AFIT/GE/ENG/93D-22

Processing of Wide-Angle Synthetic Aperture Radar Signals for Target Detection

THESIS

Kurt William Knurr
Captain, USAF

AFIT/GE/ENG/93D-22

115P8

AFIT/GE/ENG/93D-22

Processing of Wide-Angle Synthetic Aperture Radar Signals for Target Detection

THESIS

Presented to the Faculty of the Graduate School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science

Kurt William Knurr, B.S.

Captain, USAF

December, 1993

Acknowledgements

In the accomplishment of this research, I've had a great deal of help from various people. I would like to thank my faculty advisor, Dr. Byron Welsh, for his continuous guidance and encouragement throughout this endeavor. I would also like to thank the other members of my committee for their advice and direction during my research: Capt Joseph Sacchini for his incite and advice on statistical matters; Maj Robert Williams, my research sponsor, for his financial as well as academic support and interest; Dr. Michael Clark for his invaluable knowledge of MATLAB® and detection related problems. I would also like to thank Mr. Dan Zambon for the use of his hardware and infinite patience while I struggled through a myriad of technical problems. Also, Mr. Steve Kilberg for providing the software ahead of scheduled Air Force acquisition, software which I could not have completed this research without. Finally, I would like to thank Dr. Jian Li and Mrs. Patti Ryan for their input and assistance on software and mathematical problems.

A special note of appreciation for my wife, Kathy, and my daughters Rachael and Sarah for their patience and understanding during what must have seemed like an endless period of study and research. In closing, I would like to thank God, my Lord and Savior, Jesus Christ for his strengthening, patience, and support throughout this effort and my entire life. " ...everything is possible for him who believes." [Mark 9:23 (NIV)].

Kurt William Knurr

Table of Contents

Page

List of Figures

List of Tables

AFIT/GE/ENG/93D-22

Abstract

This study investigated methods of target detection using Wide-Angle Synthetic Aperture Radar (WASAR). WASAR uses multiple aspect angle Synthetic Aperture Radar (SAR) images of the same scene. The SAR images were generated using a pre-release software package from Loral Corporation. The software was able to generate 512 by 512 pixel SAR images that contained various vegetation returns which for our purposes we classified as clutter. Within this clutter, targets (M35 trucks) could be placed at random locations and orientations. The software also had the capability of generating fully-polarimetric WASAR images with multiple depression angles. This data was then processed and various detection algorithms tested to exploit the amount and diversity of information available from the multiple images. SAR images are generally known to contain large amounts of data and WASAR images contain even more due to the multiple images. Various pre-processing filters were analyzed for detection optimization. These filters included: polarimetric averaging, polarimetric span, polarimetric optimal weighting, and polarimetric whitening filter. Simple classical detection (thresholding) algorithms were evaluated using these pre-processed data sets. The use of WASAR imagery improved detection by allowing thresholds to be set higher than for simple SAR thereby avoiding false alarms yet still allowing detection of the known targets.

Processing of Wide-Angle Synthetic Aperture Radar Signals for Target Detection

I. Introduction

1.1 Problem Statement and Significance

This thesis investigates the detection of targets using Wide-Angle Synthetic Aperture Radar (WASAR). The detection and identification of military targets (buildings, vehicles, roads, runways, etc.) is important to the Air Force mission of strategic and tactical reconnaissance and bombardment. To this end, recently available WASAR imagery needs to be exploited for detection purposes. Synthetic Aperture Radar (SAR) produces high resolution imagery using a moving radar platform to synthesize the resolution cells. SAR has the advantage of being relatively weather invariant as compared to photo reconnaissance. As an expansion of SAR, WASAR provides multiple viewing angles of the same SAR image. Wide-Angle SAR improves on SAR by providing more information on a scene. By viewing a scene from several different angles, it may be possible to pick up a "glint" at one of those angles indicating the presence of a target. Also, most modern SAR systems provide the data in fully polarimetric format which will be explained more fully later. Fully polarimetric data provides even more information on a scene over that provided by a system capable of only a single polarimetric return. The diverse information obtained from WASAR provides the target detector with vast amounts of information that should make detection much easier. This thesis takes a first look at the use of Wide-Angle Synthetic Aperture Radar imagery for target detection.

1.2 Background and Problem Areas

Before moving into the processing of the WASAR images, it may be helpful to go over some of the terminology that will be used throughout this thesis. Target detection versus target recognition

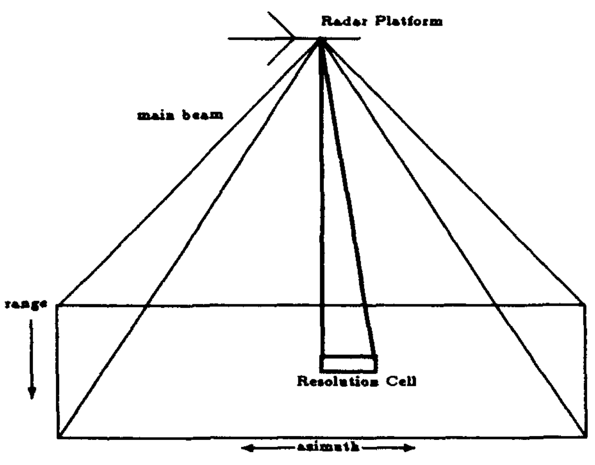


Figure 1.1 SAR Image Creation

will be discussed, along with terms associated with SAR and WASAR. Some of the problems involved in using SAR and WASAR will also be discussed.

The first concept that needs to be understood is Synthetic Aperture Radar. SAR uses a moving platform radar system to synthesize a high-resolution image. The resolution of the image is determined by the pulse width (range resolution) and the distance the platform travels (azimuth resolution). These two parameters form a rectangular resolution cell in space, as seen in Figure 1.1. Through geometrical processing, these resolution cells are then combined to create an image. Typical SAR imagery is shown in figure 2.9 on page 2-15. This image shows a forested region on a grassy plain. Interspersed within this image are targets. The targets in some instances look similar to trees in the image.

Wide-Angle Synthetic Aperture Radar (WASAR) differs from normal SAR by generating multiple images of the same scene. The images are formed over several target aspect angles. Usually these different aspect angles are the result of a very wide beam that is divided into several smaller beams, as shown in Figure 1.2. As each of these beams pass over the area of interest, the radar receives a first, a second, and a third look at the same scene from different viewing angles.

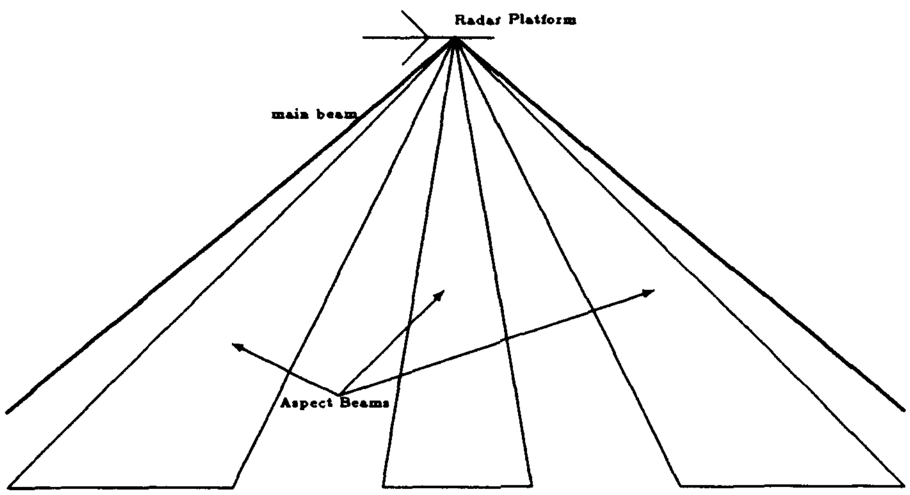


Figure 1.2 WASAR Image Creation

The number of looks depends on the number of narrow beams carved out of the wide beam. This research used seven aspects. Currently, there are no operational systems that use this technique; however, the use of WASAR for various reconnaissance and surveillance purposes is being explored by several agencies including the Air Force's Wright Laboratory at Wright-Patterson Air Force Base, Ohio. One of the problems with WASAR is the registering of the multiple images with respect to each other. Since the images are taken from different aspect angles, the coordinate system for one image is not the same as that in another. Registering the coordinate systems is important if the images are to be compared to each other in any way. Variations in the radar platform can cause major problems with this coordinate registering. For example, if the range to target distance varies from one aspect to another, the ground plain resolution will not be the same from image to image. In other words, a pixel in one image may represent 2 square feet; whereas, in another it may represent 3 square feet. This particular problem of coordinate registering can pose quite a problem when the radar system is an airborne SAR platform. In this case, either the aircraft has to fly a very precise route, or the images have to be registered using some type of signal processing system that uses not only the images but flight data.

Additionally, SAR and WASAR imagery can be generated using polarized data. Polarization is a term used to describe the electro-magnetic wave orientation of the radar transmit and receive signals. Fully polarimetric radar returns consist of three distinct modes: HH, HV, and VV. The HH term describes the horizontally polarized electro-magnetic (EM) field resulting from illumination of the scene by a horizontally polarized EM source. HV polarization describes the horizontally polarized EM field resulting from illumination by a vertically polarized EM source. In order for a radar data return set to be fully polarimetric, it **must** contain all three of these elements. Polarimetric processing is the combining of the HH, HV, and VV images in some optimal fashion in order to improve target to clutter ratios.

To this end, Holm [7] made a significant statement in his article about polarimetric properties in general. He stated that if the target and the clutter had differing geometrical characteristics, with the proper polarimetric processing, the radar return from clutter and target might be distinguishable [7:pages 148–149]. One algorithm that exploits polarimetric data is the *Polarimetric Whitening Filter* (PWF). More in-depth information on the PWF can be found in the articles co-authored by Novak [19, 20, 21]. These processing techniques may prove useful in the reduction of speckle within the SAR images. Novak states that reduction of speckle improves the ability to detect targets in the image. Novak [21] also discusses various detection algorithms using these various processing techniques. He discovered that these processors, the Optimally Weighted Filter (OWF) in particular, do not perform significantly better than working with a single polarization. He did state; however, that a "multi-look" radar system should improve the odds of detection. Since WASAR is a "multi-look" system, these processing techniques could prove to be quite useful.

Target detection is the main subject of this thesis. *Detection* of a target involves the determination of the presence or absence of the target. This thesis does not deal with the *identification* of a target. Since we don't know any specific parameters of the target, the problem of detection becomes more difficult. However, there are certain assumptions about the target that can be made.

By assuming a *target* is man-made, we can make certain speculations about its characteristics. Man-made objects have a tendency to be regularly shaped, that is composed of flat, spherical, and/or cylindrical shapes. Buildings for example are usually composed of flat walls and flat or pyramidally shaped roofs. Naturally occurring objects usually have irregular surfaces, such as a forested area, a shrub filled plain, or to a certain extent even a grassy plain is relatively irregular in appearance compared to a building. Smaller man-made objects, such as vehicles, also appear to be regularly shaped.

Two techniques are currently used for target detection; change detection and threshold detection. Change detection uses two separate images of the same scene to determine any differences between the two images. Normally, this would involve a temporal change, a change in time between the two images. However, it is possible to exploit some of these change detection techniques when using the multiple images associated with WASAR. Problems arise with change detection when the object being detected doesn't vary with aspect angle. The other detection technique, thresholding, involves the comparison of pixel intensities with a known value or threshold. All pixels above this threshold level are then identified as target pixels. Problems with thresholding techniques occur when the intensities of the clutter and target are indistinguishable making it harder to set an appropriate threshold. **The goal of this research is to use a combination of these two techniques in tandem with an optimal polarimetric processing technique to perform target detection.**

Difficulties with detection in general using WASAR stem from the relatively recent availability of WASAR data. How classical detection techniques will perform on WASAR imagery is unknown. No published techniques are available for optimally combining the WASAR images. Theoretically though, clutter returns will stay relatively the same throughout aspect changes; whereas, target returns (e.g. returns from flat plates) will change more drastically over aspect variations. If this information could be exploited by using some type of change detection technique, perhaps target detection could become more probable than using a single image. Though most techniques of change detection [1, 6, 28] involve the use of the same image (same frequency, aspect angle, depression angle, etc. except for a temporal difference), some of these detection techniques may find application in change detection between aspect angles.

1.3 Concluding Remarks

As stated previously, this thesis deals with exploiting data available through Wide-Angle SAR for the purpose of target detection. The initial problem is data availability, since Wide-Angle SAR concepts are relatively new. In order to work with WASAR data, the data must first be available and realistic. The second part of the problem with working with WASAR images reveals itself logically as a registration problem between the various aspect angles. The coordinate system in one image must be related to the coordinate system in another. The third step is pre-processing in which the images are combined polarimetrically to optimize any following detection algorithm. This step is an attempt to use the polarimetric diversity advantageously in detecting targets. The last step of course is the detection phase. The ultimate goal of this research is to determine if the use of WASAR instead of SAR improves the ability to detect targets. These four areas are discussed in the following chapters.

II. Pre-Detection Processing

2.1 Data Generation

The initial step in the detection of a target within a SAR image is obtaining a SAR image for manipulation. Since real WASAR imagery is not currently available, it is necessary to generate the image data using a new software package from Loral. The software allows the generation of an image with multiple targets (M35 trucks) randomly placed throughout the image region. Various images were generated for detection comparison including: targets in grass, targets in a scene with 25% forestation, and targets in a scene with 50% forestation. Within the images, any forested region contains deciduous trees in fall season with variable tree density. The simulated SAR system operates at 1200 MHz (L-band) with constant spatial resolution. The output is fully polarimetric complex image data. An L-band frequency is used because this range of frequencies has been shown to have foliage penetration capabilities over the use of higher frequencies. The targets are randomly interspersed throughout the entire image. Imagery data is generated for a depression angle of 15^o at aspect angles of $\pm 45^o$, $\pm 30^o$, $\pm 15^o$, and 0^o. This data is then input to a MATLAB® routine which displays the log magnitude value of the images as shown in Figure 2.1. This figure shows all 21 images involved in the seven aspect fully polarimetric image data set. More specific information on the WASAR imagery is contained in Appendix A.

Within the scene, each pixel represents approximately 2 square feet of the actual ground scene. Each image is composed of 512×512 pixels. The $+45^o$ HH image displayed in Figure 2.1 is also shown enlarged in Figure 2.9 on page 2-15. The images appear to show a forested stretch of land within a grassy area. Notice that the higher returns (whiter areas) correspond to the trees and targets.

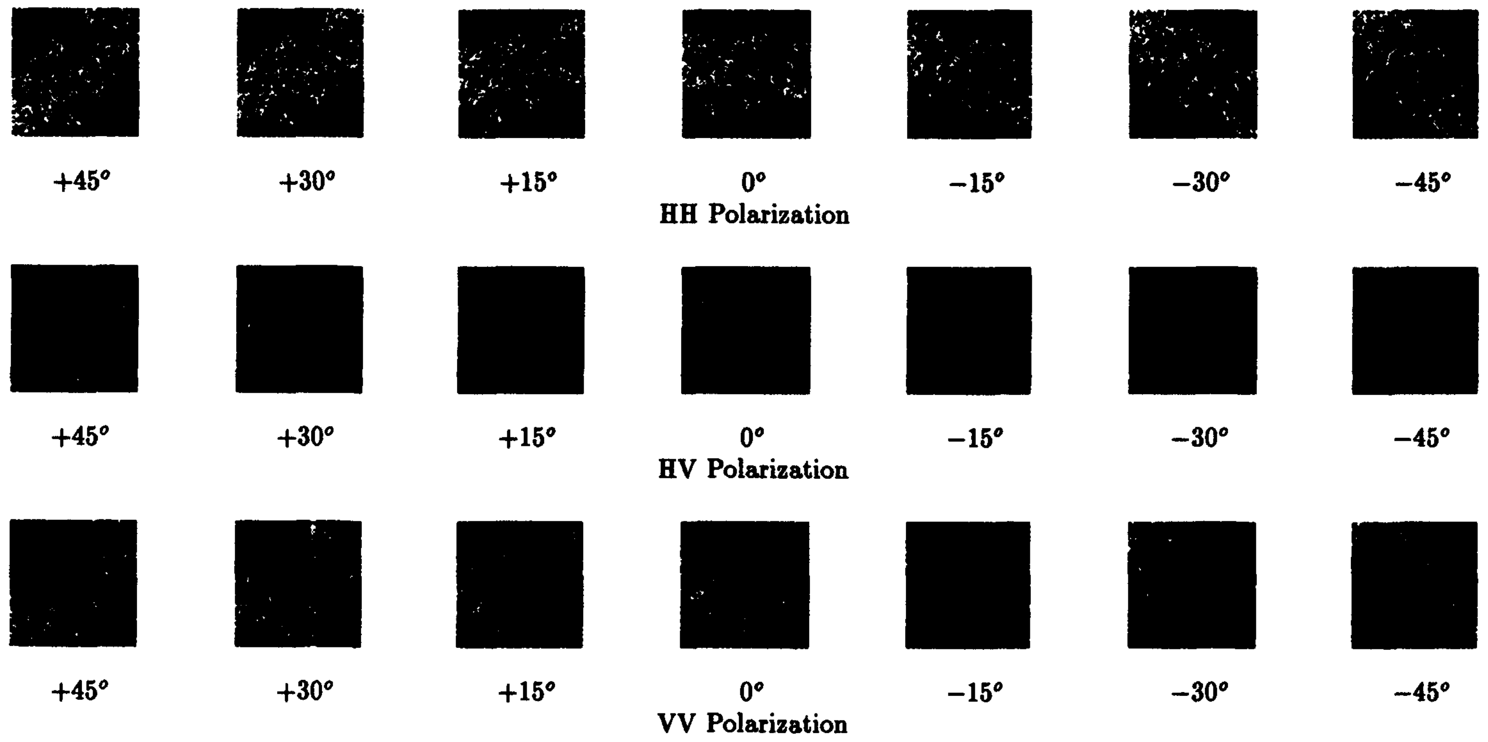
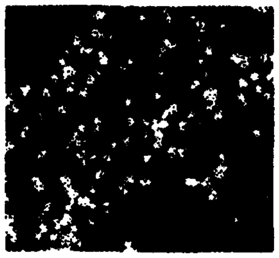


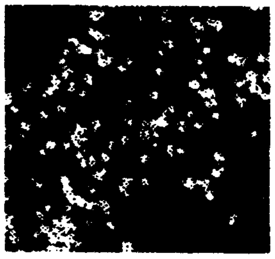
Figure 2.1 Images Displayed Before Rotation

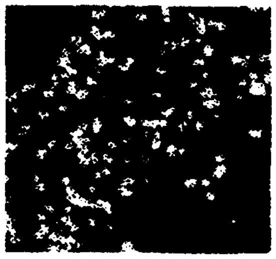
2.2 Aspect Angle Registration, Image Rotation

The next step in the detection process is to devise some scheme of registering the various images so that pixel coordinates in one aspect relate to those in another. If the images are not "ground plane" images, this could be a very difficult problem involving projection of the image into the ground plane and interpolation. Fortunately, the images generated for this thesis are in the ground plane and require only a simple rotation in order to register pixel-to-pixel between the various images. This assumption was verified by using the rotational method described in the following paragraph and then matching objects within the images. It may also be of interest to note that the center of rotation necessary to match the images turned out to be the very center of each image. The rotation process used in this research also caused the image registration between various aspect angles to be off by one pixel at the most. This means that an object, after rotation of the image, may have moved to a position that was shifted by one pixel from its true position. This offset was not deemed to be a large problem, and for the most part it could not be corrected since each image is a discrete set of pixels.

The images generated by the Loral software were contained in a 512 × 512 matrix of pixel values. The choice of rotation is arbitrary, as long as all aspects are rotated to the same coordinate system. In order to simplify the rotation algorithm, all aspects are rotated to the most positive aspect, 45°. This means that one image, the +45° image, does not have to be rotated at all, and the −45° image can be rotated using an intrinsic MATLAB® function. After rotation, it should be obvious to the reader that the overlapping or common area between the different image aspects is a shape approaching a circle. Since MATLAB® works more easily with square images, the images are reduced to a square area that fits inside of this overlapping circular region. The result is the previously mentioned 512 × 512 images are now reduced to 363 × 363 pixel images; however, the coordinates for one image map directly to the coordinates in any other image.


(a) HH +45°


(b) HH 0°


(c) HH -45°

(d) HV +45°

(e) HV 0°

(f) HV -45°

(g) VV +45°

(h) VV 0°

(i) VV -45°

Figure 2.2 Images Displayed After Rotation

The images generated from the last section are shown rotated in Figure 2.2. Only the $\pm 45^o$ and 0^o aspect angles are shown. Notice that the images are very similar to each other; however, there are unique features to each of the nine images.

2.3 Pre-processing

After the images are rotated, various types of pre-processing on the polarimetric data can be done. It is desirable to find some optimum way of combining the HH, HV, and VV images in order to optimize target detection. Four methods of combining these images are explored: simple average, span, optimal weighting, and polarimetric whitening filter. The span, optimal weighting, and polarimetric whitening filter were all proposed by Novak [20]. For derivations of these particular pre-processing techniques please refer to the article indicated. The Novak techniques are used in this research with certain qualifications: 1) measured statistics were used instead of actual statistics, and 2) the entire image was used and was not separated according to clutter and target when determining statistics. Measured statistics were used instead of actual data mainly because in a real-world image, all that is available is the measured data. Any method that reduces the background clutter and enhances the target return is beneficial in this preprocessing stage.

2.3.1 Polarimetric Average.

The first method this research uses is the polarimetric average, which is simply the average of the HH, HV, and VV images

$$\text{ave} = \frac{1}{3}\left(|\text{HH}| + |\text{HV}| + |\text{VV}|\right). \tag{2.1}$$

Equation 2.1 represents a pixel-by-pixel operation carried out on each of the different aspect angles. The $|\cdot|$ symbol indicates a pixel-by-pixel magnitude, or modulus, value and is not to be confused with a determinant of a matrix. The images that result from this method are shown in Figure 2.3.

(a) +45° (b) 0° (c) -45°

Figure 2.3 Images Displayed After Rotation and Polarimetric Averaging

2.3.2 Polarimetric Span. The next method explored is the Polarimetric Span, which is another type of averaging technique. The span operation is given by

$$\text{span} = |\text{HH}|^2 + 2\,|\text{HV}|^2 + |\text{VV}|^2 . \tag{2.2}$$

Due to the squaring within Equation 2.2, the higher intensity values are emphasized and the impact of lower intensity returns are lessened. Notice that the results from a span operation are very similar to those obtained for the average (see Figure 2.4), though the background (grass) is darker, implying that the contrast has been increased. Notice that in Equation 2.2, phase information is completely lost. If the actual values of the averaged and span-encoded images were examined and compared, they would be vastly different as can be seen by Equations 2.1 and 2.2.

(a) +45° (b) 0° (c) -45°

Figure 2.4 Image Displayed After Rotation and Polarimetric Span

2.3.3 Polarimetric Optimal Weighting.

The Polarimetric Optimal Weighting function, takes the span one step further in that it uses statistical information from the image itself to weight the various polarimetric returns. The weighting function is described by

$$\text{pow} = |\text{HH}|^2 + k_2 |\text{HV}|^2 + k_3 |\text{VV}|^2 . \tag{2.3}$$

Where the constants, k_2 and k_3, are given by

$$k_2 = \frac{1 + |\rho|^2}{\epsilon}, \tag{2.4}$$

$$k_3 = \frac{1}{\gamma}, \tag{2.5}$$

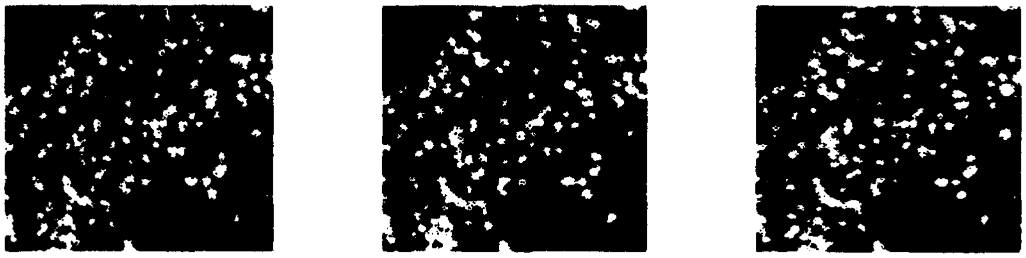
and

$$\rho = \frac{\text{E}\{\text{HH} \cdot \text{VV}^*\}}{\sqrt{\text{E}\left\{|\text{HH}|^2\right\}\text{E}\left\{|\text{VV}|^2\right\}}}, \tag{2.6}$$

$$\epsilon = \frac{\text{E}\left\{|\text{HV}|^2\right\}}{\text{E}\left\{|\text{HH}|^2\right\}}, \tag{2.7}$$

$$\gamma = \frac{\text{E}\left\{|\text{VV}|^2\right\}}{\text{E}\left\{|\text{HH}|^2\right\}}. \tag{2.8}$$

Notice that Equations 2.6, 2.7, and 2.8 use the statistical mean of the image ($E\{\cdot\}$). This is a measured mean and therefore an approximation of the actual mean of the image. As the data set gets larger, approaches infinity, the measured mean will approach the actual mean. The actual mean would be the mean of the probability density function which was used to create the cluttered image. In a real-world image only the measured data is available for determining statistical information; therefore, it is more prudent to test algorithms using the measured statistics even if the actual statistics are available. The results of optimal weighting are shown in Figure 2.5.



(a) +45° (b) 0° (c) -45°

Figure 2.5 Image Displayed After Rotation and Polarimetric Optimal Weighting

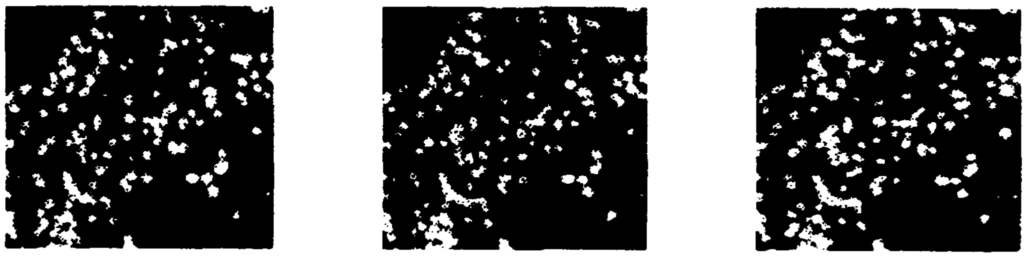
2.3.4 Polarimetric Whitening Filter. Finally, the Polarimetric Whitening Filter is described by

$$\text{pwf} = \frac{|HH|^2}{\sigma_{HH}(1-|\rho|^2)} + \frac{|VV|^2}{\sigma_{HH}(1-|\rho|^2)\gamma} + \frac{|HV|^2}{\sigma_{HH}\epsilon} - \frac{2|\rho|}{\sigma_{HH}(1-|\rho|^2)\sqrt{\gamma}}|HH||VV|\cos(\phi_{HH}-\phi_{VV}-\phi_{\rho}), \tag{2.9}$$

where ρ, ϵ, and γ are as defined previously in Equations 2.6, 2.7, and 2.8 respectively. The parameter σ_{HH} is defined as

$$\sigma_{\mathrm{HH}} = \mathrm{E}\left\{|\mathrm{HH}|^2\right\}. \tag{2.10}$$

The parameters ϕ_{HH}, ϕ_{VV}, and ϕ_{ρ} are the phase angles of the indicated subscripts. The results of using this PWF are shown in Figure 2.6. Notice the similarities between the Optimal Weighting



(a) +45° (b) 0° (c) -45°

Figure 2.6 Images Displayed After Rotation and Polarimetric Whitening

function Equation 2.3 and the Polarimetric Whitening Filter (PWF) defined by Equation 2.9. The PWF takes advantage of not only statistical information relating to the image, but also exploits phase data. The images as they are displayed in this figure appear to be very similar to the optimally weighted images. If these whitened images were displayed in the same manner as the optimally weighted images, the display would be almost totally white. In order to display the whitened images and be able to see any detail, the intensity of the display was diminished. Most of the images that are displayed in this document use the same intensity setting so that comparisons can be made between the various images. A few of the figures do use a different image intensity setting or imaging method though. When a variation in the imaging technique is used, a note will be made of the fact. This variation in intensity does not affect any of the procedures accomplished in this research since the intensity levels are adjusted only during display for ease of viewing.

2.4 Statistical Analysis

As stated previously, the polarimetric combining techniques might prove useful for improving our target detection capability. In order to measure this improvement, some metric must be used to compare the various polarimetric techniques. To this end, there are two metrics that are analyzed in order to quantify whether the previously mentioned polarimetric techniques improved the images in some way. The first metric is the standard deviation to mean ratio of the image. The other metric is target to clutter ratio. The target to clutter ratio is important since any decrease in this ratio would also decrease the probability of detection.

The first metric, the standard deviation to mean ratio, is defined by Novak [20:page 294]. This metric is computed by calculating the standard deviation of the pixel intensities of the entire image and dividing by the mean value of the pixel intensities of the whole image. This ratio is then converted to dB and used to compare the various polarimetric processing techniques.

Aspect	+45°	+30°	+15°	0°	-15°	-30°	-45°
Standard Deviation to Mean Ratio (dB)							
HH	41.42	38.80	35.28	43.10	46.34	42.39	39.15
HV	41.69	35.69	33.00	38.05	39.89	33.27	36.57
VV	43.75	33.63	32.77	47.41	39.95	44.19	45.57
averaged	36.18	36.32	33.54	41.48	37.17	37.69	37.54
spaned	30.46	23.52	23.43	29.80	30.75	30.60	30.44
optimal	28.14	23.67	23.73	27.21	28.24	28.23	28.05
whitened	27.08	24.41	24.38	25.83	27.05	27.78	27.02
Target to Clutter Ratio (dB)							
HH	35.91	38.12	31.96	40.43	36.44	34.27	35.81
HV	32.93	31.83	31.57	31.69	33.56	26.84	29.44
VV	36.96	30.76	32.44	32.07	30.14	36.41	35.33
averaged	35.49	35.51	31.14	35.88	33.73	35.35	36.54
spaned	35.35	32.42	32.09	35.81	35.47	34.88	35.55
optimal	35.08	33.05	32.68	35.31	35.00	34.66	35.28
whitened	34.76	33.26	33.03	34.89	34.63	34.55	35.00

Table 2.1 Statistics for Scene 1: Targets in Grass

The second metric, target to clutter ratio, is used to determine if any of the polarimetric processing techniques enhance target detection capabilities. Since the target locations are known, it is fairly simple to find the intensity of the pixels within a square of 11 × 11 pixels centered about the known target coordinates. An 11 × 11 square was chosen to insure that only pixels associated with a target are included in the target intensity values. The largest possible square was chosen in order to include as many pixel returns as possible to obtain an accurate target mean. All the target pixel intensities are averaged for a particular image and then divided by the average pixel intensity of the entire image. The mean of the image is taken over the entire image (including the targets) with the assumption that the image is large (in number of pixels) and the number of target pixels is small. All of the scenes generated for this research contain 8 to 10 targets and the image size after rotation is 363 x 363. Therefore, since the number of targets is small and the image size is relatively large, the impact of the target returns on the image mean is minimal. Also, notice from Table 2.1 that the target to clutter ratios are very large anyway. Subtraction of the targets from the clutter mean would only serve to increase the ratio even higher.

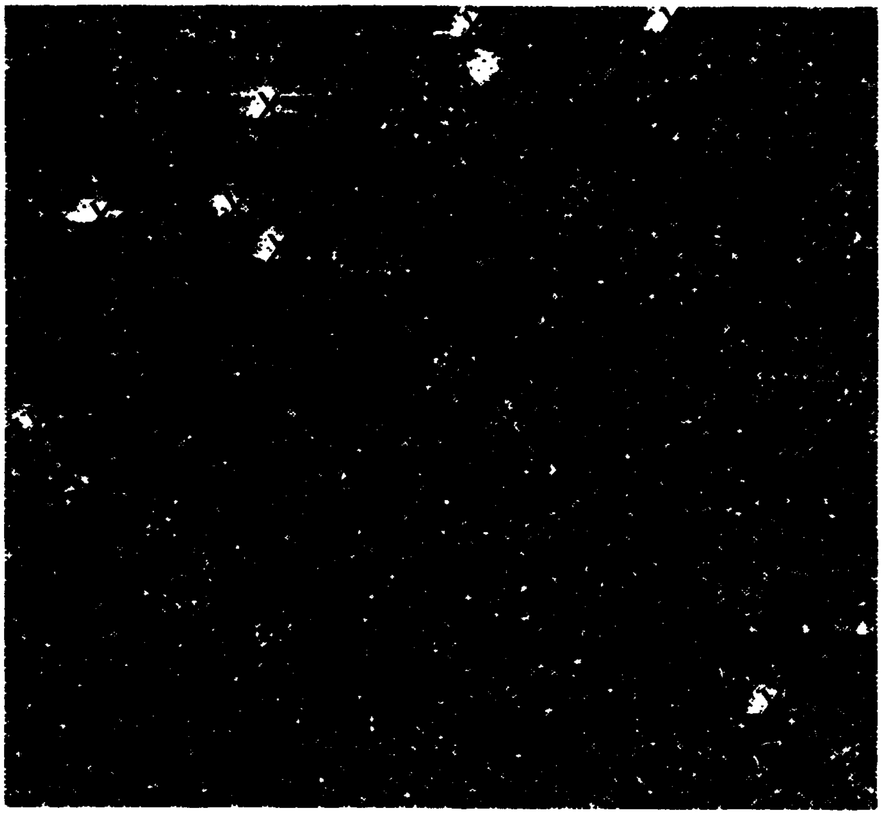


Figure 2.7 Scene 1: Targets in Grass (+45° Aspect HH Image After Rotation)

2.4.1 Scene 1: Targets in Grass. The first scene analyzed contains targets in grass. The +45° HH image is shown in Figure 2.7. The target locations are designated in the figure using *X*s. The returns from the trucks are slightly to the left and center of each *X*. Coordinates of the targets are available in a file generated by the Loral software. Notice that there appears to be a return without an X beside it. This is most likely a stray tree that was placed in the scene by the generation software. The software designer was contacted about this problem and the existence of a tree within an all grass scene could not be confirmed; however, the designer did state that it was possible.

The standard deviation to mean and target to clutter ratios for scene 1 are found in Table 2.1. Notice that in the standard deviation to mean ratio portion of the table, the relative values in the table decrease as you go down the columns of the processed images. This result is expected since the polarimetric processing techniques are designed to reduce this metric. As expected, the polarimetric whitening filter performs the best on average in reducing standard deviation to mean ratios. Additionally, target to clutter ratio, stays relatively the same (or slightly decreases) down the columns of the Table 2.1. This trend indicates that the polarimetric processing techniques do not improve target detection chances appreciably, and in some cases may actually decrease the odds. A decrease in the target to clutter ratio, results in the target becoming more and more indistinguishable from clutter intensities.

2.4.2 Scene 2: Targets in Trees (25% foliage). The second scene analyzed contains targets in trees and grass with 25% forestation. The +45° HH image is shown in Figure 2.8. Table 2.2 indicates that the techniques proposed by Novak [20] – span, optimal weighting, and pwf – perform the best on this data set for reducing standard deviation to mean ratios. The target to clutter ratio decreases slightly from the un-processed image, and in this case, the averaging technique performed the best.

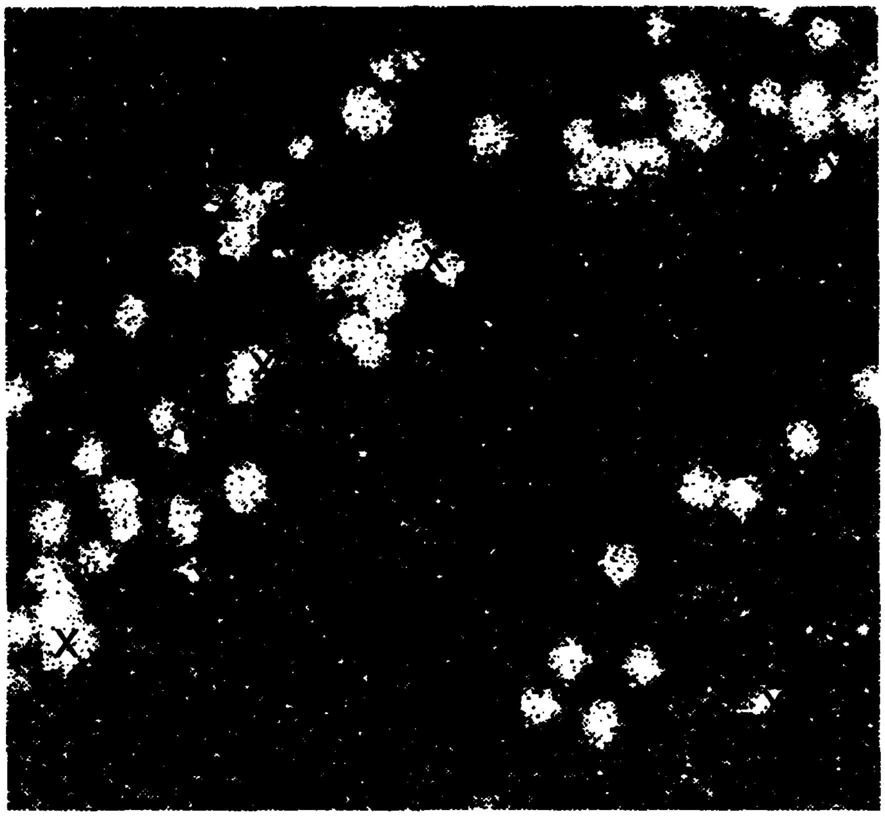


Figure 2.8 Scene 2: Targets in Trees and Grass (25% foliage) (+45° Aspect HH Image After Rotation)

Aspect	+45°	+30°	+15°	0°	-15°	-30°	-45°
Standard Deviation to Mean Ratio (dB)							
HH	38.24	40.31	46.33	39.68	38.33	30.80	34.84
HV	33.88	30.79	33.67	27.10	27.82	29.86	32.56
VV	38.60	34.25	31.49	36.93	45.70	30.68	32.17
averaged	35.20	45.24	39.56	35.04	34.26	28.11	30.06
spaned	17.47	14.63	19.43	26.52	26.60	18.36	19.64
optimal	17.55	17.01	18.17	23.93	24.50	19.05	19.12
whitened	17.89	17.51	18.69	22.69	24.25	18.45	19.59
Target to Clutter Ratio (dB)							
HH	37.74	37.27	46.92	39.01	35.82	33.06	34.27
HV	16.07	30.20	33.32	28.83	24.84	30.05	25.43
VV	40.14	37.35	32.78	20.08	45.75	32.03	29.41
averaged	39.38	46.26	37.48	34.42	37.95	31.69	31.77
spaned	30.05	27.55	28.64	35.71	33.50	29.78	30.87
optimal	30.74	29.55	28.88	34.31	33.24	30.90	31.71
whitened	30.89	29.61	29.10	33.84	33.09	30.75	31.84

Table 2.2 Statistics for Scene 2: Targets in 25% Foliage

Notice that on average, the target to clutter ratio has dropped as compared to targets in grass alone (compare Tables 2.1 and 2.2). This result is expected. It is reasonable to believe that a target in the forest would be a little bit harder to detect than that same target in grass. Also notice that the standard deviation to mean ratios are lower for scene 1 than for scene 2.

2.4.3 Scene 3: Targets in Trees (50% foliage). The third scene analyzed contains targets in trees and grass with 50% forestation. The +45° HH image is shown in figure 2.9. The statistical information is contained in Table 2.3. Information is missing from the +30° aspect due to the corruption of the VV polarized image (see the +30° VV image displayed in Figure 2.1 on page 2-2). The cause of this corruption is unknown, but, was a function of the parameters used in the generation of this particular scene. It may also be of interest to the reader, this is the scene that is used for the figures contained in the pre-processing section.

Notice in Table 2.3 that the standard deviation to mean ratio experiences large reductions as the pre-processing technique becomes more complex, moving down the columns. The polarimetric whitening filter again offers the best performance for this metric. Also, notice that the target to clutter ratio is not reduced significantly by the pre-processing stage. However, comparing Tables 2.1,

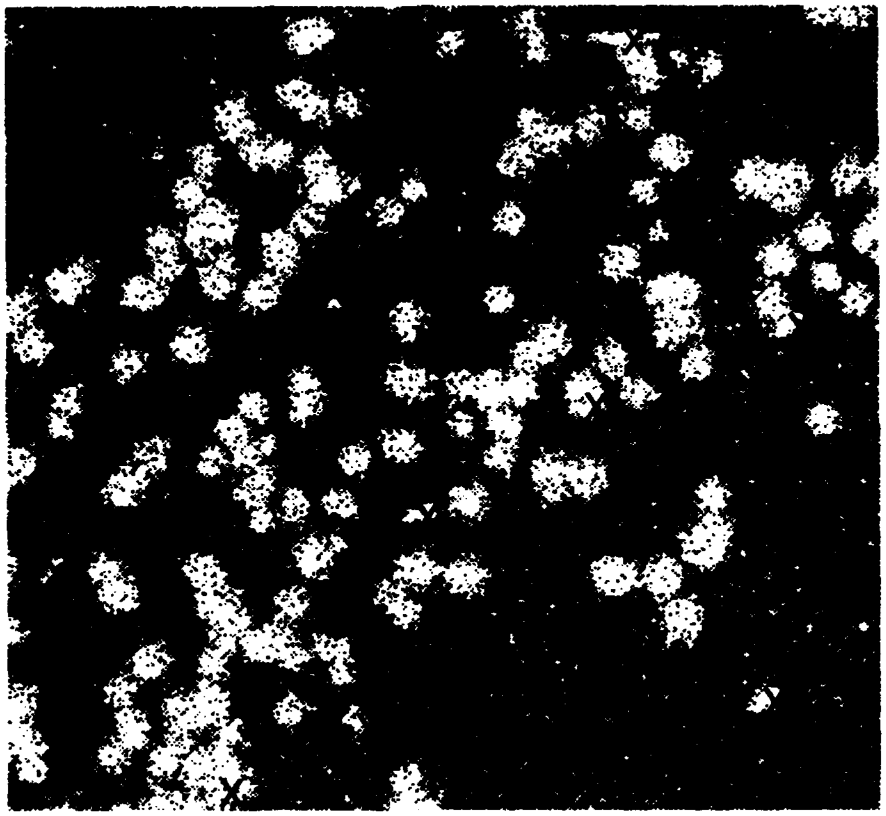


Figure 2.9 Scene 3: Target in Trees and Grass (50% foliage) (+45° Aspect HH Image After Rotation)

Aspect	+45°	+30°	+15°	0°	-15°	-30°	-45°
Standard Deviation to Mean Ratio (dB)							
HH	24.37	-	25.48	39.41	33.57	25.66	33.45
HV	28.43	-	26.27	27.96	24.77	27.32	23.19
VV	30.46	-	35.03	33.87	31.71	23.90	32.25
averaged	22.82	-	22.82	28.47	30.09	21.84	25.01
spaned	10.19	-	17.03	25.13	13.26	12.03	11.65
optimal	12.88	-	15.80	23.07	15.24	13.25	13.24
whitened	13.23	-	16.66	22.14	14.96	12.53	13.84
Target to Clutter Ratio (dB)							
HH	24.94	-	25.24	41.28	31.95	26.30	37.47
HV	24.36	-	25.00	22.97	28.06	26.92	22.06
VV	31.43	-	32.12	28.00	37.44	25.85	21.40
averaged	25.48	-	18.97	24.57	36.27	22.46	25.49
spaned	22.84	-	23.22	32.39	24.51	22.98	24.30
optimal	24.43	-	23.33	31.10	26.17	24.55	25.75
whitened	24.80	-	23.79	30.64	26.17	24.57	26.10

Table 2.3 Statistics for Scene 3: Targets in 50% Foliage

2.2, and 2.3, we can see a large drop in target to clutter ratios. This is to be expected since the forested portion of our scene is becoming thicker. If the entire scene were covered with forest, the target to clutter ratio might be expected to drop even further, perhaps down to unity. If this were to happen, say targets in a very thick forest, detection would become very difficult. This extreme case could not be explored since the Loral software only allowed a maximum of 50% forestation.

2.4.4 Statistical Overview. The purpose of any polarimetric pre-processing is to improve target detection capability (over the use of a single polarimetric image). It would appear from the three data sets that were analyzed in this research that the polarimetric pre-processing for the most part does not appreciably affect target detection capability as quantified by the target to clutter ratios. The use of a pre-processing stage does not improve the target to clutter ratio, and in some cases can actually decrease the ratio. However, the decreases resulting from this research are not unacceptable. A decrease would be considered unacceptable if the target to clutter ratio were to approach unity. The results from this chapter are inconclusive in favoring one technique over another for optimization of target detection since the target to clutter ratio stays relatively constant.

III. Target Detection

The main purpose of this thesis is target detection. To this end, threshold detection is accomplished on individual aspect angles. Threshold detection in its basic form is simply the comparison of the pixel intensity levels with a preset value or threshold. If the intensity of the pixel being tested is above this threshold, then that pixel is returned as a "target". After the threshold detector is used on all the image aspects, a comparison of the returned pixel coordinates is accomplished. For this thesis, this comparison was simply an accumulation of all the coordinates. In set terminology, this is known as a union set of the coordinates of the individual aspects. The threshold, γ, is set by arbitrarily selecting a value defined between the maximum and minimum pixel intensities

$$\gamma = \min\{|\text{pixel value}|\} + [\max\{|\text{pixel value}|\} - \min\{|\text{pixel value}|\}]\,\alpha. \tag{3.1}$$

where α takes on values ranging from 0 to 1. In the results to follow, the parameter α is plotted as the independent variable. The parameter α is referred to as the threshold and allowed to range from zero to unity. Notice that this equation and the realization in the code, "detect2.m" on page B-47, is different. The minimum pixel values range from 10^{-5} to 10^{-3}; whereas, the maximum is generally in the 10^2 range. Which indicates that the minimum is approximately zero when compared to the maximum pixel value and can be ignored in the formula, making the code realization equivalent to Equation 3.1.

Two detection methods are used. For the first method of detection, the threshold is found for the single $+45^o$ aspect image and then used for detection on the other aspects. In essence, this procedure sets a permanent threshold at some absolute value. For the second method of detection, a separate threshold is derived using Equation 3.1 for each image. This second method, then, allows a "floating" threshold of sorts. Once again, detection is done on individual aspects and the target coordinates combined. Threshold detection is also accomplished on one image aspect (the

+45° aspect in particular) and the results compared to detection using all the image aspects, which equates to a comparison of SAR versus WASAR.

Also, it may be of interest to note, a target is considered detected if the detection algorithm returns a coordinate within 15 pixels of the known coordinate. Since the coordinates of the targets are known, it is possible to determine the pixel size of the returns associated with the targets. Several of the targets were observed in several aspect angles and they, the targets, were found to have returns that ranged in size from 12×12 up to 15×15 pixels. So a pixel patch of 20×20 was chosen to ensure that any pixels within this block would be grouped into the same target. Of course, this pixel patch would need to change with different size targets. The size of this patch also limits the target resolution capability; the ability to distinguish between two targets that are close together.

The four pre-processing techniques discussed in the last chapter are used as well as simply using the HH polarized image by itself. Detection after using these various pre-processing stages also determines whether any of the pre-processing techniques improve the target detection capability. Any large variations in the detection performance should stand out if the threshold value is varied over a wide range.

Figure 3.1 shows an enlarged view of 100 x 100 pixel area around a typical target within the images generated in this research. The target is the patch of high intensity returns in the center of the image. As stated previously this is an M-35 truck and there can be as many as 15 trucks in a scene at various target aspect angles to the probing radar system. Notice that there are other pixels within the image that are very close in intensity to the target pixels. Herein lies the crux of the target detection problem. If clutter pixels have intensities as high as the target's, then a simple threshold detector identifies these clutter pixels as targets, resulting in false alarms.

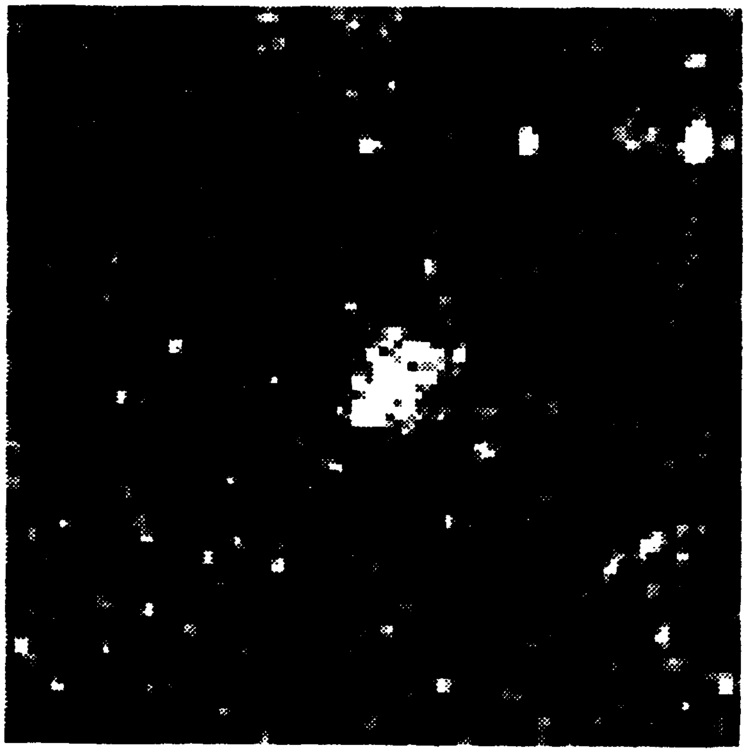
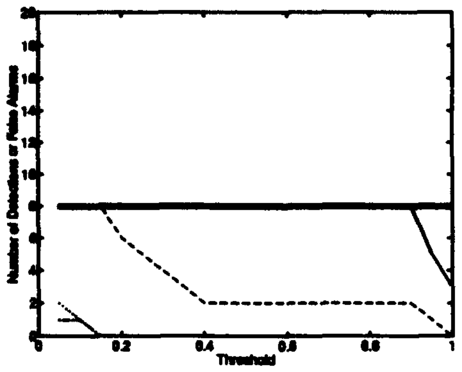


Figure 3.1 Enlargement of Typical Target

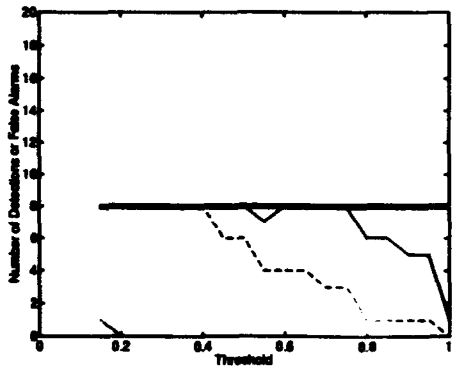
3.1 Scene 1: Targets in Grass

Figures 3.2 and 3.3 show the results of using multiple aspect angle detection. The figures show five different plots which indicate the results of using various pre-processing techniques. Notice that for the first scene, 8 targets are common to all of the rotated image sets. Also, notice that when targets are out in the open as they are in this scene, false alarms are not a big problem. When man-made objects are placed in grass (or shrubs for that matter), setting the threshold is not a major problem. Notice that false alarms do not occur in this algorithm until the threshold is set below 0.2. At this point the threshold is becoming lower than the intensity values present in the grassy area. With this obviously easy detection situation, the SAR and WASAR methods of detection work relatively the same, except that with WASAR, the threshold can operate over a much wider range and still detect all of the targets. Notice that both the SAR and WASAR are able at some point to detect all the targets and at the same time not realize any false alarms. This point in the detection and false alarm curves will not always be an easy situation to attain, as will be observed in scenes two and three to follow. The two detection techniques, pre-set in Figure 3.2 and scaled threshold in Figure 3.3, are fairly indistinguishable in detection performance. Both techniques are able to attain a point at which all the known targets are detected and no false alarms are indicated.

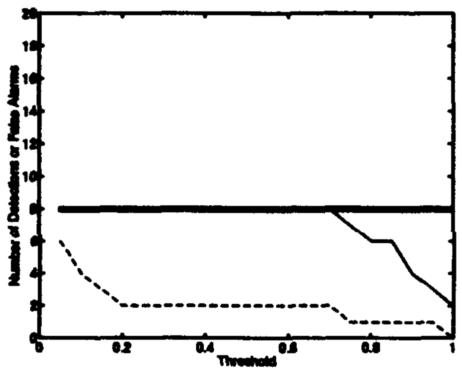
A particular anomaly of interest can be found in the Averaged plot in Figures 3.2. Notice that between 0.5 and 0.6 threshold values, the number of targets detected is 7; whereas, at points greater and less than these thresholds the number of targets detected is 8. This would seem intuitively impossible. It indicates that as the threshold is lowered at this point, instead of increasing the odds of detecting a target, a target is lost. This can be easily explained. As pixels are grouped together to form a target after pixel detection, their coordinates are simply averaged together. The grouping is within a 20 pixel block as explained previously. Now, if two targets happen to be very close, say within the 20 pixel block of each other or touching, these two targets will get grouped into one coordinate. This is what is known as target resolution, the ability to distinguish between two



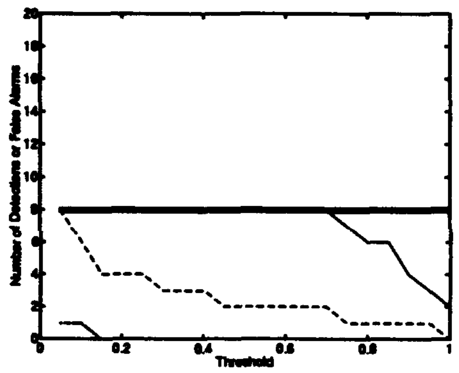
(a) HH only



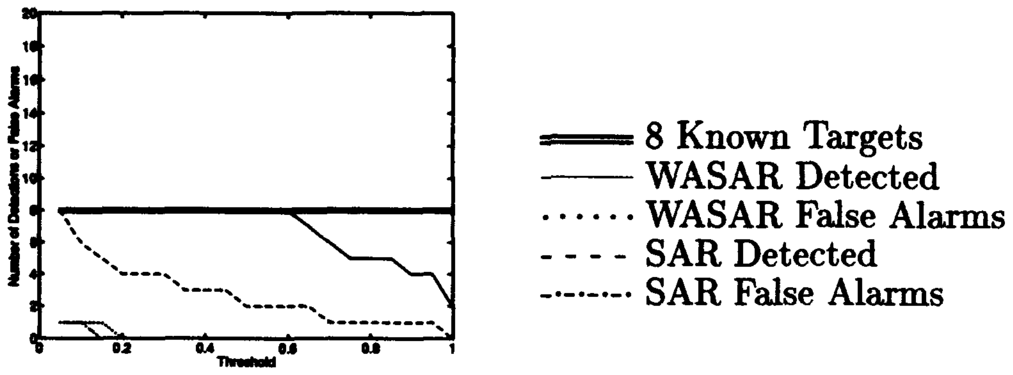
(b) Averaged



(c) Spanned

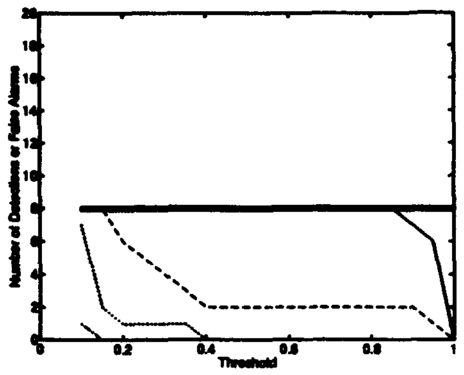


(d) Optimal

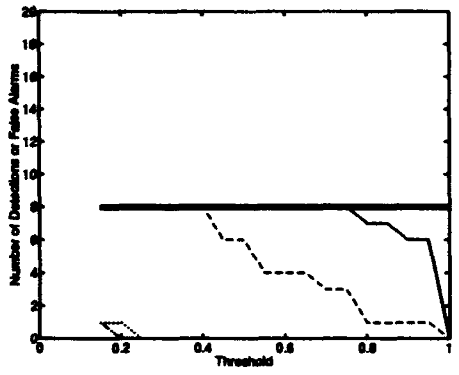


(e) Whitened

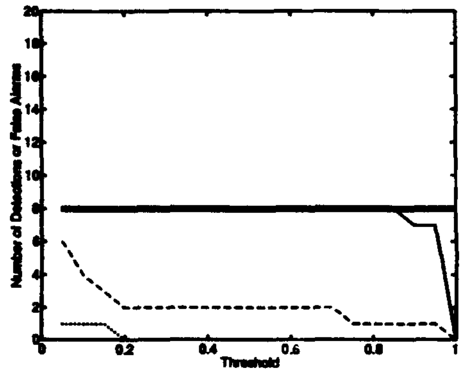
Figure 3.2 Detection Results for Targets in grass (pre-set threshold)



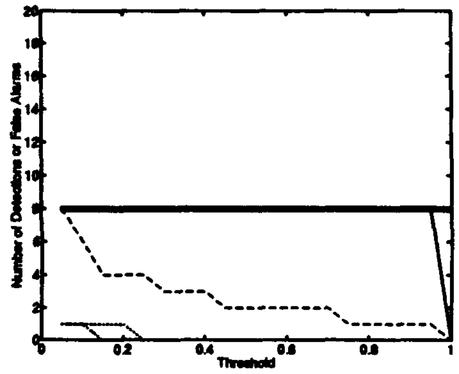
(a) HH only



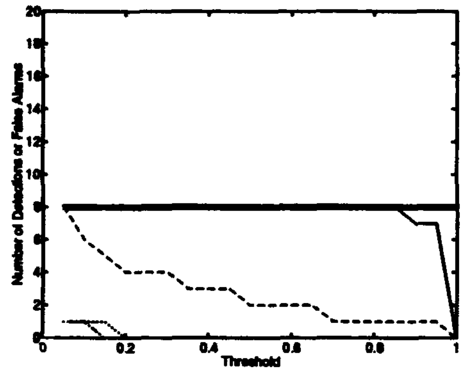
(b) Averaged



(c) Spanned



(d) Optimal



(e) Whitened

═══ 8 Known Targets
─── WASAR Detected
······ WASAR False Alarms
- - - - SAR Detected
-·-·-·- SAR False Alarms

Figure 3.3 Detection Results for Targets in grass (scaled threshold)

closely spaced targets. This definitely is a problem for a real world system, since, ideally, all targets should be identified and associated coordinates placed near the center of the target. However, for the purposes of this thesis, investigating the utility of WASAR over SAR, this problem is not fatal. All we need do is take note of the anomaly and ensure that the cause does not affect the algorithm, which it doesn't.

3.2 Scene 2: Targets in Trees and Grass (25% foliage)

Figure 3.4 shows an enlarged view of 100 x 100 pixel area around a typical tree within the images generated in this research. The tree is the patch of high intensity returns in the center of the image. All of the trees in the images in this research are deciduous trees in fall season with a varying trunk density between trees. The number of trees in the scene depends on the percentage of foliation selected during the generation of the image. Notice the similarity between the return for a tree, Figure 3.4, and that of a target, Figure 3.1. Both seem to be of the same intensity, though of different shapes. This can be a major problem in detecting targets buried in forest clutter. If the target does not have a considerably higher return than the trees, it can be very difficult to run a simple threshold detection technique over the image. On the fringe areas of Figure 3.4 more trees can be seen in the area. It may be of particular interest to note, the high intensity return on the upper edge near the left is a tree and a truck close together.

Figures 3.5 and 3.6 shows the results of using multiple aspect angle detection. Nine targets are known to be present in this scene. Threshold values are determined for the cross-over point, defined as the point at which the number of targets detected equals the number of false alarms. This point is important since at the cross-over, anything detected has a 50/50 chance of actually being a target. The cross-over point can be clearly seen in all of the plots for both the SAR and WASAR processed images. Notice that the false alarm rate, the rate at which the number of false alarms increases as the threshold value is decreased, seems to be the same for both SAR and WASAR

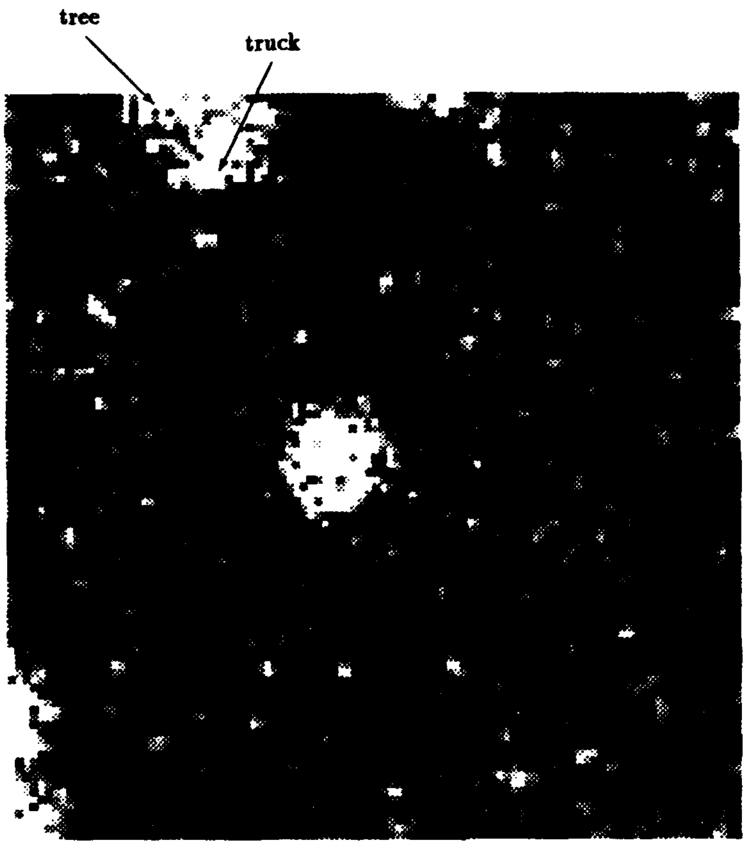
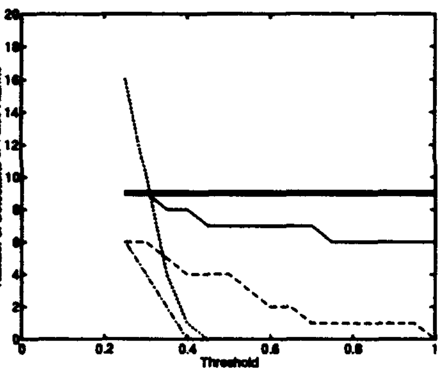
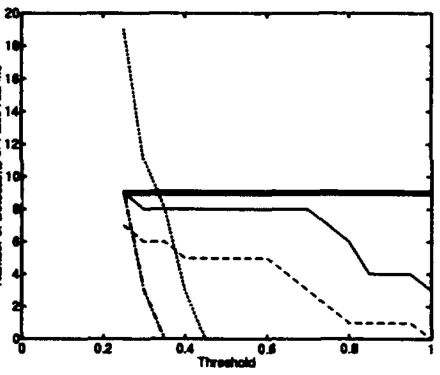


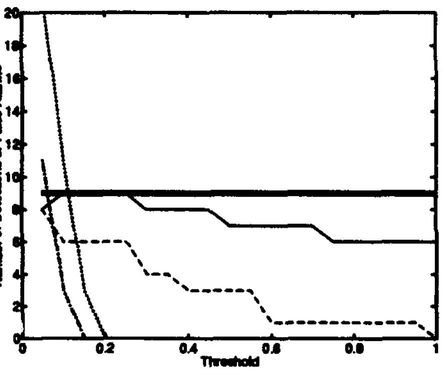
Figure 3.4 Enlargement of Typical Tree



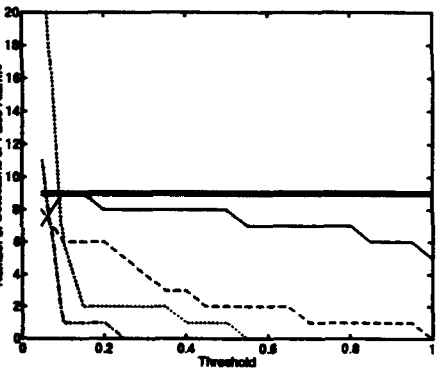
(a) HH only



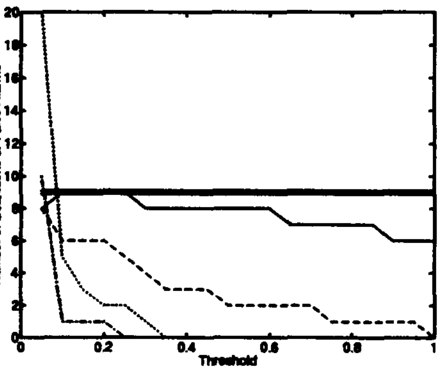
(b) Averaged



(c) Spanned



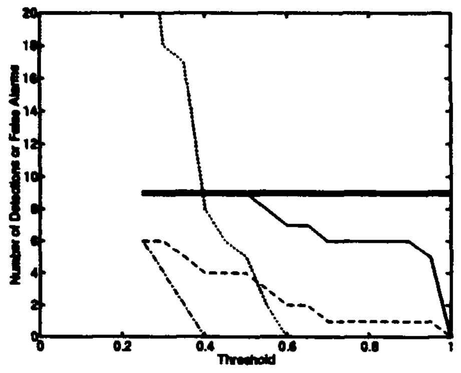
(d) Optimal



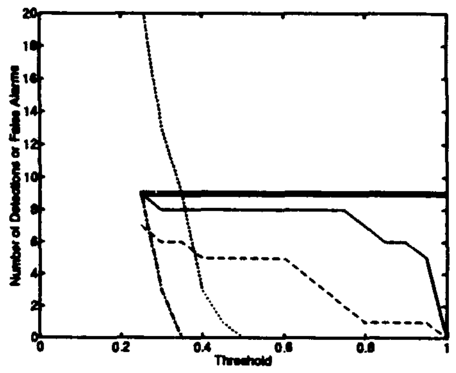
(e) Whitened

═══ 9 Known Targets
──── WASAR Detected
······ WASAR False Alarms
- - - - SAR Detected
-·-·-·- SAR False Alarms

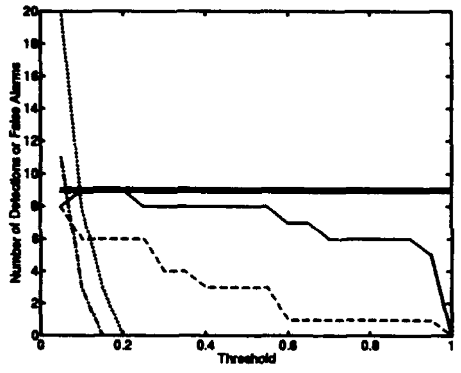
Figure 3.5 Detection Results for Targets in 25% foliage (pre-set threshold)



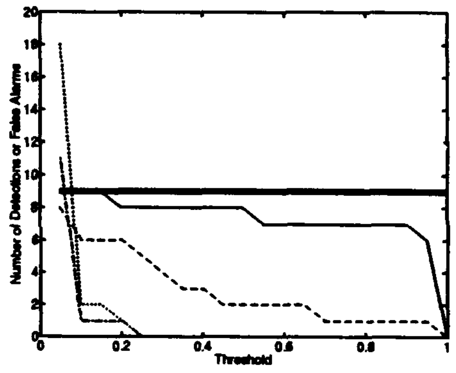
(a) HH only



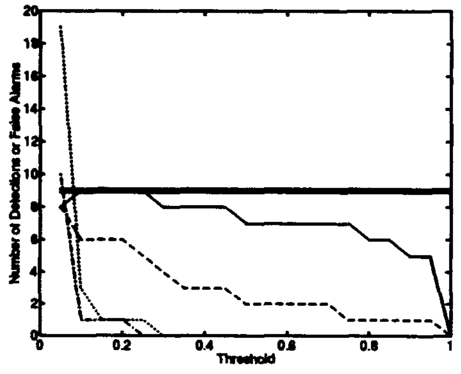
(b) Averaged



(c) Spanned



(d) Optimal



(e) Whitened

═══ 9 Known Targets
——— WASAR Detected
······ WASAR False Alarms
- - - - SAR Detected
-·-·-·- SAR False Alarms

Figure 3.6 Detection Results for Targets in 25% foliage (scaled threshold)

even though for WASAR the false alarms start at a higher threshold. The most important thing to notice is that at the higher thresholds, past the point at which false alarms occur, the WASAR is able to detect more of the targets on average than the SAR. This would support the hypothesis that the threshold can be set higher, to decrease false alarms, and still pick up most of the targets. These plots show that at some point within the various image aspects, most of the targets will "flare-up" higher than the surrounding clutter. So, with WASAR a few targets can be missed in a single aspect image and then picked up in another aspect.

Notice the difference in performance of the two detection methods. The first item to notice is that with scaled thresholding, the number of targets detected goes to zero when the threshold is raised to 1 (100%). This is to be expected on a single aspect thresholding system such as the scaled thresholding technique. As the threshold is raised toward the maximum pixel value, the number of targets detected drops off. The significant comparison on this point however is observed by using the preset thresholding technique charts. As can be clearly seen in Figure 3.5, the number of targets detected using WASAR does not drop to zero at 100% cut-off. This would indicate that in at least one of the images, the target pixel intensity (in absolute measurement) is higher than the maximum pixel intensity in the $+45^\circ$ image. Remember that in the preset thresholding technique the threshold is set using only the most positive aspect image in the image set. Also, notice that the various processing techniques yield varying results. The optimal weighting and the whitening filter both pull the false alarm curve zero cross-over to the right. For the start of the false alarm curves, the Spanned images seem to perform the best. The point at which the first false alarm occurs is quite low, at a threshold of 0.2.

Notice the differences in the performance of the detection algorithm on scene one, Figures 3.2 and 3.3, as compared to the performance on scene two, Figures 3.5 and 3.6. As the number of trees in the scene increase, the detection of targets becomes more difficult. This is not an unexpected result. As the targets become more and more obscure, it will be more difficult to pick them out.

3.3 Scene 3: Targets in Trees and Grass (50% foliage)

Figures 3.7 and 3.8 show the results of using multiple aspect angle detection. 10 targets are common to all of the image sets. Notice the difficulty with choosing an optimal threshold for this scene. At no point can we attain the ideal situation in which all the known targets are detected and no false alarms are indicated. Notice in Figure 3.7 that the Spanned case comes close. If the cutoff is set at 30% for a scaled threshold, all ten targets are detected and one false alarm results. A few of the other cases come close, but none realize the important goal of no false alarms and all targets detected. It is also interesting to note, that the WASAR seems to perform better than using normal SAR. WASAR comes closer in all cases to detecting all of the targets and except for a few cases, the false alarm rates appear to be similar; though, once again, WASAR false alarms begin at a higher threshold.

Notice the similarity of the detection performance between scenes two and three, Figures 3.5, 3.6, 3.7, and 3.8. Both are very similar in the rates at which detection falls off and the rate at which false alarms increase. This similarity would indicate that the amount of forestation does not impact the detection performance of these two techniques appreciably. Specific points of interest (points at which false alarms drop to zero, the cross-over point, and point at which all targets are detected) do differ though. As expected, when more trees are present, the false alarms start at higher cut-offs, the point at which all targets are detected starts at a lower cut-off, and so on.

The results from the previous sections indicate that WASAR definitely gives better performance over the use of single-aspect SAR. The actual performance of the detection algorithm is dependent on the amount of forestation present in an image. Also, the performance of the pre-set and scaled methods are fairly indistinguishable.

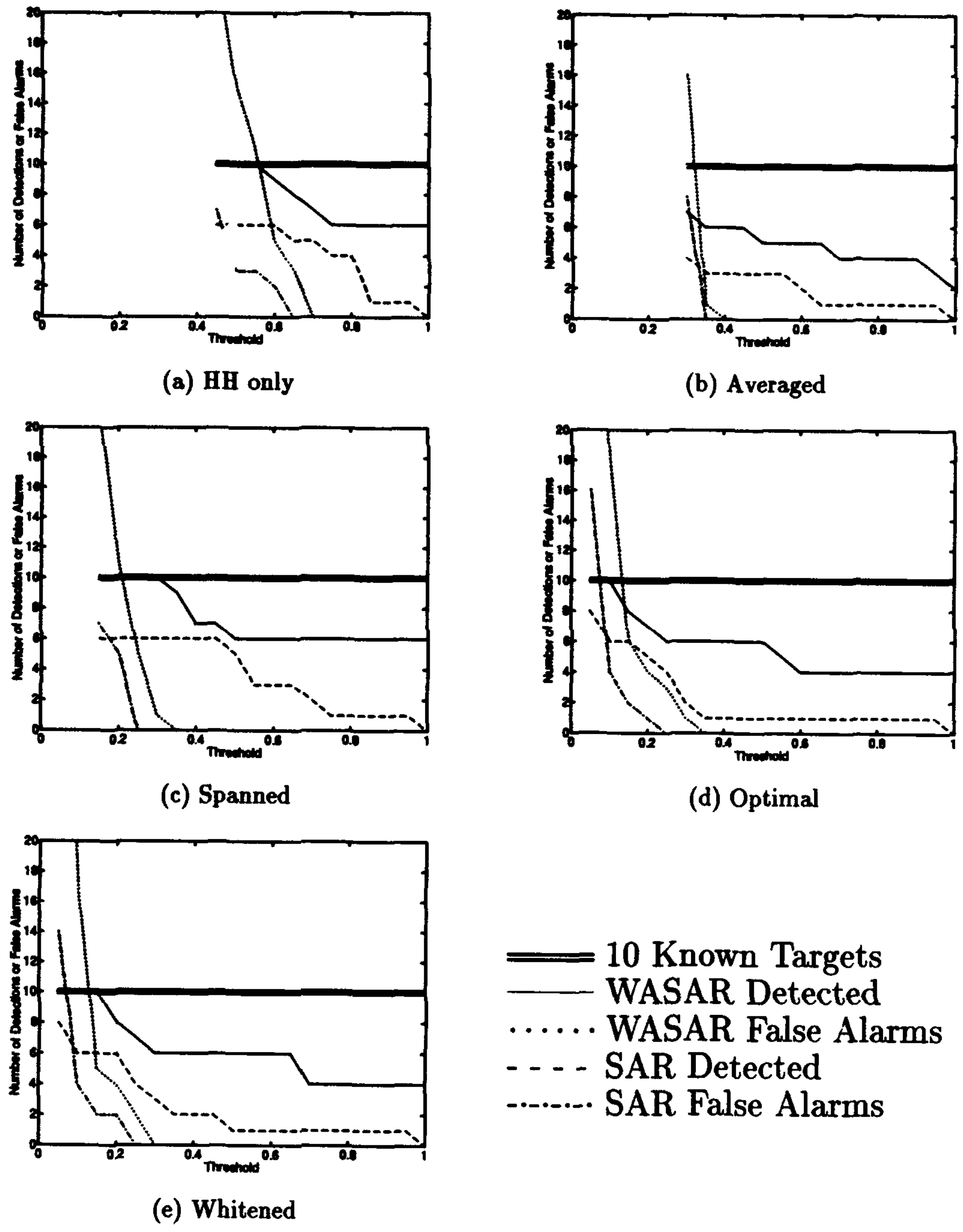
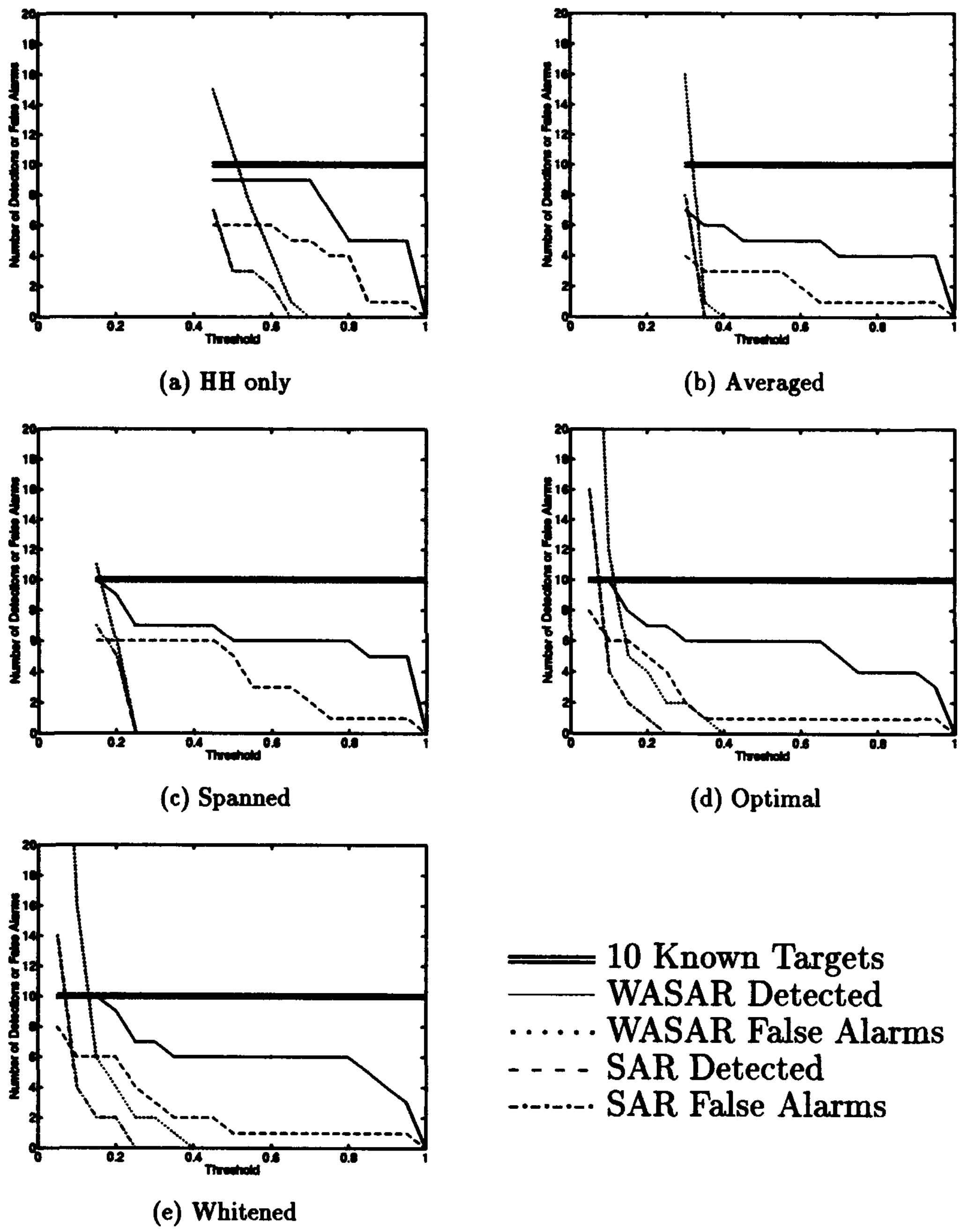


Figure 3.7 Detection Results for Targets in 50% foliage (pre-set threshold)



(a) HH only

(b) Averaged

(c) Spanned

(d) Optimal

(e) Whitened

Figure 3.8 Detection Results for Targets in 50% foliage (scaled threshold)

3.4 Subtraction Detection Method

Another detection technique is explored but the results are inconclusive. This other technique involves the comparison of the various aspect angles within WASAR by subtraction. In this research, the method of subtraction explored is one in which all aspects are subtracted individually from the most positive aspect image. The theory behind subtraction is that across the aspects, the returns of the clutter will not vary appreciably; whereas, the target returns will vary considerably from one aspect to another. A simple understanding of this approach can be gained from an example using the visible light spectrum. No matter from which aspect angle a tree is observed, the tree looks very similar. Now this is not to say that there aren't variations in the angular views. The cross-sectional size of the tree may change, the apparent extent of the foliage coverage may change, etc. However, generally speaking, the tree looks relatively the same. Now compare this to various angular views of say a semi-truck. The trucks cross-sectional size changes greatly from say a head on view compared to a side view. If the truck is a flat bed, it may appear on the average to be very tall from a front view but much lower on average from the side.

Now if we extend this example into the realm of radar waves, we can theorize as to the variation of the tree and truck while being illuminated by a radar beam. The main return from a tree is the result of the radar wave bouncing off of the tree trunk. In a simple sense, a tree trunk is a cylindrical object. If we illuminate this cylindrical object from various aspect angles perpendicular to the cylindrical axis, the return looks the same. The next highest return on trees are the result of the radar wave bouncing off of various sized and oriented branches. If we assume a random pattern to these branches, the conglomerate return of the combined branches will appear to be somewhat of a spherical shape, or conical in the case of coniferous trees. Either one of these shapes will clearly yield similar returns across various aspect angles. The last variable to account for is the leaves. The leaves will have a tendency to yield a much smaller return, but still be similar to the branches in conglomerate shape and size. Therefore, the return here is also similar across the aspect angles. So, a tree theoretically will have very little variation across aspect angles.

The truck on the other hand, when illuminated by the radar will vary greatly with aspect angle. We can assume that the truck is composed mainly of flat plates and we know from radar measurements that a flat plate yields its highest return for radar illuminations that are perpendicular to the surface. So, the truck will have high returns at the sides and fronts and smaller returns result when illuminated at skew angles of the flat sides and front – when the radar is coming in at a corner of the truck. Therefore, the truck has largely varying aspect returns.

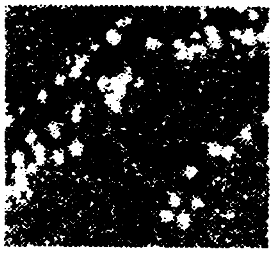
Now if we have a scene in which natural clutter and trucks are present and we view it from various angles, the clutter will appear to be similar in all scenes. However, the truck return will change from angle to angle. Theoretically if these two scenes were to be subtracted, the clutter return should go to zero and the target should remain relatively high. In the situation in this research in which the coordinates of the two images may be off by a pixel due to the rotation method used, the subtraction may not be as effective. However, if there is a mismatch of pixels, the bulk of each tree should still subtract with the possibility of a high intensity ring around the tree position that was not subtracted away.

Statistically speaking, subtraction is similar to a summing operation. In other words, the new mean of the subtracted images will be the difference of the individual means. If we assume that the two clutter scenes are identically distributed random variables then the mean becomes zero. The variance of the sum of two identically distributed random variables is twice the single image variance. So, this tells us that the clutter variation has become larger. Now the targets when subtracted will yield the difference between the two aspect angles. Difficulties at this point may arise if the difference in the target aspects is not greater than the variation in the clutter. In this case the subtracted target is being swamped by the new clutter variation.

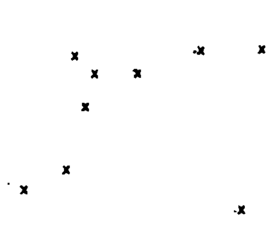
This subtraction technique is used for this research with only a slight variation. After subtraction of an individual image from the most positive image, a minimum pixel value is added back into the image to ensure that no zero value pixels are present. This is done purely for imaging purposes and in no way impacts on the usability of the results. After subtraction and addition of the minimum, the resultant image is run through a threshold detector. The detection method described earlier in this chapter as the scaled threshold method is used. This subtraction technique is explored only to see if any advantages existed over the more simple cumulative detection techniques discussed in the previous sections.

For this research, the subtraction method is used on scene two since it is somewhat of an average scene of the three generated. The results of subtraction on the HH rotated image data set are shown in Figure 3.9. The image in the center at the top is the original $+45^o$ HH image. All the images in this figure are displayed using an auto-scaled imaging technique. This means that the image intensities are automatically scaled to the highest and lowest pixel intensities to insure that all of the pixel values are displayed. This was necessary since after subtraction, the pixels will have a tendency to be grouped very tightly around a few intensity values with very little variation, as compared to an un-subtracted image. If the images were displayed using a simple intensity adjustment, the images would show very little detail since they would appear to be a uniform intensity across the entire image.

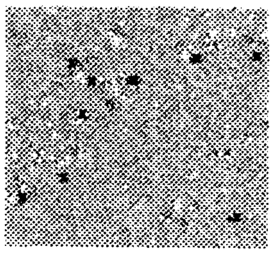
Notice in Figure 3.9, that the contrast in the images is very small. This small contrast indicates that our hypothesis on subtraction is correct, the clutter is subtracting from itself. In a few of the subtractions, the targets really stand out. Notice especially in the $+45^o-(0^o)$ subtraction, three of the targets can be clearly seen as high intensity pixels in three of the corners. In order to pick out the target locations in the darker images, compare the lighter images where the "Xs" can be clearly seen to the darker images. Notice also, that in some of the subtracted images, the trees do still have return values, $+45^o-(+15^o)$ and $+45^o-(-45^o)$.



HH: +45°



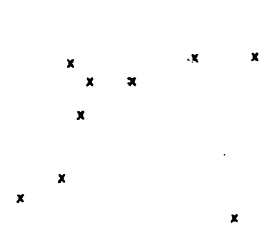
HH: +45° – (+30°)



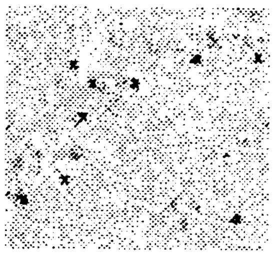
HH: +45° – (+15°)

HH: +45° – (0°)

HH: +45° – (–15°)

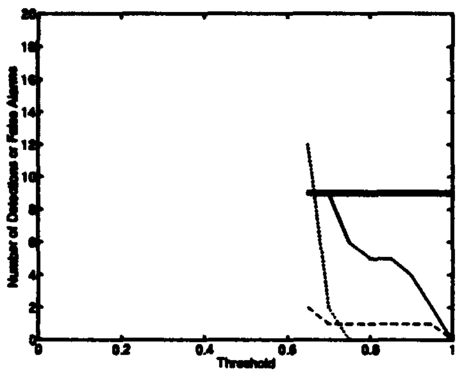


HH: +45° – (–30°)

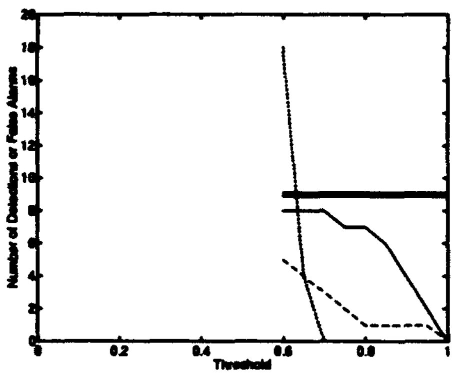


HH: +45° – (–45°)

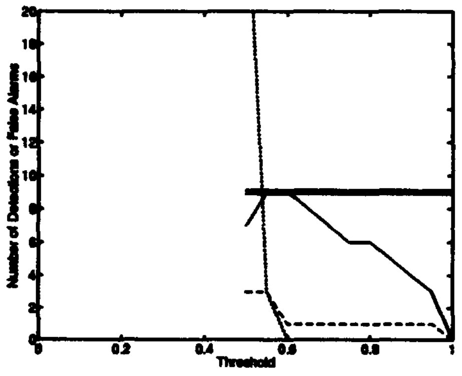
Figure 3.9 Images Displayed After subtraction



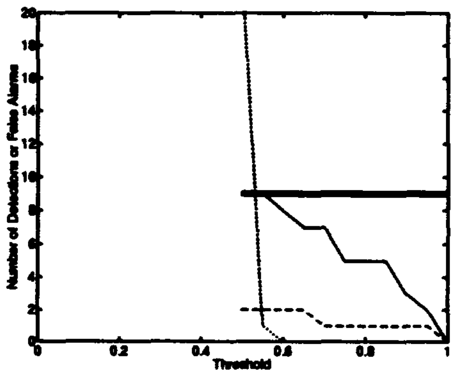
(a) HH only



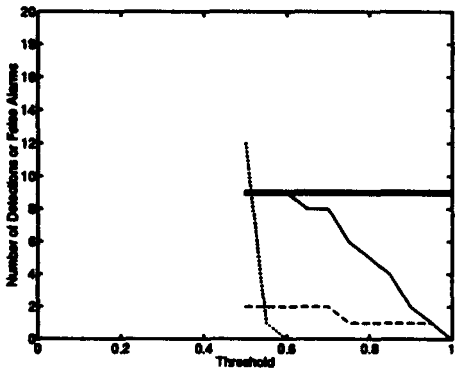
(b) Averaged



(c) Spanned



(d) Optimal



(e) Whitened

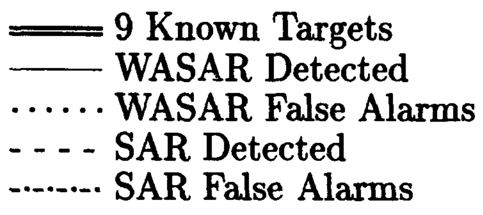


Figure 3.10 Detection Results for Targets in 25% foliage (subtraction)

Figure 3.10 indicates the results of the subtraction detection method. If these results are compared to those found in Figure 3.6 you will notice that the threshold values for subtraction are higher; meaning that for WASAR, for the same number of targets detected, the threshold must be set higher. This higher threshold value also occurs for the number of false alarms. Notice that the SAR curves, both number detected and number of false alarms, are exactly the same in both Figure 3.6 and 3.10. This is an expected result since for SAR only one image is available and no others are available for subtraction. Therefore, for the single-aspect SAR case, the method of detection is exactly the same as that used in the previous sections, and the results are exactly the same. The results in Figure 3.10 indicate that for this particular method of threshold detection, subtraction does not appreciably improve our odds of detection. However, these results do not exclude the possibility that another detection method on the subtracted images might improve the detection capability of WASAR.

IV. Findings and Conclusions

4.1 Results of This Research

The results from this research indicate that Wide-Angle Synthetic Aperture Radar definitely improves the odds of detection over the use of single aspect Synthetic Aperture Radar. In all of the detection methods used in this research, pre-set threshold, scaled threshold, and subtraction, the use of WASAR always performed better by detecting more of the targets over a larger range of threshold values.

One of the discouraging results of this research, is that the polarimetric processing techniques used in this thesis did not seem to improve our detection performance appreciably. In most of the cases researched, the HH alone performed just as well as any of the polarimetrically combined cases. This would seem intuitively incorrect, since more information is provided by the three polarimetrically diverse images, it should improve the odds of detection. The results did not support this.

As expected, L-band (1200 MHz) radar performed well for foliage penetration. For most of the cases tested in this research, the majority of the targets could be detected even if they were hidden under foliage. For large amounts of foliage, as in scene three, the false alarm rate climbs quickly and eventually defeats our detection algorithm at the cross-over point.

Lastly, this research indicated that subtraction of the aspect angles does show potential. The subtraction of the aspects indicates that the targets do vary from aspect to aspect; while, the clutter does not vary as much. The results from the pre-set and scaled threshold detection methods also support the conclusion that the targets will vary with aspect angle.

4.2 Future Work

The results of this thesis indicate that continuing work in this new facet of Synthetic Aperture Radar is very much needed. Numerous sub-areas of research could be explored more fully:

1. Use of more sophisticated detection techniques on the subtraction method discussed in the previous chapter, such as the use of a CFAR (Constant False Alarm Rate) detector.

2. Use of multiple depression angle images. This thesis used data sets that only contained images from a depression angle of 15°. It may be possible to use the diversity between depression angles in the same manner as the aspect diversity was used in this research.

3. Use of multiple frequency radar systems. The image sets generated for this research used only a single frequency, 1200 MHz. It may be possible that probing a scene with multiple frequencies, with comparison of the various returns, might yield some new and better detection techniques.

4. Use of various other target classes and mixture of these classes. This thesis contained only one type of target, M35 trucks. With future updates of the Loral software, more target classes will be available and these new target classes should be explored fully and any existing detection algorithms should be updated to include them.

5. Variation of seasons on the same scene. It is possible that variations in season will have a dramatic effect on the performance of any detection algorithm. Since only fall season was explored in this research, other seasons would need to be explored as well to verify any detection algorithm.

6. Use of netted and un-netted targets in the same scene. In a real world scene, the possibility definitely exists of having netted and un-netted targets within the same area. Any detection algorithm developed would ideally be able to pick out both types of targets.

7. Exploration of other polarimetric techniques and specific uses of HV and VV returns.

Research in this area could also be expanded into use of multiple sensor arrays in which not only radar data is used, but multi-spectrum, sonar, etc. are combined in an optimal target detection algorithm. Research in this area may also find application some day in non-military applications such as medical imaging. Here again, the list of possible non-military applications is endless; geological testing, structural integrity testing for civil engineering, oceanic biological tracking, etc. As can be clearly seen from these incomplete lists, this research is definitely not complete by any means, and won't be for many many years to come.

Appendix A. Review of Loral CTD Software

The software that was used in this research to generate the data was provide by Loral Corp. The software was provided pre-release as a courtesy. The ultimate purchaser is Wright Laboratory, Target Recognition Branch. The version that was made available at this time was a stripped down version of the software that will be made available to Wright Laboratory in the near future. The following paragraphs describe some of the capabilities of the software and how it was used for the research done in this thesis.

The SAR image data generation routine is a user interactive VAX-Fortran program that also uses resident data files in creating image data files. No imaging capabilities exist with this particular software package. The data is written to complex VAX-binary files. For our purposes, it was necessary to convert this data from binary into an ASCII file for transfer to a Sparc station for imaging and processing using the high-level language MATLAB®.

Once invoked, the data generation software will query the user for several pieces of information. The following is a list of the prompts:

1. PV-Wave available: This item was not available on the version that was delivered to us.
2. Debug Mode: The debug mode was supposed to be available in our version; however, though it was used a few times, it did not yield any information.
3. Local or regional statistical variance: Only local variance was available on our version.
4. Seed Value: This value specified the seed value used by the random number generator. If this value were changed, a different scene would result.
5. File Name: This file name was used as the prefix for the data files that would result. Each polarimetry, aspect angle, depression angle, and frequency would have a different suffix since it was a different image. The resultant output image file contained a .img extension.

6. Polarization Isolation: This is the amount of separation that was needed for the various polarimetries. A typical value was -20 dB.

7. Percentage of foliage extent: This value would indicate how much of the scene was covered by foliage. A maximum of 50% was allowed. If 0% were chosen, the image would be un-forested.

8. Frequency Mode: The number of frequencies that would be used in the simulated radar system. Choices were single, dual, triple, and quad.

9. Frequency Number: The frequency of the probing radar system. Values could range between 100 to 1500 MHz. User would be prompted for each frequency depending on number of different frequencies chosen in last item.

10. Constant resolution or constant integration aperture: This item is fairly self-explanatory. For the research in this paper, constant resolution was chosen in order to facilitate registering the images with each other.

11. Initial depression angle: This would give thc initial depression angle of the airborne or elevated SAR system. Values of 15^o to 60^o were allowed.

12. Number of depression angles: The number of different depression angles as explained in the previous item. The depression angles would begin with the initial angle and then be incremented 15^o for each number that is input at this stage.

13. Desired polarization: Self-explanatory. Choices were HH, VV, HV, or fully polarimetric.

14. View Aspects: This would indicate the different aspect angles for which a scene would be viewed from. Choices were: single pass, $+/- 45^o$, $+/- 30^o$, and $+/- 15^o$. When other than single pass was chosen, the 0^o angle was also generated. It was possible by entering all other parameters the same except this one to generate the same scene but with more aspect angles. So it was possible to have up to seven different viewing angles.

15. Reference image for change detection: This was supposed to generate the same scene with a temporal difference; however, on the version that we received the scene would not change.

16. Terrain type: Grass or short vegetation (shrubs). Only grass was available for the un-forested regions of the image.

17. Tree type: Coniferous, Deciduous, or a mixture of both could be specified.

18. Average leaf extent: This specified how much of an individual tree would be covered with leaves on the average. A typical value was 7 meters. Note: if 0% foliage extent were chosen in item 7, this number was ignored.

19. Season: Fall or summer could be chosen.

20. Large scene simulation: Large scene simulation was not available on this version.

21. Target classes: Allows the entering of no, one type, or multiple types of targets. The version provided to AFIT only had available none or one type. If none was selected, then the prompts would jump to item (26).

22. Target class: Targets were listed as small, medium, large, and corner reflectors. Small (M35 trucks) and corner reflectors were only available with our version. If the small target class is selected, the software attempts to place up to 15 targets in the images. The number that actually show up in the image is not directly controllable by the operator.

23. Netted targets: Self-explanatory.

24. All target conditions or deployed only: If deployed were selected, the targets would only appear in the forest. With all conditions, they may appear anywhere in the image.

25. Random or specified target aspects: Targets could only be placed randomly with the version available to us. If targets were selected, then fifteen M35 trucks would be randomly placed with random aspect angles.

26. Manual or random seeding of trees: Only random seeding of trees was available.

27. Trunk density in trunks per square meter: Trunk density up to about 0.01 trunk per square meter was allowed for our version.

28. Variable tree density: Self-explanatory.

29. Calibration error: This item not available.

30. ISLR: Not available.

31. Interference: Not available.

32. CTD icon: Not available.

33. Flight path: Not available.

34. Ground truth: Would result in a ground truth .pic image file. Though this file was generated, it could not be displayed in the same manner as the .img files; therefore, it could not be verified as to its accuracy.

35. Output image file: This would allow the writing of the file to the file name specified previously.

36. Integer/Complex/Compress: Specifies the format of the output file.

The software also had safeguards for the most part that would not allow the input of data that was outside the specified ranges. However, when certain routines were not available due to our having a limited version, the routine would die.

The software created several files depending on options. The .lis files that were created contained a listing of the target coordinates for that particular file. The .img files were the complex VAX-binary image data files. A .pic file may have been created if a ground truth image were requested.

After the image files were created, it was necessary to put them into some type of format that could be transferred to the Sparc station for manipulation in MATLAB®. This was accomplished by an encoding routine. This routine would take the binary files and write them to a .dat file that was in ASCII format. This file then could be in theory read by any machine.

The following parameters were used for all the data sets that were generated:

1. PV-Wave available: No
2. Debug Mode: No
3. Local or regional statistical variance: Local
4. Seed Value: 57
5. Polarization Isolation: -20.0
6. Frequency Mode: single (1)
7. Frequency Number: 1200
8. Constant resolution or constant integration aperture: resolution
9. Initial depression angle: 15
10. Number of depression angles: 1
11. Desired polarization: fully polarimetric (3)
12. View Aspects: all were generated (2,3,4)
13. Reference image for change detection: No
14. Terrain type: grass
15. Tree type: Deciduous (2)
16. Average leaf extent: 7.0
17. Season: Fall
18. Large scene simulation: No
19. Target classes: single (1)
20. Target class: small (3)
21. Netted targets: No
22. All target conditions or deployed only: all
23. Random or specified target aspects: random
24. Manual or random seeding of trees: random
25. Trunk density in trunks per square meter: 0.008
26. Variable tree density: yes
27. Calibration error: no
28. ISLR: no
29. Interference: no
30. CTD icon: no
31. Flight path: no
32. Ground truth: no
33. Output image file: no
34. Integer/Complex/Compress: complex

The following were the variant parameters for the first data set:

1. File Name: t1/
2. Percentage of foliage extent: 0.0

The following were the variant parameters for the second data set:

1. File Name: t2/
2. Percentage of foliage extent: 0.25

The following were the variant parameters for the third data set:

1. File Name: t3/
2. Percentage of foliage extent: .50

Appendix B. Computer Routines

B.1 VAX File

The following is a translation program that took the complex binary output of the Loral ctd software on a VAX and wrote it to an ASCII file for transfer to the MATLAB® routines which resided on a Sun workstation. This routine was written originally by Steve Kilberg of Loral and modified by Patti Ryan for our specific needs at AFIT.

```
c23456789012345667890 Program IQ_BYT.FOR

      complex      iq(512,512)
      real         max,min
      integer      n1,ns, lunit
      character*80 infile,outfile

      type *, 'Enter input filename'
      accept 200, infile
 200  format(a80)

      type *, 'Enter output filename'
      accept 200, outfile

      type *, 'Reading input file'
      call read_iq(infile,iq,512)
      type *, 'Input file read'

      ns = 512
      lout = 47
C open a new file for sequential formatted access
C  use format statement 300
      open(unit=lout,name=outfile,form='formatted', status = 'new')

 300  format(g10.3,2x,g10.3)
 400  format(/)


C  another possibility
C 300  format(g10.3)


      do 10 i = 1,512

          write(lout, 300) (iq(j,i), j = 1,512)
          write(lout,400)
  10  continue


C try opening a new file for unformated access
```

```
C open(unit=lunit, name=outfile, form='unformatted', status = 'new')
C
C print out loop for unformatted

C      do 10 i = 1,512
C         write(6,*)(iq(j.i), j=1,512)
C 10  continue

      close(lunit)
      stop
      end

C*************************************************************

      subroutine read_iq(infile,array,imgsize)

      complex        array(512,512)
      integer        luin,nline,nsamp,imgsize
      character*80   infile

      nline = imgsize
      nsamp = imgsize

      luin = 66
      open(unit=luin,name=infile,form='unformatted',
     &     access='direct',recordtype='fixed',status = 'old')

      do i = 1,nline
read(luin 'i) (array(j,i), j = 1,nsamp)
      end do

      close(unit = luin)

      return
      end
C*************************************************************
```

B.2 Sun Files

The scripts contained in this appendix are the MATLAB® routines that were used to load, process, and display the image data that was generated by the Loral software. The first routine is what could be called the main program. This is the routine that is invoked every time. *sar.m* contains the menu that calls all the other major routines. Each file contains a header which explains the purpose of the file. Comments on particular phases of the code were not added where text output messages explained the procedure. Maximum run times are hard-coded and would necessarily be cpu load dependent. Quirks do not necessarily indicate problems with the code, they might simply be items of particular interest for that routine or function.

MATLAB® does have an imaging quirk that was discovered during this research. MATLAB® does not display images in the same coordinate system as it plots. When using the "image()" function to display the image matrices, the displayed figure will transpose the image. But, when a plot of target coordinates is overlayed on this image, the coordinates are plotted true to axis coordinates. So, a plot and an image in the same figure will not use the same coordinates. In order to correct for this idiosyncrasy, it was necessary to transpose the image before displaying so that plotting could be overlayed on the images. For the particular coding involved, observe the file called "display.m".

The code that has been written for this research is not optimal in any sense of the word. The code was written for my understanding primarily, for optimization of available memory (swap and disk) secondly, and speed lastly. Since the purpose of this thesis was conceptual, coding optimization was not given a high priority; however, since MATLAB® is a high-level C language, direct conversion should be no problem. After conversion, optimization might best be handled by a software engineer.

```
%*****************************************************************************
%     ROUTINE sar.m
%*****************************************************************************
%  Purpose: Main menu routine.  Creates a figure for display of menu and other
%           data.
%
%  Called by: MatLab
%  Calls: files1.m, input_data.m, filt.m, detection.m, show_stat.m,
%         display1.m, display2.m, del1.m, del2.m
%
%*****************************************************************************

clear all

% check for existence of "files.mat"; if it doesn't exist, create it
if exist('files.mat') ~= 2
  files1
  save files directory file_name prep detect
end
clear

% Menu title
figure('Position',[635,5,515,400], ...
       'Name','SAR Processing','NumberTitle','off');

% menus
menu1 = uimenu('Label',' Input  ');
  uimenu(menu1,'Label','Input .dat files','Callback','input_data');

menu1 = uimenu('Label',' Pre-processing ');
  uimenu(menu1,'Label','None','Callback','filt(0)');
  uimenu(menu1,'Label','Rotation Only','Callback','filt(1)');
  uimenu(menu1,'Label','Average','Callback','filt(2)');
  uimenu(menu1,'Label','Span','Callback','filt(3)');
  uimenu(menu1,'Label','Optimal Weighting','Callback','filt(4)');
  uimenu(menu1,'Label','Polarimetric Whitening','Callback','filt(5)');
  uimenu(menu1,'Label','Create all', ...
        'Callback','filt(0);filt(1);filt(2);filt(3);filt(4);filt(5)');

menu1 = uimenu('Label',' Detection ');
  uimenu(menu1,'Label','None','Callback','detection(0)');
  uimenu(menu1,'Label','Multiple Detection (preset threshold)', ...
              'Callback','clear all;detection(2)');
  uimenu(menu1,'Label','Multiple Detect    (scaled threshold)', ...
              'Callback','clear all;detection(1)');
  uimenu(menu1,'Label','Subtraction', ...
              'Callback','clear all;detection(3)');

menu1 = uimenu('Label',' Statistics ');
  uimenu(menu1,'Label','Statistics on images','Callback','show_stat');
```

```
menu1 = uimenu('Label',' Output ');
  uimenu(menu1,'Label','Display images','Callback','display1(0)');
  uimenu(menu1,'Label','Display pwf images','Callback','display1(-1)');
  uimenu(menu1,'Label','Display images (autoscale)','Callback','display1(1)');
  uimenu(menu1,'Label','Create eps file','Callback','display2');

menu1 = uimenu('Label',' Exit ');
  uimenu(menu1,'Label','Exit from Matlab','Callback','quit');
  uimenu(menu1,'Label','Exit to Matlab','Callback','del1;clear;delete(1)');
  uimenu(menu1,'Label','Reset', ...
         'Callback','del1;clear;delete(1);sar;');

menu1 = uimenu('Label',' Cleanup ');
  uimenu(menu1,'Label','Clear figures','Callback','del1');
  uimenu(menu1,'Label',' Delete .mat files','Callback','del2;');

clear menu1

%***************************************************************************




%***************************************************************************
%     ROUTINE files1.m
%***************************************************************************
%  Purpose: Used to initialize files.mat.  This file must be changed when a
%           new data set is being loaded.  All .mat files need to be deleted
%           from current directory.
%
%  Called by: sar.m
%
%  Quirks:
%    1) Variables are hard-coded.  File must be edited to update
%
%***************************************************************************

directory = '/tmp_mnt/home/hawkeye8/kknurr/images/ctdimages';
file_name = ['t2_p45' ; 't2_p30' ; 't2_p15' ;'t2_000' ; ...
             't2_m15' ; 't2_m30' ; 't2_m45'];
prep = 'non';
detect = 0;

%***************************************************************************
```

```
%***************************************************************************
%     ROUTINE input_data.m
%***************************************************************************
% Purpose:  Loads data from .dat (ASCII) files and creates an unrotated
%           image matrix.  Also, loads target coordinates.  Fully polar
%           images are then stored in one of seven unrotatedX.mat files
%
% Called by: sar.m
% Calls: targlis.m
%
% Quirks:
%   1) Directory changes are hard-coded
%
%***************************************************************************

time = fix(clock);

% setup message window
  figure(1)
  cla
  hold on
  axis('xy','off')
  mess1 = text(.1,.6,' ');
  tmess = text(.1,1,' ');
  amess = text(.1,.95,' ');

set(tmess,'String',['Starting Time: ' num2str(time(1,4)) ':' ...
                    num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(amess,'String','Max Run Time: 90 minutes'); pause(1)

if exist('files.mat') ~= 2
  set(mess1,'String','Files.mat information must be loaded first')
    pause(5)
  figure(1)
  cla
  hold off

else
  load files
  prep = 'non';
  save files file_name directory prep detect

% load images three at a time (one aspect at a time)
for i = 1 : size(file_name,1)

  set(mess1,'String',['Loading ' file_name(i,:) ' Target Coordinates'])
    pause(2)
    targets = targlis( [file_name(i,:) '_dat.lis'] );

%***********change directories (hard code)**********
 cd /tmp_mnt/home/hawkeye8/kknurr/images/ctdimages
```

```
%******************************************************
  set(mess1,'String',['Loading ' file_name(i,:) ' HH polarimetry'] )
    pause(1)
    fid = fopen([file_name(i,:) '_hh.dat'],'r');
    hh = fscanf(fid,'%f');
    status = fclose(fid);
  set(mess1,'String',['Loading ' file_name(i,:) ' HV polarimetry'] )
    pause(1)
    fid = fopen([file_name(i,:) '_hv.dat'],'r');
    hv = fscanf(fid,'%f');
    status = fclose(fid);
  set(mess1,'String',['Loading ' file_name(i,:) ' VV polarimetry'] )
    pause(1)
    fid = fopen([file_name(i,:) '_vv.dat'],'r');
    vv = fscanf(fid,'%f');
    status = fclose(fid);
  clear fid status
%***********change directories (hard code)**********
 cd /tmp_mnt/home/hawkeye8/kknurr/images
%******************************************************

  set(mess1,'String',['Creating ' file_name(i,:) ' HH image file'] )
    pause(1)
    hh = hh(1:2:length(hh)-1) + j*hh(2:2:length(hh));
    hh = reshape(hh,sqrt(length(hh)),sqrt(length(hh)));
  set(mess1,'String',['Creating ' file_name(i,:) ' HV image file'] )
    pause(1)
    hv = hv(1:2:length(hv)-1) + j*hv(2:2:length(hv));
    hv = reshape(hv,sqrt(length(hv)),sqrt(length(hv)));
  set(mess1,'String',['Creating ' file_name(i,:) ' VV image file'] )
    pause(1)
    vv = vv(1:2:length(vv)-1) + j*vv(2:2:length(vv));
    vv = reshape(vv,sqrt(length(vv)),sqrt(length(vv)));

  set(mess1,'String',['Saving as unrotated' num2str(i) '.mat'] )
    pause(1)
  % Select file name
  if i == 1
    save unrotated1 hh hv vv targets
  elseif i == 2
    save unrotated2 hh hv vv targets
  elseif i == 3
    save unrotated3 hh hv vv targets
  elseif i == 4
    save unrotated4 hh hv vv targets
  elseif i == 5
    save unrotated5 hh hv vv targets
  elseif i == 6
    save unrotated6 hh hv vv targets
  elseif i == 7
    save unrotated7 hh hv vv targets
  end

  clear hh hv vv targets
end

set(mess1,'String','Input of files complete')
  pause(5)
  figure(1)
  cla
  hold off

end

clear

%*********************************************************************
```

```
%*****************************************************************************
%     FUNCTION targlis.m
%*****************************************************************************
%  Purpose:  Loads data from .lis (ASCII) files and creates an unrotated
%            target coordinate matrix. Strips off header and retains only the
%            coordinate information from file as it is formated by ctd
%            software.
%
%  Called by: input_data.m
%
%  Quirks:
%    1) Directory changes are hard-coded
%*****************************************************************************

function [targets] = targlis(file_name)

%***********change directories (hard code)**********
 cd /tmp_mnt/home/hawkeye8/kknurr/images/ctdimages
%**************************************************

  fid = fopen(file_name,'r');

% strip off header
  read_in = 'n';
  while read_in ~= ';'
    read_in = fscanf(fid,'%s',1);
    header = [header read_in];
  end
  read_in = fscanf(fid,'%s',5);

% read in target coordinates
    read_in = fscanf(fid,'%s',1);
  while read_in ~= []
    targets = [targets ; fscanf(fid,'%f',1) fscanf(fid,'%f',1)];
    read_in = fscanf(fid,'%s',10);
    read_in = fscanf(fid,'%s',1);
  end

status = fclose(fid);

%***********change directories (hard code)**********
 cd /tmp_mnt/home/hawkeye8/kknurr/images
%**************************************************

clear fid status header read_in

%*****************************************************************************
```

```
%****************************************************************************
%     ROUTINE rot.m
%****************************************************************************
%  Purpose: Rotates .mat files that are created by input_data.m.
%
%  Called by: sar.m
%  Calls: rotation.m
%
%  Quirks:
%    1)  Rotation amount is hard-coded for 7 aspect angles at 15 degree
%        increments.  Automatic coding would not work (see anotated code).
%
%****************************************************************************

time = fix(clock);

% setup message windows
  figure(1)
  cla
  hold on
  axis('xy','off')
  mess1 = text(.1,.6,' ');
  tmess = text(.1,1,' ');
  amess = text(.1,.95,' ');
  trmess = text(.1,.9,' ');
  armess = text(.1,.85,' ');

set(tmess,'String',['Starting Time: ' num2str(time(1,4)) '.' ...
                     num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(amess,'String','Max Run Time: 260 minutes');
set(armess,'String','Max Rotation Time: 13 minutes'); pause(1)

load files

if exist('unrotated1.mat') ~= 2  % else go to end
  set(mess1,'String','Unrotated files are not in this directory')
    pause(5)
  figure(1)
  cla
  hold off

else
% load images three at a time
for i = 1 : size(file_name,1)

  set(mess1,'String',['Loading unrotated' num2str(i) '.mat'] )
    pause(1)
  if i == 1
    load unrotated1
  elseif i == 2
    load unrotated2
  elseif i == 3
    load unrotated3
  elseif i == 4
    load unrotated4
  elseif i == 5
    load unrotated5
  elseif i == 6
    load unrotated6
  elseif i == 7
    load unrotated7
  end

%*********************************************************************
% The following rotation angle determination is broken, angle must be
% hardcoded at this point.  This code attempted to use the filename
% information to determine amount of rotation.
%*********************************************************************
  % Determine angle of rotation
%    if i > 1
%      for j=1:size(file_name,2)
%        if file_name(i,j)=='_' ...
%           & ( file_name(i,j+1)=='p' | file_name(i,j+1)=='0' )
%          number = abs(file_name(i,j+2:j+3));
%          number = round(abs( abs(file_name(1,j+2:j+3)) - number ));
%        elseif file_name(i,j)=='_' & file_name(i,j+1)=='m'
%          number = -abs(file_name(i,j+2:j+3));
%          number = round(abs( abs(file_name(1,j+2:j+3)) - number ));
%        end
%      end
%    else
%      number = 0;
%    end
%if i == 1
%  number = 0;
%elseif i == 2
%  number = 3;
%elseif i == 3
%  number = 6;
%end

number = i-1;  % hard-coding for 7 aspect angles

%****************************************************

  % rotate over "number" of 15 degree increments

  time = fix(clock);
  set(trmess,'String',['Rotation start Time: ' num2str(time(1,4)) ':' ...
                       num2str(time(1,5)) ':' num2str(time(1,6)) ], ...
                       'Visible','on');
  set(mess1,'String',['Rotating ' file_name(i,:) ' target coordinates'])
  pause(1)
temp = 0 * ones(512,512);
for j=1:size(targets,1)
  if targets(j,1) ~= 0
    temp( targets(j,1),targets(j,2) ) = 255;
  end
end
temp = rotation(temp,number);
[k,l] = find(temp==255);
targets = [ k l ];
clear j k l temp

time = fix(clock);
set(trmess,'String',['Rotation start Time: ' num2str(time(1,4)) ':' ...
                   num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(mess1,'String',['Rotating ' file_name(i,:) ' HH image file'])
  pause(1)
  hh = rotation(hh,number);

time = fix(clock);
set(trmess,'String',['Rotation start Time: ' num2str(time(1,4)) ':' ...
                   num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(mess1,'String',['Rotating ' file_name(i,:) ' HV image file'])
  pause(1)
  hv = rotation(hv,number);

time = fix(clock);
set(trmess,'String',['Rotation start Time: ' num2str(time(1,4)) ':' ...
                   num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(mess1,'String',['Rotating ' file_name(i,:) ' VV image file'] )
  pause(1)
  vv = rotation(vv,number);

set(trmess,'Visible','off');

set(mess1,'String',['Saving as rotated' num2str(i) '.mat'] )
  pause(1)
if i == 1
  save rotated1 hh hv vv targets
elseif i == 2
  save rotated2 hh hv vv targets
elseif i == 3
  save rotated3 hh hv vv targets
elseif i == 4
  save rotated4 hh hv vv targets
elseif i == 5
  save rotated5 hh hv vv targets
elseif i == 6
  save rotated6 hh hv vv targets
elseif i == 7
  save rotated7 hh hv vv targets
```

```
    end

    clear hh hv vv targets
  end

  set(mess1,'String','Rotation of files complete')
    pause(5)
    figure(1)
    cla
    hold off

  prep = 'rot';
  save files directory file_name prep detect

  end

  %*****************************************************************************
```

```
%**************************************************************************
%     FUNCTION rotation.m
%**************************************************************************
%  Purpose:  Rotates an image input by "number" of fifteen degree increments.
%            Also cuts image down to overlapping square region.
%
%  Called by: rot.m
%
%  Quirks:
%    1) Uses short cut for 0 and +/- 90 degree rotations.
%
%**************************************************************************

function [temp2] = rotation(temp1,number)

% number indicates the number of 15 degree increment rotations.
  phi = number * (-pi/12);   % angle of rotation

% (a,b) central point of rotation
  a = round(size(temp1,1)/2);
  b = round(size(temp1,2)/2);

if phi == 0
  temp2 = temp1;  % 0 degree short cut
elseif phi == pi/2
  temp2 = rot90(temp1,-1);  % 90 degree short cut
elseif phi == -pi/2
  temp2 = rot90(temp1,1);  % 90 degree short cut
else
  temp2 = min(min(temp1)) * ones(size(temp1,1),size(temp1,2));
  for i=1:size(temp2,1)
    for j=1:size(temp2,2)
      r = sqrt((i-a)^2 + (j-b)^2);
      if i==a & j>=b
        theta = 0;
      elseif i==a & j<b
        theta = pi;
      else
        theta = atan2((j-b),(i-a));
      end
      x = round(r * cos(theta + phi)) + a;
      y = round(r * sin(theta + phi)) + b;
      if x>=1 & x<=size(temp1,1) & y>=1 & y<=size(temp1,2)
        temp2(i,j) = temp1(x,y);
      end
    end
  end
end

% cutting down image to square image of overlapping region
  l = a - round(sqrt( (a^2) / 2 ));
  u = size(temp1,1) - 1;
  temp2 = temp2(1:u,1:u);

%******************************************************************************
```

```
%******************************************************************************
%     FUNCTION filt.m
%******************************************************************************
%  Purpose:  Selects the type of pre-processing that is used.  If the
%            pre-processed files do not exist, it creates them.
%
%  Called by: sar.m
%  Calls: ave.m, span.m, ows.m, pwf.m
%
%******************************************************************************

function [] = filt(newprep)

time = fix(clock);

% setup message window
  figure(1)
  cla
  hold on
  axis('xy','off')
  mess = text(.1,.6,' ');
  tmess = text(.1,1,' ');
  amess = text(.1,.95,' ');

set(tmess,'String',['Starting Time: ' num2str(time(1,4)) ':' ...
           num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(amess,'String','Max Run Time: 5 minutes'); pause(1)

load files
  if newprep == 0
    prep = 'non';
    if exist('unrotated1.mat') ~= 2
      input_data
    end
  elseif newprep == 1
    prep = 'rot';
    if exist('rotated1.mat') ~= 2
      rot
    end
  elseif abs(newprep) == 2
    prep = 'ave';
    if exist('averaged.mat') ~= 2
      newprep = -1;
    end
  elseif abs(newprep) == 3
    prep = 'spn';
    if exist('spaned.mat') ~= 2
      newprep = -1;
    end
  elseif abs(newprep) == 4
    prep = 'ows';
    if exist('optimal.mat') ~= 2
      newprep = -1;
    end
  elseif abs(newprep) == 5
    prep = 'pwf';
    if exist('whitened.mat') ~= 2
      newprep = -1;
    end
  end

save files directory file_name prep detect

if newprep < 0          % process the files, otherwise files will be loaded
if prep=='ave' | prep=='spn' | prep=='ows' | prep=='pwf'
  for i = 1:size(file_name,1)

    set(mess,'String',['Loading rotated ' file_name(i,:)])
      pause(1)
    if i == 1
      load rotated1
    elseif i == 2
      load rotated2
    elseif i == 3
      load rotated3
    elseif i == 4
      load rotated4
    elseif i == 5
      load rotated5
    elseif i == 6
      load rotated6
    elseif i == 7
      load rotated7
    end

    set(mess,'String',['Pre-processing ' file_name(i,:)])
      pause(1)
    if prep == 'ave'
      temp = ave(hh,hv,vv);
    elseif prep == 'spn'
      temp = span(hh,hv,vv);
    elseif prep == 'ows'
      temp = ows(hh,hv,vv);
    elseif prep == 'pwf'
      temp = pwf(hh,hv,vv);
    end

    if i == 1
      aspect1 = temp;
    elseif i == 2
      aspect2 = temp;
    elseif i == 3
```

```
      aspect3 = temp;
    elseif i == 4
      aspect4 = temp;
    elseif i == 5
      aspect5 = temp;
    elseif i == 6
      aspect6 = temp;
    elseif i == 7
      aspect7 = temp;
    end
    clear temp

  end
end
  if prep == 'ave'
    set(mess,'String','Saving as averaged')
      pause(1)
    save averaged aspect1 aspect2 aspect3 aspect4 ...
                  aspect5 aspect6 aspect7 targets
  elseif prep == 'spn'
    set(mess,'String','Saving as spaned')
      pause(1)
    save spaned   aspect1 aspect2 aspect3 aspect4 ...
                  aspect5 aspect6 aspect7 targets
  elseif prep == 'ows'
    set(mess,'String','Saving as optimal')
      pause(1)
    save optimal  aspect1 aspect2 aspect3 aspect4 ...
                  aspect5 aspect6 aspect7 targets
  elseif prep == 'pwf'
    set(mess,'String','Saving as whitened')
      pause(1)
    save whitened aspect1 aspect2 aspect3 aspect4 ...
                  aspect5 aspect6 aspect7 targets
  end

end

% Finish up
  set(mess,'String','Processing Complete')
    pause(5)
  figure(1)
  cla
  hold off

%******************************************************************************
```

```
%****************************************************************************
% FUNCTION ave.m
%****************************************************************************
%  Purpose: Averages the polarimetric images to create a new image.
%
%  Called by: filt.m
%
%***************************************************************************

function [average] = ave(hh,hv,vv)

    average = (hh + hv + vv)/3;

%****************************************************************************
```

```
%****************************************************************************
%     FUNCTION span.m
%****************************************************************************
%  Purpose: Creates a span image from the polarimetric images.
%
%  Called by: filt.m
%
%***************************************************************************

function [sp] = span(hh,hv,vv)

    sp =  abs(hh).^2 + 2*(abs(hv).^2) + abs(vv).^2;

%****************************************************************************
```

```
%******************************************************************************
%     FUNCTION ows.m
%******************************************************************************
% Purpose:  Creates an Optimally Weighted Sum of the polar images.
%
% Called by: filt.m
%
%*****************************************************************************

function [y] = ows(hh,hv,vv)

  rho = mean(mean(hh.*conj(vv))) ...
        /sqrt( mean(mean(abs(hh).^2)) .* mean(mean(abs(vv).^2)) );
  epsilon = mean(mean(abs(hv).^2)) / mean(mean(abs(hh).^2));
  gamma = mean(mean(abs(vv).^2)) / mean(mean(abs(hh).^2));
  k2 = (1 + abs(rho)^2)/epsilon;
  k3 = 1/gamma;

  y =  abs(hh).^2 + k2*(abs(hv).^2) + k3*(abs(vv).^2);

%******************************************************************************


%******************************************************************************
%     FUNCTION pwf.m
%******************************************************************************
% Purpose: Creates a Polarimetrically Whitened image from the polar images
%
% Called by: filt.m
%
%*****************************************************************************

function [y] = pwf(hh,hv,vv)

  rho = mean(mean(hh.*conj(vv)))...
        /sqrt( mean(mean(abs(hh).^2)) .* mean(mean(abs(vv).^2)) );
  epsilon = mean(mean(abs(hv).^2)) / mean(mean(abs(hh).^2));
  gamma = mean(mean(abs(vv).^2)) / mean(mean(abs(hh).^2));
  sigma = mean(mean((abs(hh)).^2));

  k1 = 1/(sigma * (1 - (abs(rho))^2));
  k2 = 1/(sigma * epsilon);
  k3 = 1/(sigma * (1 - (abs(rho))^2) * gamma);
  k4 = (2 * abs(rho))/(sigma * (1 - (abs(rho))^2) * sqrt(gamma));

  y =  k1*(abs(hh)).^2 + k2*(abs(hv)).^2 + k3*(abs(vv)).^2  ...
     - k4*(abs(hh).*abs(vv).*cos(angle(hh)-angle(vv)-angle(rho)));

%******************************************************************************
```

```
%**************************************************************************
%     FUNCTION display1.m
%**************************************************************************
%  Purpose: Displays the current pre-processed image data set on the screen.
%           Also will indicate known target coordinates in selected images.
%
%  Called by: sar.m
%  Calls: display.m, del1.m
%
%  Quirks:
%    1)  If data set uses more than three aspects, figures will overlap
%
%**************************************************************************

  function [] = display1(auto)

% clearing figures
  del1

% setup message window
  figure(1)
  cla
  hold on
  axis('xy','off')
  mess = text(.1,.6,'broke before anything ran');

load files

% choose between polarimatric and processed images

if (prep=='non' |  prep=='rot') & detect~=3     % polarimetric
  for i = 1:size(file_name,1)

    if prep == 'non'
      set(mess,'String',['Loading unrotated ' file_name(i,:)])
        pause(1)
      if i == 1
        load unrotated1
      elseif i == 2
        load unrotated2
      elseif i == 3
        load unrotated3
      elseif i == 4
        load unrotated4
      elseif i == 5
        load unrotated5
      elseif i == 6
        load unrotated6
      elseif i == 7
        load unrotated7
      end
```

```
elseif prep == 'rot'
  set(mess,'String',['Loading rotated ' file_name(i,:)])
    pause(1)
  if i == 1
    load rotated1
  elseif i == 2
    load rotated2
  elseif i == 3
    load rotated3
  elseif i == 4
    load rotated4
  elseif i == 5
    load rotated5
  elseif i == 6
    load rotated6
  elseif i == 7
    load rotated7
  end
end

if size(file_name,1) > 3
  rect = [(i-1)*105+5,465,200,200];
else
  rect = [(i-1)*210+5,465,200,200];
end
  set(mess,'String',['Displaying ' file_name(i,:) ' hh'])
  pause(1)
hand = figure('Position',rect,  ...
              'Name',[file_name(i,:) ' hh'],'NumberTitle','off');
display(hh,hand,auto)
% Indicate targets
  for j=1:length(targets)
    if targets(j,1) ~= 0
      text(targets(j,1),targets(j,2),'X')
    end
  end

if size(file_name,1) > 3
  rect = [(i-1)*105+5,465-230,200,200];
else
  rect = [(i-1)*210+5,465-230,200,200];
end
  set(mess,'String',['Displaying ' file_name(i,:) ' hv'])
  pause(1)
hand = figure('Position',rect,  ...
              'Name',[file_name(i,:) ' hv'],'NumberTitle','off');
display(hv,hand,auto)

if size(file_name,1) > 3
  rect = [(i-1)*105+5,465-460,200,200];
else
```

```
      rect = [(i-1)*210+5,465-460,200,200];
    end
      set(mess,'String',['Displaying ' file_name(i,:) ' vv'])
      pause(1)
    hand = figure('Position',rect,  ...
                   'Name',[file_name(i,:) ' vv'],'NumberTitle','off');
    display(vv,hand,auto)

  end

elseif prep=='ave' | prep=='spn' | prep=='ows' | prep=='pwf' | detect==3

  if detect ~= 3
    if prep == 'ave'
      set(mess,'String','Loading averaged')
        pause(1)
      load averaged
    elseif prep == 'spn'
      set(mess,'String','Loading spaned')
        pause(1)
      load spaned
    elseif prep == 'ows'
      set(mess,'String','Loading optimal')
        pause(1)
      load optimal
    elseif prep == 'pwf'
      set(mess,'String','Loading whitened')
        pause(1)
      load whitened
    end
  elseif detect == 3
    set(mess,'String','Loading Subtracted')
      pause(1)
    load det3
  end

  for i = 1:size(file_name,1)

    set(mess, ...
    'String',['Displaying ' file_name(i,:) ' ' prep ])
      pause(1)
    if size(file_name,1) > 3
      rect = [(i-1)*105+5,5,200,200];
    else
      rect = [(i-1)*210+5,5,200,200];
    end
    hand = figure('Position',rect,  ...
           'Name',[file_name(i,:) ' ' prep ],   ...
           'NumberTitle','off');
    if i == 1
      display(aspect1,hand,auto)
    elseif i == 2
      display(aspect2,hand,auto)
    elseif i == 3
      display(aspect3,hand,auto)
    elseif i == 4
      display(aspect4,hand,auto)
    elseif i == 5
      display(aspect5,hand,auto)
    elseif i == 6
      display(aspect6,hand,auto)
    elseif i == 7
      display(aspect7,hand,auto)
    end

    if i == 1 | detect >= 1
    % Indicate targets
      for j=1:length(targets)
        if targets(j,1) ~= 0
          text(targets(j,1),targets(j,2),'X')
        end
      end

    end
  end

end

  set(mess,'String','Output Complete')
    pause(5)
  figure(1)
  cla
  hold off

%****************************************************************************
```

```
%*****************************************************************************
%     ROUTINE display2.m
%*****************************************************************************
%  Purpose: Makes encapsulated postcript file of all displayed figures except
%           for figure 1, the menu figure.
%
%  Called by: sar.m
%
%  Quirks:
%    1) setup for maximum of 21 displayed images (22 figures).
%
%*****************************************************************************

% setup message window
  figure(1)
  cla
  hold on
  axis('xy','off')
  mess = text(.1,.6,' ');

% check to see that figures are displayed

h = figure('visible','off');
if h <= 2
  set(mess,'String','Figures are not displayed at this time')
    pause(5)
  del1

else
  delete(h)
  h = h - 1;
  for i=2:h
    figure(i)
      set(mess,'String',['Printing Figure' i])
       pause(1)
    if i == 2
       print fig2 -deps
    elseif i == 3
       print fig3 -deps
    elseif i == 4
       print fig4 -deps
    elseif i == 5
       print fig5 -deps
    elseif i == 6
       print fig6 -deps
    elseif i == 7
       print fig7 -deps
    elseif i == 8
       print fig8 -deps
    elseif i == 9
       print fig9 -deps
      elseif i == 10
         print fig10 -deps
      elseif i == 11
         print fig11 -deps
      elseif i == 12
         print fig12 -deps
      elseif i == 13
         print fig13 -deps
      elseif i == 14
         print fig14 -deps
      elseif i == 15
         print fig15 -deps
      elseif i == 16
         print fig16 -deps
      elseif i == 17
         print fig17 -deps
      elseif i == 18
         print fig18 -deps
      elseif i == 19
         print fig19 -deps
      elseif i == 20
         print fig20 -deps
      elseif i == 21
         print fig21 -deps
      end
      elseif i == 22
         print fig22 -deps
      end
   end

end
% Finish up
  set(mess,'String','Output Complete')
    pause(5)
  figure(1)
  cla
  hold off

clear

%*******************************************************************************
```

```
%*****************************************************************************
%     ROUTINE del1.m
%*****************************************************************************
%  Purpose:  Deletes the current set of figures except for figure 1 which is
%            the menu figure window.
%
%  Called by: sar.m, display1.m, del2.m
%
%****************************************************************************

h = figure('Visible','off');

for i = h:-1:2
  delete(i)
end

clear h i

figure(1)
  cla
  hold on
  axis('xy','off')

%*****************************************************************************
```

```
%*******************************************************************************
%     FUNCTION display.m
%*******************************************************************************
%  Purpose:  Log encodes the image, scales, and adjusts the intensity of the
%            image for display purposes.
%
%  Called by: display1.m
%
%  Quirks:
%    1) Intensity adjustments are hard-coded.
%
%  NOTE:
%    MatLab displays the images as transposes, x and y coordinates are
%    reversed.  In order to correct this so that target coordinates could
%    be plotted over the image and match, the image is transposed just before
%    displaying.
%
%******************************************************************************

function display(X,fig,autoscale)

% Converting to dB and scaling
  X = 20*log10(abs(X));

% Selecting maximum and minimum
  if autoscale <= 0  % images are clipped at -50 and 20 dB
    minimum = -50;
    maximum = 20;
  elseif autoscale == 1  % images are ranged between max and min pixels
    minimum = min(min(X));
    maximum = max(max(X));
  end

% scaling
  range = abs(maximum - minimum);
  if range ~= 0
    X = round(((X - minimum)/range)*255);
  else
    X = 0 * ones(size(X,1),size(X,2));
  end

% clipping out of range values
  if autoscale == 0
    [i,j] = find(X<0);
    for k = 1 : length(i)
      X(i(k),j(k)) = 0;
    end
    [i,j] = find(X>255);
    for k = 1 : length(i)
      X(i(k),j(k)) = 255;
    end
  end

% correcting for Matlabs imaging axis transposition
  X = X';

% Display image
  figure(fig)
  set(fig,'PaperPosition',[1,3,6,5.75],'NextPlot','new')
  hold on
  axis('xy','off')
  colormap(gray)
  if autoscale == 0
    image(X-75)
  elseif autoscale == -1
    image(X-150)
  elseif autoscale == 1
    image(X-115)
  end
  hold off

%******************************************************************************
```

```
%******************************************************************************
%     ROUTINE del2.m
%******************************************************************************
%  Purpose: Deletes all existing .mat  and .tex files in current directory
%           that would be created by a session of sar.m
%
%  Called by: sar.m
%  Calls: del1.m
%
%*****************************************************************************

if exist('unrotated1.mat') == 2
  delete('unrotated1.mat')
end
if exist('unrotated2.mat') == 2
  delete('unrotated2.mat')
end
if exist('unrotated3.mat') == 2
  delete('unrotated3.mat')
end
if exist('unrotated4.mat') == 2
  delete('unrotated4.mat')
end
if exist('unrotated5.mat') == 2
  delete('unrotated5.mat')
end
if exist('unrotated6.mat') == 2
  delete('unrotated6.mat')
end
if exist('unrotated7.mat') == 2
  delete('unrotated7.mat')
end

if exist('rotated1.mat') == 2
  delete('rotated1.mat')
end
if exist('rotated2.mat') == 2
  delete('rotated2.mat')
end
if exist('rotated3.mat') == 2
  delete('rotated3.mat')
end
if exist('rotated4.mat') == 2
  delete('rotated4.mat')
end
if exist('rotated5.mat') == 2
  delete('rotated5.mat')
end
if exist('rotated6.mat') == 2
  delete('rotated6.mat')
end
```

```
if exist('rotated7.mat') == 2
  delete('rotated7.mat')
end
if exist('rotatedd.mat') == 2
  delete('rotatedd.mat')
end
if exist('det1.mat') == 2
  delete('det1.mat')
end
if exist('det2.mat') == 2
  delete('det2.mat')
end
if exist('det3.mat') == 2
  delete('det3.mat')
end

if exist('averaged.mat') == 2
  delete('averaged.mat')
end
if exist('spaned.mat') == 2
  delete('spaned.mat')
end
if exist('optimal.mat') == 2
  delete('optimal.mat')
end
if exist('whitened.mat') == 2
  delete('whitened.mat')
end

if exist('statistics.mat') == 2
  delete('statistics.mat')
end
if exist('stat1.tex') == 2
  delete('stat1.tex')
end
if exist('stat2.tex') == 2
  delete('stat2.tex')
end
if exist('det1.eps') == 2
  delete('det1.eps')
end
if exist('det2.eps') == 2
  delete('det2.eps')
end
if exist('det3.eps') == 2
  delete('det3.eps')
end

if exist('files.mat') == 2
  delete('files.mat')
end
```

```
del1

clear

%**************************************************************************************
```

```
%*******************************************************************************
%     ROUTINE show_stat.m
%*******************************************************************************
%  Purpose: Displays the statistical characteristics of the current data set
%           If statistics.mat does not exist will create it.  Creates two .tex
%           files for input into a LaTeX document, stat1.tex contains the
%           standard deviation to mean ratios and stat2.tex contains the
%           target to clutter ratios.
%
%  Called by: sar.m
%  Calls: stat.m
%
%******************************************************************************

if exist('statistics.mat') ~=2
  stat
else
  load statistics.mat
  load files
end

% setup figure window
  figure(1)
  cla
  hold on
  axis('xy','off')

% main title
  text(.2,1.05,'Standard Deviation to Mean Ratio (dB)')

subtitles = ['unrot HH' ; 'unrot HV' ; 'unrot VV' ; ...
             'rot HH   ' ; 'rot HV   ' ; 'rot VV   ' ; ...
             'averaged' ; 'spaned   ' ; 'optimal  ' ; 'whitened'];

% sub titles and output of standard deviation to mean
  for i = 1 : size(file_name,1)
    text( (i*0.15)-.1,1,file_name(i,:) )
  end
  for i = 1:10
    text(-.15,1-(.05*i),subtitles(i,:))
  end
  for i = 1:size(file_name,1)
    for j = 1:10
      text( (i*0.15)-.1,1-(.05*j), num2str(sdm(j,i)) )
    end
  end

% title
  text(.2,.45,'Target to Clutter Ratio (dB)')

% sub titles and output of target to clutter
```

```
  for i = 1 : size(file_name,1)
    text( (i*0.15)-.1,.4,file_name(i,:) )
  end
  for i = 1:10
    text(-.15,.4-(.05*i),subtitles(i,:))
  end
  for i = 1:size(file_name,1)
    for j = 1:10
      text( (i*0.15)-.1,.4-(.05*j), num2str(tc(j,i)) )
    end
  end

% print to ascii file for reading by LaTex
  fid = fopen('stat1.tex','w');
  for i = 1:10
    fprintf(fid,'%s',subtitles(i,:));
    for j = 1:size(file_name,1)
      fprintf(fid,'\t & %4.2f',sdm(i,j));
    end
    fprintf(fid,'\t \\\\  \\hline \n');
  end
  fclose(fid);

  fid = fopen('stat2.tex','w');
  for i = 1:10
    fprintf(fid,'%s',subtitles(i,:));
    for j = 1:size(file_name,1)
      fprintf(fid,'\t & %4.2f',tc(i,j));
    end
    fprintf(fid,'\t \\\\  \\hline \n');
  end
  fclose(fid);

clear

%******************************************************************************
```

```
%*****************************************************************************
%     ROUTINE stat.m
%*****************************************************************************
%  Purpose:  Creates statistics.mat which contains the statistical data on the
%            current image set.
%
%  Called by: show_stat.m
%  Calls: targclut.m
%
%  Quirks:
%    1) will not give information on files that have not been created.
%
%*****************************************************************************

time = fix(clock);

% setup messages
  figure(1)
  cla
  hold on
  axis('xy','off')
  mess1 = text(.1,.6,' ');
  mess2 = text(.1,.5,' ');
  tmess = text(.1,1,' ');
  amess = text(.1,.95,' ');

set(tmess,'String',['Starting Time: ' num2str(time(1,4)) ':' ...
           num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(amess,'String','Max Run Time: 17 minutes'); pause(1)

% sdm = standard deviation to mean ratio (in db)
% tc = target to clutter ratio (in db)
%
%       matrices have the following form
%
%       unrotated HH plus       unrotated HH zero       unrotated HH minus
%       unrotated HV plus       unrotated HV zero       unrotated HV minus
%       unrotated VV plus       unrotated VV zero       unrotated VV minus
%       rotated HH plus         rotated HH zero         rotated HH minus
%       rotated HV plus         rotated HV zero         rotated HV minus
%       rotated VV plus         rotated VV zero         rotated VV minus
%       averaged plus           averaged zero           averaged minus
%       spaned plus             spaned zero             spaned minus
%       optimal plus            optimal zero            optimal minus
%       pwf plus                pwf zero                pwf minus

load files

% unrotated

if exist('unrotated1.mat') == 2
```

```
  for i = 1:size(file_name,1)
   set(mess1,'String',['Loading unrotated ' file_name(i,:)])
      pause(1)
    if i == 1
      load unrotated1
    elseif i == 2
      load unrotated2
    elseif i == 3
      load unrotated3
    elseif i == 4
      load unrotated4
    elseif i == 5
      load unrotated5
    elseif i == 6
      load unrotated6
    elseif i == 7
      load unrotated7
    end

    set(mess2,'String',['Standard deviation:mean ' file_name(i,:)], ...
              'Visible','on')
      pause(1)
    sdm(1,i) = 20*log10(abs( std(std(hh))/mean(mean(hh)) ));
    sdm(2,i) = 20*log10(abs( std(std(hv))/mean(mean(hv)) ));
    sdm(3,i) = 20*log10(abs( std(std(vv))/mean(mean(vv)) ));

    set(mess2,'String',['Target:clutter ' file_name(i,:)],'Visible','on')
      pause(1)
    tc(1,i) = targclut(hh,targets);
    tc(2,i) = targclut(hv,targets);
    tc(3,i) = targclut(vv,targets);
    set(mess2,'Visible','off')
      pause(1)

  end
end

% rotated

if exist('rotated1.mat') == 2
  for i = 1:size(file_name,1)
   set(mess1,'String',['Loading rotated ' file_name(i,:)])
      pause(1)
    if i == 1
      load rotated1
    elseif i == 2
      load rotated2
    elseif i == 3
      load rotated3
    elseif i == 4
      load rotated4
    elseif i == 5
      load rotated5
    elseif i == 6
      load rotated6
    elseif i == 7
      load rotated7
    end

    set(mess2,'String',['Standard deviation:mean ' file_name(i,:)], ...
              'Visible','on')
      pause(1)
    sdm(4,i) = 20*log10(abs( std(std(hh))/mean(mean(hh)) ));
    sdm(5,i) = 20*log10(abs( std(std(hv))/mean(mean(hv)) ));
    sdm(6,i) = 20*log10(abs( std(std(vv))/mean(mean(vv)) ));

    set(mess2,'String',['Target:clutter ' file_name(i,:)],'Visible','on')
      pause(1)
    tc(4,i) = targclut(hh,targets);
    tc(5,i) = targclut(hv,targets);
    tc(6,i) = targclut(vv,targets);
    set(mess2,'Visible','off')
      pause(1)

  end
end

clear hh hv vv

% averaged

if exist('averaged.mat') == 2
  set(mess1,'String','Loading averaged ')
    pause(1)
  load averaged
  for i = 1 : size(file_name,1)
    if i == 1
      temp = aspect1;
    elseif i == 2
      temp = aspect2;
    elseif i == 3
      temp = aspect3;
    elseif i == 4
      temp = aspect4;
    elseif i == 5
      temp = aspect5;
    elseif i == 6
      temp = aspect6;
    elseif i == 7
      temp = aspect7;
    end
    set(mess2,'String',['Standard deviation:mean ' file_name(i,:)], ...
              'Visible','on')
        pause(1)
      sdm(7,i) = 20*log10(abs( std(std(temp))/mean(mean(temp)) ));
      set(mess2,'String',['Target:clutter ' file_name(i,:)],'Visible','on')
        pause(1)
      tc(7,i) = targclut(temp,targets);
    end
    set(mess2,'Visible','off')
      pause(1)
  end

  % span

  if exist('spaned.mat') == 2
    set(mess1,'String','Loading spaned ')
      pause(1)
    load spaned
    for i = 1 : size(file_name,1)
      if i == 1
        temp = aspect1;
      elseif i == 2
        temp = aspect2;
      elseif i == 3
        temp = aspect3;
      elseif i == 4
        temp = aspect4;
      elseif i == 5
        temp = aspect5;
      elseif i == 6
        temp = aspect6;
      elseif i == 7
        temp = aspect7;
      end
      set(mess2,'String',['Standard deviation:mean ' file_name(i,:)], ...
              'Visible','on')
        pause(1)
      sdm(8,i) = 20*log10(abs( std(std(temp))/mean(mean(temp)) ));
      set(mess2,'String',['Target:clutter ' file_name(i,:)],'Visible','on')
        pause(1)
      tc(8,i) = targclut(temp,targets);
    end
    set(mess2,'Visible','off')
      pause(1)
  end

  % optimal

  if exist('optimal.mat') == 2
    set(mess1,'String','Loading optimal ')
      pause(1)
    load optimal
  for i = 1 : size(file_name,1)
    if i == 1
      temp = aspect1;
    elseif i == 2
      temp = aspect2;
    elseif i == 3
      temp = aspect3;
    elseif i == 4
      temp = aspect4;
    elseif i == 5
      temp = aspect5;
    elseif i == 6
      temp = aspect6;
    elseif i == 7
      temp = aspect7;
    end
    set(mess2,'String',['Standard deviation:mean ' file_name(i,:)], ...
              'Visible','on')
      pause(1)
    sdm(9,i) = 20*log10(abs( std(std(temp))/mean(mean(temp)) ));
    set(mess2,'String',['Target:clutter ' file_name(i,:)],'Visible','on')
      pause(1)
    tc(9,i) = targclut(temp,targets);
  end
  set(mess2,'Visible','off')
    pause(1)
end

% pwf

if exist('whitened.mat') == 2
  set(mess1,'String','Loading whitened ')
    pause(1)
  load whitened
  for i = 1 : size(file_name,1)
    if i == 1
      temp = aspect1;
    elseif i == 2
      temp = aspect2;
    elseif i == 3
      temp = aspect3;
    elseif i == 4
      temp = aspect4;
    elseif i == 5
      temp = aspect5;
    elseif i == 6
      temp = aspect6;
    elseif i == 7
      temp = aspect7;
    end
    set(mess2,'String',['Standard deviation:mean ' file_name(i,:)], ...
              'Visible','on')
      pause(1)
    sdm(10,i) = 20*log10(abs( std(std(temp))/mean(mean(temp)) ));
    set(mess2,'String',['Target:clutter ' file_name(i,:)],'Visible','on')
      pause(1)
    tc(10,i) = targclut(temp,targets);
  end
  set(mess2,'Visible','off')
    pause(1)
end

clear aspect1 aspect2 aspect3 aspect4 aspect5 aspect6 aspect7

% wrap up

set(mess1,'String','Saving as statistics.mat')
  pause(1)
save statistics sdm tc

set(mess1,'String','Statistical Routine Complete')
  pause(5)
  figure(1)
  cla
  hold off

%*******************************************************************************
```

```
%****************************************************************************
%     FUNCTION targclut.m
%****************************************************************************
%  Purpose:  calculates the target to clutter ratio of an image, given the
%            known coordinates of targets
%
%  Called by: stat.m
%
%  Quirks:
%    1) assumes a "large" image with small number of targets.
%
%****************************************************************************

function [y] = targclut(x,targc)
  % x - input image
  % targc - target coordinates

r = 5;  % 5 chosen to get good average over target, but not include clutter.

% Note that in the following algorithm, if target is in corner it will not
% be used
targr = [];
for i = 1:size(targc,1)
  temp = [];
  if targc(i,1) ~= 0 & targc(i,1)>=r+1 & targc(i,2)>=r+1 & ...
      targc(i,1)<=size(x,1)-r-1 & targc(i,2)<=size(x,2)-r-1
    temp = x( targc(i,1)-r:targc(i,1)+r , targc(i,2)-r:targc(i,2)+r );
  elseif targc(i,1) ~= 0 & targc(i,2)>=r+1 & ...
      targc(i,1)<=size(x,1)-r-1 & targc(i,2)<=size(x,2)-r-1
    temp = x( 1:targc(i,1)+r , targc(i,2)-r:targc(i,2)+r );
  elseif targc(i,1) ~= 0 & targc(i,1)>=r+1 & ...
      targc(i,1)<=size(x,1)-r-1 & targc(i,2)<=size(x,2)-r-1
    temp = x( targc(i,1)-r:targc(i,1)+r , 1:targc(i,2)+r );
  elseif targc(i,1) ~= 0 & targc(i,1)>=r+1 & targc(i,2)>=r+1 & ...
      targc(i,2)<=size(x,2)-r-1
    temp = x( targc(i,1)-r:size(x,1) , targc(i,2)-r:targc(i,2)+r );
  elseif targc(i,1) ~= 0 & targc(i,1)>=r+1 & targc(i,2)>=r+1 & ...
      targc(i,1)<=size(x,1)-r-1
    temp = x( targc(i,1)-r:targc(i,1)+r , targc(i,2)-r:size(x,2) );
  end
  targr = [targr ; reshape(temp, size(temp,1)*size(temp,2) ,1) ];
end

y = 20*log10(abs( mean(mean(targr))/mean(mean(x)) ));
         % mean for clutter was taken over entire image to simplify routine
         % contribution of target to entire image should be small if
         % number of targets is small and image is large

%****************************************************************************
```

```
%*****************************************************************************
%     ROUTINE del1.m
%*****************************************************************************
%  Purpose:  Deletes the current set of figures except for figure 1 which is
%            the menu figure window.
%
%  Called by: sar.m, display1.m, del2.m
%
%*****************************************************************************

h = figure('Visible','off');

for i = h:-1:2
  delete(i)
end

clear h i

figure(1)
  cla
  hold on
  axis('xy','off')

%*****************************************************************************
```

```
%*****************************************************************************
%     FUNCTION detection.m
%*****************************************************************************
%  Purpose:  Selects the form of the detection and displays coordinates of
%            known and detected targets.
%
%  Called by: sar.m
%  Calls: detect1.m, detect2.m
%
%  Quirks:
%    1) Not set up to handle unrotated images.
%    2) Cut-off value for determining threshold value is hard-coded.
%
%****************************************************************************

function [] = detection(type)

time = fix(clock);

% setup message
  figure(1)
  hold on
  axis('xy','off')
  cla
  mess = text(.1,.6,' ');
  tmess = text(.1,1,' ');
  amess = text(.1,.95,' ');

set(tmess,'String',['Starting Time: ' num2str(time(1,4)) ':' ...
           num2str(time(1,5)) ':' num2str(time(1,6)) ]);
set(amess,'String','Max Run Time: indeterminant'); pause(1)

load files

if type == 0
  detect = 0;
  set(mess,'String','No Detection Done')
    pause(5)
    figure(1)
    cla
    hold off
elseif type == 1
  detect = 1;
  type = -1;
elseif type == 2
  detect = 2;
  type = -1;
elseif type == 3
  detect = 3;
  type = -1;
end
```

```
save files directory file_name prep detect

if type < 0
  if prep == 'rot'
    set(mess,'String','Loading rotated')
      pause(1)
    if exist('rotatedd.mat') == 2
      load rotatedd.mat
    else
      if exist('rotated1.mat') == 2
        load rotated1
        aspect1 = hh;
      end
      if exist('rotated2.mat') == 2
        load rotated2
        aspect2 = hh;
      end
      if exist('rotated3.mat') == 2
        load rotated3
        aspect3 = hh;
      end
      if exist('rotated4.mat') == 2
        load rotated4
        aspect4 = hh;
      end
      if exist('rotated5.mat') == 2
        load rotated5
        aspect5 = hh;
      end
      if exist('rotated6.mat') == 2
        load rotated6
        aspect6 = hh;
      end
      if exist('rotated7.mat') == 2
        load rotated7
        aspect7 = hh;
      end
      save rotatedd aspect1 aspect2 aspect3 ...
           aspect4 aspect5 aspect6 aspect7 targets
    end
  elseif prep == 'ave'
    set(mess,'String','Loading averaged')
      pause(1)
    load averaged
  elseif prep == 'spn'
    set(mess,'String','Loading spaned')
      pause(1)
    load spaned
  elseif prep == 'ows'
    set(mess,'String','Loading optimal')
```

```
      pause(1)
    load optimal
  elseif prep == 'pwf'
    set(mess,'String','Loading whitened')
      pause(1)
    load whitened
  end

  if detect == 1
set(mess,'String',['Multiple Aspect Detection on ' prep ' (scaled threshold)'])
      pause(1)
    detect2
  elseif detect == 2
set(mess,'String',['Multiple Aspect Detection on ' prep ' (preset threshold)'])
      pause(1)
    detect2
  elseif detect == 3
    set(mess,'String',['Subtraction on ' prep])
      pause(1)
    detect2
  end

% Output information
  figure(1)
  cla
if prep == 'non'
  text(0,1.05,'Must select a pre-processed form!')
  pause(5)
  cla
end

end

%*******************************************************************************
```

```
%*****************************************************************************
%     FUNCTION detect1.m
%*****************************************************************************
%  Purpose:  Subtracts two aspects from each other.
%
%  Called by: detection.m
%  Calls: detect2.m
%
%****************************************************************************

function [y] = detect1(aspect1,aspect2)

% difference between aspect 1 and aspect2
    y = ( abs(aspect1) - abs(aspect2) );

% adding minimum value back in
  minimum = 0.001;
  y = y + min(min(y)) + minimum;

%*****************************************************************************
```

```
%****************************************************************************
%     ROUTINE detect2.m
%****************************************************************************
%  Purpose:  Uses threshold detection on multiple images.
%
%  Called by: detection.m
%
%  Calls: thresh.m
%
%  Quirks:
%    1) Radius for grouping pixels for a single target is hard-coded.
%    2) Cutoff values are hard-coded.
%
%***************************************************************************

upper = 1.0;  % upper limit of cutoff
  inc = .05;  % incrementing cutoff
lower = .05;  % lower limit of cutoff

% subtraction detection
  if detect == 3
    set(mess,'String',['Subtraction on ' prep])
      pause(1)
    if size(file_name,1) >= 1
      aspect1 = aspect1;
    end
    if size(file_name,1) >= 2
      aspect2 = detect1(aspect1,aspect2);
    end
    if size(file_name,1) >= 3
      aspect3 = detect1(aspect1,aspect3);
    end
    if size(file_name,1) >= 4
      aspect4 = detect1(aspect1,aspect4);
    end
    if size(file_name,1) >= 5
      aspect5 = detect1(aspect1,aspect5);
    end
    if size(file_name,1) >= 6
      aspect6 = detect1(aspect1,aspect6);
    end
    if size(file_name,1) >= 7
      aspect7 = detect1(aspect1,aspect7);
    end
    set(mess,'String',['Subtraction Detection on ' prep])
      pause(1)
  end

detected = [];
for cutoff = lower:inc:upper
  % set threshold using most positive aspect
  if detect == 2
    threshold = ( max(max(abs(aspect1))) + min(min(abs(aspect1))) )*cutoff;
  end
% Run detection on all aspects
  if size(file_name,1) >= 1
    if detect == 1 | detect == 3
      threshold = ( max(max(abs(aspect1))) + min(min(abs(aspect1))) )*cutoff;
    end
    targs1 = thresh(aspect1,threshold);
    if detect ~= 3
      targsc = targs1;
    end
  end
  if size(file_name,1) >= 2
    if detect == 1  | detect == 3
      threshold = ( max(max(abs(aspect2))) + min(min(abs(aspect2))) )*cutoff;
    end
    if detect ~= 3
      targsc = [targsc ; thresh(aspect2,threshold)];
    else
      targsc = thresh(aspect2,threshold);
    end
  end
  if size(file_name,1) >= 3
    if detect == 1  | detect == 3
      threshold = ( max(max(abs(aspect3))) + min(min(abs(aspect3))) )*cutoff;
    end
    targsc = [targsc ; thresh(aspect3,threshold)];
  end
  if size(file_name,1) >= 4
    if detect == 1  | detect == 3
      threshold = ( max(max(abs(aspect4))) + min(min(abs(aspect4))) )*cutoff;
    end
    targsc = [targsc ; thresh(aspect4,threshold)];
  end
  if size(file_name,1) >= 5
    if detect == 1  | detect == 3
      threshold = ( max(max(abs(aspect5))) + min(min(abs(aspect5))) )*cutoff;
    end
    targsc = [targsc ; thresh(aspect5,threshold)];
  end
  if size(file_name,1) >= 6
    if detect == 1  | detect == 3
      threshold = ( max(max(abs(aspect6))) + min(min(abs(aspect6))) )*cutoff;
    end
    targsc = [targsc ; thresh(aspect6,threshold)];
  end
  if size(file_name,1) >= 7
    if detect == 1  | detect == 3
      threshold = ( max(max(abs(aspect7))) + min(min(abs(aspect7))) )*cutoff;
    end
```

```
        targsc = [targsc ; thresh(aspect7,threshold)];
      end
    % Form composite image with targc and then detect
      temp = 0 * ones( size(aspect1,1) , size(aspect1,2) );
      for j = 1:size(targsc,1)
        temp( targsc(j,1) , targsc(j,2) ) = 255;
      end
      targsc = thresh(temp,100);
      clear j temp
    % Determine the number of false alarms and targets detected store in matrix
      fa1 = 0;  fac = 0;  d1 = 0;  dc = 0;
      r = 15;  % radius around target coordinates
      for i = 1:size(targs1,1)
        for j = 1:size(targets,1)
          if targs1(i,:)<=targets(j,:)+r & targs1(i,:)>=targets(j,:)-r
            d1 = d1 + 1;
          end
        end
      end
      for i = 1:size(targsc,1)
        for j = 1:size(targets,1)
          if targsc(i,:)<=targets(j,:)+r & targsc(i,:)>=targets(j,:)-r
            dc = dc + 1;
          end
        end
      end
      fa1 = size(targs1,1) - d1;
      fac = size(targsc,1) - dc;
      detected = [ detected ; cutoff size(targets,1) fa1 d1 fac dc];
  end

  % print to ascii file for possible future use
    fid = fopen('det.tex','w');
      for i = 1:size(detected,1)
        for j = 1:size(detected,2)
          fprintf(fid,'%4.2f \t',detected(i,j));
        end
          fprintf(fid,'\t \n');
      end
    fclose(fid);

  if detect == 3
    save det3 detected targets ...
         aspect1 aspect2 aspect3 aspect4 aspect5 aspect6 aspect7
  end

  % plot
    del1
    if detect == 1
      figure('Position',[5,5,515,400], ...
           'Name',['Scaled threshold detection on ' prep],'NumberTitle','off');
```

```
elseif detect == 2
  figure('Position',[5,5,515,400], ...
       'Name',['Preset threshold detection on ' prep],'NumberTitle','off');
elseif detect == 3
  figure('Position',[5,5,515,400], ...
       'Name',['Subtraction detection on ' prep],'NumberTitle','off');
end
plot(detected(:,1),detected(:,3),'w-.', ...
     detected(:,1),detected(:,4),'w--', ...
     detected(:,1),detected(:,5),'w:', ...
     detected(:,1),detected(:,6),'w-', ...
     min(min(detected(:,1))),(max(max(detected))+1),'k.', ...
     detected(:,1),detected(:,2)+0.1,'r-', ...
     detected(:,1),detected(:,2)-0.1,'r-')
% use next six lines to set axis for comparison of plots
  xmin = 0;
  xmax = 1.0;
  ymin = 0;
  ymax = 20;
  axis([xmin,xmax,ymin,ymax])
  set_axis = 1;
xlabel('Threshold')
ylabel('Number of Detections or False Alarms')

% move these statements to just before the clears
% in order to print the legend with the  figure
  if detect == 1
    print det1 -deps
  elseif detect == 2
    print det2 -deps
  elseif detect == 3
    print det3 -deps
  end


if set_axis == 0
  xmin = min(detected(:,1));
  xmax = max(detected(:,1));
  ymin = min(min(detected(:,2:size(detected,2))));
  ymax = max(max(detected(:,2:size(detected,2))));
end
x = (xmax + xmin) * .5;
dx = (xmax - xmin) * .1;
y = (ymax + ymin) * .75;
dy = (ymax - ymin) * .05;
legend = [ '_._._     SAR false alarms   ' ; ...
           '---       SAR detected       ' ; ...
           '......... WASAR false alarms' ; ...
           '___       WASAR detected     ' ];

text(x,y,[num2str(size(targets,1)) ' known targets'])
```

```
  for i = 1:size(legend,1)
    if i == 1
      text(x,y+(dy*i)+(dy*.5),legend(i,1:10))
    elseif i == 2
      text(x,y+(dy*i),legend(i,1:10))
    else
      text(x,y+(dy*i)+(dy*.25),legend(i,1:10))
    end
    text(x+dx,y+(dy*i),legend(i,11:size(legend,2)))
  end

  % Move print statements to here to print legend with figure

clear upper lower cutoff threshold targs1 targsc fa1 d1 fac dc
clear xmax xmin ymax ymin x dx y dy legend ans i

%****************************************************************************
```

```
%******************************************************************************
%     FUNCTION thresh.m
%******************************************************************************
%  Purpose:  Threshold detection.
%
%  Called by: detect2.m
%
%  Quirks:
%    1) Radius for grouping pixels for a single target is hard-coded.
%
%******************************************************************************

function [targs] = thresh(aspect1, threshold)

% detection
  [x,y] = find(abs(aspect1) >= threshold);
  targs = [x y];

% group returns within specified radius and average
  r = 15;    % radius of pixels
  newtargs = [];
  for i = 1:size(targs,1)
    temp = [];
    for j = 1:size(targs,1)
      if targs(j,1)<=(targs(i,1)+r) & targs(j,1)>=(targs(i,1)-r) & ...
         targs(j,2)<=(targs(i,2)+r) & targs(j,2)>=(targs(i,2)-r)
        temp = [ temp ; targs(j,:) ];
      end
    end
    temp = [round(mean(temp(:,1))) round(mean(temp(:,2)))];
    newtargs = [ newtargs ; temp / length(j)];
  end
  targs = newtargs;

% Get rid of duplicate coordinates
  for i = 1:size(targs,1)
    for j = i:size(targs,1)
      if i ~= j
        if targs(j,1)<=(targs(i,1)+r) & targs(j,1)>=(targs(i,1)-r) & ...
           targs(j,2)<=(targs(i,2)+r) & targs(j,2)>=(targs(i,2)-r)
          targs(i,:) = [0 0];
        end
      end
    end
  end
  for i = size(targs,1):-1:1
    if targs(i,1)==0
      if i ~= 1
        targs = [ targs(1:i-1,:) ; targs(i+1:size(targs,1),:) ];
      else
        targs = targs(i+1:size(targs,1),:);
```

```
            end
          end
        end

      %*******************************************************************************
```

Bibliography

1. Bello, Martin G. "A Random-Field Model-Bases Algorithm for Anomalous Complex Image Pixel Detection," *IEEE Transactions on Image Processing, 1*(2):186–196 (apr 1992).
2. Dilsavor, Ronald L. *Optimal Detection of Targets in Clutter using Ultra-Wideband, Fully-Polarimetric SAR*. Technical Report, Ohio State University, oct 1992. Final Report for: AFOSR Summer Research Program, Wright-Patterson AFB.
3. Etter, D. M. *Structured Fortran 77*. Menlo Park, CA: The Benjamin/Cummings Publishing Co, Inc., 1983.
4. Fong, Zhen-Ming and Yu hai Mao. "Some Results on Radar Clutter Spectra Measurements." *Proceedings of the 1984 International Symposium on Noise and Clutter Rejection in Radars and Imaging Sensors*. 35–40. oct 1984.
5. Fukunaga, Keinosuke. *Introduction to Statistical Pattern Recognition* (2nd Edition). Boston, MA: Academic Press, Inc., 1990.
6. Hevenor, Richard A. and Pi-Fuay Chen. *Automated Extraction of Airport Runway Patterns From Radar Imagery*. Technical Report DTIC AD-A231 809, U.S. Army Corps of Engineers, Engineer Topographic Laboratories, Fort Belvoir, Virginia, jun 1990.
7. Holm, William A. "Applications of Polarimetry to Target/Clutter Discrimination in Millimeter Wave Radar Systems," *SPEI Polarimetry: Radar, Infrared, Visible, and X-Ray, 1317*:148–153 (1990).
8. Jr., William B. Googins. *Identification of Radar Targets By Pattern Recognition*. PhD dissertation, School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH, jun 1973.
9. Kelly, E. J. "An Adaptive Detection Algorithm," *IEEE Transactions on Aerospace and Electronic Systems, AES-22*(1):115–127 (mar 1986).
10. Kreyzig, Erwin. *Advanced Engineering Mathematics* (5 Edition). New York: John Wiley and Sons, 1983.
11. Long, Maurice W. "Polarization and Statistical Properties of Clutter." *Proceedings of the 1984 International Symposium on Noise and Clutter Rejection in Radars and Imaging Sensors*. 25–32. oct 1984.
12. Marshall, S. V. and G. G. Skitek. *Electromagnetic Concepts and Applications* (2 Edition). Englewood Cliffs, NJ: Prentice-Hall, Inc., 1987.
13. Mensa, Dean L. *High Resolution Cross-Section Imaging*. Norwood, MA: Artech House, Inc., 1991.
14. Miller, Irwin and John E. Freund. *Probability and Statistics for Engineers* (3 Edition). Englewood Cliffs, NJ: Prentice Hall, Inc., 1985.
15. Miller, Robert. "Characterization of Noncoherent Ground Clutter." *Proceedings of the 1984 International Symposium on Noise and Clutter Rejection in Radars and Imaging Sensors*. 59–64. oct 1984.
16. Mott, Harold. *Polarization in Antennas and Radar*. New York: John Wiley & Sons, 1986.
17. Nathanson, Fred E. *Radar Design Principles* (2 Edition). New York: McGraw-Hill, Inc., 1991.
18. Noether, Gottfried E. *Elements of Nonparametric Statistics*. New York: John Wiley & Sons, Inc., 1967.

19. Novak, L. M., et al. "Optimal Polarimetric Processing for Enhanced Target Detection," *IEE Transactions on Aerospace and Electronic Systems*, *29*(1):234–243 (jan 1993).

20. Novak, Leslie M. and Michael C. Burl. "Optimal Speckle Reduction in Polarimetric SAR Imagery," *IEEE Transactions on Aerospace and Electronic Systems*, *26*(2):293–305 (mar 1990).

21. Novak, Leslie M., et al. "Studies of Target Detection Algorithms That Use Polarimetric Radar Data," *IEEE Transactions on Aerospace and Electronic Systems*, *AES-25*(2):150–165 (mar 1989).

22. Ogura, Hisanao, et al. "Signal Processing for the Detection of Radar Signals Hidden in Clutter Noise." *Proceedings of the 1984 International Symposium on Noise and Clutter Rejection in Radars and Imaging Sensors*. 309–314. oct 1984.

23. Oliver, Christopher J. and Richard G. White. "Real-Time SAR Change-Detection Using Neural Networks," *SPEI, Advanced Signal-Processing Algorithms, Architectures, and Implementations*, *1348*:40–50 (1990).

24. Pao, Yoh-Han. *Adaptive Pattern Recognition and Neural Networks*. Reading, MA: Addison-Wesley Publishing Co., Inc., 1989.

25. Park, Hyung-Rae, et al. "Polarization-Space-Time Domain Generalized Likelihood Ratio Detection of Radar Targets." This report is yet to be published. A preliminary copy was obtained from Dr. Li, June 1993., 1993.

26. Reed, I. S., et al. "Rapid Convergence Rate in Adaptive Arrays," *IEE Transactions on Aerospace and Electronic Systems*, *AES-10*(6):853–863 (nov 1974).

27. Reed, Irving S. and Xiaoli Yu. "Adaptive Multiple-Band CFAR Detection of an Optical Pattern with Unknown Spectral Distribution," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, *38*(10):1760–1770 (oct 1990).

28. Rye, A. J., et al. "Image Analysis Workstations," *GEC Journal of Research*, *9*(1):36–45 (1991).

29. Scharf, Louis L. *Statistical Signal Processing: Detection, Estimation, and Time Series Analysis*. Reading, Mass: Addison-Wesley Publishing Co., 1991.

30. Sekine, Matsuo and Toshimitsu Musha. "Suppression of Clutter and Detection Of Targets." *Proceedings of the 1984 International Symposium on Noise and Clutter Rejection in Radars and Imaging Sensors*. 279–284. oct 1984.

31. Shanmugan, K. Sam and Arthur M. Breipohl. *Random Signals: Detection, Estimation, and Data Analysis*. New York: John Wiley and Sons 1988.

32. Shin, R, et al. "Theoretical Models for Polarimetric Radar Clutter." *In Proceedings of the 10th DARPA MMW Symposium*. 441–450. apr 1986.

33. Stremler, Ferrel G. *Introduction to Communications Systems* (3 Edition). Addison-Wesley, 1990.

34. Taub, Herbert and Donald L. Schilling. *Principles of Communication Systems* (2 Edition). New York: McGraw-Hill Book Co., 1986.

35. Widrow, Bernard and Samuel D. Stearns. *Adaptive Signal Processing*. Englewood Cliffs, N.J.: Prentice-Hall, Inc., 1985.

36. Ziemer, R. E. and W. H. Tranter. *Principles of Communications: Systems, Modulation, and Noise* (2 Edition). Boston: Houghton Mifflin Co., 1985.

Vita

Captain Kurt W. Knurr was born on 7 December, 1962 in Bethesda, Maryland. He graduated from Omro High School in Omro, Wisconsin in 1981 and enlisted in the U. S. Air Force in 1982 as an Electronic Communications and Cryptographic Equipment Systems Repairman serving at March AFB, California. He was accepted in to the Airman's Education and Commissioning Program in 1985 and attended the University of Missouri at Rolla, graduating with a Bachelor of Science in Electrical Engineering in December 1988. Upon graduation, he received a reserve commission in the USAF from Officer Training School in May 1989 and served his first tour of duty as an officer at Dyess AFB, Texas. He began as a B-1B Electrical Engineer working for the Deputy Commander for Maintenance in the 96th Bombardment (Heavy) Wing where he provided engineering support to the entire maintenance complex and particularly to weapon system reliability (B-1B and KC-135 aircraft), maintainability, and effectiveness until June 1991. He was then chosen for the position of Officer In Charge of the B-1B Central Integrated Test System (CITS) Office. There he was responsible for maintaining and maturing a data bank of over 10,000 B-1B CITS Maintenance Codes until entering the Graduate School of Engineering, Air Force Institute of Technology, in April 1992.

Permanent address: 734 Tyler Avenue
Omro, Wisconsin 54963

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE December 1993	3. REPORT TYPE AND DATES COVERED Master's Thesis

4. TITLE AND SUBTITLE
PROCESSING OF WIDE-ANGLE SYNTHETIC APERTURE RADAR SIGNALS FOR TARGET DETECTION

5. FUNDING NUMBERS

6. AUTHOR(S)
Kurt W. Knurr

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GE/ENG/93D-22

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Major Robert Williams
WL/AARA-2
Bldg 23
2010 5th St
WPAFB, OH 45433-7001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This study investigated methods of target detection using Wide-Angle Synthetic Aperture Radar (WASAR). WASAR uses multiple aspect angle Synthetic Aperture Radar (SAR) images of the same scene. The SAR images were generated using a pre-release software package from Loral Corporation. The software was able to generate 512 by 512 pixel SAR images that contained various vegetation returns which for our purposes we classified as clutter. Within this clutter, targets (M35 trucks) could be placed at random locations and orientations. The software also had the capability of generating fully-polarimetric WASAR images with multiple depression angles. This data was then processed and various detection algorithms tested to exploit the amount and diversity of information available from the multiple images. SAR images are generally known to contain large amounts of data and WASAR images contain even more due to the multiple images. Various pre-processing filters were analyzed for detection optimization. These filters included: polarimetric averaging, polarimetric span, polarimetric optimal weighting, and polarimetric whitening filter. Simple classical detection (thresholding) algorithms were evaluated using these pre-processed data sets. The use of WASAR imagery improved detection by allowing thresholds to be set higher than for simple SAR thereby avoiding false alarms yet still allowing detection of the known targets.

14. SUBJECT TERMS
SAR, Synthetic Aperture Radar, Wide-Angle SAR, Wide-Angle Synthetic Aperture Radar, Target Detection, Target Recognition, Target Identification

15. NUMBER OF PAGES
117

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102