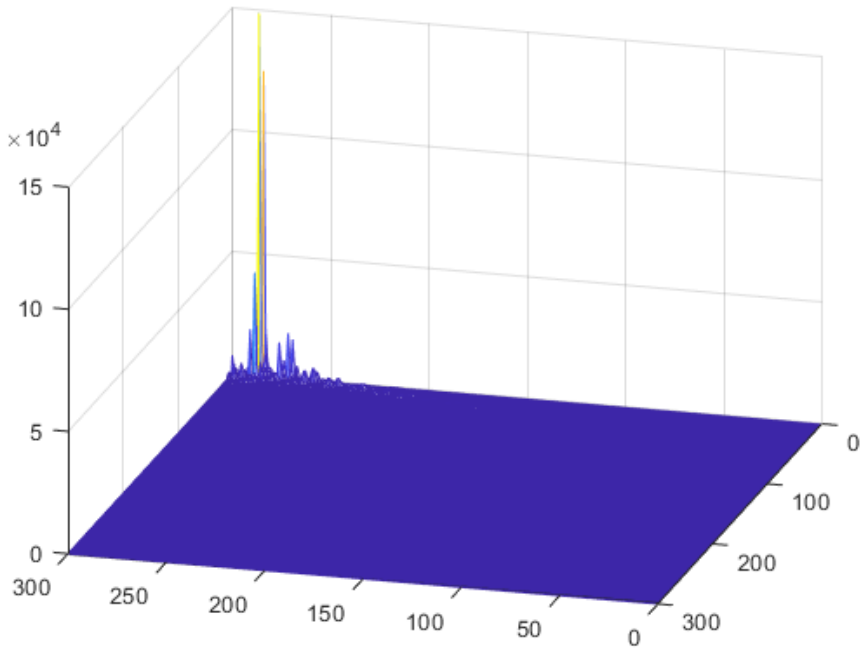
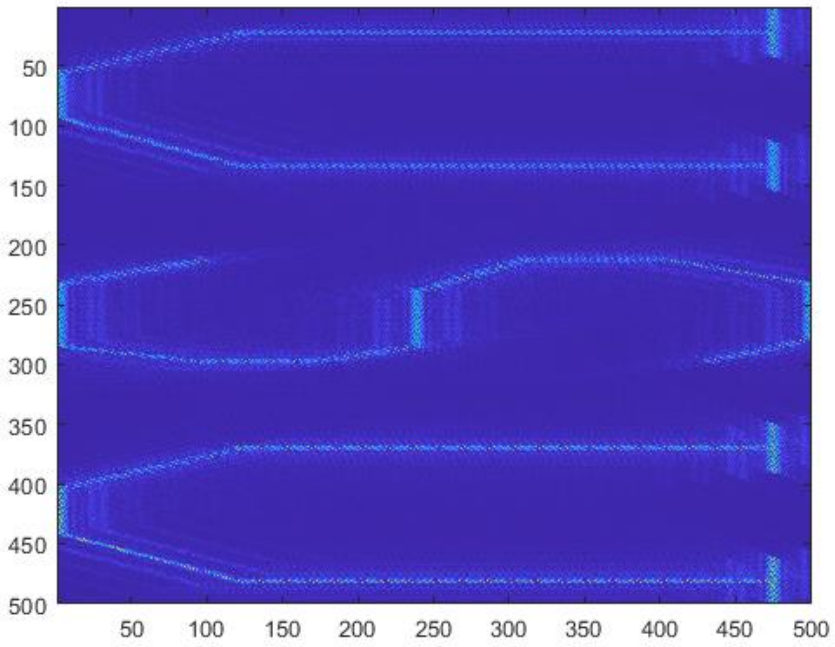
Joshua Brock
ECE5930 – Radar Signal Processing
HW 8 – Backprojection

(a) Point Target (300x300 resolution)



(b) Scene (500x500 resolution)



I learned a fair amount while doing this homework. I learned more about backprojection and how to implement it. I was also reminded of interpolation using FFT's from looking at SDL's code. I learned that while backprojection is fairly simple in theory, it can be challenging to get the indexing just right, especially when it comes to indexing into the interpolated image. Indexing was something I had some issues with. I was able to overcome those issues when I went back and watched the lecture where we went through the code and some things were further clarified.

I actually got the back projection running pretty fast. The 500x500 USU image only took about 5 minutes, which is much faster than I expected it to be. I think preallocating the arrays helped a fair amount with speed.

Code:

```
% Simulate a SAR system.
addpath(genpath('../../utils/'));

clear;

image = 2;
%% Radar info

height = 2000;                          % [m] height of radar
freqPulseRepitition = 700;              % pulse repetition frequency
sampleInterval = 1/freqPulseRepitition; % sample interval
pulseDuration = 6.0e-6;                % duration of LFM pulse (Tr)
fmRate = 4*1e6*1e6;                     % LFM FM rate
platformVelocity = 200;                 % [m/s] velocity of platform
speedOfLight = 3e8;                     % speed of light
lambda = 0.03;                % [m] wavelength of radar
carrierFreq = speedOfLight/lambda; % carrier frequency
AntennaLength = 1;                      % antenna length
Fs = 30e6;                         % (fast time) samples per second
Ts = 1/Fs;                % sample time
squintAngle = deg2rad(30);
theta3dB = lambda / AntennaLength;

plotNumber = 0;
plotNumber = plotNumber + 1;
figure(plotNumber);

if (image == 1)
    recieved = load('sarIMG1.mat');
else
    recieved = load('sarIMG2.mat');
end
```

```
miny = 7000;
maxy = 8000;
minx = 0;
maxx = 280.105;
% pointY = 7000;
% pointX = 264.25;
rangeGateTime = 5.176788819470330e-05;

%% first sample paramameters
if image == 1
    eta0 = -19.788571428571426;
    y0 = 0;
    x0 = -3.957714285714285e+03;

else
    %recieved = load('sarIMG2.mat');
    eta0 = -23.777142857142859;
    y0 = 0;
    x0 = -4.755428571428572e+03;
end

recvSignal = recieved.recv;
mesh(real(recvSignal));

%% compute azimuth distance values
nAzTimes = size(recvSignal,1);
azTimes = (0:(nAzTimes-1)) * sampleInterval + eta0;
azVals = azTimes * platformVelocity;

%% number backprop pixels
numPixY = 500; %pixels
numPixX = 500;

dY = (maxy - miny) / numPixY;
dX = (maxx - minx) / numPixX;

chirpTimeList = 0:Ts:pulseDuration;

nSamples = length(recvSignal(1,:));
chirp = mychirpcausal(chirpTimeList,pulseDuration,fmRate,0);
convolutionLength = nSamples + length(chirp) - 1;
NfftRg = 2^nextpow2(convolutionLength);
chirpFft = fft(conj(chirp),NfftRg);
```

```
%% Backprojection
reconimg = zeros(numPixX, numPixY);
jlist = zeros(numPixX);
taulist = zeros(numPixX);
idxlist = zeros(numPixX);
for k = 1:length(azTimes)   % for each azimuth position

    ry = 0;
    rx = azVals(k); % only x changes
    rz = height; % radar position

    u = rx;
    recvchirp = recvSignal(k, :);
    if (max(abs(recvchirp)) == 0) % norecvSignal here
        continue;
    end
    recvchirpfft = fft(recvchirp, NfftRg);   % transform
    rgCmpfft = recvchirpfft .* chirpFft;            % multiply
    rgCmp = ifft(rgCmpfft);                         % inverse transform
    rgCmp = rgCmp(1:convolutionLength);             % choose the piece to keep

    % for each pixel y
    for iIdx = 1:numPixY%nrowsrecon
        y = miny + (iIdx-1) * dY; % miny
        z = 0;

        a1 = rx + y*tan(squintAngle + theta3dB/2);
        a2 = rx + y*tan(squintAngle - theta3dB/2);

        numj = 0;

        for jIdx = 1:numPixX
            x = minx + (jIdx-1) * dX;

            % if the target falls in the beam
            if (x < a1 && x > a2)
                numj = numj + 1;
                jlist(numj) = jIdx; %[jlist jIdx];
                dist = norm([ry,rx,rz] - [y,x,z]);
                tautilde = dist*2/speedOfLight;
                taulist(numj) = tautilde; %[taulist tautilde];
                idx = floor((tautilde - rangeGateTime + pulseDuration/2)/Ts + 1);
                idxlist(numj) = idx; %[idxlist idx];
            end
        end
```

```
        interpRow = interp1((1:convolutionLength), rgCmp, idxlist);
        jct = 0;
        for kIdx = 1 : numj
            jIdx = jlist(kIdx);
            reconimg(iIdx,jIdx) = reconimg(iIdx,jIdx) + ...
                exp(1j*2*pi*carrierFreq*taulist(kIdx))*interpRow(kIdx);
        end

    end

end

figure(10);
imagesc(abs(reconimg));
```