgt
global rngresp
% Plot range-Doppler map
helperRangeVelocityMapPlot(XrngdopTgt,rngresp.SweepSlope,...
    fmcwwav1.SampleRate,tpri,rangeMax,vmaxunambg,Nrange,Ndoppler,lambda);
end
```

Afișează harta de unghi (Range-Angle Map)

```
function functie5()
global tpri
global fmcwwav1
global rangeMax
global vmaxunambg
global Nrange
global Ndoppler
global lambda
global XrngdopTgt
global rngresp

global XrngangdopTgt
global vrxEleSpacing
global Nangle
global tarDopplerBin
% Plot range-angle map
helperRangeAngleMapPlot(XrngangdopTgt,rngresp.SweepSlope,...
    fmcwwav1.SampleRate,rangeMax,vrxEleSpacing,Nrange,Nangle,tarDopplerBin,lambda);
end
```

Deschide fișiere PDF cu informații

```
function functie6()
    % Calea către fișierul PDF
    filePath = 'ambeleradare.pdf';

    % Deschide fișierul PDF cu aplicația asociată (Adobe, browser etc.)
    open(filePath);
end
function functie7()
    % Calea către fișierul PDF
    filePath = 'drivingscenario.pdf';

    % Deschide fișierul PDF cu aplicația asociată (Adobe, browser etc.)
    open(filePath);
end
function functie8()
    % Calea către fișierul PDF
    filePath = 'angledoplermap.pdf';

    % Deschide fișierul PDF cu aplicația asociată (Adobe, browser etc.)
    open(filePath);
end
```

V.Explicarea interfetei si a proiectului

Ce e radarul FMCW?

FMCW înseamnă Frequency Modulated Continuous Wave, adică „undă continuă modulată în frecvență”.
În loc să trimită „pulse” scurte de semnal (ca unele radare militare), radarul FMCW trimite un semnal continuu care își schimbă frecvența în timp (crește sau scade – asta se numește „chirp”).

folosite la:

asistență la frânare

cruise control adaptiv

detectare obstacole, pietoni, alte mașini etc.

Ce e interferența între radare?

2 masini cu radare:

semnalele se pot amesteca.

Radarul tău poate primi semnalul emis de altă mașină

Ce efecte are interferența?

Când două sau mai multe radare FMCW emit simultan semnale:

Confuzie la procesarea semnalului – mașina ta poate detecta obstacole care nu există sau ignora unele reale.

Valori eronate pentru distanță sau viteză – pentru că semnalul de la altă mașină poate avea altă frecvență, fază etc.

Performanță mai slabă la funcții ca frânarea automată sau parcarea asistată.

Cum se combate interferența?

Frecvențe ușor diferite

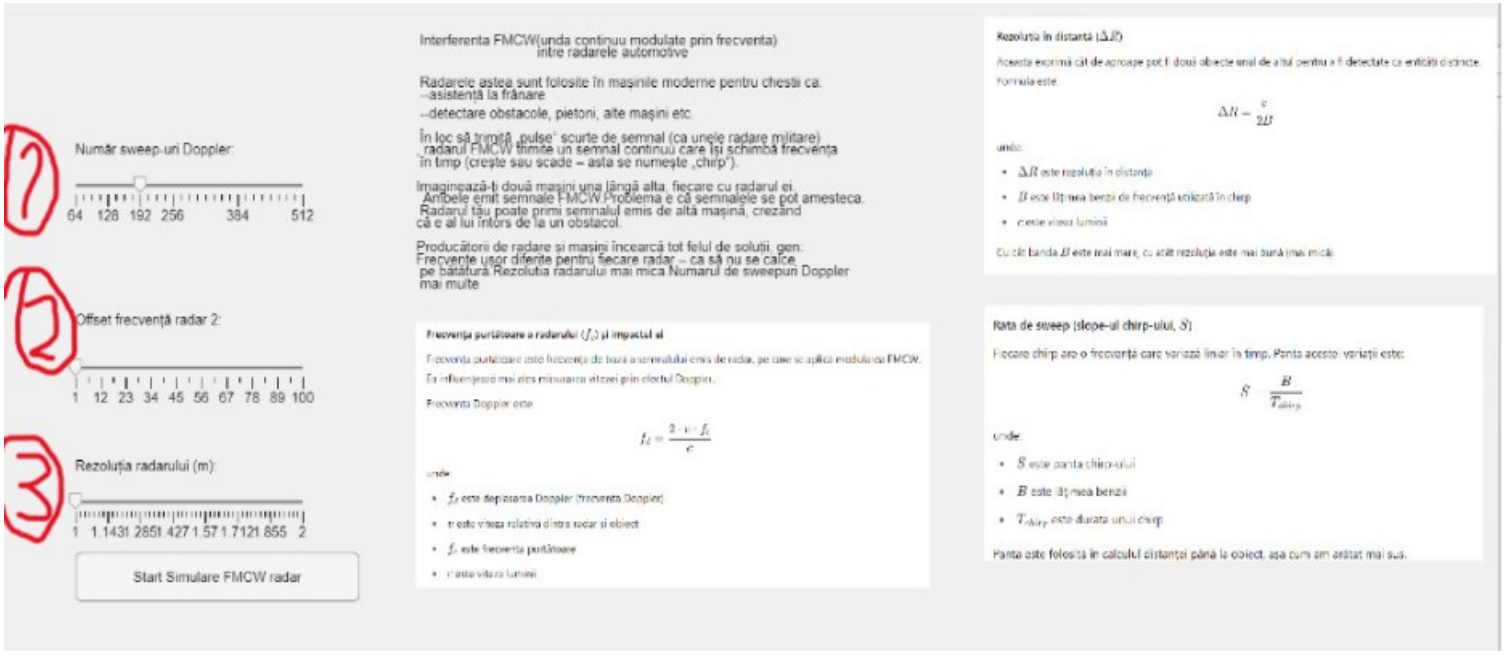
Numar de sweepuri mai dese

Rezolutia radar mai bune

Analogie(frecvente usor diferite)

E ca și cum ai avea două persoane în cameră vorbind în același timp, dar fiecare în altă limbă sau pe o frecvență ușor diferită. Tu încerci să-l asculți doar pe cel care te interesează (radarul tău), dar dacă celălalt vorbește tare, îți strică concentrarea (interferență).

Pagina principala



1.Numar sweepuri Doppler

Rata de sweep (slope-ul chirp-ului, S)

Fiecare chirp are o frecvență care variază liniar în timp. Panta acestei variații este:

$$S = \frac{B}{T_{chirp}}$$

unde:

- S este panta chirp-ului
- B este lățimea benzii
- T_{chirp} este durata unui chirp

Panta este folosită în calculul distanței până la obiect, așa cum am arătat mai sus.

Ce e „Doppler” în radar?

Efectul Doppler apare când un obiect se mișcă spre sau departe de tine.
Exemplu clasic: ai auzit cum sună o ambulanță diferit când se apropie față de când trece pe lângă tine și se îndepărtează? Asta e Doppler.

La radar, efectul Doppler înseamnă că frecvența semnalului reflectat se schimbă în funcție de viteza obiectului:

Dacă obiectul vine spre tine → frecvența crește.

Dacă se îndepărtează → frecvența scade.

Ce e un sweep?

Într-un radar FMCW, un sweep (sau „chirp”) e un semnal în care frecvența crește sau scade liniar într-un interval de timp.

Ce sunt sweep-urile Doppler?

Ca să afli viteza obiectului (cu efect Doppler), radarul:

Emite mai multe sweep-uri la rând (zeci sau sute).

Compară cum se schimbă frecvența semnalului reflectat între aceste sweep-uri.

Din diferențele mici de frecvență între chirp-uri → calculează viteza relativă a obiectului (Doppler shift).

Analogie

Gândește-te că emiți un „ping” cu o frecvență care urcă ușor →

Primesti semnalul înapoi. Dacă e shiftat față de ce ai trimis, știi că obiectul se mișcă.

Dacă faci asta de mai multe ori (mai multe sweep-uri), poți calcula nu doar distanța, dar și viteza.

2.Frecventa radarului si aplicatiile ei

Frecvența purtătoare a radarului (f_c) și impactul ei

Frecvența purtătoare este frecvența de bază a semnalului emis de radar, pe care se aplică modularea FMCW.
Ea influențează mai ales măsurarea vitezei prin efectul Doppler.

Frecvența Doppler este:

$$f_d = \frac{2 \cdot v \cdot f_c}{c}$$

unde:

- f_d este deplasarea Doppler (frecvența Doppler)
- v este viteza relativă dintre radar și obiect
- f_c este frecvența purtătoare
- c este viteza luminii

Formula ajutoatoare pentru 3

Determinarea distanței până la obiect (R)

Radarul FMCW măsoară distanța analizând diferența de frecvență dintre semnalul emis și cel reflectat.
Formula este:

$$R = \frac{c \cdot \Delta f}{2 \cdot S}$$

unde:

- R este distanța până la obiect
- c este viteza luminii în vid
- Δf este frecvența de bătaie (beat frequency) – diferența dintre frecvența semnalului transmis și cea a semnalului recepționat
- S este panta chirp-ului, adică rata de variație a frecvenței în timp

3.rezolutia in distanta a radarului (adica distanta intre obiecte)

Rezoluția în distanță (ΔR)

Aceasta exprimă cât de aproape pot fi două obiecte unul de altul pentru a fi detectate ca entități distincte.
Formula este:

$$\Delta R = \frac{c}{2B}$$

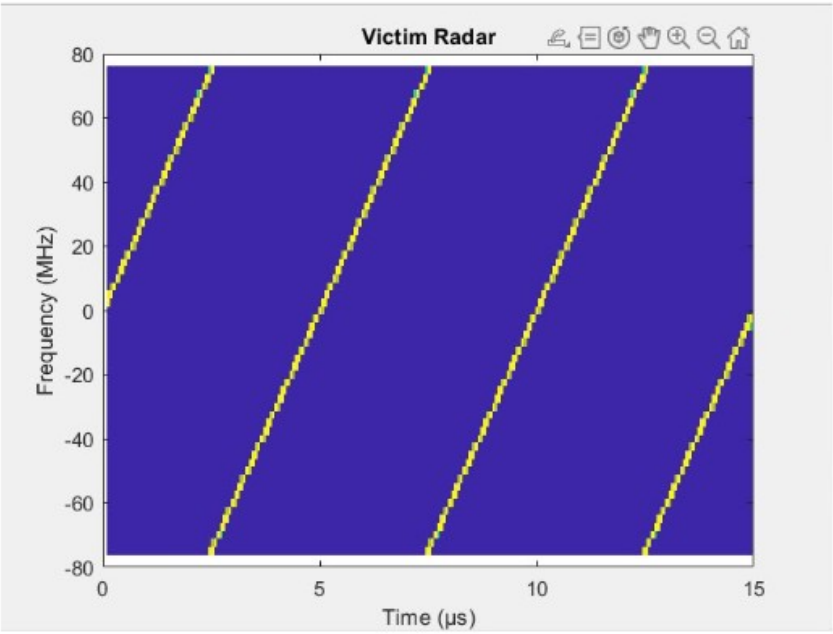
unde:

- ΔR este rezoluția în distanță
- B este lățimea benzii de frecvență utilizată în chirp
- c este viteza luminii

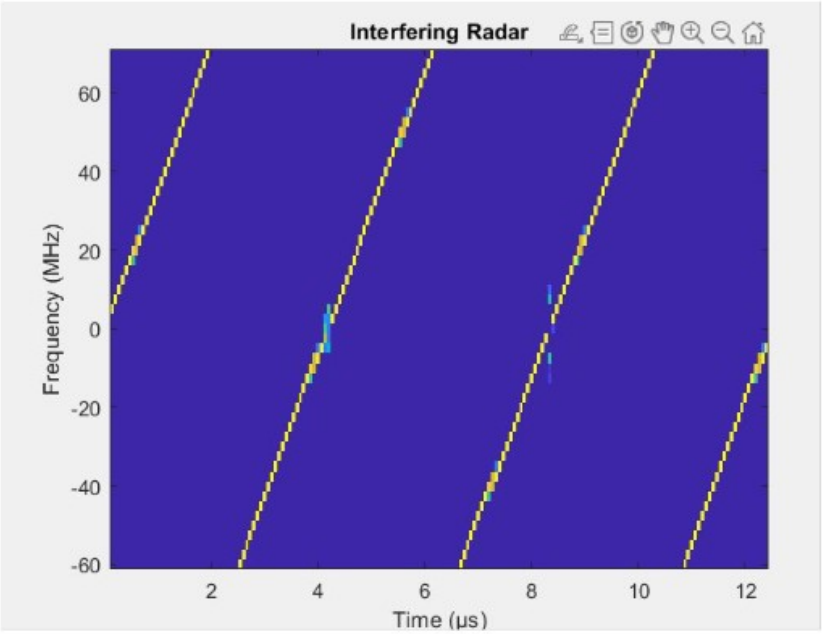
Cu cât banda B este mai mare, cu atât rezoluția este mai bună (mai mică).

Pagina secundara

Radarul victima



Radarul care interfereaza



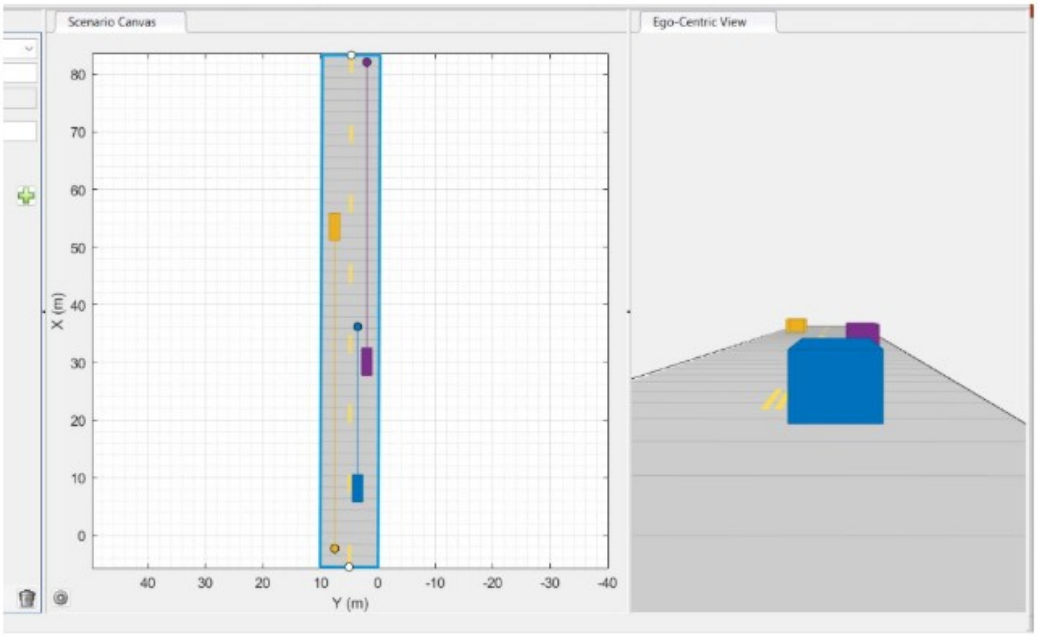
Despre Ambele Radare

A. Radarul victimă Este radarul pe care vrem să-l protejăm sau să-l studiem. În viața reală, poate fi radarul unei mașini care detectează alte vehicule din față. Acest radar trebuie să funcționeze bine, fără să fie deranjat de alte semnale.

B. Radarul interferent Este un alt radar din apropiere care poate trimite semnale ce deranjează radarul victimă. Acesta nu are intenția de a „ataca”, dar prin simpla sa funcționare poate crea probleme, cum ar fi semnale false sau pierderea țintelor.

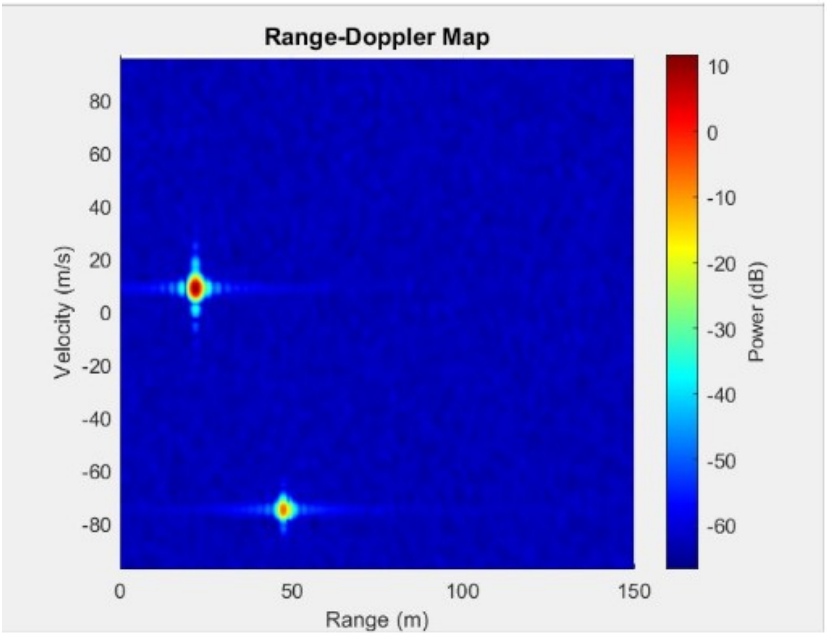
FMCW – Frequency Modulated Continuous Wave Este tipul de semnal folosit de majoritatea radarului modern. Are frecvență care crește și scade continuu într-un interval numit „chirp”. Acest tip de semnal permite măsurarea distanței și vitezei obiectelor. Rezoluția în distanță Este distanța minimă dintre două obiecte, astfel încât radarul să le vadă ca fiind două separate. O rezoluție mai mică (ex. 0.5 m) înseamnă mai multă precizie. Spectrograma Este o imagine care arată cum variază frecvențele unui semnal în timp. Se folosește pentru a analiza semnalul radar și pentru a observa cum se modifică în timpul emisiei. Reasignare (Reassignment) Este o metodă matematică folosită pentru a face spectrograma mai clară și mai precisă. Repetarea semnalului (repmat) În simulări, se repetă semnalul de radar de mai multe ori, ca și cum radarul ar emite mai multe pulsații reale, pentru a imita funcționarea continuă

Simularea Scenariului Trafic



Ambele radare (A și B) sunt integrate într-un scenariu de condus creat cu helperAutoDrivingScenario. Acest scenariu definește poziția, mișcarea și comportamentul vehiculelor pe o șosea virtuală. Prin această integrare: radarul victimă detectează ținte reale în trafic; radarul interferent transmite semnale care pot perturba detectarea; astfel se poate analiza performanța radarului în condiții reale de trafic și interferență

Doppler map



Ce este: Harta Range-Doppler este o reprezentare bidimensională care arată cum răspunde radarul în funcție de distanța (range) și viteza relativă (Doppler) a obiectelor detectate. Aceasta este folosită pentru a localiza și măsura viteza țintelor mobile.

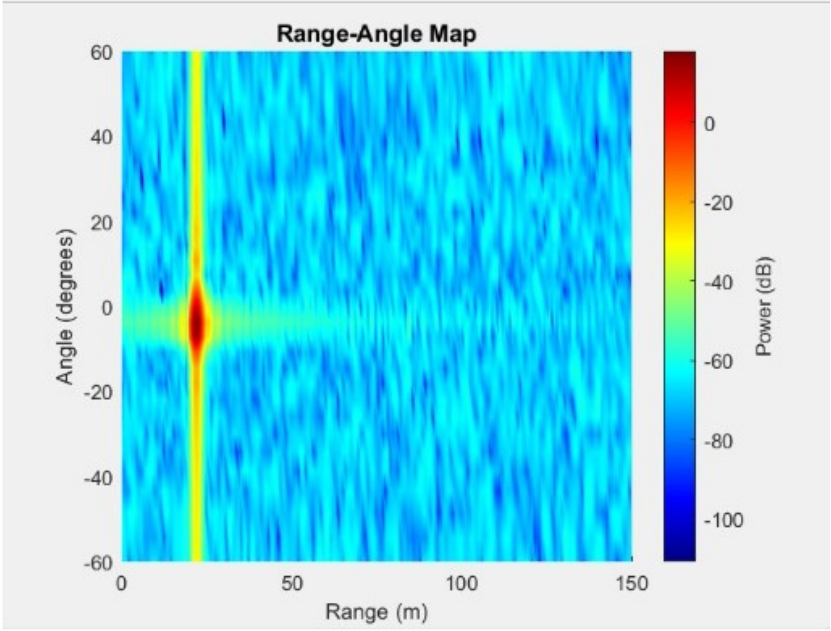
Cum se generează: Se aplică un FFT pe axa de distanță (range) – pentru fiecare sweep radar. Apoi, se aplică un FFT pe axa de timp lent (slow time) – adică între sweep-uri – pentru a detecta schimbări de fază cauzate de mișcare (viteza).

Rezultatul este o matrice complexă în care fiecare celulă indică puterea semnalului la o combinație distanță-viteză.

Ce conține: Axa X: viteza relativă (Doppler shift) Axa Y: distanța de la radar Valoarea pixelului: intensitatea semnalului reflectat

Cum e afectată: Interferența poate introduce semnale fantomă în zone greșite ale hărții (ex. viteză falsă). Zgomotul poate reduce contrastul, făcând țintele mai greu de detectat. Obiectele nemișcate (ziduri, sol) apar pe linia de viteză zero (Doppler = 0).
Ce sunt cercurile: Cercurile din Doppler map sunt artefacte vizuale cauzate de ferestrele de ponderare (ex. Hann) și de interferențe constructive/destructive în semnal. Acestea nu reprezintă obiecte reale, ci efecte de sidelobes sau aliasing spectral (efectul FFT asupra semnalului de la o țintă puternică)

Angle Map



Ce este: Harta Range-Angle reprezintă distribuția semnalelor radar în funcție de distanță și unghi de sosire (AOA – Angle of Arrival). Este folosită pentru a determina direcția în care se află o țintă față de radar.

Cum se generează: Se aplică FFT pe axa unghiulară (virtual array) – adică pe datele colectate de la mai multe antene (array). Această transformare convertește semnalul în funcție de direcția din care a venit. Pentru a obține unghiul, sistemul trebuie să fie MIMO sau să folosească o matrice de antene (virtual array), iar semnalele să fie coerente.

Ce conține: Axa X: unghiul de sosire (în grade sau radiani) Axa Y: distanța Valoarea pixelului: puterea semnalului reflectat de obiecte din acea direcție

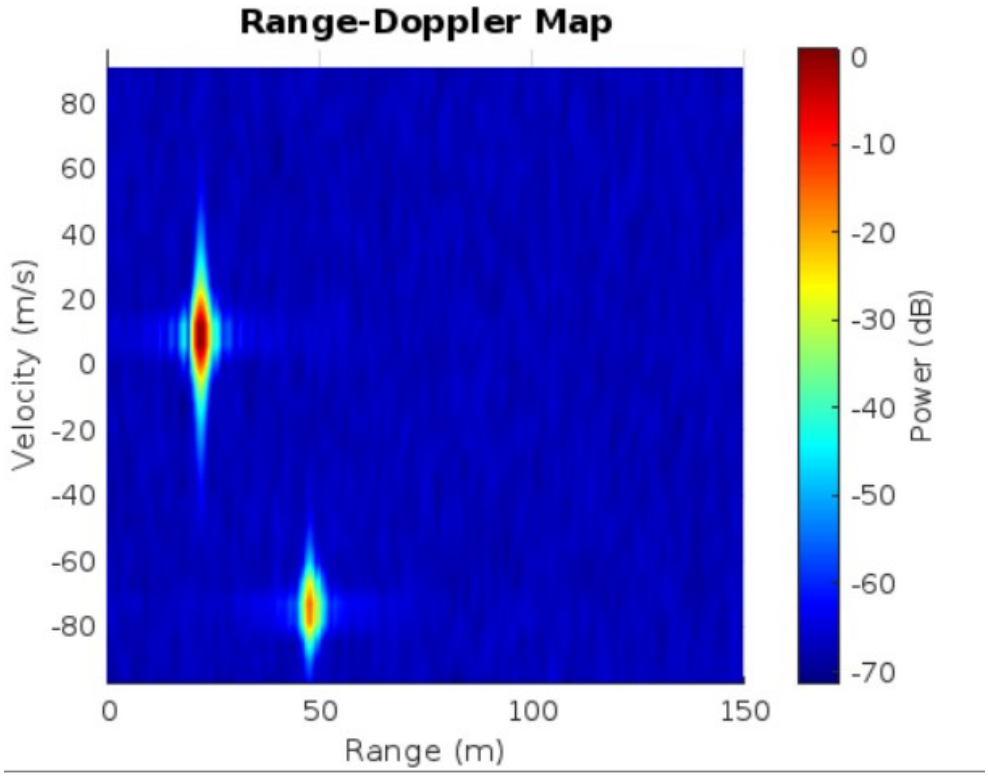
Cum e afectată: Interferența poate masca unghiul real al țintei, dând impresia că un obiect se află într-o altă direcție. Zgomotul poate reduce rezoluția unghiulară, făcând obiectele apropiate greu de separat. O antenă cu prea puține elemente sau spațiere necorespunzătoare degradează performanța. Ce sunt cercurile: Cercurile din harta unghiulară (range-angle) sunt lobi secundari sau artefacte de FFT și apar când o țintă puternică „scapă” energie în alte direcții din cauza rezoluției unghiulare limitate. Ele pot da impresia unor direcții multiple, dar doar una este corectă (vârful lobilor principali).

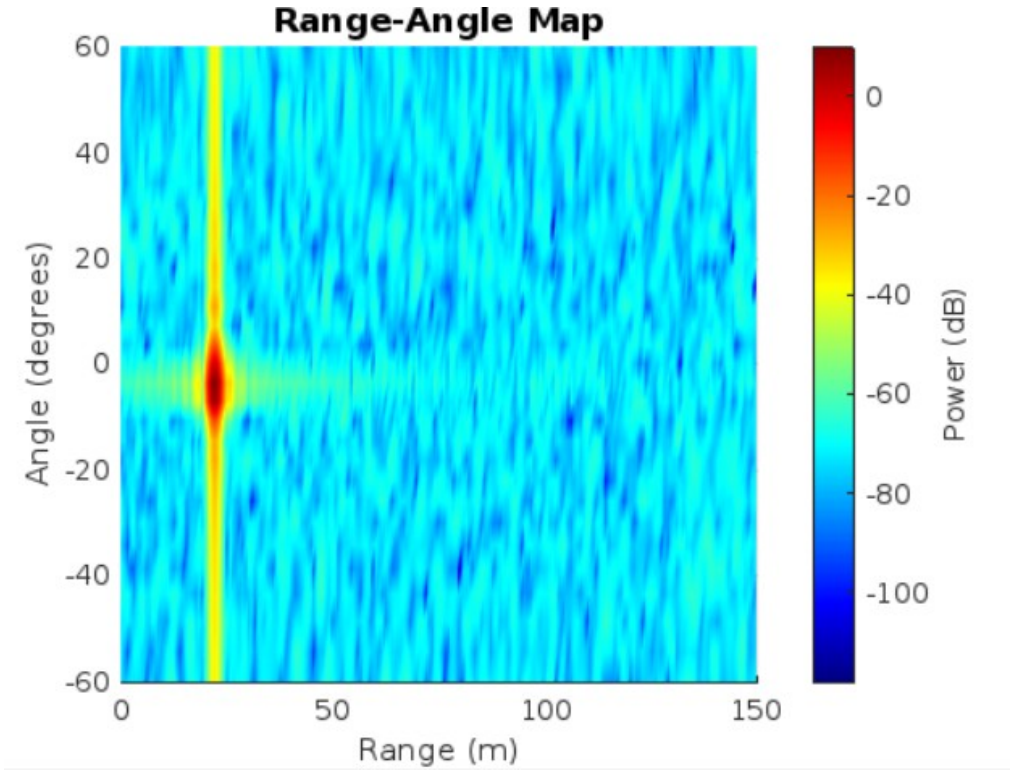
Hartă	Axa X	Axa Y	Ce vezi?	Ce sunt cercurile?
Doppler Map	Viteză (Doppler)	Distanță	Ținte mobile	Lobi laterali / interferență FFT
Angle Map	Unghi	Distanță	Direcția din care vine semnalul	Lobi secundari ai antenei virtuale

VI.Testare:

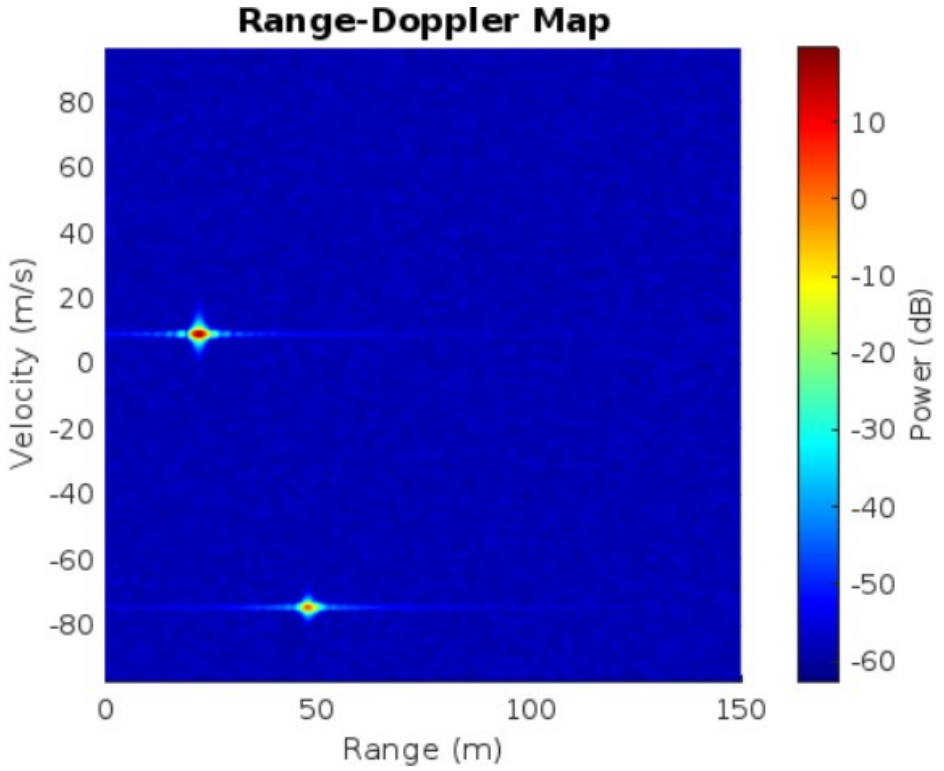
Schimbare a nr de sweepuri dopler

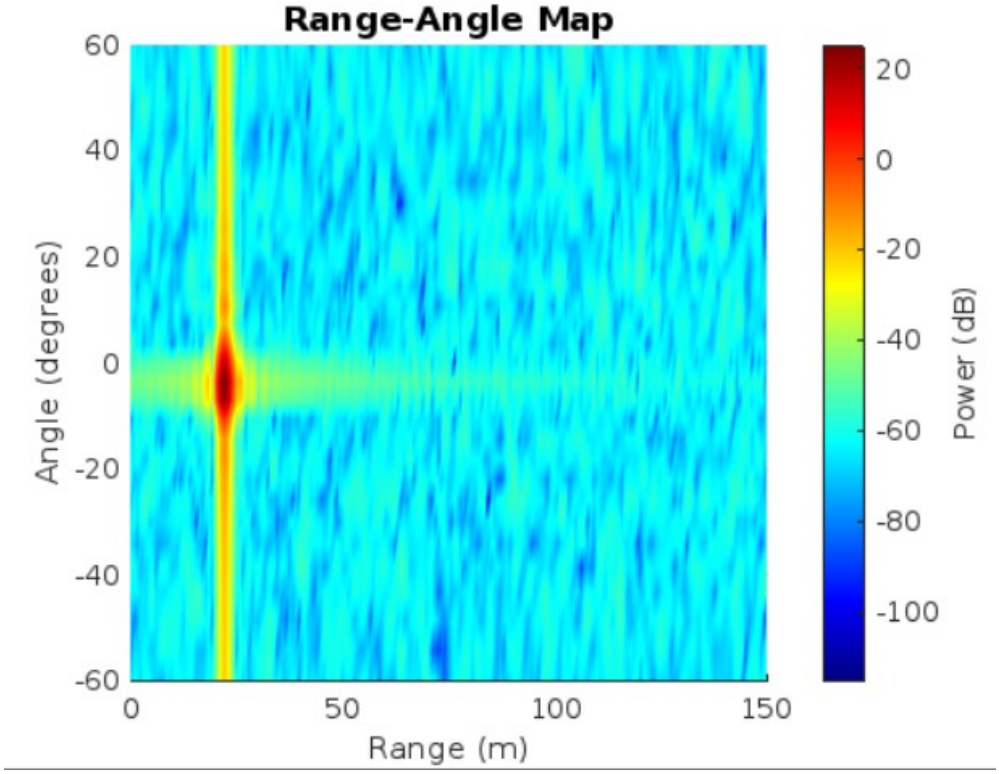
Nr sweepuri Dopler mici(64):





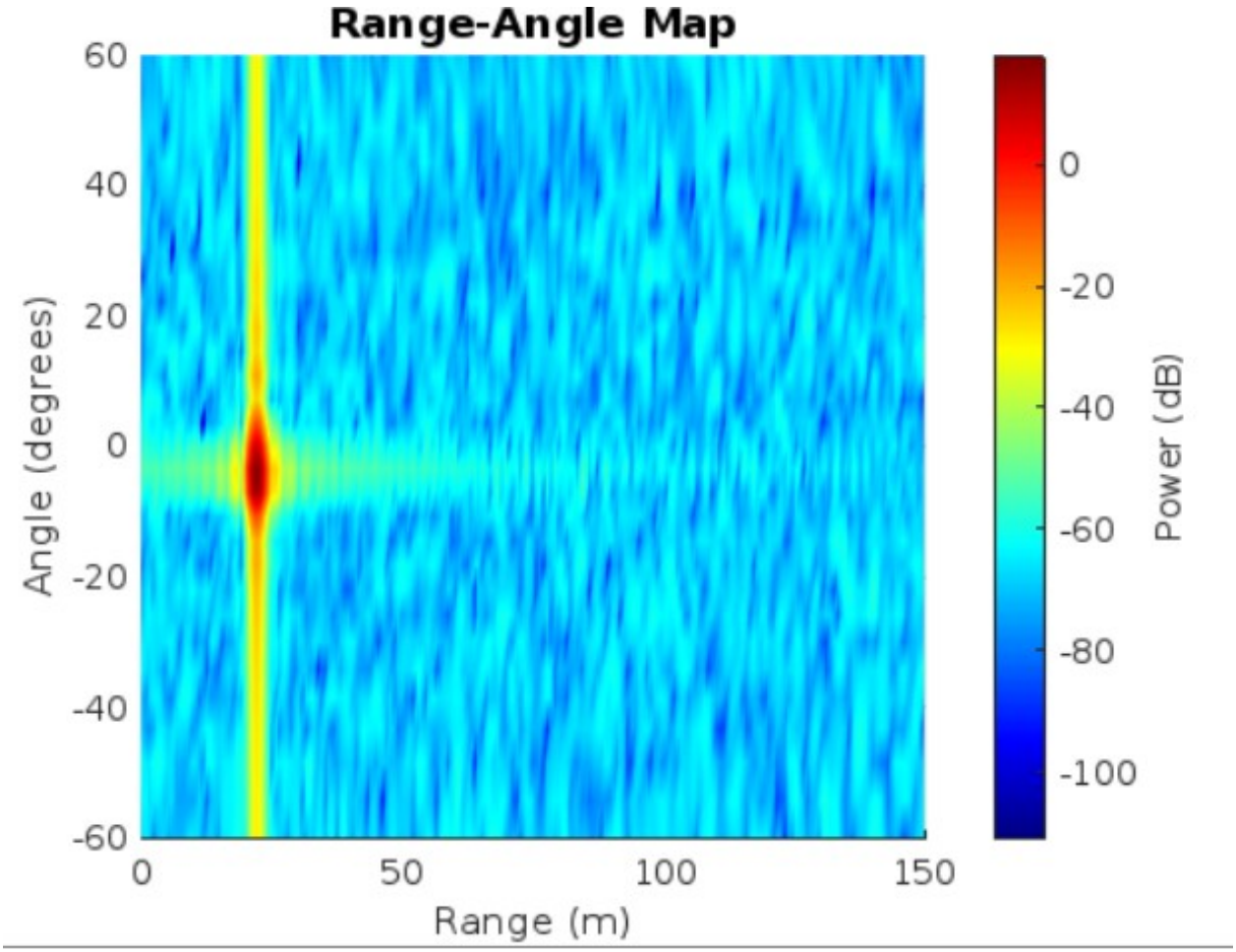
Nr sweepuri Dopler mari(512):

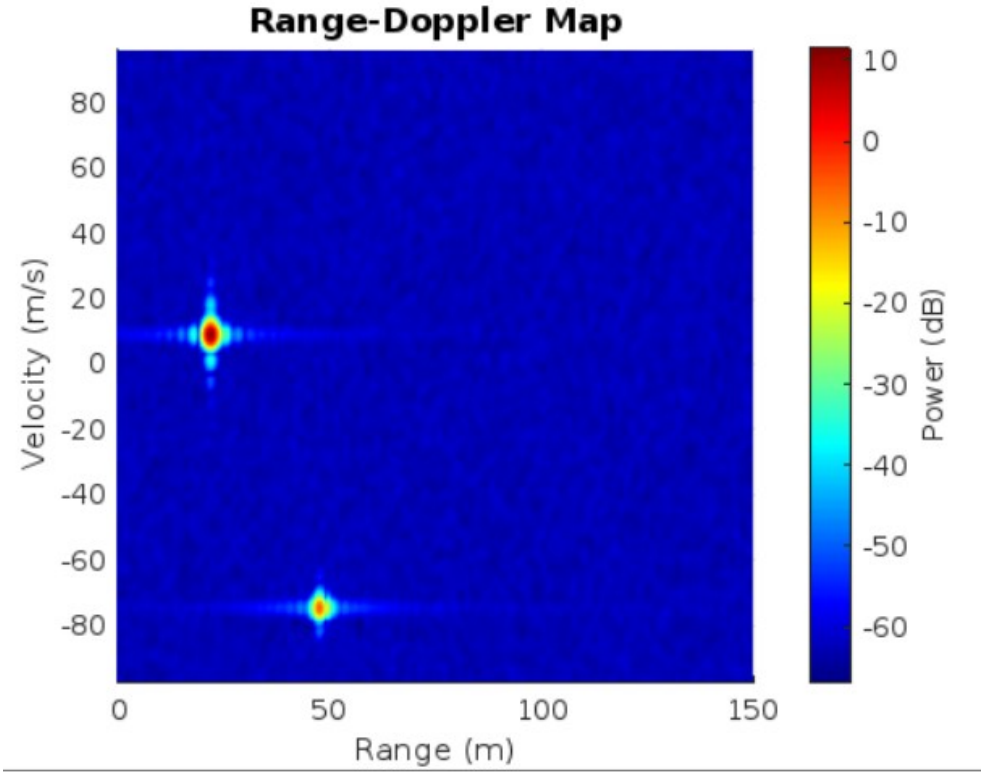




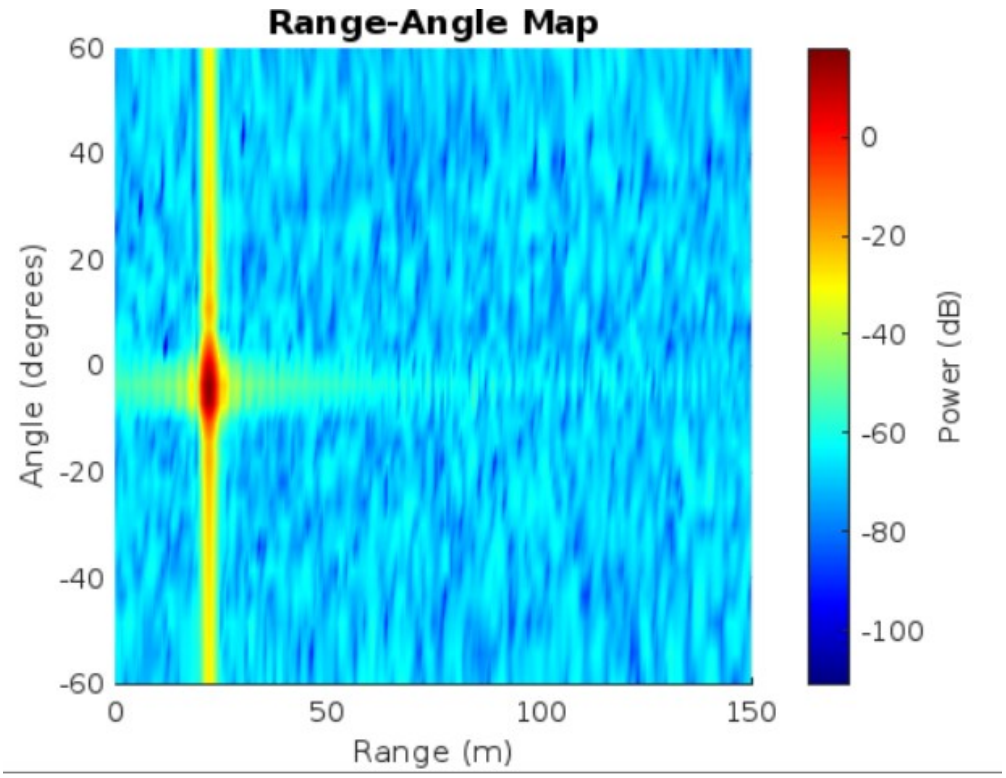
Schimbare f centrala a radar 2

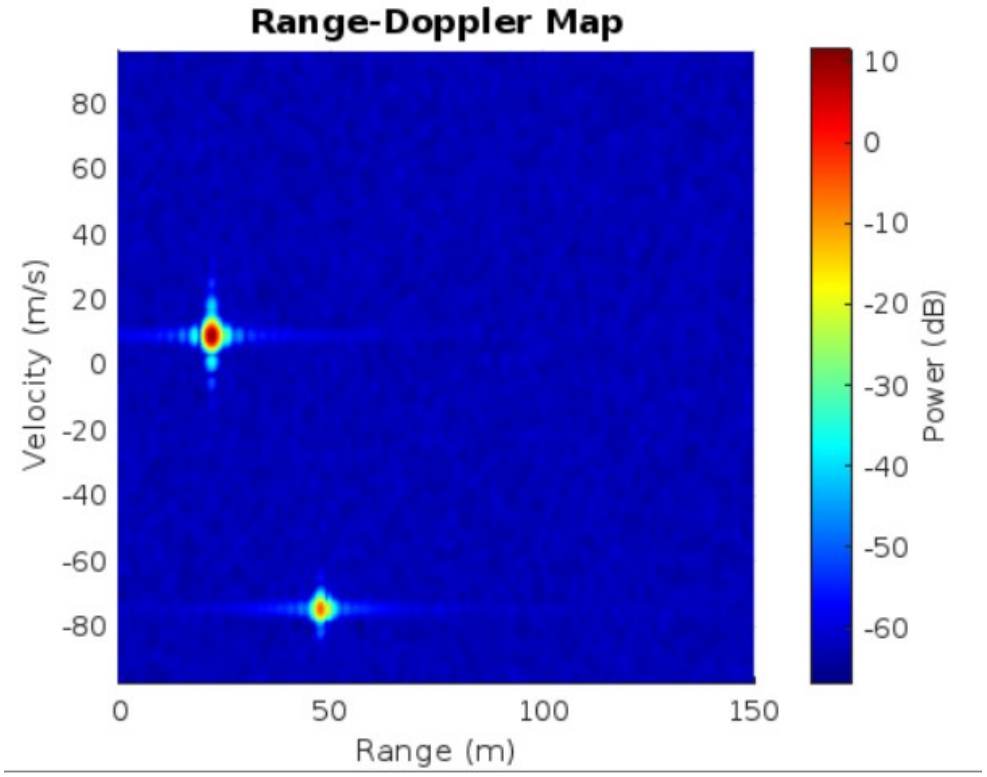
La fel





Fc2=2*fc1

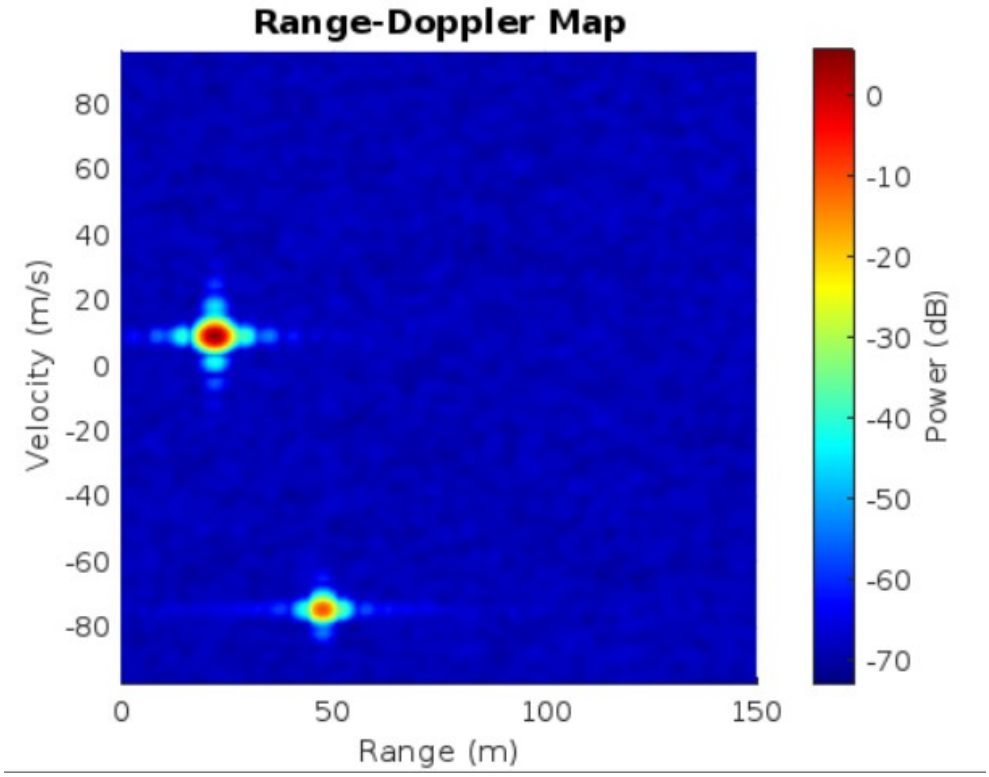


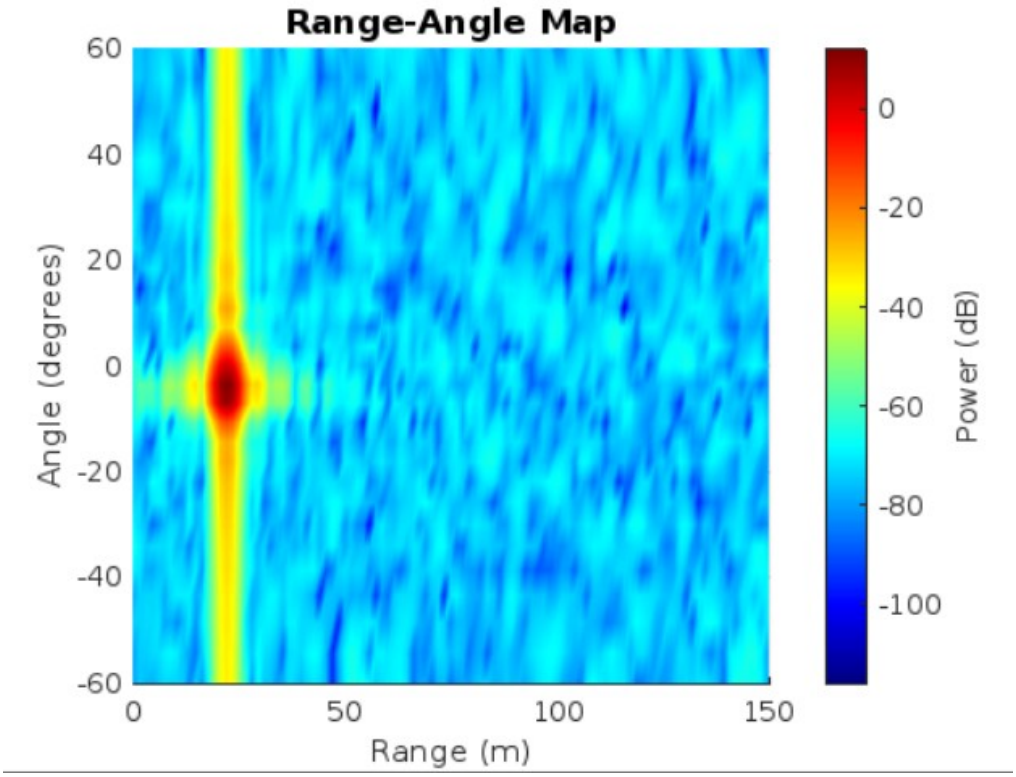


Am realizat dupa ce am modificat in cod ca la calcule nu se ia in considerare frecventele cele 2

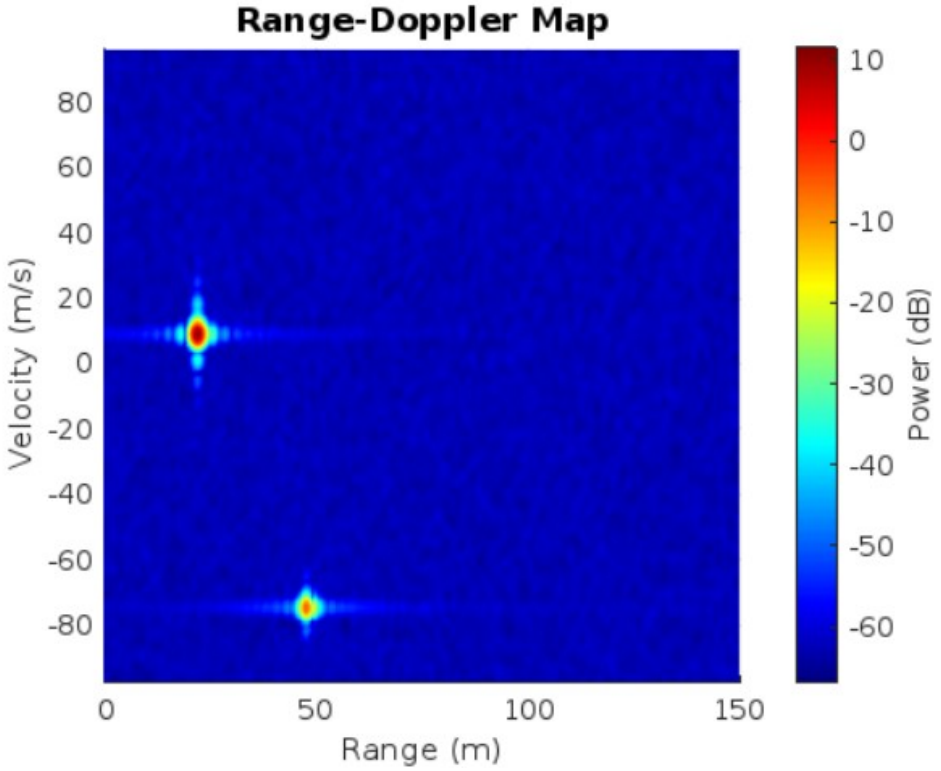
Schimbarea rezolutiei radarului

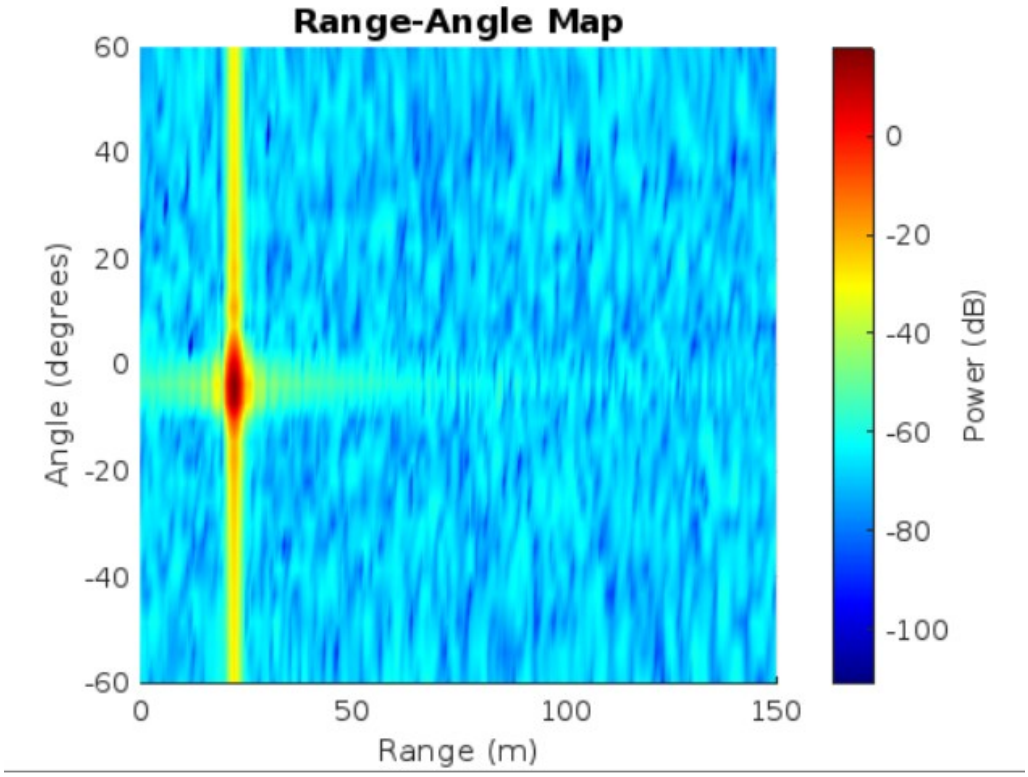
Rezolutie mare





Rezolutie mica





VII.Bibliografia a tot proiectului

Skolnik, M. I. – Introduction to Radar Systems, McGraw-Hill, 3rd Edition, 2001.
-Clasic în teoria radarului: formulele de rezoluție, efect Doppler, structura semnalelor FMCW.
Richards, M. A. – Fundamentals of Radar Signal Processing, McGraw-Hill, 2nd Edition, 2014.
-Excelent pentru partea de procesare semnal radar, Doppler FFT, sweepuri, interferență.
MathWorks Documentation – Radar Toolbox™ Documentation – MATLAB
-Exemple oficiale cu simulare radar FMCW, interferență, GUI-uri MATLAB, uiaxes, uilabel, uislider.
Wikipedia (pentru definiții și explicații simple):
-Frequency-modulated continuous-wave radar
-Radar signal processing
-Doppler effect
Wikipedia - Beamforming
Wikipedia - Short Time Fourier Transform
Wikipedia - Radar Signal Processing
Wikipedia - Pulse Repetition Frequency (PRF)
Wikipedia - Doppler Radar
MathWorks Example – FMCW Radar Simulation
-https://www.mathworks.com/help/radar/ug/fmcw-radar-signal-simulation.html