PROJECT REPORT ON MTI RADAR SYSTEM

Submitted To: Dr. Satyam Agarwal

Submitted By:

Ravishankar Kumar (2022EEM1007)

Mohd. Ubaid Wani (2022EEM1008)

Shandipan Paul (2022EEM1011)

Project Title: Design and simulation of a Moving Target Indicator Radar using MATLAB.

Objectives:

1. Identifying moving targets.
2. Clutter attenuation by using delay line canceler.
3. Find the effects of blind speed.
4. Reducing the effects of blind speed.

Procedure Followed:

- **Algorithm used in this project:**

MTI Radar:

MTI Radar (Moving Target Indicator Radar) is a type of radar system used to detect and track moving targets, such as vehicles, aircraft, or ships. Unlike other radar systems that can detect both stationary and moving targets, MTI radar is specifically designed to filter out signals from stationary objects and focus only on signals from moving targets. MTI radar works by transmitting a series of short radar pulses and then analysing the reflected signals. The radar system compares the frequency of the returned signal to the frequency of the original transmitted signal. If there is a difference in frequency, it means that the target is moving. The MTI radar system then tracks the moving target by continuously transmitting and receiving radar pulses and analysing the changes in frequency of the reflected signals.

Ground Clutter:

Ground clutter refers to the unwanted signals that are detected by a radar system when it reflects off objects on or near the ground, such as buildings, trees, hills, and other structures. These unwanted signals can interfere with the radar's ability to detect and track targets, especially at low altitudes and close range.

Delay Line canceller:

To remove these unwanted components from the received signal, a delay line canceller uses a series of delay lines and subtractors. The received signal is split into two paths, one of which is delayed by a certain amount of time using a delay line. The delayed signal is then subtracted from the original signal using a subtractor, which cancels out any components that are common to both signals.

Blind Speed:

Blind speeds in MTI (Moving Target Indicator) radar refer to the velocities of moving objects that are not detectable by the radar system. Blind speeds occur when the frequency of the radar signal is equal to the Doppler frequency of a moving object, resulting in the radar signal being cancelled out and no detection of the object. MTI radar works by comparing the frequency of the transmitted radar signal to the frequency of the received radar signal. If the frequency of the received signal is different from the frequency of the transmitted signal, it indicates that the target is moving. However, when the frequency difference between the transmitted and received signals is equal to the Doppler frequency of a moving object, the radar signal is cancelled out and no detection of the object occurs. This is known as a blind speed.

Concept of Staggered Pulse Repetition Frequency:

By transmitting pulses at different PRFs, the radar system can distinguish between targets at different ranges and velocities. The staggered PRF technique also reduces the probability of range ambiguities and blind speeds occurring, which can improve the accuracy and reliability of the radar system. In addition to mitigating range ambiguities and blind speeds, staggered PRF can also be used to improve the signal to noise ratio of the radar system by reducing the effects of clutter and interference. This is because the staggered PRFs cause the returned signals to be spread out in frequency, which makes it easier to filter out unwanted signals and extract the desired signals. Overall, staggered PRF is a powerful technique for improving the performance of radar systems, particularly in challenging environments where range ambiguities, blind speeds, and interference can be problematic.

Till here we have seen the concepts that we have to use in this project. Now we will see the methodology that we will follow further in our project.

- **Methodology with results**

Construct a radar system:

In this project basically we are working at the processing part of MTI radar system so we are here we are not designing a radar system from scratch. So, for radar designing part we are using a simple example of monostatic radar that is available on MATLAB. We have directly accessed that example for our MTI processing part by using the Load function of MATLAB. Some main predefined data that monostatic Radar system uses:

Probability of detection=0.9, Probability of false alarm=10^-6, max range=5km, range resolution=50, target radar cross section=1msq, antenna freq range=5 to 15GHz operating at 10ghz, Isotropic antenna,Tx power 5.22kw (by using integration of 10 pulses), tx gain 20dB.

Define Targets:

First, we are going to calculate the wavelength of the radar signal based on the wave speed and carrier frequency. Then we are calculating the maximum speed of a moving target that can be detected by the radar, based on the pulse repetition frequency (PRF), wavelength, and assuming a round trip distance. Gradually we are defining the position and velocity of the target, respectively, using a two-dimensional array (matrix) where each column represents a different target. After that we are defining radar cross section (RCS) of the target and the operating frequency of the radar, respectively.

Define Clutter:

We are generating first a surface gamma function, which is used to model surface reflections in the clutter. Then we are going to define constant gamma clutter object with several input parameters, including the antenna, propagation speed, operating frequency, sample rate, pulse repetition frequency (PRF), gamma function, sensor height, platform speed and direction, mounting angles, maximum clutter range, clutter azimuth span, patch azimuth span, and random seed for repeatability. Object generates a constant clutter power level with a spatial distribution that is determined by the gamma function and the specified clutter parameters.

Simulating Received Pulses using Matched Filter:

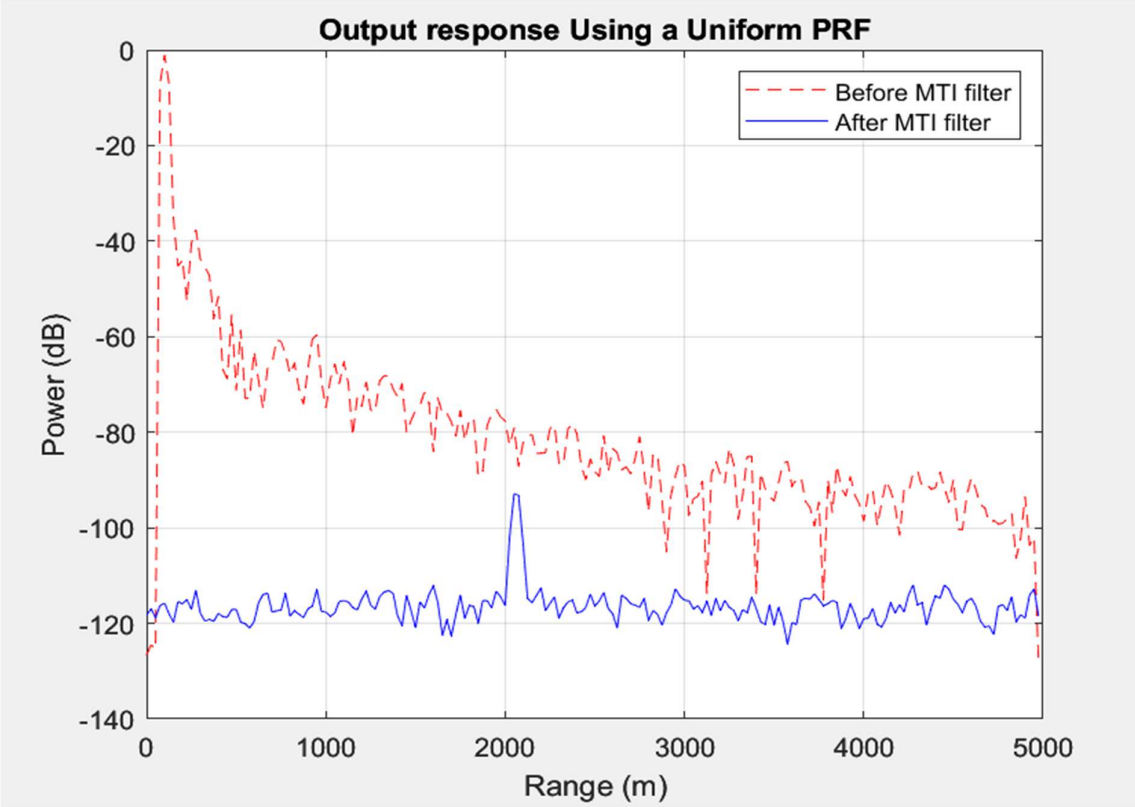
Here we are simulating 10 received pulses for the radar and targets defined earlier. We are setting the random seed of the receiver object to ensure that the simulation results can be reproduced."helperMTISimulate" function, which is a custom MATLAB function used to simulate the radar measurements. The function takes in several input parameters, including the waveform, transmitter, receiver, radiator, collector, sensor motion, target, target motion, clutter, and number of pulses. The output of the function is the received radar pulses (rxPulse),

which contains information on the signal strength, range, and velocity of the detected targets. We are passing the received signal through a passed filter.

Perform MTI Processing Using a Delay line Canceller:

Here we have used MTI filter (delay line canceller) in order to remove low frequency components like clutter. As you can see in the below graph the red curve is the output response at receiver without MTI filter. Thus, clutter will appear at low frequency as a target ambiguity for us and due to this we have missed our actual target detection. And after using MTI filter the response is shown by blue curve where you can see the clutter echoes have been removed.

And now we can see one peak in the blue curve that will tell that there is one target present. But in actual we have to detect two targets, so we are not detecting second object? the answer is due to the blind speed caused by using of delay line canceller.



for better understanding of blind speed problem, we will see the frequency response of delay line canceller. The null points are denoting the blind speed where we can not detect the signal.

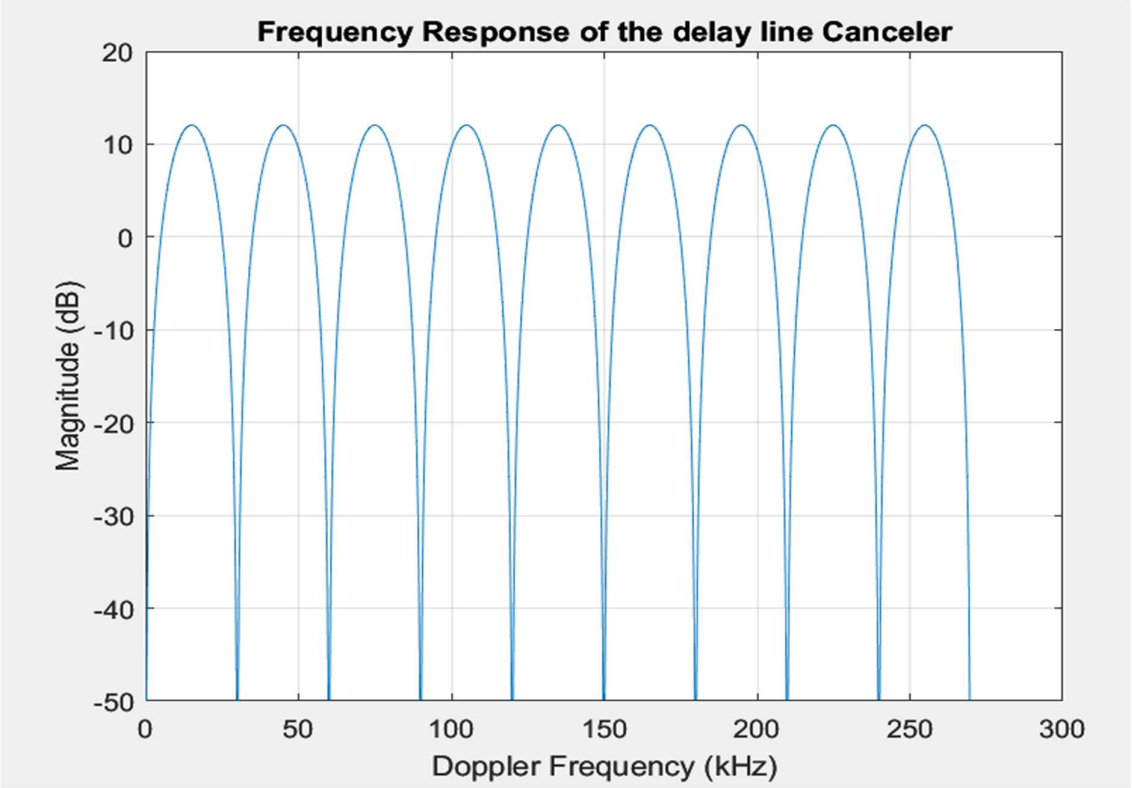
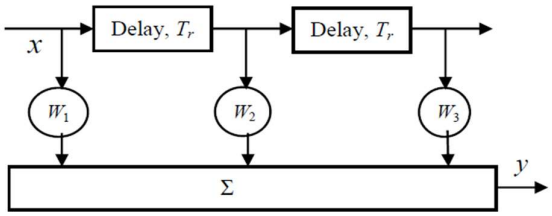


Figure- Frequency response of delay line canceller using single prf

- To reduce the effect of blind speed we have used the staggered pulse repetition frequency concept. In the below graph you can see the frequency response of the delay line canceller with staggered pulse repetition frequency.
- From the below graph you can observe that the blind speed has become almost five time to the previous one due to uses of staggered pulse repetition frequency.



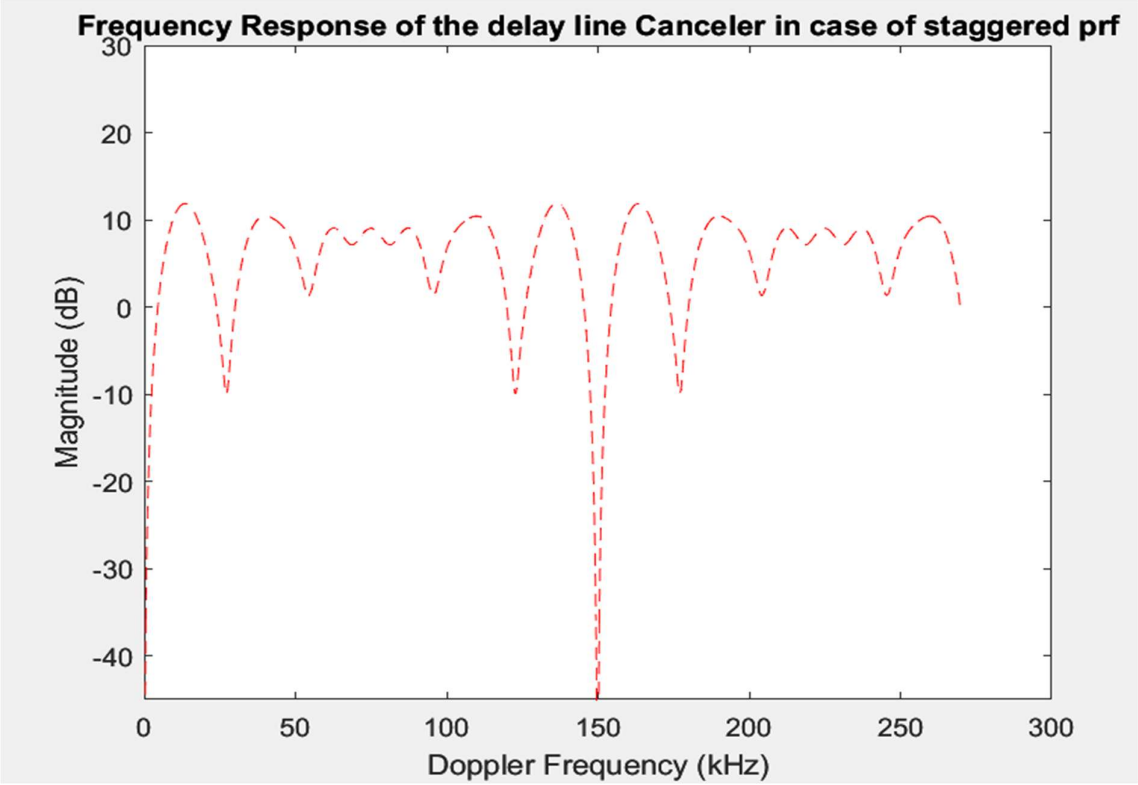
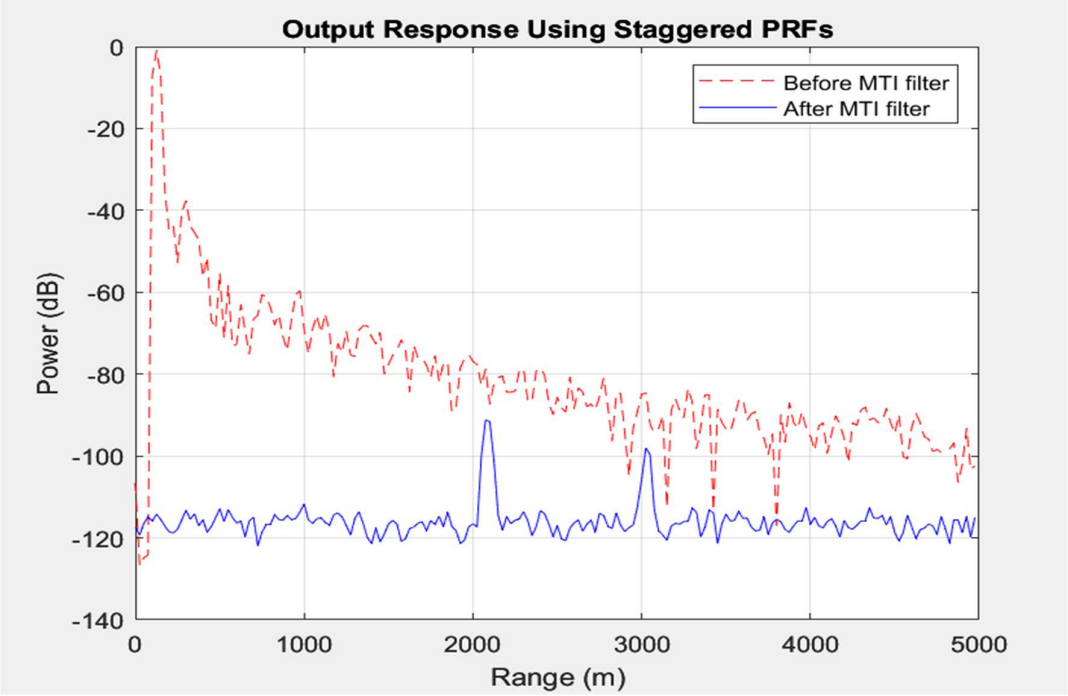


Fig- Frequency response of delay line canceller using staggered prf

- **Simulating the received pulse using staggered pulse repetition frequency**

  Here we can see, both the target is detectable now and clutter have also removed.



Conclusion: Thus, MTI processing becomes very advantageous for removing dc echoes(clutters) at low frequencies. A uniform pulse repetition frequency may miss the blind speeds however after using the Staggered pulse frequencies we are able to remove these blind speeds as well.

References:

1. Skolnik, Merrill I. Introduction to Radar Systems. McGraw-Hill, 2001.

2. Mahafza, Bassem R., and Atef Z. Elsherbeni. MATLAB Simulations for Radar Systems.

3. A. Zverev, "Digital MTI radar filters," in IEEE Transactions on Audio and Electroacoustics, vol. 16, no. 3, pp. 422-432, September 1968, doi: 10.1109/TAU.1968.1162003.

4. E. J. Barlow, "Doppler Radar," in Proceedings of the IRE, vol. 37, no. 4, pp. 340-355, April 1949, doi: 10.1109/JRPROC.1949.231638.

5. I. Cohen and N. Levanon, "Adjusting 3-pulse canceller to enhance slow radar targets," 2015 IEEE International Conference on Microwaves, Communications, Antennas and Electronic Systems (COMCAS), Tel Aviv, Israel, 2015, pp. 1-5, doi: 10.1109/COMCAS.2015.7360360.

Appendix

```
clc;
close all;
load BasicMonostaticRadarExampleData;
Radar_height = 100;
Radar.startingPosition = [0 0 Radar_height]';
prf = waveform.PRF;
fs = waveform.SampleRate;
fc = radiator.OperatingFrequency;
wavespeed = radiator.PropagationSpeed;
transmitter.CoherentOnTransmit = false;
transmitter.PhaseNoiseOutputPort = true;
receiver.PhaseNoiseInputPort = true;
signal_wavelength = wavespeed/fc;
blindspeed = dop2speed(prf,signal_wavelength)/2;
targetposition = [[1600 0 1300]',[2900 0 800]'];
targetlevel = [[100 80 0]',[-blindspeed 0 0]'];
targetmotion =
phased.Platform('InitialPosition',targetposition,'Velocity',targetlevel);
tgtrcs = [25 25];
target = phased.RadarTarget('MeanRCS',tgtrcs,'OperatingFrequency',fc);
trgamma = surfacegamma('flatland');
clutter = constantGammaClutter('Sensor',antenna,...
'PropagationSpeed',radiator.PropagationSpeed,...
'OperatingFrequency',radiator.OperatingFrequency,...
'SampleRate',waveform.SampleRate,'TransmitSignalInputPort',true,...
'PRF',waveform.PRF,'Gamma',trgamma,'PlatformHeight',Radar_height,...
'PlatformSpeed',0,'PlatformDirection',[0;0],...
'MountingAngles',[0 0 0],'ClutterMaxRange',5000,...
'ClutterAzimuthSpan',360,'PatchAzimuthSpan',10,...
'SeedSource','Property','Seed',2011);
pulsenum = 10;
receiver.SeedSource = 'Property';
receiver.Seed = 2010;
rxPulse = MTISimulate(waveform,transmitter,receiver,...
radiator,collector,sensormotion,...
target,targetmotion,clutter,pulsenum);
matchingcoeff = getMatchedFilter(waveform);
matchedfilter = phased.MatchedFilter('Coefficients',matchingcoeff);
mfiltOut = matchedfilter(rxPulse);
matchingdelay = size(matchingcoeff,1)-1;
mfiltOut = buffer(mfiltOut(matchingdelay+1:end),size(mfiltOut,1));
```

```
h = [1 -2 1];
mtiseq = filter(h,1,mfiltOut,[],2);
mtiseq = pulsint(mtiseq(:,3:end));
mfiltOut = pulsint(mfiltOut(:,3:end));
fast_time_grid = (0:size(mfiltOut,1)-1)/fs;
rangeidx = wavespeed*fast_time_grid/2;
figure
plot(rangeidx,pow2db(mfiltOut.^2),'r--',...
rangeidx,pow2db(mtiseq.^2),'b-' ); grid on;
title('Output response Using a Uniform PRF');
xlabel('Range (m)'); ylabel('Power (dB)');
legend('Before MTI filter','After MTI filter');
f = linspace(0,prf*9,1000);
hresp = freqz(h,1,f,prf);
figure
plot(f/1000,20*log10(abs(hresp)));
grid on; xlabel('Doppler Frequency (kHz)'); ylabel('Magnitude (dB)');
title('Frequency Response of the Three-Pulse Canceler');
hold on;
prf = wavespeed./(2*[6000 5000]);
pf1 = @(f)(1-2*exp(1j*2*pi/prf(1)*f)+exp(1j*2*pi*2/prf(1)*f));
pf2 = @(f)(1-2*exp(1j*2*pi/prf(2)*f)+exp(1j*2*pi*2/prf(2)*f));
sfq = (abs(pf1(f)).^2 + abs(pf2(f)).^2)/2;
figure
plot(f/1000,pow2db(sfq),'r--');
ylim([-50, 30]);
xlabel('Doppler Frequency (kHz)'); ylabel('Magnitude (dB)');
title('Frequency Response of the Three delay line Canceler in case of staggered
prf');
release(waveform);
waveform.PRF = prf;
release(clutter);
clutter.PRF = prf;
release(receiver);
receiver.Seed = 2010;
reset(sensormotion);
reset(targetmotion);
rxPulse = MTISimulate(waveform,transmitter,receiver,...
radiator,collector,sensormotion,...
target,targetmotion,clutter,pulsenum);
mfiltOut = matchedfilter(rxPulse);
mtiseq = filter(h,1,mfiltOut,[],2);
mtiseq = pulsint(mtiseq(:,3:end));
mfiltOut = pulsint(mfiltOut(:,3:end));
```

```
fast_time_grid = (0:size(mfiltOut,1)-1)/fs;
rangeidx = wavespeed*fast_time_grid/2;
figure
plot(rangeidx,pow2db(mfiltOut.^2),'r--',...
rangeidx,pow2db(mtiseq.^2),'b-' ); grid on;
title('Output Response Using Staggered PRFs');
xlabel('Range (m)'); ylabel('Power (dB)');
legend('Before MTI filter','After MTI filter');
function rxpulses = MTISimulate(waveform,transmitter,receiver,...
radiator,collector,sensormotion,...
target,tgtmotion,clutter,pulsenum)
fs = waveform.SampleRate;
prf = waveform.PRF;
fc = radiator.OperatingFrequency;
fast_time_grid = unigrid(0, 1/fs, 1/max(prf), '[)');
channel = phased.FreeSpace(...
'SampleRate',fs,...
'TwoWayPropagation',true,...
'OperatingFrequency',fc);
rxpulses = zeros(numel(fast_time_grid),pulsenum);
for m = 1:pulsenum
pulse = waveform();
steptime = length(pulse)/fs;
[sensorpos,sensorvel] = sensormotion(steptime);
[tgtpos,tgtvel] = tgtmotion(steptime);
[~,tgtang] = rangeangle(tgtpos,sensorpos);
[pulse,txstatus,phsnoise] = transmitter(pulse);
txsig = radiator(pulse,tgtang);
txsig = channel(txsig,sensorpos,tgtpos,sensorvel,tgtvel);
tgtsig = target(txsig);
csig = clutter(pulse(txstatus>0));
rxsig = collector(tgtsig,tgtang);
rxsig = receiver(rxsig+csig,~(txstatus>0),phsnoise);
rxpulses(:,m) = rxsig(1:length(fast_time_grid));
end
end
```