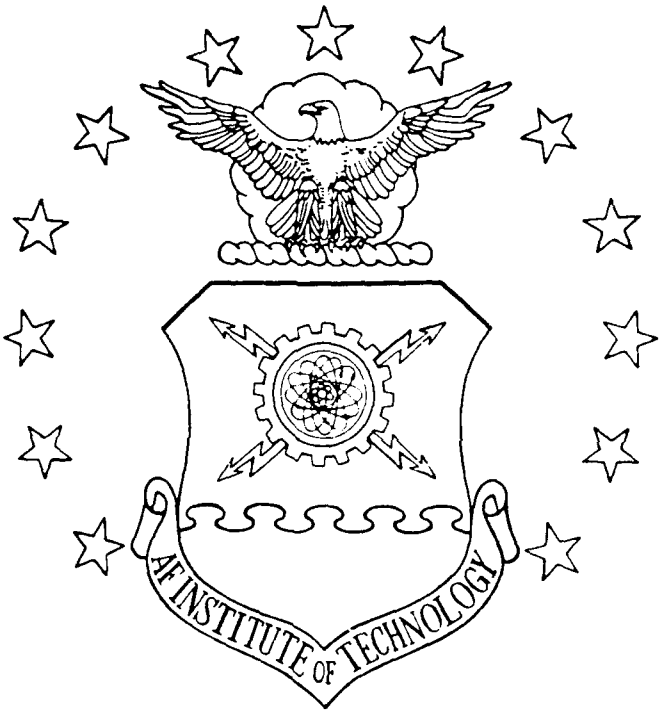


TARGET CLASSIFICATION USING SAR RADAR

THESIS

Alan Paul Callaghan
Squadron Leader, RAAF

AFIT/GE/ENG/89D-58

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/89D-58

TARGET CLASSIFICATION USING SYNTHETIC APERTURE RADAR POLARIMETRIC DATA

THESIS

Presented to the Faculty of the 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 in Electrical Engineering

Alan Paul Callaghan, BAPPSC
Squadron Leader, RAAF

June,1990

Acknowledgments

This work was sponsored by Mr Ed Zelnio, Chief, Automatic Target Recognition Technology Group, Aeronautical Systems Development (ASD) Wright Patterson Air Force Base. I am indebted to Ed Zelnio for sponsoring this work and for the considerable support that he offered during its development. I am also greatly indebted to two highly professional members of Ed Zelnio's staff, namely Mike Bryant and Kevin Willey, for their unmeasured enthusiasm and help in bringing this thesis to fruition. Their help, particularly, in the software aspects of this thesis, was invaluable.

My thanks is also extended to Dr. Kabrisky and Capt John DeBerry, who, while not on my thesis committee, provided me with important help and understanding in the Artificial Intelligence aspects of this thesis. I would also like to thank the members of my thesis committee (Dr. Andrew Terzuoli, LtCol David Meer and Dr. John Jones Jr.) for their continued support and valuable input throughout the course of this thesis.

Last, but not least, I would like to thank my wife Dorothy and my children Tara, Megan and Michael for the support and sacrifices they offered in order to make this thesis possible.

Alan Paul Callaghan

Table of Contents

List of Figures

List of Tables

Abstract

This study investigates the polarimetric behaviour of canonicals within single resolution cells. The aim of this investigation is to ascertain whether the contents of a resolution cell can be determined by examination of the polarimetric data. The canonicals addressed in this thesis effort are the dihedral and the flatplate, although the associated theory and software is readily applicable to other primitives.

The main software package used during this thesis is SarTool™, although additional ancillary software is presented such that the output of SarTool™ can be represented on the Poincare sphere.

Finally, both single level and multi-level perceptron networks are used to process the resolution cell output and return a decision on the contents of the cell. This preliminary study of the polarimetric and perceptron combination provides some valuable insight into the possible potential of polarimetrics and Artificial Intelligence in the area of target identification.

TARGET CLASSIFICATION USING SYNTHETIC APERTURE RADAR POLARIMETRIC DATA

I. Introduction

1.1 Background

The origins of radar theory can be traced to the early part of the century; however, practical radar systems only gained prominence during World War II where they were used extensively by the British Chief of Naval Operations as Early Warning (E.W.) devices. Even in those early days operators recognized that the theory of radar was not an exact science [1]. Variations in target detection ranges and in false alarm rates led to a flurry of postwar studies which ultimately resulted in the first generation of radar systems. These radars are commonly referred to as *conventional radars.*

1.1.1 Conventional Radars. Chapter 2 of this document refers the interested reader to texts which review the theory of conventional radars. From a systems point of view the limitation of conventional radars is their inability to identify certain tactical targets. This lack has limited the use of conventional radars to that of target detection systems. Simple target detection has proven less than satisfactory

to both military and civilian users of radars. From a military perspective modern weapons have resulted in increased Missile Engagement Zones (MEZ) and the subsequent development of "stand off and attack" doctrines which necessitate beyond visual range identification of potential targets. Similarly, the increasing use of space by civilian agencies, such as Land Satellite (LANDSAT) has necessitated the long range identification of certain Geophysical features, for example, coastlines, weather patterns, and crops.

1.1.2 SAR Systems. Since the mid 1960s the limitations of conventional radars and the advent of new technologies has prompted considerable interest in the development of SAR systems. Several good texts which review the theory of SAR systems are referenced in the literature review section of this paper. Basically SAR systems achieve better resolution performance than conventional radars due to the former's ability to use the platform's path as if it were a physical aperture. The use of the platform's flight path as a synthetic aperture, coupled with advanced processing algorithms and sizeable computer resources, results in resolutions significantly better than those attainable by conventional radars. Consequently SAR systems have become the "second generation of radars" and have been widely accepted for use in both military and civilian applications.

1.1.3 Polarimetric Research. Despite their improved performance over conventional radars, modern SAR systems are not able to perform certain tactical tasks,

for example, the identification of tanks and aircraft. Current research indicates that the only way SAR systems will be able to achieve the performance necessary for these tasks is by exploiting the polarization information contained in the radar return.

During the early days of SAR development, scientists noted that the returns from targets were invariably de-polarized (i.e. they returned to the receiver at a different polarization to that which they were transmitted on). Further investigation revealed that this process of de-polarization was not a completly random mechanism.

This revelation raised questions as to whether or not greater target identification (and hence classification) could be attained if the polarization of radar returns was included in the process of target analysis. Consequently, the study of Polarimetrics began. According to radar historians, this research started in the early 1950's and continued into the 1960's when it lapsed due to the following reasons:

> 1)Polarization-target dependence was an incompletely understood phenomenon; 2) Because polarization diversity techniques require antenna polarization control or agility on transmit and/or dual polarization channels on reception, the radar system complexity increased significantly and 3) the knowledge of polarization behaviour of complex target requires extensive and expensive dual polarization measurements [2:42].

In 1970, Huynen [3] revived interest in polarimetric studies. This revived interest, coupled with important theoretical and technical advances in polarimetric analysis, continues today. The ultimate objective of this research is to significantly improve the classification capabilities of SAR systems.

1.1.4 Problem Statement. The objective of this study is to analyze the polarimetric behaviour of certain canonical shapes within a single resolution cell of a SAR.

1.1.5 Scope. The study of polarimetrics is extensive and encompasses several mathematical and scientific disciplines pertinent to target definition. However, due to the time constraints imposed by the thesis program, the scope of this thesis effort will be limited to:

1. Establishing the accuracy of the SarTool^{tm} software in predicting the polarimetric behaviour of the canonical shapes used within this thesis.
2. Investigating ways in which SarTool^{tm} data can be effectively displayed using either the Poincare sphere or the Matrix Laboratory Software (MATLAB^{tm}) or a combination of the two.
3. Examining the polarimetric behaviour of individual canonical shapes within a single SAR resolution cell.
4. Examining the behaviour of two or more canonical shapes co-located within a single SAR resolution cell.
5. Investigating mean for discriminating between various shapes and combinations of shapes within a resolution cell.

1.2 Summary

Since the development of the first radar system in World War II there has been much ongoing research directed at improving the resolution performance of operational radar systems. This research is of paramount importance as the resolution performance of a radar to a large extent determines the overall capability of that radar.

Current SAR systems have resolution capabilities far in excess of conventional radars. Nevertheless, even with this enhanced resolution, today's SAR systems are incapable of performing several critical functions, for example, the identification of tactical targets. Furthermore, there are physical and system constraints which preclude these radars from attaining greater target classification performance unless polarimetrics are incorporated into the target discrimination process.

Unfortunately, the polarimetric behaviour of targets is an incompletely understood phenomenon. However, there is sufficient understanding of polarimetrics to indicate that this avenue of research may ultimately lead to target classification performances far in excess of those currently attainable. Such an improvement in radar performance would significantly increase the usefulness of SAR systems to civilian and military users alike.

This thesis effort is directed at increasing the knowledge of polarimetrics by investigating the polarimetric behaviour of certain canonical shapes within a single SAR resolution cell and by examining ways in which the resulting polarimetric

information can be used in the decision process.

II. Literature Search

2.1 Introduction

As mentioned in Chapter 1, the study of polarimetrics encompasses a large number of research activities; accordingly, before progressing further, a literature review is necessary to identify the material relevant to this thesis effort.

2.2 Scope

The literature reviewed here provides an overview of the development and potential of Polarimetric studies as they pertain to the current areas of Radar Imaging.

2.3 Organization

To fully appreciate the relevance and theory of Polarimetrics, one must first understand how the study of Polarimetrics applies to Radar imaging. Moreover, as the study of Polarimetrics is primarily theoretical (with only a few practical examples), the preponderance of papers on the subject deal mainly with the mathematical modelling of expected responses. Consequently, there is a need to review the mathematical tools and modelling processes currently being proposed by the leading authorities in this area of study. Accordingly, this Literature Review will be organized in the following manner:

1. Background. This section of the review will consist of the following subsections:

(a) Conventional Radars.

(b) SAR Systems.

2. Mathematical Tools. This section will briefly introduce the literature regarding the currently accepted mathematical tools of Polarimetrics, in particular the following topics will be reviewed:

(a) The Polarization Ellipse.

(b) The Poincare Sphere.

(c) The Scattering Matrix.

3. Results of Previous Polarimetric Studies. This section will provide a brief overview of several technical papers which compare actual Radar Images with those obtained via modelling with the Mathematical tools mentioned above.

2.4 Background

2.4.1 Conventional Radars. The theory of conventional radars is well known and is the subject of several textbooks. A particularly excellent treatment of the theory of conventional radars is found in [4].

As mentioned in Chapter 1, the salient limitation of conventional radars is their lack of both down range and cross range resolution. For example, typical radar parameters for an aircraft search radar are: 1) RF 3 GHZ, 2)Pulse Width .5 microseconds, 3) Aperture size 5 meters and 4) "look" angle of 3 degrees. Application

of conventional radar theory via equations (2.1) and (2.2) show that the best possible down range and cross range resolutions (on a target at 100km) are 75 meters and 640 meters respectively [5]:

$$\Delta R_r = \frac{\tau c}{2cos\alpha} \tag{2.1}$$

where,

R_r = down range resolution
τ = pulse width
α = look angle

$$\Delta R_c = R\frac{\lambda}{L}.64 \tag{2.2}$$

where,

R_c = cross range resolution
λ = radar wavelength
L = length of flight path

2.4.2 SAR Systems. The above resolutions are capable of providing simple functions, for example, target detection or coastline mapping. However, more advanced capabilities, such as target and crop identification, require considerably better resolutions than those attainable via conventional radars. Accordingly SAR systems

were developed which which could yield resolutions less than 1 meter [6]. Chapter 1 points out that these much improved resolutions are possible due to significant technological advances and improvements in processing techniques. A definitive treatment of these techniques is beyond the scope of this thesis; however, a brief review of SAR theory is included in the following chapter. A theoretical aspect of immediate relevance is that modern SAR systems exploit all available radar parameters except one. The exploited parameters are: time, frequency, phase, bearing and amplitude. The one parameter not used by current SAR systems is the polarization of the Electro-Magnetic (EM) wave.

The need for even better resolution performance than is achievable with conventional SAR systems has led to the study of polarimetrics which, as mentioned earlier, involves understanding the polarization behaviour of E.M. radiation. An integral part of this essentially theoretical study is the use of several mathematical tools.

2.5 Mathematical Tools.

Much of the research into polarimetrics is theoretical and based on the development and use of mathematical tools. Accordingly, this section will review the literature concerning these tools.

Many authors such as [7] and [8], have spent considerable research effort in using and improving the math and processing techniques associated with polarimet-

ric analy... The consensus among these authors is that the following three basic mathematical tools are fundamental to the study of polarimetrics:

1. The Polarization Ellipse.
2. The Poincare Sphere
3. The Scattering Matrix

2.5.1 Polarization Ellipse. All polarization states can be described using the Polarization Ellipse [9], an example of which appears in Figure 2.1. A brief description of the significance of Figure 2.1 is given below:

1. Ellipticity. The ellipticity angle χ reveals the polarization angle of the wave. For example, an ellipticity angle of 0° would denote linear polarization whereas an angle of 45° would indicate circular polarization. Moreover, the sign of the ellipticity angle denotes the "handedness" of the wave i.e. whether the wave is travelling towards an observer (right handed) or away from an observer (left handed).
2. Ellipse Orientation Angle. The Ellipse Orientation Angle Ψ represents the "tilt" or orientation of the ellipse relative to a defined axis. The tilt or orientation of an EM wave is an important consideration in defining a wave in a specified coordinate system.

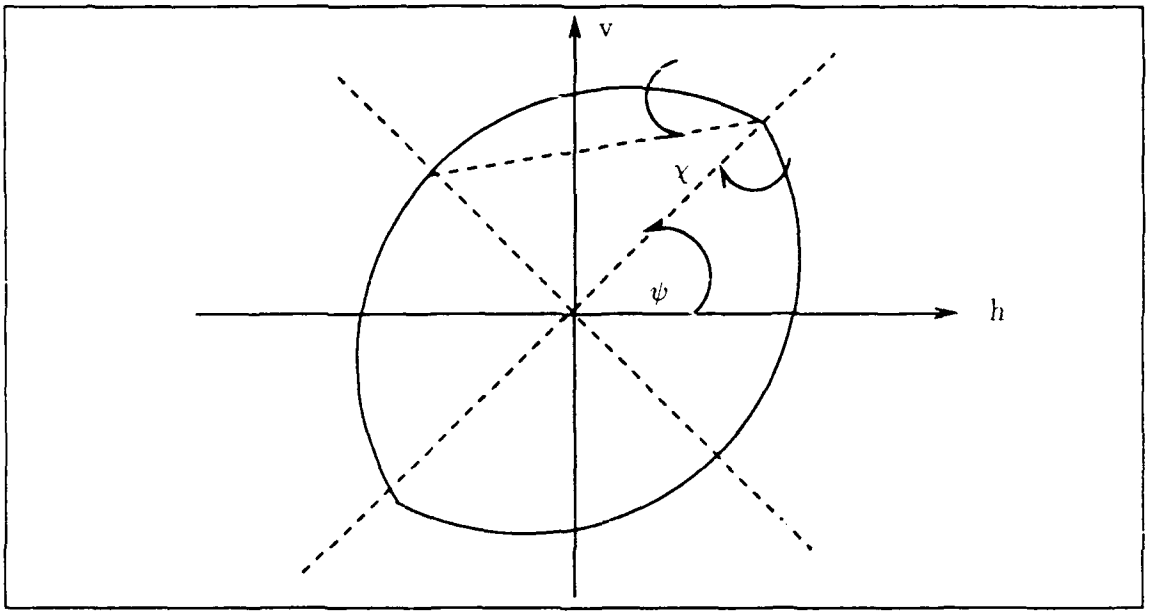


Figure 2.1. The Polarization Ellipse [2]

2.5.2 Poincare Sphere. The Poincare Sphere is described in considerable detail in most papers pertaining to polarimetric studies. Kennaugh is generally accepted as being the leading authority on the Poincare sphere, and during last 30 years he has written some excellent papers on its applications to polarimetrics [10]. The mapping of the Polarization ellipse onto the Poincare sphere is accomplished as shown in Figure 2.2. Basically, each ellipticity angle χ and orientation angle Ψ can be uniquely mapped to a point on the sphere having longitude 2Ψ and latitude 2χ. Examination of the sphere reveals that the poles represent the circular polarizations, the equator represents the linear polarizations and the hemispheres represent the "handedness" of the wave. The power in the EM wave is proportional to the radius of the sphere. One of the great strengths of the Poincare sphere is that

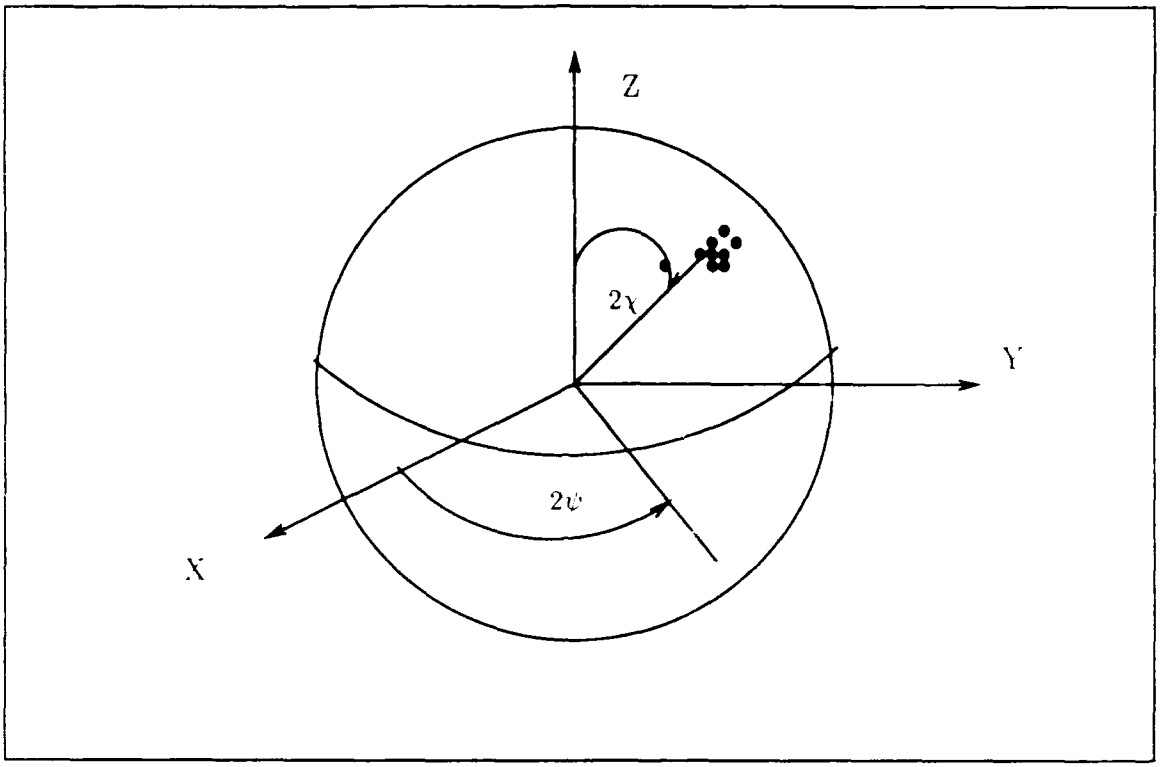


Figure 2.2. The Poincare Sphere [8]

it can be used to show the statistical or stochastic variations of polarizations. For example the shaded area in Figure 2.2 indicates the degree of polarization jitter in the measured wave's ellipticity and orientation angles.

2.5.3 Scattering Matrix. There is a considerable amount of scientific literature devoted to the scattering matrix because mathematical techniques, coupled with new processing systems, allows the scattering behaviour of any object to be modelled as a complex matrix. The theory is that if the amplitude and polarization of an EM wave is measured (before and after it is scattered by a target) then a

matrix can be formulated which can simulate the same behaviour as the scatterer. The logical extrapolation of this is that if a "data bank" of these matrices exists, each identified with particular canonical shape, then any composite shape can be identified by a process of matrix comparison. Equally important, if the scattering behaviour of basic shapes is known (particularly in a polarization sense), then two adjacent objects may be distinguished from each other by making finite comparisons of their scattering behaviour [11].

A particularly useful illustration showing the applicability of matrix mathematics to scattering problems can be found in a 1983 article by Becker [12]. Furthermore, a paper written by Polikarpov [13] in 1985 shows how the scattering matrix can be measured using linear regression techniques and the method of least squares. The following chapter will address portions of this theory and will show how matrix mathematics can be combined with the Poincare sphere to assist in the analysis of polarimetric data.

2.6 Results of Previous Polarimetric Studies

Several papers have analyzed the interrelationships between a radar's polarization and the associated return from a target. The following section summarizes the three most topical of those papers.

2.6.1 Radar Backscattering From Trees. This paper mentioned earlier that both civilian and military authorities were involved in SAR systems and research

into polarimetrics. A paper which typifies the interest of the civilian sector was written in 1988 as a result of Polarimetric studies conducted on various tree types during the 1987 growing season [14]. This study concluded that the measured Radar Cross Section (RCS) of trees varied according to the selected radar polarization. Moreover, by a series of exhaustive measurements the author was able to show that a discernible difference existed between the RCS behaviour of different species of trees. These conclusions are significant as they demonstrate that the use of polarimetric data may lead to classification capabilities well beyond those attainable by even the highest resolution non polarimetric SAR systems.

2.6.2 Dihedral Corner Reflectors. A seemingly inordinate amount of the unclassified polarimetric data is concerned with studies which examine the RCS characteristics of dihedral corner reflectors [15]. However, the motive behind this apparent pre-occupation with dihedrals is that they typify man made structures. Such structures tend to be angular and have high RCS's whereas natural objects usually have "softer " lines and lower RCS's. This important Geometrical observation is instrumental in the matrix formulations mentioned earlier in this paper. Moreover, at a gross level this differing RCS behaviour explains why many military platforms e.g. Stealth Aircraft have predominantly curved shapes [16].

Two very good discussions on the importance of dihedrals to polarimetric studies are to be found in [17] and [18]. In the former paper the author demonstrates excellent correlation between the predicted and the measured reaction of a dihedral

corner reflector to varying degrees of polarization. This encouraging result is somewhat tempered by [19] which concludes that although math models can predict polarimetric behaviour, those same models can give misleading information if the target geometry is not carefully considered. This "constrained success" typifies the current status of Polarimetric studies.

2.7 *Summary*

The study of Polarimetric radar theory is an emerging field with its own subset of scientific jargon, an understanding of which is necessary in order to comprehend the books and journals written on this subject. Similarly, an appreciation of the differences in performance between conventional radars and SAR systems is central to understanding the transition between the well known technology of the conventional systems and the "still being investigated" technology of the Polarimetric SAR systems. Finally, there are several mathematical tools, in varying stages of development, which can be used to predict the scattering behaviour of canonical shapes.

Current attempts to synthesize the tools and the technology into a reliable predictive mechanism have met with guarded success. Complete success will be difficult to achieve due to the many parameters involved and the statistical nature of EM propagation. Nevertheless, the latent benefits of integrating Polarimetrics into the radar recognition and identification process are considerable and have the potential to revolutionise the performance capabilities of future SAR systems.

III. Theory

3.1 Introduction

As mentioned in Chapter 2 there is a need to present the applicable SAR theory so that the methodology and results of this thesis effort can be placed in the context of their potential to contribute to the overall improvement of SAR performance. There is also a need to address how practical systems could use the derived polarimetric data in the decision or classification process. Accordingly,this chapter will consist of the following sections:

1. Non Polarimetric SAR Resolution Limits.
2. Thesis Methodology.
3. Perceptron Network Theory.

3.2 Non Polarimetric SAR Resolution Limits

3.2.1 Categories of SAR Systems. SAR systems can be classified into one of three major categories. These categories, and a brief discussion of their operational concepts, are as follows:

1. Strip Map SAR. In strip map SAR the radar's real beam is fixed in both azimuth and elevation. Accordingly, the area to be mapped is a function of the platform's flight path.

2. Spotlight SAR. In spotlight SAR the radar is steered such that the area of interest is illuminated throughout the processing interval.

3. Scan Mode. Scan mode SAR is a mixture of both strip and spotlight SAR. In scan mode the area to be illuminated is a function of both platform path and beam steering.

4. Inverse SAR (ISAR). The theory associated with ISAR has much in common with the previously mentioned SAR modes. The single most fundamental difference between SAR and ISAR is that the latter relies on the target's motion, rather than the platform's motion, to perform the mapping function.

The radar prediction software used for this thesis effort is the Synthetic aperture radar prediction Tool (SarTool^{tm}). SarTool^{tm} was developed by The Analytic Sciences Corporation (TASC) and is capable of simulating all of the above radar modes. A brief description of the salient details of SarTool^{tm} is contained in Appendix A. For the purposes of this thesis, the software input files were configured to simulate the ISAR mode of operation. This involved holding the SAR platform stationary and spotlighting a rotating pedestle upon which the targets were located. Accordingly, the following theory will address the resolution cell sizes applicable to spotlight/ISAR systems and the sampling requirements of those cells.

3.2.2 Cross Range Resolution. The cross range resolution of a SAR system can be explained in terms of physical constraints (i.e. antenna size) or in terms

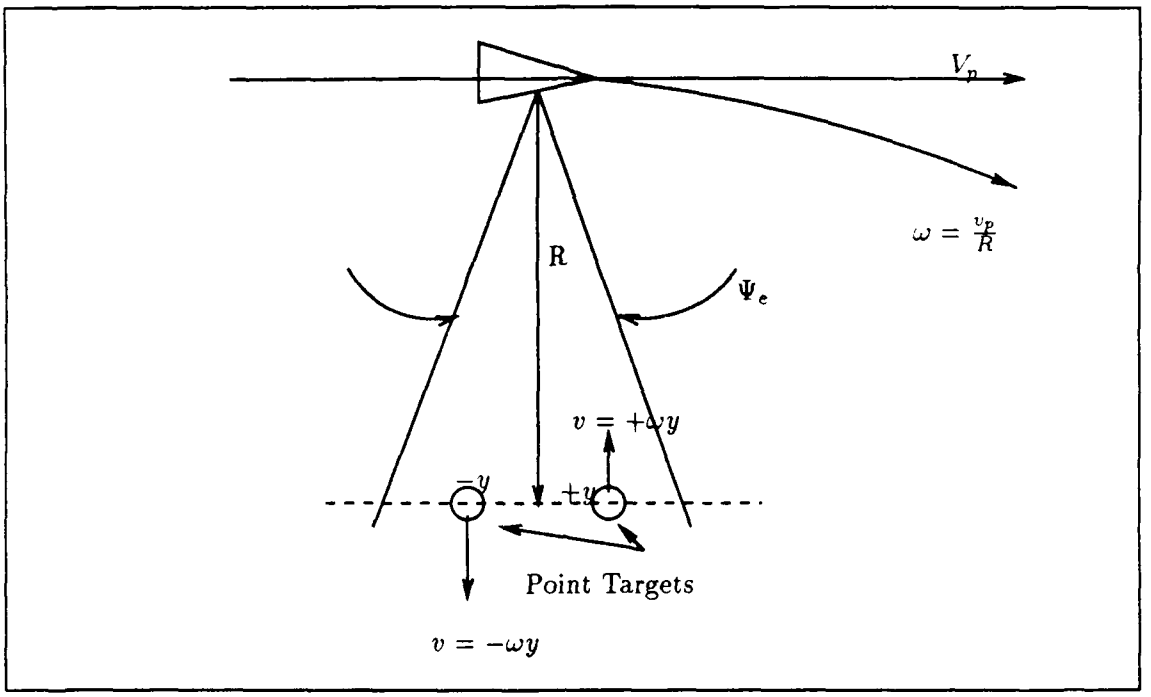


Figure 3.1. Doppler Cross Range Resolution[20]

of discernible differences in target doppler. The latter approach is more applicable to ISAR: accordingly, the following development, [20] defines cross range resolution primarily in terms of target doppler. Wehner's doppler development centers on Figure 3.1, where:

R	=	range to centroid of targets
V_p	=	straight line velocity of platform
ω	=	angular velocity of platform
Ψ	=	equivalent rectangular beamwidth

In accordance with Figure 3.1 target 1 and target 2 will have respective doppler shifts of $-(2\tilde{f}/c)wy$ and $+(2\tilde{f}/c)wy$, where $\tilde{f}$ is the radar's center frequency. Ac-

cordingly, the difference in doppler frequency between the two targets is given by equation (3.1) shown below:

$$\sigma f_d = \frac{4}{c}\omega y \tilde{f} \tag{3.1}$$

Rearranging equation (3.1) and substituting $\Delta f = \sigma f_d$ = system doppler frequency resolution leads to the cross range resolution (Δr_c) being defined by the following equation:

$$\Delta r_c = \frac{c}{2\omega \tilde{f} \Delta \tilde{f}_d} \tag{3.2}$$

Recognizing that for ideal processing of a uniform signal $\delta f_d = 1/T$ then equation (3.2) can be re-cast in any one of the following forms:

$$\Delta r_c = \frac{\lambda}{2\omega T} \tag{3.3}$$

$$\Delta r_c = \frac{\lambda}{2\psi_e} \tag{3.4}$$

$$\Delta r_c = \frac{\lambda}{2\Delta\Phi} \tag{3.5}$$

where

$\Delta\Phi$ = look angle = ωT

Wehner also shows, via integration of the response for a rectangular beam

antenna, that the cross range resolution can also be expressed by equation (3.6) which is shown below:

$$\Delta r_c = \frac{l}{2} \tag{3.6}$$

where

l = aperture length of the physical antenna in the azimuth plane

3.2.3 Slant Range Resolution. SAR systems rely on some form of transmitter pulse coding to achieve slant range resolution improvement over conventional radars. In SarTool^{tm} this pulse coding takes the form of *CHIRP* or pulse compression coding. The frequency - time representation of a chirp pulse is shown in Figure 3.2

As illustrated in Figure 3.2 the chirp pulse has a unique characteristic over every finite time interval. If this were not the case then SAR radars could only achieve slant range resolutions of $CT_1/2$ where T_1 = transmitter pulse width. However, to satisfy the Nyquist sampling theorem each echo from the linear FM pulse must be sampled at β times per second. This leads to the following simple expression for slant range resolution:

$$\Delta r_s = \frac{C}{2\beta} \tag{3.7}$$

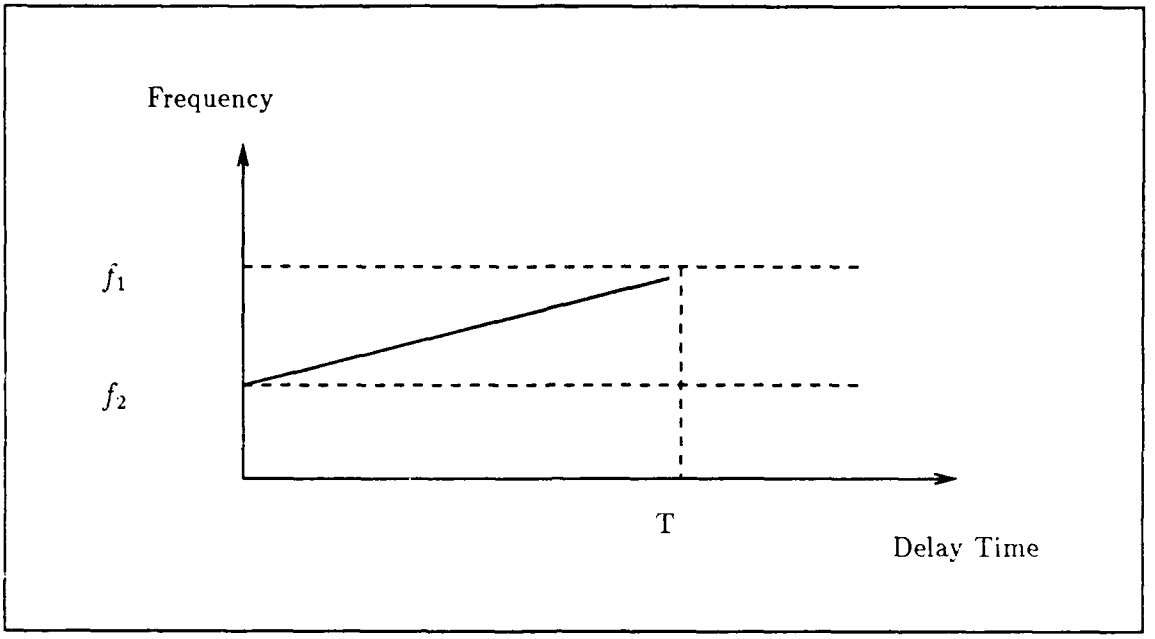


Figure 3.2. Chirp or Pulse Compression

where

Δr_s = slant range resolution

The actual sampling criteria for both cross and slant range resolution can be derived using Figure 3.3. To adequately sample in slant and range the following relationships must apply:

$$\eta_s \Delta r_s \geq \frac{CT_1}{2} \tag{3.8}$$

$$\eta_c \Delta r_c \geq R\psi \tag{3.9}$$

where

η_s = samples collected during T_1

η_c = samples collected along r ψ

Rearranging equation (3.8) leads to :

$$\eta_s \geq \frac{CT_1}{2\Delta r_s} = T_1\Delta \tag{3.10}$$

where

Δ = chirp pulse bandwidth

Similarly, re-arranging equation(3.9)leads to:

$$\eta_c \geq \frac{R\psi}{\Delta r_c} \tag{3.11}$$

or

$$\eta_c \geq \frac{2R\Delta\phi\psi_e}{\lambda} \tag{3.12}$$

The above discussion enables the maximum resolution capabilities of a conventional SAR system to be calculated given that the necessary sampling criteria are adhered to. Both equation (3.10) and equation(3.12) are predicated on one complex sample per resolution cell. SarTool^{tm} satisfies the number of transmitted pulses and

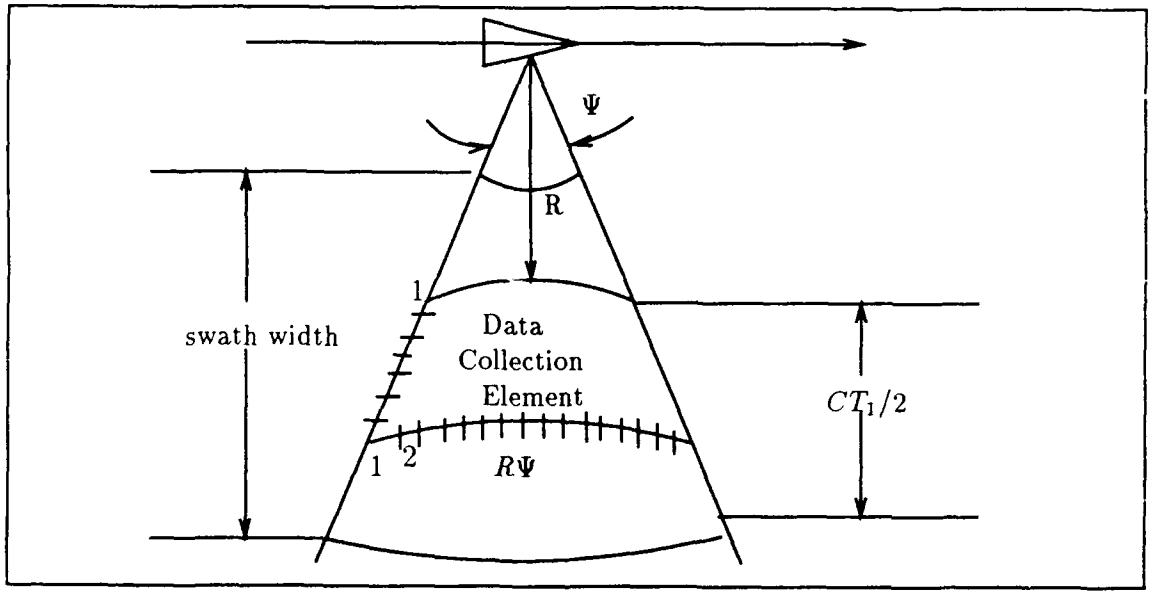


Figure 3.3. SAR Data Sampling [20]

the necessary increment angles by equation (3.13) and equation (3.14).

$$np = \frac{ARC}{\Delta\Theta} \tag{3.13}$$

$$\Delta\Theta = \frac{C}{2nadacos(\Phi)} \tag{3.14}$$

where

np	=	number of transmitted pulses
ARC	=	desired portion of angular arc of the turntable
$\Delta\Theta$	=	angular increment in aspect angle
na	=	number of azimuth bins
da	=	azimuth resolution
Φ	=	elevation boresight

Of note is that although the preceeding discussion determines the number of pulses required for a particular resolution it does not address the use of polarimetric information. Accordingly, the resulting resolution relationships are absolute limitations of conventional SAR systems. As mentioned in Chapter 1 this thesis effort is directed at ascertaining whether or not intra resolution cell classification can be achieved via the use of polarimetric data. Consequently the following section will address the methodology used to collect and analyze the intra resolution cell polarimetric data.

3.3 Thesis Methodology

3.3.1 Introduction. As indicated in the previous section of this chapter, current SAR systems have target identification limits that are unassailable without the use of polarimetrics. Consequently, this thesis, (which is intended to be the first of several similar efforts) is directed towards assessing the viability of including polarisation within classification algorithms. In particular this thesis addresses the following areas of concern:

1. The validation of SarTool^{tm}.
2. The development of a composite software package to aid subsequent investigations into polarimetric behaviour.
3. The examination of polarimetric data from single and double bounce canonicals within a single resolution cell.

3.3.2 The Validation of SarTool ^{tm}. The sponsor of this thesis was interested in determining how accurately the documentation of SarTool^{tm} reflected the performance of the software. Due to the limited time available, and the secondary nature of this task, it was agreed to validate only the dihedral section of the documentation and software. This validation entailed programing MATHCAD^{tm} and Mathematica^{tm} (both of which are math packages) with the equations purportedly embedded in the SarTool^{tm} algorithms. To facilitate this validation, the parameters used in the math packages and SarTool^{tm} corresponded to those use in an IEEE

article written by Griessner and Balanis [17] and the azimuthal scope was limited to 90 degrees. The actual comparison was achieved by "stripping" output SarTool^{tm} data and plotting that data within a Mathematica^{tm} routine.

In addition to the above validation several experiments were conducted in order to ascertain how responsive SarTool^{tm} was to varying input configurations. The resulting plots and associated findings of the above investigations are contained in Chapter IV of this report.

3.3.3 A Composite SarTool ^{tm} Software Package. In the early stages of this thesis it became evident that SarTool^{tm}, in its pristine form, would not readily support an investigation of polarimetrics. The major shortfalls of SarTool^{tm} were assessed as involving the input reflector files and the output image files. The input reflector files are the means whereby the canonicals are defined and then moved and rotated within the radar's field of view. These files are normally formed as outputs of the EUCLID solid modeller sorfware which was unavailable to the sponsor and attempts to alter the reflector files dynamically without the Euclidean software proved to be impractical. The normal output of SarTool^{tm} consists of parametric tables and image files, neither of which were in a suitable form for polarimetric analysis. As a result of these shortcomings the software packages described below were designed and built.

1. Reflector File Software. This software enables the operator to select any combination of SarTool^{tm} canonical shapes and translate or rotate those shapes within the resolution cell. The reflector file software can then be used as an input to a commercial software package called "d3d" which allows visual inspection of the canonicals.

2. Shell program. The shell program is a routine which converts the above ASCI reflector files into binary files and then "runs" SarTool^{tm} on each of those files. The resulting output SarTool^{tm} data files are then "stripped" and moved by File Transfer Protocol (FTP) from the SUN workstation to a Macintosh computer where they are processed by the Mathematica^{tm} routine.

3. Strip Software. The software which performs the "stripping" function is run from within the shell programme and is used to transform the output from the SarTool^{tm} Range Profile Data Base (RPDB) files into the "in phase" and "quadrature" components. These in phase and quadrature files are subsequently used as the input to the Mathematica^{tm} routine.

4. Mathematica^{tm}. A routine was written within Mathematica^{tm} which transformed the stripped SarTool^{tm} data into polarization plots and as "animated" or stationary polarization points on the Poincare sphere. The Mathematica^{tm} routine was based on Boerner's article [21] and is further described in the following paragraphs.

3.3.4 The Poincare Sphere. The Poincare sphere was briefly introduced in Chapter 2 of this report and was identified as being of considerable use in the study of polarimetrics. The applicability of the Poincare sphere to the study of polarization is due to its ability to completely characterize the polarimetric behaviour of a radar target. This is achieved by plotting the optimal polarizations (co-polarization (CO-POL) and cross polarization (X-POL)) on the Poincare sphere. Kennaugh was the first to identify these "optimal polarizations" [22] and his results are used in the development of Boerner's article the relevant theory of which is shown below:

Assuming that no losses are incurred then the transformation from one orthogonal polarization to another may be expressed as shown by equation(3.15) below:

$$[S'(A'B')] = [T]^T[S(AB)][T] \tag{3.15}$$

where:

T is the normalized unitary transformation matrix which can be expressed in the form of equation(3.16):

$$[T] = (1 + \rho\rho^*)^{-1/2} \begin{bmatrix} 1 & -\rho^* \\ \rho & 1 \end{bmatrix} \tag{3.16}$$

ρ = the complex transformation parameter.

Substituting equation(3.16) into equation(3.15) and expanding leads to the following equations(3.17, 3.18, 3.19, 3.20)

$$S'_{A'A'} = (1+\rho\rho^*)^{-1}[S_{AA} + \rho^2 S_{BB} + \rho(S_{AB} + S_{BA})] \quad (3.17)$$

$$S'_{A'B'} = (1+\rho\rho^*)^{-1}[-\rho^* S_{AA} + \rho S_{BB} + S_{AB} - \rho\rho^* S_{BA})] \quad (3.18)$$

$$S'_{B'A'} = (1+\rho\rho^*)^{-1}[-\rho^* S_{AA} + \rho S_{BB} + S_{BA}) - \rho\rho^* S_{AB}] \quad (3.19)$$

$$S'_{B'B'} = (1+\rho\rho^*)^{-1}[\rho^{*2} S_{AA} + S_{BB} - \rho^*(S_{AB} + S_{BA})] \quad (3.20)$$

The above relationships satisfy the transformation invariants i.e.

$$\det([S(A,B)]) = \det([S'(A'B')]) = \text{invariant}$$

and finally

$$\text{Span}([S(A,B)]) = |S_{AA}|^2 + |S_{AB}|^2 + |S_{BA}|^2 + |S_{BB}|^2 = \rho = \text{Span}([S'(A'B')])$$

The previously mentioned CO-POLS (for minimum polarization) are obtained by setting $S'_{A'A'}$ and/or $S'_{B'B'}$ in equation(3.17 or3.20) to zero and for the monostatic case leads to equation(3.21) shown below:

$$\rho^c_{1,2} = \frac{-S_{AB} +/- \sqrt{S^2_{AB} - S_{BB}S_{AA}}}{S_{BB}} \quad (3.21)$$

Similarly, by setting $S'_{A'B'}$ and $S_{B'A'}$ to zero the X-POL (for maximum polarization)

is defined by the following equation(3.22):

$$\rho^{x}_{1,2} = \frac{R_1 + / - \sqrt{r_1^2 + 4R_2R_3}}{2R_2} \tag{3.22}$$

where

$$\begin{aligned} R_1 &= (|S_{BB}|^2 - |S_{AA}|^2 - S_{AB}S^{*}_{BA} + S_{BA}S^{*}_{AB}) \\ R_2 &= (S_{BB}S^{*}_{BA} + S_{BA}S^{*}_{AA}) \\ R_3 &= (S_{AA}S^{*}_{AB} + S_{AB}S_{BB}) \end{aligned}$$

The latitude and longitude for both the CO-POL and the X-POL are determined via equation(3.23) and equation(3.24) as shown below:

$$\cos(\theta) = -\frac{1 - |q|^2}{1 = |q|^2} \tag{3.23}$$

and

$$\tan(\phi') = -\frac{Im|q|}{Re|q|} \tag{3.24}$$

The variable q in equation(3.23 and 3.24) is defined by the following equation(3.25):

$$q = \frac{1 - j\rho}{1 + j\rho} \tag{3.25}$$

The operation and other relevant facets of the Mathematica ^{tm} routine are documented within the programme a copy of which appears at Appendix B. Upon

initialization, the Mathematica ^{tm} routine automatically reads in data files and any required operative programmes, for example parametric or animation routines. Programme output consists of Poincare data and polarization plots which reflect the polarization behaviour of the canonical within the resolution cell. Examples of this output are discussed in the results section of this thesis.

3.3.5 Polarimetric Data Within a Cell. The above theory and associated software provided a ready and visible means whereby intra resolution cell polarimetrics could be examined. The actual methodology employed is contained in the remainder of this chapter.

The first phase of the experiment consisted of verifying that a pure two bounce canonical (dihedral) and a pure single bounce canonical (flat plate) exhibited the correct relationships between their individual optimal polarizations. This was achieved by "placing" each of the shapes centrally in the resolution cell and examining the resulting output. Subsequent to this phase separate experiments were conducted with each of the canonicals to determine whether or not the polarization characteristics were a function of the shapes' position within the resolution cell. This involved moving the canonicals in all three dimensions and then examining the resulting series of data for corelation. The actual paths that the shapes were moved along are shown in Chapter IV.

The second phase of the experiment entailed placing the two canonicals within a single resolution cell and examining the resulting composite polarization charac-

teristics. Initially the geometric center of the two canonicals was placed in the center of the cell and the returns were analyzed in an attempt to ascertain whether their combined prescence could have been determined without apriori knowledge. Subsequent to this the composite shape was moved around within the cell to determine any positional dependency of their combined polarization signature. The final segment of this phase involved simultaneously moving each shape along a different path within the resolution cell to ascertain the dependence of the polarization characteristics on canonical speration. The exact intra cell trajectories used are contained in ChapterIV.

The final phase of the experiment involved changing the relative sizes of the two canonicals in an attempt to ascertain whether there was a size threshold above which one of the canonicals' polarization characteristics would dominate the cell's polarization return. The size of the canonicals used in this and the two previous phases are detailed in ChapterIV.

In each of the above phases SarTool^{tm} was configured such that both the azimuthal scan was limited to $\pm$ 45 degrees either side of the shapes' center while the "elevational angle" was held constant at 90 degrees. Furthermore in all cases the canonicals were held "face on" to the transmitter as illustrated in Figure 3.4

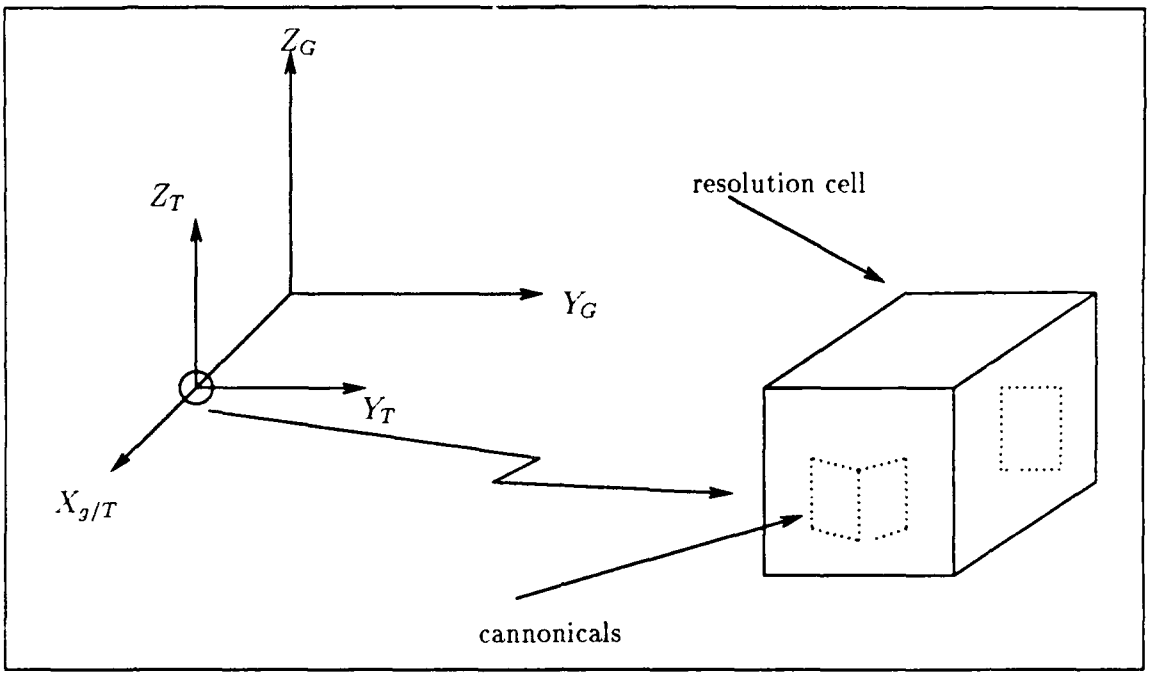


Figure 3.4. SarTool^{tm} Target Orientation

3.4 Perceptron Analysis

At the conclusion of the above experiments an investigation into the applicability of perceptron networks to the task of target classification was conducted. This investigation was not intended to be all encompassing but was designed to provide an initial insight into the potential of coupling polarimetric data to perceptron network analysis. Accordingly, the next few paragraphs will briefly summarize the characteristics of the two perceptron networks used during this thesis effort.

3.4.1 Single Level Perceptron. This network was based on an IEEE article written by Lippman [23] in which he presents a perceptron convergence procedure originally developed by Rosenblatt [24]. Rosenblatt's algorithm is based on the

single layer perceptron. This type of network can employ continuous as well as discrete inputs and is capable of quickly recognizing simple patterns. The algorithm developed by Rosenblatt involved initializing the connection weights and threshold values to small random non zero values. The perceptron is then presented with the classes of interest and is prompted to make a classification. When the network returns an incorrect classification the weights are updated, otherwise they remain unchanged. Accordingly, over a period of time, the weights converge such that the network achieves its maximum classification performance. A gain term exists within the algorithm which allows the operator to opt for faster adaptation weights at the expense of weight stability. The resulting network is ultimately capable of constructing a hyperplane which bisects the two separable classes thereby facilitating simple class classification. Figure 3.5 shows the basic perceptron unit and associated hyperplane. A program was written which incorporated Rosenblatt's algorithm and was used as a classification aid during the experimental analysis section of this thesis. A copy of this program and some sample outputs appears at Appendix B.

3.4.2 Multi Layer Perceptron. As mentioned above the single level perceptron is adept at recognizing simple and readily separable data. Unfortunately, the network experiences much difficulty when the decision regions of the classes overlap. Accordingly, the more complex experimental data was used as an input to a multi layer perceptron network. The network used was developed by G.Tarr [25] and employed Werbose's back propagation algorithm with a momentum term included, as

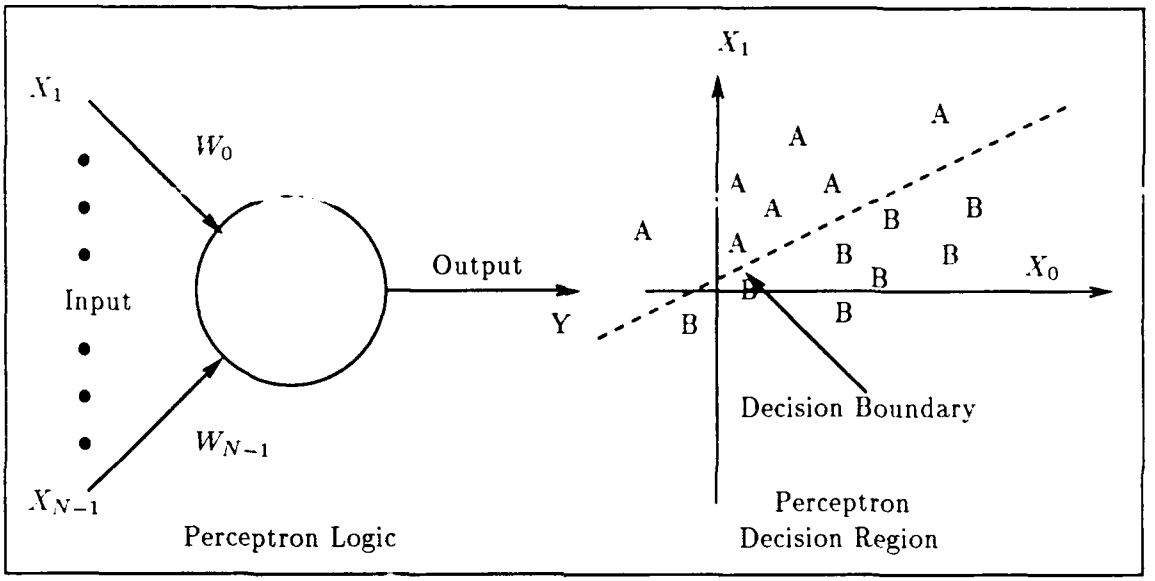


Figure 3.5. Single Layer Perceptron

described in Lippman's article [23]. The resulting program is an extremely useful piece of software which allows the operator to structure the network according to the level and complexity of the input data. The network used in this thesis effort consisted of 56 input nodes and two hidden layers. The first and second hidden layer consisted of 20 and 10 nodes respectively. Further details regarding this network appear in Chapter 4.

3.5 Summary

The methodology adopted in this thesis involved understanding the limitations of non polarimetric SAR resolution and then designing a software environment conducive to the examination of polarimetric SAR data. The examination of the polarimetric data was ultimately directed at ascertaining whether polarimetric an-

alyzes could achieve intra cell identification without apriori knowledge. Perceptron networks were used during this process. The results of the above endeavours are contained within Chapter 4 of this report.

IV. Results

4.1 Introduction

As noted in Chapter 3, this thesis was intended to investigate three primary areas of concern. These areas are as detailed below:

1. The validation of SarTool^{tm}.
2. The development of a composite software package.
3. The investigation of intra resolution cell polarimetrics.

In addition to the above primary areas of interest the collected polarimetric data was processed and then used as input to two perceptron networks in an attempt to gauge the potential of Artificial Intelligence (AI) in the discrimination of polarimetric patterns. The results of the AI endeavour and of the primary areas of investigation are described in the following sections of this chapter.

4.2 The Validation of SarTool^{tm}

The development of SarTool^{tm} is an ongoing process and as such the sponsor of this thesis needed to know how well the existing documentation correlated with the currently released version of the software. Because an exhaustive validation of the software and documentation was beyond the scope and time limitations of this thesis effort, it was agreed to only investigate the dihedral and flatplate canonicals

for correspondence between theoretical results, documented algorithms and software performance. The basis for the validation was chosen as being Griesser's article titled *Backscatter Analysis of Dihedral Corner Reflectors Using Physical Optics and the Physical Theory of Diffraction* [17] in which he presents the theoretical and measured data of several dihedrals. Griesser used a 9.4GHZ radar on a dihedral with dimensions of 5.6086 λ. The Equations shown in [26] were coded using MATHCAD^{tm} and Mathematica^{tm} and were then compared to those shown in Griesser's article. In both cases there were marked differences between Griesser's article and the output predicted by the SarTool^{tm} documentation. Investigations with TASC revealed that the original equations for the dihedral cited in the TASC documentation were incorrect although the actual software returned the correct prediction.

A further anomaly with the SarTool^{tm} package was discovered during the preliminary stages of experiment design. The version of SarTool^{tm} held by the sponsor failed to acknowledge the presence of canonicals within a resolution cell unless the canonicals were in excess of 15 λ (λ being the wavelength of the target radar). TASC acknowledged this inadequacy and released a software patch which apparently corrects the problem.

One final point of interest regarding the performance of SarTool^{tm} is that when two canonicals are within a single resolution cell the software does not make provision for any interactions between the two shapes. This results in the cell response being merely an addition of the individual responses of the two canonicals. This revelation

will be discussed in greater detail in the following section of this chapter.

The above observations regarding the accuracy or limitations of SarTool^{tm} are not intended to be extrapolated into a general judgement of the entire software package. Nevertheless, even this limited investigation did reveal several unexpected anomalies and certainly indicates that any future experiments or thesis efforts involving SarTool^{tm} should be preceded by an examination of the involved software modules and their documentation. It should also be noted that TASC personnel were most helpful on every occasion when these problems were brought to their attention.

4.3 The Development of a Composite Software Package

A requirement of this thesis was to develop a software package capable of interfacing with SarTool^{tm} such that SAR data could be extracted and used to investigate radar polarimetric behavior within a single resolution cell. Appendix B contains listings of the software developed and used during this thesis effort. The following paragraphs discuss the functions of each of these software modules and illustrates how they interact to form a composite data collection software package.

4.3.1 Data Collection Procedure. Initially several directories were established on the SUN workstation corresponding to each of the major experiments described in the following section. Within each of these directories were sub directories dedicated to the various combinations of geometric shapes being examined. Each of the sub directories contained the following three files:

1. SarTool^{tm}.dat File.

2. Canonical Reflector File.

3. $Mod_{rfl}.dat$ file.

The first two of the above files are integral parts of the SarTool^{tm} package and as such are described in Appendix A. The third file i.e. the $mod_{rfl}.dat$ file defines the start and finish position of the canonicals and the number of "steps" or increments taken between those two positions. The $mod_{rfl}.dat$ file is written by the operator according to the required experiment configuration and is used as an input to the mod_{rfl} software described in Appendix A. SarTool^{tm} is invoked at the start position, the stop position and at every increment position. A sample $mod_{rfl}.dat$ file is shown in Table(4.1).

Table 4.1. Example of a $Mod_{rfl}.dat$ File

'dihed'	NAME OF INPUT REFLECTOR FILE
'10'	NUMBER OF INCREMENTS
'2'	CANONICAL IDENTIFIER
'X,Y,Z'	START POSITION OF CANONICAL
'X,Y,Z'	FINISHING POSITION OF CANONICAL

The last three lines of Table(4.1) can be repeated up to 100 times thereby allowing a maximum of 100 canonicals to be simultaneously moved through the radar's field of view. As mentioned earlier one of the aims of this thesis was to facilitate the ready acquisition of polarimetric data; accordingly, the data collection process was automated as much as possible and consists of the following steps:

1. *Step 1.* In each SUN experiment sub directory there are initially three files i.e. the SarTool^{tm}.dat file, the Canonical Reflector File, the $Mod_{rfl}.dat$ file and a shell program entitled "experiment.sh". At the commencement of each experiment the shell program is invoked which then performs the following functions:

   (a) Runs mod_{rfl} thereby creating formatted reflector files corresponding to the input parameters of the $mod_{rfl}.dat$ file.

   (b) Converts the formatted reflector files to binary files.

   (c) Copies the above binary files into a temporary reflector files which are subsequently "called" by the Sar.dat file.

   (d) Invokes SarTool^{tm} thereby resulting in a rpdb file for each "step" position contained within the $mod_{rfl}.dat$ file.

   (e) Runs the dbstrip program which outputs stripped "in phase" and "quadrature" data corresponding to each of the rpdb files. The series of files containing the "stripped" data are titled "$test_{out}.0$" through to "$test_{out}.X$" where X is the number of increments taken through the resolution cell.

   (f) Erases all of the reflector files to facilitate the next run.

2. *Step 2.* All of the above actions associated with the shell program are "transparent" to the operator. However, when the shell program has finished running

the resulting "$test''_{out}$ output files must be transferred to the Macintosh computer via File Transfer Protocol and NCSA Telnet.

3. *Step 3.* Once the test output files are transferred to the Macintosh they are individually processed by the Mathematica^{tm} routine with the resulting output being stored within individual experiment sub directories.

4. *Step 4.* The output from the Mathematica^{tm} routine is then converted into an appropriate form for input into either the single level perceptron or into G.Tarr's multi level perceptron network. Conversion for Tarr's network is performed by the "convert" program which is listed in Appendix B.

4.3.2 Ancillary Software. The above procedures required both coupling between the SUN workstation and the Macintosh computer and the use of a transfer protocol, for example, NCSA Telnet. Moreover, an IBM (AT) Personal Computer was used quite extensively as a development tool, thereby necessitating the use of the TOPS Translator. Input file editing was done via a combination of Macintosh generic software and the VI editor.

A diagrammatic representation showing the integration of the composite software package is shown in Figure 4.3.2.

4.4 Intra-Cell Polarimetrics

The investigation of Intra-Cell polarimetrics was centered around four basic experimental blocks. A discussion of these experimental blocks and an interpretation

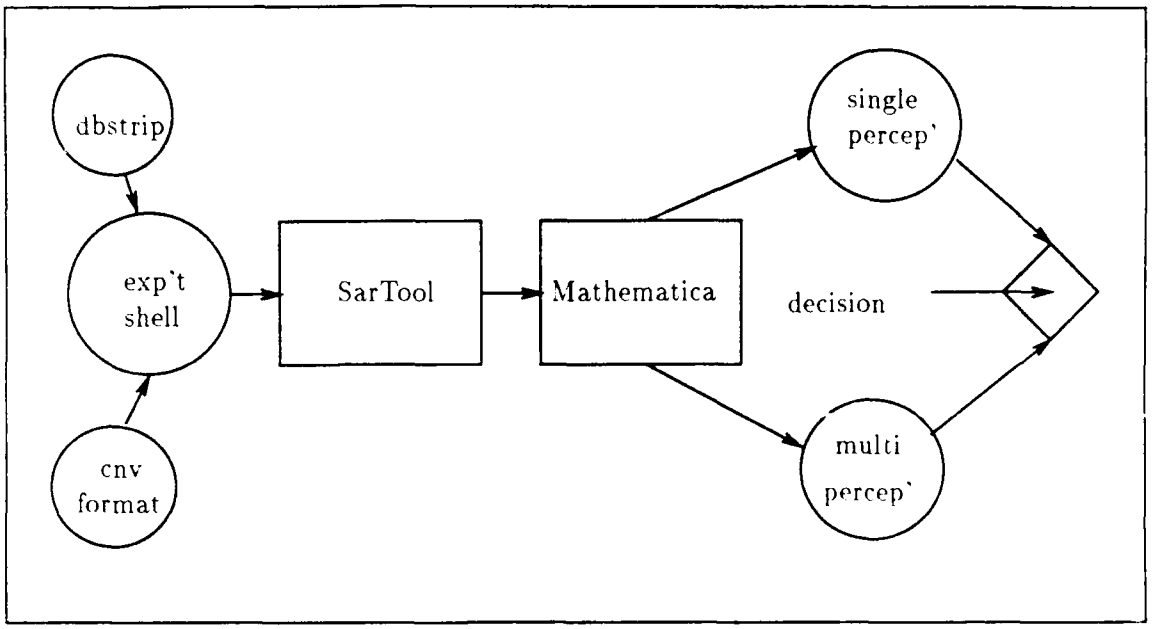


Figure 4.1. Integration of Software Packages

of their results is contained in the following paragraphs.

4.4.1 Experimental Block One. Experimental block one consisted of three sub experiments. The objective of these sub experiments was to ascertain whether or not SarTool^{tm} made any distinction based on the position of the canonical within the resolution cell. To determine this the SarTool^{tm}.dat file was configured as shown in Table 4.4.1.

In addition to the above configuration the resolution cell was given a nominal height of 2m thereby resulting in a 2m cube resolution cell. This cell and the numbering convention adopted are shown in Figure 4.4.1.

Sub experiment one was divided into three sections as described overleaf:

Table 4.2. SarTool^{tm} Sensor and System Data (Sartool.dat)

'exp1'	TITLE OF SIMULATION RUN
'dihed'	SPECIFIES INPUT REFLECTOR FILE
'0'	X POSITION OF SENSOR
'0'	Y POSITION OF SENSOR
'1'	Z POSITION OF SENSOR
'90'	FLIGHT PATH ANGLE OF SENSOR
'0'	DIVE ANGLE OF SENSOR
'0'	AVERAGE SENSOR SPEED (SAR MODE)
'0'	X POSITION OF TARGET REL TO MAP CENTER
'0'	Y POSITION OF TARGET REL TO MAP CENTER
'0'	Z POSITION OF TARGET REL TO MAP CENTER
'0'	EULER ANGLE PHI FOR TGT FRAME ROTN
'0'	EULER ANGLE THETA FOR TGT FRAME ROTN
'0'	EULER ANGLE PSI FOR TGT FRAME ROTN
'9.4 GHZ'	TRANSMITTED FREQUENCY
'2.0'	ELEVATN 3db BEAMWIDTH (DEG)
'2.0'	AZIMUTH 3db BEAMWIDTH (DEG)
'.01'	ELEVATION BORESIGHT ANGLE (DEG)
'90.0'	AZIMUTH BORESIGHT ANGLE (DEG)
'2.0'	RANGE RESOLUTION (m)
'2.0'	AZIMUTH RESOLUTION (m)
'1'	NUMBER OF RANGE BINS
'1'	NUMBER OF AZIMUTH BINS
'200'	NUMBER OF PULSES TRANSMITTED
'1'	TARGET DATABASE RECALCULATION I/VAL
'f'	TERRAIN GENERATION OPTION (SAR)
'0'	AVERAGE TERRAIN RCS (dbsm)
't'	SAR SPOLTLIGHT OPTION
't'	ISAR MODE OPERATION
'0'	ASPECT ANGLE INCREMENT FOR QKTOOL
'f'	MOTION COMPENSATION OPTION
'f'	REFLECTOR THINNING OPTION
'0'	THINNING THRESHOLD VALUE (dbsm)
't'	TARGET SHADOWING OPTION
't'	TERRAIN SHADOWING OPTION
'f'	REFLECTOR I/P REPORT OPTION
't'	REFLECTOR D/BASE REPORT OPTION
't'	RANGE PROFILE OUTPUT OPTION

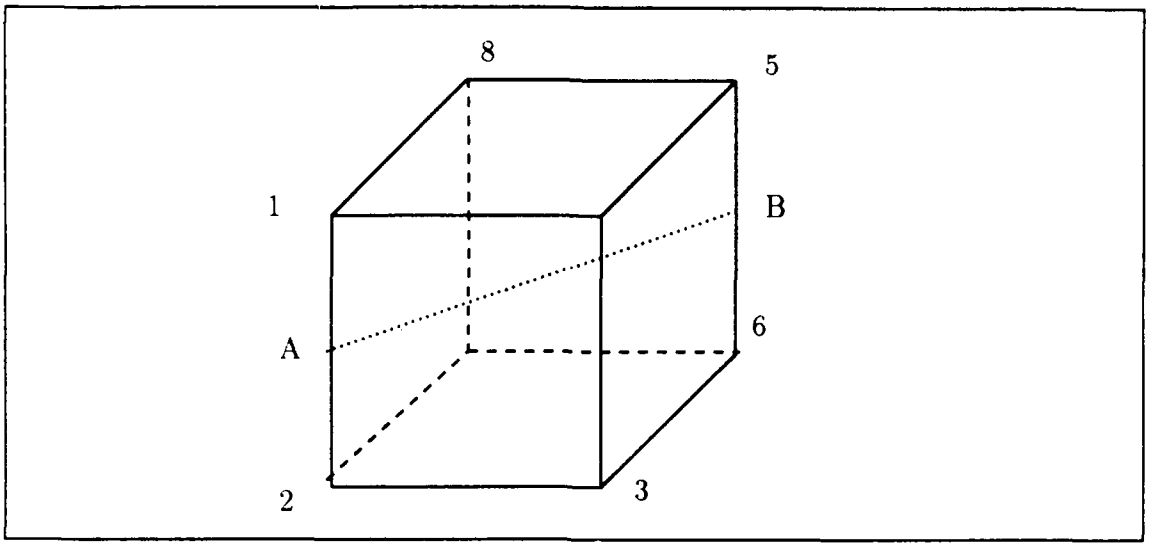


Figure 4.2. Experimental SAR Resolution Cell

1. Dihedral. A dihedral having sides of length 8.944 cm was moved from corner 1 to corner 6 in 10 steps.

2. Flatplate. A square flatplate (17.88cm) was moved in 10 increments along the same path as the dihedral above.

3. Combination. The two canonicals mentioned above were combined and moved together in 10 steps from corner 1 to corner 6.

Sub experiments two and three used the same sized canonicals and were divided into the same three sections as shown above. However, in sub experiment 2 the path taken was from corner eight to corner three while in sub experiment 3 the path selected was from A to B. Samples of the relevant Poincare sphere, co-polarization

and span plots taken during Experiment block 1 are shown in Figures 4.3 to 4.11. Note that the figure nomenclature "x/y" signifies experiment x, sub experiment y.

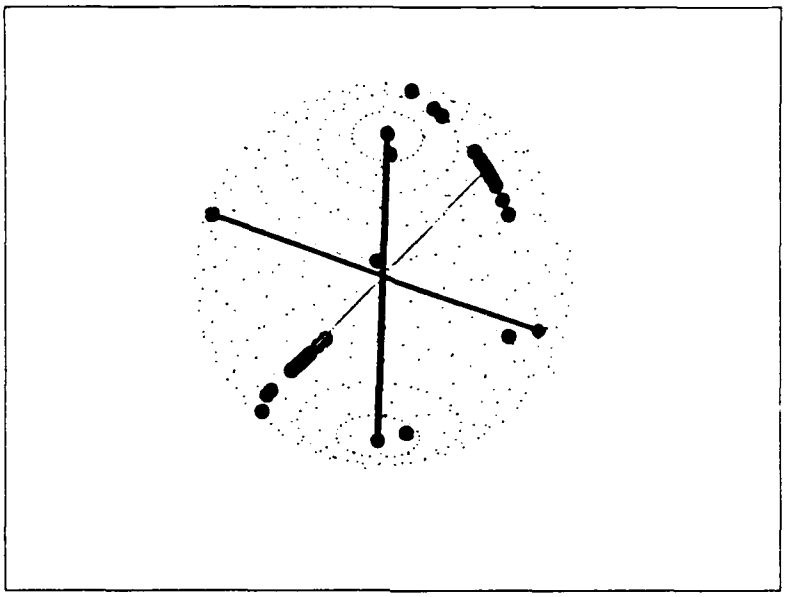


Figure 4.3. Exp 1/1 - Dihedral (Path 1 to 6) Poincare Sphere

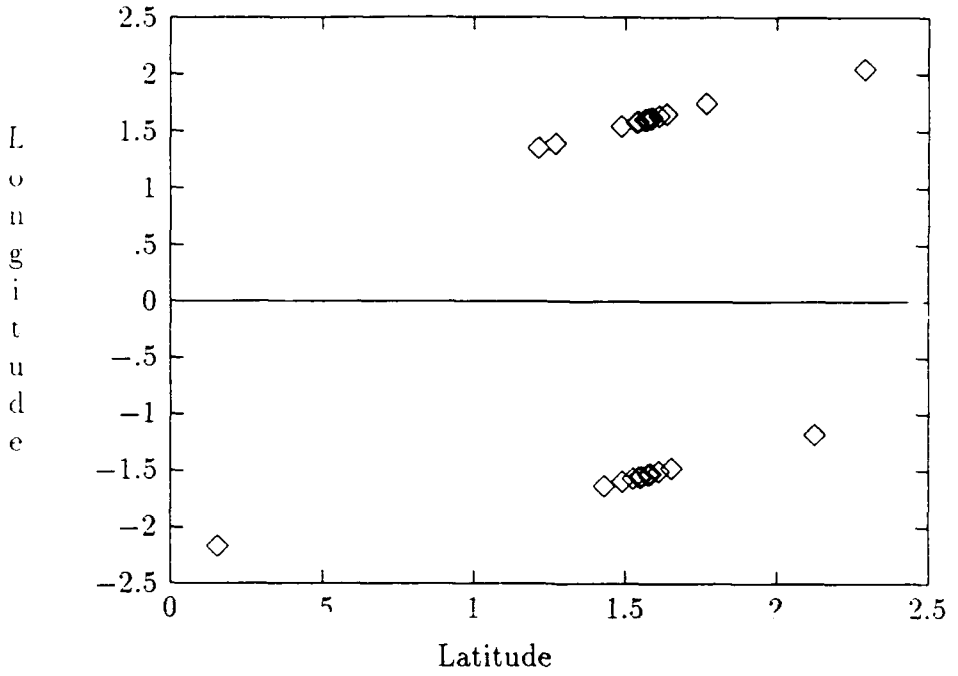


Figure 4.4. Exp 1/1 - Dihedral (Path 1 to 6) Poincare Plot

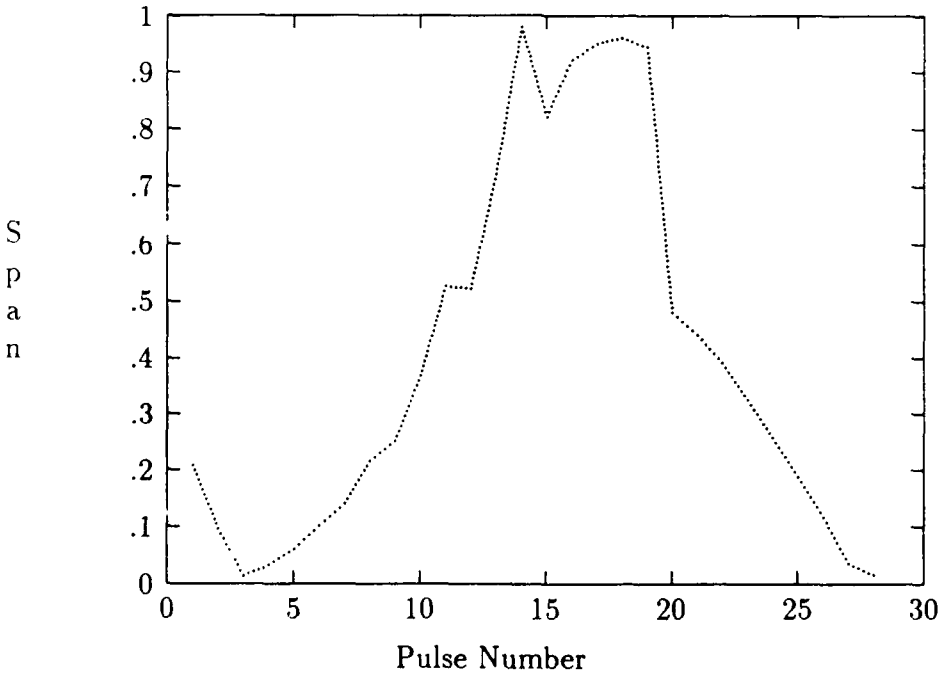


Figure 4.5. Exp 1/1 - Dihedral (Path 1 to 6) Span Plot

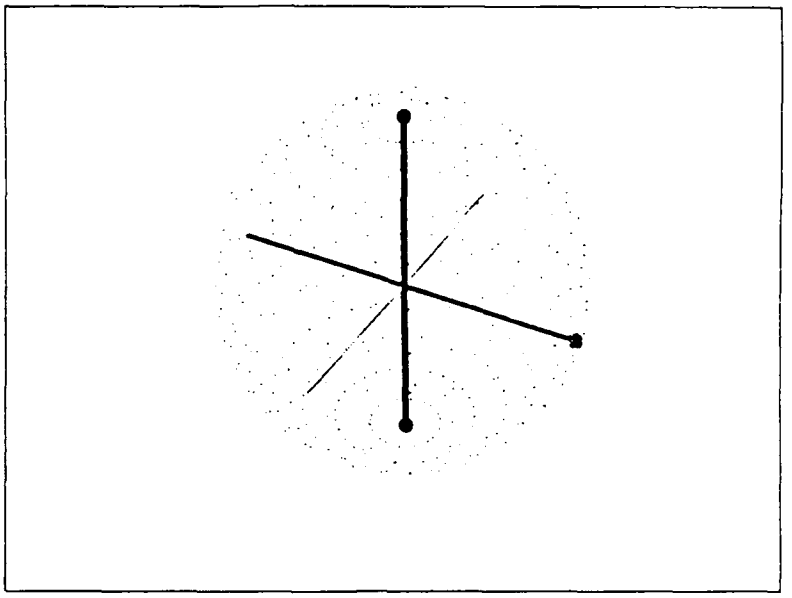


Figure 4.6. Exp 1/2 - Flatplate (Path 8 to 3) Poincare Sphere

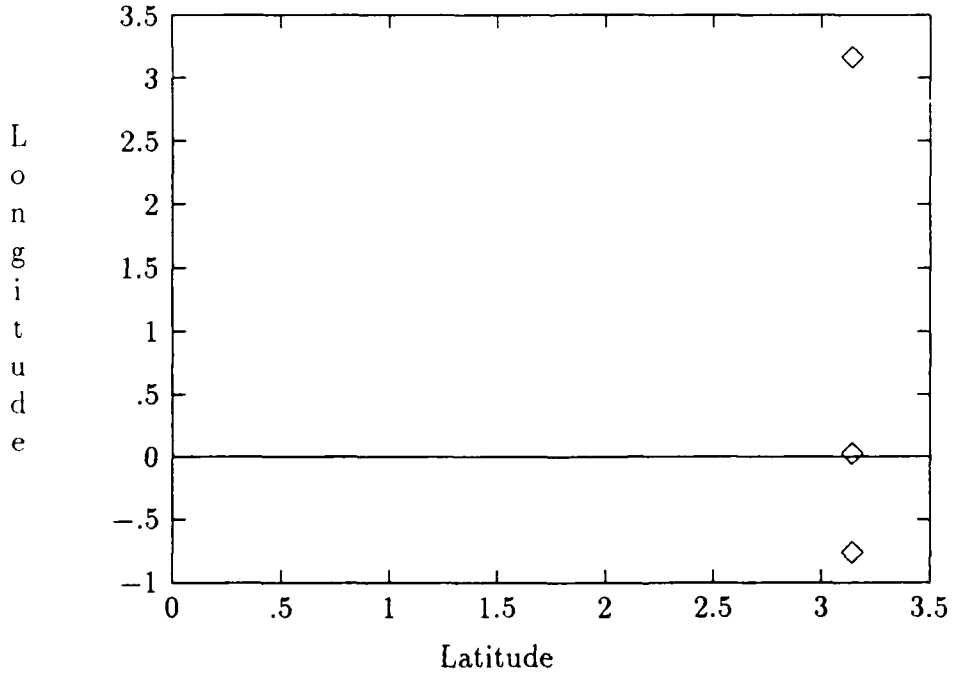


Figure 4.7. Exp 1/2 - Flatplate (Path 8 to 3) Poincare Plot

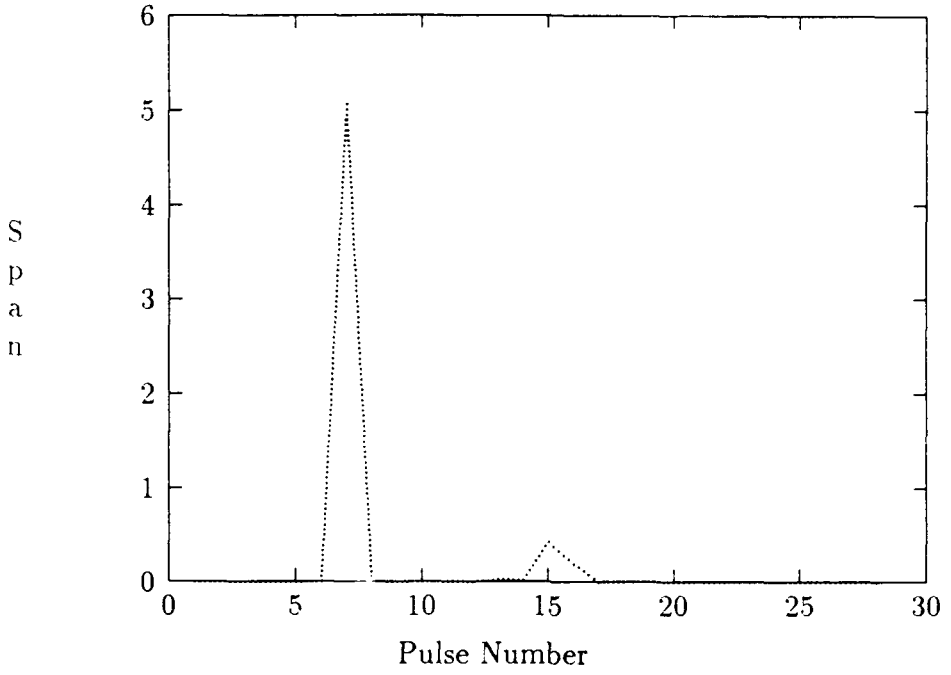


Figure 4.8 Exp 1/2 - Flatplate (Path 8 to 3) Span Plot

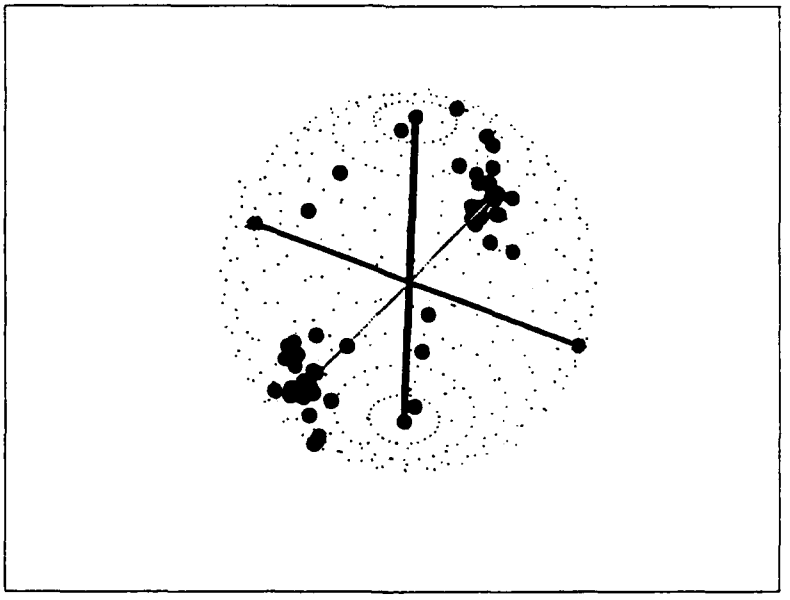


Figure 4.9. Exp 1/3 - Combination (Path A to B) Poincare Sphere

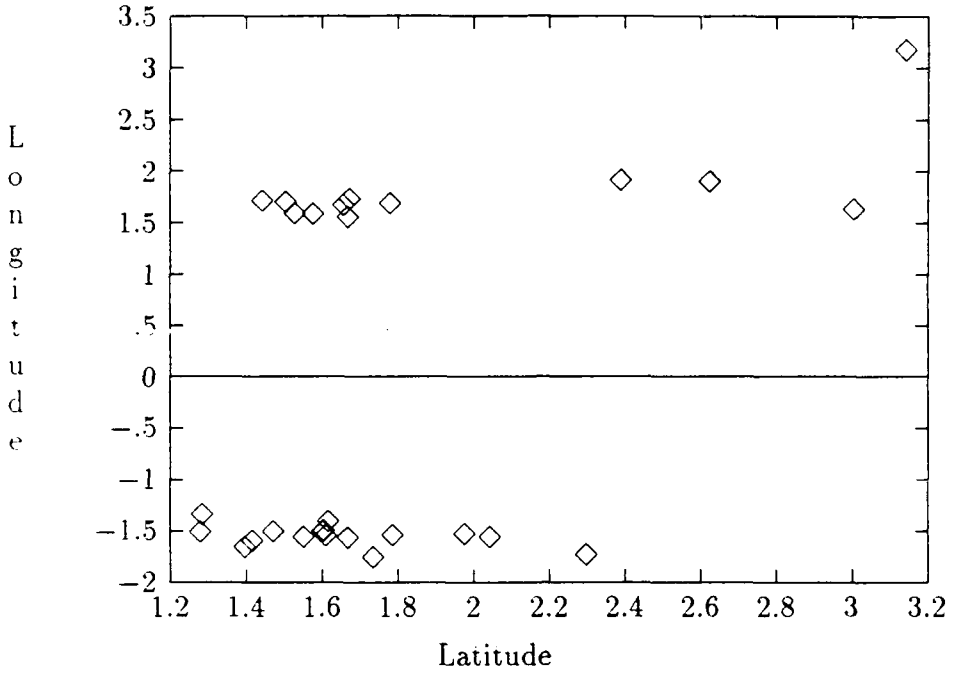


Figure 4.10. Exp 1/3 - Combination (Path A to B) Poincare Plot

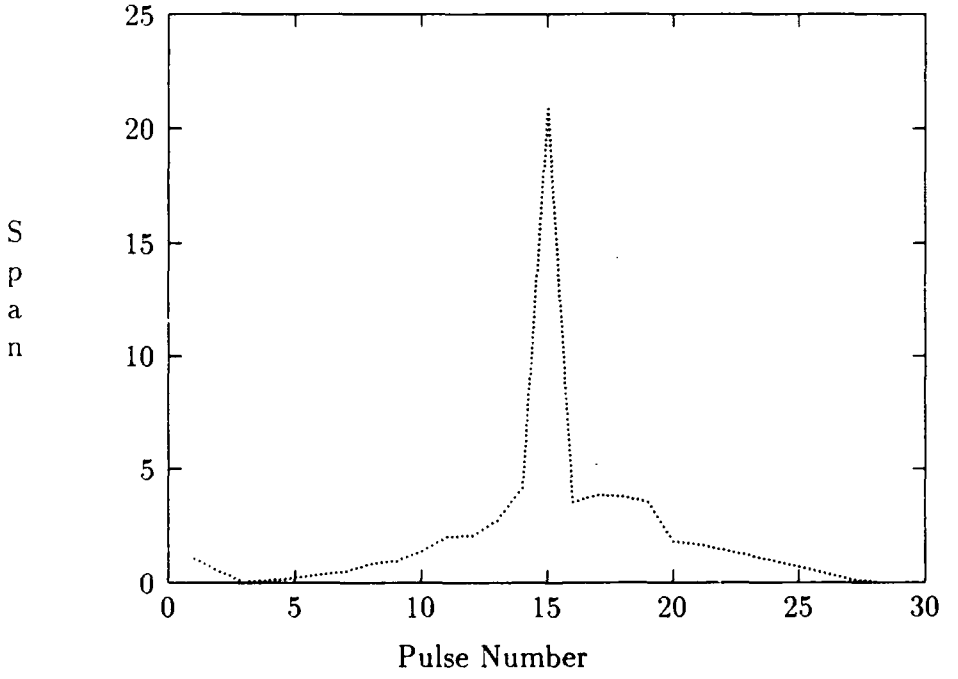


Figure 4.11. Exp 1/3 - Combination (Path A to B) Span Plot

Examination of the of the outputs from experiment one showed that different canonicals resulted in clearly different Poincare sphere plots. Moreover, the cell's response was found to be constant, regardless of the canonical's position within the cell.

4.4.2 Experimental Block Two. Experimental block two consisted of three sub experiments which were designed to measure the interaction of one canonical on another canonical when both the canonicals are co-located within the same resolution cell. The three sub experiments are described below:

1. Exp2 Sub Experiment 1. In this experiment a dihedral was moved from corner 2 to corner 6 in five increments while at the same time a flatplate was moved from corner 8 to corner 4 in the same number of increments.
2. Exp2 Sub Experiment 2. This sub experiment was a virtual repeat of the above sub experiment except that the canonical paths were reversed i.e. the dihedral was moved from corner 8 to corner 4 while the flatplate was moved from corner 2 to corner 6.
3. Exp2 Sub Experiment 3. In this experiment one dihedral was moved from corner 2 to corner 6 while another dihedral was moved form corner 8 to corner 4.

Sample outputs from Experiment 2 are shown in Figures 4.12 to 4.20.

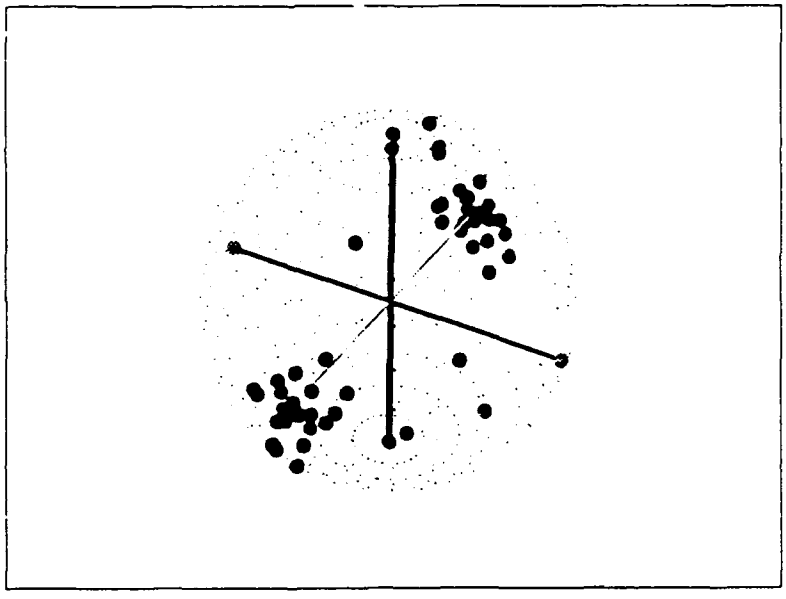


Figure 4.12. Exp. 2/1 Dihedral Path (2 to 6) Flatplate Path (8 to 3) Poincare Sphere

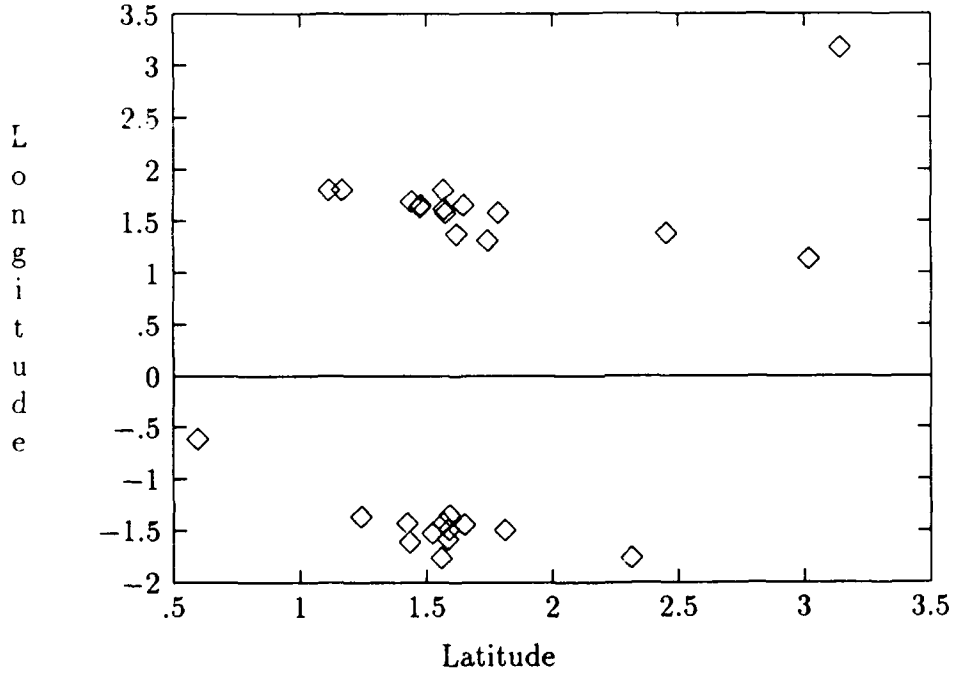


Figure 4.13. Exp. 2/1 Dihedral (Path 2 to 6) Flatplate (Path 8 to 3) Poincare Plot

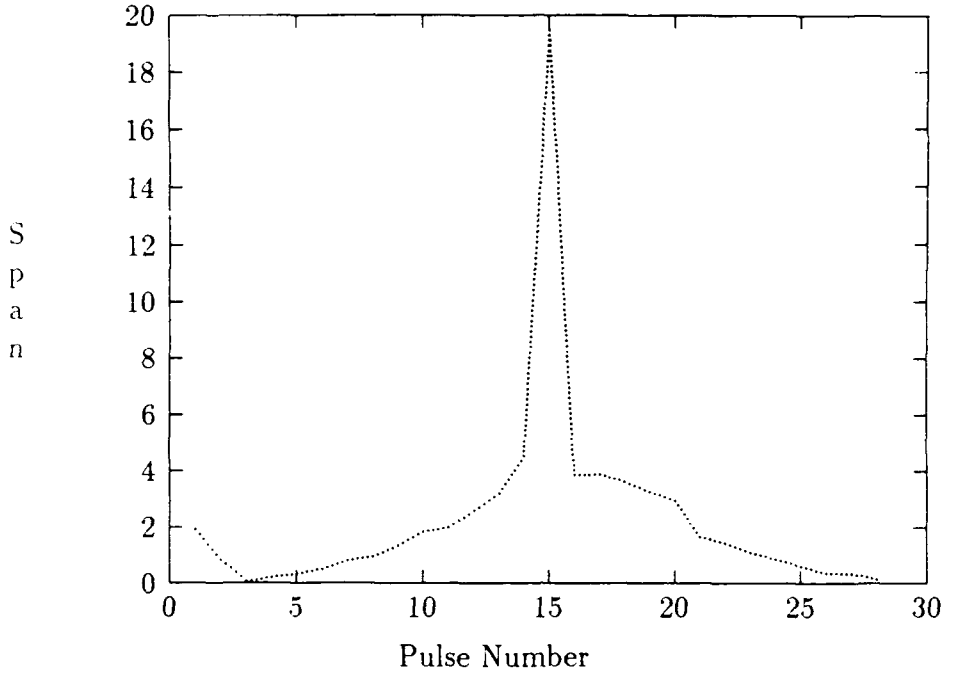


Figure 4.14. Exp. 2/1 Dihedral (Path 2 to 6) Flatplate (Path 8 to 3) Span Plot

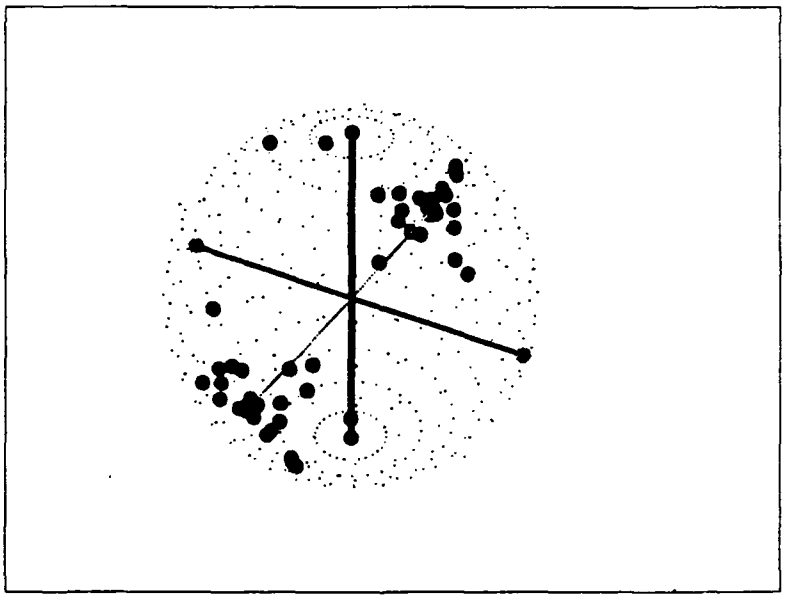


Figure 4.15. Exp. 2/2 Dihedral (Path 8 to 4) Flatplate Path(2 to 6) Poincare Sphere

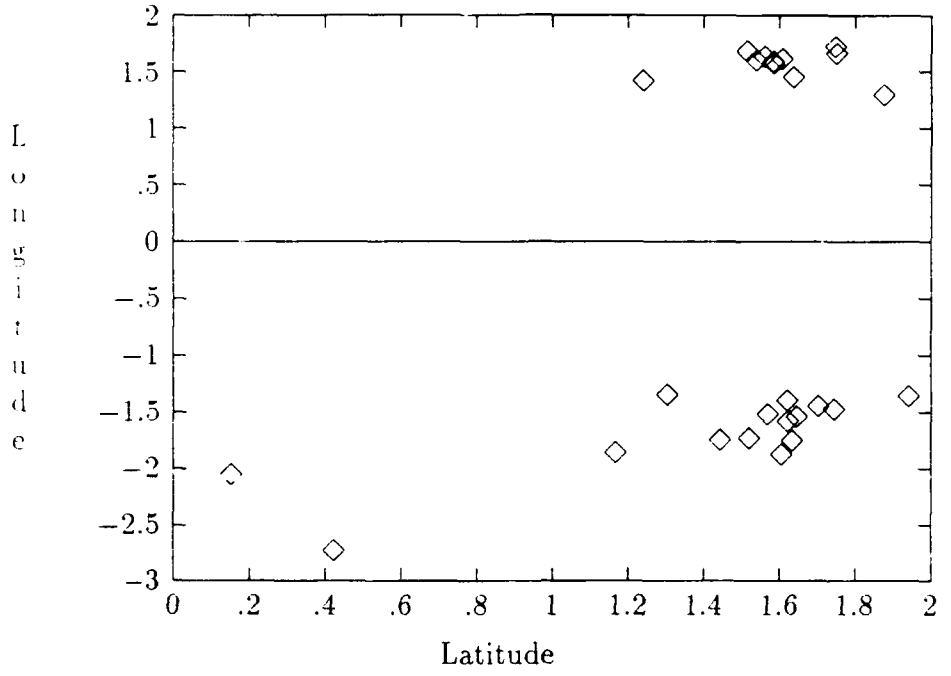


Figure 4.16. Exp. 2/2 Dihedral (Path 8 to 4) Flatplate (Path 2 to 6) Poincare Plot

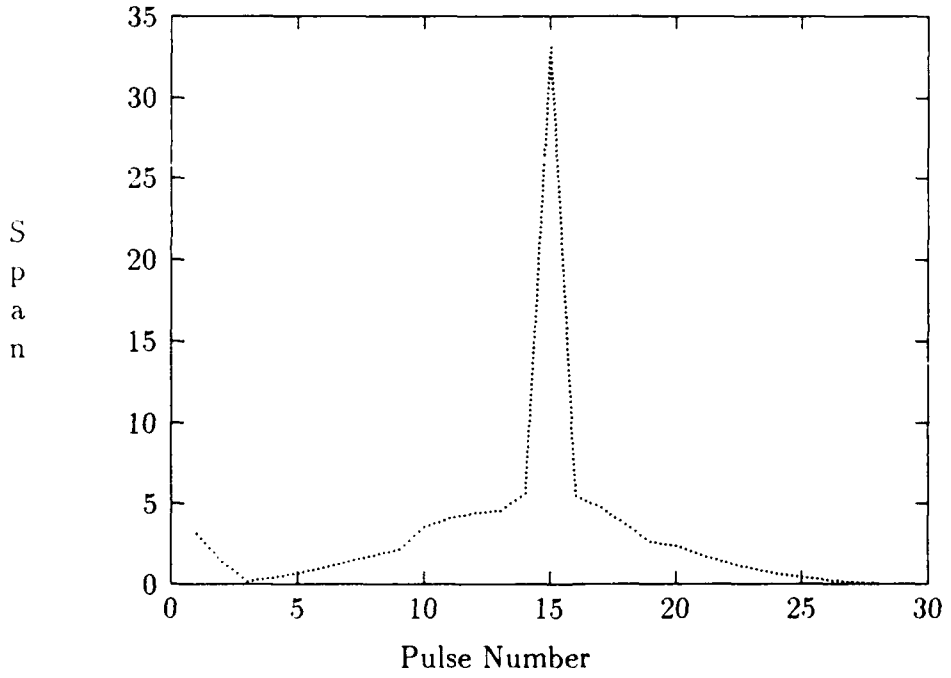


Figure 4.17. Exp. 2/2 Dihedral (Path 8 to 4) Flatplate (Path 2 to 6) Span Plot

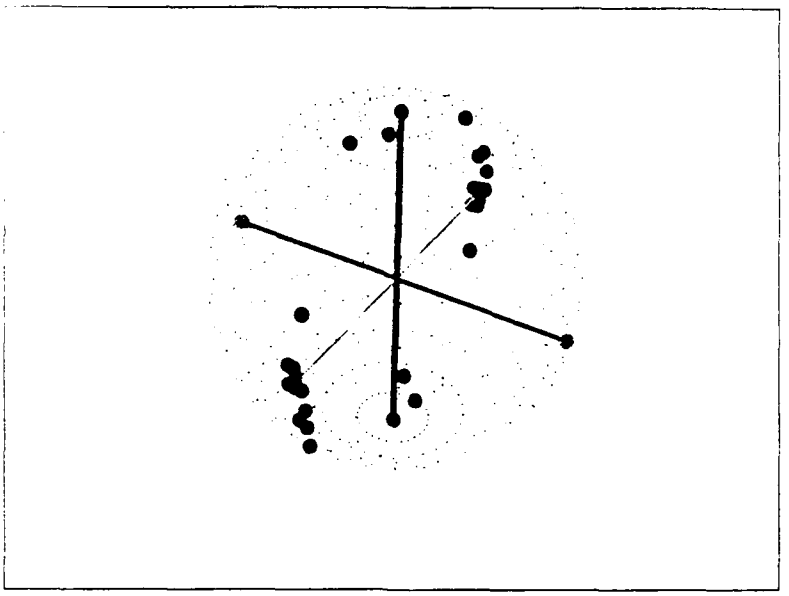


Figure 4.18. Exp. 2/3 - Dihedral (Path 2 to 6) Dihedral(Path 8 to 4) Poincare Sphere

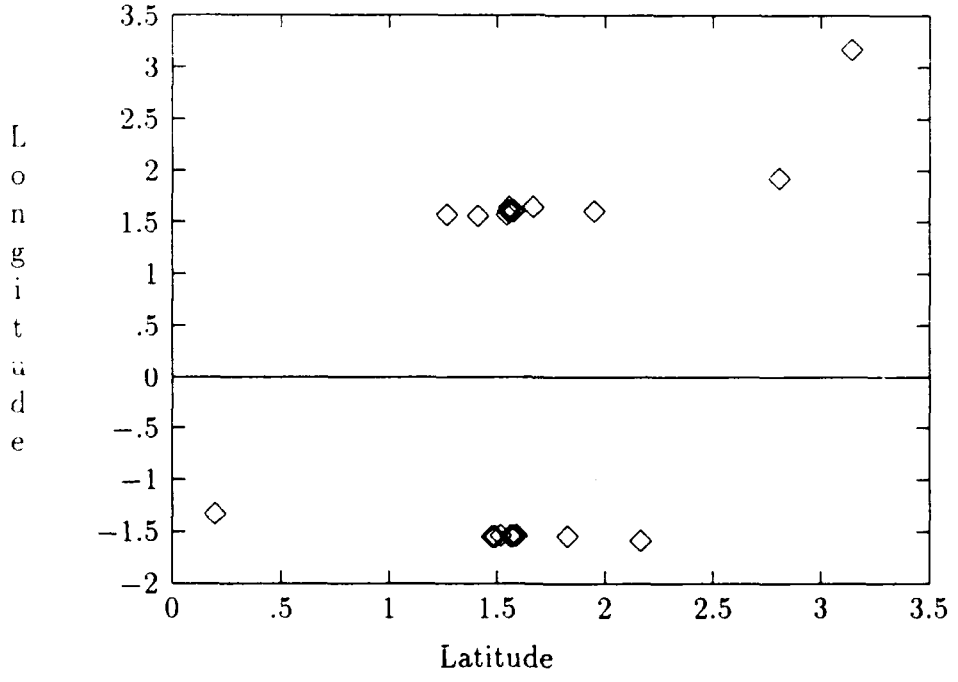


Figure 4.19. Exp. 2/3 - Dihedral (Path 2 to 6) Dihedral(Path 8 to 4) Poincare Plot

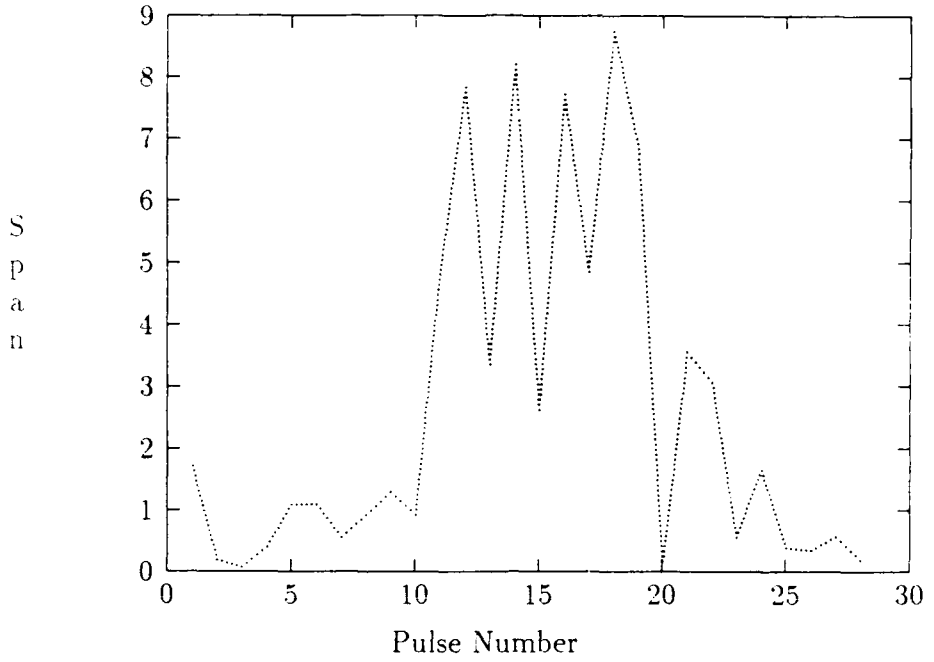


Figure 4.20. Exp. 2/3 - Dihedral (Path 2 to 6) Dihedral(Path 8 to 4) Span Plot

Analysis of the output from experiment 2 revealed that the SarTool^{tm} algorithms do not make any provision for interaction between the two canonicals. This was evident in that the cell's response at the start positions of each sequence was identical to the cell's response at each subsequent interval. This is an important revelation, the impact of which will be discussed in a later section of this thesis. Also of interest in experiment block 2 were the Poincare patterns which resulted from combining two canonicals in a single resolution cell. These patterns showed clearly discernible traits of each primitive's co-polarization signature. This result influenced the conduct of future experiments and is further detailed in the following paragraphs.

Table 4.3. Experiment Block Three: Dihedral and Flatplate Sizes

Dihedral Size (cm)	Flatplate Size (cm)	Exp No.
35.776	37.888	3a
53.664	37.888	3b
71.552	37.888	3c
89.440	37.888	3d
8.944	37.888	3e
5.963	37.888	3f
4.472	37.888	3g
3.578	37.88	3h

4.4.3 Experimental Block Three. This section of the experiment was devoted to determining the effect of changing relative canonical sizes on the Poincare sphere output. The experimental procedure involved holding the size of the flatplate constant whilst first increasing and then decreasing the size of the dihedral. The actual sizes used during this experiment are detailed in Table 4.4.3.

The outputs from experiments 3a, 3e and 3h are shown in Figures 4.21 to 4.29 respectively.

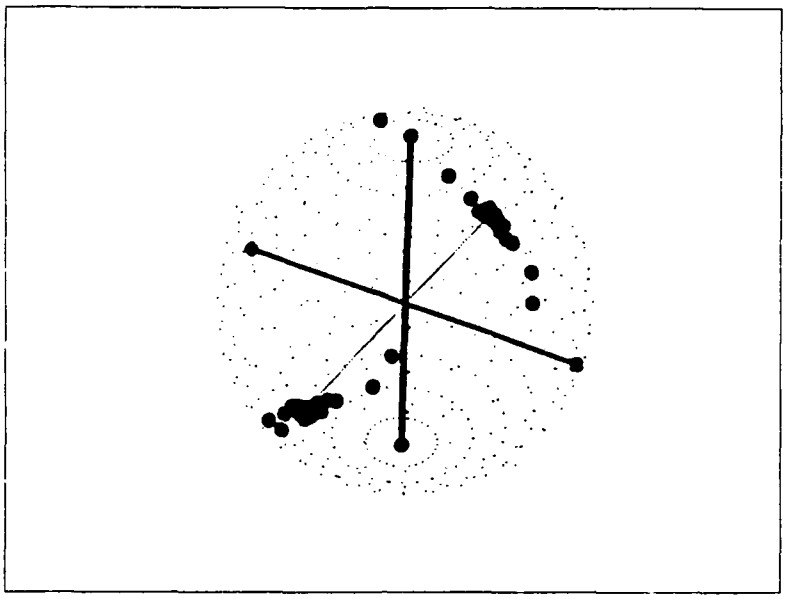


Figure 4.21. Exp. 3a Combination - Poincare Sphere

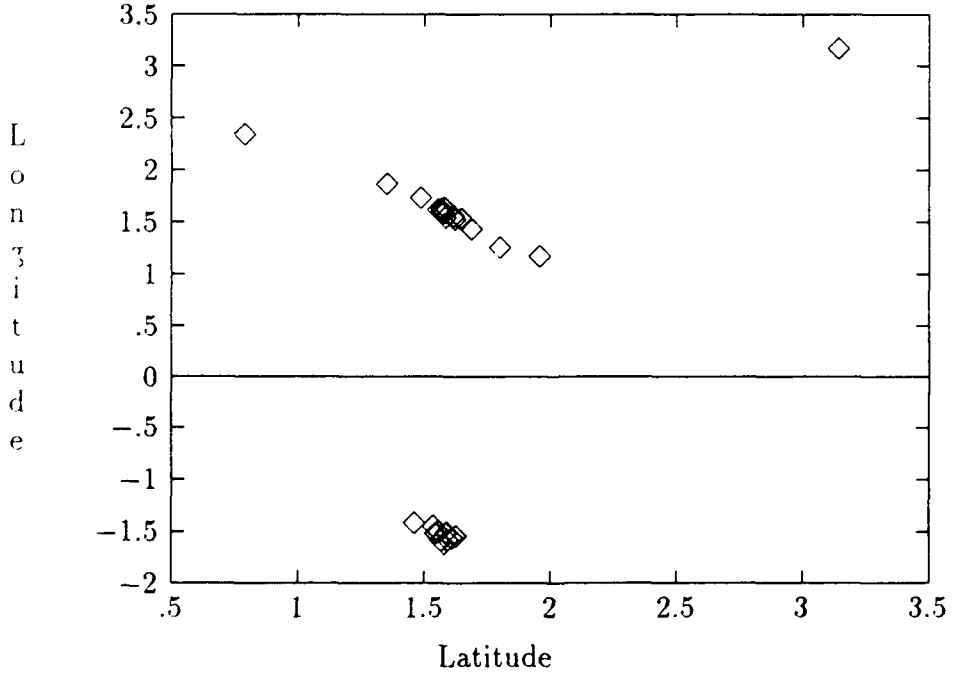


Figure 4.22. Exp. 3a Combination - Poincare Plot

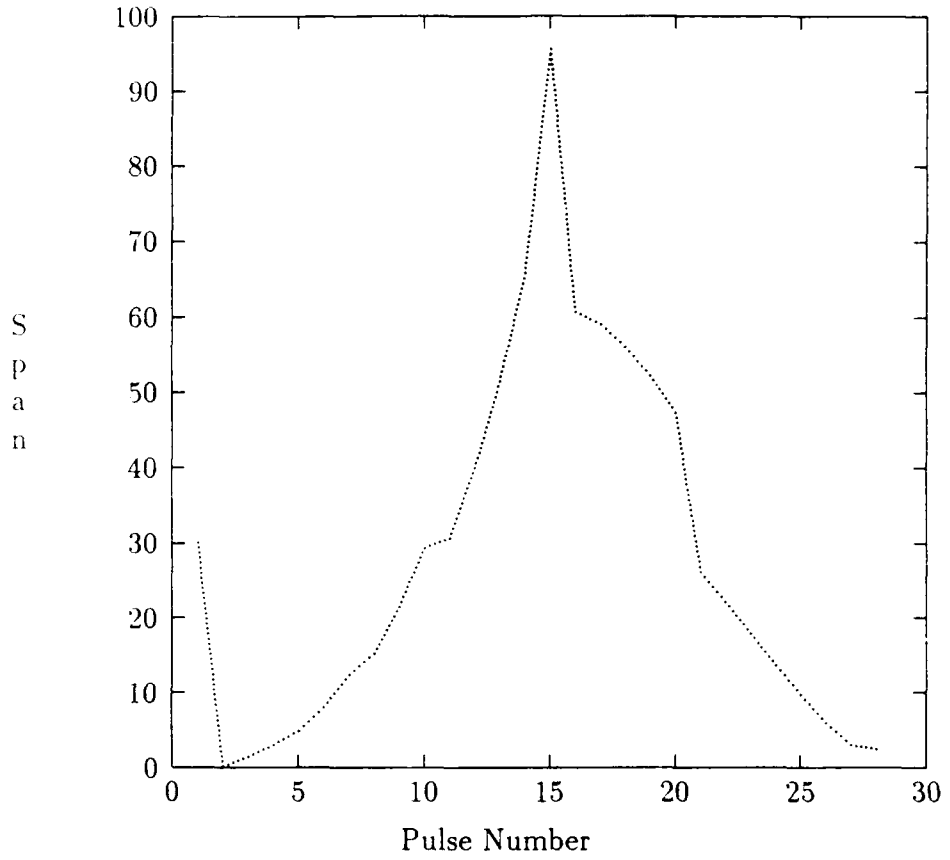


Figure 4.23. Exp. 3a Combination - Span Plot

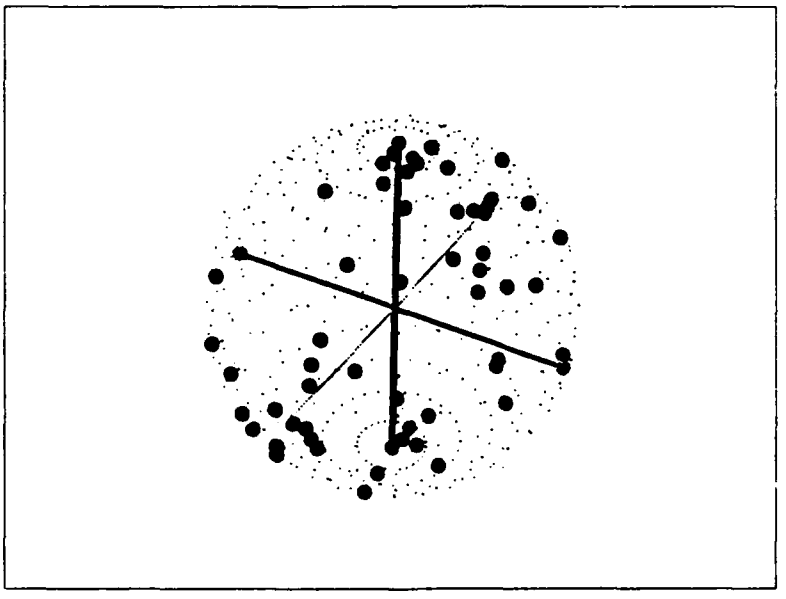


Figure 4.24. Exp. 3e Combination - Poincare Sphere

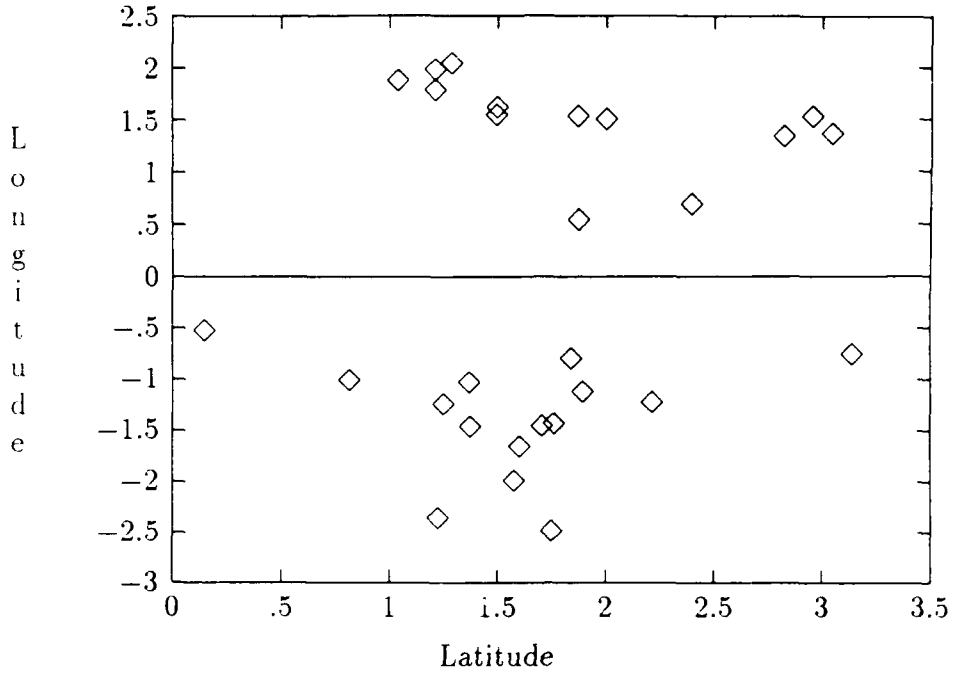


Figure 4.25. Exp. 3e Combination - Poincare Plot

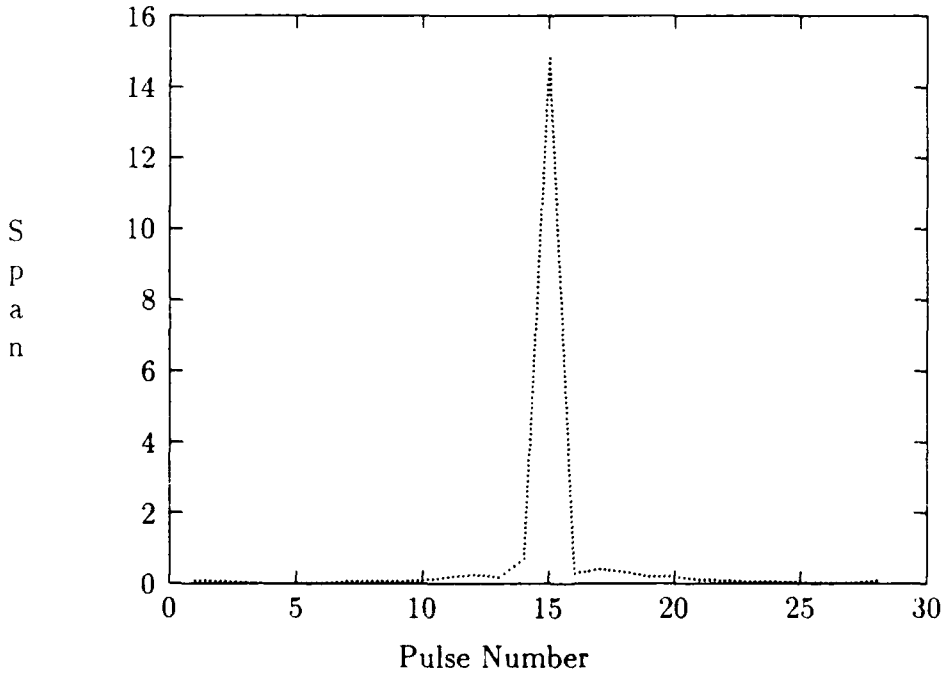


Figure 4.26. Exp. 3e Combination - Span Plot

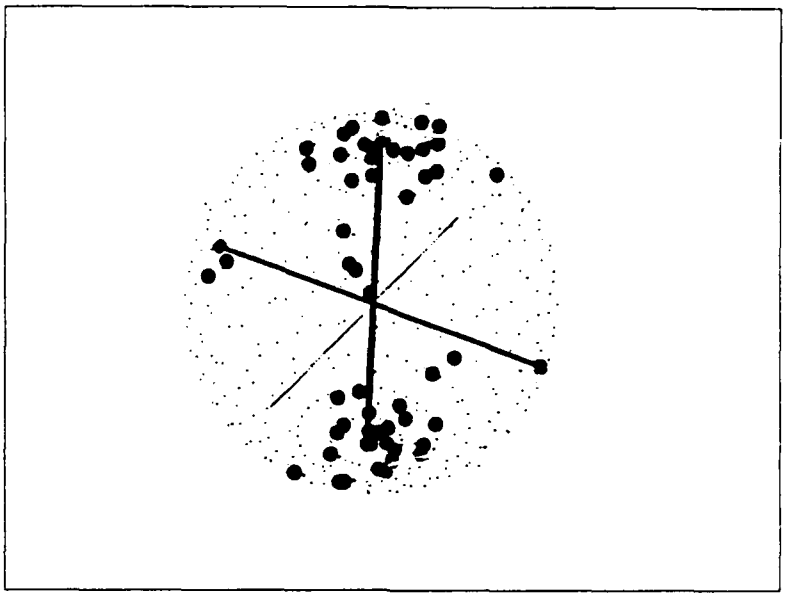


Figure 4.27. Experiment 3h - Poincare Sphere

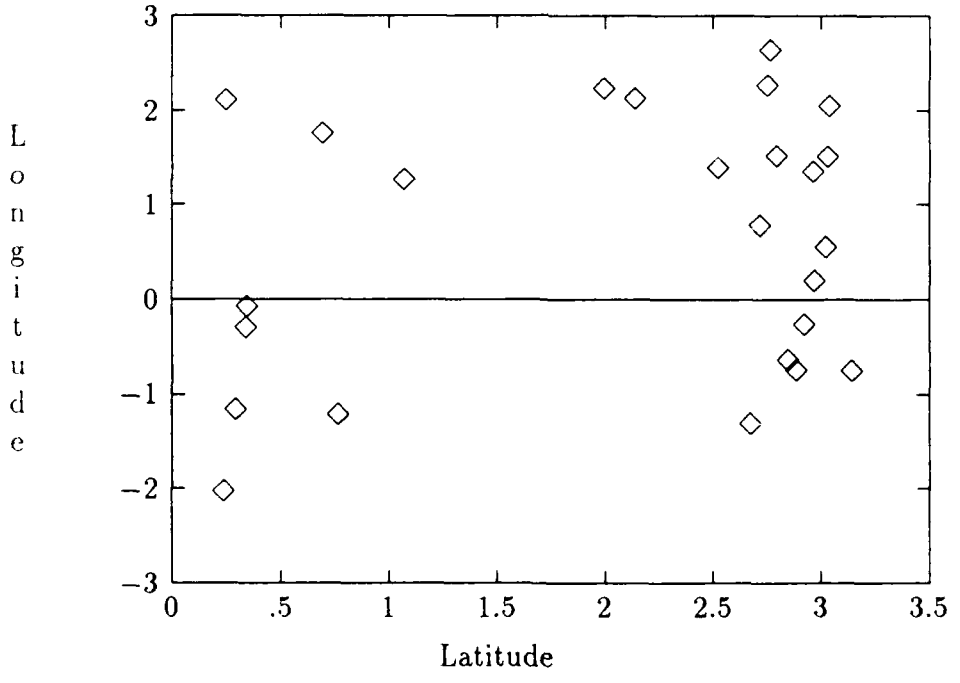


Figure 4.28. Exp. 3h Combination - Poincare Plot

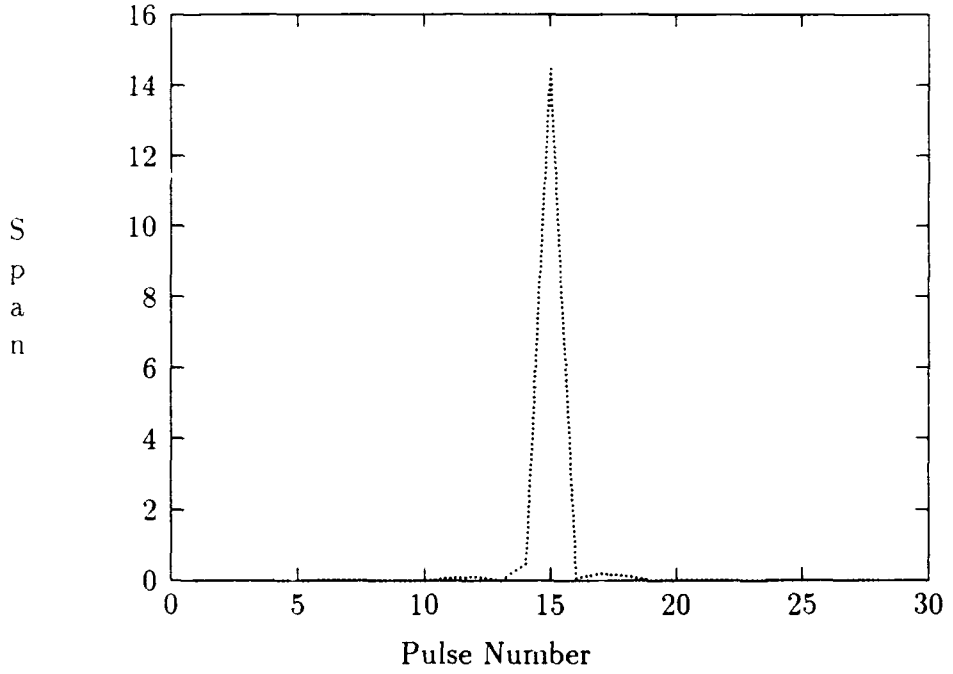


Figure 4.29. Exp. 3h Combination - Span Plot

The above results proved to be of significant interest. The pattern in experiment 3a was consistent with previous results in that it clearly showed the dominance of the dihedral co-polarization nulls over those of the flat plate. However, in experiments 3e and 3h the flatplate is assuming dominance due to its increased size over the dihedral (see Table 4.4.3). Accordingly, there is a marked movement of the co polarization nulls towards the "poles" of the sphere. A point of particular interest revealed by experiment block three was that this pattern, although different for various size combinations, appeared to have a consistent, predictable trend. This trend is evidenced by noting the similarities between the Poincare sphere results of experiment 3e and experiment 3h. The span of the co-polarization nulls was also of interest in that it showed a clear distinction between when the dihedral was the dominant factor and when the flat plate assumed dominance. The results of experiment 3 prompted an investigation into the possible role that perceptron networks could play in performing identification of cell contents via an examination of the co-polarization null patterns. Accordingly, in order to investigate this possibility further one final batch of experiments, Experimental block four, were performed. These experiments, numbered 6 through to 48 and 70 through 105 are described in the following section.

4.4.4 Experimental Block Four. The primary purpose of this block of experiments was to collect sufficient data on dihedrals and the combination of dihedrals and flatplates so that tests could be conducted using perceptron networks. Accordingly,

experiments 6 through 49 involved varying size dihedrals centered within the resolution cell while experiments 70 through 105 contained various sized combinations of flatplates and dihedrals. The exact configuration of these experiments is shown in Table 4.4.4. Note that the dihedral size started at 5cm square in experiment 6 and was incremented by 5cm for each subsequent experiment such that in experiment 45 the dihedral was 200 cm square. The sizes of the dihedrals in experiments 46 through 48 are as shown.

4.5 Discussion of Poincare Plots

Of note in each of the above experiment blocks was the "telling" characteristics of the Poincare sphere plots. Although theory predicts distinctive plots from isolated canonicals the level of information retrievable from the sphere plots, when combinations of canonicals were involved, was not expected. This latent potential for information extraction is evident in most of the discussed plots i.e. even when one canonical was physically dominant it was unable to completly mask the polarimetric effect of the other canonical. Specific examples of this effect can be seen by examining the Poincare plots of experiment 3e and 3h. In both these experiments the flat plate is physically the dominant canonical and yet the Poincare plots show co-polarization nulls which clearly indicate the prescence of other than a single flat plate. Similarly in incidences where the dihedral is the physically dominant canonical, for example in experiment 86, the distribution of its co-polarization nulls within the Poincare plot suggest that another canonical may be present. In addition to

Table 4.4. Experiment Block Four: Dihedrals and Combinations

Dihedral Size (cm)	Flatplate Size (cm)	Exp No.
5 - 200	Not Applicable	6 - 45
27	Not Applicable	46
33	Not Applicable	47
64	Not Applicable	64
20	20	70
20	40	71
20	60	72
20	80	73
20	100	74
40	20	75
40	40	76
40	60	77
40	80	78
40	100	79
60	20	80
60	40	81
60	60	82
60	80	83
60	100	84
80	20	86
80	40	87
80	60	88
80	80	89
80	100	90
100	20	91
100	40	92
100	60	93
100	80	94
120	10	95
120	20	96
120	40	97
120	60	98
120	80	99
120	100	100
160	20	101
160	40	102
160	60	103
160	80	104
160	200	105

the examples shown in this chapter there are several experiment outputs contained in Appendix C all of which show the same type of effects discussed above. These somewhat encouraging observations led to a preliminary analysis of the Poincare plots using perceptron networks.

4.6 Perceptron Network Analysis

The apparent trend of the above results, in addition to those presented in Appendix C, led to a series of experiments regarding the applicability of perceptron networks to polarimetric recognition tasks. These investigations were not intended to be exhaustive but were designed to yield an initial assessment of the viability of employing A.I. in polarimetric SAR radars. The perceptron experiments were divided into two primary blocks: those involving a single layer perceptron network and those involving a multi-level perceptron network. The results of these experiments are contained in the following paragraphs.

4.6.1 Single Level Perceptron Network. The Mathematica^{tm} output was formatted as latitude and longitude vectors and then used as input to the single level perceptron network which appears in Appendix B. The network was written such that it could either train and test on the same set of data or train on one set of data and test on another set of data. A random number generator (tied to the computer clock) was used to randomize the initial perceptron weights and the selection of train and test vectors. The results gained using this network are shown in Table 4.6.1.

Table 4.5. Single Level Perceptron Classification Results

Experiment Number	Vectors Used	Percentage Right
1	1d1f	100
2	2d2f	100
3	3d3c	52
4	1d3c	48
5	1d4c	54

Note that the vector nomenclature xkxk decodes as shown below:

1. x. x equals the experiment block number as discussed above.

2. k. k signifies the type of canonical responsible for the input vector: f = flatplate, d = dihedral and c = combination flatplate and dihedral.

The results of experiments 1 and 2 above make intuitive sense in that the data representing a dihedral and a flatplate is clearly separable (refer to example outputs in previous section of this chapter). Accordingly, the perceptron needed very little exposure, for example 10 vectors of each class, before it could attain a 100% classification accuracy.

The results of experiments 3,4 and 5 above emphasize the degree of overlap which exists between a dihedral and a flatplate/dihedral combination. This overlap is evident in the output examples contained in this chapter. The network in experiments 3,4 and 5 was trained for 2000 iterations on an equal number of dihedral and combination vectors after which it was tested on 500 vectors. The poor performance

was not unexpected and indicates the inability of a single layer network to separate "single hit" complex data.

The above experiments suggest that, notwithstanding the obvious limitations of single layer perceptrons, they may be suited to performing the initial evaluation of cell contents. Furthermore a single layer network is clearly capable of providing a "count" of single and double bounce returns from a particular cell of interest. These counts could then be used to estimate the contents of the cell.

Although the above single level "count" procedure may have applicability in the early stages of decision making it does not harness the sequence of patterns evident in the Poincare sphere plots. These patterns can only be harnessed by treating the whole sequence, not just individual returns, as an input to a perceptron network. Accordingly, the additional experiments of block four were run and used as inputs to G.Tarr's multi-level perceptron network.

4.6.2 Multi Level Perceptron Network. The output of all the dihedrals and dihedral/flatplate combinations were converted into a suitable form for input into Tarr's network by the "convert" program shown in Appendix B . A total of 93 vectors were collected, 46 of these vectors were used to train the network while the remaining 47 were used as previously "unseen" testing vectors. The actual network configuration used consisted of 56 input nodes, two output nodes and two hidden layers. The first hidden layer had 20 nodes whilst the second layer consisted of 10 nodes. A total of 30 separate runs were made with the initial weights being

randomly initialized and the order of vector presentation being different for each individual network run. The classification results of these runs are shown in Figures 4.6.2 and 4.6.2. The distinction between "good" and "right" is that in the former case the conditional probability of correctly classifying the cell contents is between .5 and .9 while to be gauged as "right" the conditional probability must be equal to or greater than .9. Figure 4.6.2 shows the decrease in recognition error as the number of training iterations is increased. Table 4.6 shows the conditional probabilities each of the decision outputs. As in the previous discussion the letter d represents a dihedral while the letter c represents the combination of a dihedral and a flatplate.

The results for the multi layer perceptron network are of particular interest due to their sharp training rise time and the associated classification performance. This indicates that the network finds the two patterns readily separable. Of note is that the above performance was made only on polarimetric data i.e. consistent with the scope of this project the span data was not used as a classification aid. The inclusion of span data would almost certainly enhance the net's classification performance. Also, on a cautionary note, it should be emphasized that all of the experiments conducted during this thesis were conducted in a "noise free" environment. Furthermore, as mentioned earlier, SarTool^{tm} does not cater for inter canonical interference.

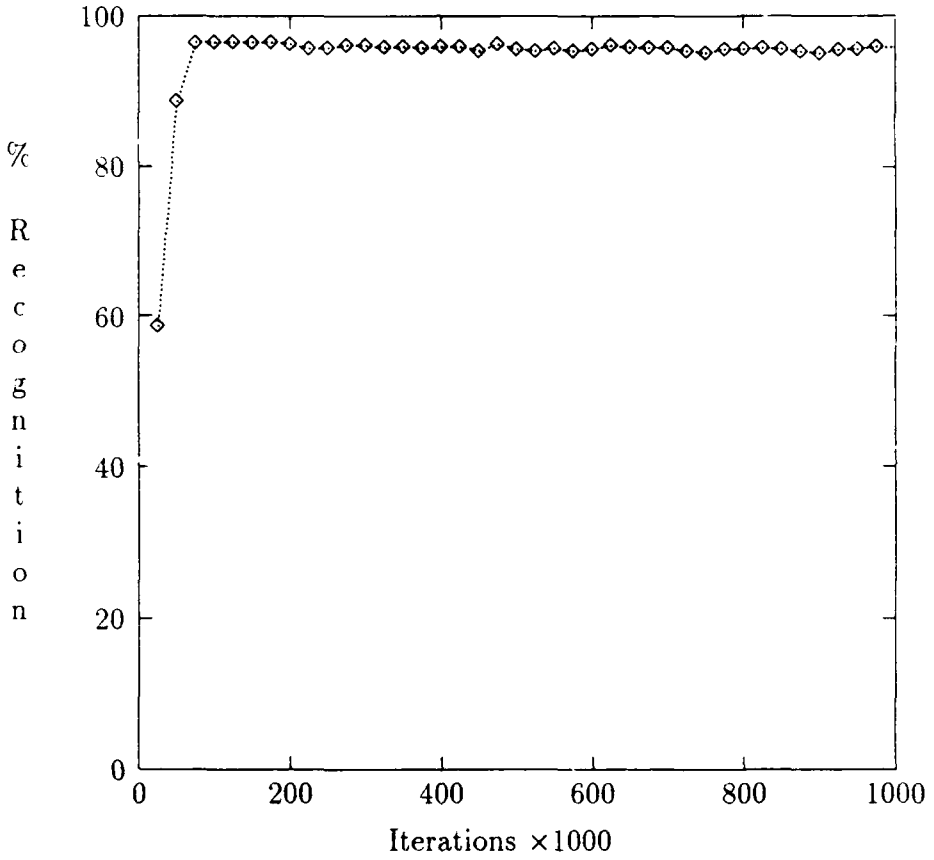


Figure 4.30. Percent Classified "Good"

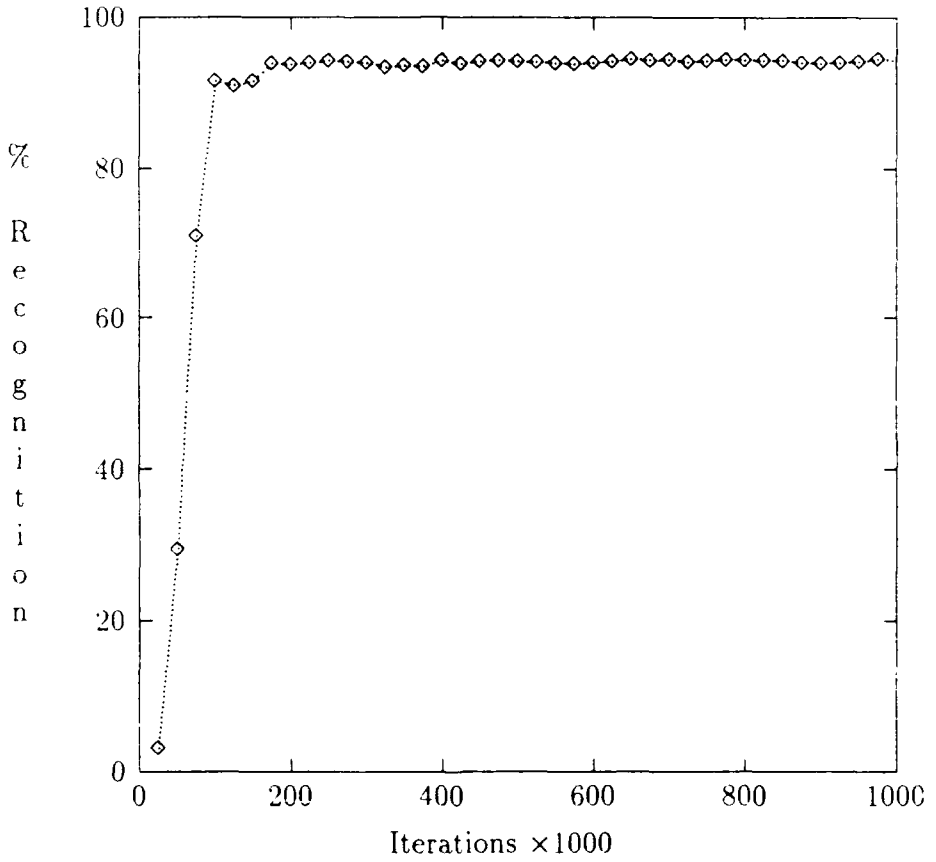


Figure 4.31. Percent Classified "Right"

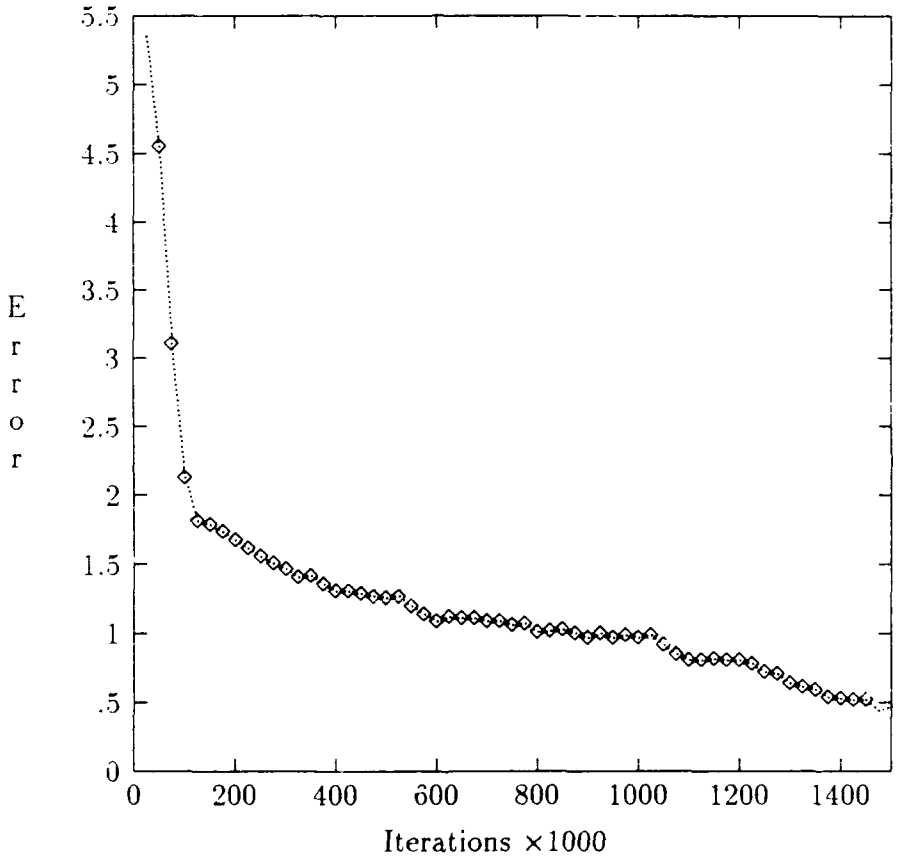


Figure 4.32. Recognition Errors

Table 4.6. Conditional Probability Table

	PDF(C/C)	PDF(D/C)	PDF(C/D)	P(D/D)	P(good)
net1	0.958	0.042	0.000	1.000	0.978
net2	0.917	0.083	0.000	1.000	0.957
net3	0.917	0.083	0.000	1.000	0.957
net4	0.875	0.125	0.000	1.000	0.935
net5	0.917	0.083	0.000	1.000	0.957
net6	0.875	0.125	0.000	1.000	0.935
net7	0.958	0.042	0.000	1.000	0.978
net8	0.917	0.083	0.000	1.000	0.957
net9	0.917	0.083	0.000	1.000	0.957
net10	0.958	0.042	0.000	1.000	0.978
net11	0.917	0.083	0.000	1.000	0.957
net12	0.917	0.083	0.000	1.000	0.957
net13	0.917	0.083	0.000	1.000	0.957
net14	0.875	0.125	0.000	1.000	0.935
net15	0.917	0.083	0.000	1.000	0.957
net16	0.917	0.083	0.000	1.000	0.957
net17	0.917	0.083	0.000	1.000	0.957
net18	0.875	0.125	0.000	1.000	0.935
net19	0.917	0.083	0.000	1.000	0.957
net20	0.875	0.125	0.000	1.000	0.935
net21	0.875	0.125	0.000	1.000	0.935
net22	0.875	0.125	0.000	1.000	0.935
net23	0.875	0.125	0.000	1.000	0.935
net24	0.917	0.083	0.000	1.000	0.957
net25	0.917	0.083	0.000	1.000	0.957
net26	0.917	0.083	0.000	1.000	0.957
net27	0.958	0.042	0.000	1.000	0.978
net28	0.917	0.083	0.000	1.000	0.957
net29	0.958	0.042	0.000	1.000	0.978
net30	0.875	0.125	0.000	1.000	0.935
Average	0.911	0.089	0.000	1.000	0.954
Variance	0.001	0.001	NA	NA	0.000
St Devn	0.028	0.028	NA	NA	0.015

4.7 Summary

Notwithstanding the provisos of the above paragraph, the results discussed during this chapter provide some useful information on the polarimetric behavior within a resolution cell. Moreover, although SarTool^{tm} has certain limitations, it can (when combined with other software) provide a reasonable medium for the investigation of radar polarimetrics. The results gained during this thesis support this contention and seem to indicate that classification of intra cell contents is possible using only polarimetric data. Finally, the use of single and multi layer perceptron networks appears to have much potential in this classification process.

V. Conclusions

5.1 Introduction

This thesis effort was directed at an examination of the polarimetric behavior of canonicals within a resolution cell. An integral and necessary part of this venture involved a limited evaluation of SarTool^{tm} and the development of several software modules such that SarTool^{tm} could be readily employed in future polarimetric studies. The conclusions reached during the course of these endeavors and suggestions for future research efforts are discussed in the following paragraphs.

5.2 Evaluation of SarTool^{tm}

Time constraints and the primary objectives of this thesis precluded an extensive evaluation of SarTool^{tm}. Nevertheless, an early investigation of SarTool^{tm}, and subsequent discoveries made during the course of this thesis allow the following main conclusions to be made:

1. Documentation. The level and content of SarTool^{tm} documentation appears to be lagging the development of the software. An example of this observation is the apparent confusion which exists over the documented, versus coded, reflection algorithms.

2. Configuration. Early problems were evident in SarTool's^{tm} "acknowledgement" of small canonicals within resolution cells. Although this problem has appar-

ently been remedied there seemed to be no structured way in which the user of the code could access certain software configuration details, for example Version listings and "patch" status.

3. Performance. The implementation of SarTool^{tm} is such that, except for a shadowing option, the algorithms do not cater for the interaction of canonicals which are co-located within the same resolution cell.

The above conclusions are not intended as criticisms of SarTool^{tm}. However, they are included for their potential worth in future preliminary stages of experiment design.

5.3 Developed Software

The software developed during this thesis has been exhaustively described within the body of this thesis and software listings appear in Appendix B. Although the mentioned ancillary software greatly enhanced the use of SarTool^{tm} there is potential for further improvements in the automation of result gathering. At present each SarTool^{tm} run (i.e. each movement of a canonical) must be individually processed by Mathematica^{tm} before being converted for use by a perceptron network (or other post processing device). A future worthwhile endeavour would be to batch process both the inputs and the outputs to Mathematica^{tm} such that each run of SarTool^{tm} would automatically produce a decision output.

5.4 *Recognition of Intra - Cell Canonicals*

The results gained during this thesis indicate that worthwhile information can be extracted by examination of the polarization contents of the Poincare sphere. A particularly promising observation was that the presence of a dihedral reflector can be detected by visual inspection of the Poincare sphere even when the dihedral is co-located with a flatplate of much larger dimensions. As the first in a series of studies on this topic this thesis was constrained to only examining the classic single and double bounce canonicals i.e. the flatplate and the dihedral. Notwithstanding this constraint the software that was developed throughout this thesis can be applied equally well to the investigation of any canonical within single or multiple resolution cells. Moreover, the investigation of canonicals where $S_{AB} and S_{BA}$ are not always equal to zero would facilitate incorporating the cross polarization nulls as well as the co-polarization nulls into the identification process.

A logical extension to the Poincare sphere part of this thesis would be to ultimately build targets based on as many as necessary of the 13 canonicals modelled within SarTool^{tm}. The resulting outputs could then be processed by some form of A.I. (perceptron networks are discussed in the next section) in an attempt to gain target recognition. As indicated during the previous chapter this thesis was conducted in a "noise free" environment and within certain practical limitations of SarTool^{tm}; accordingly, any extension to this effort should include validation of the theoretically predicted result within an anechoic chamber.

5.5 *Perceptron Networks in Polarimetrics*

The use of single and multi layer perceptron networks in the latter stages of this thesis effort proved to be of much interest. Accordingly, the following paragraphs will briefly summarize the findings made regarding these networks and will suggest areas of future potential coupling between perceptron networks and polarimetric studies.

5.5.1 Single Level Perceptron. The single perceptron network required little training before it was able to perform extremely well in the identification of single canonicals. However, even with extensive training, the same network was clearly incapable of correctly identifying a combination of canonicals. This inability arises from the methodology used by the single perceptron network to process the cell returns. The single perceptron examines each individual return from the cell and classifies that return as originating from either a single or double bounce canonical. This process eliminates the possibility of any true pattern recognition being conducted on the returns. Nevertheless, useful information regarding cell contents may be extracted by even a single layer perceptron based on relative counts of single and double bounce returns. The network contained in Appendix B would be ideal for this application as it can maintain an ongoing count of the processed data.

5.5.2 Multi Layer Perceptron. The preliminary results gained using a multi level perceptron network for decision making appear to be extremely promising. With little training the multi level network was able to gain very high levels of

correct identification (approximately 97%) using only co-polarization polarimetric data. Should cross polarization and span data be incorporated into the decision process it seems possible that worthwhile decision analysis could be obtained even if the library of choices was increased to cover multiple combinations of canonicals.

Ultimately a structured approach involving a combination of single and multi layer networks may prove to be computationally efficient way to perform analyzes of cell content. Initial decisions regarding the presence of single canonicals could be made very quickly by single layer networks while the multi layered networks could be used to process the more complex returns characteristic of combinations of canonicals. Implicit in the above discussion is that SAR processing for this type of analyzes would need to be based on "multiple looks" at each resolution cell rather than on the traditional "one hit" approach.

5.6 Summary

The aims of this thesis effort were met: the potential SarTool^{tm} package to examine polarimetric data was harnessed by ancillary software and multiple experiments examining cell polarimetrics were conducted. In addition an initial investigation on the applicability of perceptron networks to the polarimetric decision process was conducted. Logical extensions to this thesis include a theoretical and measured study of multiple combinations of canonicals. Finally, the incorporation of all polarimetric and span data into a specifically tailored perceptron network would seem

to be an extremely promising avenue of further research.

Appendix A. *Relevant Theory of SarTool^{tm}*

A.1 Introduction

SarTool^{tm} is a software package which simulates the behavior of a polarimetric (HH, HV, VH, VV) SAR thereby allowing the scattering behavior of several canonical shapes to be examined. This Appendix provides a brief overview of SarTool^{tm} as it applies to this thesis effort.

A.2 Background

SarTool^{tm} was developed for the USAF by The Analytic Sciences Corporation (TASC). The original contract number was F33615-87-C- 1404 while the performance of the system is described in a TASC Technical Information Memorandum (TIM) numbered 5417- 2B([26]). The reader should be aware that SarTool^{tm} is still subject to ongoing development; accordingly, updated versions of the software and the associated documentation are periodically released.

A.3 Overview of SarTool^{tm} Operation

SarTool^{tm} is intended to provide Electromagnetic Prediction of target signatures. As such SarTool^{tm} uses Physical Optics modelling to predict the interaction between certain canonical geometric shapes and a fully polarimetric SAR. The "library" of geometric shapes consists of thirteen canonicals including elliptical base

top hat reflectors, cone reflectors, ogive reflectors, cavity reflectors, complex cavity reflectors, elliptical plate reflectors, polygonal base top hat reflectors, cylinder reflectors, curved dihedral reflectors, ellipsoid reflectors, trihedral reflectors, dihedral reflectors and polygonal plates. The use of Solid Objects Modelling allows these shapes to represent most targets of interest. The "normal" output of SarTool^{tm} is a SAR image the shape and characteristics of which depend on the shapes illuminated by the radar.

The user "interacts" with the SarTool^{tm} software via a number of input files which are described in the following paragraphs.

A.3.1 SarTool^{tm}.dat file. The SarTool^{tm}.dat file is the "master" input file which essentially configures the relationship between the radar and the targets. Input parameters include radar characteristics, radar placement, target geometry/orientation and electromagnetic environmental considerations for example shadowing optio. Other options include range bins and target compensation options. An example of a SarTool^{tm}.dat file used during this thesis is shown in Table(A.3.1) overleaf.

A.3.2 Reflector Files. The reflector files are the means whereby the operator defines composition and placement of the radar target. As mentioned in Chapter 3 the reflector files are normally generated by the "EUCLID" software and as such are not readily amenable to manual construction. An example of the inputs to a

Table A.1. SarTool^{tm} Sensor and System Data (Sartool.dat)

'exp1'	TITLE OF SIMULATION RUN
'dihed'	SPECIFIES INPUT REFLECTOR FILE
'0'	X POSITION OF SENSOR
'0'	Y POSITION OF SENSOR
'1'	Z POSITION OF SENSOR
'90'	FLIGHT PATH ANGLE OF SENSOR
'0'	DIVE ANGLE OF SENSOR
'0'	AVERAGE SENSOR SPEED (SAR MODE)
'0'	X POSITION OF TARGET REL TO MAP CENTER
'0'	Y POSITION OF TARGET REL TO MAP CENTER
'0'	Z POSITION OF TARGET REL TO MAP CENTER
'0'	EULER ANGLE PHI FOR TGT FRAME ROTN
'0'	EULER ANGLE THETA FOR TGT FRAME ROTN
'0'	EULER ANGLE PSI FOR TGT FRAME ROTN
'9.4 GHZ'	TRANSMITTED FREQUENCY
'2.0'	ELEVATN 3db BEAMWIDTH (DEG)
'2.0'	AZIMUTH 3db BEAMWIDTH (DEG)
'.01'	ELEVATION BORESIGHT ANGLE (DEG)
'90.0'	AZIMUTH BORESIGHT ANGLE (DEG)
'2.0'	RANGE RESOLUTION (m)
'2.0'	AZIMUTH RESOLUTION (m)
'1'	NUMBER OF RANGE BINS
'1'	NUMBER OF AZIMUTH BINS
'200'	NUMBER OF PULSES TRANSMITTED
'1'	TARGET DATABASE RECALCULATION I/VAL
'f'	TERRAIN GENERATION OPTION (SAR)
'0'	AVERAGE TERRAIN RCS (dbsm)
't'	SAR SPOTLIGHT OPTION
't'	ISAR MODE OPERATION
'0'	ASPECT ANGLE INCREMENT FOR QKTOOL
'f'	MOTION COMPENSATION OPTION
'f'	REFLECTOR THINNING OPTION
'0'	THINNING THRESHOLD VALUE (dbsm)
't'	TARGET SHADOWING OPTION
't'	TERRAIN SHADOWING OPTION
'f'	REFLECTOR I/P REPORT OPTION
't'	REFLECTOR D/BASE REPORT OPTION
't'	RANGE PROFILE OUTPUT OPTION

Table A.2. Reflector File

'DH'	DIHEDRAL IDENTIFIER
'x'	X POSITION OF DIHED CENTER IN TGT FRAME
'y'	Y POSITION OF DIHED CENTER IN TGT FRAME
'z'	Z POSITION OF DIHED CENTER IN TGT FRAME
'phi'	TRANSFORMATION ANGLE FROM REFL TO TGT FRAME
'theta'	TRANSFORMATION ANGLE FROM REFL TO TGT FRAME
'psi'	TRANSFORMATION ANGLE FROM REFL TO TGT FRAME
'd1'	LENGTH OF DIHED IN CREASE DIMN OF REF FRAME
'd2'	LENGTH OF PLATE PERP TO CREASE DIMN
'sop'	SOLID OBJECTS PRIMITIVES INDICES

reflector data file is shown in Table A.2.

A.3.3 Image.Opts File. This file specifies several processing and imaging options used in SarTool^{tm}'s imaging algorithms. These options include weighting functions, scaling functions and the determination of dynamic ranges. Because the traditional imaging representation of SarTool^{tm} data was not used in this thesis the image.opts file will not be discussed further although the interested reader may find additional information in Reference([26].

A.3.4 SarTool^{tm} Output Files. There are two main output files associated with the above operator input files. These two files are briefly described below:

1. Range Profile Data Base (RPDB) file. This file contains in phase and quadrature phase data for the four polarimetric components. This file is created every time SarTool^{tm} is invoked with the associated data being stripped by the *db*

strip program which is contained in Appendix(B).

2. Statistics File. This file contains the history of the processed output. The statistics file did not play a prominent part in this thesis effort and will therefore not be discussed further in this Appendix.

A.3.5 Sensor Geometry Parameters. All of the input files necessitate some degree of translation between the ground and target frames; accordingly Figure(A.1) demonstrates the relationship between these two co-ordinate frames.

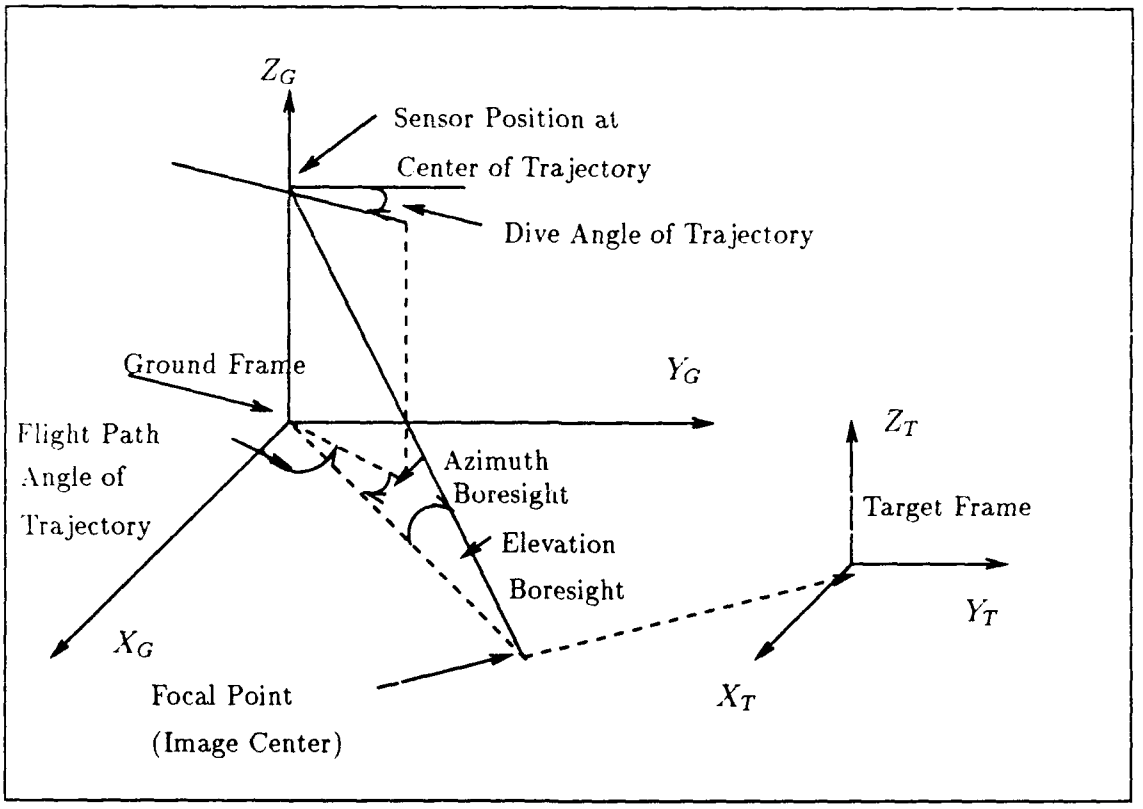


Figure A.1. SAR Data Sampling

[20]

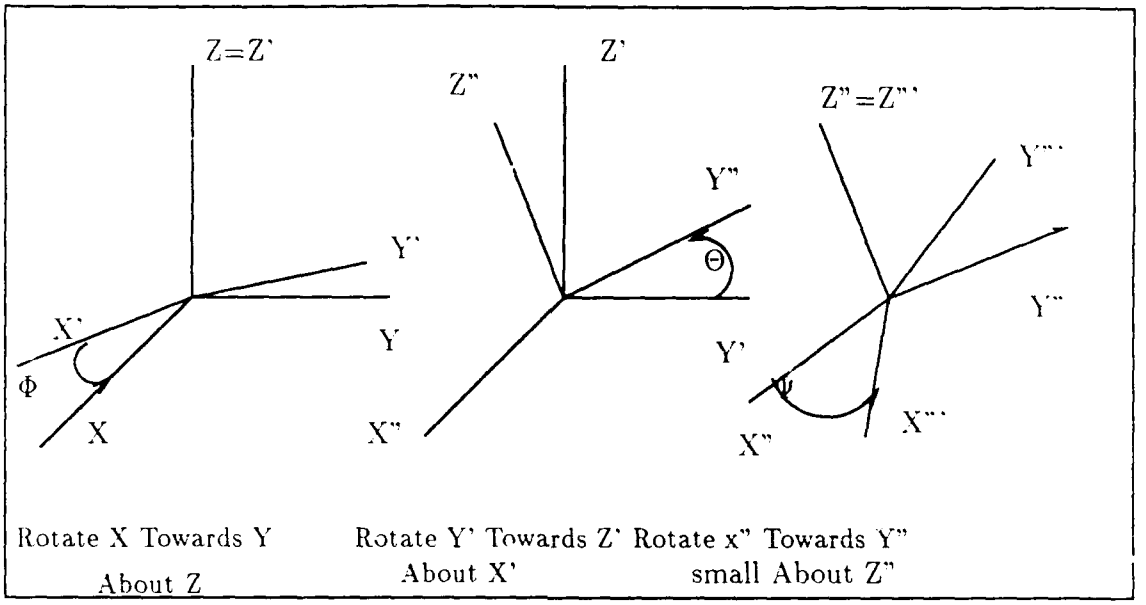


Figure A.2. SarTool^{tm} Euler Angles of Rotation

The Euler angles referred to in the input files determine the degree of rotation of the detailed axis. The actual rotations involved are described below and demonstrated in Figure A.3.5.

1. Euler Angle Φ. The angle Φ is used to define the amount by which the x axis is rotated towards the y axis about the z axis.

2. Euler Angle Θ. The angle Θ is used to define the amount by which the y axis is rotated towards the z axis about the x axis.

3. Euler Angle Ψ. The angle Ψ is used to define the amount by which the x axis is rotated towards the y axis about the z axis.

Appendix B. *Source Code of Thesis Software*

As mentioned in Chapter 3 this thesis effort entailed the design of several software packages so that polarimetric behaviour could be readily examined. The operational function of this software is described in Chapter 3. To assist in any future research endeavours in this field a source listing of each of the packages appears below:

B.1 Reflector File Software

The following piece of software belongs to the sponsor of this thesis effort and was written by M. Bryant.

```
/*****
MOD_RFL.C - Program to enable easy modification to SarTool
    formatted reflector database files.
*****/


#include <stdio.h>
#include <string.h>
#include <math.h>

#define max_primitives 100

FILE *dat_file, *input_file, *output_file;
char new_name[80], temp[80], header[80];
int num_FP, num_DH, num_TH, num_EL, num_CD, num_CY, num_TP,
 num_EP, num_CC, num_CV, num_UG, num_CN, num_ET, index;

typedef struct
{ double x, y, z;
} cartesian;
```

```
cartesian start, end, increment;

typedef struct
{ double phi, theta, psi;
} euler;

typedef struct
{ char type[3];
cartesian location;
euler orientation;
union
{ struct { double x1, x2, y2, x3, y3; } FP;
struct { double d1, d2; } DH;
struct { double d, mxpkdb; } TH;
struct { double axisa, axisb, axisc; } EL;
struct { double d1, d2, rcurv, pkdbm; } CD;
struct { double axisa, axisb, height; } CY;
struct { double axisa, axisb, height; } TP;
struct { double a, b; } EP;
struct { double a, b, height, xbmwth, ybmwth, pkdbsm; }


CC;
struct { double a, b, height, xbmwth, ybmwth, pkdbsm; }


CV;
struct { double a, length; } OG;
struct { double a, b, len1, len2; } CN;
struct { double axisa, axisb, cyoffx, cyoffy,
       angrot, axisc, axisd, height; } ET;
} unique;
int sopa, sopb, sopc;
} primitive;

primitive p[max_primitives];
int i;

void read_header()
{ if (NULL==fgets(header, 80, input_file))
{ puts("*****Error reading header*****\n");
```

```
exit();
}
}

void read_FP(num)
int num;
{ ir ((13!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %lf %lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.FP.x1, &p[i].unique.FP.x2,
&p[i].unique.FP.y2,
&p[i].unique.FP.x3, &p[i].unique.FP.y3,
&p[i].sopa))||(0!=strcmp(p[i].type, "FP")))
{ printf("*****Error reading flatplate %d*****\n", num);



exit();
}
++1;
}

void read_flatplates()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_FP))
{ puts("*****Error reading number of flatplates*****\n");



exit();
}
for (num=0;num<num_FP;++num) read_FP(num);
}

void read_DH(num)
```

```
int num;
{ if ((11!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %d %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.DH.d1, &p[i].unique.DH.d2,
&p[i].sopa, &p[i].sopb))||(0!=strcmp(p[i].type, "DH")))



{ printf("*****Error reading dihedral %d*****\n", num);


exit();
}
++i;
}

void read_dihedrals()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_DH))
{ puts("*****Error reading number of dihedrals*****\n");



exit();
}
for (num = 0;num<num_DH;++num) read_DH(num);
}

void read_TH(num)
int num;
{ if ((12!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %d %d %d\n",
```

```
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.TH.d, &p[i].unique.TH.mxpkdb,
&p[i].sopa, &p[i].sopb,
&p[i].sopc))||(0!=strcmp(p[i].type, "TH")))
{ printf("*****Error reading trihedral %d*****\n", num);



exit();
}
++i;
}

void read_trihedrals()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_TH))
{ puts("*****Error reading number of trihedrals*****\n");



exit();
}
for (num = 0;num<num_TH;++num) read_TH(num);
}

void read_EL(num)
int num;
{ if ((11!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
```

```
&p[i].unique.EL.axisa, &p[i].unique.EL.axisb,
&p[i].unique.EL.axisc, &p[i].sopa))
||(0!=strcmp(p[i].type, "EL")))
{ printf("*****Error reading Ellipsoid %d*****\n", num);



exit();
}
++i;
}

void read_ellipsoids()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_EL))
{ puts("*****Error reading number of ellipsoids*****\n");



exit();
}
for (num = 0;num<num_EL;++num) read_EL(num);
}

void read_CD(num)
int num;
{ if ((13!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %lf %d %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.CD.d1, &p[i].unique.CD.d2,
&p[i].unique.CD.rcurv, &p[i].unique.CD.pkdbm,
&p[i].sopa, &p[i].sopb))
||(0!=strcmp(p[i].type, "CD")))
{ printf("*****Error reading Curved Dihedral %d*****\n",
```

```
num),
exit();
}
++i;
}

void read_curveddihedrals()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_CD))
{ puts("*****Error reading number of curved
dihedrals*****\n");
exit();
}
for (num = 0;num<num_CD;++num) read_CD(num);
}

void read_CY(num)
int num;
{ if ((11!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.CY.axisa, &p[i].unique.CY.axisb,
&p[i].unique.CY.height, &p[i].sopa))
||(0!=strcmp(p[i].type, "CY")))
{ printf("*****Error reading Cylinder %d*****\n", num);


exit();
}
++i;
}

void read_cylinders()
{ int num;
```

```
if (1!=fscanf(input_file, "%d\n", &num_CY))
{ puts("*****Error reading number of cylinders*****\n");



exit();
}
for (num = 0;num<num_CY;++num) read_CY(num);
}

void read_TP(num)
int num;
{ if ((12!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %d %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.TP.axisa, &p[i].unique.TP.axisb,
&p[i].unique.TP.height, &p[i].sopa,
&p[i].sopb))
||(0!=strcmp(p[i].type, "TP")))
{ printf("*****Error reading Top Hat %d*****\n", num);

exit();
}
++i;
}

void read_tophats()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_TP))
{ puts("*****Error reading number of top hats*****\n");


exit();
}
for (num = 0;num<num_TP;++num) read_TP(num);
```

```
}

void read_EP(num)
int num;
{ if ((10!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.EP.a, &p[i].unique.EP.b,
&p[i].sopa))
||(0!=strcmp(p[i].type, "EP")))
{ printf("*****Error reading Elliptical Plate %d*****\n",


num);
exit();
}
++i;
}

void read_ellipticalplates()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_EP))
{ puts("*****Error reading number of elliptical
plates*****\n");
exit();
}
for (num = 0;num<num_EP;++num) read_EP(num);
}

void read_CC(num)
int num;
{ if ((14!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %lf %lf %lf %d\n",
```

```
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.CC.a, &p[i].unique.CC.b,
&p[i].unique.CC.height, &p[i].unique.CC.xbmwth,
&p[i].unique.CC.ybmwth, &p[i].unique.CC.pkdbsm,
&p[i].sopa))
||(0!=strcmp(p[i].type, "CC")))
{ printf("*****Error reading Complex Cavity %d*****\n",

num);
exit();
}
++i;
}

void read_complexcavities()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_CC))
{ puts("*****Error reading number of complex
cavities*****\n");
exit();
}
for (num = 0;num<num_CC;++num) read_CC(num);
}

void read_CV(num)
int num;
{ if ((14!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %lf %lf %lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.CV.a, &p[i].unique.CV.b,
```

```
&p[i].unique.CV.height, &p[i].unique.CV.xbmwth,
&p[i].unique.CV.ybmwth, &p[i].unique.CV.pkdbsm,
&p[i].sopa))
||(0!=strcmp(p[i].type, "CV")))
{ printf("*****Error reading Cavity %d*****\n", num);
exit();
}
++i;
}

void read_cavities()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_CV))
{ puts("*****Error reading number of cavities*****\n");


exit();
}
for (num = 0;num<num_CV;++num) read_CV(num);
}

void read_OG(num)
int num;
{ if ((10!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.OG.a, &p[i].unique.OG.length,
&p[i].sopa))
||(0!=strcmp(p[i].type, "OG")))
{ printf("*****Error reading Ogive %d*****\n", num);
exit();
}
++i;
}
```

```
void read_ogives()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_OG))
{ puts("*****Error reading number of ogives*****\n");
exit();
}
for (num = 0;num<num_OG;++num) read_OG(num);
}

void read_CN(num)
int num;
{ if ((12!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %lf %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.CN.a, &p[i].unique.CN.b,
&p[i].unique.CN.len1, &p[i].unique.CN.len2,
&p[i].sopa))
||(0!=strcmp(p[i].type, "CN")))
{ printf("*****Error reading Cone %d*****\n", num);
exit();
}
++i;
}

void read_cones()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_CN))
{ puts("*****Error reading number of cones*****\n");
exit();
}
for (num = 0;num<num_CN;++num) read_CN(num);
}

void read_ET(num)
int num;
```

```
{ if ((17!=fscanf(input_file, "%s %lf %lf %lf %lf %lf %lf %lf


%lf %lf %lf %lf %lf %lf %lf %d %d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.ET.axisa, &p[i].unique.ET.axisb,
&p[i].unique.ET.cyoffx, &p[i].unique.ET.cyoffy,
&p[i].unique.ET.angrot, &p[i].unique.ET.axisc,
&p[i].unique.ET.axisd, &p[i].unique.ET.height,
&p[i].sopa, &p[i].sopb))
||(0!=strcmp(p[i].type, "ET")))
{ printf("*****Error reading Elliptical Base Top Hat
%d*****\n", num);
exit();
}
++i;
}

void read_ellipticalbasetophats()
{ int num;
if (1!=fscanf(input_file, "%d\n", &num_ET))
{ puts("*****Error reading number of elliptical base top


hats*****\n");
exit();
}
for (num = 0;num<num_ET;++num) read_ET(num);
}

void write_header()
{ if (EOF==fprintf(output_file, "%s\n", header))
{ puts("*****Error writing header*****\n");
exit();
}
}
```

```
void write_FP(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf

%5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.FP.x1, p[i].unique.FP.x2, p[i].unique.FP.y2,



p[i].unique.FP.x3, p[i].unique.FP.y3,
p[i].sopa))
{ printf("*****Error writing flatplate %d*****\n", num);



exit();
}
++i;
}

void write_flatplates()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_FP))
{ puts("*****Error writing number of flatplates*****\n");



exit();
}
for (num = 0;num<num_FP;++num)
{ write_FP(num);
}
}

void write_DH(num)
int num;
```

```
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.DH.d1, p[i].unique.DH.d2,
p[i].sopa, p[i].sopb))
{ printf("*****Error writing dihedral %d*****\n", num);


exit();
}
++i;
}

void write_dihedrals()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_DH))
{ puts("*****Error writing number of dihedrals*****\n");



exit();
}
for (num = 0;num<num_DH;++num) write_DH(num);
}

void write_TH(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d %5d %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.TH.d, p[i].unique.TH.mxpkdb,
p[i].sopa, p[i].sopb, p[i].sopc))
{ printf("*****Error writing flatplate %d*****\n", num);
```

```
exit();
}
++i;
}

void write_trihedrals()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_TH))
{ puts("*****Error writing number of flatplates*****\n");



exit();
}
for (num = 0;num<num_TH;++num) write_FP(num);
}


void write_EL(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.EL.axisa, p[i].unique.EL.axisb,
p[i].unique.EL.axisc, p[i].sopa))
{ printf("*****Error writing Ellipsoid %d*****\n", num);



exit();
}
++i;
}

void write_ellipsoids()
{ int num;
```

```
if (EOF==fprintr(output_file, "%5d\n", num_EL))
{ puts("*****Error writing number of ellipsoids*****\n");



exit();
}
for (num = 0;num<num_EL;++num) write_EL(num);
}

void write_CD(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d %5d\n",



p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
                p[i].orientation.psi,
p[i].unique.CD.d1, p[i].unique.CD.d2,
p[i].unique.CD.rcurv, p[i].unique.CD.pkdbm,
p[i].sopa, p[i].sopb))
{ printf("*****Error writing Curved Dihedral %d*****\n",


num);
exit();
}
++i;
}

void write_curveddihedrals()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_CD))
{ puts("*****Error writing number of curved
dihedrals*****\n");
exit();
}
for (num = 0;num<num_CD;++num) write_CD(num);
```

```
}

void write_CY(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
                p[i].orientation.psi,
p[i].unique.CY.axisa, p[i].unique.CY.axisb,
p[i].unique.CY.height, p[i].sopa))
{ printf("*****Error writing Cylinder %d*****\n", num);


exit();
}
++i;
}

void write_cylinders()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_CY))
{ puts("*****Error writing number of cylinders*****\n");



exit();
}
for (num = 0;num<num_CY;++num) write_CY(num);
}

void write_TP(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
                p[i].orientation.psi,
```

```
p[i].unique.TP.axisa, p[i].unique.TP.axisb,
p[i].unique.TP.height, p[i].sopa,
p[i].sopb))
{ printf("*****Error writing Top Hat %d*****\n", num);

exit();
}
++i;
}

void write_tophats()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_TP))
{ puts("*****Error writing number of top hats*****\n");


exit();
}
for (num = 0;num<num_TP;++num) write_TP(num);
}

void write_EP(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.EP.a, p[i].unique.EP.b,
p[i].sopa))
{ printf("*****Error writing Elliptical Plate %d*****\n",


num);
exit();
}
++i;
}

void write_ellipticalplates()
```

```
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_EP))
{ puts("*****Error writing number of elliptical
plates*****\n");
exit();
}
for (num = 0;num<num_EP;++num) write_EP(num);
}

void write_CC(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf

%10.3lf %5d\n",
p[i].type,
&p[i].location.x, &p[i].location.y, &p[i].location.z,


&p[i].orientation.phi, &p[i].orientation.theta,
&p[i].orientation.psi,
&p[i].unique.CC.a, &p[i].unique.CC.b,
&p[i].unique.CC.height, &p[i].unique.CC.xbmwth,
&p[i].unique.CC.ybmwth, &p[i].unique.CC.pkdbsm,
&p[i].sopa))
{ printf("*****Error reading Complex Cavity %d*****\n",

num);
exit();
}
++i;
}

void write_complexcavities()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_CC))
{ puts("*****Error writing number of complex
cavities*****\n");
exit();
}
for (num = 0;num<num_CC;++num) write_CC(num);
```

```
}

void write_CV(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf

%10.3lf %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.CV.a, p[i].unique.CV.b,
p[i].unique.CV.height, p[i].unique.CV.xbmwth,
p[i].unique.CV.ybmwth, p[i].unique.CV.pkdbsm,
p[i].sopa))
{ printf("*****Error writing Cavity %d*****\n", num);
exit();
}
++i;
}

void write_cavities()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_CV))
{ puts("*****Error writing number of cavities*****\n");


exit();
}
for (num = 0;num<num_CV;++num) write_CV(num);
}

void write_OG(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
```

```
p[i].orientation.psi,
p[i].unique.OG.a, p[i].unique.OG.length,
p[i].sopa))
{ printf("*****Error writing Ogive %d*****\n", num);
exit();
}
++i;
}

void write_ogives()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_OG))
{ puts("*****Error writing number of ogives*****\n");
exit();
}
for (num = 0;num<num_OG;++num) write_OG(num);
}

void write_CN(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %5d\n",


p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.CN.a, p[i].unique.CN.b,
p[i].unique.CN.len1, p[i].unique.CN.len2,
p[i].sopa))
{ printf("*****Error writing Cone %d*****\n", num);
exit();
}
++i;
}

void write_cones()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_CN))
{ puts("*****Error writing number of cones*****\n");
```

```
exit();
}
for (num = 0;num<num_CN;++num) write_CN(num);
}

void write_ET(num)
int num;
{ if (EOF==fprintf(output_file, "%4s %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf %10.3lf

%10.3lf %10.3lf %10.3lf %5d %5d\n",
p[i].type,
p[i].location.x, p[i].location.y, p[i].location.z,
p[i].orientation.phi, p[i].orientation.theta,
p[i].orientation.psi,
p[i].unique.ET.axisa, p[i].unique.ET.axisb,
p[i].unique.ET.cyoffx, p[i].unique.ET.cyoffy,
p[i].unique.ET.angrot, p[i].unique.ET.axisc,
p[i].unique.ET.axisd, p[i].unique.ET.height,
p[i].sopa, p[i].sopb))
{ printf("*****Error writing Elliptical Base Top Hat
%d*****\n", num);
exit();
}
++i;
}

void write_ellipticalbasetophats()
{ int num;
if (EOF==fprintf(output_file, "%5d\n", num_ET))
{ puts("*****Error writing number of elliptical base top


hats*****\n");
exit();
}
for (num = 0;num<num_ET;++num) write_ET(num);
}

void read_input_file()
{ input_file = fopen("INPUT.RFL", "r");
```

```
if (input_file == NULL)
{
puts ("*** Can't open input file ***");
exit (0);
}
i=0;
read_header();
read_flatplates();
read_dihedrals();
read_trihedrals();
read_ellipsoids();
read_curveddihedrals();
read_cylinders();
read_tophats();
read_ellipticalplates();
read_complexcavities();
read_cavities();
read_ogives();
read_cones();
read_ellipticalbasetophats();
}

void write_rfl_file()
{ i=0;
write_header();
write_flatplates();
write_dihedrals();
write_trihedrals();
write_ellipsoids();
write_curveddihedrals();
write_cylinders();
write_tophats();
write_ellipticalplates();
write_complexcavities();
write_cavities();
write_ogives();
write_cones();
write_ellipticalbasetophats();
}

cartesian assign_cartesian(arg1, arg2, arg3)
double arg1, arg2, arg3;
```

```
{
cartesian result;
result.x=arg1;
result.y=arg2;
result.z=arg3;
return (result);
}

cartesian add_cartesian(c1, c2)
cartesian c1, c2;
{
cartesian result;
result.x=c1.x+c2.x;
result.y=c1.y+c2.y;
result.z=c1.z+c2.z;
return (result);
}

void translate(index)
int index;
{
p[index].location=add_cartesian(p[index].location, increment);



}

void calculate_inc()
{
increment.x=(end.x-start.x)/9.0;
increment.y=(end.y-start.y)/9.0;
increment.z=(end.z-start.z)/9.0;
}

int mod()
{
increment=assign_cartesian(start);
translate();
calculate_inc();
for (i=0;i<10;++i)
{ strcpy(new_name, "test_fmt");
sprintf(temp, "%-3d", i);
```

```
strcat(strcat(new_name, "."), temp);
output_file = fopen(new_name, "w");
if (output_file == NULL)
{ puts ("*** Can't open output file ***");
exit (0);
}
write_rfl_file();
close(output_file);
translate();
}
}

void read_dat_file()
{
dat_file = fopen("MOD_RFL.DAT", "r");
if (input_file == NULL)
{
puts ("*** Can't open data file ***");
exit (0);
}

if (3!=(fscanf(dat_file, "%lf %lf %lf\n",
                &start.x, &start.y, &start.z)))
{ puts("*** Error reading data file ***");
exit(0);
}
if (3!=(fscanf(dat_file, "%lf %lf %lf\n", &end.x, &end.y,
&end.z)))
{ puts("*** Error reading data file ***");
exit(0);
}
if (1!=(fscanf(dat_file, "%d", &index)))
{ puts("*** Error reading data file ***");
exit(0);
}
}

void main()
{
read_input_file();

read_dat_file();
```

```
mod();
}
```

B.2 Shell Programme

```
% This section converts the formatted ASCI files into a binary form
for use by SarTool

cnvunf test_fmt.0  test_rfl.0
cnvunf test_fmt.1  test_rfl.1
cnvunf test_fmt.2  test_rfl.2
cnvunf test_fmt.3  test_rfl.3
cnvunf test_fmt.4  test_rfl.4
cnvunf test_fmt.5  test_rfl.5
cnvunf test_fmt.6  test_rfl.6
cnvunf test_fmt.7  test_rfl.7
cnvunf test_fmt.8  test_rfl.8
cnvunf test_fmt.9  test_rfl.9

%Each of the following units sequentially copies the subject
test_rfl files into a test.rfl which is called by the SarTool.dat
file and used as the subject file for each SarTool run.   The
resulting output is placed into a numbered rpdb file and then
"stripped" by the strip programme.   The output of the strip
programme is then copied to a test_out file for later use by the
Mathematica routine.

cp test_rfl.0 test.rfl
sartool  sartool.dat rpdb.0
dbstrip rpdb.0 test_out.0

cp test_rfl.1 test.rfl
sartool  sartool.dat rpdb.1
dbstrip rpdb.1 test_out.1

cp test_rfl.2 test.rfl
sartool  sartool.dat rpdb.2
dbstrip rpdb.2 test_out.2
```

```
cp test_rfl.3 test.rfl
sartool  sartool.dat rpdb.3
dbstrip rpdb.3 test_out.3

cp test_rfl.4 test.rfl
sartool  sartool.dat rpdb.4
dbstrip rpdb.4 test_out.4

cp test_rfl.5 test.rfl
sartool  sartool.dat rpdb.5
dbstrip rpdb.5 test_out.5

cp test_rfl.6 test.rfl
sartool  sartool.dat rpdb.6
dbstrip rpdb.6 test_out.6

cp test_rfl.7 test.rfl
sartool  sartool.dat rpdb.7
dbstrip rpdb.7 test_out.7

cp test_rfl.8 test.rfl
sartool  sartool.dat rpdb.8
dbstrip rpdb.8 test_out.8

cp test_rfl.9 test.rfl
sartool  sartool.dat rpdb.9
dbstrip rpdb.9 test_out.9

% The final line of the shell programme removes the test.rfl file
in preperation for the next SarTool run.

rm test.rfl
```

B.3 Strip Software

The strip software "strips" the output of SarTool and converts it into in phase and quadrature components for later use by the Mathematica routine.

```
c-------------------------------------------------------------
c
c                    program dbstrip
c
c-------------------------------------------------------------
c
c      Routine -- dbstrip (standalone)
c
c
c      Purpose -- To extract the magnitude of the 4 polarizations
c                 of a SarTool range profile database file
c
c
c      Usage:      dbstrip SarTool_database_file, Output_file
c
c
c      Command-line arguments:
c
c
c (1) Input file name (example: rpdb_mm_dd.xyz)
c             implicit undefined (a-z)

c --- The following are declarations for the database history
 variables

      double precision sldsvr, rarexv, toolv,  xsnsr0, ysnsr0,
                       zsnsr0, fltang, divang, snsspd, xtgt,ytgt,
                       ztgt,phit,thetat, psit, vxtgt,  vytgt,
                       vztgt, axtgt, aytgt,aztgt, freq, dbwel,
                       dbwaz, dborel, dboraz, aspinc, dr, da,
                       surfac, bckatt, thntrs, dbter(4), rtod

      integer          jdate(9), irxdat(9), nr, na, np, i, nintrv,
                       imagtr, iphstr iargc, nargs

      logical          isrflg, sptflg, shdflg, terflg, bshflg,
                       cmpflg, thnflg,
                       wgtflg, mltflg, movflg
      character*50 title
      character*30 rrxttl
      character*14 tgtnam
```

```
      real*8 I11, Q11, I12, Q12, I21, Q21, I22, Q22
      real*8 PAmp, PPhase

      double precision P11(500,2), P12(500,2), P21(500,2),
                       P22(500,2)

      integer inputunit, outputunit
      logical create
      character*50 infil, outfil

c --- Begin executables

      data rtod /57.29577951/

c --- Check the command line for the proper number of arguments

      nargs = iargc()
      if (nargs .lt. 1) then
         write(*,*)
              ' Usage: dbstrip SarTool_database_file,
Output_text_file'
         stop
      endif

c --- If all okay, retrieve the command line argument

      call getarg(1,infil)
      call getarg(2,outfil)

c --- Define the SarTool database input file logical unit number

      inputunit = 9
      outputunit = 11

c --- Open the file containing the range profile database

      inquire (file = infil, exist = create)
      if (.not. create) then
         write (6,*)'Input file',infil,'does not exist.'
         stop
      endif
```

```
      open (unit   =  inputunit,
     .      file   =  infil,
     .      access = 'direct',
     .      form   = 'unformatted',
     .      status = 'old',
     .      recl   =  64)

c --- Read parameters from the database header

      read(inputunit,rec= 1) title, tgtnam
      read(inputunit,rec= 2) jdate
      read(inputunit,rec= 3) rrxttl
      read(inputunit,rec= 4) sldsvr, rarexv,toolv
      read(inputunit,rec= 5) irxdat
      read(inputunit,rec= 6) xsnsr0,   ysnsr0,   zsnsr0,   fltang,
     .                       divang,   snsspd,   xtgt,     ytgt
      read(inputunit,rec= 7) ztgt,     phit,     thetat,   psit,
     .                       vxtgt,    vytgt,    vztgt,    axtgt
      read(inputunit,rec= 8) aytgt,    aztgt,    freq,     dbwel,
     .                       dbwaz,    dborel,   dboraz,   aspinc
      read(inputunit,rec= 9) dr,       da,       surfac,   bckatt,
     .                       dbter(1), dbter(2), dbter(3), dbter(4)
      read(inputunit,rec=10) thntrs
      read(inputunit,rec=11) nr,       na,       np,       nintrv,
     .                       imagtr,   iphstr,   isrflg,   sptflg,
     .                       shdflg,   terflg,   bshflg,   cmpflg,
     .                       thnflg,   wgtflg,   mltflg,   movflg

      DO 100, i=12, np+11
          read(inputunit,rec=i) I11, Q11, I12, Q12, I21, Q21, I22,
                                Q22
          P11(i-11,1)=I11
          P11(i-11,2)=Q11
          P12(i-11,1)=I12
          P12(i-11,2)=Q12
          P21(i-11,1)=I21
          P21(i-11,2)=Q21
          P22(i-11,1)=I22
          P22(i-11,2)=Q22
100   continue

      close (inputunit)
```

```
      open (outputunit, file=outfil)
      do 200, i=1, np

          PAmp=(P11(i,1)**2 + P11(i,2)**2)**.5
          IF (P11(i,1).EQ.0) THEN
             PPhase=0
          ELSE
             PPhase=DATAN2(P11(i,2),P11(i,1))
            END IF
          write(outputunit,900) PAmp,PPhase

          PAmp=(P12(i,1)**2 + P12(i,2)**2)**.5
          IF (P12(i,1).EQ.0) THEN
             PPhase=0
          ELSE
             PPhase=DATAN2(P12(i,2),P12(i,1))
          END IF
          write(outputunit,900) PAmp,PPhase

          PAmp=(P21(i,1)**2 + P21(i,2)**2)**.5
          IF (P21(i,1).EQ.0) THEN
             PPhase=0
          ELSE
             PPhase=DATAN2(P21(i,2),P21(i,1))
          END IF
          write(outputunit,900) PAmp,PPhase

          PAmp=(P22(i,1)**2 + P22(i,2)**2)**.5
          IF (P22(i,1).EQ.0) THEN
             PPhase=0
          ELSE
             PPhase=DATAN2(P22(i,2),P22(i,1))
          END IF
          write(outputunit,900) PAmp,PPhase

200   continue
      close (outputunit)

      stop

c --- Format statements
```

```
 900  format(e12.5)

      end
```

B.4 Mathematica Routine

The following programme is an abridged version of the Mathematica routine. Each intermediate unit of the routine is preceeded by functional documentation.

```
The following two files contain the "source code" for the
parametric plots and the Poincare sphere

The next line reads in the In Phase and Quadrature data from
SarTool using the " Strip Programme".

fulllist=ReadList [InputString ["input file name"],
{{Real, Real}, {Real, Real}, {Real, Real}, {Real, Real}}];

interval=7;
np=200;
pulses=Floor[np/interval];

m={};

Block[  {i},
m=Table[fulllist[[i]],{i,1,np+1-interval,interval}];
 ];

The following four tables tabulate  the Scattering  Matrix
components
for each pulse

saa=Table[(m[[i,1,1]]*Exp[I m[[i,1,2]]]),{i,pulses}];
```

```
sba=Table[(m[[i,3,1]]*Exp[I m[[i,3,2]]]),{i,pulses}];

sbb=Table[(m[[i,4,1]]*Exp[I m[[i,4,2]]]),{i,pulses}];

sab=Table[(m[[i,2,1]]*Exp[I m[[i,2,2]]]),{i,pulses}];
```

The next two lines calculate the intermediate variables rho 1@2 (p1c @ p2c) which are used to calculate q1c & q2c which are in turn used for the eventual calculation of the latitude & longitude of the copols on the Poincare Sphere.

```
p1c= Table[ If[sbb[[i]]==0,
                              If[sab[[i]]==0,1,1],
                              (-(sab[[i]])
                              + Sqrt[(sab[[i]])^2
                              - sbb[[i]]*saa[[i]]])/(sbb[[i]])]
,{i,pulses}];

p2c= Table[ If[sbb[[i]]==0,
                              If[sab[[i]]==0,-1,-Sign[sab[[i]]]
                              *Infinity],
                              (-(sab[[i]])
                               - Sqrt[(sab[[i]])^2
                              - sbb[[i]]*saa[[i]]])/(sbb[[i]])]
,{i,pulses}];
```

The following section determines the values of r1,r2 @ r3 which are intermediate results used to determine the variables rho 1@2 (p1x @ p2x) which in turn are used to calculate the latitude and longitude of the crosspols on the Poincare Sphere.

```
r1= Table[ ( (Abs[sbb[[i]]])^2
                              -(Abs[saa[[i]]])^2)
                              -sab[[i]]*Conjugate[sba[[i]]]
                              +sba[[i]]*Conjugate[sab[[i]]]
                  ,{i,pulses}];

r2= Table[ ( sbb[[i]]*Conjugate[sba[[i]]]
                              +sba[[i]]*Conjugate[saa[[i]]]    )
,{i,pulses}];
```

```
r3= Table[ ( saa[[i]]*Conjugate[sab[[i]]]
                          +sab[[i]]*Conjugate[sbb[[i]]]    )
,{i,pulses}];

p1x= Table[ If[r2[[i]]==0,
                   if[r1[[i]]>0,Sign[r1[[i]]]*Infinity,0],
( r1[[i]]
+ Sqrt[r1[[i]]^2
+ 4*r2[[i]]*r3[[i]]])/(2*r2[[i]])]
 ,{i,pulses}];

p2x= Table[ If[r2[[i]]==0,
                   If[r1[[i]]<0,Sign[r1[[i]]]*Infinity,0],
( r1[[i]]
- Sqrt[r1[[i]]^2
+ 4*r2[[i]]*r3[[i]]])/(2*r2[[i]])]
 ,{i,pulses}];
```

The following sections determine the values of q for each of the p values. These q values are subsequently used to determine the theta and phi values of the cross and copol nulls on the Poincare Sphere.

```
q1x= Table[If[SameQ[p1x[[i]],Infinity] || SameQ[p1x[[i]],
                        -Infinity]
                        ,-1
                        ,If[SameQ[p1x[[i]],1. I]
                            ,1
                            ,(1-I*p1x[[i]])/(1+I*p1x[[i]])]]
,{i,pulses}];

q2x=Table[If[SameQ[p2x[[i]],Infinity] || SameQ[p2x[[i]],
                        -Infinity]
                       ,-1
                       ,If[SameQ[p2x[[i]],1. I]
                           ,1
                           ,(1-I*p2x[[i]])/(1+I*p2x[[i]])]]
,{i,pulses}];

q1c=Table[If[SameQ[p1c[[i]],1. I]
,1
```

```
,((1-I*p1c[[i]])/(1+I*p1c[[i]]))]
,{i,pulses}];


q2c=Table[If[SameQ[p2c[[i]],1. I]
,1
,((1-I*p2c[[i]])/(1+I*p2c[[i]]))]
,{i,pulses}];
```

The following section determines the co-latitude for each of the above q values.

```
crosspole1Lat=Table[N[ArcCos[-((1-(Abs[q1x[[i]]])^2)
                        /(1+(Abs[q1x[[i]]])^2))],5]
        ,{i,pulses}];

crosspole2Lat=Table[N[ArcCos[-((1-(Abs[q2x[[i]]])^2)
                        /(1+(Abs[q2x[[i]]])^2))],5]
        ,{i,pulses}];

copole1Lat=Table[If[SameQ[p1c[[i]],1.I]
                        ,0
                        ,N[ArcCos[-((1- bs[q1c[[i]]])^2)
                        /(1+(Abs[q1c[[i]]])^2))]],5]
        ,{i,pulses}];

copole2Lat=Table[If[SameQ[p2c[[i]],i.I]
                         ,0
                         ,N[ArcCos[-((1-(Abs[q2c[[i]]])^2)
                         /(1+(Abs[q2c[[i]]])^2))]],5]
        ,{i,pulses}];
```

The following section determines the longitude of each of the cross and copol nulls.

```
crosspole1Long=Table[If[Re[q1x[[i]]]==0,
                        If[Im[q1x[[i]]]==0,
                           N[ArcTan[-1.0],5],
                           -Sign[Im[q1x[[i]]]]*N[Pi/2,5]],
```

```
                              N[ArcTan[Re[q1x[[i]]],-Im[q1x[[i]]]]
                              ],5 ]
        ,{i,pulses}];

crosspole2Long=Table[If[Re[q2x[[i]]]==0,
                        If[Im[q2x[[i]]]==0,
                           N[ArcTan[-1.0],5],
                           -Sign[Im[q2x[[i]]]]*N[Pi/2,5]],
                           N[ArcTan[Re[q2x[[i]]],-Im[q2x[[i]]]]
                           ],5 ]
        ,{i,pulses}];

copole1Long=Table[If[Re[q1c[[i]]]==0,
                        If[Im[q1c[[i]]]==0,
                           N[ArcTan[-1.0],5],
                           -Sign[Im[q1c[[i]]]]*N[Pi/2,5]],
                           N[ArcTan[Re[q1c[[i]]],-Im[q1c[[i]]]]
                           ],5]
        ,{i,pulses}];

copole2Long=Table[If[Re[q2c[[i]]]==0,
                        If[Im[q2c[[i]]]==0,
                           N[ArcTan[-1.0],5],
                           -Sign[Im[q2c[[i]]]]*N[Pi/2,5]],
                           N[ArcTan[Re[q2c[[i]]],-Im[q2c[[i]]]]
                           ],5 ]
        ,{i,pulses}];



The following section plots the co and cross polarizations on the
Poincare Sphere.

copoleList=Table[{copole1Lat[[i]], copole1Long[[i]]},{i,pulses}];

                        Do[AppendTo[copoleList,{copole2Lat[[i]],
                        copole2Long[[i]]}]
        ,{i,pulses}];

crosspoleList=Table[{crosspole1Lat[[i]],
crosspole1Long[[i]]},{i,pulses}];
```

```
Do[AppendTo[crosspoleList,{crosspole2Lat[[i]]
                          , crosspole2Long[[i]]}]
        ,{i,pulses}];
```

The following is appended to the Mathematica sphere graphic

```
         Thickness[.009], Line[{{0.,0.,1.},{0.,0.,-1.}}],
         Thickness[.005], Line[{{1.,0.,0.},{-1.,0.,0.}}],
         Thickness[.001], GrayLevel[.5],Line[{{0.,1.,0.},
                          {0.,-1.,0.}}], GrayLevel[0] };
                  copolelen = Length[copoleList];
                  copoleplot=Table[{PointSize[.02],
                  Point[{Sin[copoleList[[i,1]]]
                            Cos[copoleList[[i,2]]],

                  crosspolelen = Length[crosspoleList];
                  crosspoleplot=Table[ {PointSize[.02],
                  GrayLevel[.5],Point[{Sin[crosspoleList[[i,1]]]
                             Cos[crosspoleList[[i,2]]],

          AppendTo[sphere,copoleplot];
          AppendTo[sphere,crosspoleplot];
```

The following graphics shows the co and cross polarizations plotted on the Poincare sphere. The copolarizations are the darker of the points. Refer to main body of thesis for examples of this output.

```
test=Graphics3D[sphere];
Show[test, Boxed -> False];
```

The following graph shows the span of the returns.

```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2

                  + Abs[sbb[[i]]]^2),{i,1,pulses}]

ListPlot[crosspole1Lat,PlotJoined -> True, PlotRange ->{-3.2,3.2},
                  PlotLabel -> "crosspole1 Lat"]

test1=ListPlot[crosspole1Long,PlotJoined->True,PlotRange->{-3.2,3
               .2},
```

```
                    PlotLabel ->"crosspole1 Long"]

ListPlot[copole1Lat,PlotJoined->True,PlotRange->{-3.2,3.2},
                    PlotLabel -> "copole1 Lat"]

ListPlot[copole1Long,PlotJoined->True,PlotRange->{-3.2,3.2},
                    PlotLabel ->"copole1 Long"]

ListPlot[crosspole2Lat,PlotJoined -> True, PlotRange ->{-3.2,3.2},
                    PlotLabel -> "crosspole2 Lat"]

test2=ListPlot[crosspole2Long,PlotJoined->True,PlotRange->{-3.2,3
                    .2},
                    PlotLabel ->"crosspole2 Long"]

ListPlot[copole2Lat,PlotJoined->True,PlotRange->{-3.2,3.2},
                    PlotLabel -> "copole2 Lat"]

ListPlot[copole2Long,PlotJoined->True,PlotRange->{-3.2,3.2},
                    PlotLabel -> "copole2 Long"]

z=Table[{copole1Lat[[i]],copole1Long[[i]]},{i,1,pulses}]

ListPlot[z, PlotLabel->"copole1Lat versus copole1.Long"]

y=Table[{copole2Lat[[i]],copole2Long[[i]]},{i,1,pulses}]

Show[test1,test2]
```

B.5 Convert Programmme

The following programme converts the mathematica output into a form suitable for input to G. Tarr's multi level perceptron network.

```
DIM elem$(100)
INPUT "What file contains the filenames to be processed" , n
```

```
OPEN "n" FOR INPUT AS #1
DO UNTIL EOF(1)
   INPUT #1, file$
   OPEN file$ FOR BINARY AS #2
   OPEN "dummy.dat" FOR BINARY AS #3
   DO UNTIL EOF(2)
      GET #2,  elem
      IF elem$ = "{" OR elem$ = "}" THEN
               elem$ = STRING$(1, 0)
      ELSEIF elem$ = "," THEN
               elem$ = STRING$(1, 13)
               PUT #3,  elem
                elem$ = STRING$(1, 10)
               PUT #3,  elem
      ELSE
               PUT #3,  elem
      END IF

   LOOP
   CLOSE #2, #3
LOOP
KILL file$
NAME "dummy.dat" AS file$
END
```

B.6 Single Perceptron Network

The following programme creates a single level perceptron. The basis of this programme was written for an A.I. unit by the author and Capt J. De-Barry. The author then modified the programme such that it could train and test on operator selected inputs and return decision files and outputs. Extensive documentation appears at the end of the programme.

```
'DECLARE SUB instruct ()
'DECLARE SUB init (eta!, weight#(), theta!, count!, right!, wrong!,
maxcount!, O$, s!, train!, n!, r, outfile$, trainfile$)
'DECLARE SUB dist1 (vector#(), class!)
'DECLARE SUB dist2 (vector#(), class!)
'DECLARE SUB dist3 (vector#(), class!)
'DECLARE SUB dist4 (vector#(), class!)
'DECLARE SUB readin (DATAFILE!, vect!(), class!(), n!, trainfile$)
'DECLARE SUB dist5 (vector#(), class!, class!(), vect!(), count!,
n!)
'DECLARE SUB feedforward (guess!, vector#(), weight#(), theta!,
count!, train!, maxcount!, x!, y2)
'DECLARE SUB backprop (guess!, class!, eta!, weight#(), vector#(),
theta!, y2)
'DECLARE SUB backprop2 (guess!, class!, right!, wrong!, ratio!,
weight#())
'DECLARE SUB display (vector#(), ratio!, weight#(), theta!, d!,
guess!, class!, count!, maxcount!, s!, train!, r, outfile$, O$)
DIM vector#(1), weight#(1)
DIM vect(450, 2)
DIM class(450)
RANDOMIZE TIMER

   DEF FNy1# (m, x, B) = (-m) * x + B

   SCREEN 9
   CALL instruct
   DO

      CALL init(o,q,eta, weight#(), theta, count, right, wrong,
    maxcount, O$, s, train,n,r,p )
      WINDOW (-6.5, -6.5)-(6.5, 6.5)
      VIEW SCREEN (PMAP(-3.25 + r, 0), PMAP(3.25 + s,1))-(PMAP(3.25
      + r, 0), PMAP(-3.25 + s, 1))

      blockstep = maxcount
      DO
         IF O$ = "1" THEN
            CALL dist1(vector#(), class)
         ELSEIF O$ = "2" THEN
            CALL dist2(vector#(), class)
         ELSEIF O$ = "3" THEN
```

```
   CALL dist3(vector#(), class)
ELSEIF O$ = "4" THEN
   CALL dist4(vector#(), class)
ELSEIF O$ = "5" THEN
       IF count = 1 THEN
         CALL readin(q,DATAFILE, vect(), class(), n,
                     trainfile$,testfile$)
       ELSEIF count > 1 THEN
         CALL dist5(k,vector#(), class, class(), vect(),
                    count, n,o)
       END IF
END IF
CALL feedforward(guess, vector#(), weight#(), theta,
            count, train, maxcount, x, y2)
   IF train = maxcount OR count < train THEN
      CALL backprop(guess, class, eta, weight#(),
             vector#(), theta, y2)
   ELSEIF train <> maxcount AND  count > train THEN
       IF count = train +1 AND q<>1 AND O$="5" THEN
        CALL readin2,DATAFILE, vect(),class(),o,
     trainfile$,testfile$)
       END IF
      CALL backprop2(guess, class, right, wrong, ratio,
                     weight#())
   END IF
CALL display(k,p,vector#(), ratio, weight#(), theta, d,
            guess, class, count, maxcount, s, train, r,
            outfile$, O$)
count = count + 1
IF count = maxcount + 1 THEN
   DO
    LOCATE 23, 8
     IF train = maxcount OR train > maxcount THEN
       INPUT "Do you want to continue training (Y/N)"; A$

     ELSEIF train < maxcount THEN
       INPUT "Do you want to continue testing (Y/N)"; A$

     END IF
   LOOP UNTIL A$ = "y" OR A$ = "Y" OR A$ = "n" OR A$ = "N"

   LOCATE 23, 6
```

```
            PRINT "
    "
            IF A$ = "y" OR A$ = "Y" THEN
               maxcount = maxcount + blockstep
               LOCATE 1, 25
               PRINT "DYNAMIC";
            END IF
         END IF
      LOOP UNTIL count = maxcount + 1
      LOCATE 23, 8
      INPUT "Do you want run the simulation again (Y/N)"; A$
      CLS
      WINDOW
      VIEW
      CLS
   LOOP UNTIL A$ = "n" OR A$ = "N"


END

SUB backprop (guess, class, eta, weight#(1), vector#(1), theta, y2)

'      IF guess <> class THEN
         FOR I = 0 TO 1
            weight#(I) = weight#(I) +(eta/2) * (class - y2) *
vector#(I)
         NEXT I
         theta = theta - eta / 2 * (class - y2)
'      END IF
'      LOCATE 13,1
'      PRINT "b/1 weight#(0)=" weight#(0)
'      PRINT "b/1 weight#(1)=" weight#(1)
END SUB

SUB backprop2 (guess, class, right, wrong, ratio, weight#(1))
'      LOCATE 15,1
'      PRINT "b/2 weight#(0)= " weight#(0)
'      PRINT "b/2 weight#(1)= " weight#(1)
      IF guess <> class THEN
        wrong = wrong + 1
      ELSEIF guess = class THEN
         right = right + 1
```

```
      END IF
      ratio = right / (right + wrong)
      LOCATE 10, 1
      PRINT "Testing Data"
      PRINT "wrong="; wrong
      PRINT "right="; right
      PRINT "%correct="; ratio * 100
END SUB

SUB display (k,p,vector#(1), ratio, weight#(1), theta, d, guess,
class, count, maxcount, s, train, r, outfile$, O$)
  SHARED trainfile$, testfile$
      STATIC oldm, oldb
      IF count = 1 THEN
         LOCATE 1, 25
         PRINT "INITIAL PERCEPTRON DECISION SPACE";
         LOCATE 2, 23
          IF O$ = "5" THEN
            PRINT "INPUT TRNG/TST FILES  ";
trainfile$;"/";testfile$
          END IF
         oldm = 1
         oldb = .2
      ELSEIF count = 2 THEN
         LOCATE 1, 25
         PRINT "DYNAMIC";
      ELSEIF count + 1 >= maxcount THEN
         LOCATE 1, 25
         PRINT "  FINAL";
      END IF
      IF count < train THEN
        IF O$="5" THEN
         LOCATE 3, 34
         PRINT "TRAINING PHASE"
        ELSEIF O$<>"5" THEN
         LOCATE 2,34
         PRINT "TRAINING PHASE"
        END IF
      ELSEIF count > train THEN
       IF O$="5" THEN
        LOCATE 3,27
        PRINT "TESTING PHASE - WEIGHTS FROZEN"
```

```
        ELSEIF O$<>"5" THEN
         LOCATE 2,27
         PRINT "TESTING PHASE- WEIGHTS FROZEN"
         END IF
       END IF
       LOCATE 4, 1
       PRINT "vector#(0)="; CSNG(vector#(0))
       PRINT "vector#(1)="; CSNG(vector#(1))
       PRINT "class="; class
       PRINT "guess="; guess
       LINE (-3.2 + r, FNy1#(oldm, -3.2, oldb) + s)-(3.2 +
r,FNy1#(oldm, 3.2, oldb) + s), 0
       LINE (0 + r, 3.2 + s)-(0 + r, -3.2 + s), 15
       LINE (-3.2 + r, 0 + s)-(3.2 + r, 0 + s), 15
       newm# = weight#(0) /weight#(1)
       newb# = theta /weight#(1)
       LINE (-3.2 + r, FNy1#(newm#, -3.2, newb#) + s)-(3.2 +
r,FNy1#(newm#, 3.2, newb#) + s), 2
       IF count = 1 THEN
          LOCATE 5, 46
          PRINT "3.2";
          LOCATE 17, 45
          PRINT "-3.2";
          LOCATE 11, 25
          PRINT "-3.2";
          LOCATE 11, 68
          PRINT "3.2";
          LOCATE 23, 8
          INPUT "Press enter to begin training ...", A$
          LOCATE 23, 8
          PRINT "                                          "
       END IF
          IF class = 1 THEN
          PSET (vector#(0) + r, vector#(1) + s), 14
       ELSEIF class = -1 THEN
          PSET (vector#(0) + r, vector#(1) + s), 9
       END IF
       oldm = weight#(0) / weight#(1)
       oldb = theta / weight#(1)
       LOCATE 19, 1
       PRINT "W(0) = "; CSNG(weight#(0));
       LOCATE 19, 40
```

```
      PRINT "W(1) = "; CSNG(weight#(1))
      LOCATE 20, 20
      PRINT "Theta = "; CSNG(theta)
      LOCATE 21, 1
      PRINT "ITERATION - "; count;
      LOCATE 21, 40
      PRINT "INTERCEPT = "; CSNG(oldb)
      LOCATE 22, 20
      PRINT "SLOPE = "; CSNG(oldm)
      LOCATE 23, 8
      PRINT "                                                    "
      corr# = ratio*100
      IF O$ = "5" AND p=1 AND count > train THEN
         OPEN outfile$ FOR APPEND AS #2
         IF count = train + 1  THEN
            PRINT #2, "                    ";outfile$;"
             "
            PRINT #2,
            PRINT #2,"      Selection     Class       Guess
                   %Correct"
            PRINT #2, SPC(9) count SPC(9) class SPC(9) guess SPC(9)
                  corr#
         ELSEIF count > train + 1 THEN
            PRINT #2, SPC(9) count SPC(9) class SPC(9) guess SPC(9)
            corr#
         END IF
         CLOSE #2
      END IF


END SUB

SUB dist1 (vector#(1), class)
      vector#(0) =3*(RND * 2 - 1)
      IF vector#(0) > 0 THEN
         vector#(1) = RND *3
         class = 1
      ELSEIF vector#(0) < 0 THEN
         vector#(1) = RND * (-3)
         class = -1
      END IF
END SUB
```

```
SUB dist2 (vector#(1), class)
    IF RND > .5 THEN
      class = 1
      vector#(0) = 3*( RND * (-.9) - .1)
      vector#(1) = 3*(RND * (.6) + .1 )
    ELSE
      class = -1
      vector#(0) =3*(RND * 1.2 - .6)
      IF vector#(0) < .3 THEN
         vector#(1) = 3*(RND * (-.4) - .1)
      ELSE
         vector#(1) =3*( RND * 1.2 - .5)
      END IF

    END IF
END SUB

   SUB dist3 (vector#(1), class)
      vector#(0) = RND * 6 - 3
      IF vector#(0) > .6 THEN
         vector#(1) = 3*(RND - .2)
         class = 1
      ELSEIF vector#(0) < -.6 THEN
         vector#(1) = 3*(RND * (-1) + .2)
         class = -1
      ELSE
         IF RND > .5 THEN
            vector#(1) = 3*(RND - .2)
            class = 1
         ELSE
            vector#(1) = 3*(RND * (-1) + .2)
            class = -1
         END IF
      END IF
  END SUB

SUB dist4 (vector#(1), class)
    IF RND > .5 THEN
      class = 1
      vector#(0) =3*(RND * (-.9) - .1)
      vector#(1) =3*(RND * (.6) + .1)
```

```
    ELSE
      class = -1
      vector#(0) =3*(RND - .4)
      IF vector#(0) < .3 THEN
         vector#(1) = 3*(RND * (-.4) - .1)
      ELSE
         vector#(1) = 3*(RND * .8 - .7)
      END IF

    END IF
END SUB

SUB dist5 (k,vector#(1), class, class(1), vect(2), count, n,o)
    IF trainfile$ = testfile$  THEN
       DO UNTIL k <> 0
       k = CINT(RND * n)
       LOOP
    ELSEIF trainfile$<>testfile$ THEN
       DO UNTIL k<>0
       k = CINT(RND * o)
       LOOP
    END IF
     LOCATE 8,1
     PRINT "vector selected=" k
     vector#(0) = vect(k, 1)
     vector#(1) = vect(k, 2)
          class = class(k)
              k = 0
END SUB

SUB feedforward (guess, vector#(1), weight#(1), theta, count,
train, maxcount, x, y2)

     Z = vector#(0) * weight#(0) + vector#(1) * weight#(1) - theta

     IF Z < -1 OR Z > 1 THEN
       IF Z > 0 THEN
         guess = 1
         y2 = 1
       ELSEIF Z < 0 THEN
         guess = -1
         y2 = -1
```

```
      END IF
    ELSEIF Z>0 THEN
      guess =1
      y2 =Z
    ELSEIF  Z<0 THEN
      guess = -1
      y2 = Z
    END IF

   LOCATE 15, 1
   PRINT "Initialization params"
   PRINT "train="; train
   PRINT "total iterations="; maxcount
  END SUB

SUB init (o,q,eta, weight#(1), theta, count, right, wrong,
maxcount, O$, s, train,n,r,p )
     SHARED outfile$, trainfile$, testfile$
     r = 1!
     s = 1!
     count = 1
     wrong = 0
     right = 0
     weight#(0) = RND * .2
     weight#(1) = (RND - 1) * .2
     theta = (RND - 1) * .1
     DO
        LOCATE 10, 18
        PRINT "Choose class separation type 1, 2, 3, 4 or 5: ";

        LOCATE 12, 21
        PRINT "(1) linearly separable (quadrant I and III)";
        LOCATE 13, 21
        PRINT "(2) linearly unseparable";
        LOCATE 14, 21
        PRINT "(3) overlapping";
        LOCATE 15, 21
        PRINT "(4) linearly separable (arbitrary)";
        LOCATE 16, 21
        PRINT "(5) read in data";
        LOCATE 18, 18
        INPUT "Your choice"; O$
```

```
      LOOP UNTIL O$ = "1" OR O$ = "2" OR O$ = "3" OR O$ = "4" OR
O$ = "5"
      CLS
      LOCATE 12, 24
      INPUT "How many total iterations"; maxcount
      CLS
      LOCATE 12, 14
      INPUT "Do you wish to test as well as train the network
(Y/N)"; A$
      CLS
        IF A$ = "y" OR  A$ = "Y" THEN
          LOCATE 12,24
          INPUT "How many training iterations"; train
          CLS
        ELSEIF A$ = "n" OR A$ = "N" THEN
           train = maxcount
        END IF

        IF O$ = "5" THEN
          LOCATE 12, 24
          INPUT "What training input file name"; trainfile$
          CLS
          LOCATE 12,24
          INPUT  "What testing input file name"; testfile$
            IF trainfile$ = testfile$ THEN
             q=1
            END IF
          CLS
          LOCATE 12, 24
          INPUT "How many entries on training file"; n
          CLS
            IF q<>1 THEN
              LOCATE 12,24
              INPUT " How many entries on testing file";o
              CLS
            END IF
          LOCATE 12, 18
          INPUT "Do you wish to write an output file (Y/N)";P$
          CLS
            IF P$ = "n" or P$ = "N" THEN
             p=0
            ELSEIF P$ = "y" or P$ = "Y" THEN
```

```
            p=1
            LOCATE 12,24
            INPUT "What output file"; outfile$
            CLS
           END IF
        ELSEIF O$ <> "5" THEN
          LOCATE 12, 24
          PRINT SPC(65);
        END IF
      LOCATE 12, 17
      INPUT "What training gain value (0.3 suggested)"; eta
      CLS
      END SUB

SUB instruct
   CLS
   LOCATE 8, 24
   PRINT "            PERCEPT              ";
   LOCATE 10, 23
   PRINT "THE PERCEPTRON TRAINING SIMULATOR";
   LOCATE 12, 35
   FOR I = 1 TO 6000
   NEXT I
   CLS
   DO
      LOCATE 12, 25
      INPUT "Do you want instructions"; r$
   LOOP UNTIL r$ = "y" OR r$ = "Y" OR r$ = "n" OR r$ = "N"
   CLS
   IF r$ = "y" OR r$ = "Y" THEN
      LOCATE 2, 4
      PRINT "TRAINING PARAMETERS"
      LOCATE 4, 6
      PRINT "You will be prompted to enter several training
      parameters."
      LOCATE 5, 6
      PRINT "First you will be asked to select the type of class
      separation"
      LOCATE 6, 6
      PRINT "you want.  Next you will be prompted for the number
      of total"
      LOCATE 7, 6
```

```
PRINT "iterations to run before pausing.  You may input any
positive"
LOCATE 8, 6
PRINT "integer.  You will then be asked whether or not you
wish to"
LOCATE 9, 6
PRINT "perform testing with the net.   A positive response
to this"
LOCATE 10, 6
PRINT "input will cause the weights to be frozen at the
specified"
LOCATE 11, 6
PRINT "training number after which the net will use those
weights to"
LOCATE 12, 6
PRINT "make decisions on subsequent inputs.   If you
initially "
LOCATE 13, 6
PRINT "selected option 5 (read in data) you will be prompted
to input"
LOCATE 14, 6
PRINT "the number of entries to be read from the input file.
You will"
LOCATE 15,6
PRINT "also be asked to enter the input and output file
names. With"
LOCATE 16,6
PRINT "selection 5 there is also the capability to test from
a file
LOCATE 17,6
PRINT "other than the one trained on this is achieved simply
by
LOCATE 18,6
PRINT "answering the questions given during program
initialization Finally,"
LOCATE 23, 6
INPUT "Press <RETURN> to continue ...", A$
CLS

LOCATE 2, 6
PRINT "You will be prompted for a training gain.  The
suggested value "
```

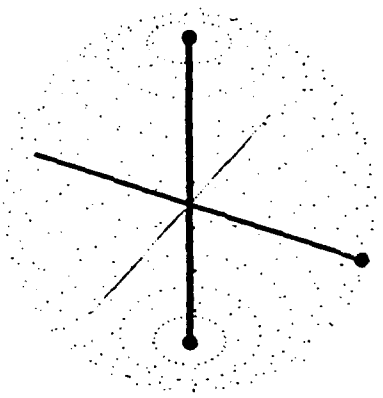
```
LOCATE 3, 6
PRINT "is 0.3, any number between 0.5 and 1 appears to work.
Larger"
LOCATE 4, 6
PRINT "values train faster, but the decision boundary may
oscillate"
LOCATE 5, 6
PRINT "around an optimal solution when input classes overlap.

Small "
LOCATE 6, 6
PRINT "values train slower, but the decision boundary will
transition"
LOCATE 7, 6
PRINT "smoothly to a fairly stable solution.  The program and
output"
LOCATE 8, 6
PRINT "have been structured such that the operation and
results are"
LOCATE 9, 6
PRINT "readily apparent.  For example, at the end of the
training"
LOCATE 10, 6
PRINT "phase the operator is alerted to the fact that the
weights"
LOCATE 11, 6
PRINT "have been frozen and that the net is operating in the
test"
LOCATE 12, 6
PRINT "mode.    All relevant data is then displayed and
updated "
LOCATE 13, 6
PRINT "including the ratio of correct decisions made by the
net"
LOCATE 15, 6
PRINT "The following page describes some of the options
available"
LOCATE 16, 6
PRINT "to the operator"
LOCATE 23, 6
INPUT "Press <RETURN> to continue ... ", A$
CLS
```

```
LOCATE 2, 4
PRINT "OPTIONS"
LOCATE 4, 6
PRINT "There are several options available.  As previously
mentioned,"
LOCATE 5, 6
PRINT "you may opt to continue training or testing after the
specified"
LOCATE 6, 6
PRINT "number of iterations has been reached.  If you opt to
continue,"
LOCATE 7, 6
PRINT "the net will start at the current iteration count and
continue "
LOCATE 8, 6
PRINT "until another set of the specified iterations is
complete. You may"
LOCATE 9, 6
PRINT "again opt to continue operation.  Once you quit
operating, you may"
LOCATE 10, 6
PRINT "opt to re initializw and train or test another net
under different"
LOCATE 11, 6
PRINT "parameters,if you choose not to train another net, the
program terminates."
LOCATE 12, 6
PRINT "One other useful option is  provided. This program is
meant to"
LOCATE 13, 6
PRINT "stand alone and does not contain any printer routines.
However, if"
LOCATE 14, 6
PRINT "you have the EGA screen dump utility 'EGADMP', you can
load it into"
LOCATE 15, 6
PRINT "memory first (it is a TSR) and then run PERCEPT.  Any
time the "
LOCATE 16, 6
PRINT "simulator waiting for a response to a prompt, you can
print out a"
```

```
      LOCATE 17, 6
      PRINT "screen dump to an EPSON FX,MX compatible printer.
      Depress "
      LOCATE 18, 6
      PRINT "<SHIFT-PrtSc> for medium resolution (110 dpi) or
      <SHIFT-PrtSc-h> for"
      LOCATE 19, 6
      PRINT "high resolution (240 dpi).  After the printer
      finishes, answer the"
      LOCATE 20, 6
      PRINT "prompt and the program will continue."
      LOCATE 23, 6
      INPUT "Press <RETURN> start the simulation...", A$
      CLS
   END IF
END SUB

SUB readin (q,DATAFILE, vect(2), class(1), n, trainfile$,testfile$)

       OPEN trainfile$ FOR INPUT AS #1
         FOR I = 1 TO n
           INPUT #1, vect(I, 1), vect(I, 2)
           INPUT #1, class(I)
         NEXT I
       CLOSE #1
END SUB

SUB readin2(q,DATAFILE,vect(2),class(1),o,trainfile$,testfile$)
        OPEN testfile$ FOR INPUT AS #3
          FOR I = 1 TO o
            INPUT #3, vect(I,1), vect(I,2)
            INPUT #3, class(I)
          NEXT I
       CLOSE#3
END SUB
```

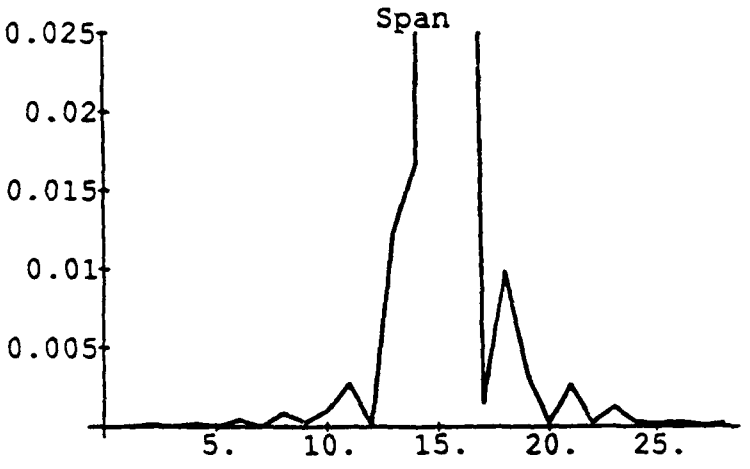
Appendix C. *Mathematica Output*

Experiment 1/2 Flatplate 17.88cm

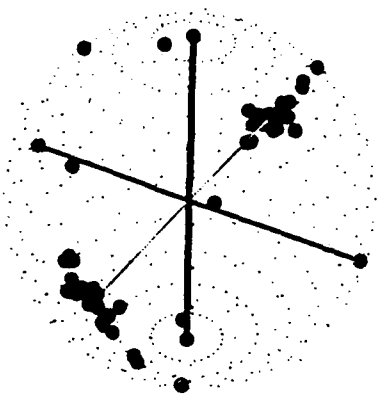


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

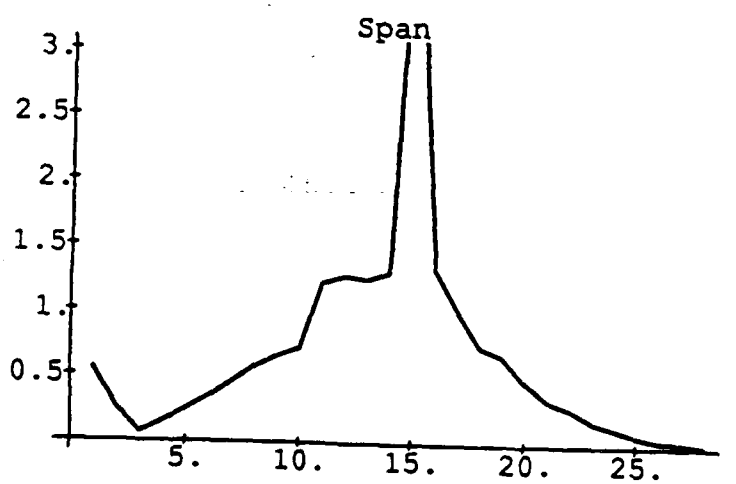


Experiment 1/2 Dihedral 8.944cm Flatplate 17.88cm

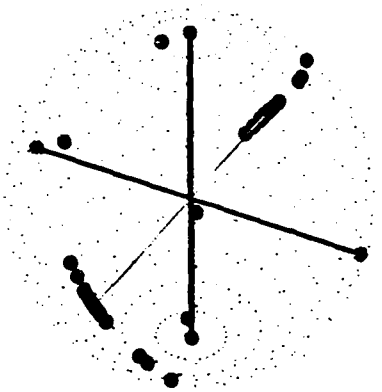


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

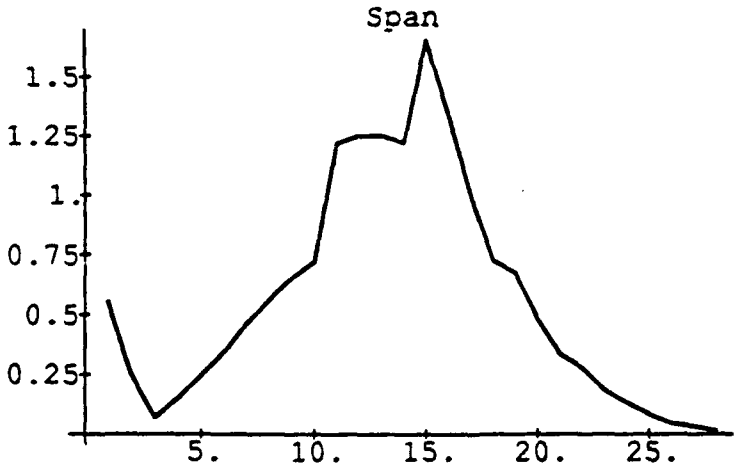


Experiment 1/2 Dihedral 8.944 cm

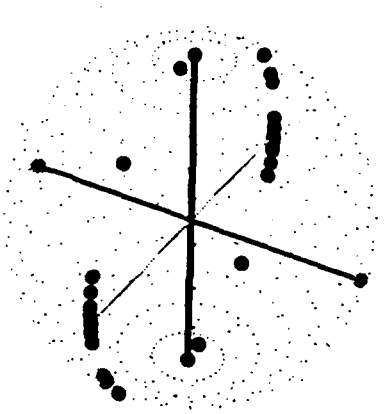


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

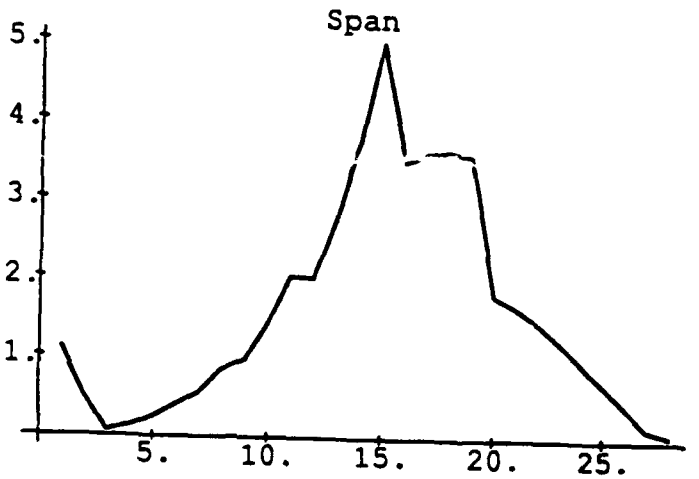


Experiment 1/3 Dihedral 8.944 cm

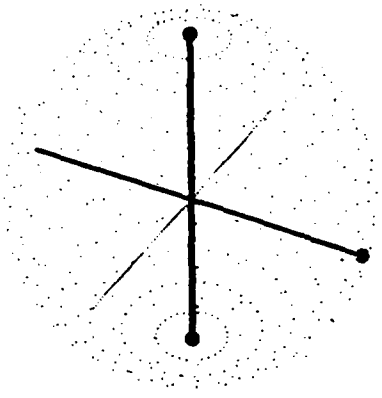


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

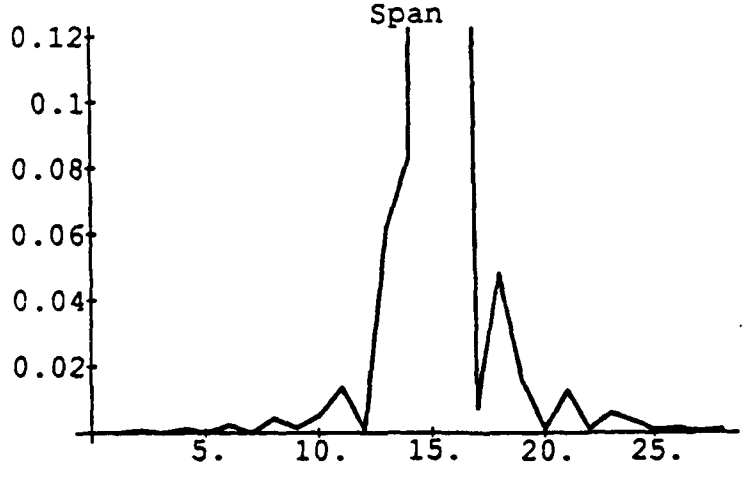


Experiment 1/3 Flatplate 17.88cm



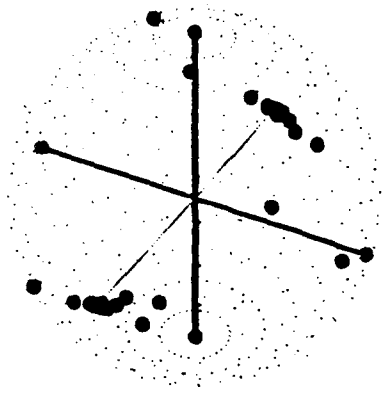
```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```



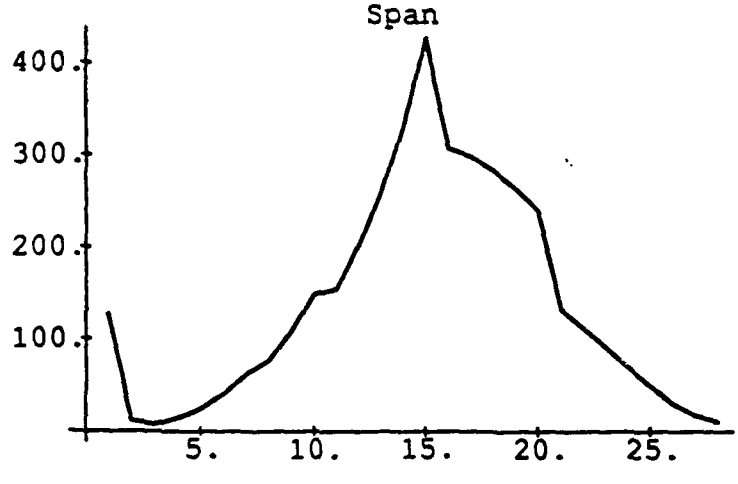

```
-Graphics-
```

Experiment 3b Dihedral 53.664cm Flatplate 37.888cm

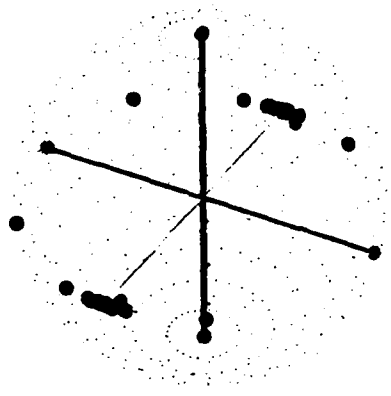


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

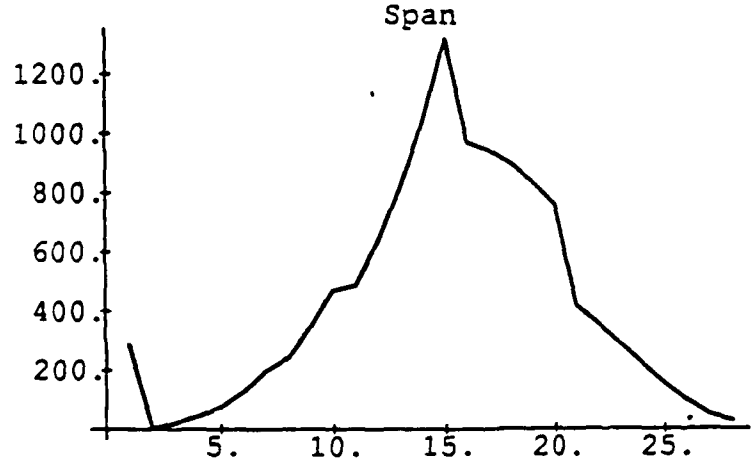


Experiment 3c Dihedral 71.552cm Flatplate 37.88cm

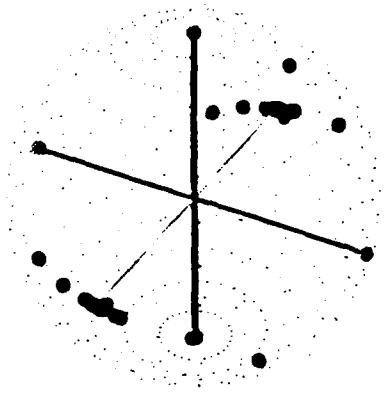


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

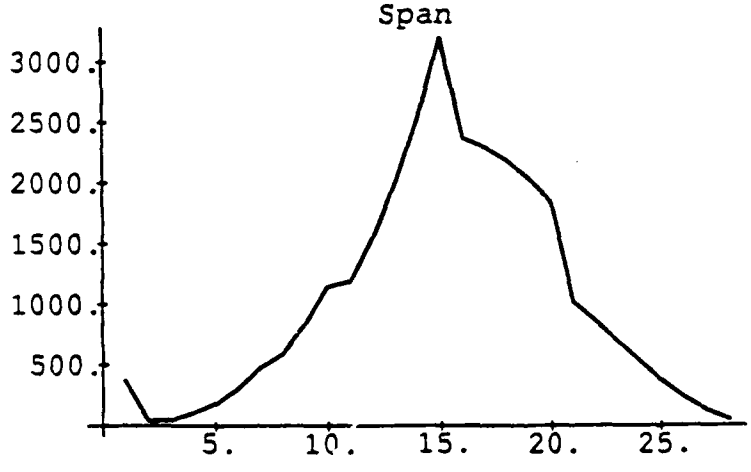


Experiment 3d Dihedral 89.44cm Flatplate 37.888cm

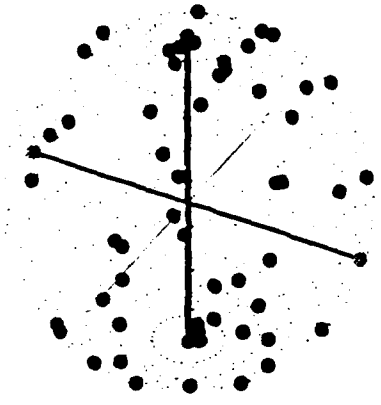


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

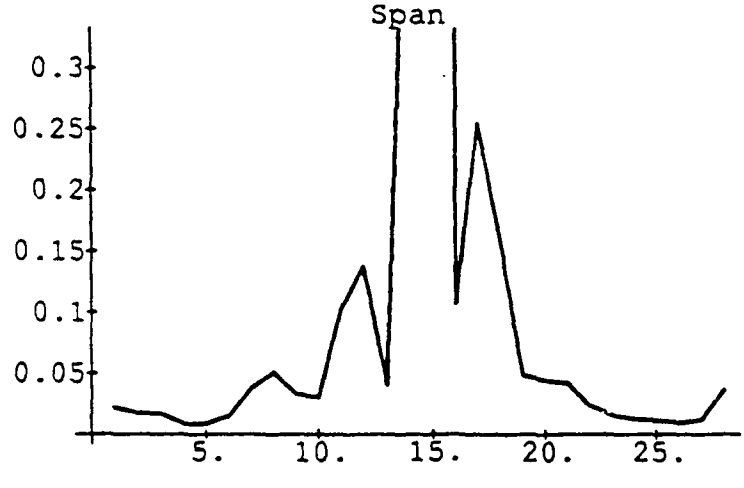


Experiment 3f Dihedral 5.963cm Flatplate 37.888cm

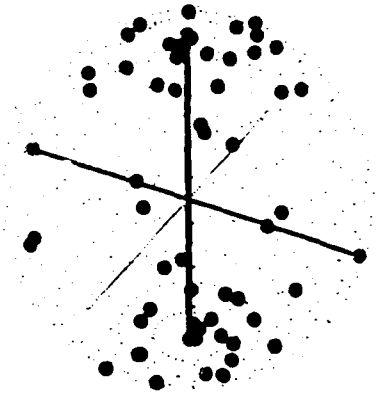


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

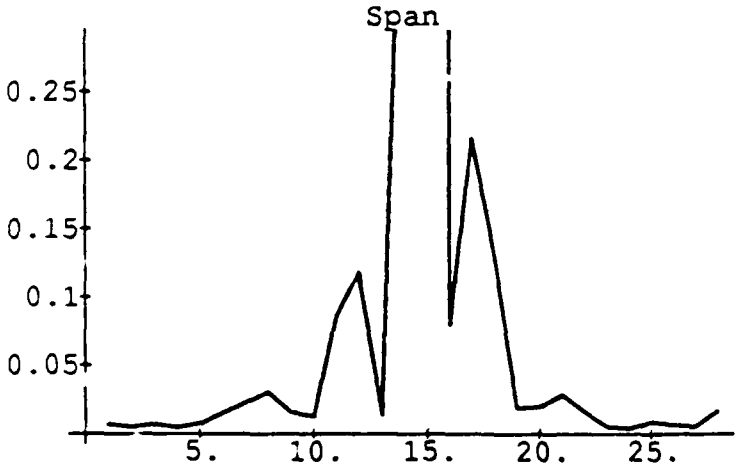


Experiment 3g Dihedral 4.472cm Flatplate 37.88cm

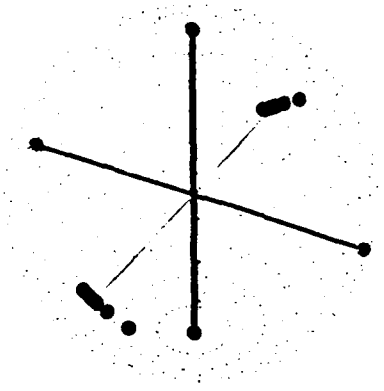


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

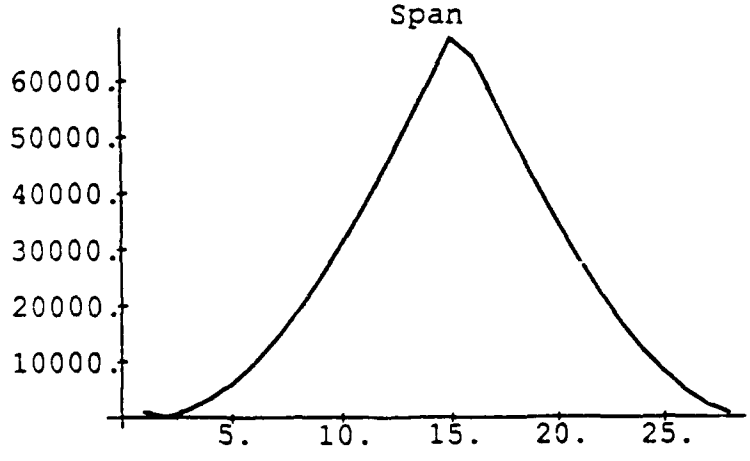


Experiment 27 Dihedral 110cm

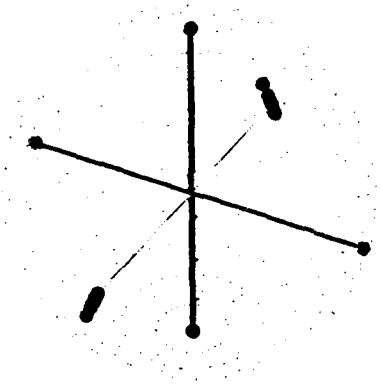


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

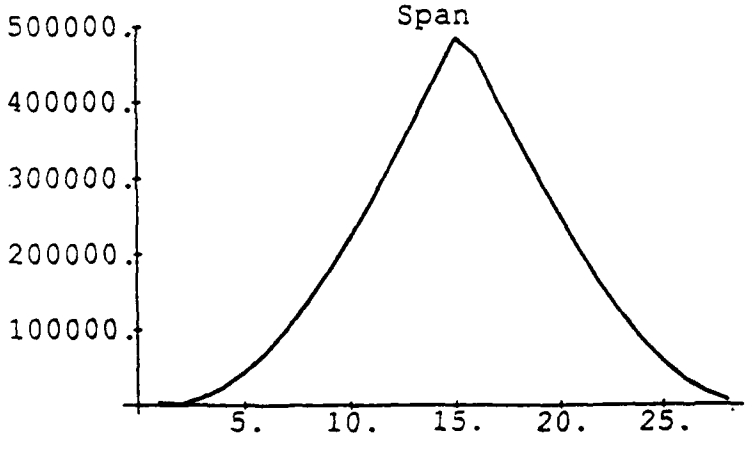


Experiment 41 Dihedral 180cm



```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```



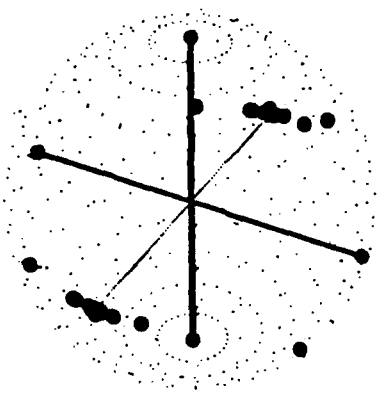
```
copole2Long=Table[If[Re[q2c[[i]]]==0,
                        If[Im[q2c[[i]]]==0,
                              N[ArcTan[-1.0],5],
                             -Sign[Im[q2c[[i]]]]*N[Pi/2,5]],
                     N[ArcTan[Re[q2c[[i]]],-Im[q2c[[i]]]] ],5 ]
               ,{i,pulses}];
```

The following section plots the co and cross polarizations on the Poincare Sphere.

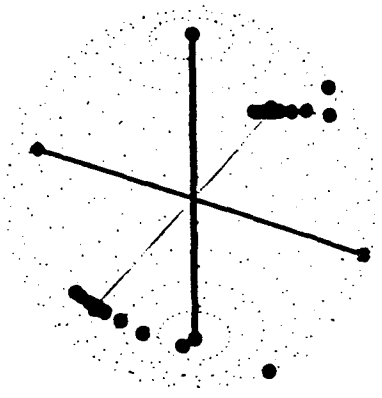
```
copoleList=Table[
                   {copole1Lat[[i]], copole1Long[[i]]}
             ,{i,pulses}];
```

The following is the sphere graphic

```
test=Graphics3D[sphere];
Show[test, Boxed -> False];
```

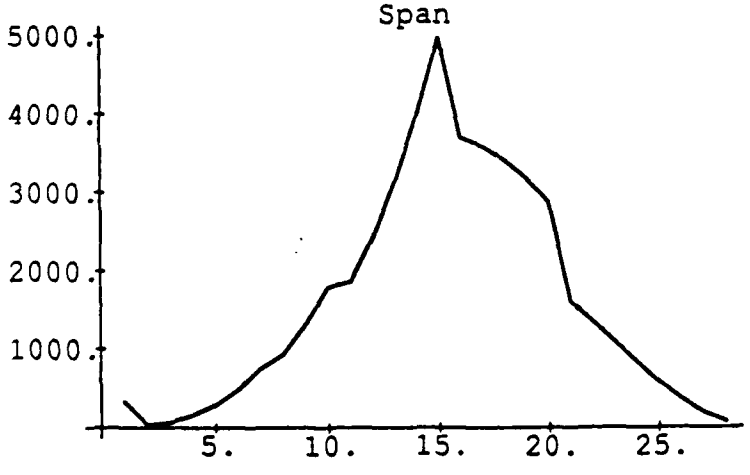


Experiment 91 Dihedral 100cm Flatplate 20cm

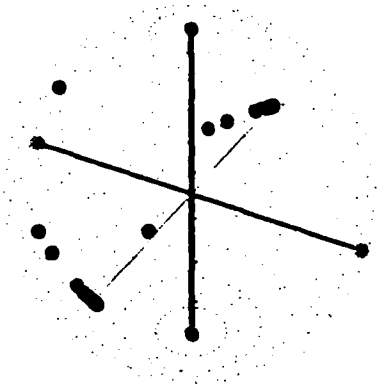


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

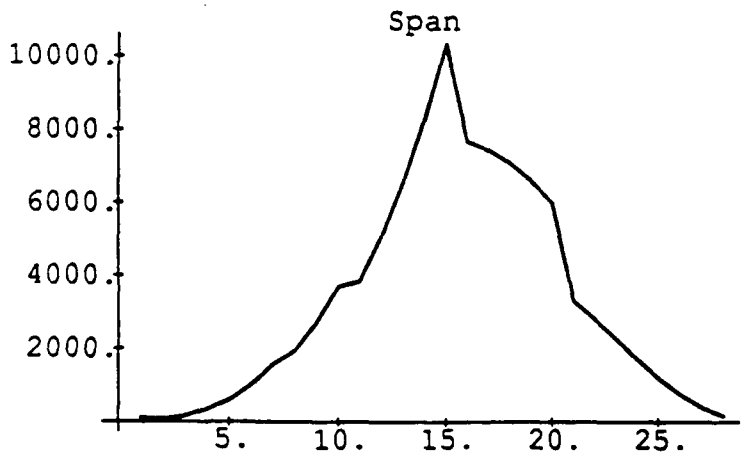


Experiment 95 Dihedral 120cm Flatplate 10cm

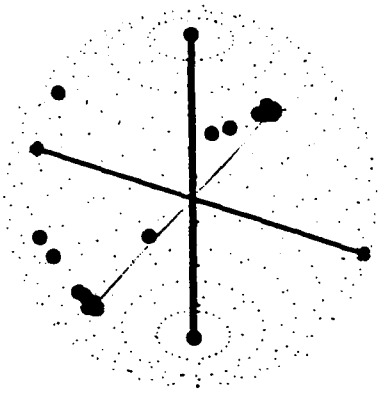


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

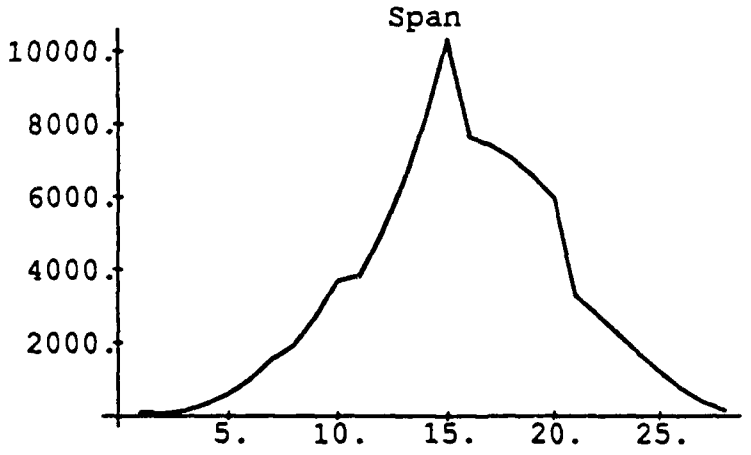


Experiment 99 Dihedral 120cm Flatplate 80cm

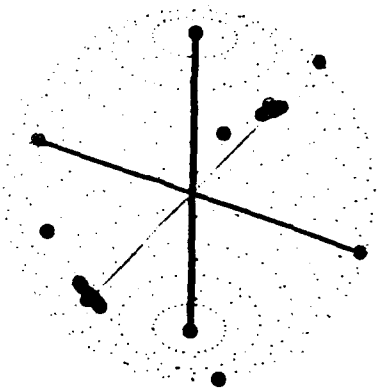


```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```

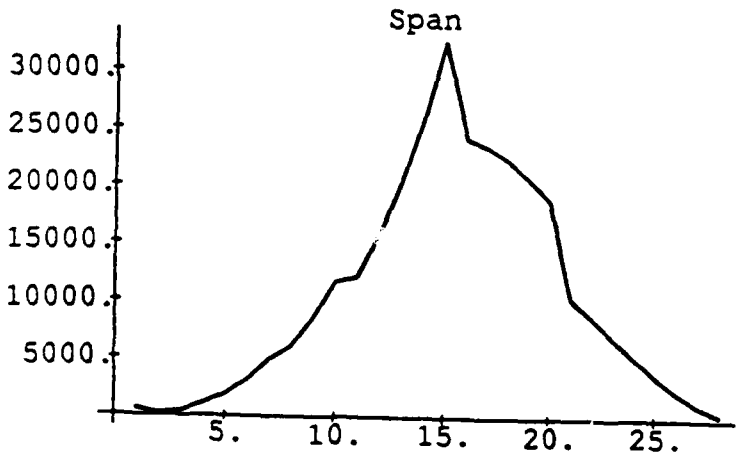


Experiment 103 Dihedral 160cm Flatplate 60cm



```
span = Table[(Abs[saa[[i]]]^2 + Abs[sba[[i]]]^2 +Abs[sab[[i]]]^2 + Abs[sbb[[i]]]
```

```
ListPlot[span,PlotJoined -> True, PlotLabel->"Span"]
```



Bibliography

1. M. I. Skolnik, *Radar Handbook.* McGraw-Hill Book Company, 1970.
2. D. Guili, "Historical View of Polarimetric Technology," *IEEE Antennas and Propagation Transactions*, pp. 42–50, Febuary 1986.
3. J. R. Huynen, 1970. Ph.D Dissertation.
4. J. L. Eaves and E. K. Reedy, *Principle of Modern Radar*, vol. 1. Van Nostrand Reinhold Company, 1987.
5. D. Meer, April 1989. Advanced Radar Class Notes, Air Force Institute of Technology.
6. J. Difranco and W. Rubin, *Radar Detection*, vol. 1. Artech House Inc, 1980.
7. A. B. Kostinski and W. Boerner, "On Foundations of Radar Polarimetry," *IEEE Antennas and Propagation Transactions*, pp. 1395–1403, December 1986.
8. J. J. V. Zyl *et al.*, "Imaging Radar Polarization Signatures: Theory and Observation," *IEEE Transcations*, p. 500, August 1987.
9. J. J. V. Zyl *et al.*, "Imaging Radar Polarization Signatures: Theory and Observation," *IEEE Transcations*, pp. 529–543, August 1987.
10. E. M. Kennaugh, "Polarization Dependence of RCS - A Geometric Interpretation," *IEEE Antennas and Propagation Transactions*, pp. 412–413, March 1981.
11. M. M. Rackson, "High Resolution Polarimetric Radar Precision Limits," *IEEE Antennas and Propagation Transactions*, pp. 8–10, 1984.
12. B. Beker *et al.*, "Antenna Polarization and Scattering Matrix Measurements in Mixed Polarization Bases," *IEEE Antennas and Propagation Transactions*, pp. 116–121, 1983.
13. S. Polikarpov, "A Method For the Static Measurement of the Absolute Scattering Matrix of Radar Targets," *Radiotekhnika*, pp. 61–64, 1984.
14. R. M. Narayanan, "Radar Backscatter of Trees at 215 GHZ," *IEEE Antennas and Propagation Transactions*, pp. 217–229, May 1988.
15. J. Fath, A. Terzuoli, and E. Zelnio, "Full Polarimetric Scattering from a Dihedral Corner Reflector," *Radar Handbook*, pp. 514–517, May 1987.
16. A. Terzuoli, July 1988. Class Notes, Air Force Institute of Technology.
17. T. Griesser and C. A.Balanis, "Backscatter analysis of dihedral corner reflectors using physical optics and the theory of diffraction," *IEEE Antennas and Propagation Transactions*, pp. 1137–1147, October 1987.

18. P. Corona and Others, "Backscattering by Loaded and Unloaded Dihedral Corners," *IEEE Antennas and Propagation Transactions*, pp. 1148–1153, October 1987.

19. H. A. Zebker and L. Norikane, "Radar Polarimetric Measures Orientation of Calibration Corner Reflectors," *IEEE Antennas and Propagation Transactions*, pp. 1680–1687, December 1987.

20. D. R. Wehner, *High Resolution Radar*, vol. 1. Artech House, 1985.

21. W. M. Boerner *et al.*, "Polarization Dependence in Electromagnetic Inverse Problems," *IEEE Antennas and Propagation Transactions*, pp. 262–271, March 1981.

22. E. M. Kennaugh, "Effects of Type of Polarization on Echo Characteristics Monostatic Case," September 1949.

23. R. P. Lippman, "An Introduction To Computing With Neural Networks," *IEEE Antennas and Propagation Transactions*, pp. 4–22, April 1987.

24. R. Rosenblatt, *Principles of Neurodynamics*. Spartan Books, 1959.

25. G. L. Tarr, 1988. Master's Thesis.

26. S. Zabele *et al.*, 1989. TASC Technical Information Memorandum.

Vita

Squadron Leader Alan P. Callaghan [redacted] After completing secondary education he joined the Royal Australian Air Force in 1970 as a Radio Technician. After completing Technician training he applied for aircrew training and was commissioned as an Air Electronics Officer in 1974. After a tour on Anti Submarine Warfare (ASW) aircraft and several staff postings Squadron Leader Callaghan was posted to Civil Schooling in 1986 where he finished his undergraduate science degree in Physics. Squadron Leader Callaghan was then posted to an Electronic Warfare position before entering the Air Force Institute of Technology in 1988.

[redacted]

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION / AVAILABILITY OF REPORT
2b. DECLASSIFICATION / DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S) AFIT/GE/ENG/89D-58	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION School of Engineering 6b. OFFICE SYMBOL (*If applicable*)	7a. NAME OF MONITORING ORGANIZATION
6c. ADDRESS (*City, State, and ZIP Code*) Air Force Institute of Technology Wright - Patterson AFB OH 45433-6583	7b. ADDRESS (*City, State, and ZIP Code*)
8a. NAME OF FUNDING / SPONSORING ORGANIZATION ASD -Ed Zelnio 8b. OFFICE SYMBOL (*If applicable*)	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS (*City, State, and ZIP Code*) Wright Patterson AFB OH 45433-6583	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11. TITLE (*Include Security Classification*)
Target Classification Using Synthetic Aperture Radar Polarimetric Data

12. PERSONAL AUTHOR(S)
Alan P. Callaghan, BAPPSC, RAAF

13a. TYPE OF REPORT	13b. TIME COVERED FROM _____ TO _____	14. DATE OF REPORT (*Year, Month, Day*)	15. PAGE COUNT

16. SUPPLEMENTARY NOTATION

17. COSATI CODES			18. SUBJECT TERMS (*Continue on reverse if necessary and identify by block number*)
FIELD	GROUP	SUB-GROUP	Polarimetric Radar, Synthetic Aperture Radar, Target Recognition, Artificial Networks.

19. ABSTRACT (*Continue on reverse if necessary and identify by block number*)

This study investigates the polarimetric behaviour of canonicals within single resolution cells. The aim of this investigation is to ascertain whether the contents of a resolution cell can be determined by examination of the polarimetric data. The canonicals addressed in this thesis effort are the dihedral and the flatplate, although the associated theory and software is readily applicable to other primitives.

The main software package used during this thesis is SarTool, although additional ancillary software is presented such that the output of SarTool can be represented on the Poincare sphere.

Finally, both single level and multi-level perceptron networks are used to process the resolution cell output and return a decision on the contents of the cell. This preliminary study of the polarimetric and perceptron combination provides some valuable insight into the possible potential of polarimetrics and Artificial Intelligence in the area of target identification.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Andrew Terzuoli, PhD	22b. TELEPHONE (*Include Area Code*) 513 255 9267 / 22c. OFFICE SYMBOL AFIT/ENG

DD Form 1473, JUN 86 *Previous editions are obsolete* SECURITY CLASSIFICATION OF THIS PAGE