Synthetic Aperture Radar Imaging Simulated in MATLAB

A Thesis

presented to

the Faculty of the

California Polytechnic State University, San Luis Obispo

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Matthew Schlutz

June 2009

Supported by Raytheon Space and Airborne Systems Division

COMMITTEE MEMBERSHIP

TITLE: Synthetic Aperture Radar Imaging Simulated in MATLAB

AUTHOR: Matthew Schlutz

DATE SUBMITTED: June 2009

COMMITTEE CHAIR: Professor John Saghri

COMMITTEE MEMBER: Professor Xiao-Hua Yu

COMMITTEE MEMBER: Professor Wayne Pilkington

Abstract

Synthetic Aperture Radar Imaging Simulated in MATLAB

Matthew Schlutz

This thesis further develops a method from ongoing thesis projects with the goal of generating images using synthetic aperture radar (SAR) simulations coded in MATLAB. The project is supervised by Dr. John Saghri and sponsored by Raytheon Space and Airborne Systems. SAR is a type of imaging radar in which the relative movement of the antenna with respect to the target is utilized. Through the simultaneous processing of the radar reflections over the movement of the antenna via the range Doppler algorithm (RDA), the superior resolution of a theoretical wider antenna, termed synthetic aperture, is obtained. The long term goal of this ongoing project is to develop a simulation in which realistic SAR images can be generated and used for SAR Automatic Target Recognition (ATR). Current and past Master's theses on ATR were restricted to a small data set of Man-portable Surveillance and Target Acquisition Radar (MSTAR) images as most SAR images for military ATR are not released for public use. Also, with an in-house SAR image generation scheme the parameters of noise, target orientation, the elevation angle or look angle to the antenna from the target and other parameters can be directly controlled and modified to best serve ATR purposes or other applications such as three-dimensional SAR holography.

At the start of the project in September 2007, the SAR simulation from previous Master's theses was capable of simulating and imaging point targets in a two dimensional plane with limited mobility. The focus on improvements to this simulation through the course of this project was to improve the SAR simulation for applications to more

complex two-dimensional targets and simple three-dimensional targets, such as a cube. The input to the simulation uses a selected two-dimensional, grayscale target image and generates from the input a two-dimensional target profile of reflectivity over the azimuth and range based on the intensity of the pixels in the target image. For three-dimensional simulations, multiple two-dimensional azimuth/range profiles are imported at different altitudes. The output from both the two-dimensional and three-dimensional simulations is the SAR simulated and RDA processed image of the input target profile.

Future work on this ongoing project will include an algorithm to calculate line of sight limitations of point targets and processing optimization of the radar information generation implemented in the code so that more complex and realistic targets can be simulated and imaged using SAR for applications in ATR and 3D SAR holography.

Acknowledgments

I first would like to thank my faculty advisor, Dr. John Saghri, for his support and guidance throughout the course of the project. Also, I want express my gratitude to Raytheon Space and Airborne Systems, specifically Jeff Hoffner and Kurt Spaeter, for their continuing support, approval of funding for this ongoing project, and allowing collaboration with industry to exist. Without the efforts of both Dr. Saghri and Raytheon, this project would not be possible. As for the previous work done on the ongoing SAR simulation at Cal Poly, I would like to thank Lynn Kendrick for laying the groundwork for the SAR simulation to be developed, Brian Zaharris for developing the two-dimensional RDA and SAR simulation, and Paul Mason for further expanding on two-dimensional SAR simulation and beginning work on the three-dimensional SAR simulation. Finally, I would like to thank Jessica Kiefer, John Hupton, and Scott Seims for their assistance and collaboration throughout the course of the project as they worked on their own theses in different areas of SAR.

Table of Contents

Table of Figures

I. SAR Introduction

Synthetic aperture radar offers dramatically improved image resolution over radar without sophisticated post processing by utilizing the movement of the antenna with respect to the target. Due to diffraction, a smaller aperture, or antenna length, such as the kind that could fit on the side of a plane or on a satellite, will produce a beam that is much wider once it has made its way to earth or to a target at a long distance as shown in Figure 1. The wider the beam on the target, the more difficult it is to discern when reflections are coming from the target and thus locate the target. In order improve the resolution without post processing, a larger antenna would have to be used that would generate a narrower beam pattern on the target and make it easier to distinguish target features. This is counterintuitive, but due to diffraction, the rate of divergence of the beam pattern is disproportional to the width of the aperture. If an antenna platform is moving with respect to the target, many beams can be sent from and received at the antenna as shown in Figure 1 and when these reflections are analyzed together by coherent combination using the range Doppler algorithm (RDA), they can generate an image with the resolution of a much wider "synthetic" aperture. The RDA uses matched filtering to generate an image of the radar illuminated target by correlating the obtained noisy raw SAR signal with template reflections from ideal radar scattering. [4, 5 ,9]

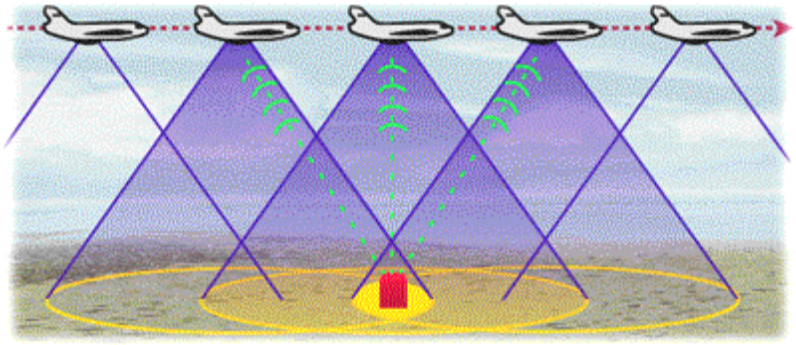

Figure 1: Multiple Radar Beams[5]

II. SAR History

The history of SAR and radar in general is important when investigating methods of SAR imaging. Radar was initially developed in World War II for tracking of ships and aircrafts. The radio frequency used had the advantage of being able to penetrate through heavy weather and darkness to locate targets when remote sensing with visible light would have failed. Conventional radar systems operate by measuring range to the target from the time it requires for the radar pulse to travel to and echo back from the target and direction to the target from how much of the signal was received compared to the direction of the antenna. The Doppler shift of the pulse as it reflects off of the target also carries information on the speed of the target. Dennis Gabor, the engineer at British Thomson-Houston who invented holography in 1947, laid the groundwork for SAR with the principles of wavefront reconstruction. Carl Wiley of Goodyear Aerospace discovered that post processing of the Doppler shift information provided the ability to obtain finer resolution in the direction of the travel of the beam. Using the Doppler shift post processing technique and wavefront reconstruction theory, in 1951 Wiley developed the process known as SAR imaging by which two-dimensional images of targets and the earth's surface could be constructed using radar. [3,4, 9]

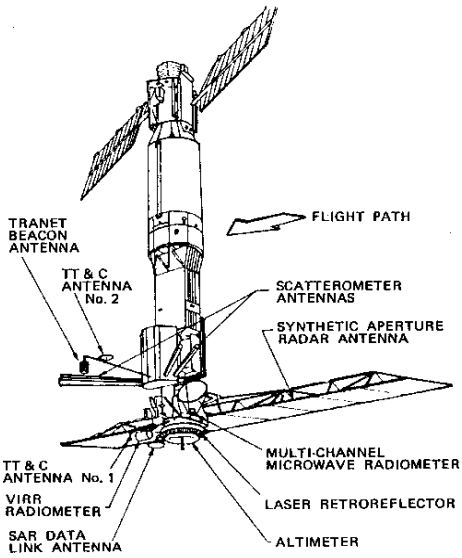

Figure 2: SEASAT[2]

It was not until the 1970s that military SAR technology was released to

the civilian community and radar imaging came into wide use to complement visual sensors. Radar imaging solutions allowed for increased capabilities when the optical sensors were rendered useless due to poor sun exposure or weather conditions. Though early work with SAR imaging was done with aircraft, it was not until 1978 that the launch of the NASA's first earth orbiting satellite for remote sensing of the oceans, SEASAT shown in Figure 2, that the use of radar for remote sensing became widespread practice. SEASAT used the L-Band frequency of 1.175 GHz at an altitude of 800 km, an angle of incidence of 23 degrees, and with a swath width of 100 km. It was able to obtain images with a resolution of 25m in the range and azimuth directions. The azimuth is the direction parallel to the movement of the platform and the range is normal to the azimuth or perpendicular to the movement of the platform.[2,4,7]

Prior to SEASAT, optical processing was the only form of SAR processing that existed. The radar data, after being collected and stored on film via amplitude modulation of a cathode tube by the radar signal from the receive antenna, was substantially unfocused and needed to be sent through a phase sensitive processor. With Fourier optics principles, it was discovered that using a coherent laser, a carefully positioned system of lenses could take the Fourier transform of the data and perform filtering in the spatial frequency domain before doing another Fourier transform and returning to the final image in the spatial domain. The limitations of this optical processing approach and the need to store the radar data on film prompted research in digital processing for SAR processing on satellites and resulted in the development of a digital processor for SEASAT. The first digital processing algorithm, the RDA, was developed by MacDonald Dettwilder and Jet Propulsion Lab in 1978 for use on SEASAT. The

technique of processing the SAR data separately in the range and azimuth domains was implemented digitally with the use of fast Fourier transforms (FFT) instead of lenses. In 1978, this algorithm could develop at 40 square kilometer image with a resolution of 25 square kilometers in a brisk 40 hours. Average modern computers can process the same image in less than a quarter of a second. SEASAT was operational for 105 days after its launch on October 10, 1978 before a critical short circuit failure ended the mission. However, the implementation of the digital SAR processor on SEASAT proved operational and effective and modifications of the RDA are widely used today in various applications. [4,7]

III. Prior SAR Progress

The ultimate goal of this ongoing project is to develop a piece of code which uses raw SAR signals to generate images of targets that can be used for ATR implementations. However, this project started out as a one-dimensional implementation of radar range finding. Lynn Kendrick developed the one-dimensional radar range finding simulation in 2005 as her Cal Poly Senior project. In a one-dimensional simulation consisting of only range and no moving platform, it is not strictly speaking SAR as the movement of the antenna cannot be utilized to synthetically obtain a larger antenna and there would be no range cell migration correction or azimuth processing phase, but a matched filter, the basic filtering mechanism of the range-Doppler algorithm, was coded and simulated to find of the range of point targets in single range direction. The full two-dimensional SAR simulation and range-Doppler algorithm was developed in 2007 as a Master's thesis by Brian Zaharris. Zaharris' simulation arrayed point targets in a two-dimensional plane of azimuth and range, with the platform traveling in the same plane along the azimuth. Zaharris' simulation also made use of Kalman filtering of the raw SAR signal portion of the code to allow accurate imaging of limited mobility point targets at their final positions. Paul Mason used Brian Zaharris' two-dimensional range-Doppler algorithm and adapted it to two-dimensional objects such geometrical shapes and letters composed of arrays of point targets with the azimuth and range location of each point defined in input profiles. Masons' simulation took a longer amount of time to simulate than Zaharris' and was not as well documented or user friendly. Paul Mason also attempted to implement three-dimensional SAR imaging using Zaharris' two-dimensional simulation as a starting point, but eventually decided due to time limitations to focus on the two-

dimensional simulation. In order to move into three-dimensional SAR simulations, a major flaw in the two-dimensional SAR simulation needed to be solved first. Each point target requires a reflection to be calculated during each stage in the flight over the duration of the flight. With the realistic SAR parameters used, 900 reflections are calculated for each point target in the two-dimensional SAR simulation and due to the complexity of the radar reflection equation used, without optimization of the code each point target would require 30 seconds to calculate each reflection. For larger images, such as MSTAR images which are of size 128x128, the simulation would take over five days to complete. Also, due to lack of optimized memory allocation, practical memory requirements limited Zaharris' simulation to about 20 point targets. Finally, each target profile consisted of the range and azimuth position of every point written out in the MALAB simulation file. A more efficient means for defining each point target position for complex two-dimensional shapes would be required. All of these limitations needed to be addressed in the two-dimensional simulation before it could be adapted into a three-dimensional simulation as the extra dimension would only amplify the problems.[5,11]

IV. SAR Imaging Simulation Setup

The purpose of this section is to describe the two-dimensional SAR imaging geometry and related terminology that is used in the MATLAB simulation. The target, or the object being imaged, is composed of point targets or radar scatters. The point targets are arrayed in a Cartesian type coordinate system space defined by range, azimuth, and altitude as analogs of x, y and z directions. The altitude direction is omitted in the two-dimensional simulation. The platform in this simulation is an antenna attached to a plane traveling at an orbital velocity, V_s, along the azimuth direction and at the midpoint in the flight, the distance to the target equals the range of closest approach or minimum range to target, R_O in the simulation and denoted by X_C Figure 3. As an aircraft is used in the simulation, the curvature of the earth is considered negligible and the orbital velocity is approximately equal to the platform velocity, V_r. The velocity of the radar beam along the ground is V_g. For an earth orbiting satellite case, V_s would be approximately 6% higher than V_r and V_g would be 6% lower than V_r. As the radar dish is

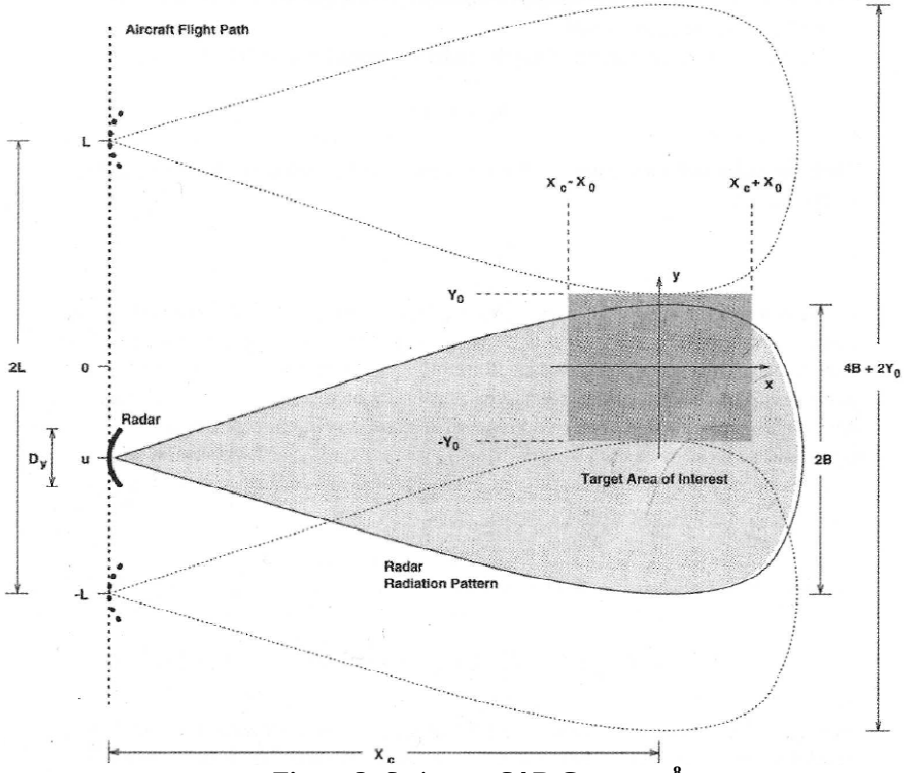

Figure 3: Stripmap SAR Geometry[8]

pointed in a fixed direction during the duration of the flight, this type of SAR is referred to as stripmap SAR, the geometry of which is depicted in Figure 3. As the platform traverses the azimuth, the radar beam sweeps along the ground. The part of the radar beam touching the ground, shown in the circles to the right, is called the beam footprint. The beamwidth is the cone extending from the antenna to the beam footprint. Targets in the beam footprint reflect back radar signals which are then received by the antenna. The obtained radar reflections are processed with the RDA, to obtain the final SAR image.[3,8]

V. Transmitted SAR Signal

The transmitted radar signal, $s_{tx}(t)$, is assumed to be of the form in Equation 1 below in the simulation. The signal is a function of range time or quick time, t. The carrier frequency, f_o, is 4.5 GHz. The chirp pulse duration, T_r, is 2.5 μs and the range chirp or FM rate, K_r, is +40 MHz/μs, which is called an up-chirp because it is positive. Other important parameters are the signal bandwidth, B_o in Equation 2 below, which is 100 MHz and the range resolution, ρ_r in Equation 3 below, which is approximately 1.5 m.

$$s_{tx}(t) = rect\left(\frac{t}{T_r}\right) cos\{2\pi f_o + \pi K_r t^2\} = w_r(t) cos\{2\pi f_o + \pi K_r t^2\} \quad (1)$$

$$B_o = |K_r| T_r \quad (2)$$

$$\rho_r \approx \frac{c}{2} \frac{1}{|K_r| T_r} = \frac{c}{2\, B_o} \quad (3)$$

Figure 4 depicts the transmitted radar signal as a cosine with a linearly ramping up frequency over a transmit duration followed by a null receive duration. The transmit window is called the pulse envelope, w_r, and defines the duration of the transmission. During the receive duration, the antenna waits to receive reflected radar signals from the targets contained in a one-dimensional range slice echo as function of quick time. [3]

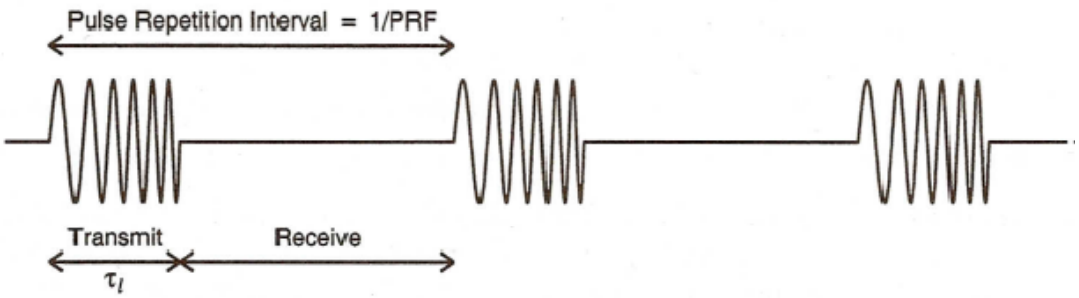

Figure 4: Transmitted Radar Pulse[3]

The magnitude of the radar signal at the antenna during the transmit and receive durations are depicted in succession in Figure 5. One over the combined transmit and receive duration is called the pulse repetition frequency, PRF, and defines the amount of pulses transmitted per second. The PRF for the simulation is 300 Hz and the simulation duration is 3 seconds. This equates to 900 transmitted radar pulses over the duration of the simulation. The result of the plot of the magnitude of each range slice echo as a function of range and azimuth is the raw SAR signal space.

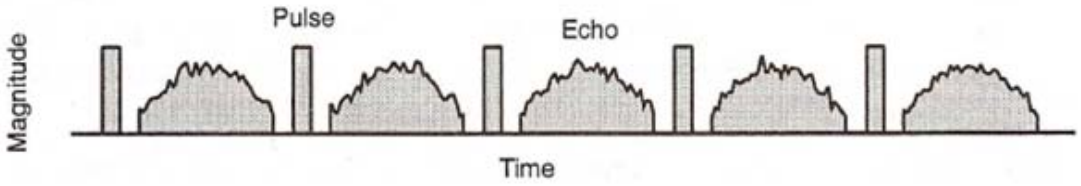

Figure 5: SAR Antenna Activity[3]

VI. Received SAR Signal

The raw SAR received radar signal, $s_{rx}(t,\eta)$, for the simulation is assumed to be of the form shown in Equation 4 below after quadrature demodulation which removes the high frequency carrier wave and brings the signal to baseband. This a three-dimensional signal with two time dimensions. The time dimensions are range time/quick time, t, and azimuth time/slow time, η. The raw SAR signal can be displayed graphically as shown in Figure 8. Equation 4 is shown below as a summation of the reflections from Mdifferent point targets. The MATLAB algorithm uses this equation to generation all of the reflections over the duration of the flight. Quadrature demodulation causes the signal to be imaginary and have a phase and a magnitude. Prior to quadrature demodulation, the signal is the original transmitted signal, which is time delayed, attenuated, phase shifted amplitude modified due to azimuth beam pattern affects, and has additive white Gaussian noise (AWGN) added. This signal shown in Equation 5.

$$s_{rx}(t,\eta) = \sum_{m=0}^{M-1}\left[F_m w_r\left(t-\frac{2R_m(\eta)}{c}\right) w_a(\eta-\eta_c) e^{-j4\pi\left(\frac{f_o R_m(\eta)}{c}\right)+j\pi K_r\left(t-\frac{2R_m(\eta)}{c}\right)^2}\right] + n_m(t,\eta) \quad (4)$$

$$s_{rx}(t,\eta) = \sum_{m=0}^{M-1}\left[F_m w_r\left(t-\frac{2R_m(\eta)}{c}\right) w_a(\eta-\eta_c) cos[2\pi f_o(t-R_m(\eta)/c) + \pi K_r(t-2R_m(\eta)/c)^2+\psi]\right] + n_m(t,\eta) \quad (5)$$

The time delay is $\frac{2R_m(\eta)}{c}$, the attenuation factor from reflection at the target is F_m, the phase shift from reflection at the target is ψ, the azimuth beam pattern amplitude modification is $w_a(\eta-\eta_c)$ as shown in Equation 6, and the additive white Gaussian noise is $n_m(t,\eta)$. The time delay is calculated by the distance the radar beam travels,

twice the instantaneous slant range as shown in Equation 7, divided by the speed of the radar beam, approximately the speed of light. The attenuation factor, F_m, is a scalar value from 0 to 1 representing the normalized reflectivity of each point target. Phase shift information, ψ, is not used in the simulation. The azimuth beam pattern amplitude modification, $w_a(\eta - \eta_c)$, is named due to the geometrical shape of the beam pattern in the azimuth plane as shown in Figure 6. The center node of the beam pattern produces the largest reflection strength, but the smaller side nodes also produce reflections and the overall received signal strength from a point target over azimuth time, η, resembles a sinc squared function centered at the beam center crossing time η_c, which is the azimuth time at which the center of the beam pattern crosses the center target area. The azimuth beam width, β_{bw} used in Equation 6 for calculation of the azimuth beam pattern, is calculated in Equation 8 and is inversely proportional to the actual antenna length, L_a.[3]

$$\omega_a(\eta) \approx sinc^2\left(\frac{0.886\theta(\eta)}{\beta_{bw}}\right) \tag{6}$$

$$R_m(\eta) = \sqrt{{R_{o_m}}^2 + {V_r}^2\,\eta^2} = \sqrt{(X_o + x_m)^2 + {V_p}^2\,\left(\eta + \frac{y_m}{V_p}\right)^2} \tag{7}$$

$$\beta_{bw} = 0.866\lambda/L_a \tag{8}$$

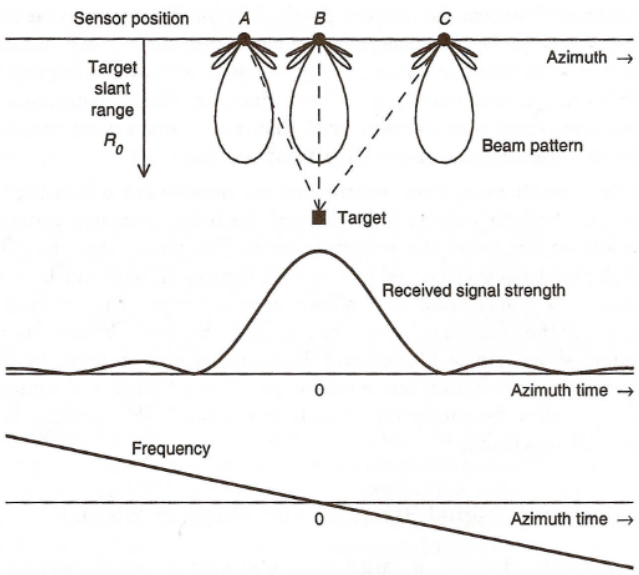

Figure 6: Azimuth Beam Pattern[3]

Important in understanding the effectiveness of SAR processing is the azimuth resolution, ρ_a, as show in Equation 9 below. Simplifications and approximations due to the airplane platform as opposed to satellite in the simulation lead to the approximation of the azimuth resolution to be $L_a/2$. The antenna length parameter in the MATLAB simulation is set to 2 m, leading to an azimuth resolution of 1 m, which is superior to the range resolution of 1.5 m. The azimuth resolution is first shown before simplification as a function of radar beam ground velocity, V_g, squint angle, θ_{sq}, azimuth bandwidth, and Δf_{dop},. The Doppler bandwidth equation is shown in Equation 10 as a function of the orbital velocity, squint angle, wavelength, and 3 dB azimuth width of the main lobe of the azimuth radar beam shown in Figure 6. [3]

$$\rho_a = \frac{0.886 V_g cos\theta_{sq,c}}{\Delta f_{dop}} \approx \frac{L_a}{2} \tag{9}$$

$$\Delta f_{dop} = \frac{2V_s cos\theta_{sq}}{\lambda} \theta_{bw} \tag{10}$$

Equation 10: Doppler Bandwidth

Squint angle, θ_{sq}, used in Equations 9 and 10, is labeled below in Figure 7 as the angle between the slant range vector and the zero Doppler plane. Squint angle varies as a function of slow time, η, decreasing as the platform approaches the target and increasing as the platform moves away from the target as shown in Equation 11 below. The maximum squint angle, $\theta_{sq_{max}}$, calculation is shown in Equation 12 below and for the MATLAB simulation is 0.859°, which is low due to the simulation flight duration of 3 seconds. [3]

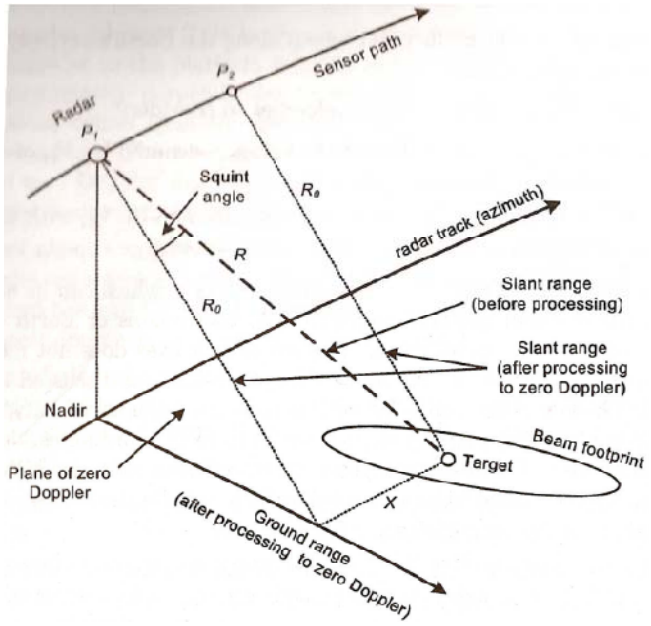

Figure 7: SAR Slant Range and Squint Angle Geometry[3]

$$\theta_{sq} = arccos\left(\frac{R_{om}}{R_m(\eta)}\right) \tag{11}$$

$$\theta_{sq_{max}} = arccos\left(\frac{R_{om}}{R_m(\eta)}\right)\Big|_{R_m(\eta)=\left(\frac{dur}{2}\right)V_p} = arccos\left(\frac{2R_{om}}{durV_p}\right) \tag{12}$$

The benefit of the SAR processing can be seen in the calculation of the azimuth resolution without it as shown in Equation 13, which is called real aperture radar resolution. The real aperture radar using the parameters of the simulation is 590.67 m, which is two orders of magnitude greater than the synthetic azimuth resolution of 1 m.

$$\rho'_a = R(n_c)\theta_{bw} = \frac{0.886R(n_c)\lambda}{L_a} \tag{13}$$

The received and demodulated radar signal of Equation 4 is referred to as the SAR signal space as it is still in its raw form and the two-dimensional image of the magnitude of the two-dimensional imaginary signal as shown in Figure 8 below would not allow recognition of targets. The final image obtained after processing the SAR signal space is called the SAR image space, which is the image used for ATR.

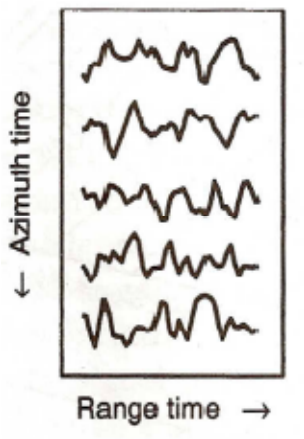

Figure 8: SAR Image Construction[3]

VII. Range Doppler Algorithm

The RDA processes the raw SAR data calculated from Equation 4 to produce the SAR image space or final image. The RDA performs matched filtering separately in the Fourier transformed range and azimuth domains. The Fourier transforms are calculated via fast Fourier transforms (FFTs) for processing time efficiency. Range cell migration correction (RCMC) is performed in the range time and azimuth frequency domain. This domain is called the range-Doppler domain and gives its name to the algorithm as performing the RCMC in this domain is the defining feature of the algorithm when compared to other SAR processing algorithms.

A block diagram of the RDA is shown in Figure 9. The raw signal space SAR input is the two-dimensional signal as shown in Figure 8. The two-dimensional signal is first analyzed as a series range time signals for each azimuth bin. Each range time signal undergoes matched filtering in the range frequency/azimuth time domain through range FFTs applied to the range time signals. After each signal is transformed back into the range time/azimuth time domain, the result is the range compressed signal as the matched filtering was performed in the range frequency domain. In order to obtain azimuth compression, azimuth matched filtering must be performed. The range compressed signal is then composed into a series of signals with respect to azimuth time at different range bins. Each azimuth signal is Fourier transformed via an azimuth FFT and RCMC is performed before azimuth matched filtering in the range-Doppler domain. After azimuth matched filtering of each signal and azimuth inverse fast Fourier transforms (IFFTs), the final target image is obtained. More in depth analysis of these processes and example RDA steps on a single point target in two-dimensional geometry SAR follows.

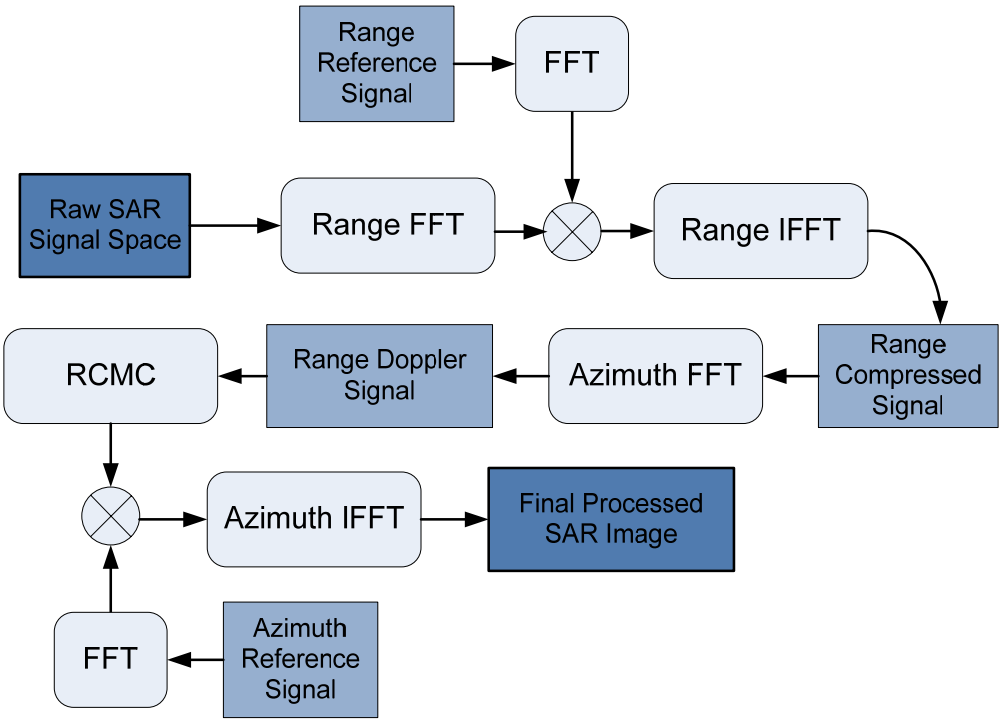

Figure 9: RDA Block Diagram

A main technique of the RDA is matched filtering, a common technique in communications. Matched filtering is the correlation of a template signal with an unknown signal, which is the equivalent of convolution of an unknown signal with a time reversed template, to detect the presence of the template signal in the unknown signal. This detection is effective even in low signal to noise ratio (SNR) cases. In the example in Figure 10 below, the transmitted radar signal is denoted as *s(t)* and the received radar signal is modeled as a time delayed version of *s(t)*. The matched filter template, *h(t)*, is the time reversed version of *s(t)* and the convolution of the two produces a compressed pulse of energy centered around the time delay of radar reflection. This is a one-dimensional radar range finding system of the type implemented by Lynn Kendrick. [1,6]

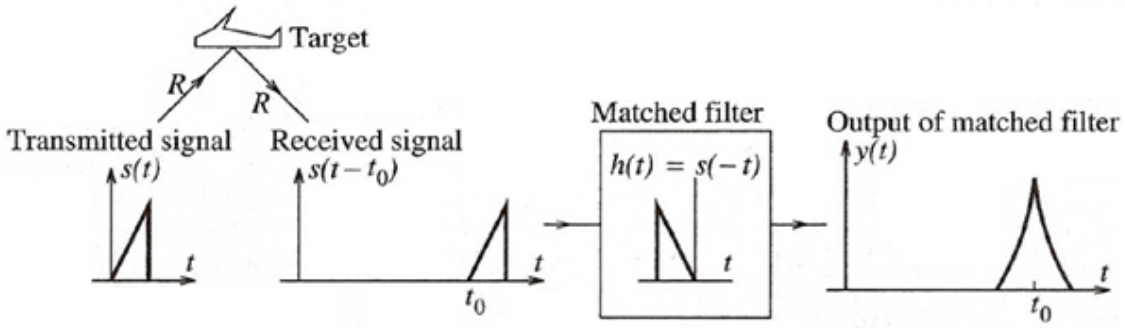

Figure 10: Radar Range Finding Matched Filter Example[1]

Instead of correlation in the time domain, multiplication by the complex conjugate in the Fourier domain, which is an equivalent operation, is performed for speed as it is equivalent and less processing intensive. Matched filtering in the simulation is termed pulse compression as the energy of the received SAR signal converges to or is compressed to the regions of template signal detection. This process is enhanced by the chirp signal used in the transmitted radar signal construction as there is more information embedded for detection. To improve computational efficiency the FTT is used, which is a radix-2 algorithm for efficient computation of the discrete Fourier transform, (DFT) and its inverse. The radix-2 feature of the FFT constrains the number of processed time samples to be an integer multiple of two. FFTs use block processing through simultaneous computations of different inputs which makes them highly efficient. [1,6]

Aside from matched filtering, the other main component of the RDA is the RCMC. RCMC is needed due to the hyperbolic trend with respect to azimuth time η of the instantaneous slant range $R_m(\eta)$ as shown in Equation 7 causing range cell migration (RCM). The RCM with respect to azimuth frequency, f_η, in the range-Doppler domain, range time and azimuth frequency domain, is shown in Equation 14 as it is calculated in the simulation. The approximation in Equation 14 is close for low squint angles, which is assumed in the simulation. The azimuth frequency is found through its calculation with

Equation 15 using the azimuth FM rate, K_a. In the simulation, the RCM is rounded to nearest integer as the migration must be calculated in discrete "cells" to be corrected for during the RCMC process. The cells are shifted to counter RCM in the azimuth frequency domain prior to azimuth matched filtering to perform RCMC. In the next section, each step of the RDA in the simulation will be discussed in detail.[8]

$$R_{rd(f_\eta)} = \frac{R_{o_m}}{\sqrt{1-\frac{c^2 {f_\eta}^2}{4{V_r}^2 {f_O}^2}}} \approx \frac{\lambda^2 R_{o_m} {f_\eta}^2}{8{V_r}^2} \tag{14}$$

$$f_\eta \approx -K_a \eta \approx \frac{2{V_r}^2 \eta}{\lambda R_{o_m}} \tag{15}$$

VIII. Prior Two-Dimensional SAR Imaging Simulation

In spring 2007 Brian Zaharris completed his Master's Thesis on the MATLAB simulation of SAR using the RDA to image point targets with limited mobility. Paul Mason, who collaborated with Brian Zaharris, also completed his Master's Thesis in spring 2007 by modifying Zaharris' RDA for two-dimensional shapes composed of arrays of point targets. Initially, Mason attempted to move the simulation into using targets in a three-dimensional geometry, but scaled back due to the added difficulty in the time constraint of the project. The goal of this project starting in Fall 2007 was to investigate two-dimensional target imaging techniques and modifications of the MATLAB algorithm for three-dimensional targets. Mason's code was a prime starting point due to it already having an implementation of two-dimensional objects composed of arrays of point targets, but the simulation was not as robust or easy to troubleshoot as Zaharris' in which the code was shorter and more efficient . As Zaharris' MATLAB code of point target SAR simulation in two-dimensions with the RDA processing provided the necessary components with which to design more complex two-dimensional and eventually three-dimensional target simulations, it was eventually selected as the starting platform.[5,11]

The RDA defined in the previous section was designed and originally implemented by Zaharris. To better understand how the two-dimensional point target simulation works, an example using a single point in the center of the target area follows. The parameters for the simulation can be found in Appendix A. The simulation used is the modified two-dimensional SAR simulation in Appendix B. It was based on Zaharris' simulation and the simulation parameters and results are similar for the case of a single

point target. The SAR simulation in Appendix B, as with all SAR simulations in this report, use the FFT and IFFT functions in Appendix E. First, the two-dimensional SAR signal space is calculated using Equation 4. The result is shown in Figure 11. The signal space is shown with a jet color map, with cooler colors representing lower magnitudes and hotter colors representing higher magnitudes. AWGN can be seen throughout the image space and the SAR echoes are shown spanning the entire azimuth and the center of the range. The SAR echo energy is spread over the azimuth following a sinc squared decay pattern near the beam center crossing time as predicted by Figure 6, which is due to the Doppler effect. The sinc squared side bands are not visible due to the flight duration being limited to 3 seconds and the maximum squint angle 0.859°.[11]

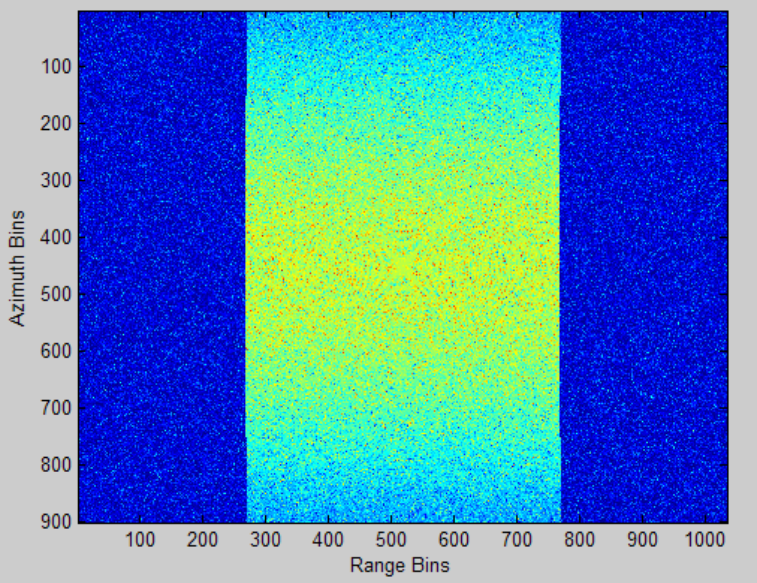

Figure 11: Raw SAR Signal Space

The magnitude of the complex exponential term contains high frequency sinusoidal components in both of the azimuth slow time, η, and range quick time, t, directions. A zoomed in portion of the center of the signal space is shown in Figure 12.

The frequency of the sinusoid increases further away from the target due to the positive up-chirp of the FM rate in Equation 1, the transmitted radar signal.

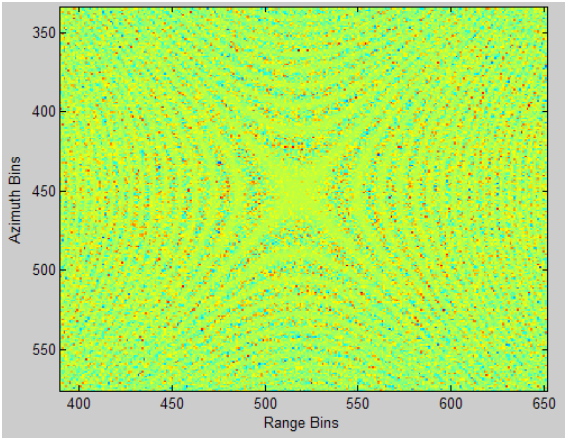

Figure 12: Zoomed in Center Raw SAR Signal Space

The SAR signal space of Figure 11 is fed into the RDA shown in Figure 9. The first part of the process is to assemble the range reference signal in the range frequency domain as shown in Figure 13. The upper left corner contains the range time magnitude of the reflection and the upper right corner contains the range time phase of the reflection. The lower left corner contains the range frequency magnitude component and the lower right corner contains the range frequency phase component.

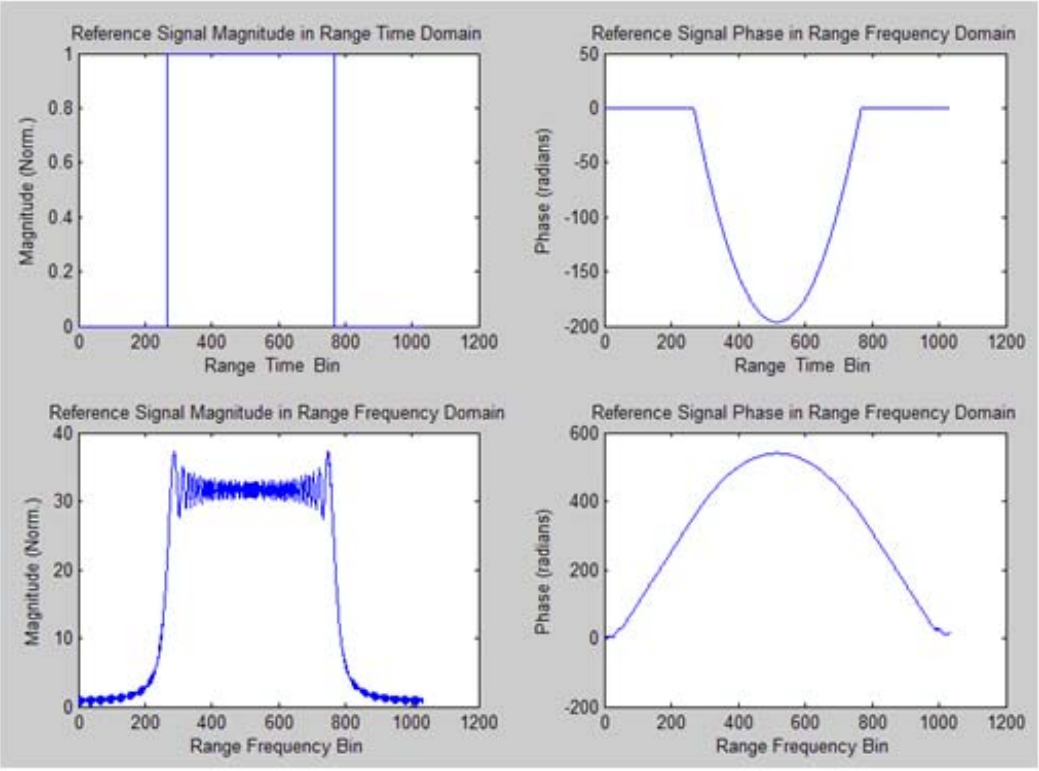

Figure 13: Range Reference Signal

After the range reference signal is constructed, the range cell migration is calculated using the approximation for low squint angle in Equation 14. Figure 14 below shows the results of Equation 14 when implemented in the simulation and rounded to the nearest integer as the range migration can only be corrected in discrete cells, the size of which given by the resolution of the image. As expected from Equation 14, the range cell migration is even with respect to azimuth frequency. After the range reference signal and range cell migration are computed in the RDA, the next step is to compress the SAR signal space in the range direction.

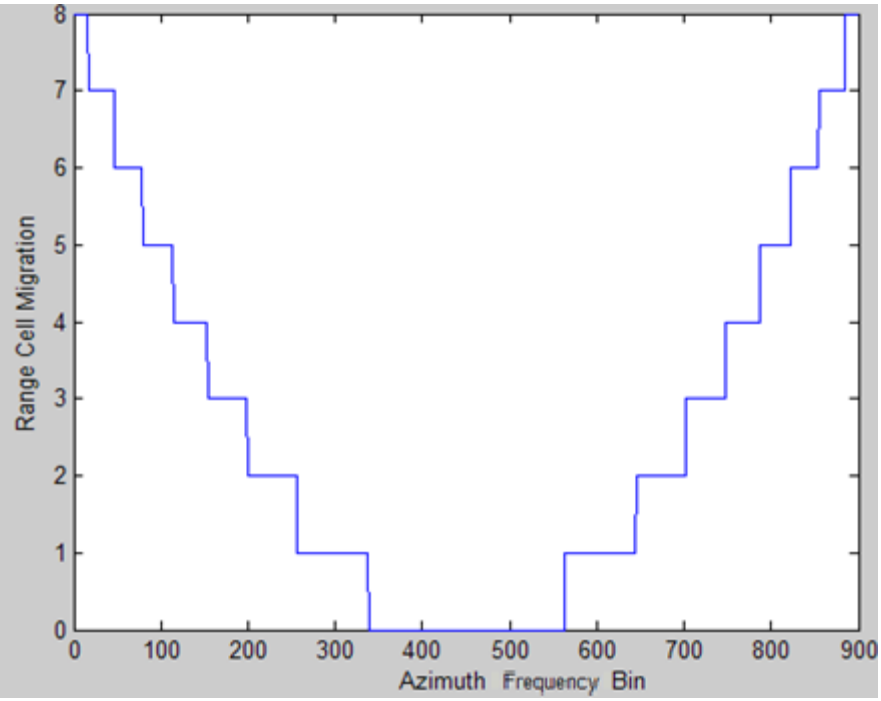

Figure 14: Range Cell Migration

As shown in the RDA block diagram in Figure 9, the raw SAR signal fed into the RDA is sent through a range time domain FFT which takes the Fourier transform of each range time signal. There are PRF, 300 Hz, times the flight duration, 3 seconds, or 900 acquired range time signals in the simulation. The plot at the upper left section of Figure 15 shows the magnitude of an example range time signal acquired halfway through the flight or the 450th signal acquired. The signal resembles the reference range time signal shown in Figure 13, but contains AWGN. The upper right plot shows the FFT of the center azimuth range time signal. The lower left plot shows the output of the matched filter or multiplication of the FFT of the time reversed range reference signal shown in the lower left corner of Figure 13 and the FFT of the range time signal shown in the upper right corner of Figure 15. The term pulse compression gets its name as the input of the matched filter shown in the upper left corner of Figure 15 comes out as a sharp and compressed pulse at the detected location of the reference signal shown in the lower right

corner of Figure 15 after the IFFT of the lower left corner signal. This operation is performed for each, all 900, range signals and the output of this matched filtering process is the range compressed signal shown in Figure 16.

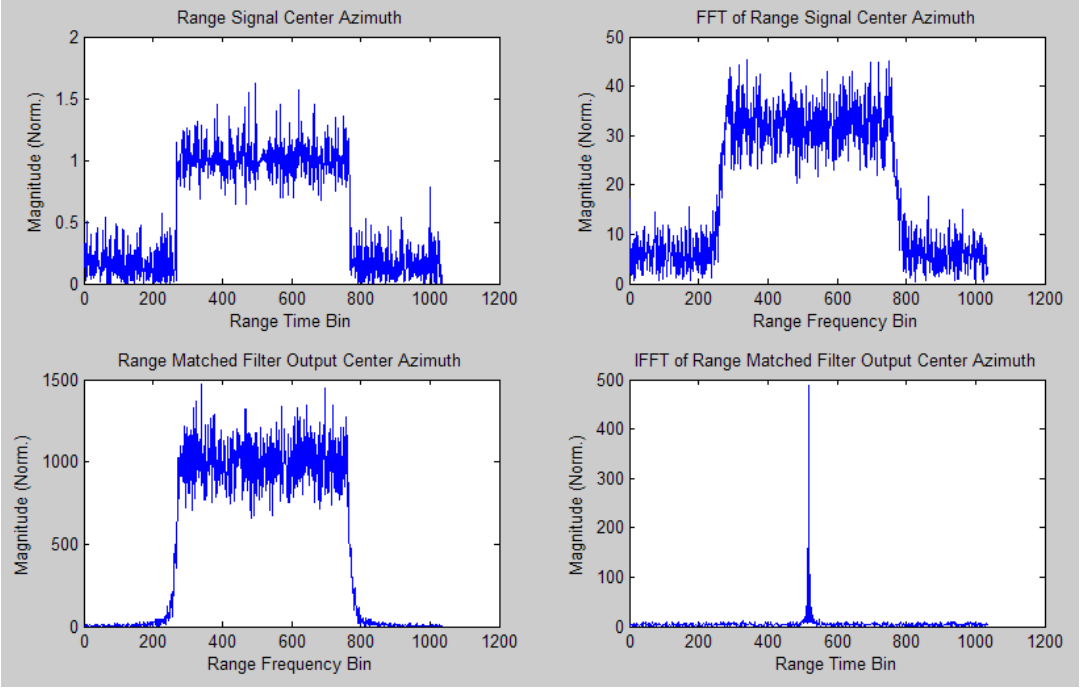

Figure 15: Range Time Signal at Center Azimuth Compression Example

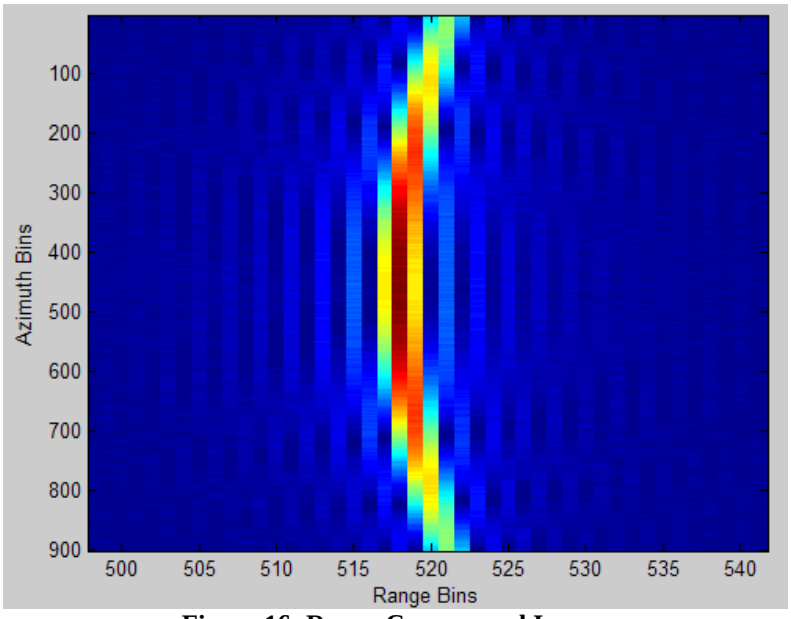

Figure 16: Range Compressed Image

The range compressed signal is compressed in range only and still requires the azimuth matched filter to be compressed in azimuth. Also, the range cell migration predicted before RDA filtering and shown in Figure 14 can be seen in Figure 16. Take note of the compression in the range time scale from Figure 11 to Figure 16. To emphasize the range cell migration, the image is zoomed into in range. One phenomenon of the simulation which has not been properly explained yet in previous SAR imaging simulation projects at Cal Poly is the energy ripples shown to the left and right of the range compressed signal that are lower in magnitude. Ideally, only the center curved signal would be shown. One possible explanation for this is that the range window has sharp edges, which coupled with a finite time duration will result in spectral leakage in the Fourier transformed domain. Matched filtering with spectral leakage could result in the low magnitude ripples side ripples that are observed in Figure 16. A possible solution to this is to use a smoothed window such as a Hann or Blackman window which has lower spectral leakage and will reduce side ripples. In the modifications of the two-dimensional SAR RDA algorithm section of this report, the use of a Hann window to eliminate the side ripples will be addressed.

In the azimuth frequency and range time domain, referred to as the range-Doppler domain, first the RCMC is performed and then the azimuth matched filtering is performed. Figure 17 shows the image when converted back to the azimuth time domain with FFTs after the RCMC. The energy along the azimuth is shifted to counter cell migration in the range direction, RCM, according to the forecasted shift shown in Figure 14.

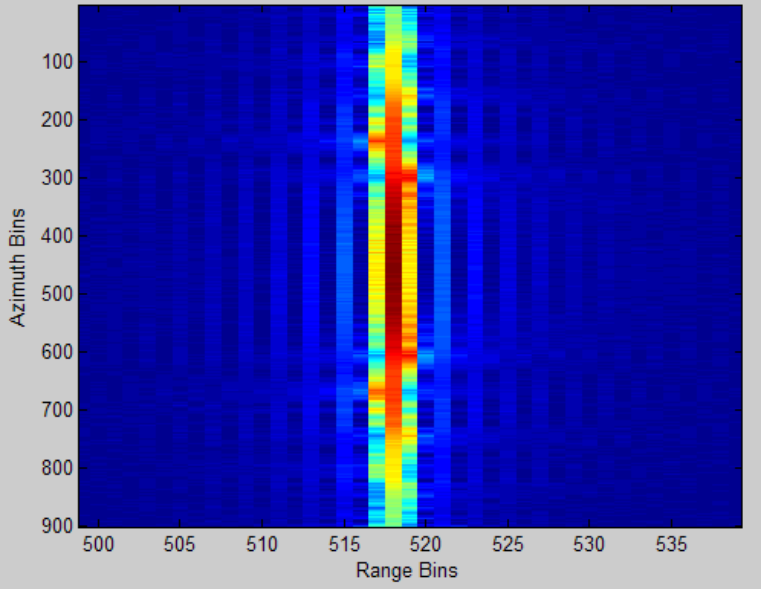

Figure 17: Image after RCMC

The azimuth matching filtering is the final filtering procedure performed on the image before it is converted back to the azimuth time and range time domain and the final processed image can be viewed. The azimuth matched filtering is performed in the same fashion as the range matched filtering, by using a azimuth reference signal similar to the range reference signal shown in Figure 13. The final SAR image generated from the two-dimensional simulation of a single point target is shown in Figure 18.

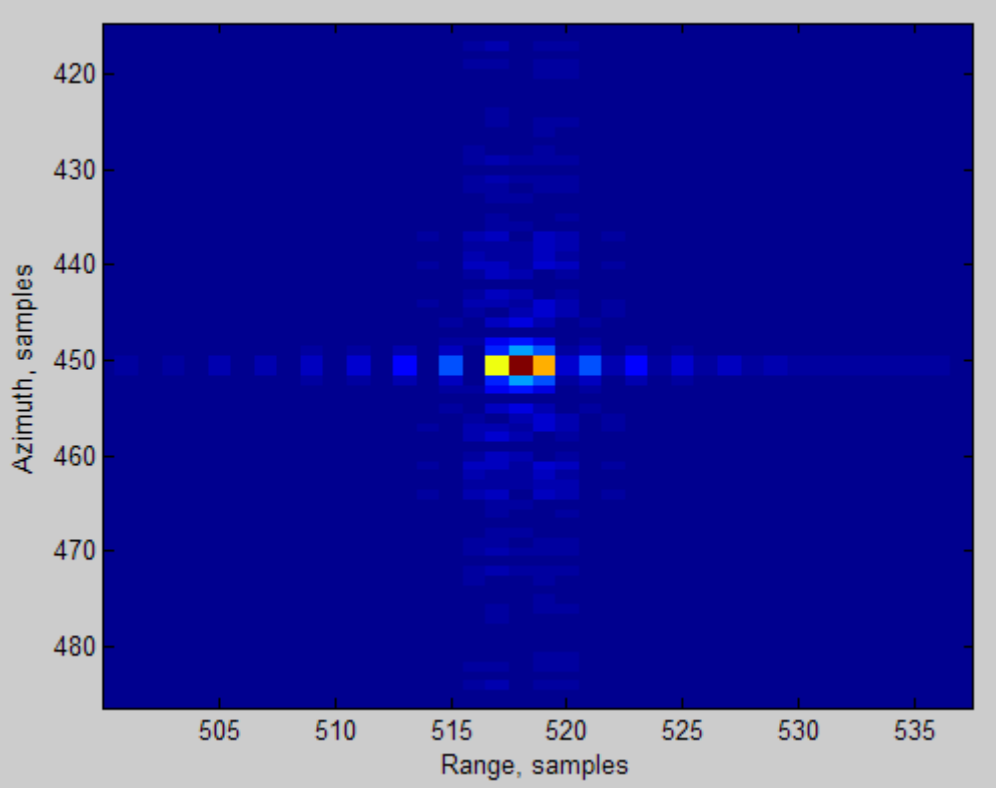

Figure 18: Final Processed SAR Image of Single Point Target

The two-dimensional simulation developed by Brian Zaharris had limitations and a lot of improvements still needed to be made before applications to ATR would be realistic, but it was a good starting platform for two-dimensional SAR simulations with which to improve upon and add to. The next section of this report will document the improvements made to the two-dimensional SAR imaging simulation.

IX. Two-Dimensional SAR Imaging Optimization

Upon starting the project in fall 2007, the SAR simulation supported up to 25 point targets with limited mobility. Each point target's location and reflectivity were individually defined through lines written in code. The memory intensive part of the code was the requirement to save the reflections from each point target separately. Each radar reflection is 1034 range cells long and there are 900 reflections for each point target. Processing a 16 by 16 point target profile would require 256 two-dimensional arrays of doubles of size 900 by 1034, which exceeds the memory limitations of a computer with 2 GB of memory. This section of the report will detail modifications made to the two-dimensional SAR simulation to improve efficiency to allow larger target profiles, the addition of a new target import mechanism, and finally an analysis of the ripple phenomenon seen in Figure 18 and possible solutions for it. The final modified two-dimensional SAR simulation is added to the end of this report in Appendix B.

First, modifications were made to simulation to sum all 256 reflections, as show in Equation 4, and reduce the raw SAR signal space shown in Figure 11 to a single two-dimensional double array of size 900 by 1034 despite the number of point targets. Though this modification is realistic because the signal space reflections from each point target would not be able to be distinguished by the antenna, the limited mobility Kalman filter previously in the simulation relied on knowing what reflections came from what target and became inoperable. The RDA takes minimal time to compute due to the speed of the FFTs used in it when compared to the SAR echo generation sequence. Calculating the radar echo signals in Equation 4 consumes most of the processing time. The SAR echo generation section of the code was rewritten to optimize the speed with which the

code was generated. Generating the SAR echoes from 11 point targets with the original code took 10 minutes. With this processing time, a 16 by 16 pixel image would take over four hours for the SAR echoes to be generated. Further optimizations were made to the echo generation sequence to optimize processing efficiency such as multiplying out constraints before the for loop and using a single constant to call the value in the loop to minimize operations. Currently, a 16 by 16 pixel image takes ten minutes to process for a SAR image to be generated.

The image import feature of the SAR simulation is used to specify the location and reflectivity of point targets based on the location and intensity of the pixels in the imported image. Current import images are 8-bit or 256 intensity level grayscale images of size 16 by 16 pixels, where the intensity level is normalized to produce point target reflectivity. This produces a 256 point target profile. From each point target, 900 reflections are calculated due to the pulse repetition frequency of 300 Hz and the total flight duration of 3 seconds. Sample import images are shown in Figure 19 and Figure 20. Figure 19 is a target modeled after a Boyevaya Mashina Pekhoty (BMP), which is an amphibious tracked infantry combat vehicle. Figure 20 is a tank modeled after a T-84. The vehicles were selected due to varying size, features, length to width ratios, and turret position. The images were designed in Adobe Photoshop and saved in 8-bit grayscale GIF format.

Figure 20: BMP Target Image Profile

Figure 19: Tank Target Image Profile

The simulations for each of the targets shown in Figure 19 and 20 above takes 10 minutes to compute, with 9.5 minutes spent solely in the SAR echo generation sequence. The final processed images of the two targets are shown in Figure 21 and 22. Target images such as these could be used for limited ATR applications. However, the energy for each point target is not entirely located where it should be in the final image and can be seen leaking into the range and azimuth directions.

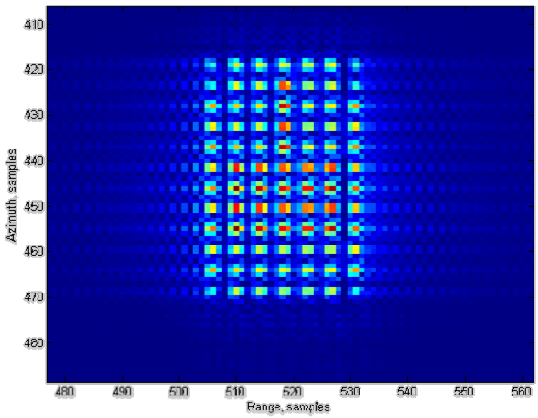

Figure 21: BMP Final Processed Image

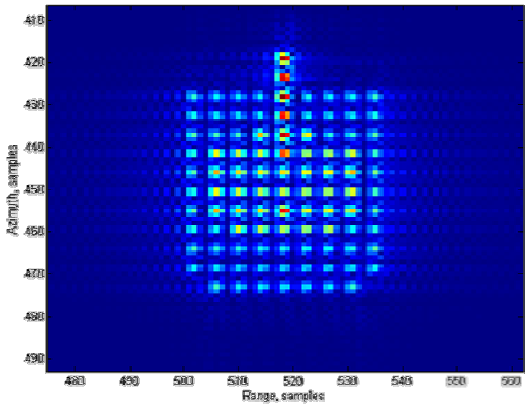

Figure 22: Tank Final Processed Image

To the sides of the targets, bleeding or ripples can be seen in the range direction and the added noise to the echo radar signal prior to processing also has an effect on the image. Bleeding also occurs in the azimuth direction, but less so. The RCMC is visually effective as pixels can be seen to not be warped in the range direction. The turrets in the

input target images, which were the highest intensity, are also the highest intensity, red, in the output SAR images.

The explanation presented for the range and azimuth ripples at the end of the prior two-dimensional SAR section of the report was spectral leakage, which is the spreading of energy in the frequency domain due to signals being finite limited in time domain. For instance, the Fourier transform of an infinite time length sine wave is a delta function at the frequency of the sine wave, which is finite in frequency. The width of the delta function is zero and there are no side lobes. In practice, windows are used to time limit the length of the sine wave which causes spectral leakage as shown in the upper left corner of Figure 23 if a rectangular window such as the one in the upper right corner of Figure 23 is used. Because the range reference signal is a finite length rotating phasor as shown in the top half of Figure 13, the Fourier transform contains moderate amounts of spectral leakage as shown in the bottom half of Figure 13, which can be termed low dynamic range. A possible solution for this is to use a smoothed window such a Hann window shown in the bottom half of Figure 23.[10]

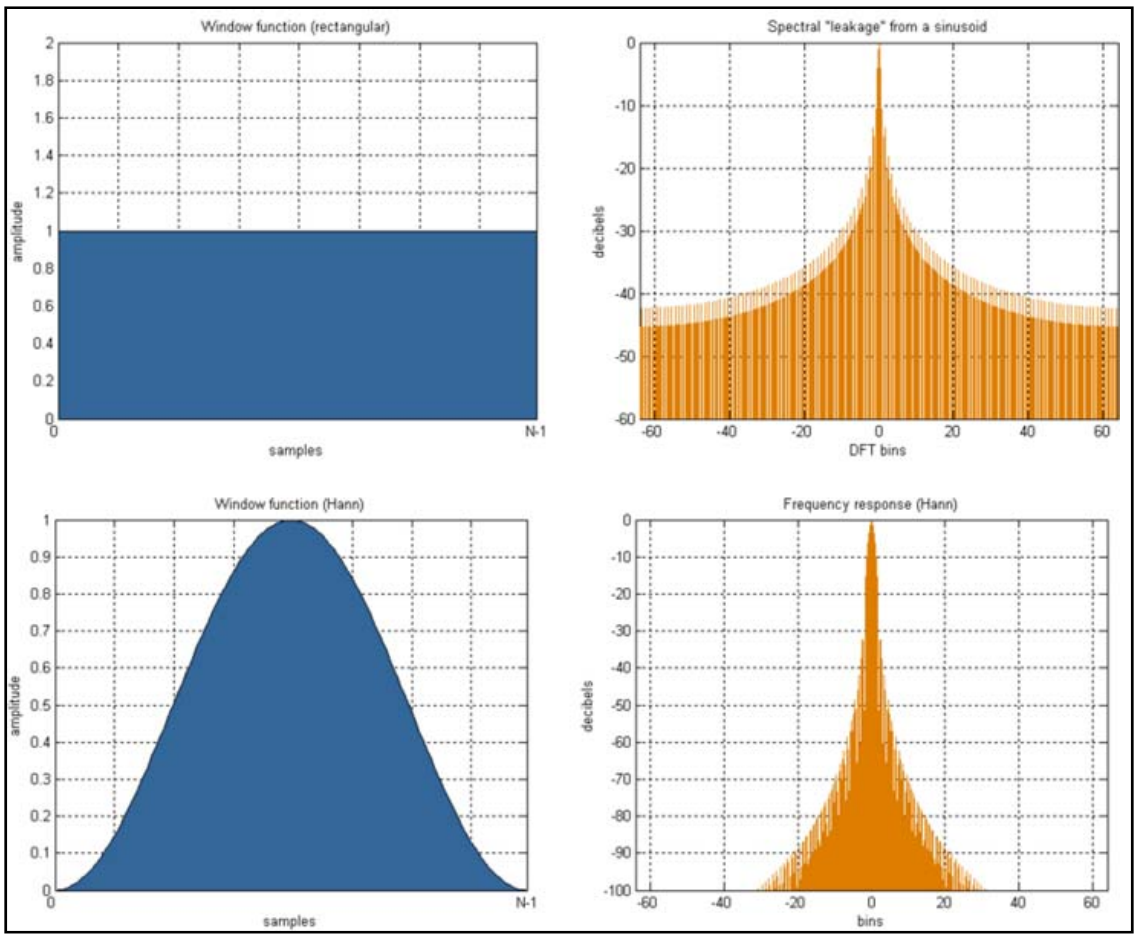

Figure 23: Rectangular Window versus Hann Window[10]
(upper left: Rectangular Window, upper right: Rectangular Window FFT, lower left: Hann Window, lower right: Hann Window FFT)

A Hann window, named after Julius von Hann, provides less spectral leakage and more dynamic range in the Fourier domain as shown in the bottom right corner of Figure 23, but this comes at the cost of resolution as can be seen in the wider center lobe. As an experiment to determine the effectiveness of the Hann window, the range rectangular window, $w_r(t)$, in Equation 1 is changed from a rectangular window into a Hann window, $w_{Hann}(t)$, as show in Equation 16 below. The variable r_{bins} is the number of bins or cells in the range direction, which is 1034 in the simulation and T_p is the transmit pulse duration.[10]

$$s_{tx}{}'(t) = w_{Hann}(t)cos\{2\pi f_o + \pi K_r t^2\} = 0.5\left\{1 - cos\left(\frac{2\pi t(r_{bins})/T_p}{r_{bins}-1}\right)\right\} cos\{2\pi f_o + \pi K_r t^2\} \quad (16)$$

The raw SAR signal space generated from using the Hann window for the range window is shown in Figure 24 below. Differences from the rectangular window raw SAR signal space shown in Figure 11 can be seen in the elliptical nature of the pattern as opposed to rectangular.

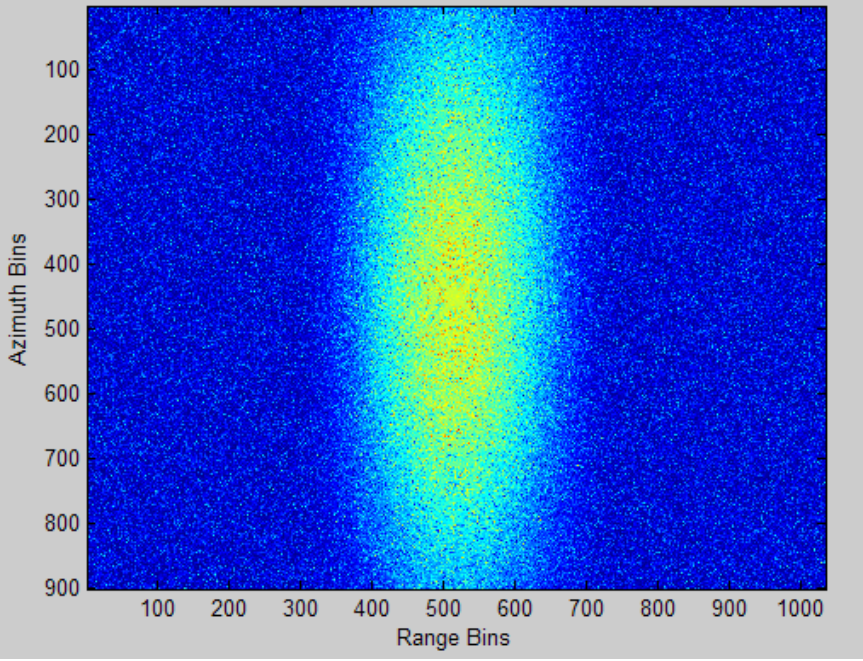

Figure 24: Hann Window SAR Signal Space

The range reference signal was modified from its rectangular form shown in Figure 13 to the Hann window form shown in Figure 25. The upper left magnitude of the range reference signal resembles the Hann window shown in the upper left of Figure 23. As opposed to spectral leakage and ripples shown in the lower left corner of Figure 13 or the magnitude of the Fourier transform of the rectangular window range reference signal, the Hann window range reference signal magnitude in the Fourier transform domain has

no ripples and less spectral leakage in the Fourier transform domain as shown in the lower left corner of Figure 25.

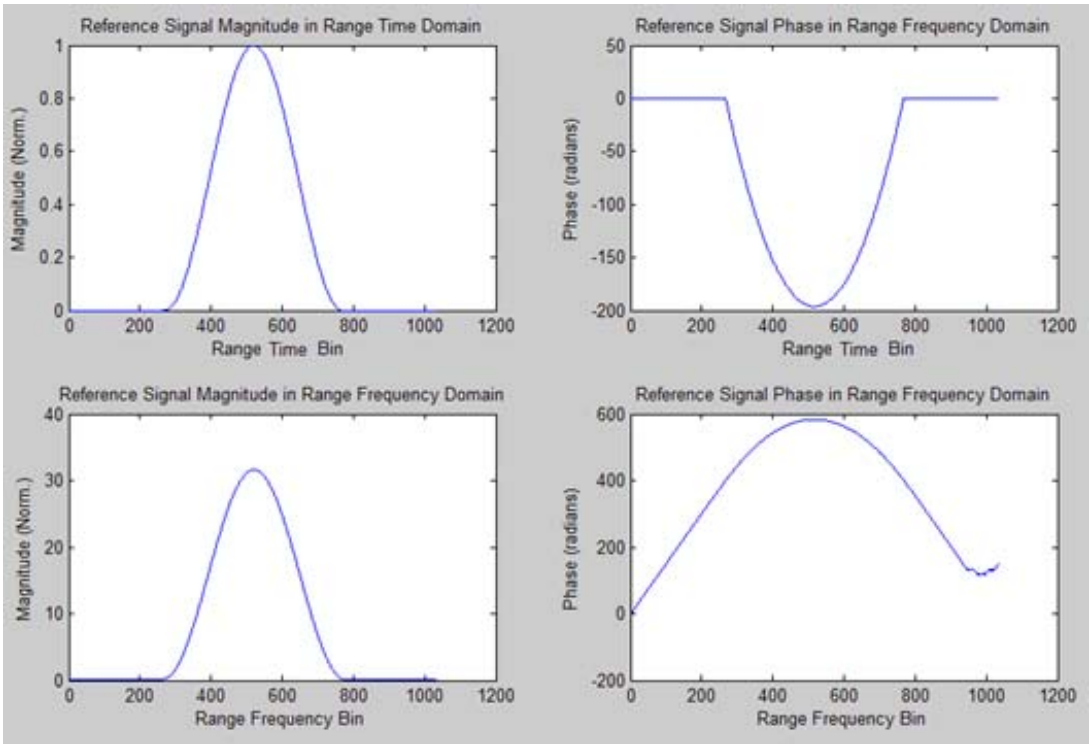

Figure 25: Hann Window SAR Reference Signal

Figure 26 displays the Hann window being used for range compression at the center azimuth bin in comparison to the rectangular window range compression in Figure 15. The effect of the Hann window on the range ripples can be first seen in Figure 27 of the range compressed and RCMC SAR signal. The ripples are of lower magnitude than in Figures 16 and 17 of the corresponding rectangular window images and the ripples are wider.

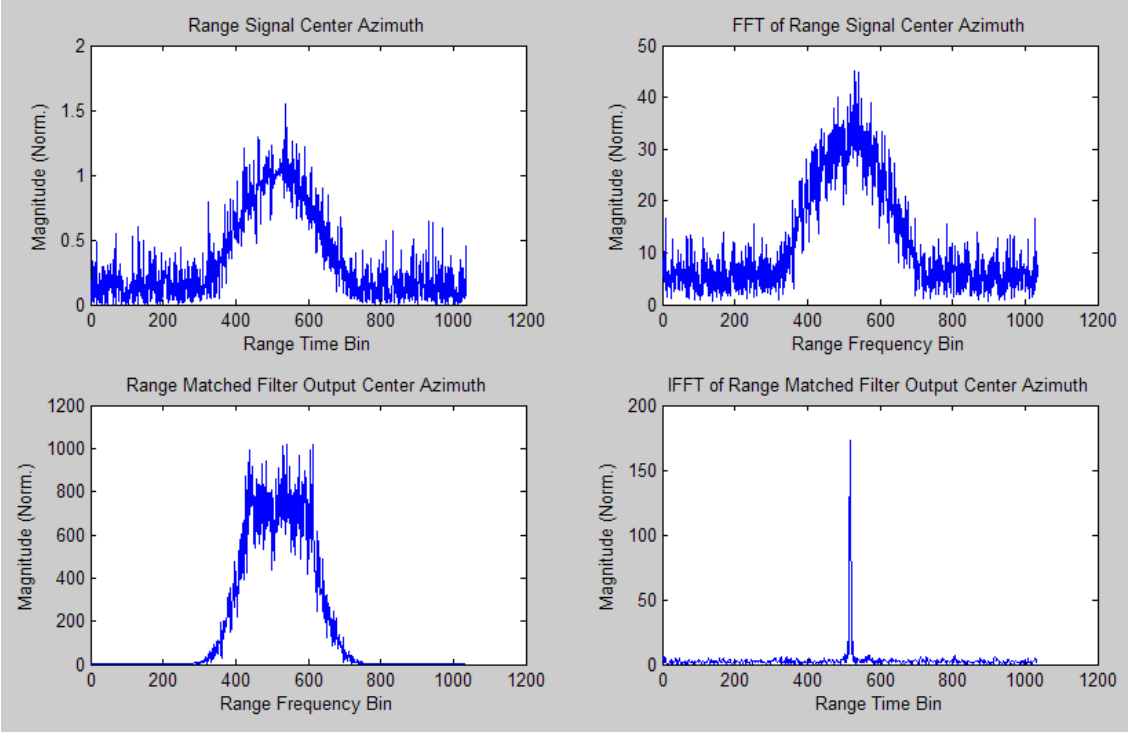

Figure 26: Hann Window Point Target Center Azimuth Range Signal Compression

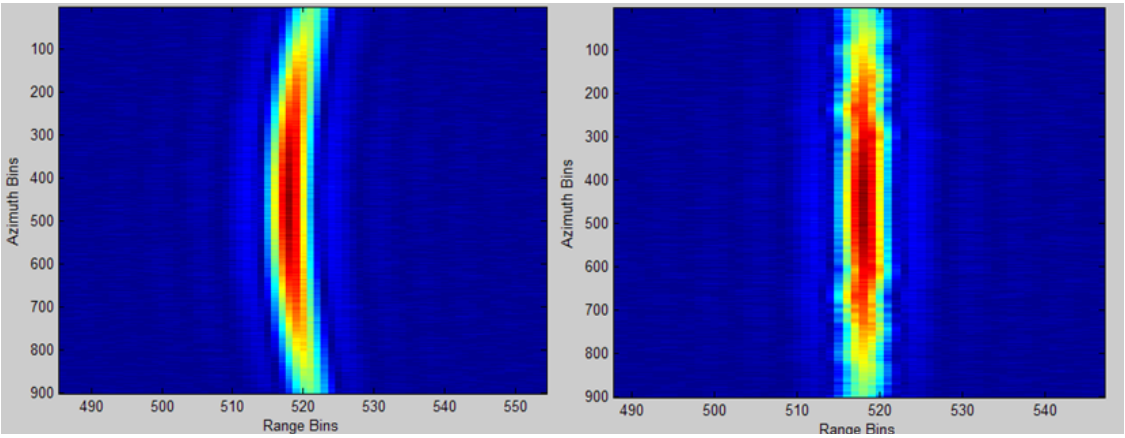

Figure 27: Hann Window Point Target Range Compressed (left), RCMC (right)

The widening and lowering of the magnitude of the range ripples can be more easily seen in a side by side comparison of the final image of a point target using a range rectangular window for the transmitted pulse versus a Hann window as show in Figure 29. The lower magnitude side ripples will allow point targets to be placed closer together in two-dimensional target simulations such as the results shown in Figures 21 and 22 of

the BMP and tank, as the leakage spectral leakage interference will be reduced. However, the reduction of the spectral leakage interference or improvement in dynamic range comes at the cost reduced range resolution which can be seen in the Hann window final image result of Figure 28 in the widening of the resolved point target.

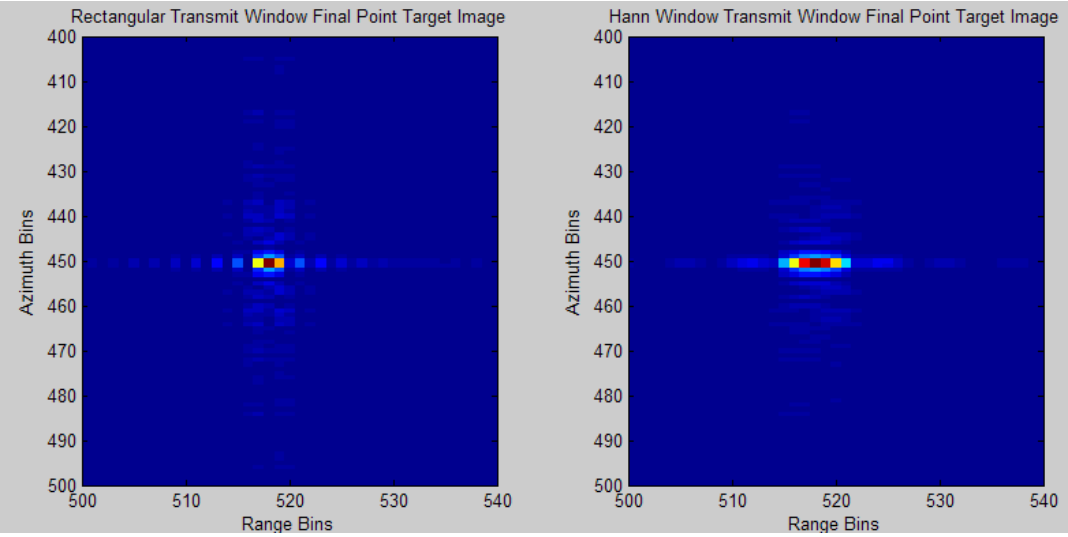

Figure 28: Final Image Rectangular (left) versus Hann (right) Range Transmit Window

A decision was made to classify resolution in the simulation as more important than dynamic range and the Hann window is left commented out in the two-dimensional simulation code in Appendix B, but can be reimplemented if chosen and point target separation can be reduced. Aside from side ripples, the major limitation of the two-dimensional SAR simulation for application towards ATR is the time required for processing of the SAR echo generation Equation 4 which becomes worse for larger and more complex targets. A summary of the accomplishments, limitations, and ways to improve the two-dimensional simulation follow in the next section of this report.

X. Two-Dimensional SAR Imaging Simulation Conclusion

The size of the input images of 16 by 16 pixels limits the distinguishable features that can exist in the target profiles. Larger images would improve the ability of the simulation to be used for ATR. Also, a method for efficiently generating reflections from a larger set a point targets is needed before the simulation can be adapted for SAR using three-dimensional target profiles and SAR holography. Methods for improving the time efficiency of the radar echo generation sequence are porting the code, in whole or in part, to C or a more processing time efficient programming language and/or using multiple streams to compute the echo reflections in parallel as the reflections are independent of each other. To maximize processing power, the simulations were run on a computer with an Intel Quad Core Q6600 CPU at 2.4 GHz and 2 GB of RAM, which is faster than the previous machine the code was run on and has more potential through multicourse processing that can be exploited.

To optimize the SAR echo generation sequence, both methods of porting the code to C and using parallel processing can be fully tested to analyze their effectiveness at optimizing the time efficiency of the radar echo generation sequence. Estimation analysis concluded that parallel processing with four streams will cut down processing time to 33.33% and porting from MATLAB will cut down the processing time similarly. Table 1 shows time estimates for the processing time, T_p, in minutes for several target sizes under no optimization “NO”, parallel processing optimization “PP”, porting to C “C”, and both porting to C and using parallel processing “C & PP”. Target sizes are defined by the pixel width of the target image profile and the corresponding number of point targets. A 128 by 128 pixel image should be able to be processed in one hour and eleven minutes, which is

the time to process an image of a two-dimensional target profile with the resolution of MSTAR images. The current and projected target size and processing time are shown in bold in Table 1.

Target Width (pixels)	**16**	64	**128**
Point Targets (#)	256	4096	16384
T_P NO (min.)	**10.00**	160.00	640.00
T_P PP (min.)	3.33	53.33	213.33
T_P C (min.)	3.33	53.33	213.33
T_P C & PP (min.)	1.11	17.78	**71.11**

Table 1: Processing Time Estimates

XI. Three-Dimensional SAR Imaging Implementation

Given the effectiveness of the implementation of the two-dimensional target SAR simulation and one of the only remaining limitations being processing optimization, the remainder of work done on the SAR project on this thesis was on the development of a three-dimensional implementation of a SAR simulation. The two-dimensional SAR simulation is implemented in a coordinate system constructed of solely azimuth and range distances. The three-dimensional SAR simulation will require a new coordinate system that includes altitude. In the two-dimensional simulation the platform and all point targets were effectively at a constant altitude of zero. In the new simulation the point targets will be assigned a location, $(\boldsymbol{x_m, y_m, z_m})$, in the three dimensional space and the platform will be assigned a constant altitude, $\boldsymbol{Z_o}$, as shown in Figure 29 below. The three-dimensional slant range, $\boldsymbol{R(n)}$ or distance from the platform to the point target, changes as function of azimuth/slow time, $\boldsymbol{n}$, as shown in Equation 17. In the simulation, the variables that are set to define the three-dimensional geometry are the look angle, θ_l, and the range of closest approach, $R_{o\,3D}$. The squint angle, θ_{sq}, is also labeled in Figure 29 as the angle between the slant range and zero Doppler plane.

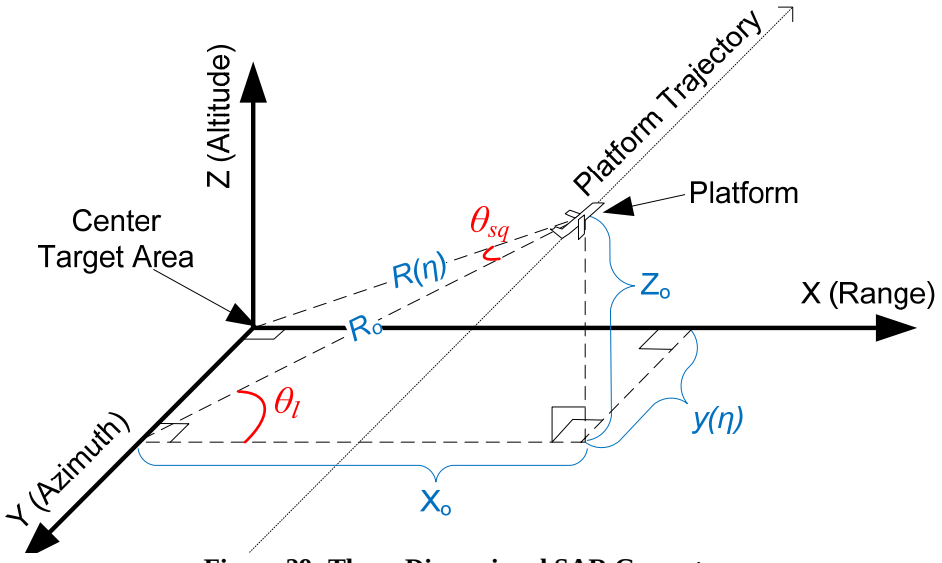

Figure 29: Three-Dimensional SAR Geometry

$$R(n)_{3D} = \sqrt{\boldsymbol{R_{o_{3D}}}^2 + {V_p}^2\, n^2} = \sqrt{(X_o + x_m)^2 + (Z_o + z_m)^2 + {V_p}^2 \left(n + \frac{y_m}{V_p}\right)^2} \qquad (17)$$

Using Equations 18 and 19 derived from the geometry in Figure 29, the altitude and minimum range distance can be derived in the form of solving a system of two equations with two unknowns by using the look angle and minimum range distance defined at the start of the simulation. For the simulation, the range of closest approach, $R_{o_{3D}}$, will be 20 km, as it was the minimum range distance in the two-dimensional SAR simulation. The look angle can range from 0° to 90°, where a 0° look angle defines the case where the altitude of the platform is zero and be verified with Equation 18. A look angle of 90° defines the case where the platform is flying directly over the center of the target area. By replacing the slant range equation in the two-dimensional shown attached in Appendix B with the three-dimensional slant range Equation 16, the SAR echo generation is modified for three-dimensional targets. The next part of the simulation modified for three-dimensionality after the geometry and SAR echo generation was the target import feature.

$$R_{o\,3D} = \sqrt{{X_o}^2 + {Z_o}^2} \tag{18}$$

$$\tan(\theta_l) = Z_o / X_o \tag{19}$$

The target import feature used to import a single grayscale target profile image and set it as the only altitude level. For the three-dimensional simulation, a series of profile images can be imported with each level representing an altitude level. For example, a simple image set named pole3 is shown in Figure 30. Pole3 consists of 3 images "pole1.gif", "pole.gif", and "pole3.gif" which correspond to range and azimuth point target profile layers at altitude levels of 0, 1, and 2 meters. Geometrically, Pole3 is made of a four point target square base, with two more point target rising in line above one of the corners of the base as shown in Figure 31.

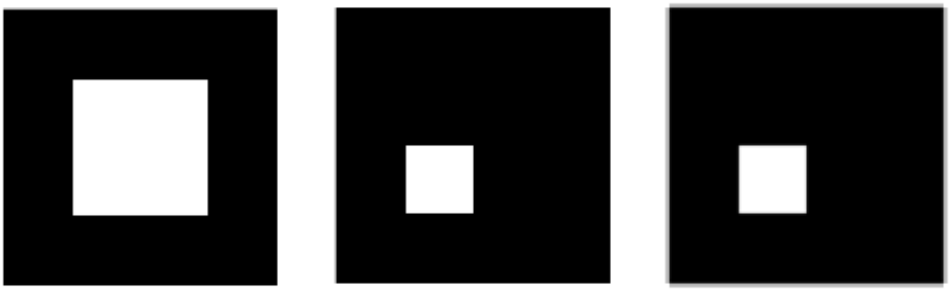

Figure 30: Pole3 Target Profile
(left: "Pole1.gif", middle: "Pole2.gif", right: "Pole3.gif"**)**

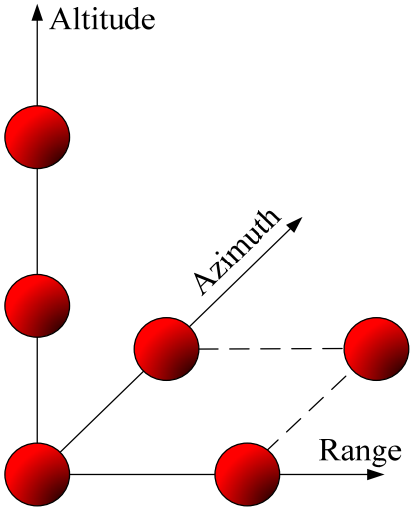

Figure 31: Pole3 Target Geometry

The last modification needed to adapt the two-dimensional SAR simulation into a three-dimensional SAR simulation is to the RDA. This is done by incorporating the constant altitude parameter, $\mathbf{Z_o}$, in the range reference signal, azimuth reference signal, and range cell migration correction equations. This is done in the code by replacing the minimum range distance, $\mathbf{X_o}$, with the three-dimensional range of closest approach, $\mathbf{R_{o3D}}$.

The three-dimensional SAR MATLAB file is attached in Appendix C. Image profiles of the form [base name][number of levels] can be imported such as pole3 which imports images "pole1.gif", "pole2.gif" and "pole3.gif". Along with look angle and range of closest approach, a target rotation angle can be specified which rotates the target counter clockwise on the azimuth/range plane.

For an example of the three-dimensional SAR simulation the following images are the results of a simulation of the pole3 target profile shown in Figure 32 with a 20 km range of closest approach, 0° rotation, and 45° look angle. These parameters result with the platform at an altitude of 14.1421 km with a minimum range distance of 14.1421 km. Figure 32 displays the raw SAR signal space for three-dimensional pole3 target set and Figure 33 displays the range compressed SAR image of the pole3 target.

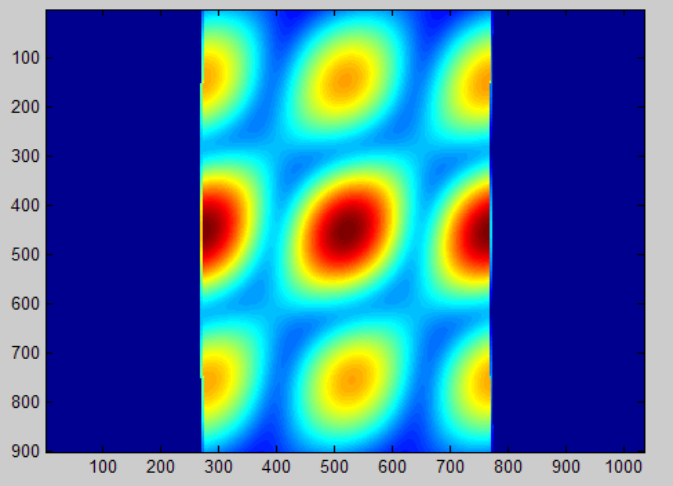

Figure 32: Pole3 Raw SAR Signal Space (45°)

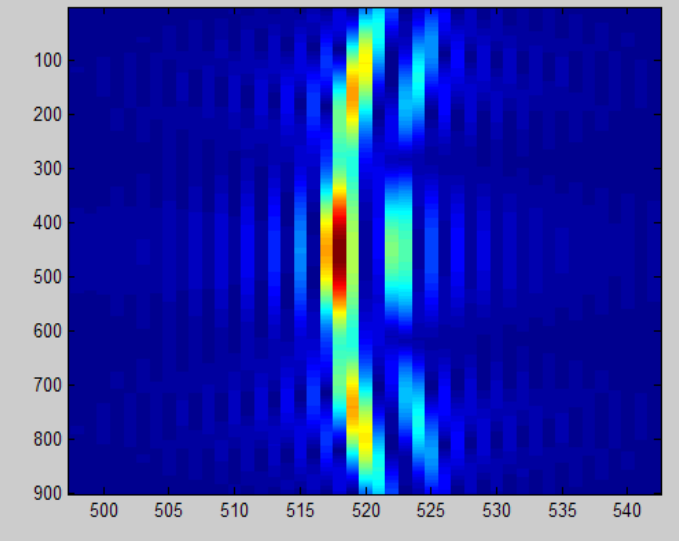

Figure 33: 3D Pole Range Compressed Image (θ_i=45°)

To compare the effectiveness of the three-dimensional SAR simulation over varying look angles, Figure 34 displays the final processed SAR image for look angles of 0°, 20°, 45° and 65° on the upper left, upper right, lower left, and lower right respectively.

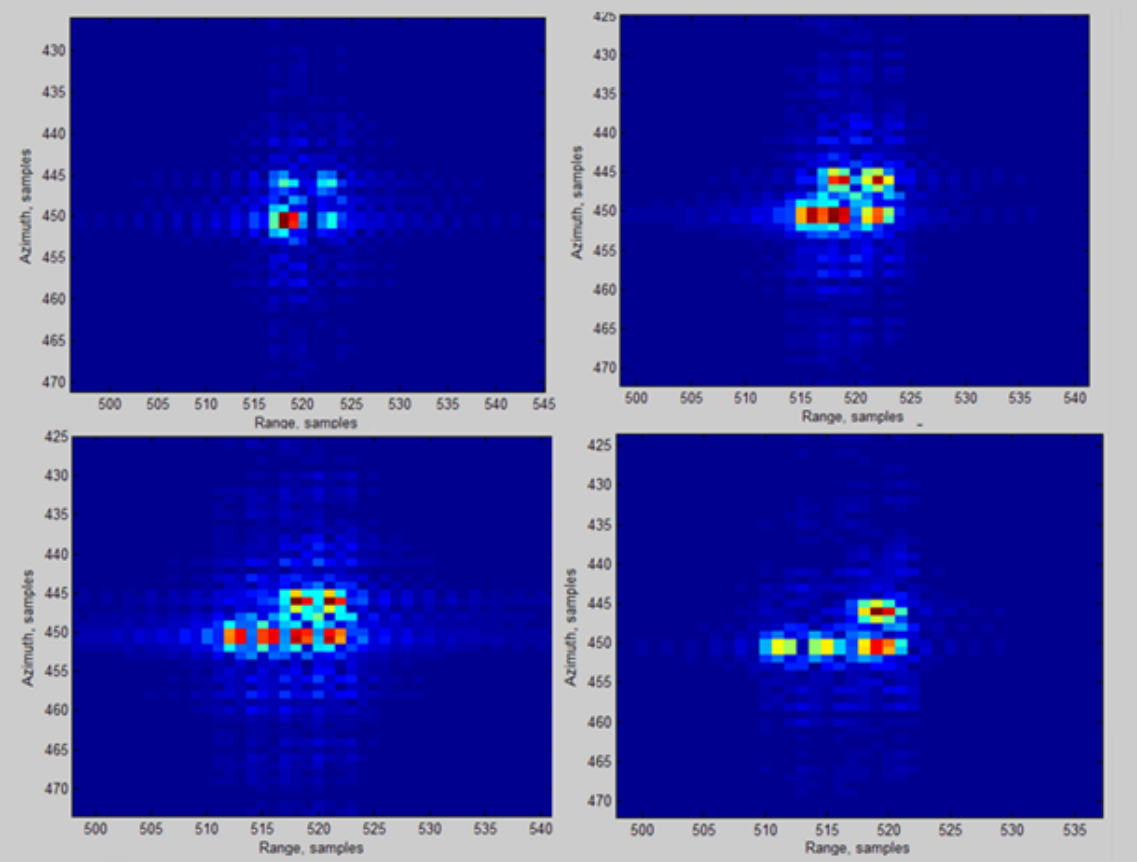

Figure 34: 3D Pole Final Images
(upper left: $\boldsymbol{\theta_l}$=0°, upper right: $\boldsymbol{\theta_l}$=20°, lower left: $\boldsymbol{\theta_l}$=45°, lower right: $\boldsymbol{\theta_l}$=60°)

From Figure 34, it can be seen that at a look angle of 0°**,** it appears as if the point targets are on top of each other. There is no warping in the azimuth direction over the varying look angles. However, as the look angle increases from 0° to 45° point target locations in the final image spread apart in the range direction with respect to their altitude. From 45° to 60°, point targets at the same altitude warp closer together as can be seen in the bottom right corner of Figure 34. An explanation for this is that as the altitude beam propagation dominates over the range beam propagation, the resolution in the range direction worsens. The range resolution in Equation 3 more accurately describes the resolution along beam propagation. In this case the range resolution, $\rho_{r_{3D}}$, is found by multiplying the resolution in Equation 3 by a function of the look angle as shown in

Equation 20 below. The altitude resolution, ρ_z, is related to the range resolution as shown in Equation 21.

$$\rho_{r_{3D}} \approx \frac{c}{2\,B_o}\cos\left(\theta_l\right) \tag{20}$$

$$\rho_z \approx \frac{c}{2\,B_o} - \rho_{r_{3D}} \approx \frac{c}{2\,B_o}\sin\left(\theta_l\right) \tag{21}$$

The main limitation of the three-dimensional SAR MATLAB simulation aside from the limitations on target size from practical processing time requirements is that all point targets will reflect at all times their reflectivity value regardless of line of sight visibility. This makes simulating a complex target for ATR impossible as features such as shadows would not appear and the resulting image would not be distinguishable. In order to better understand how line of sight selective reflectivity will work, the next section of the report outlines a procedure for simulating a three-dimensional cube and hard coding which point targets will reflect the radar signal depending line of sight to the platform.

XII. Cube Target Simulation

The cube target MATLAB SAR imaging simulation attached at the end of this report as Appendix D generates the SAR echoes from the line-of-sight point targets based on the geometry of a cube and the position of the platform. The cube size is composed of point targets in a 3x3x3 arrangement and over two different stages of point targets reflecting, 19 point targets will reflect at any given time as shown in Figure 35 with the altitude, z, and range, x, directions labeled for orientation. This is done so only the point targets on the visible faces of the cube reflect. The cube target profile is composed of two stages of three levels of 9 pixel gray scale images. The first stage is the view of the cube as the platform approaches the cube and the second view of cube is as the platform is leaving cube. At both stages three sides of the cube are visible and the stages switch halfway through the flight. The flight duration is still 3 seconds as with the two-dimensional simulation and the PRF remains 300 Hz.

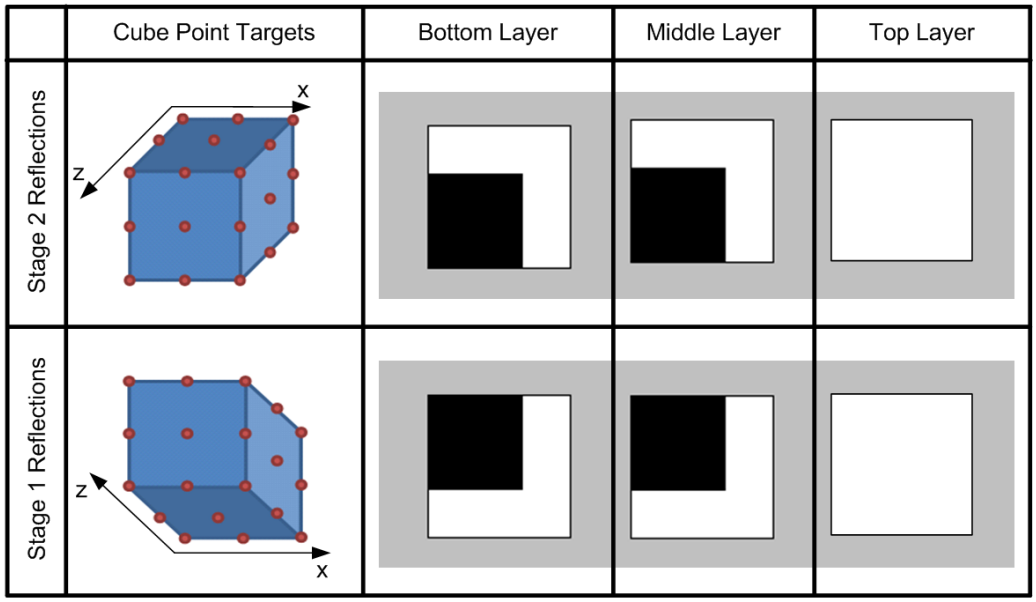

Figure 35: 3x3x3 Cube Simulation Line-of-Sight Based Point Target Reflection Stages

The result of this simulation for a look angle, θ_l, of 45° and target rotation on the range/azimuth plane, θ_r, of 0° is shown in Figure 36. The final processed image shows 15 point targets which represent the top and face of the cube nearest the platform. This is the image of the cube the platform would see at the midpoint in the flight. The side and top faces of the cube in the final image are the two faces of the cube both stages share. In order to better prove the effectiveness of the cube simulation the option of rotating one the range/azimuth plane the cube was implemented.

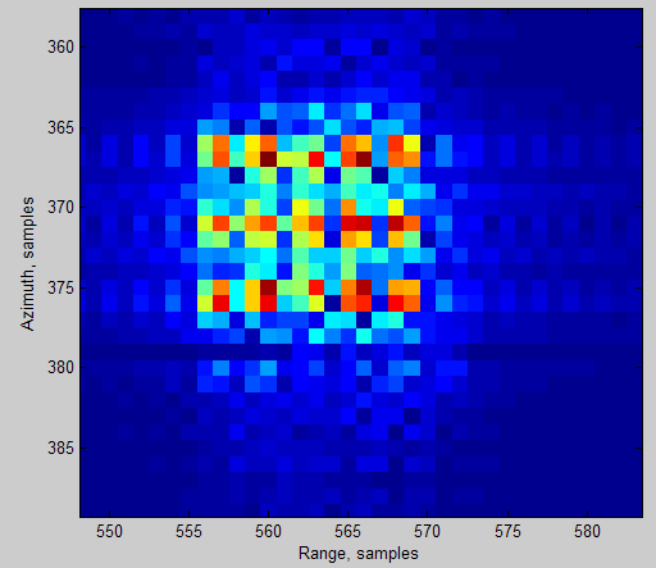

Figure 36: Cube Simulation Final Image ($\theta_l = 45°, \theta_r = 0°$)

In the situation of a 45° rotation of the cube as shown in Figure 37 below only one stage is needed. With a flight duration of 3 seconds and a platform velocity of 200 m/s, the maximum azimuth distance from the target is 300 meters. As range of closest approach is 14.1421 km, the corresponding maximum squint angle according to Equation 11 is 1.215° and in order for more than one stage to be needed or more than the three faces of the cube shown in Figure 37 below to be seen, the maximum squint angle would have to be greater than 45°.

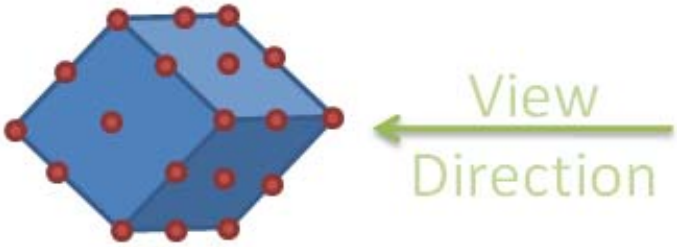

Figure 37: 45° Rotated Cube Point Target Three-Dimensional Profile

The MATLAB algorithm used to rotate the target image profile does not rotate the position of the exact 15 point targets shown in the target representation in Figure 37. Instead, the MATALB function imrotate() is used which performs a counter-clockwise rotation on the image and uses nearest neighbor interpolation. In the 45° rotated cube simulation, the target profile of stage 1 in Figure 35 has this operation performed on it and the output image target profile set is shown in Figure 38. There are now 29 point targets of varying intensity on the three layers which make up the corner view of the cube shown in Figure 37.

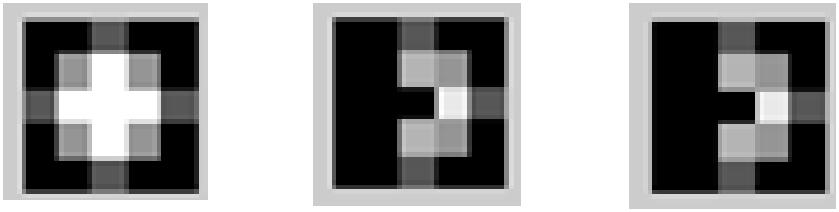

Figure 38: 45° Rotated Cube Image Target Profiles
(left: bottom layer, middle: middle layer, right: top layer)

The final processed image from the 45° rotated cube simulation is shown in Figure 39. The point target position and intensity correspond the import target profile images of Figure 38. Side ripples still remain in the final processed image. The three-dimensional nature of the cube is more easily distinguishable in this image than that of the side view of the 0° rotated cube in Figure 36. However, due to the small size of the

target, a more effective means of portraying the accuracy of the three-dimensional target SAR imaging is needed.

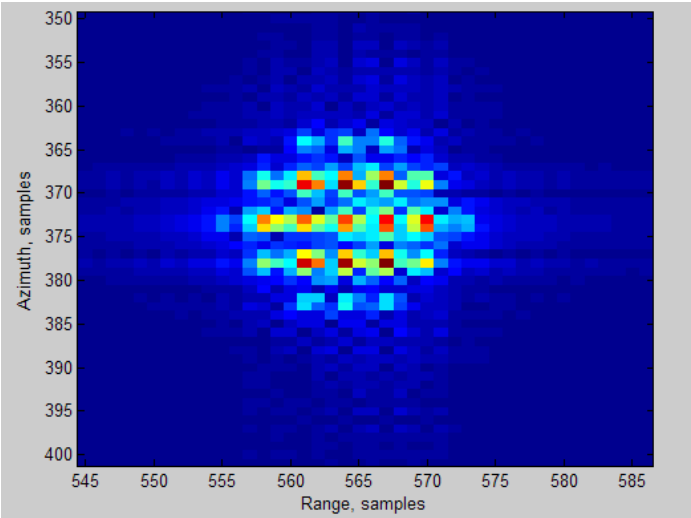

Figure 39: Cube Simulation Final Image ($\theta_l = 45°, \theta_r = 45°$)

To verify the accuracy of the three-dimensional target imaging SAR simulation, the above 45° rotated cube simulation is performed again, but in three stages by splitting the rotated cube target profile of Figure 37 into the top, side and front components shown in Figure 10. Each section is assigned its own three levels of the type shown in Figure 38 and they are rotated 45° counter-clockwise and put through the cube MATLAB simulation in Appendix D.

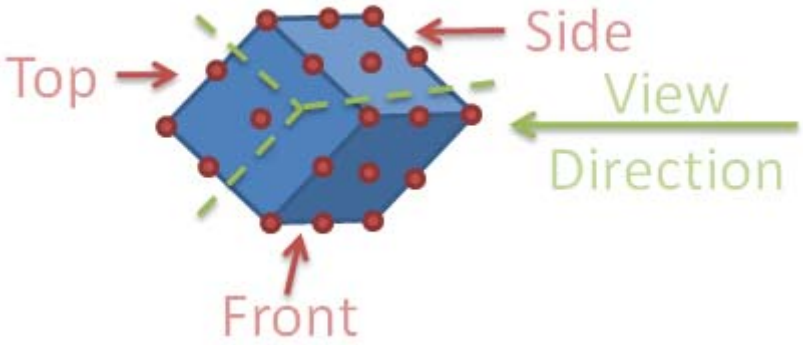

Figure 40: 45° Rotated Cube Point Target Sections (Top, Front and Side)

The output of this simulation is shown in Figure 41 for the top, side and front parts of the cube corresponding to the left, middle right sections of the figure. Each composite image contains its own side ripples but closely resembles the sections of the final image in Figure 39 which they should.

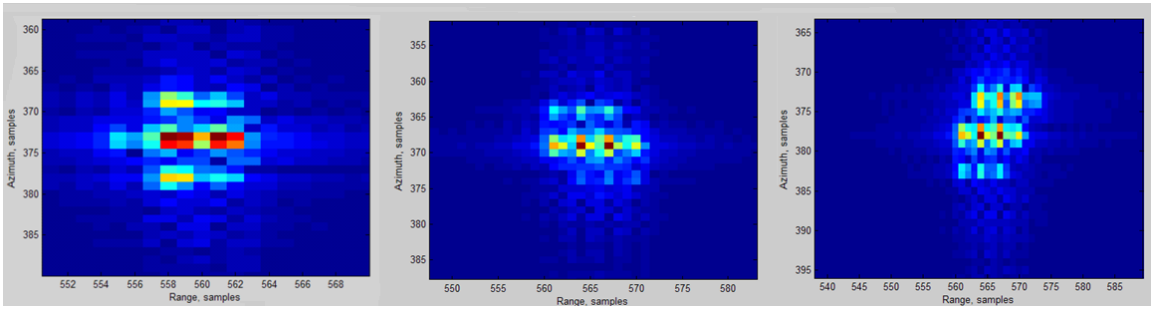

Figure 41: 45° Rotated Cube Section Final Images (left: top, middle: side, right: front)

Using Photoshop, the three different sections are cut out and scaled to the same azimuth and range cell spacing and superimposed over each other. Once this is done, they form the composite image shown in the right side of Figure 42. This matches up accurately to the original 45° rotated cube final image shown on the left side of Figure 42 which demonstrates the effectiveness of the cube SAR imaging simulation in Appendix D.

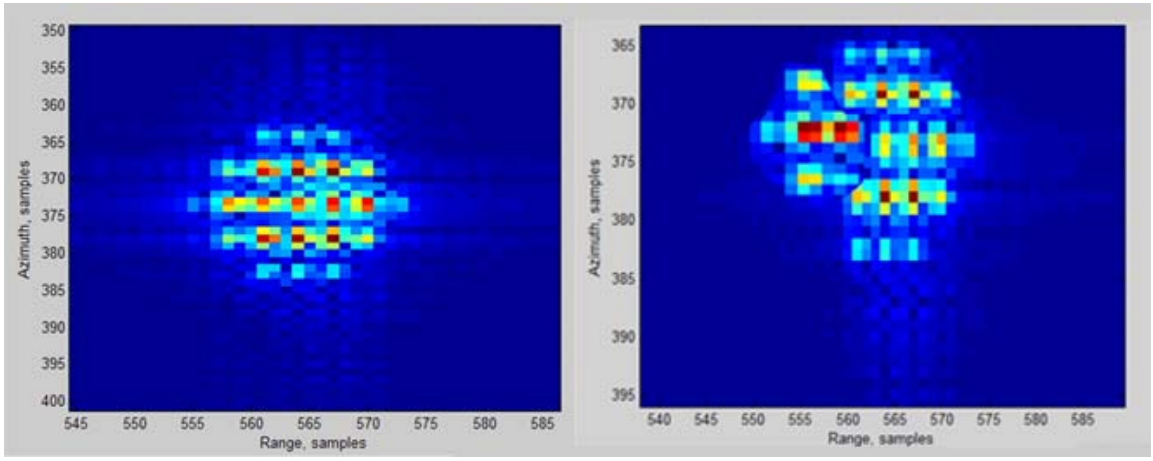

Figure 42: 45° Rotated Cube Final Image Comparison (left: full cube, right: 3 Section Composite)

XIII. Three-Dimensional SAR Imaging Simulation Conclusion

The three dimensional-SAR imaging simulation is an important step towards the goal of making a process with which realistic SAR images can be generated for use in ATR testing and development. The modifications to the two-dimensional simulation including the geometry, echo generation and RDA were effective in adapting the simulation to an extra dimension. The main effect on simulation result is the modification to range resolution as shown in Equation 19 as a function of the look angle and is inversely related to the altitude resolution in Equation 20. Limitations of the three-dimensional SAR imaging simulation include target size due to the processing time constraints and line-of-sight point target reflection decisions which restricts the ease which new target profiles can be simulated. The processing time limitation is a carryover from the two-dimensional SAR imaging simulation, which was caused by the processing requirements of the SAR echo generation sequence. To overcome the line-of-sight point target reflection decision problem, the cube simulation was conducted and a cube was able to be accurately imaged with the three-dimensional simulation from different rotations and look angles. For more complex targets, a better method of constructing two-dimensional targets and deciding line of sight reflectivity will be needed. The SAR imaging of the cube was confirmed operational by imaging different portions of the cube separately and combining them into a composite image to compare the original image as shown in Figure 42. Being able to image three-dimensional targets from different look angles and rotations is a principle requirement of a SAR imaging simulation to be implemented with SAR ATR algorithms or three-dimensional SAR Holography and has been accomplished.

XIV. Future Work

There are several areas of future work for continuations on this project. An important area of future work is the optimization of the SAR echo generation sequence so that more complex targets can be simulated and imaged. As discussed at the end of the two-dimensional SAR imaging results and conclusion section, the processing of the SAR echo generation sequence can be optimized through either porting the section of the code to C, a more efficient coding language than MATLAB, or/and using multi-core processing. Multithreading, which can make use of a multi-core processor on a single computer or use network processing of multiple computers for even increased processing efficiency so that more point targets can be simulated and imaged. A better method of three-dimensional target profile image generation can be developed such the use of a three-dimensional CAD program. Applications of the three-dimensional SAR code include implementing it with the SAR holography program coded by John Hupton or SAR ATR algorithms. Other three-dimensional imaging setup geometries can be implemented besides stripmap SAR. Circular SAR (CSAR) offers improved ATR efficiency over three-dimensional stripmap SAR imaging by having the platform spotlight scan of the target area over a 360° rotation. An elevation circular SAR (E-CSAR) could also be developed, which acquires synthetic aperture data linearly over the altitude and circularly in the range and azimuth domains. An E-CSAR simulation can generate a three-dimensional reconstruction of the target, the goal of the SAR holography simulation.[9] One last area of work for this simulation, either two-dimensional or three-dimensional, would be the reimplementation of the SAR tracking algorithm removed from the two-dimensional SAR imaging simulation used at the start of this project which

was developed by Brian Zaharris. The motion tracking can be accomplished with a Kalman filter or other dynamic system filter through processing of the raw SAR signal space.

List of References

1. Ambardar, Ashok. Analog and Digital Signal Processing: Second Edition. Pacific Grove, CA: Brooks/Cole Publishing Company, 1999.

2. Butler, M.J.A., et al. The application of remote sensing technology to marine fisheries: an introductory manual. *Food and Agirculture Organization Coporate Document Repository.* [Online] 1988. [Cited: April 5, 2009.] http://www.fao.org/docrep/003/t0355e/T0355E05.HTM. ISBN: 9251026947.

3. Cumming, Ian G and Wong, Frank H. *Digital Processing of Synthetic Aperture Radar Data: Algorithms and Implementations.* Norwood, MA : Artech House, Inc., 2005.

4. Glossary of remote sensing terms. *Natural Resources Canada.* [Online] Canada Centre for Remote Sensing, November 21, 2005. [Cited: April 5, 2009.] http://www.ccrs.nrcan.gc.ca/glossary/index_e.php?id=2284.

5. Goodman, Joseph. *Introduction to Fourier Optics: Third Edition*. Englewood, CO: Roberts & Company Publishers, 2005.

6. Mason, Paul Ryan. *MATLAB Simulation of Two-Dimensional SAR Imaging By Range Doppler Algorithm.* San Luis Obispo, CA : California Polytechnic State University San Luis Obispo California, 2007. Master's Thesis.

7. Matched Filtering. *Wikipedia*. [Online] Wikimedia Foundation. September 1, 2008. [Cited: November 3, 2008.] http://en.wikipedia.org/wiki/Matched_filter.

8. Oleary, Ellen. SEASAT 1978. *Imaging Radar @ Southport.* [Online] Jet Propulsion Laboratories, February 10, 1998. [Cited: April 28, 2009.] http://southport.jpl.nasa.gov/scienceapps/seasat.html.

9. Soumekh, Mehrdad. *Synthetic Aperture Radar Signal Processing with MATLAB Algorithms.* New York, NY: Wiley & Sons, Inc., 1999.

10. Synthetic Aperture Radar. *Wikipedia*. [Online] *Wikimedia Foundation*. May 20, 2008. [Cited: November 6, 2008.] http://en.wikipedia.org/wiki/Synthetic_aperture_radar.

11. Window Function. *Wikipedia*. [Online] Wikimedia Foundation. May 1, 2009. [Cited: May 9, 2009.] http://en.wikipedia.org/wiki/Window_function.

12. Zaharris, Brian. *Two-Dimensional Synthetic Aperture Radar Imaging and Moving Target Tracking Using the Range Doppler Algorithm Simulated in MATLAB.* San Luis Obispo, CA : California Polytechnic State University San Luis Obispo California, 2006. Master's Thesis.

Appendix A: Two-Dimensional SAR MATLAB Simulation Parameters

Symbol	Parameter Name	Value
PRF	Pulse Repetition Frequency	300 Hz
dur	Duration	3 seconds
V_p	Platform Velocity	200 m/s
f_o	Carrier Frequency	4.5 GHz
La	Antenna Actual Length	2 m
X_c	Minimum Range Distance	20 km
X0	Half Target Area Width	200 m
Tp	Chirp Pulse Duration	2.5 μs
B0	Baseband BW (+/-)	100 MHz
σ_{noise}	AWGN Standard Deviation	0.2 (normalized/unitless)
a_{bins}	Azimuth Bins	900
r_{bins}	Range Bins	1034

Appendix B: 2D SAR Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Simulation of 2-D SAR Imaging using Range Doppler Algorithm
%
%   Matthew Schlutz
%   Cal Poly, San Luis Obispo
%   SAR ATR Project
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Simulation2D()
clear

% INITIALIZATION
% Main SAR Parameters
PRF=300; % Pulse Repetition Frequency (Hz)
dur=3; % Time of Flight (sec), PRF*dur = received echoes
vp=200; % Velocity of platform
fo=4.5e9; % Carrier frequency (4.5GHz)
La=2; % Antenna length actual
Xc=20000; % Range distance to center of target area
X0=200; % Half Target Area Width (Target is located within [Xc-
X0,Xc+X0])
Tp=.25e-5; % Chirp Pulse Duration
B0=100e6; % Baseband bandwidth is plus/minus B0\
target_name='pole1'; %Name of Target Profile Image (GIF Grayscale
Image)

%Noise
noise=0; % Set this flag to add noise to signal
std_dev=.2; % Standard Deviation of Noise

% General Variables
cj=sqrt(-1);
c=3e8; % Propagation speed
ic=1/c; % Propagation frequency
lambda=c/fo; % Wavelength (60cm for fo = 4.5e9)
eta=linspace(0,dur,PRF*dur)'; % Slow Time Array

% Range Parameters
Kr=B0/Tp; % Range Chirp Rate
dt=1/(2*B0); % Time Domain Sampling Interval
Ts=(2*(Xc-X0))/c; % Start time of sampling
Tf=(2*(Xc+X0))/c+Tp; % End time of sampling

% Azimuth Parameters
Ka=(2*vp^2)./(lambda*(Xc)); % Linear Azimuth FM rate

% Measurement Parameters
rbins=2*ceil((.5*(Tf-Ts))/dt); % Number of time (Range) samples
t=Ts+(0:rbins-1)*dt; % Time array for data acquisition
s=zeros(PRF*dur,rbins); % Echoed signal array

% Target Initialization
target=imread(target_name,'gif'); %Select Input Target Profile
[M N]=size(target); ntarget=M*N;
```

```
tnum=1; xn=zeros(ntarget,1); yn=xn; Fn=xn; % Target Intialization
Variables
for m=1:M
    for n=1:N
        xn(tnum)=(n-N/2);
        yn(tnum)=(M/2-m+1);
        Fn(tnum)=double(target(m,n))/255;
        tnum=tnum+1;
    end
end
stretch=3;
xn=xn*stretch; yn=yn*stretch; %Stretch out Target Profile

% GENERATE ECHOES
for j=1:(PRF*dur);
    for i=1:ntarget;
        wa=sinc(La*(atan(vp*(eta(j)-dur/2+yn(i)/vp)/Xc))/lambda).^2;
        R=sqrt((Xc+xn(i))^2+vp^2*((eta(j)-dur/2+yn(i)/vp)^2));
        td=t-2*R/c;
        if noise==1
            s(j,:)=s(j,:)+std_dev*randn(size(s(j,:)))...
                +Fn(i)*wa*exp(-cj*(4*pi*fo*ic*R)+cj*pi*Kr*...
                (td.^2-td*Tp)).*(td >= 0 & td <= Tp);
        else
            s(j,:)=s(j,:)+Fn(i)*wa*exp(-cj*(4*pi*fo*ic*R)+cj*pi*Kr*...
                (td.^2-td*Tp)).*(td >= 0 & td <= Tp);
        end
%         if noise==1
%           s(j,:)=s(j,:)+std_dev*randn(size(s(j,:)))...
%                 +Fn(i)*wa*exp(-cj*(4*pi*fo*ic*R)+cj*pi*Kr*...
%                 (td.^2-td*Tp)).*(0.5*(1-
cos((2*pi()*td*rbins/Tp)/(rbins-1)))).*(td >= 0 & td <= Tp);
%         else
%             s(j,:)=s(j,:)+Fn(i)*wa*exp(-
cj*(4*pi*fo*ic*R)+cj*pi*Kr*...
%                 (td.^2-td*Tp)).*(0.5*(1-
cos((2*pi()*td*rbins/Tp)/(rbins-1)))).*(td >= 0 & td <= Tp);
%         end
    end
    if mod(j,50)==0
        display(j/9) % Echo Gen. Percent Complete
    end
end

% RANGE DOPLER ALGORITHM (RDA)
% Range Reference Signal
td0=t-2*(Xc/c);
pha20=pi*Kr*((td0.^2)-td0*Tp);
s0=exp(cj*pha20).*(td0 >= 0 & td0 <= Tp);
% s0=exp(cj*pha20).*(0.5*(1-cos((2*pi()*td*rbins/Tp)/(rbins-1)))).*(td0
>= 0 & td0 <= Tp);
fs0=fty(s0); % Reference Signal in frequency domain

% Power equalization
amp_max=1/sqrt(2); % Maximum amplitude for equalization
afsb0=abs(fs0);
P_max=max(afsb0);
```

```
I=find(afsb0 >= amp_max*P_max);
fs0(I)=((amp_max*(P_max^2)*ones(1,length(I)))./afsb0(I))...
    .*exp(cj*angle(fs0(I)));
deltaR=(2*lambda^2*(Xc).*(Ka*(dur*0.5-eta)).^2)/(8*vp^2); % RCM
cells=round(deltaR/.56); % .56 meters/cell in range direction
rcm_max=9; %maximum range cell migration
fs=zeros(PRF*dur,rbins); fsm=fs; fsmb=fs; smb=fs; fsac=fs; sac=fs;

% Range Compression
for k=1:(PRF*dur);
    fs(k,:)=fty(s(k,:));    % Range FFT
    fsm(k,:)=fs(k,:).*conj(fs0);% Range Matched Filtering
    smb(k,:)=ifty(fsm(k,:));    % Range IFFT
end;

% Azimuth Reference Signal
smb0=exp(cj*pi*Ka.*eta.*(2*eta(PRF*dur/2+1)-eta));
fsmb0=ftx(smb0); % Azimuth Matched Filter Spectrum
for l=1:rbins;
    fsmb(:,l)=ftx(smb(:,l)); % Azimuth FFT
end;

% Range Cell Migration Correction (RCMC)
fsmb2=fsmb;
for k=1:dur*PRF;
    for m=1:rbins-rcm_max
        fsmb2(k,m)=fsmb(k,m+cells(k));
    end
end;
for l=1:rbins;
    fsac(:,l)=iftx(fsmb2(:,l)); % Azimuth IFFT
end;

% Azimuth Compression
for l=1:rbins;
    fsac(:,l)=fsmb2(:,l).*conj(fsmb0); % Azimuth Matched Filtering
    sac(:,l)=iftx(fsac(:,l)); % Azimuth IFFT / Final Target Image
end;

% Plot Final Results
figure(1), imagesc(abs(sac))
xlabel('Range, samples'), ylabel('Azimuth, samples')
```

Appendix C: 3D SAR Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Simulation of 3-D SAR Imaging using Range Doppler Algorithm
%
%   Matthew Schlutz
%   Cal Poly, San Luis Obispo
%   SAR ATR Project
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function Simulation3D()
clear

% INITIALIZATION
% Main SAR Parameters
PRF=300; % Pulse Repetition Frequency (Hz)
dur=3; % Time of Flight (sec), PRF*dur = received echoes
vp=200; % Velocity of platform
fc=4.5e9; % Carrier frequency (4.5GHz)
La=2; % Antenna length actual
theta=45; %Look Angle in degrees
rotation=0;
Ro=20e3; % Minimum Distance to Center of Target
Xc=Ro*cos(theta*pi/180); % Range distance to center of target area
Zc=Ro*sin(theta*pi/180); % Altitude of platform
X0=200; % Half Target Area Width (Target is located within [Xc-
X0,Xc+X0])
Tp=.25e-5; % Chirp Pulse Duration
B0=100e6; % Baseband bandwidth is plus/minus B0
targets='pole3'; %Select Input Target Profile

%Noise
noise=0; % Additive White Gaussian Noise Flag (1=noise on, 0=noise off)
std_dev=.06; % Standard Deviation of Noise

% General Variables
cj=sqrt(-1);
c=3e8; % Propagation speed
ic=1/c; % Propagation frequency
lambda=c/fc; % Wavelength (60cm for fc = 4.5e9)
eta=linspace(0,dur,PRF*dur)'; % Slow Time Array

% Range Parameters
Kr=B0/Tp; % Range Chirp FM Rate
dt=1/(2*B0); % Time Domain Sampling Interval
Ts=(2*(sqrt(Xc^2+Zc^2)-X0))/c; % Start time of sampling
Tf=(2*(sqrt(Xc^2+Zc^2)+X0))/c+Tp; % End time of sampling

% Azimuth Parameters
Ka=(2*vp^2)./(lambda*(sqrt(Xc^2+Zc^2))); % Linear Azimuth FM rate

% Measurement Parameters
rbins=2*ceil((.5*(Tf-Ts))/dt); % Number of time (Range) samples
t=Ts+(0:rbins-1)*dt; % Time array for data acquisition
s=zeros(PRF*dur,rbins); % Echoed signal array
```

```
% Target Initialization
target=imrotate(imread([targets(1:length(targets)-1) '1'],'gif')...
    ,rotation,'bilinear'); %samples target
O=sscanf(targets(length(targets)),'%f'); %number of layers
[M N]=size(target); ntarget=M*N*O; %number of point targets in 3D Space
tnum=1; xn=zeros(ntarget,1); yn=xn; Fn=xn; zn=xn;
for i=1:O %loop through target layers
    target=imrotate(imread([targets(1:length(targets)-1)
num2str(i)],...
        'gif'),rotation,'bilinear');
    for m=1:M
        for n=1:N
            xn(tnum)=(n-N/2);
            yn(tnum)=(M/2-m+1);
            zn(tnum)=i-1;
            Fn(tnum)=double(target(m,n))/255;
            tnum=tnum+1;
        end
    end
end
stretch=3;
xn=xn*stretch; yn=yn*stretch; zn=zn*stretch; %Stretch out Target
Profile

% RAW SAR SIGNAL GENERATION
for j=1:(PRF*dur);
    for i=1:ntarget;
            wa=sinc(La*(atan(vp*(eta(j)-
dur/2+yn(i)/vp)/sqrt(Xc^2+Zc^2)))/lambda).^2;
            R=sqrt((Xc+xn(i))^2+(Zc-zn(i))^2+vp^2*(eta(j)...
                -dur/2+yn(i)/vp).^2);
            td=t-2*R/c;
            if noise==1
                s(j,:)=s(j,:)+wgn(1,rbins,std_dev)...
                    +Fn(i)*wa*exp(-cj*(4*pi*(fc*ic)*R)+cj*pi*Kr*...
                    ((td.^2)-td*Tp)).*(td >= 0 & td <= Tp);
            else
                s(j,:)=s(j,:)+Fn(i)*wa*exp(-
cj*(4*pi*(fc*ic)*R)+cj*pi*Kr*...
                    ((td.^2)-td*Tp)).*(td >= 0 & td <= Tp);
            end
    end
    if mod(j,50)==0
        display(j/9) % Echo Gen. Percent Complete
    end
end

% RANGE DOPLER ALGORITHM (RDA)
% Range Reference Signal
td0=t-2*sqrt(Xc^2+Zc^2)/c;
pha20=pi*Kr*((td0.^2)-td0*Tp);
s0=exp(cj*pha20).*(td0 >= 0 & td0 <= Tp);
fs0=fty(s0); % Reference Signal in frequency domain

% Power equalization
amp_max=1/sqrt(2); % Maximum amplitude for equalization
afsb0=abs(fs0);
```

```
P_max=max(afsb0);
I=find(afsb0 >= amp_max*P_max);
fs0(I)=((amp_max*(P_max^2)*ones(1,length(I)))./afsb0(I))...
    .*exp(cj*angle(fs0(I)));
deltaR=(lambda^2*(sqrt(Xc^2+Zc^2)).*(Ka*(dur*0.5-eta)).^2)/(8*vp^2); %
RCM
cells=round(deltaR/.56); % .56 meters/cell in range direction
fs=zeros(PRF*dur,rbins); fsm=fs; fsmb=fs; smb=fs; fsac=fs; sac=fs;

% Range Compression
for k=1:(PRF*dur);
    fs(k,:)=fty(s(k,:));
    fsm(k,:)=fs(k,:).*conj(fs0);
    smb(k,:)=ifty(fsm(k,:));
end;

% Azimuth Reference Signal
smb0=exp(cj*pi*Ka.*eta.*(2*eta(PRF*dur/2+1)-eta));
fsmb0=ftx(smb0); % Azimuth Matched Filter Spectrum
for l=1:rbins;
    fsmb(:,l)=ftx(smb(:,l)); % Azimuth Fourier Transform
end;

% Range Cell Migration Correction (RCMC)
for k=1:dur*PRF/2;
    for m=1:rbins-9
        fsmb(k,m)=fsmb(k,m+cells(k));
        fsmb(PRF*dur-k,m)=fsmb(PRF*dur-k,m+cells(k));
    end
end;

% Azimuth Compression
for l=1:rbins;
    fsac(:,l)=fsmb(:,l).*conj(fsmb0); % Azimuth Matched Filtering
    sac(:,l)=iftx(fsac(:,l)); % Final Target Image
end;

% Plot Final Results
figure(1), imagesc(abs(sac)), title('Final Processed Image')
xlabel('Range, samples'), ylabel('Azimuth, samples')
```

Appendix D: Cube Simulation SAR Code

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   3-D Cube Simulation for SAR Imaging using Range Doppler Algorithm
%
%   Matthew Schlutz
%   Cal Poly, San Luis Obispo
%   SAR ATR Project
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function SimulationCube()
clear

% INITIALIZATION
% Main SAR Parameters
PRF=300; % Pulse Repetition Frequency (Hz)
dur=3; % Time of Flight (sec), PRF*dur = received echoes
abins=PRF*dur; % Number of slow time (Azimuth) samples
vp=200; % Velocity of platform
fc=4.5e9; % Carrier frequency (4.5GHz)
La=2; % Antenna length actual
theta=45; %Look Angle in degrees
rotation=45;
Ro=20e3; % Minimum Distance to Center of Target
Xc=Ro*cos(theta*pi/180); % Range distance to center of target area
Zc=Ro*sin(theta*pi/180); % Altitude of platform
X0=200; % Half Target Area Width (Target is located within [Xc-
X0,Xc+X0])
Tp=.25e-5; % Chirp Pulse Duration
B0=100e6; % Baseband bandwidth is plus/minus B0

%Noise
noise=1; % Additive White Gaussian Noise Flag (1=noise on, 0=noise off)
std_dev=.06; % Standard Deviation of Noise

%3D Cube Flags
cubestg=1; %Cube stage being processed (cube_stg1 & cube_stg2)
d=Xc*tan(rotation); %Cube view stage switch offset
if (d>PRF*dur/2) || (d<-PRF*dur/2)
    two_stage=0;
else
    two_stage=1;
end

% General Variables
cj=sqrt(-1);
c=3e8; % Propagation speed
ic=1/c; % Propagation frequency
lambda=c/fc; % Wavelength (60cm for fc = 4.5e9)
eta=linspace(0,dur,abins)'; % Slow Time Array

% Range Parameters
Kr=B0/Tp; % Range Chirp FM Rate
dt=1/(2*B0); % Time Domain Sampling Interval
Ts=(2*(sqrt(Xc^2+Zc^2)-X0))/c; % Start time of sampling
Tf=(2*(sqrt(Xc^2+Zc^2)+X0))/c+Tp; % End time of sampling
```

```
% Azimuth Parameters
Ka=(2*vp^2)./(lambda*(sqrt(Xc^2+Zc^2))); % Linear Azimuth FM rate

% Measurement Parameters
rbins=2*ceil((.5*(Tf-Ts))/dt); % Number of fast time (Range) samples
t=Ts+(0:rbins-1)*dt; % Time array for data acquisition
s=zeros(abins,rbins); % Echoed signal array

% Target Initialization
process=1;
while process==1
    targets='cube_stg1_3';%Select Input Target Profile
    if cubestg==2
        targets='cube_stg2_3';
    end
    target=imread([targets(1:length(targets)-1) '1'],'gif'); %samples
target
    O=sscanf(targets(length(targets)),'%f'); %number of layers
    [M N]=size(target); ntarget=M*N*O; %number of point targets in 3D
Space
    tnum=1; xn=zeros(ntarget,1); yn=xn; Fn=xn; zn=xn;
    for i=1:O %loop through target layers
        target=imread([targets(1:length(targets)-1) num2str(i)],'gif');
        for m=1:M
            for n=1:N
                xn(tnum)=(n-N/2)*cos(rotation)-(M/2-m+1)*sin(rotation);
                yn(tnum)=(n-N/2)*sin(rotation)+(M/2-m+1)*cos(rotation);
                zn(tnum)=i-1;
                Fn(tnum)=double(target(m,n))/255;
                tnum=tnum+1;
            end
        end
    end
    stretch=6;
    xn=xn*stretch; yn=yn*stretch; zn=zn*stretch; %Stretch Target
Profile
    if two_stage==0
        process=0;
    else
        if cubestg==1
            xnc=zeros(2,length(xn)); ync=xnc; znc=xnc; Fnc=xnc;
        end
        xnc(cubestg,:)=xn; ync(cubestg,:)=yn; znc(cubestg,:)=zn;
        Fnc(cubestg,:)=Fn;
        cubestg=cubestg+1;
        if cubestg==3
            process=0;
        end
    end
end

% RAW SAR SIGNAL GENERATION
for j=1:(abins);
    %Select Cube Faces Shown (Line of Sight)
    if two_stage==1
        if j<PRF*dur/2+d
```

```
            xn=xnc(1,:);
            yn=ync(1,:);
            zn=znc(1,:);
            Fn=Fnc(1,:);
        else
            xn=xnc(2,:);
            yn=ync(2,:);
            zn=znc(2,:);
            Fn=Fnc(2,:);
        end
    end
    %Echo Gen.
    for i=1:ntarget;
            wa=sinc(La*(atan(vp*(eta(j)-
dur/2+yn(i)/vp)/sqrt(Xc^2+Zc^2)))/lambda).^2;
            R=sqrt((Xc+xn(i))^2+(Zc-zn(i))^2+vp^2*(eta(j)...
                -dur/2+yn(i)/vp).^2);
            td=t-2*R/c;
            if noise==1
                s(j,:)=s(j,:)+wgn(1,rbins,std_dev)...
                    +Fn(i)*wa*exp(-cj*(4*pi*(fc*ic)*R)+cj*pi*Kr*...
                    ((td.^2)-td*Tp)).*(td >= 0 & td <= Tp);
            else
                s(j,:)=s(j,:)+Fn(i)*wa*exp(-
cj*(4*pi*(fc*ic)*R)+cj*pi*Kr*...
                    ((td.^2)-td*Tp)).*(td >= 0 & td <= Tp);
            end
    end
    if mod(j,50)==0
        display(j/9) % Echo Gen. Percent Complete
    end
end

% RANGE DOPLER ALGORITHM (RDA)
% Range Reference Signal
td0=t-2*sqrt(Xc^2+Zc^2)/c;
pha20=pi*Kr*((td0.^2)-td0*Tp);
s0=exp(cj*pha20).*(td0 >= 0 & td0 <= Tp);
fs0=fty(s0); % Range Reference Signal in frequency domain
% Power Equalization of Range Reference Signal
amp_max=1/sqrt(2); % Maximum amplitude for equalization
afsb0=abs(fs0);
P_max=max(afsb0);
I=find(afsb0 >= amp_max*P_max);
fs0(I)=((amp_max*(P_max^2)*ones(1,length(I)))./afsb0(I))...
    .*exp(cj*angle(fs0(I)));

% Range Compression
fs=zeros(abins,rbins); fsm=fs; fsmb=fs; smb=fs; fsac=fs; sac=fs;
for k=1:(abins)
    fs(k,:)=fty(s(k,:));
    fsm(k,:)=fs(k,:).*conj(fs0);
    smb(k,:)=ifty(fsm(k,:));
end

% Azimuth Reference Signal
smb0=exp(cj*pi*Ka.*eta.*(2*eta(abins/2+1)-eta));
```

```
fsmb0=ftx(smb0); % Azimuth Matched Filter Spectrum
for l=1:rbins
    fsmb(:,l)=ftx(smb(:,l)); % Azimuth Fourier Transform
end

% Range Cell Migration Correction (RCMC)
deltaR=(lambda^2*(sqrt(Xc^2+Zc^2)).*(Ka*(dur*0.5-eta)).^2)/(8*vp^2); %
RCM
RCM=round(deltaR/.56); % .56 meters/cell in range direction
for k=1:dur*PRF/2;
    for m=1:rbins-9
        fsmb(k,m)=fsmb(k,m+RCM(k));
        fsmb(abins-k,m)=fsmb(abins-k,m+RCM(k));
    end
end;

% Azimuth Compression
for l=1:rbins;
    fsac(:,l)=fsmb(:,l).*conj(fsmb0); % Azimuth Matched Filtering
    sac(:,l)=iftx(fsac(:,l)); % Final Target Image
end;

% Plot Final Results
figure(1), imagesc(abs(sac)), title('Final Processed Image')
xlabel('Range, samples'), ylabel('Azimuth, samples')
```

Appendix E: MATLAB FFT Support Function Code for SAR Simulations

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Inverse FFT w.r.t. the second variable %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function s=ifty(fs)
s=fftshift(ifft(fftshift(fs.'))).';


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Inverse FFT w.r.t. the first variable %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function s=iftx(fs)
s=fftshift(ifft(fftshift(fs)));


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Forward FFT w.r.t. the second variable %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function fs=fty(s)
fs=fftshift(fft(fftshift(s.'))).';


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Forward FFT w.r.t. the first variable %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function fs=ftx(s)
fs=fftshift(fft(fftshift(s)));
```