

SFND4- Radar Target Generation and Detection

The submission	1
FMCW Waveform Design	1
Simulation Loop	1
Range FFT (1st FFT)	2
2D CFAR	3

The submission

The submission contains the following file + the matlab script.

FMCW Waveform Design

In below snapshot, I provide initialization parameters for the radar. Slope value= $2.045e15$

```

r - C:\Users\azzouz\EMEA\work\projects\SFND_Radar_Target_Generation_Tracking\radar-target-generation-and-detect...
Clutter_CFAR_AoA_m x Clustering_Tracking.m x radar-target-generation-and-detection.m x +
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
max_range=200;
resolution=1;
max_velocity=70;
c=3*10e8;

%% User Defined Range and Velocity of target
% *%TODO* :
% define the target's initial position and velocity. Note : Velocity
% remains constant
R=110;
v=-20;

%% FMCW Waveform Generation
% *%TODO* :
%Design the FMCW waveform by giving the specs of each of its parameters.
% Calculate the Bandwidth (B), Chirp Time (Tchirp) and Slope (slope) of the FMC
% chirp using the requirements above.
B=c/(2*resolution);
time_max_range=2*max_range/c;
Tchirp=5.5*time_max_range;
slope=B/Tchirp
  
```

Command Window

```

slope =
    2.0455e+15

fx >>
  
```

Simulation Loop

To generate the mix signal, we take the following steps:

- Initialize timestep + compute position according to constant speed model.
- Compute round trip time τ
- Compute the emitted and received signals Tx & Rx
- Mix the signals

The steps are highlighted in red color in below screenshot.

```

ditor - C:\Users\azzouz\work\projects\SFND_Radar_Target_Generation_Tracking\radar-target-generation-and-detect
Clutter_CFAR_AoA_m x Clustering_Tracking.m x radar-target-generation-and-detection.m x +

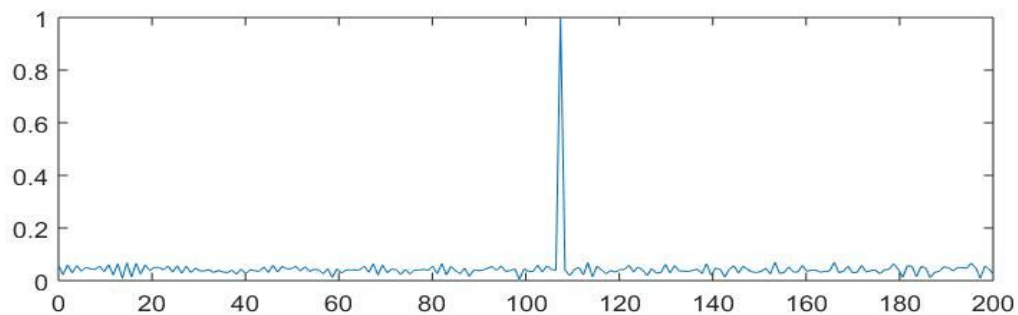
%% Signal generation and Moving Target simulation
% Running the radar scenario over the time.

for i=1:L
    % *%TODO* :
    %For each time stamp update the Range of the Target for constant velocity
    time=t(i);
    Rt=R+v*time;
    % *%TODO* :
    %For each time sample we need update the transmitted and
    %received signal
    Tx(i) = cos(2*pi*(fc*time+slope*(time^2)/2));
    tau=2*Rt/c;
    Rx(i) = cos(2*pi*(fc*(time-tau)+slope*((time-tau)^2)/2));
    % *%TODO* :
    %Now by mixing the Transmit and Receive generate the beat signal
    %This is done by element wise matrix multiplication of Transmit and
    %Receiver Signal
end
Mix = Tx.*Rx;

```

Range FFT (1st FFT)

We apply the fft on the first column of the **reshaped** and **mixed** signal. After normalization, taking absolute value and truncating first half (OK because sinal is real), we obtain below curve.



2D CFAR

First, I introduce the approach used to compute the output signal:

Train								
		Guard						
				Target				

I used the following values for Tr, Td, Gr, Gd and offset:

```
%Select the number of Training Cells in both the dimensions.
Tr=8; Td=10;
% *%TODO* :
%Select the number of Guard Cells in both dimensions around the Cell under
%test (CUT) for accurate estimation
Gr=4; Gd=4;
guardKernel=zeros(2*Gr+1,2*Gd+1);
trainKernel=padarray(guardKernel,[Tr Td],1);

% *%TODO* :
% offset the threshold by SNR value in dB
offset=10;
```

In order to compute the mean of the noise cells on the **db2pow(RBM)**, we're actually performing a convolution between **db2pow(RBM)** and a kernel of size $(2 \times Td + 2 \times Gd + 1) \times (2 \times Tr + 2 \times Gr + 1)$ where all green cells are equal to **1** and the others to **0**. For this purpose, I have used the **conv2** matlab function.

For this purpose I went through the following steps:

1. Build noise matrix

- Convolution with kernel composed of 1 for training cells and zero elsewhere
- Normalize the obtained matrix by the kernel sum to get the means
- Convert to db using **pow2db**
- Add offset

2. Build resulting matrix

- If $RDM < noise:0$, else 1

Below screenshots show the 2d range fft then the final result

