CV-580 Polarimetric SAR data processing software
A user's guide to COASP and CHASP

Terry Potter

The scientific or technical validity of this Contract Report is entirely the responsibility of the Contractor and the contents do not necessarily have the approval or endorsement of Defence R&D Canada.

This work was completed in July 2007.
CV-580 Polarimetric SAR data processing software

A user's guide to COASP and CHASP

Terry Potter

Prepared By:
Terry M. Potter Consulting
206 Chipmunk St., Limoges, Ont. K0A 2M0

Contract Number: W7714-0207-16/001/SV
Contract Scientific Authority: Chen Liu, 613-993-8381

The scientific or technical validity of this Contract Report is entirely the responsibility of the Contractor and the contents do not necessarily have the approval or endorsement of Defence R&D Canada.

This work was completed in July 2007.

Defence R&D Canada – Ottawa

Contract Report
DRDC Ottawa CR 2007-188
August 2007
Principal Author

Original signed by Chen Liu

Chen Liu
Scientific Authority

Approved by

Original signed by Paris Vachon

Paris Vachon
AH/ RAST

Approved for release by

Original signed by Pierre Lavoie

Pierre Lavoie
Chari/Document Review Panel

© Her Majesty the Queen in Right of Canada, as represented by the Minister of National Defence, 2007
© Sa Majesté la Reine (en droit du Canada), telle que représentée par le ministre de la Défense nationale, 2007
Abstract

For several years, DRDC Ottawa has been acquiring and processing airborne polarimetric radar imagery from the Environment Canada Convair-580 aircraft. This guide describes how to use the various software packages to take polarimetric signal data from an 8-mm mammoth tape and create useful end-products for further analysis. This process uses software written for DRDC and software originally written by CCRS. They are combined into a logical data flow and there are several shell scripts that speed up the processing. A directory structure is described to make the greatest use of these shell scripts.

The processing begins by describing how to get the signal data off the tape, and then separating the image data from the noise data and downloading 4 different polarization channels. The data are checked and the noise data evaluated. Other required data sources include inertial navigation data from the aircraft and a differentially processed GPS data set. These are processed and checked with quality control software. Then the main processor, COASP is run to process the noise and then produce an image over a calibration site. This is followed by collection of data from the available calibration devices, data quality assessment, and use of a program that produces the calibration parameters. All the calibration parameters are then verified. Once an image is produced, a program called CHASP is used for a detailed analysis of moving ocean targets. CHASP consists of two main software packages; there is a complete description of how to use them. All the processes are described in a logical order.
Résumé

Depuis plusieurs années, RDDC Ottawa acquiert et traite des images de radar polarimétrique aéroporté recueillies à bord de l’aéronef Convair-580 d’Environnement Canada. Le présent guide explique comment utiliser les divers progiciels qui servent à récupérer des données de signaux polarimétriques sur une bande Mammoth 8 mm et à créer des produits finals utiles aux fins d’analyses ultérieures. Ce procédé fait appel à des logiciels écrits pour RDDC et à des logiciels écrits initialement pour le CCT. Ils sont combinés en un flux de données logique, et plusieurs séquences de commandes en langage naturel accélèrent le traitement. Une structure de répertoires visant l’exploitation maximale de ces séquences est décrite.

Le traitement consiste d’abord à décrire comment les données des signaux seront récupérées sur la bande, puis à séparer les données d’image des données de bruit et à télécharger 4 différents canaux de polarisation. Les données sont vérifiées et les données de bruit sont évaluées. Les autres sources de données nécessaires comprennent les données de navigation par inertie de l’aéronef et un ensemble de données de GPS différentiel. Elles sont traitées et vérifiées à l’aide d’un logiciel de contrôle de la qualité. Puis le processeur principal, COASP, traite les données de bruit et produit une image sur un site d’étalonnage. Les étapes suivantes consistent à recueillir les données provenant des dispositifs d’étalonnage disponibles, à évaluer la qualité des données et à exécuter un programme qui génère les paramètres d’étalonnage. Ceux-ci sont ensuite vérifiés. Une fois l’image produite, un programme appelé CHASP est utilisé pour l’analyse détaillée des cibles océaniques mobiles. Ce programme se compose de deux progiciels principaux ; une description complète de leur utilisation est présentée. Tous les processus sont décrits dans un ordre logique.
CV-580 Polarimetric SAR Data Processing Software

A User’s Guide to COASP and CHASP

Version 1             July 16, 2007

*Terry Potter

*Terry M. Potter Consulting, under contract to DRDC Ottawa, Contract Serial No. W7714-0207-16/001/SV
Table of Contents

Table of Contents ..................................................................................................................... 1

Acknowledgements ................................................................................................................... 4

INTRODUCTION ...................................................................................................................... 5

DOWNLOADING THE SIGNAL DATA ..................................................................................... 6

sigdmp2/dispanc3/ezesigdmp ..................................................................................................... 6

    sigdmp2 .................................................................................................................................. 6
    dispanc3 ................................................................................................................................. 8
    ezesigdmp ............................................................................................................................ 9

Where the Image Ends & the Noise Begins ............................................................................. 10

CHECKING THE DATA ............................................................................................................. 12

Check Header Files, Ancillary Data, Fine Gain .......................................................................... 12

Verify Signal and Noise Files Sizes and Headers ..................................................................... 12

    ezeancdisp / ezeancendisp .................................................................................................... 12

    Check for Nadir Line ............................................................................................................. 14

    ancmat2 .............................................................................................................................. 15
    ancgain.m ............................................................................................................................. 17

NOISE EVALUATION ............................................................................................................... 18

    Noise Evaluation Using Selection.sh .................................................................................. 18

PROCESSING MOCOMP DATA ............................................................................................. 22

    Downloading & Checking MAID Data ................................................................................ 22

    InQC ..................................................................................................................................... 24

COASP PROCESSING ............................................................................................................. 26

    Running COASP ................................................................................................................ 26

    COASP Processing of the Noise ......................................................................................... 28

        Editing chasp1n.conf ....................................................................................................... 29
        Processing the Noise Results. .......................................................................................... 31
        ezechaspnoise .................................................................................................................. 33
The Noise is Bad! What do I do? ................................................................. 34
Creating Noise Images for Recording and Verification ................................. 34

PROCESSING FILES TO DETERMINE CALIBRATION PARAMETERS .............. 38
Editing chasp1a.conf .................................................................................. 38
Finding the Calibration Parameters .............................................................. 42
Renaming / Creating Files ........................................................................... 42
xv_GASP_detect ......................................................................................... 42
ezechasprecal ............................................................................................... 42
Finding the Calibrator Coordinates, coordinates.dat .................................... 43
Targanal ........................................................................................................ 44
CALQC.sh .................................................................................................... 45
Checking the Focus ...................................................................................... 48
GenCal1_05.sh ............................................................................................ 48
ezeecalcopy ................................................................................................ 51
Calibration Incidence Angle ......................................................................... 51
Getting Intermediate Cal Files into a Spreadsheet ....................................... 51
maketiff_chasp.m ........................................................................................ 52
Extracting the CalSite for an Image .............................................................. 53

PROCESSING AND CHECKING THE CALIBRATED IMAGE ....................... 57
Editing Chasp1c.conf .................................................................................. 57
The Calibration Check ................................................................................ 60
The Phase Check ......................................................................................... 61
Balance.sh .................................................................................................. 63
CHASPPQC.sh ............................................................................................ 65
PRINTOUTS .............................................................................................. 68

OTHER UTILITIES .................................................................................... 69
TransposeCoasp and transposeChasp .......................................................... 69
Extraction Programs ................................................................................... 69
Acknowledgements

The author would like to thank Marina Dragošević for her thorough review of the first draft of this document and for many helpful suggestions, Paris Vachon for his help in editing and preparing the document for publication, and Chen Liu for her suggestions.
**Introduction**

This guide is a “living document” that will document the procedures to use the software developed for the COASP and CHASP processors at DRDC Ottawa. This is Version 1 and is a User’s Guide for the versions of the software in use at DRDC as of March 31, 2007. The software, hardware and associated procedures will continue to evolve with time. In the future at logical times, there will be a Version 2, Version 3 etc. For a complete description of the algorithms and software see reference 1. Several programs are required to make the entire process work. In order to improve data flow and minimize errors, a given directory structure is created and several shell scripts have been written that utilize this structure.

The software works on a UNIX platform. There is also a Linux version. The computer displays shown are from SGI IRIX or Linux. There could be small variations in the display or commands depending on which flavour of UNIX is used. Another requirement is MATLAB. Several of the functions are MATLAB programs. The UNIX shell scripts are written in a TC shell. The Linux shell scripts used with CHASP are written in a BASH shell. The scripts must be pointed to the correct shell if the command pointing to the shell is not already present.

COASP was originally known as CHASP1 and that is why the name appears in some commands. What’s now known just as CHASP was originally known as CHASP2. That name can also appear in commands. At the time of this document, CHASP only works on a Linux machine.

In the guide, unless otherwise indicated, all line and pass numbers, device names and file names are examples only. You would substitute your own numbers.

Most files of data will have the following in their names. “I” line number, “p” pass number, and polarization channel. An example is l2p3hh. The noise files have the same format except that the channel is followed by “n”. An example is l2p4hvn. All data files will have this type of prefix. They can have one of several extensions. The main ones are:

- .sig signal data
- .anc ancillary data
- .rc COASP image file
- .hdr header file
- .mat MATLAB matrix file (MATLAB is a trademark of the MathWorks, Inc.)
- .chasp CHASP image file

Five input data sources are required to fully process a given flight line of polarimetric SAR data:

- Primary Base Station GPS data
- Secondary Base Station GPS data
- Aircraft GPS data
- Aircraft MAID data
- SAR raw signal data

The primary GPS base station position is determined by referencing it to a known survey monument. From this, a GPS solution for the position of the secondary base station is determined. The aircraft GPS data is differentially corrected with concurrent data from the two GPS base stations. The GPS file must be in ASCII or MATLAB (version 4) matrix format. There are several software packages capable of producing a GPS ASCII file. The author uses Waypoint Consulting Inc.’s GrafNav/GrafNet software.

In this report, all the computer output is displayed in blue. All user input in this guide will be in bold type. Some of the computer displays are quite lengthy. They have been truncated at each end, with the words “AND CONTINUING ON” in between. Important passages are in bold or are preceded by “*NOTE*.”.
**Downloading the Signal Data**

**sigdmp2/dispanc3/ezesigdmp**

The first step in processing is to copy the signal data recorded to the workstation. On a 5 GB 8-mm tape, it is typically one line/pass (lb#p#) with four channels of image data and four channels of noise data. On a mammoth tape or DLT tape, there will be several lines and passes. Each line/pass will have 4 channels of signal data. Each file of signal data consists of image signal data followed by noise signal data. By convention, the channels are in the following order: HH, HV, VH, VV. The signal data are copied from tape to the workstation by the program **sigdmp2**. Ancillary data are also bundled in with the image and noise and are separated out by sigdmp2. It also creates header files. This program can be used to copy all the signal data from tape to disk. Optionally, a shell script, **ezesigdump** can be used alone or in combination with sigdmp2. Using the shell script will save a lot of time and effort.

*NOTE*: There will be times when you will not be able to tell where the image ends and the noise begins. This can be due to a number of reasons such as the radar operator forgetting to write down or incorrectly recording the "transmitter off time" or problems in stripping the data that result in an incomplete line or a “Julian Day Error”.

There is a solution as long as one complete channel (image + noise) can be transferred to a hard disk. See the section "**Where the Image Ends & the Noise Begins**". The solution requires knowledge of some other programs that are described in this User's Guide.

**********************************************************************************

**sigdmp2**

The following is a description of how to use sigdmp2. This also includes instructions on using the program **dispanc3**. The blue text is what will appear on the workstation monitor. The bold text on the left hand side is what you should key in.

**sigdmp2**

The following will appear:

```
+---------------------------------------------------------------------------------------+
|                                                                                       |
|                                  SIGDMP                                              |
|                              (Version 2.0)                                             |
|       IRIS Video Signal Data Tape-To-Disk Transcription System  |
|                                                                                    |
+---------------------------------------------------------------------------------------+
```

```
<<<<< Input Device >>>>>
```

1. /dev/rmt/tps3d4v.8200 (dirac)
2. /dev/rmt/tps3d4v.8500 (dirac)
3. /dev/rmt/tps3d6v.8200 (dirac)
4. /dev/rmt/tps3d6v.8500 (dirac)
5. other Selection:
Tape drive address (this is an example only).

Where \( l\#p\#h \) represents the line number and pass number for the first channel (HH). If you are downloading a noise channel, you would enter \( l\#p\#h n \)

4

Represents the total number of channels on the tape. On the 8-mm exabyte tapes, there will be no more than 4 channels. For the mammoth tapes, the total number will be the number of lines/passes multiplied by 4.

1

Corresponds to the desired channel in a given line/pass e.g.

- HH channel 1
- HV channel 2
- VH channel 3
- VV channel 4

The channels are in the order HH, HV, VH and VV for all lines/passes. The number given to a specific channel in a specific line/pass will depend on where it was recorded on the tape. For example, if there were 5 lines/passes on the tape and you wanted the HH channel of the fifth line/pass, the channel number would be 17. (4 channels \( \times \) 4 line/passes = 16 + 1 channel for the fifth line/pass).

Channels 1 - 4 could be channels 5 - 8, 9 - 12 etc., depending on how many lines/passes are on the tape.

Skipping 1 files to get Video Signal File of Pass #1
Please wait...

<<<<<<<Dump Option>>>>>

1. Dump Raw CV580 Signal Data
2. Dump Ancillary Data
3. Dump both
Selection : 

3

Always select 3. The ancillary data will be needed for almost any processing. The screen should display the total number of records on the tape at this point. Write this down! (e.g. 118855)

Acquisition time of 1st signal record on tape = 06-JUN-2002 01:36:57.34400
Range line number of 1st signal record on tape = 1
HDDT record sequence # of 1st signal record = 1496103
Total number of signal records on tape = 118855
Number of `complex' samples per record = 4096

Enter start complex sample to read from tape :

1

First complex sample (range line)
Depends on the application. This can be any number <= 4096. It is usually 4096 but can often be 2048.

Always use time when dumping the first channel from a given line/pass. Then use "dispanc3" to find the number of lines or records equivalent to the desired stop time. For all subsequent channels use lines.

This comes from the time above displayed by the program.

This time is taken from the SAR RTD EVENT MARKER AND NOISE CALIBRATION sheet under heading of END OF SCENE AND TRANSMITTER OFF TIME for the correct line and pass number. At this point the data from the first channel, (HH) will be copied to the workstation. This process can take fifteen or twenty minutes for a 5 GB tape, < 5 min. for a mammoth tape.

After the first channel has been transferred to the workstation the program dispanc3 is run.

Enter the line and pass number and the screen will show how many records are in the file, (e.g. there are 60652 records in this file).

You have to read at least 1 record to make the program happy. Enter any number <= the maximum number of records on file (60652 - 1 = 60651).

At this point you should know where the image data starts and stops and where the noise data starts and stops (e.g., image starts at 1 and stops at 60652; the noise data starts at 60653 and stops at 81228). 81227 is one (1) record smaller than the maximum size. This is necessary since each channel may not be exactly the same size. The program will generate errors if you ask it to read beyond the end of a file.

To continue on with sigdmp2, the image and noise signal dumps will use lines instead of time. The above steps are repeated for each data channel and each noise channel.

The file name is changed each time sigdmp2 is run. e.g. l3p1hnn channel 1 noise
l3p1hv  channel 2
l3p1hvn channel 2 noise
l3p1vh  channel 3
l3p1vhn channel 3 noise
l3p1vv  channel 4
l3p1vvn channel 4 noise

*NOTE*: When dumping noise files, always use lines, never time. Time results in an error.
Life can be made easier by using \texttt{ezesigdmp}. Once you perform sigdmp2 manually, or you have performed the “\textit{Where the Image Ends & the Noise Begins}” procedure, you will have all the parameters you need for \texttt{ezesigdmp}. It is a shell script that will handle the rest of copying the signal data from tape to disk so you can do something else.

Key in \texttt{ezesigdmp} and you will receive several prompts. Answer the prompts with the numbers you have already recorded. You will then be prompted to check your numbers. After that, hit return and the downloading process will start. You will see this. User entries are bold.

\texttt{pigpen 153\% ezesigdmp}
\texttt{CHECK EACH NUMBER CAREFULLY BEFORE PRESSING ENTER!!}
\texttt{********************************}
Enter Line Number:  
1
Enter Path Number:  
12
Enter Number of Files on Tape:  
24
Enter First File Number to Extract:  
5
Enter First Image Line Number:  
1
Enter Last Image Line Number:  
134099
Enter First Noise Line Number:  
134100
Enter Last Noise Line Number:  
151455
How many Range Lines to Process? (e.g. 2048, 4096)  
4096
Do you want to dump l1p12hh? (y/n)  
n
CHECK YOUR NUMBERS. Press enter to start SIGDMP2

\textbf{Do you want to dump l1p12hh? (y/n)} If “transmitter off time” was available, you would have already dumped the HH signal data using time, so the answer is “n”. If you used the image /noise procedure, you would delete the resulting files and answer “y”.

Check numbers & hit return.

Ezesigdump will take care of the rest. Go have a coffee.

There will be times, especially if you are running the shell script from a remote terminal, that you will lose the connection to the workstation, or things could otherwise crash. If \texttt{ezesigdmp} has gotten a start, you can quickly finish downloading. The script produces the following files:

\begin{tabular}{ll}
\texttt{sighh} & \texttt{sighn} \\
\texttt{sighv} & \texttt{sighvn} \\
\texttt{sigvh} & \texttt{sigvhn} \\
\texttt{sigvv} & \texttt{sigvvn} \\
\end{tabular}
Check to see which image and noise files have been downloaded. For any file that is incomplete or hasn’t been downloaded at all, use the following command:

```
sigdmp2 < sigfile  (e.g. sigdmp2 < sighvn)
```

Repeat as many times as necessary. Each download process is automatic.

**Where the Image Ends & the Noise Begins**

There are various methods of manipulating data, but if one entire channel of an image can be downloaded to hard disk, this procedure will work.

Download an entire channel using sigdmp2 and name it \#p\#h\#n. Both the signal and ancillary data must be dumped. It can be done by line or by time. The channel name doesn’t matter, but it must be followed by “n” which labels it as a noise file. 4096 samples can be used, but it is not essential. 2048 will work just as well.

Use **ancmat2** (see **ancmat2**) to convert the ancillary data to MATLAB matrix (.mat) format.

Use the script **ezecrop** to extract the profile of a single azimuth line in the centre of the file.

```
ezecrop space line# space pass# space (number of range lines/2) e.g. ezecrop 2 3 1024
```

Open MATLAB and use the program **lookatnoise_nofile.m** to view the image and noise plot. Follow the program prompts.

The noise has a distinct signature and can easily be distinguished in the last third of the image & noise plot.
Use the MATLAB Figure Window magnifying feature to find the number of the last image line. Keep using the magnifier until the line numbers resolve enough to be able to distinguish the single number at the end of the image.

If the entire channel cannot be downloaded, dump approximately the last half of the channel and follow the above procedure. The line number of the last image line must be added to the line number where the downloading started.

Although the example is real, it is somewhat idealized. You may not always see such a distinctive end to the image.
CHECKING THE DATA

Check Header Files, Ancillary Data, Fine Gain

Verify Signal and Noise Files Sizes and Headers

The signal and noise header files must be checked. Look over each header file taking care to note the transmit and receive channels for each one. The transmit and receive channels must match the names of the header files. If the transmit and receive channels do not match the file names, it usually means one of two things: There is a labeling error or a sigdmp error. Unless you know there is a labeling error, try to sigdmp the offending channels again. If that doesn't work, it probably is a labeling error. This means you must edit the header files. A shell script takes the work out of displaying each file individually with command lines. The shell script is ezehdrchk. The command is ezehdrchk 1 2 where 1 represents the line number and 2 represents the pass number. The HH header file will be displayed and each successive header will be displayed by pressing enter. COASP will check for inconsistencies during processing. This will cause it to log the error and then exit. Checking now is much easier.

*NOTE*: This next step is not necessary if you used ezesigdmp

Next do ls -l. Check carefully to make sure that the image signal files are all the same size. The noise files should also be identical, but this is not crucial. Ideally, they should not differ from each other by more than one record.

ezeancdisp / ezeancendisp

Other data to check are the first and last records of the ancillary signal (not noise) data files. 2 shell scripts have automated this otherwise tedious process. This is an example. For the first record, the program is ezeancdisp 1 6 where 1 represents the line number and 6 represents the pass number. The program brings you to a text-editing window. Four channels of ancillary data will be there. The file will look like this, but with different numbers:

**1p6 HH Channel**

Enter ancillary data file (ie fname.anc) :
There are 95653 records in this file

Enter first record to display : Enter number records to display :

*** Ancillary Parameters ***

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>24-SEP-2002 19:31: 2.50600</td>
</tr>
<tr>
<td>Rangeline number</td>
<td>1</td>
</tr>
<tr>
<td>HDDT record number</td>
<td>1283725</td>
</tr>
<tr>
<td>Range gate delay</td>
<td>46.58</td>
</tr>
<tr>
<td>Latitude (deg)</td>
<td>45.26</td>
</tr>
<tr>
<td>Longitude (deg)</td>
<td>-76.03</td>
</tr>
<tr>
<td>Groundspeed (m/s)</td>
<td>143.13</td>
</tr>
<tr>
<td>Track angle (deg)</td>
<td>72.84</td>
</tr>
<tr>
<td>Heading (deg)</td>
<td>73.49</td>
</tr>
<tr>
<td>Pitch angle (deg)</td>
<td>0.85</td>
</tr>
<tr>
<td>Roll angle (deg)</td>
<td>-0.62</td>
</tr>
<tr>
<td>Inertial altitude (m)</td>
<td>0.00</td>
</tr>
<tr>
<td>Antenna pointing angle</td>
<td>269.40</td>
</tr>
<tr>
<td>Antenna depression angle</td>
<td>-29.44</td>
</tr>
<tr>
<td>Antenna elevation angle</td>
<td>29.40</td>
</tr>
<tr>
<td>RF fine gain chan A</td>
<td>20.00</td>
</tr>
<tr>
<td>RF fine gain chan B</td>
<td>22.00</td>
</tr>
<tr>
<td>AD saturation count ch A</td>
<td>399.50</td>
</tr>
<tr>
<td>AD saturation count ch B</td>
<td>60.00</td>
</tr>
<tr>
<td>PRF/V (Hz/m/s)</td>
<td>2.32</td>
</tr>
<tr>
<td>PRF (Hz)</td>
<td>332</td>
</tr>
<tr>
<td>Altitude (ft)</td>
<td>0</td>
</tr>
<tr>
<td>Clutterlock offset (deg)</td>
<td>-0.46</td>
</tr>
<tr>
<td>Ch A Coarse Attenuation</td>
<td>6.00</td>
</tr>
<tr>
<td>Ch B Coarse Attenuation</td>
<td>12.00</td>
</tr>
<tr>
<td>Velocity correction</td>
<td>0</td>
</tr>
</tbody>
</table>
Enter ancillary data file (ie fname.anc) :
There are 95653 records in this file

Enter first record to display : Enter number records to display :

*** Ancillary Parameters ***

Time = 24-SEP-2002 19:31: 2.50800
Rangeline number = 1  HDDT record number = 1283725
RGD select code = 46.58  Range gate delay = 46.58
Latitude (deg) = 45.26  Longitude (deg) = -76.03
Groundspeed (m/s) = 143.13  Track angle (deg) = 72.84
Heading (deg) = 73.49  Pitch angle (deg) = 0.85
Roll angle (deg) = -0.62  Inertial altitude (m) = 0.00
Antenna pointing angle = 269.40  Antenna depression angle = -29.44
Antenna airframe angle = 399.50  Antenna elevation angle = 29.40
RF fine gain chan A = 20.00  RF fine gain chan B = 22.00
AD saturation count ch A = 60.00  AD saturation count ch B = 60.00
PRF/V (Hz/m/s) = 2.32  PRF (Hz) = 332
Altitude (ft) = 0  Clutterlock offset (deg) = 0.00
Ch A Coarse Attenuation = 6.00  Ch B Coarse Attenuation = 12.00
Velocity correction = 0  Yaw angle (deg) = 0

Enter ancillary data file (ie fname.anc) :
There are 95653 records in this file

Enter first record to display : Enter number records to display :

*** Ancillary Parameters ***

Time = 24-SEP-2002 19:31: 2.50800
Rangeline number = 2  HDDT record number = 1283726
RGD select code = 46.58  Range gate delay = 46.58
Latitude (deg) = 45.26  Longitude (deg) = -76.03
Groundspeed (m/s) = 143.13  Track angle (deg) = 72.84
Heading (deg) = 73.49  Pitch angle (deg) = 0.85
Roll angle (deg) = -0.62  Inertial altitude (m) = 0.00
Antenna pointing angle = 268.94  Antenna depression angle = -29.44
Antenna airframe angle = 399.50  Antenna elevation angle = 29.40
RF fine gain chan A = 20.00  RF fine gain chan B = 22.00
AD saturation count ch A = 60.00  AD saturation count ch B = 60.00
PRF/V (Hz/m/s) = 2.32  PRF (Hz) = 332
Altitude (ft) = 0  Clutterlock offset (deg) = 0.00
Ch A Coarse Attenuation = 6.00  Ch B Coarse Attenuation = 12.00
Velocity correction = 0  Yaw angle (deg) = 0
llp6 VV Channel

Enter ancillary data file (ie fname.anc) :
There are 95653 records in this file

Enter first record to display  : Enter number records to display :

*** Ancillary Parameters ***

Time = 24-SEP-2002 19:31: 2.50800
Rangeline number          =        2     HDDT record number       =  1283726
Latitude (deg)            =    45.26     Longitude (deg)           =   -76.03
Groundspeed (m/s)         =   143.13     Track angle (deg)              =    72.84
Heading (deg)             =    73.49     Pitch angle (deg)          =     0.85
Roll angle (deg)           =    -0.62     Inertial altitude (m)      =     0.00
Antenna pointing angle    =   268.94     Antenna depression angle  =   -29.44
Antenna airframe angle     =   399.50     Antenna elevation angle   =    29.40
RF fine gain chan A       =    20.00     RF fine gain chan B       =    22.00
AD saturation count ch A  =    60.00     AD saturation count ch B  =    60.00
PRF/V (Hz/m/s)             =     2.32     PRF (Hz)                 =      332
Altitude (ft)             =        0     Clutterlock offset (deg)  =     0.00
Ch A Coarse Attenuation  =     6.00     Ch B Coarse Attenuation   =    12.00
Velocity correction       =        0     Yaw angle (deg)            =        0

Go through the file and delete the 3 lines of information before each record (Enter…., There…., Enter….).
First, record the number of records in that file.  Save the edited file and exit.  The file name will be "llp#ancdat.txt".  For the last record, the script is ezeancendisp 1 6 95653.  I represents the line number and 6 represents the pass number, and 95653 is the number of records in the file that you recorded when editing llp#ancdat.txt.  Do the same editing as before.  The file name is "llp#ancendat.txt".  The files can be used for a quick check of the ancillary data.

Check for Nadir line

The majority of the time, it will be unnecessary to find the nadir line, since COASP will automatically check for it.  You are given the choice of checking when you edit the configuration (.conf) files before running COASP.  COASP will also compute the nadir line.  It does this geometrically with the GPS position of the antenna and the height above the ellipsoid, which is entered into the configuration file.  It is verified by comparing it with the slant range of the first range bin.

*NOTE*: If you already know that this is a ‘SAW OUT’ image, make sure “auto_nadir” is set to Yes in the configuration file. In SAW OUT images, there is an unusable area of 384 range lines.  This is an artifact of the range compression filter and cannot be removed.  COASP will start processing the first valid range line. This will be reflected in the “start_sample” and “number_samples” in the header file.

There could be times that you want to set the nadir line manually.  For instance, you may want to process only the bottom half of the image.  Once you have determined the nadir line that you like, go to the configuration file (chasp1c.conf).  Set “auto_nadir” to No.  Put your new nadir line opposite “skip_nadir” and run the process again.  If you are not interested in any lines before nadir, leave “range_discard” set to Yes.

The other way to check for nadir is done with the shell script ezenadir with the command line ezenadir 1 2 where 1 is the line number and 2 is the path number.  Observe the resulting image.
*NOTE*: If the image is too small, grab a corner with the mouse and drag it out.

*NOTE*: If the image is too light or too dark to see detail, right click on the image to bring up the XV2 command window.

1. Select **windows** and drag down to **color editor**.
2. Move **intensity** or click on **histeq** to change contrast of image.

If no horizontal dark line (or space) exists at the top of the image, then the nadir line is zero.

If a nadir line is not zero, then place the cursor directly below the line and right and left click at the same time to get a line number, (click middle key on three-button mouse). **Write down the line number**. This will be needed for the configuration files. When you are finished, right click on the image. Click on “QUIT” in the XV2 command window.

*NOTE*: There may be cases where you need to see more than 100 x 100 lines. (e.g., If the nadir image is completely black or a solid gray level). If this is the case, use the shell script `ezesawoutnadir 1 2` where 1 is the line number and 2 is the path number. This will produce a subimage of 500 X 500 pixels. You could be observing a "SAW OUT" image. Make a note of this, because this will affect what you enter in the configuration file.

`ancmat2`

Converts signal ancillary data to MATLAB matrix format. This will have to be done to the noise files in order to run `ancgain.m`, use the “end of image, beginning of noise” procedure, and find out the times for the noise you process. This is an example.

```
ancmat2
```

```
| +-----------------------------------------------------------------*+ |
| | ANCMAT (v2.0) | |
| | SIGDMP Ancillary to MAT-file Conversion Utility | |
| +-----------------------------------------------------------------*+ |

Enter input ancillary file name (ie. .anc) : l3p9hnn.anc

There are 17180 records in this file

Enter first record to dump : 1
Enter last record to dump : 17180
### *** SIGDMP Ancillary Parameters Available for MAT-file Conversion ***

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Time since 01-Jan-1990 (y)</td>
</tr>
<tr>
<td>2</td>
<td>Status Bit Field (y)</td>
</tr>
<tr>
<td>3</td>
<td>HDDT Record Sequence # (y)</td>
</tr>
<tr>
<td>4</td>
<td>Rangeline Number (y)</td>
</tr>
<tr>
<td>5</td>
<td>Rangedelay Select Code (y)</td>
</tr>
<tr>
<td>6</td>
<td>Rangedelay (y)</td>
</tr>
<tr>
<td>7</td>
<td>Latitude (y)</td>
</tr>
<tr>
<td>8</td>
<td>Longitude (y)</td>
</tr>
<tr>
<td>9</td>
<td>Groundspeed (y)</td>
</tr>
<tr>
<td>10</td>
<td>Track Angle (y)</td>
</tr>
<tr>
<td>11</td>
<td>True Heading (y)</td>
</tr>
<tr>
<td>12</td>
<td>Pitch Angle (y)</td>
</tr>
<tr>
<td>13</td>
<td>Roll Angle (y)</td>
</tr>
<tr>
<td>14</td>
<td>Inertial Altitude (y)</td>
</tr>
<tr>
<td>15</td>
<td>Antenna Pointing Angle (y)</td>
</tr>
<tr>
<td>16</td>
<td>Antenna Depression Angle (y)</td>
</tr>
<tr>
<td>17</td>
<td>Antenna Airframe Angle (y)</td>
</tr>
<tr>
<td>18</td>
<td>Antenna Elevation Angle (y)</td>
</tr>
<tr>
<td>19</td>
<td>RF Fine Gain for Ch. A (y)</td>
</tr>
<tr>
<td>20</td>
<td>RF Fine Gain for Ch. B (y)</td>
</tr>
<tr>
<td>21</td>
<td>AD Saturation Count Ch. A (y)</td>
</tr>
<tr>
<td>22</td>
<td>AD Saturation Count Ch. B (y)</td>
</tr>
<tr>
<td>23</td>
<td>PRF/V (y)</td>
</tr>
<tr>
<td>24</td>
<td>PRF (y)</td>
</tr>
<tr>
<td>25</td>
<td>Altitude (y)</td>
</tr>
<tr>
<td>26</td>
<td>Clutterlock Offset (y)</td>
</tr>
<tr>
<td>27</td>
<td>Ch A Coarse Attenuation (y)</td>
</tr>
<tr>
<td>28</td>
<td>Ch B Coarse Attenuation (y)</td>
</tr>
</tbody>
</table>

Menu Selection (0 exits, 29 toggles all):

- **l3p9hnn.anc** Will produce the number of records in the file. Remember or write this number down.

  1. First record to dump

  17180. This is the total number of records in the file

  29. Changes all toggles to y

  0. Exits menu and starts conversion process.

There will eventually be a finished prompt, hit enter to continue.

Repeat for **l#p#hvn.anc**

- **l#p#vhn.anc**
- **l#p#vvn.anc**
ancgain.m

cd to line & path directory containing signal files.

Open MATLAB.

ancgain

Enter line and path at prompt. l#p#

Figure windows will flash by as postscript file is created.

The resulting file will be called "l#p#ancgain.ps".
NOISE EVALUATION

Noise Evaluation Using Selection.sh

Selection.sh looks at the entire noise data set in all 4 channels. The parameters measured are power, jump detection, equivalent number of looks (ENL), contrast and predictability. They are measured in both azimuth and range. The output is a series of plots and an ASCII file called “about.bite” that gives you a range of good noise samples in azimuth and in range. This is not always accurate. You must look at the plots to determine the best noise sample for processing.

This is the program of choice since it checks both azimuth and range. There is only one command to actually run the program.

Main command is: Selection.sh <dir> l#p# [<step>]

If you are in the signal directory the command is: Selection.sh . l#p# [<step>]

The period after .sh is important. There is a space before and after the period. Step is optional. It defaults to 4. A better value to use is 32. It speeds up the program. The program can take from a few to 60 minutes or more to run. The actual time is dependent on several parameters such as system load, bad data, size of noise file etc.

An example is:

Selection.sh . l3p5 32

OR

Selection.sh /pigpen4/tpotter/Asar303pro/l3p5 32

3 is the line number, 5 is the pass number, 32 is the step number.

What you will see is a variation of the following:

pigpen 1% Selection.sh . l1p1 32

STARTING /pigpen0/tpotter/local/bin/Marina_build/NoiseQC/bin/Selection.sh . l1p1 32 0
Raw data header in ./l1p1hnn.hdr.
Extracting hh signal noise statistics

STARTING /pigpen0/tpotter/local/bin/Marina_build/NoiseQC/bin/GoodBite1.sh . l1p1hnn 32 0
Raw data in ./l1p1hnn.sig.
Scanning and analysing ./l1p1hnn.sig in 597 chips, order 3.

statis -z -i ./l1p1hnn.sig -D 4096 19124 -d 4096 32 -l 3
changes -a A1p1hnn.var -j 39.631000 -t 39.631000
A1p1hnn - Jumps detected in power
changes -a A1p1hnn.look -j 0.096330 -t 0.096330
A1p1hnn - Jumps detected in ENL
changes -a A1p1hnn.kurt -j 0.1 -t 0.1
A1p1hnn - Jumps detected in contrast level
changes -a A1p1hnn.flat -j 0.1 -t 0.1
A1p1hnn - Jumps detected in covariance structure
changes -a A1p1hnn.pred -j 0.1 -t 0.1
A1p1hnn - Jumps detected in predictibility
FINISHED /pigpen0/tpotter/local/bin/Marina_build/NoiseQC/bin/GoodBite1.sh . l1p1hnn 32 0
AND CONTINUING ON

STARTING /pigpen0/tpotter/local/bin/Marina_build/NoiseQC/bin/ViewSelected.sh . l1p1 32 0
Selected azimuth: 10976 19072 8096
Selected range: 0 4096 4096
FINISHED /pigpen0/tpotter/local/bin/Marina_build/NoiseQC/bin/ViewSelected.sh . l1p1 32 0

All looks good in the selected interval - See plots by gnuplot l1p1.plt
from line 10976
Time = 22-MAR-2004 19:45:58.06700 ( 448832758.07 since 01.01.1990.0:0:0)
to line 19072
Time = 22-MAR-2004 19:46:22.63500 ( 448832782.63 since 01.01.1990.0:0:0)
from range 0
to range 4096
FINISHED /pigpen0/tpotter/local/bin/Marina_build/NoiseQC/bin/Selection.sh . l1p1 32 0

There will be several files generated by this program. Most can be removed after the plots have been
viewed and stored. The lines from where the plots come together to the end are usually your good noise
sample.

*NOTE*: The most important numbers are the ones on the noise plots such as contrast. The caption will
say "Azimuth (smooth segment from ##### to #####)". It will say the same for Range. These numbers
may not agree with about.bite. Only use the smooth segment number ranges. Below is the one
exception to this rule.

*NOTE*: Always check where the “jump detection” occurs. This is where the RF gain is applied to the
noise. All noise samples must be taken after the RF gain has been applied. Never before. There have
been instances of bad noise samples where the “smooth segment” detected by the program has been before
the RF gain.

The results of Selection.sh can be seen with the command: gnuplot l#p#.plt

*NOTE*: Before removing any files, execute this command: Gnu2PS.sh l#p#.plt

You then must hit return 20 times. Each time you hit return, it will convert one of the plots to postscript
and display the name of the plot that has been converted. It will generate a file named l#p#.plt.ps. If the
extraneous files are removed first, the postscript file will be empty.

The following commands can be executed to remove extraneous files:

```
rm *.det
rm *.int
rm *.info
rm All*
rm Sel*
```

Do not remove about.bite
Once you have a range of good lines, you must choose 4096 successive lines for your noise sample. Choose near the end of the smooth segment and you usually can’t go wrong.

CHASP1 doesn’t need to know the time of the first and last line, but it should be recorded. This is how you find it.
At this point you will have converted the noise signal files to matrix files. Open MATLAB and find the times of the beginning and end of the noise sample with the following MATLAB commands:

```matlab
>> load l1p1hhn.mat
>> whos
Name                      Size                    Bytes  Class
ad_saturation_a           1x19124               152992  double array
ad_saturation_b           1x19124               152992  double array
altitude                  1x19124               152992  double array
antenna_airframe          1x19124               152992  double array
antenna_depression        1x19124               152992  double array
antenna_elevation         1x19124               152992  double array
antenna_pointing          1x19124               152992  double array
chA_coarse_atten          1x19124               152992  double array
chB_coarse_atten          1x19124               152992  double array
clutterlock               1x19124               152992  double array
groundspeed               1x19124               152992  double array
hddtrec                   1x19124               152992  double array
heading                   1x19124               152992  double array
inertial_altitude         1x19124               152992  double array
latitude                  1x19124               152992  double array
longitude                  1x19124               152992  double array
pitch                     1x19124               152992  double array
prf                       1x19124               152992  double array
prf_over_v                1x19124               152992  double array
rangedelay                1x19124               152992  double array
rangeline                 1x19124               152992  double array
rfgaina                   1x19124               152992  double array
rfgainb                   1x19124               152992  double array
rngdelay_sel_code         1x19124               152992  double array
roll                      1x19124               152992  double array
status                    1x19124               152992  double array
time                      1x19124               152992  double array
track                     1x19124               152992  double array

Grand total is 535472 elements using 4283776 bytes

>> timecode(time(12000));
000001 22-Mar-2004 082-19:46:01.18700
>> timecode(time(16096));
```

In this example, 12000 and 16096 are the beginning and end lines of the noise sample. Record these.

Starting in the signal file directory, create a directory called “pro”. When doing this the first time, use the names proposed in this guide. There are shell scripts that will not work without these specific directory names. If you don’t like these names, they can always be renamed when you are finished. You will also have to edit the shell scripts if you want to use them. Use a consistent name. This makes sense if you want to write shell scripts for file manipulation.

```
mkdir pro

cp l#p#hh.anc pro

cd pro
```
PROCESSING MOCOMP DATA

Mocomp processing is a series of steps applying GPS and MAID data to determine aircraft and antenna track and motion. The end result is a matrix file called #p#prf.mat. Along the way, there are several plots to observe to check data quality. There are other methods for doing this, but this one is easiest and provides the best quality.

The GPS data is processed with a commercial software package called GrafNav. The end result is a large file in ASCII format.

Go to whatever directories or files that contain your GPS and MAID data, and template for the coordinates.dat file. Copy the following files to pro.

- GPS ASCII file
- file#.dat (MAID file)
- coordinates.dat

Downloading & Checking MAID Data

There are 2 MAID files for each pass. A long one and a short one. The long files are acquired during the pass and the short one, between passes. The long files are the ones that will be processed. For a given line/pass, count how many passes it is from the beginning of the flight. Then count up the equivalent number of long MAID files and choose the appropriate one. Use polmaid and the checking procedure below to make sure you have the right MAID file.

The MAID data will usually arrive on a 5 GB 8-mm tape. The UNIX file transfer command “dd” is used to copy the files from tape to disk. Create a directory with an appropriate name. Use the “mt” command to check the tape status and position. Be sure to add the flags “nr” (no rewind between operations) and “s” (swap bytes). There could be some variation in your device name, but otherwise, enter the commands as they appear following:

```
pigpen 4% mkdir MAID_Icebergs
pigpen 5% cd MAID_Icebergs/
pigpen 6% mt -f /dev/rmt/tps1d5nrsv.8500 status
   Controller: SCSI
   Device: EXABYTE: EXB-85058HE-00000112
   Status: 0x20266
   Drive type: 8mm(8500) cartridge
   Media : READY, write protected, at BOT

pigpen 7% dd if=/dev/rmt/tps1d5nrsv.8500 of=file01.dat bs=1024b
   0+0 records in
   0+0 records out

pigpen 8% dd if=/dev/rmt/tps1d5nrsv.8500 of=file02.dat bs=1024b
   0+91 records in
   0+91 records out

pigpen 9% dd if=/dev/rmt/tps1d5nrsv.8500 of=file03.dat bs=1024b
   0+2469 records in
   0+2469 records out

pigpen 10% dd if=/dev/rmt/tps1d5nrsv.8500 of=file04.dat bs=1024b
   0+12 records in
   0+12 records out

pigpen 11% dd if=/dev/rmt/tps1d5nrsv.8500 of=file05.dat bs=1024b
   0+2140 records in
   0+2140 records out
```
AND CONTINUING ON

```
pigpen 27% dd if=/dev/rmt/tps1d5nrsv.8500 of=file21.dat bs=1024b
  0+968 records in
  0+968 records out
pigpen 28% dd if=/dev/rmt/tps1d5nrsv.8500 of=file22.dat bs=1024b
  0+3 records in
  0+3 records out
pigpen 29% dd if=/dev/rmt/tps1d5nrsv.8500 of=file23.dat bs=1024b
  0+0 records in
  0+0 records out
pigpen 30% dd if=/dev/rmt/tps1d5nrsv.8500 of=file24.dat bs=1024b
  0+0 records in
  0+0 records out
pigpen 31% dd if=/dev/rmt/tps1d5nrsv.8500 of=file25.dat bs=1024b
  Read error: No space left on device
  0+0 records in
  0+0 records out
```

The MAID data starts with an empty file, then a short file and a long file. The long and short files continue alternately almost to the end. There are two empty files at the end of the MAID data. When you see the message "Read error: No space left on device", you know you are done.

Next, use the **polmaid** program to convert the MAID file from binary to MATLAB matrix format.

**polmaid file#.dat file#.mat**

Before proceeding further, you must make sure that the MAID file is the right one. Open MATLAB and load file#.mat.

```
>> load file12.mat
>> whos

    Name      Size              Bytes  Class          Attributes
    ______    _______           ______ _______          _______
    acmd      1x20378          163024 double array
    apos      1x20378          163024 double array
    desired_track_angle  1x1020  8160  double array
    epos      1x20378          163024 double array
    ivspeed   1x16303         130424 double array
    maid_f_gndspd  1x20378  163024 double array
    maid_f_heding  1x16303  130424 double array
    maid_f_pitch  1x32605  260840 double array
    maid_f_roll  1x32605  260840 double array
    maid_f_trkang  1x20378  163024 double array
    ppos      1x20378          163024 double array
    timeacmd  1x20378          163024 double array
    timeapos  1x20378          163024 double array
    timedta   1x1020           8160  double array
    timegndspd  1x20378  163024 double array
    timeheding  1x16303  130424 double array
    timeivspeed  1x16303  130424 double array
    timepitch  1x32605  260840 double array
    timeroll  1x32605  260840 double array
    timetrkang  1x20378  163024 double array

Grand total is 401452 elements using 3211616 bytes
```
Use the exact wording of the timecode command, including the semicolon. Otherwise, your answer will be incomprehensible. Compare the time values with the PC-MAID Recorder Start and Stop Times on the “SAR REAL TIME STATION SHEET”. There should be a close, but not an exact match. If the match is not close, you will have to process another MAID file.

**InQC**

Once you have the right MAID file, the next command is:

```
InQC -a GPS file -x ancillary data file -I MAID matrix file -o output matrix file -l time offset
```

```
InQC -a GPS.txt -x l#p#hh.anc -i file#.mat -o l#p#prf.mat -l 14
```

In the above command, the GPS data files must be ASCII format, and the ancillary files in binary format. The matrix format of the GPS and ancillary files can be used with different modifiers, but they must be opened and then saved in MATLAB version 4 format. Using matrix files the command is:

```
InQC -m GPS.mat -y l#p#hh.mat -i file#.mat -o l#p#prf.mat
```

There are several variations in the basic commands including provisions for byte order. They can be found by typing in the InQC command with no arguments.

```
localhost 2% InQC
Usage: InQC -a|m|M <gps_file> -x|X|y|Y <anc_file> -i|I <ins_maid_file> [-o|O <out_prf_file>]
(a to specify the input GPS Ashtech ascii file )
(M to specify the input GPS MAT binary file - native )
(-x to specify the input ANC binary structured file - native )
(-X to specify the input ANC binary structured file - swap bytes )
(-Y to specify the input ANC MAT binary file - native )
(-y to specify the input ANC MAT binary file - swap bytes )
(-I to specify the input INS MAT binary file - native )
(-o to specify the output qcPRF MAT binary file - native )
(-O to specify the output qcPRF MAT binary file - swap bytes )
```

Ignore the error message in this case.

Read InQC.log carefully, especially if the program finished with an error. If no PRF file has been written, you won’t be able to process the image.

Observe the output with the following commands:

```
ShowQC1.sh
ShowQC2.sh
```
You must hit enter after viewing each plot. The last two plots showing PRF and spacing take much longer to load than the rest of the plots. Be patient. After carefully observing the output, and recording anything bad or unusual, convert them into postscript files.

```
Gnu2PS.sh show1.plt
Gnu2PS.sh show2.plt
```

You must hit enter after the title of each plot is displayed. Wait until you see the title of each plot. You will end up with two files: `show1.plt.ps` and `show2.plt.ps`. Print these out using a colour printer.

Remove the “show” files:

```
rm show_*
```

There is one last step before processing. Go to where you keep the current copies of the COASP configuration files and the chirp file. Copy the following files to the `pro` directory:

```
chasp1n.conf
chasp1a.conf
chasp1c.conf
marina_2.tmp
```

You are now ready for COASP processing.
Different configuration files are used for different stages of the processing. The program COASP does not change. COASP is block oriented and will take the time needed to process the number of lines set in the block_lines parameter in the configuration files. The minimum size of the processing block is 4096 lines. It will not work with fewer lines. You can specify any smaller number of lines to be processed, but the time taken will be for 4096 lines. A common block size is 8192. COASP will keep on processing until it reaches the end of a block. In the header file, you will often see that more lines were processed than were asked for, because COASP will process until it comes to the end of the last block that was needed for the image. For larger files, how long COASP takes to run is mainly dependent on the number of azimuth lines it has to process and the system load. 20,000 lines can take an hour. Full images should be processed overnight since they will take several hours depending on their size.

If the required data to be processed is not at the beginning of the file, there will be a delay of several seconds to a few minutes before you see a response. The program has to work its way from the beginning of the file to the point where you want to start processing. This is the response you would see.

*pigpen 21% chasp1 -w chasp1n.conf*

memBytes 134217728

IPC status from /dev/kmem as of Tue Aug 2 15:59:58 2005

T ID KEY MODE OWNER GROUP CREATOR CGROUP CBYTES QNUM QBYTES LSPID LRPID STIME RTIME CTIME

Message Queues:

T ID KEY MODE OWNER GROUP CREATOR CGROUP NATTCH SEGSZ CPID LPID ATIME DTIME CTIME

Shared Memory:

m 1 0x53637445 --rw-r--r-- root sys root sys 1 48124 807 807 9:56:53 no-entry 9:56:53

T ID KEY MODE OWNER GROUP CREATOR CGROUP NSEMS OTIME CTIME

Semaphores:

IPC status from /dev/kmem as of Tue Aug 2 15:59:58 2005

T ID KEY MODE OWNER GROUP CREATOR CGROUP CBYTES QNUM QBYTES LSPID LRPID STIME RTIME CTIME

Message Queues:

T ID KEY MODE OWNER GROUP CREATOR CGROUP NATTCH SEGSZ CPID LPID ATIME DTIME CTIME

Shared Memory:

m 1 0x53637445 --rw-r--r-- root sys root sys 1 48124 807 807 9:56:53 no-entry 9:56:53

T ID KEY MODE OWNER GROUP CREATOR CGROUP NSEMS OTIME CTIME

Semaphores:

*NOTE*: If the program quickly returns a prompt without this display, it means something has gone wrong. Open the chasp1.log file. more chasp1.log. Look towards the end of the file. It will give you a good indication of what went wrong. Missing files are often the culprit. Correct the error and run COASP again.
If nothing major has gone wrong, a prompt will eventually appear after the above display. This means the processing has completed.

The chasp1.log file is a very complete record of all the operations carried out during CHASP1 processing. ezelochk is a shell script that only works on the chasp1.log file. It displays any Warnings that were generated during processing, any instances where the program failed to perform an operation and the final status which is whether the program completed successfully or completed with an error or failed. A successful process on a small file will appear something like this:

```
pigpen 28% ezelochk
pigpen 29%
```

If you are processing a large file or the file size is not an exact multiple of the block size (which it rarely will be), ezelochk can give you something like this:

```
pigpen 144% ezelochk
2005-08-10_17:12:58:[W] Warning, cannot read prf/v, prf file value will be applied.
2005-08-11_07:12:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:12:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:12:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:17:01:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:17:01:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:17:01:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:17:01:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:17:01:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:17:01:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:21:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:21:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:21:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:21:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:21:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:21:30:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
```

2005-08-11_07:26:02:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:26:02:[W] Warning from doIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:26:02:[W] Warning from endIQmeasure: Only part of the block could be used for the IQ statistics.
2005-08-11_07:26:38:[W] Warning from setMocomp: skipping georeferencing at the end of PRF.
2005-08-11_07:26:38:[W] Warning from setMocomp: skipping georeferencing at the end of PRF.
pigpen 145%

All the warnings are the result of COASP processing to the end of a block even when there is no more data. These warnings and failures can be ignored as long as the last line has “Program finished successfully”. If the program finished with an error or failed, you must find out why and take appropriate action.

**COASP Processing of the Noise**

Noise is processed first, since all subsequent processing requires the noise values. To process the noise, you must edit the configuration file “chasp1n.conf”. One of the parameters required is the mean PRF over V. PRF over V information is in the PRF file (l#p#prf.mat) created by InQC. It is a MATLAB matrix file, and PRF over V has a varying value, so a small MATLAB program called meanprfv.m was created to find the mean PRF over V value. The pro directory should contain the PRF file. In the pro directory, open MATLAB and use the following command.

```
>> meanprfv
Enter Line Number: 1
Enter Path Number: 5
```

And the answer is: 2.575506

>>

Record the answer.
Chasp1n.conf looks a lot like this:

```plaintext
; WorkOrder
; mission_id 334_l1p4_CoCoNaut
noise_data_file /pigpen4/tpotter/334_pro/l1p4
channels 4
chirp_file marina_2.tmp
prf_file
first_line 18001
last_line 22096
SAW OUT
geoid_over_ellipsoid 0.0
georef_grid_rg 0
doppler_grid_rg 50
satur_grid_rg 50
channel_align No
range_discard No
motion_compensation No
motion_restore No
azimuth_focus Yes
RangeDoppler Yes
ZD_frame Yes
azimuth_align No
antenna_gain No
calibration No
window_shape 2.8
full_bandwidth
relative_dc_offset 0.0
relative_v_offset 0.0
reference_range 1
spread_loss 0
auto_nadir No
skip_nadir 0
antenna_hp
antenna_vp
antenna_hs
antenna_vs
prf_over_v 2.575506
;
; System parameters
;
wavelength 0.0565646000
range_shift_hh 0
range_shift_hv 0.52
range_shift_vv 0.93
range_shift_vh 1.33
azimuth_shift_hh 0
azimuth_shift_hv 0
azimuth_shift_vv 0.5
azimuth_shift_vh 0.5
range_slant_spacing 4.0
saw_delay 13.289e-06
boresight_offset_hs -1.585
```
boresight_offset_vs        -1.585
boresight_offset_hp        -3.330
boresight_offset_vp        -3.330
antenna_ang_min            -73.0
antenna_ang_inc            1.0
azimuth_beamwidth          3.03
;
; Processor parameters
;
processors                 1
block_lines                 4096
parts                      4
channels_together          1
change_anc_endian          No
change_output_endian       No
interp_size                 11
interp_fractions           16
mocomp_type                 Flat

Most of the parameters are already set and should not be changed. The parameters that are highlighted are most likely to be changed, sometimes with almost every image.

Mission_id: include, as a minimum, asar number, line number, pass number, trial name.

Noise_data_file: This is the path to the noise signal data. At the end of the path add a slash with the l#p# prefix for that data (e.g. /l2p8).

First_line: The first line of the noise sample you have chosen after running Selection.sh and observing the results. Only the azimuth lines are chosen. The range is always 4096.

Last_line: The last line of the chosen noise sample. Be sure the noise sample is 4096 azimuth lines long.

SAW: you must indicate whether the image is SAW-IN or SAW-OUT with IN or OUT.

prf_over_v: The PRF file is not included with the noise parameters. The program only requires the mean PRF over V. The procedure for finding this is given above.

range_shift_hv, range_shift vv, range_shift_vh: These parameters will vary depending on the date that the image was acquired. If the image was acquired in September 2004 or later, then the parameters above are correct. If the image was acquired before then, the correct parameters are as follows:

    range_shift_hh: 0
    range_shift_hv: 0
    range_shift_vv: 1.25
    range_shift_vh: 1.25

change_anc_endian: This will be NO if the file was created and run on a platform of the same endianess. This will be YES if the file was created and run on machines with different endianess.

change_output_endian: As above, this will depend on where the files will be used and the application. Generally, if they will be used on a platform of the same endianess, the answer is NO. If they will be used on a platform of different endianess, the answer is YES.
Other parameters to note: the main chirp_file for processing CHASP1 imagery is \texttt{marina\_2.tmp}, but this can be changed. Make sure the block_lines are \texttt{4096}. If the number is less the program won’t run. If the number is more, you will end up with a lot of useless data that will make it impossible to analyze the noise. Leave the mocomp_type as \texttt{Flat}. This is the least accurate MOCOMP type and takes the least amount of processing time. You don’t need MOCOMP accuracy for noise. Processors is for machines that have more than one processor. The more processors you are able to use, the shorter the processing time. Be considerate of others, if you are using a multi-user machine.

\textbf{*NOTE*}: When you process data with COASP, the results of the processing will be found in the signal data directory (\texttt{l\#p\#??rc} and \texttt{l\#p\#??rc.hdr OR l\#p\#??nrc OR l\#p\#??nrc.hdr}). The PRF file and the chirp file should be in the same directory as the configuration (.conf) file, usually \texttt{pro}. If not you will have to include the full path name with the file name when you edit the configuration files.

You should still be in the \texttt{pro} directory. The next command will be:

\texttt{chasp1 –w chasp1n.conf}

This process won’t take too long for the noise. It will depend on the workstation load.

\textbf{Processing the Noise Results}

Sometime (20 to 30 minutes) after the command “\texttt{chasp1 –w chasp1n.conf}” has been entered, a simple system prompt will indicate its completion. While in the directory \texttt{pro}, create a sub-directory called \texttt{noise}. Again, this name is optional but the directory structure should remain the same. Do the following operations.

\texttt{cp chasp1n.conf noise} \hspace{1cm} \text{Copy chasp1n.conf to noise}

\texttt{mv chasp1.log noise} \hspace{1cm} \text{Move the CHASP1 log file to noise}

\texttt{cd ..} \hspace{1cm} \text{Move up to the signal file directory}

\texttt{mv *rc.hdr pro/noise directory} \hspace{1cm} \text{Move the header files created by CHASP1 to the noise directory}

\texttt{mv *.rc pro/noise} \hspace{1cm} \text{Move the data files created by CHASP1 to the noise directory}

\texttt{cd pro/noise} \hspace{1cm} \text{Move down to the noise directory}

\texttt{ls} \hspace{1cm} \text{Make sure all the correct files are there}

\texttt{ezelogchk} \hspace{1cm} \text{Make sure the processing has been successful}

Next, another shell script is used to display one channel of the noise. The HH channel is usually displayed, but any other channel can be. The script is called \texttt{ShowImg.sh}. The arguments are line number, pass number, channel and then processing format. An example is:

\texttt{ShowImg.sh l\#p\#hnn rc}

Look at the image. The noise should be an even gray level with a consistent texture. This is a smoothed image. The \texttt{raw} image will show a consistent grainy texture.
The bite noise has to be determined. It is taken as an average over the entire range of the noise image. A shell script called `PowerBite.sh` does this. The command is:

```
PowerBite.sh l#p#hh
```

This must be repeated for each channel. This is a tedious way of doing things, so another shell script was written called `ezebitenoise`.

The EZE Life - `ezebitenoise`

The arguments are line number and pass number (e.g. `ezebitenoise 1 4`). The results will appear similar to this:
pigpen 134% ezechitenoise 7 8

```bash
statis -C -i l7p8hvn.rc -D 4096 4096 -d 4096 1254 -o 2048 2047
l7p8hvn.rc power 2.1112e-03
Warning: there is no file called l7p8hvc.hdr.
Please insert the following line to the header before running Gencal
mean_S_noise_rf_gain 2.1112e-03
```

```bash
statis -C -i l7p8hvrc.rc -D 4096 4096 -d 4096 1254 -o 2048 2047
l7p8hvnc.rc power 1.2526e-03
Warning: there is no file called l7p8hvrc.hdr.
Please insert the following line to the header before running Gencal
mean_S_noise_rf_gain 1.2526e-03
```

```bash
statis -C -i l7p8vvn.rc -D 4096 4096 -d 4096 1254 -o 2048 2047
l7p8vvn.rc power 1.8923e-03
Warning: there is no file called l7p8vvc.hdr.
Please insert the following line to the header before running Gencal
mean_S_noise_rf_gain 1.8923e-03
```

```bash
statis -C -i l7p8vhn.rc -D 4096 4096 -d 4096 1254 -o 2048 2047
l7p8vhn.rc power 1.0693e-03
Warning: there is no file called l7p8vhc.hdr.
Please insert the following line to the header before running Gencal
mean_S_noise_rf_gain 1.0693e-03
```

pigpen 135%

The noise image power or **mean_S_noise_rf_gain** are the numbers you want. Record these numbers. They are needed for calibrated images. If the numbers do not have a suffix of **e-03, e-04** or even **e-05**, suspect that something is wrong. There is not a lot of difference in bite noise for a given trial, but there can be exceptions. You will notice that there are warnings because the script `PowerBite.sh` is looking for files that are not there. It has more than one function. These warnings can be ignored at this stage. The script also produces a text file of the bite noise for easy reference.

You will also notice that a lot of files have been added to the noise directory. These are ancillary files generated by the script. This is normal.

**ezechaspnoise**

A shell script has been created to make life easy after the COASP noise processing has been completed. It is called **ezechaspnoise**. It has 2 arguments, the line number and pass number. Instead of following the procedure above, just enter the command

**ezechaspnoise 7 8**

The script creates a **noise** directory and copies the configuration and log files to it. It then moves the .rc and header files from the signal directory into **noise**. Next, the script `ezechitenoise` is executed. This produces the noise text file, which is displayed so you can copy the noise values. Finally, it displays an image of the HH noise channel for a quick check.

It is recommended that you do the noise processing procedure step by step several times before using the shell script. Otherwise, when something goes wrong, you won’t know where to start troubleshooting.
The Noise is Bad! What do I do?

Noise files can have “bad data”. Bad data is any deviation from an even grey texture.

If the noise is not good, look at Selection.sh plots. That can tell you where good noise begins & ends in both range & azimuth. With chasp1n.conf, you can only choose the azimuth portion of the noise to process. The range will be 4096. If there is bad noise, do not run the script ezbibitenoise. Instead, determine the 1st & last line of the good noise. Selection.sh is very helpful, but do not choose the first and last lines of good noise before observing the noise image. You may have to visually choose the first and last good noise lines. Choose the largest continuous portion of good noise. Then use the script PowerBite.sh. The arguments are line number, pass number, channel, the 1st good noise line and the last good noise line. The command line is:

```
PowerBite.sh l#p#hh 1st good noise line last good noise line
```

e.g. PowerBite.sh l2p3hh 338 4096

Repeat this for all 4 channels. The results should be the same as “normal” good noise (suffixes of e-03, e-04 or e-05). It is just being averaged over a smaller area. Record these values. The best way is to edit a file called l#p#bitenoise.txt. The copy and paste the PowerBite results. It will also be consistent with the other noise directories.

Creating Noise Images for Recording and Verification

When the noise is processed, you only get to see a smoothed image of the HH channel of the noise. This procedure creates images of all the noise channels and displays them all in a JPEG image that you can print out. First you must convert the file names to PolGASP format. This is done with the shell script ezerc2polnoise. The files can be converted to COASP name format with ezepol2rcnoise. Then you use the shell script ezenoise. All these scripts have two arguments, the line number and the pass number. It will access another program, which converts the noise image files to MATLAB matrix files. A MATLAB program is then used to create the final JPEG image. This is an example.

```
pigpen 5% cd noise

pigpen 6% ls
l3p9bitenoise.txt l3p9hhnrc.hdr l3p9vhn.rc l3p9vvn.rc.chip
l3p9hbn.rc l3p9vhn.rc l3p9vvn.rc.chip l3p9vvnrc.info
l3p9hbn.rc.chip l3p9vhn.rc.chip l3p9vhn.rc.info l3p9vvn.rc.sta
l3p9hbn.rc.info l3p9vhn.rc.info l3p9vhnrc.hdr
l3p9hbn.rc.sta l3p9vhn.rc.sta l3p9vhnrc.hdr
l3p9hbn_x1.pgm l3p9vhnrc.hdr l3p9vvn.rc

pigpen 7% ezerc2polnoise 3 9

pigpen 8% ls
l3p9bitenoise.txt l3p9hhnpolgasp.img l3p9vhn.rc.chip l3p9vvnrc.info
l3p9hbn.rc.chip l3p9vhn.rc.chip l3p9vhn.rc.info l3p9vvn.rc.sta
l3p9hbn.rc.info l3p9vhn.rc.info l3p9vhn.rc.sta l3p9vvnrc.hdr
l3p9hbn.rc.sta l3p9vhn.rc.sta l3p9vhnrc.hdr l3p9vvnrc.img
l3p9hbn_x1.pgm l3p9vhnrc.hdr l3p9vhnrc.img
l3p9hbnpolgasp.hdr l3p9vhnrc.hdr l3p9vvnrc.img
```

34
pigpen 9% ezenoise 3 9
------------------------------------
dataTool
------------------------------------
Start time: Wed Jan 31 15:46:05 2007

Input file: l3p9hhnpolgasp.img
Output file: l3p9hhnoise.mat
l3p9hhnoise.hdr not found.
Number of Rows: 4096
Number of Cols: 4096
Sample bytes 8
128 x 4096
4096 x 4096
Tile size 128x4096
Number of tiles: 32
Number of tiles: 32
processing...
T0 ***
T1 ***
T2 ***
T3 ***
T4 ***
T5 ***
T6 ***
T7 ***
T8 ***
T9 ***
T10 ***
T11 ***

AND CONTINUING ON

T30 ***
T31 ***
------------------------------------

This process is repeated for all 4 channels. Then open MATLAB. The program is RCnoisyimage.m.
>> RCnoisyimage
Enter full date of noise image: June 07, 2006

Enter line: 3

Enter path: 9

*******************************************************
HH NOISE IMAGE CREATED
*******************************************************

*******************************************************
HV NOISE IMAGE CREATED
*******************************************************

*******************************************************
VH NOISE IMAGE CREATED
*******************************************************

*******************************************************
VV NOISE IMAGE CREATED
*******************************************************

pcmd =
lp -dlj3_c

dev =
-dljet3

Results can be found in l3p9noiseimage.jpg
>> quit
Then, you can remove the matrix files and their associated header files. Finally, convert the name format. This can be done any time after the noise is processed. If you are in a hurry to get the main processing done, leave it to the end.

*NOTE*: If you are processing a calsite, edit chasp1a.conf. Else skip this and go to Edit chasp1c.conf.
PROCESSING FILES TO DETERMINE CALIBRATION PARAMETERS

If there is a calibration site in the image to be processed, then an image of 20,000 lines is an ideal size. A small area of about 10,000 azimuth lines on either side of the calibration site centre should be initially processed, just for calibration. This is easy if you know where the calibration site is in the image. This is normally not the case. What is normally known are the geographic coordinates of the calibration site. If you know this, then the program *cpa* can give you the azimuth line of the calibration site. Latitude, longitude (west longitude is negative), altitude, and the PRF file are required. The command line format is

```
cpa -m prf_file -f latitude -l longitude -a altitude
```

An example result is:

```
pigpen 16% cpa -m l7p8prf.mat -f 49.0781 -l -125.7931 -a 20.365
Found CPA @ line 39510 distance 14473.945842 elevation angle 61.195535
```

Now you have all the information you need to edit the configuration file.

**Editing chasp1a.conf**

The file will look a lot like this:

```
; WorkOrder
;
mission_id    295_l7p8_Quest
input_data_file /pigpen5/tpotter/295test/l7p8/l7p8
channels      4
chirp_file    marina_2.tmp
prf_file      l7p8prf.mat
first_line    29510
last_line     49510
SAW
geoid_over_ellipsoid 43.1
; choose:
georef_grid_rg 3
doppler_grid_rg 50
satur_grid_rg 50
channel_align Yes
range_discard Yes
motion_compensation Yes
motion_restore Yes
azimuth_focus Yes
RangeDoppler Yes
ZD_frame Yes
azimuth_align Yes
antenna_gain No
calibration No
auto_nadir Yes
skip_nadir 0
window_shape 2.8
full_bandwidth
; do you wish to overwrite the PRF value for
prf_over_v
use_mean_prf_over_v No
relative_dc_offset 0
```
relative_v_offset 0
; radiometry:
reference_range 1
spread_loss 0
antenna_hp
antenna_vp
antenna_hs
antenna_vs
; calibration:
; from the sheet in dB
AvTx_power
; from GenCal in dB
K_prime_hh
K_prime_hv
K_prime_vv
K_prime_vh
; from GenCal in degrees
Phase_corr_hh
Phase_corr_hv
Phase_corr_vv
Phase_corr_vh
; power (from PowBite not in dB)
AvBITE_hh 2.1112e-03
AvBITE_hv 1.2526e-03
AvBITE_vv 1.8923e-03
AvBITE_vh 1.0693e-03
; ; System parameters
; wavelength 0.0565646000
range_shift_hh 0
range_shift_hv 0.52
range_shift_vv 0.93
range_shift_vh 1.33
azimuth_shift_hh 0
azimuth_shift_hv 0
azimuth_shift_vv 0.5
azimuth_shift_vh 0.5
range_slant_spacing 4.0
saw_delay 13.289e-06
boresight_offset_hs -1.585
boresight_offset_vs -1.585
boresight_offset_hp -3.330
boresight_offset_vp -3.330
antenna_ang_min -73.0
antenna_ang_inc 1.0
azimuth_beamwidth 3.03
CalConstant 19.0
; ; Processor parameters
; processors 1
block_lines 8192
parts 4
channels_together 1
change_anc_endian No
Most of the parameters are already set and should not be changed. The parameters that are highlighted will have to be changed with almost every image. Some of the changes have already been explained in "Editing chasp1n.conf". They will be repeated here for ease of reading.

**Mission_id:** include as a minimum asar number, line number, pass number, trial name.

**Input_data_file:** This is the path to the signal data. At the end of the path add a slash with the l#p# prefix for that data (e.g. /l2p8).

**prf_file:** This is the main mocomp file produced by InQC. **This must be included or the image will not process.** Every image has a different PRF file.

**First_line:** The first line of the calibration sample you have chosen. The procedure for choosing a calibration data set is described above.

**Last_line:** The last line of the chosen calibration sample.

**SAW:** you must indicate whether the image is SAW-IN or SAW-OUT with **IN** or **OUT**.

**geoid_over_ellipsoid:** This is the height above or below the ellipsoid in meters of the calibration site. If the site is below the ellipsoid, it must be prefixed with a minus sign.

**motion_restore:** This must be set to YES when doing calibration. Motion compensation is used in the processing, but the original image must be restored in order to run Targanal and do the subsequent calibration.

**power (from PowBite not in dB):** The next 4 entries are the bite noise values obtained by running **PowerBite.sh** or **ezebitenoise**.

**range_shift_hv, range_shift_vv, range_shift_vh:** These parameters will vary depending on the date that the image was acquired. If the image was acquired in September 2004 or later, then the parameters above are correct. If the image was acquired before then, the following parameters should be changed as follows:

- **range_shift_hh:** 0
- **range_shift_hv:** 0
- **range_shift_vv:** 1.25
- **range_shift_vh:** 1.25

**change_anc_endian:** This will be **NO** if the file was created and run on a platform of the same endianess. This will be **YES** if the file was created and run on machines with different endianess.

**change_output_endian:** As above, this will depend on where the files will be used and the application. Generally, if they will be used on a platform of the same endianess, the answer is **NO**. If they will be used on a platform of different endianess, the answer is **YES**.

**mocomp_type:** Use **Part** for this processing. It is more accurate than **Flat** but doesn’t have the time.
demand of Each. If you find you cannot get complete results from Targanal or GenCall_05.sh, then you may have to reprocess using Flat.

Other parameters to note: the main chirp_file for processing CHASP1 imagery is marina_2.tmp, but this can be changed. Make sure the block_lines are at least 4096. If the number is less the program won’t run. A “normal” number for block_lines is 8192. A different number can be chosen but it cannot be < 4096 and must be a multiple of 4096.

Run the calibration file.

chasp1 –w chasp1a.conf

When the processing completes, create a sub-directory called precal. Again, this name is optional but the directory structure should remain the same. Do the following operations.

cp chasp1a.conf precal Copy chasp1a.conf to precal

mv chasp1.log precal Move the CHASP1 log file to precal

cd .. Move up to the signal file directory

mv *.rc.hdr pro/precal Move the header files created by CHASP1 to the precal directory

mv *.rc pro/precal Move the data files created by CHASP1 to the precal directory

cd pro/precal Move down to the precal directory

ls Make sure all the correct files are there

ezelogchk Make sure the processing has been successful

There is one more step that should be done before the file names are converted. The parameter “mean_S_noise_rf_gain” must be added to the header files in order for the program GenCall_05.sh to be run. If that parameter is not there, GenCall_05.sh will kick you out. There is an easy way to do this. The following procedure will add the mean_S_noise_rf_gain parameter to the end of each header file.

cp *rc.hdr ../noise Copy header files to noise directory

cd ../noise Go to the noise directory

PowerBite.sh l#p#hh Use the PowerBite script and repeat for each channel

OR

ezebitenoise # # Use the ezebitenoise script which will automatically do all 4 channels. The 2 numbers are line number & pass number.

mv l#p??rc.hdr ../precal move the header files with the added parameter back to precal

If this fails, you can always edit the header files and manually add the parameter name and value.
**Finding the Calibration Parameters**

**Renaming / Creating Files**

In order to calibrate an image the program Targanal must be run. To avoid re-inventing the wheel, the program Targanal is used with COASP files to find the calibrators and determine their responses, among other things. In order to do this, certain parameters had to be added to the CHASP1 header file. All the CHASP1 file names have to be renamed to PolGASP format names (e.g. `mv l1p2hh.rc l1p2hhpolgasp.img`). This is time-consuming and tedious, so a shell script was created to do it automatically. It is called “ezerc2polname”. In order to convert PolGASP format names back to COASP names, there is another script called “ezepol2rcname”. Each script requires 2 arguments, the line number and pass number. An example is:

```
ezerc2polname 1 8
```

After doing this, use `ls` to check the file names and make sure they have been converted. Targanal also requires a header file known as `l#p#polgasp.hdr`. This header file is normally created by PolGASP processing. Targanal only requires certain parameters from this header file. In order to satisfy this requirement, a shell script was written to create the PolGASP header file with the parameters that Targanal needs. It is called “ezepolgasphdr”. As usual, the arguments are line number and pass number. An example is:

```
ezepolgasphdr 1 8
```

**xv_GASP_detect**

In order to calibrate an image, you will need to know the names and approximate locations of all the calibrators ahead of time.

In order to run Targanal, a special file of the calibrator coordinates on the image must be prepared. To do this you have to see the calibrators. Since the file names have been changed to PolGASP format, you can use a program called `xv_GASP_detect`. This will create a basic 8-bit high-contrast image in a single polarimetric channel. The HH channel is normally used. The high contrast is optimal for finding calibrators. The program `ShowImg.sh` could also be used, but `xv_GASP_detect` creates a better image. The command is followed by the name of the HH image file and followed by “y”. The “y” must be included or the program will not run. An example is shown here:

```
xv_GASP_detect l1p8hhpolgasp.img y
```

The program will produce a file called `l#p#hhpolgaspdet.xv`, or in the example `l1p8hhpolgaspdet.xv`. If you make an error or for some reason the program must be run again, you must remove any existing `.xv` file. The program will not run if a `.xv` file exists. Also if the `.img` file is > 2 GB, the program will not execute. This is another reason for the calibration image being kept small.

**ezechasprecal**

After the calibration file is processed by COASP, there are a lot of processing steps and a lot of potential for error. To make life a whole lot easier, `ezechasprecal` was created. It requires two arguments, the line number and pass number. It does everything except find the calibrator coordinates.

It starts by making a directory called precal and then copies chasp1a.conf, coordinates.dat template, chasp1.log and the “doppler” & “extreme” files into precal. It moves up one level to the signal data, retrieves the “.rc” and “.rc.hdr” files and moves them to precal. It then goes to precal, and copies the header files to the noise directory. It moves to the noise directory and executes the `ezebitenoise` script. Then the altered header files are moved back to the precal directory. `ezechasprecal` goes to the precal directory and executes the script `ezerc2polname` followed by `ezepolgasphdr`. Once this is done, `xv_GASP_detect` is run. This script takes a few minutes to complete.
As mentioned before, it is recommended that you do the calibration processing procedure step by step several times before using the shell script. Otherwise, when something goes wrong, you won't know where to start troubleshooting.

**Finding the Calibrator Coordinates, coordinates.dat**

This “.xv” file is compatible with the imaging utility XV2. Open XV2 and load this file. The image that first appears will have a large vertical range and a very small horizontal range. Grab the right edge of the image with the mouse and drag it out to double or triple its size. This makes viewing easier. You will need to know where to look for the calibration site. Hold down the Ctrl key and left click the mouse. This will zoom in on the area under the mouse pointer. A rectangular outline will appear. The rectangle can be moved as long as the mouse button is depressed. Move the outline to the desired position. Release the mouse button before you release the Ctrl key. You must release the keys in this order or the Zoom In will not work. To Zoom Out, hold down the Ctrl key and right click.

Zoom in on the calibration site until all the calibrators just fill the window. If the image looks bad and grainy, click on the Display button of the XV2 GUI (Right clicking toggles the XV GUI display). Then click on smooth. Move the mouse pointer on top of one of the calibrators. On a 3-button mouse, click the centre button. On a 2-button mouse click both buttons. A small box will appear in one corner of the image. It will have 5 numbers. The first 2 numbers are the coordinates of the calibrator. The first number is range, the second is azimuth. Record these numbers. They will be needed. For every calibrator, you will need a clutter sample. Look at the background surrounding the calibrator. Choose an equivalent background that is not near any of the calibrators and click on it. Record these coordinates. Repeat this operation for all calibrators.

The recorded coordinates must be entered into a file called coordinates.dat. This file has a specific structure that must be maintained for proper execution of Targanal. It is a text file, so editing is straightforward. Find a copy in some other data set and edit the file. Remember that the coordinates are in the line above the calibrator. An example is given.

```
pigpen 35% more coordinates.dat
1 2 669 6537 48 0.000000
Noah
1 3 669 6636 48 0.000000
Noah clutter
1 2 669 6223 48 0.000000
Serafina
1 3 669 6298 48 0.000000
Sera clutter
2 2 669 5619 48 0.000000
PowerHog
2 3 669 5686 48 0.000000
Pow clutter
4 1 669 6374 48 0.000000
DREO
4 3 669 6452 48 0.000000
DREO clutter
5 1 669 6073 48 0.000000
DREP
5 3 669 6145 48 0.000000
DREP clutter
5 1 669 5760 48 0.000000
DREV
5 3 669 5835 48 0.000000
DREV clutter
```
All the numbers in the file are important.

The first number is an arbitrary target number.
The calibrator and its clutter sample must have the same target number.

The second number is the target type.
1 - corner reflector
2 - ARC
3 - clutter

The third number is the range coordinate.
The fourth number is the azimuth coordinate.

The fifth number is the size of the mask placed around a target. The program searches for the point of maximum intensity within that mask. The mask size can be varied (e.g., 32, 128), but it must be the same for all calibrators. 64 or 48 are good numbers for the mask. The actual number used will depend on the separation of the calibrators. It is a good idea to check the corner reflector and ARC locations that Targanal has chosen to make sure it hasn’t selected a nearby target. If it has, you will have to make your mask smaller. The masks of the calibrators and of the clutter samples must not overlap.

The sixth number is there because the program likes it. It is necessary.

Targanal

You are now ready to run Targanal. Targanal is a MATLAB program that runs several functions and sub-programs. They should all be stored in one directory and the MATLAB path set up to access this directory. `targanal.m` needs 2 input files to run: "coordinates.dat" and "l#p#polgasp.hdr". Open MATLAB and enter the following command. This assumes the path has been set up.

`targanal('l#p#polgasp.hdr','coordinates.dat');`

The program starts by displaying the number of targets detected. The program quickly scrolls through its operations, but you cannot see the scrolling because your view is blocked by the constant pop-up of MATLAB windows. At the end of the program, type in "quit" and all the windows will disappear. If the MATLAB windows block your command line window, you can eliminate them with Ctrl w. Keep Ctrl w depressed until all the windows disappear. Another way is to minimize the Targanal windows until you can put the mouse on your command line window and type quit.

The program produces 3 output files:
`l#p#polgasp.ps`
`l#p#targanal.arc`
`l#p#targanal.cr`

The postscript file contains graphical plots and numerical values of the ARC recirculations and the corner responses for each channel. The ".arc" and ".cr" files contain the numerical values only that were generated by the program and displayed with the graphical plots. They are needed for input to `GenCall1_01.sh`. Print out the postscript file. Use a colour printer if one is available.

Remember that 4 MATLAB windows will be produced for each calibrator. If during the execution of the program, you hear a “beep” and the number of the window displayed is not the number of calibrators
multiplied by 4, then the program has an error. The windows will have to be closed and the postscript file examined to see which calibrator caused the error. Look at the text that is displayed in the command line window. There could be a clue to the error. If you get a “calibrator out of range” type of message, your range and azimuth coordinates may be reversed.

**Ghostview** is best for viewing the postscript output. Just type in `ghostview` (gv or ggv etc. on some systems) followed by the postscript file name. Change the orientation to landscape for easier viewing. Targanal processes the calibrators in the order that they appear in `coordinates.dat`. It processes one channel at a time in the following order: HH, HV, VV, VH. Check the file carefully for errors and NaN values. If you have an error, go through the results to see which calibrator generated the error. If you have several calibrators, you can remove the offending calibrator from coordinates.dat. Be sure to remove the corresponding clutter sample. Run Targanal again. This often works. Sometimes, varying the size of the mask will work. Targanal must complete without errors or the image can’t be calibrated.

To calibrate an image you must have at the very least, one ARC and one corner reflector. Generally, the more calibrators you have, the better. It greatly increases your confidence in a given set of results if the calibrators are in general agreement.

Before moving on to the calibration, there is one more essential check. It is called:

**CALQC.sh**

This program checks the calibrators for saturation and also checks the accuracy of the georeferencing. There are Linux and UNIX versions. You need to be in the precal directory. Type in CALQC.sh to find the requirements.

**CALQC.sh**

Usage: `/home/tpotter/bin/CALQC.sh <pathSig> <pathImg> <linepass> <pol> <coordFile>`

You need the path to the signal data, the path to the image data, line & pass (l#p#), the polarimetric channel and the calibrator coordinates (coordinates.dat) file. If you also want to do the georeference check, you will have to add the PRF file to the command line. This is an example from Linux.

```
CALQC.sh /rde/tpotter/Iceberg/381icepro /rde/tpotter/Iceberg/381icepro/newIpro/precal_new_improved l3p10 hh coordinates.dat l3p10newprf.mat
Signal header file is /rde/tpotter/Iceberg/381icepro/l3p10hh.hdr
Image header file is /rde/tpotter/Iceberg/381icepro/newIpro/precal_new_improved/l3p10hhpolgasp.hdr
Transposed: 0, sig size: 4096 107180, image from 182 68878 defined 8 windows
Extract from 1033 74350 in sig data
+ statis -z -i /rde/tpotter/Iceberg/381icepro/l3p10hh.sig -D 4096 107180 -d 32 4096 -x 1033 74350
+ cat pgm_header /rde/tpotter/Iceberg/381icepro/l3p10hh.sig.chip
+ histo -z -i /rde/tpotter/Iceberg/381icepro/l3p10hh.sig.chip -D 32 4096 -d 32 4096 -H 128
+ mv /rde/tpotter/Iceberg/381icepro/l3p10hh.sig.chip.his arc1hh.his

Extract from 1033 74203 in sig data
+ statis -z -i /rde/tpotter/Iceberg/381icepro/l3p10hh.sig -D 4096 107180 -d 32 4096 -x 1033 74203
+ cat pgm_header /rde/tpotter/Iceberg/381icepro/l3p10hh.sig.chip
+ histo -z -i /rde/tpotter/Iceberg/381icepro/l3p10hh.sig.chip -D 32 4096 -d 32 4096 -H 128
+ mv /rde/tpotter/Iceberg/381icepro/l3p10hh.sig.chip.his arc2hh.his

```

45
Extract from 1014 74201 in sig data

The response of each calibrator in the indicated channel is plotted as a histogram and displayed. An ARC and a corner from a certain trial are displayed. Look for the extreme values at each end of the histogram. The peaks should be very small. If they are not small as this example shows, the calibrator is saturated. If it is possible to avoid using a saturated calibrator, do it. For ARCs, use the second recirculation as a starting point. You will have to carefully edit the `targanal.ARC` file so that the GenCal program thinks it’s legitimate. Corners are more of a problem since there is only one point to work with. Use a non-saturated corner if possible. See how it compares with other calibration line values. Does it pass the calibration checks?

To plot histograms again use gnuplot with:

To look at histograms open files:

These files should be saved. The `.plt` files can be saved as postscripts (`Gnu2PS.sh`).
CALQC carries on with a geo-location test. The coordinates of the calibrators are calculated with IMG2GEO.sh. The actual location should be available from the field report of the trial. There is uncertainty in the results, but they should be close. If not, you will have to find out why.

Checking geolocation for ARCs:
Checking geolocation for CRs:
+ IMG2GEO.sh 13p10newprf.mat coordinates.arc
/rde/tpotter/Iceberg/381icepro/newIpro/precal_new_improved/13p10hhpolgasp.hdr
Using files: 13p10newprf.mat coordinates.arc
/rde/tpotter/Iceberg/381icepro/newIpro/precal_new_improved/13p10hhpolgasp.hdr
+ disprf -m 13p10newprf.mat -n 69729 -N 69729 -r 29178.60693400
+ disprf -m 13p10newprf.mat -n 69729 -N 69729 -r 28590.60693400
Right look: lat 53.394610 long -60.173416 alt 0.000000
Right look: lat 53.391520 long -60.180895 alt 0.000000
+ IMG2GEO.sh 13p10newprf.mat coordinates.cr
/rde/tpotter/Iceberg/381icepro/newIpro/precal_new_improved/13p10hhpolgasp.hdr
Using files: 13p10newprf.mat coordinates.cr
/rde/tpotter/Iceberg/381icepro/newIpro/precal_new_improved/13p10hhpolgasp.hdr
+ disprf -m 13p10newprf.mat -n 69710 -N 69710 -r 29178.60693400
+ disprf -m 13p10newprf.mat -n 69710 -N 69710 -r 28590.60693400
Right look: lat 53.394588 long -60.173391 alt 0.000000
Right look: lat 53.391455 long -60.180970 alt 0.000000
"
Checking the Focus

Once you have run a successful Targanal, print out the postscript file. Go to the “Co-Registration & Focus info” template part of the Processing Parameter sheet. Label each calibrator. All ARCs require 4 channels and all corners 2 channels. Cut & Paste as necessary. From the postscript copy the “Azimuth Line #” and “Peak Location (sample#)” of the 0 recirculation. Do this for every channel of every ARC. Differences are calculated automatically. For the corners, in the “Peak Statistics” box, use the azimuth and line from the second entry of “Peak Value”. Also copy the “Azimuth 3dB Width” and the “Range 3dB Width”. For the corners use the HH and VV channels only. If the focus is way off, you should be worried.

GenCal1_05.sh

If the image being processed is a calibration line, there will be no K’ and phase correction values yet. This program will calculate those values.

You should be in the precal directory at this point. Create another directory called “genCALresults”.

mkdir genCALresults

Also type in the command pwd and highlight the result. It will save a lot of keystrokes in the first GenCal command.

In order to run GenCal, You will have to have a calibration work directory set up. At the beginning, it will not have the “ANAL*”, “AVE*”, “CAL*” files. These files are generated and over-written every time GenCal is run. It should have all the other following files:

<table>
<thead>
<tr>
<th>File</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANAL_ARC.HHnumrec</td>
<td>Bite.sh</td>
</tr>
<tr>
<td>ANAL_ARC.HHpositions</td>
<td>Bitesh_messages.txt</td>
</tr>
<tr>
<td>ANAL_ARC.HHrecirc</td>
<td>CAL_ALL.k</td>
</tr>
<tr>
<td>ANAL_ARC.HHsep</td>
<td>CAL_ALL.phase</td>
</tr>
<tr>
<td>ANAL_ARC.HVnumrec</td>
<td>CAL_ARC.caption</td>
</tr>
<tr>
<td>ANAL_ARC.HVpositions</td>
<td>CAL_ARC.k</td>
</tr>
<tr>
<td>ANAL_ARC.HVrecirc</td>
<td>CAL_ARC.phase</td>
</tr>
<tr>
<td>ANAL_ARC.HVsep</td>
<td>CAL_CR.caption</td>
</tr>
<tr>
<td>ANAL_ARC.VHnumrec</td>
<td>CAL_CR.k</td>
</tr>
<tr>
<td>ANAL_ARC.VHpositions</td>
<td>GenCal.sh</td>
</tr>
<tr>
<td>ANAL_ARC.VHrecirc</td>
<td>GenCal1_01.sh</td>
</tr>
<tr>
<td>ANAL_ARC.VHsep</td>
<td>Replicate.sh</td>
</tr>
<tr>
<td>ANAL_ARC.VVnumrec</td>
<td>ShowC.sh</td>
</tr>
<tr>
<td>ANAL_ARC.VVpositions</td>
<td>ShowGnu.sh</td>
</tr>
<tr>
<td>ANAL_ARC.VVrecirc</td>
<td>ShowGnu.sh.bck</td>
</tr>
<tr>
<td>ANAL_ARC.VVsep</td>
<td>ShowL.sh</td>
</tr>
<tr>
<td>ANAL_ARC.numrec</td>
<td>about.sh</td>
</tr>
<tr>
<td>ANAL_ARC.pols</td>
<td>append</td>
</tr>
<tr>
<td>ANAL_ARC.recirc</td>
<td>arc.</td>
</tr>
<tr>
<td>AVE_CAL.k</td>
<td>caption</td>
</tr>
<tr>
<td>AVE_CAL.phase</td>
<td>chkphase</td>
</tr>
<tr>
<td>AVE_CR.k</td>
<td>chkphase_CAL_ARC.phHH.log</td>
</tr>
<tr>
<td>Append.sh</td>
<td>chkphase_CAL_ARC.phHV.log</td>
</tr>
<tr>
<td>chkphase_CAL_ARC.phHH.log</td>
<td>hornGain</td>
</tr>
<tr>
<td>chkphase_CAL_ARC.phVV.log</td>
<td>joinRecirc</td>
</tr>
<tr>
<td>CAL_ALL.phase</td>
<td>legend.ARC</td>
</tr>
<tr>
<td>CAL_ARC.caption</td>
<td>legend.CR</td>
</tr>
<tr>
<td>CAL_ARC.k</td>
<td>links</td>
</tr>
<tr>
<td>CAL_CR.caption</td>
<td>pickGain</td>
</tr>
<tr>
<td>GenCal.sh</td>
<td>ptInteg</td>
</tr>
<tr>
<td>GenCal1_01.sh</td>
<td>ptSlant</td>
</tr>
<tr>
<td>Replicate.sh</td>
<td>rcsTrih</td>
</tr>
<tr>
<td>ShowC.sh</td>
<td>statDQ</td>
</tr>
<tr>
<td>ShowGnu.sh.bck</td>
<td>table.ARC</td>
</tr>
<tr>
<td>ShowL.sh</td>
<td>table.CR</td>
</tr>
<tr>
<td>about.sh</td>
<td>values</td>
</tr>
<tr>
<td>append</td>
<td>values</td>
</tr>
<tr>
<td>arc.</td>
<td>values</td>
</tr>
<tr>
<td>caption</td>
<td>values</td>
</tr>
<tr>
<td>chkphase</td>
<td>values</td>
</tr>
<tr>
<td>chkphase_CAL_ARC.phHH.log</td>
<td>values</td>
</tr>
<tr>
<td>chkphase_CAL_ARC.phHV.log</td>
<td>values</td>
</tr>
</tbody>
</table>

There is a Make file and a setup file for installing the calibration work directory. Some minor modifications have been made. The main script is now GenCal1_05.sh. Make sure you have this latest version.
The first command you use is **GenCal1_05.sh**. If you can't remember the required parameters, just type in the command and you will get this response.

**GenCal1_05.sh**
Usage:
GenCal1_05.sh PathALinePass PathTLinePass AntennaDB
Example:
GenCal1_05.sh /WhereIsAnc/l8p8 /WhereIsTarg/l8p8 /WhereIsANTENNADB/

The first parameter is the full path to the signal ancillary data. This would be your `/l#p#` or signal data directory. The path must end with `/l#p#`.

The second parameter is the full path to the targets directory. This would be your `pro/precal` directory. Again the path must end with `/l#p#`.

The third parameter is the full path to the `antennaDB` files. These could be in your `/bin` directory or somewhere else. In this case the path ends with a forward slash, `"/"`.

There is a space between each parameter. An example is:

```
```

The automatic formatting in word has messed up the example. When you put in the command, you just keep typing until it is complete. The long path names required is why "pwd" was highlighted above.

There are several lines that go flying past on the screen. The important ones are displayed.

**AND CONTINUING ON**

```
** EXTRACTING MORE INFO FROM
/pigpen4/tpotter/chasptest5/24sep02/l1p3/l1p3hnh.anc

Control Station Data for /pigpen4/tpotter/chasptest5/24sep02/l1p3/l1p3h

1. Average Tx power     (    0.000)
2. PRF                            (    0.000)
3. PRF/V                        (    0.000)
```

Enter # to modify : 1

Enter the appropriate values.

```
Control Station Data for /pigpen4/tpotter/Poltest5/24sep02/l1p3/l

1. Average Tx power     (   -2.270)
2. PRF                            (  327.000)
3. PRF/V                        (    2.320)
```

Enter # to modify (0 exits menus) : 0
Deployment Parameters for S8

1. Hypotenuse length   (0.000)
2. Base elevation      (0.000)
3. Left front azimuth  (0.000)
4. Magnetic offset     (0.000)

Enter # to modify (0 exits menus):

This menu will come up for every corner reflector calibrator. **All values must be filled in.**

After all the values for all the corners have been filled in, the RCS, K’ and phase constants for the corners are calculated and weighted averaging is applied and recirculations are matched.

Deployment Parameters for Powerhog

1. Rx horn polarization angle  (0.000)
2. Tx horn polarization angle  (0.000)
3. HH phase shift correction  (-180.000)
4. HV phase shift correction  (-180.000)
5. VV phase shift correction  (-180.000)
6. VH phase shift correction  (-180.000)

Enter # to modify:

**All parameters must be filled in for all ARCs.**

The horn polarization angles will be 45° unless you know it is different.

The phase shift correction is already filled in and set at the default for most ARCs. Some ARCs (e.g., Gemini) are different. A table exists with phase shift corrections for known ARCs.

After the ARC menus are filled in, this the final response.

** CALCULATING GAIN FACTORS FOR POLS

ARC RCS: HH, VV, HV, VH:

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>27.69</td>
<td>26.30</td>
<td>26.995</td>
<td>26.995</td>
</tr>
<tr>
<td>20.86</td>
<td>20.17</td>
<td>20.515</td>
<td>20.515</td>
</tr>
<tr>
<td>18.82</td>
<td>18.35</td>
<td>18.585</td>
<td>18.585</td>
</tr>
<tr>
<td>14.70</td>
<td>12.96</td>
<td>13.830</td>
<td>13.830</td>
</tr>
<tr>
<td>10.16</td>
<td>8.15</td>
<td>9.155</td>
<td>9.155</td>
</tr>
</tbody>
</table>
** CALCULATING K' FOR ARCS
ARC K': HH, HV, VV, VH:
115.93 129.455 118.17 129.105
115.93 129.485 118.17 129.295
115.94 130.115 118.18 129.995
115.94 130.115 118.18 129.995
115.93 130.01 118.17 129.41
115.93 130.315 118.17 129.985

** CALCULATING PHASE CONSTANTS
ARC phase const.: HH, HV, VV, VH:
0 49.87 287.75 31.66
0 47.34 285.79 30.08
0 58.72 300.36 42.56
0 57.10 287.86 34.05
0 61.82 286.00 43.28
0 69.99 298.44 47.72
0 36.43 244.58 19.18

** COMPUTING WEIGHTED AVERAGE K' AND PHASE
K'HH = 115.931761  K'VV = 118.173665  K'HV = 129.490752  K'VH = 129.136574
PhHH = 0.000000  PhVV = 287.972595  PhHV = 51.128677  PhVH = 32.896076

The K's and phase corrections are right there. For a hard copy of the results print out "about.". The period in "about." is part of the title of the file.

ezecalcopy

All the calibration results for the image you are presently working on are in the calibration work directory. Before you leave the calibration work directory, you should copy the appropriate files to a genCALresults directory. genCALresults should be a subdirectory of your precal directory. In a separate window, go to the genCALresults directory. Key in pwd and highlight the results. Go back to the calibration work directory and use the program ezecalcopy.

Just type in ezecalcopy. You will be prompted for the full path to the appropriate genCALresults directory. Once that is entered, all the important and unimportant calibration results will be copied to that directory.

** Calibration Incidence Angle

The calibration incidence angle is needed for further image analysis and is essential for a calibration check. The best way to find this is to take all the parameters calculated by GenCall1_05.sh and display them in a spreadsheet. This has the advantage of displaying all the parameters with labels, so that any given parameter can be quickly and easily found. This can be done with the following instructions:

** Getting Intermediate Cal Files into a Spreadsheet

ftp table.arc & table.cr to a temp directory on your PC

Go into Excel
click new file

click file/open

go to one of the table files

in the "wizard" check delimited

click next

check space

click finish

highlight spreadsheet contents & copy

paste to ARC_CR_Cal_Parameters_template.xls

delimited

If you don’t have the Excel template, get a copy from Terry Potter.

The incidence angle can readily be found by scrolling horizontally along the spreadsheet. Be sure to record the incidence angle. It will be used for the corner reflectors calibration check.

**maketiff_chasp.m**

Once the calibration site image has been processed, an image should be made and printed out. When the entire image has been processed, it should also be converted to a viewable format and printed out. The program **maketiff_chasp.m** produces an RGB colour composite of almost any given 4-channel polarimetric image. The user is asked what resampling should be used in the range and azimuth axes. Through experience, it has been found that a range to azimuth ratio of 1:8 works best for a “natural” appearance. Depending on the size of the image, you would choose resampling of 1 in range to 8 in azimuth, 2 in range to 16 in azimuth, 4 in range to 32 in azimuth etc. The result is a tiff format RGB image. This program was originally made for images produced by PolGASP, but has been modified for COASP images. To make a tiff from a PolGASP image, use the program **maketiff_chaspro.m**. The prompts are the same. The first response will display the present working directory. The first prompt is for the directory that the images are in. It is almost always the present working directory, so the directory can be highlighted and pasted into the response. You will be then be prompted for the base name of the file, azimuth resampling and then range resampling. Follow the instructions below:

```
matlab –nojvm
>>maketiff_chasp
/pigpen5/…/pro/cal
l#p#
16
2
```

Open MATLAB command line without MATLAB GUI

Enter directory of image

Enter base name of file

Enter azimuth resampling. This will usually be 16. If the image has ~34,000 azimuth lines or less, a value of 8 can be used.

Use 1 if the azimuth resampling is 8, 2 if it is 16 etc. Prepare to wait. This will take some time.
Enter any directory to store RGB TIFF image

Exits MATLAB

Right click in XV2 window

Load the file with the .tiff ending

Change printer from lpr to lp –dHP4700RM227

Click on color, landscape or portrait, max and OK.

*NOTE*: If you are making a tiff from a PolGASP image, the range and azimuth resampling must be reversed. PolGASP is a transposed format.

The program will take several minutes to complete. As the program goes through the 4 channels, at some point it may stop and give you an “out of memory” message. If this happens, you must increase the amount of resampling and run it again.

There is also a variation of maketiff_chasp.m called maketiff_chasp200.m or maketiff_chaspol200.m. The prompts in the program are exactly the same. The difference is in the result. The program is designed to enhance ocean surface features. Some of these features would normally not show up in maketiff_fix.m. The other change is that the output file will be called l#p#sp200.tiff.

Extraction of the CalSite for an Image

Once the image has been calibrated, an image of the calibrators themselves should be made to show their positions relative to any sources of interference, check for any obvious saturation and give an indication of how well focused the ARC is.

Use the .xv image produced by xv_GASP_detect. Zoom in on the calsite. Choose an upper left and lower right corner that will clearly show the calibrators and some of the surrounding area. Click on the centre mouse button and copy the coordinates for each chosen corner. Then you will have your start and end range and azimuth coordinates.

The next step is to look in one of the PolGASP or RC header files. Record the first line and the start sample. The start sample value must be added to the range coordinates. The first line must be added to the azimuth coordinates. This is because the extraction program references the full image. Subtract the first range line from the last range line and add 1. Then subtract the first azimuth line from the last azimuth line and add 1. The program will prompt you for the number of range lines to extract and the number of azimuth lines to extract.

The program is called extRCdata1_4_4.m and is run in MATLAB. If the file names are in PolGASP format, they will have to be converted to RC format. Use ezepol2rcname line# pass#. Open MATLAB and enter extRCdata1_4_4. You will be prompted for the name of the HH header file. This program is based on a program for extracting PolGASP format images. Because of this, what is called range is actually the azimuth and what is called azimuth is actually the range. The program will tell you what range of lines are in what it calls range and azimuth. Pay careful attention to what the prompts say. It is easy to use, but takes a bit of practice. Once the extraction is complete, use the following command: imagesc(abs(hhData)). Any calibrators should stand out, even if nothing else does. If you are happy with the area covered, save the extraction. Give it any name you want. The command is save my-image.

Most of the time you will want to make an RGB image of the extraction. The next command would then be clear all. Then load my-image. The program is extMakeRGB1_0.m and that is the command. You
are given a choice of 2 types of RGB. Most of the time you will use 1. If it does not work, just use the command `extMakeRGB1_0` again and choose 1.

The program is demonstrated below:

```
>> extRCdata1_4_4
  Enter the HH header filename: l1p10hhrc.hdr
  retVal =
    384
  retVal =
    384
  retVal =
    3712
  retVal =
    71501
  retVal =
    71501
  retVal =
    23900

  Image range lines: 71501 to 95400
  Image azimuth lines: 384 to 4095

  Enter the first range line to extract: 84500
  Enter the number of range lines to extract: 200
  Enter the first azimuth line to extract: 417
  Enter the number of azimuth lines to extract: 100

  retVal =
    3712

  retVal =
    23900
```
retVal =

   6926.420410

retVal =

   9306.7695312500

retVal =

   /pigpen4/tpotter/328_chasp1/328_asarpro/l1p10/l1p10hh.rc

retVal =

   23-SEP-2004 20:03:24.00100

retVal =

   2.314430

Information about the extracted data:
Aircraft altitude:         6926.4204100000
Near slant range distance: 9306.7695312500
PRFoverV:                  2.3144300000

>> imagesc(abs(hhData))
>> save my_image
>> clear all
>> load my_image
>> extMakeRGB1_0

polGASP/chasp1 RGB Image Creation
(1) Standard RGB Image
(2) Use stretch factor of 200 (good for marine environments)

Enter your choice (1 or 2): 1
>> save 328_my_image
>>

When the process is completed, as well as having images of each channel and an RGB, there will be a
structure called “A” which contains a lot of information about the extracted image.

>> load 348_127p10_calsite.mat
>> whos
Name           Size            Bytes      Class
A              1x1             2300      struct array
RGB            745x266x3     594510    uint8 array
hhData         745x266       3170720   double array (complex)
hvData         745x266       3170720   double array (complex)
vhData         745x266       3170720   double array (complex)
vvData         745x266       3170720   double array (complex)
Grand total is 1387337 elements using 13279690 bytes

>> A

A =

    Tile: '/pigpen4/tpotter/348Mpro/l27p10h'
    ImageDateTime: '17-OCT-2005 15:55:59.54700'
    Source: '/pigpen4/tpotter/348Mpro/l27p10hh.rc'
    StartRange: 981
    EndRange: 1246
    StartAzimuth: 58453
    EndAzimuth: 59197
    NearIncAngle: 50.5107
    FarIncAngle: 54.7192
    Altitude: 6.6726e+03
    near_srd: 6.7046e+03
    PRFoverV: 2.5669
    DateExtracted: '21-Nov-2005'
    ProgramName: 'extRCdata'
    Version: '1.4.4'
    extMakeRGBversion: '1.0'
PROCESSING AND CHECKING THE CALIBRATED IMAGE

Now that you have all the calibration parameters from the noise, the precal image and `genCal1_05.sh`, the final calibrated image can be processed. Polarimetric images are invariably large, so setting up a shell script to run them overnight is a good idea. In fact, processing a smaller image (20,000 lines) is a better idea in case changes have to be made. The first thing to do is edit the `chasp1c.conf` file.

**Editing Chasp1c.conf**

```bash
pigpen 84% more chasp1c.conf
; WorkOrder
;
mission_id        326_l1p8_sep2304_CoCoNaut
input_data_file   /pigpen5/tпотter/326_pro/l1p8/l1p8
channels          4
chirp_file        marina_2.tmp
prf_file          l1p8prf.mat
first_line        1
last_line         246129
SAW               OUT
geoid_over_ellipsoid -15.27
; choose:
georef_grid_rg    3
doppler_grid_rg   50
satur_grid_rg     50
channel_align     Yes
range_discard     Yes
motion_compensation Yes
motion_restore    Yes
azimuth_focus     Yes
RangeDoppler      Yes
ZD_frame          Yes
azimuth_align     Yes
antenna_gain      Yes
 calibration      Yes
auto_nadir        Yes
skip_nadir        0
cut_out           No
window_shape      2.8
full_bandwidth
; do you wish to overwrite the PRF value for prf_over_v
use_mean_prf_over_v No
relative_dc_offset 0
relative_v_offset 0
; radiometry:
reference_range 1
spread_loss 3
antenna_hp       /pigpen0/tпотter/local/bin/antennaDB/hp.dat
antenna_vp       /pigpen0/tпотter/local/bin/antennaDB/vp.dat
antenna_hs       /pigpen0/tпотter/local/bin/antennaDB/hs.dat
antenna_vs       /pigpen0/tпоттер/local/bin/antennaDB/vs.dat
; calibration:
; from the sheet in dB
AvTx_power       -2.76
```
; from GenCal in dB
K_prime_hh  116.630
K_prime_hv  132.162
K_prime_vv  119.040
K_prime_vh  133.033
; from GenCal in degrees
Phase_corr_hh  0.0
Phase_corr_hv  38.1446
Phase_corr_vv  283.7161
Phase_corr_vh  57.6891
; power (from PowBite not in dB)
AvBITE_hh  6.1389e-03
AvBITE_hv  3.0216e-03
AvBITE_vv  5.8382e-03
AvBITE_vh  2.6663e-03
;
; System parameters
;
wavelength  0.0565646000
range_shift_hh  0
range_shift_hv  0.52
range_shift_vv  0.93
range_shift_vh  1.33
azimuth_shift_hh  0
azimuth_shift_hv  0
azimuth_shift_vv  0.5
azimuth_shift_vh  0.5
range_slant_spacing  4.0
saw_delay  13.289e-06
boresight_offset_hs  -1.585
boresight_offset_vs  -1.585
boresight_offset_hp  -3.330
boresight_offset_vp  -3.330
antenna_ang_min  -73.0
antenna_ang_inc  1.0
azimuth_beamwidth  3.03
CalConstant  19.0
;
; Processor parameters
;
processors  1
block_lines  8192
parts  4
channels_together  1
change_anc_endian  No
change_output_endian
interp_size  11
interp_fractions  16
; possibilities for mocomp_type are Flat, Part, Each
mocomp_type  Part
pigpen 85%
Most of the parameters are already set and should not be changed. The parameters that are highlighted will have to be changed with almost every image. Some of the changes have already been explained in Editing chasp1n.conf and chasp1a.conf. They will be repeated here for ease of reading.

**Mission_id:** include as a minimum asar number, line number, pass number, trial name and date.

**Input_data_file:** This is the path to the signal data. At the end of the path add a slash with the l#p# prefix for that data (e.g. /l2p8).

**prf_file:** This is the main mocomp file produced by InQC. **This must be included or the image will not process.** Every image has a different PRF file.

**First_line:** The first line of the image to be processed, usually 1

**Last_line:** The last line of the image to be processed.

**SAW:** You must indicate whether the image is SAW-IN or SAW-OUT with **IN** or **OUT**.

**geoid_over_ellipsoid:** This is the height above or below the ellipsoid in meters of the image. If the image has a range of elevations, choose the elevation of the targets of interest. If the site is below the ellipsoid, it must be prefixed with a minus sign.

**motion_restore:** This must be set to YES when doing calibration. Motion compensation is used in the processing, but the original image must be restored in order to run Targanal and do the subsequent calibration.

**antenna_hp, antenna_vp, antenna_hs, antenna_vs:** The path to these parameters has to be set up to match your file configuration.

**K_prime_hh, K_prime_hv K_prime_vv K_prime_vh:** These are the K’ values obtained by processing the GenCal1_05.sh script.

**Phase_corr_hh, Phase_corr_hh, Phase_corr_hh, Phase_corr_hh:** These are the Phase Correction values obtained by processing the GenCall_05.sh script.

**power (from PowBite not in dB):** The next 4 entries are the bite noise values obtained by running PowerBite.sh or ezebitenoise.

**range_shift_hv, range_shift_vv, range_shift_vh:** These parameters will vary depending on the date that the image was acquired. If the image was acquired in September 2004 or later, then the parameters above are correct. If the image was acquired before then, the following parameters should be changed as follows:

```plaintext
range_shift_hh: 0
range_shift_hv: 0
range_shift_vv: 1.25
range_shift_vh: 1.25
```

**change_anc_endian:** This will be NO if the file was created and run on a platform of the same endianess. This will be YES if the file was created and run on machines with different endianess.

**change_output_endian:** As above, this will depend on where the files will be used and the application. Generally, if they will be used on a platform of the same endianess, the answer is NO. If they will be used on a platform of different endianess, the answer is YES.
mocomp_type: Use Part for this processing. It is more accurate than Flat but doesn’t have the time demand of Each. If you find you cannot get complete results from Targanal or GenCal1_05.sh, then you may have to reprocess using Flat.

Other parameters to note:

chirp_file: the main one for processing CHASPI imagery is marina_2.tmp, but this can be changed

block_lines: Make sure these are at least 4096. If the number is less the program won’t run. A “normal” number for block_lines is 8192. A different number can be chosen but it cannot be \(< 4096 and must be a multiple of 4096.

Auto_nadir: COASP automatically checks for the nadir line. This is normally left on. This is especially important in SAW OUT images where the nadir line will be 384. If you want to set the nadir line manually, set auto_nadir to No and set Skip_nadir to the desired nadir line. To see all the lines before nadir, set range_discard to No. If you want to start processing an image from a range line other than nadir, you would set it here.

Cut_out: This is the second parameter to set when you process a specific number of range lines. Cut_out specifies the number of range lines to be processed. Skip_nadir specifies the starting range line.

Run the calibration file.

chasp1 –w chasp1c.conf

Once the image is processed, make a tiff image. Perform the following checks on the large processed image or on a smaller (~20,000 azimuth lines) processed and calibrated sub-image. The smaller image is easier to work with.

**The Calibration Check**

The calibration can be checked in the following way. Take the integrated response of a corner reflector in HH or VV as calculated by targanal.m. Find out the incidence angle at the calibrator (Check Calibration Incidence Angle, page 50). From the integrated response, subtract \(10^\ast(LOG10(SIN of\ Incidence\ angle))\). This gives an adjusted response. The adjusted response should be equal to the theoretical response of the corner reflector. In practice, this never happens due to a number of possible error sources. We have considered the calibration to be correct if the difference between the theoretical response and the adjusted response is less than 1.0 dB.

Run **targanal.m** as detailed previously on a calibrated image. This can be the full final image that is processed or it can be a small (20,000 lines) processed and calibrated for the purpose. If you use the full image, you will have to alter the calibrator coordinates by adding the appropriate number of azimuth lines. Using the small processed image is the preferred method if you want to use minimum time and space. If you have to correct errors or reprocess, a full image will take a lot more time.

A spreadsheet has been specially prepared for the calibration check. On your PC, get a copy of the CalCheck spreadsheet. Edit it for calibrator names and theoretical responses. Fill in the image number and incidence angle. \(10^\ast(LOG10(SIN of\ Incidence\ angle))\) is automatically calculated.

Go to the cal directory. Use **ghostview** to display l#p#polgasp.ps. Go to the pages for the peak analysis of each corner reflector in the HH and VV channels. In the box of Peak Statistics find the Integrated Response. In the bottom of this box is the name of the calibrator. Copy the integrated response to the appropriate box in the spreadsheet. Calculations will be done automatically. Right away you can see if the difference is between 1.0 and -1.0. Repeat this for all corner reflectors.
**The Phase Check**

For this, you use the extracted calibration site image. Open MATLAB and load the `.mat` file of the calibration site. Then use the command:

```matlab
imagesc(abs(hhData))
```

In the MATLAB window, click on the “+” button and place the mouse over the first re-circulation of one of the ARCs. Click on the centre of the re-circulation until it almost fills the window. Look at the numbers in the range and azimuth. Using integers only, record the numbers that best encompass the brightest pixel(s) in the re-circulation. As a general guideline, the numbers you end up with in the range should not differ from each other by more than 4. The same guideline applies to the azimuth. It can take a bit of experimenting to obtain an optimum result. Repeat this procedure for all the calibrators. Record these numbers. They are necessary for the next step. Stay in MATLAB and run the program `phaseplot.m`.

```matlab
>> load l3p9calsite.mat
>> whos
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Size</th>
<th>Bytes</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>1x1</td>
<td>2186</td>
<td>struct array</td>
</tr>
<tr>
<td>hhData</td>
<td>634x490</td>
<td>4970560</td>
<td>double array (complex)</td>
</tr>
<tr>
<td>hvData</td>
<td>634x490</td>
<td>4970560</td>
<td>double array (complex)</td>
</tr>
<tr>
<td>vhData</td>
<td>634x490</td>
<td>4970560</td>
<td>double array (complex)</td>
</tr>
<tr>
<td>vvData</td>
<td>634x490</td>
<td>4970560</td>
<td>double array (complex)</td>
</tr>
</tbody>
</table>

Grand total is 1242791 elements using 19884426 bytes

```matlab
>> phaseplot
What is the ASAR number? 380

Line number = 3

Pass number = 9

Enter a short date (e.g. Oct 1805): Jun 0706

What is the calibrator name? DREV

Input first range coordinate: 202

Input second range coordinate: 204

Input first azimuth coordinate: 225

Input second azimuth coordinate: 228
Do you wish to change the range / azimuth coordinates? (y/n): n

pcmd =
lp -dlj3_c
dev =
-dlj3

pcmd =
lp -dlj3_c
dev =
-dlj3

380 l3p9 Jun 0706

Intensity - HHVV: -0.0514214 VVHH: -0.0514214 HVVH: 0.00776503 VHHV: 0.00776503
Mean Angle HHVV = 3.95885 degrees
Mean Angle VVHH = -3.95885 degrees
Mean Angle HVVH = -2.67537 degrees
Mean Angle VHHV = 2.67537 degrees

Do another Calibrator Conjugate from the same image? (y/n): y
What is the calibrator name? Powerhog

Input first range coordinate

ETC.

The program then displays a polar diagram of the result. It will then ask you if you want to change coordinates. If the plot looks bad, you may want to. Answer no and the program saves the plot as a JPEG image. It will then ask if you want a polar plot for another calibrator from the same image. Answer yes until you run out of calibrators.

There will be 4 plots on each sheet. It displays the conjugates of HHVV, VVHH, HVVH and VHHV. For the corner reflectors, all the circles in the HHVV and VVHH plots should line up approximately on zero from the centre to the right hand side. For the ARCS, the line-up on zero should be for the HVVH and VHHV plots. If that is the case, there is a good phase correction. If not, you will have to figure out why and do more work. It should look something like this:
All the relevant parameters are listed in the titles. Varying any one of these in the program will produce a different saved image, so don’t be afraid to experiment. You can choose the best one later. If you only want one plot with one conjugate or single plots, then use the program `phaseplot_single.m`. Just follow the prompts.

**Balance.sh**

`Balance.sh` is a quality control test to give you an idea of how good or how bad the image is by comparing the HV and VH channels. It DOES NOT correct or change anything. `Balance.sh` should be performed on all images with a landmass, not just the calibration images. In an ideal situation, the magnitude of HV will equal the magnitude of VH for all samples. This doesn’t happen in the real world. `Balance` shows & quantifies the difference between HV and VH for the given sample (chip) size.

The ideal target area is over land where there is no moving target (wind on foliage is a moving target, so are most bodies of water). The larger the area is, the better. A larger area means a longer processing time, usually a few minutes. Typing in `Balance.sh` gives you this:

```
pigpen 55% Balance.sh
Usage:
  /pigpen0/tpotter/local/bin/ChaspStuff/Balance.sh PROCESSOR LP R0 A0 R A
Example PolGASP:
  /pigpen0/tpotter/local/bin/ChaspStuff/Balance.sh P 1lp8 1000 12000 256 1024
Example COASP:
  /pigpen0/tpotter/local/bin/ChaspStuff/Balance.sh C 1lp8 1000 12000 256 1024
Example CHASP:
  /pigpen0/tpotter/local/bin/ChaspStuff/Balance.sh H 1lp8 1000 12000 256 1024
```

The required parameters are: Processor (COASP, PolGASP or CHASP), line #, pass #, centre range, centre
azimuth, number of range lines and number of azimuth lines. Determining all this is tedious, so a shell
script was created to make it easier and record the results. All you have to do is find 2 corners of your
chosen area.

dezebalance requires 2 arguments, line # and pass #. It is set up for COASP processing. The script can be
edited for other processors. You are prompted for beginning and end of range and azimuth lines. It then
produces the Balance.sh command line you need. Next, it opens a UNIX script called lip#_balance.txt.
You then copy and paste the Balance.sh command line into the script. It will execute. Give it time. The
results will be recorded in the script. When it is done, type exit to close the script.

pigpen 54% ezebalance 3 10
Enter first Range Line:
368
Enter last Range Line:
4012
Enter first Azimuth Line:
18144
Enter last Azimuth Line:
29568
Balance.sh C l3p10 2190 23856 3644 11424
Script started, file is l3p10_balance.txt
Word too long.
pigpen 1% Balance.sh C l3p10 2190 23856 3644 11424
statis -C -D 4064 109032 -d 3644 11424 -o 2190 23856 -i l3p10pol.rc
balance -d 3644 11424 -e rc.chip -i l3p10
input files l3p10vv.rc.chip l3p10vh.rc.chip l3p10hv.rc.chip l3p10hh.rc.chip
dimension 3644 x 11424
Phv = 1.132493e-02 Phv = 8.192008e-03 <hv conj(vh)> = (9.421973e-03) + j(-5.328087e-04) Presidual =
4.536714e-04 ratio = 4.005945e-02
hv ~ ( (1.150142) + j(-0.065040) )vh = 1.151980 exp(j(-3.236606 deg))
pigpen 2% exit
Script done, file is l3p10_balance.txt

You will have to edit the script to make it more readable. The best way is to copy another balance file and
etit. The result should look something like this. The values in bold are copied onto the bottom lines of
the file.

pigpen 48% more l3p10_balance.txt
Script started on Wed Nov 8 11:37:30 2006

pigpen 1% Balance.sh C l3p10 2190 23856 3644 11424
statis -C -D 4064 109032 -d 3644 11424 -o 2190 23856 -i l3p10pol.rc
balance -d 3644 11424 -e rc.chip -i l3p10
input files l3p10vv.rc.chip l3p10vh.rc.chip l3p10hv.rc.chip l3p10hh.rc.chip
dimension 3644 x 11424
Phv = 1.132493e-02 Phv = 8.192008e-03 <hv conj(vh)> = (9.421973e-03) + j(-5.328087e-04) Presidual =
4.536714e-04 ratio = 4.005945e-02
hv ~ ( (1.150142) + j(-0.065040) )vh = 1.151980 exp(j(-3.236606 deg))

Result
----
Balance done on land

Magnitude Scale (Ideal = 1) 1.151980
Rotation Angle (degrees) (Ideal = 0) -3.237
Ratio (smaller = better) 0.040060

64
Once you have processed an entire calibrated image, this test can be performed. You will notice 6 files in your processed image directory. `extreme_0` to `extreme_3` and `doppler_0` to `doppler_3`. If these files are not present, this test cannot be performed. No arguments are required.

A total of 13 plots will be displayed and stored in a file called `test.plt`. Save these files with `Gnu2PS.sh`.

First is the geographic coverage of the image. The grid lines should be evenly distributed. The 4 corners of the image are represented by tiny green squares or crosses. They should be just on the outside edge.

The next 8 plots are extreme value plots. The first 4 are large extreme values, which is the maximum dynamic range + - 32. The next 4 are the number of small extreme values (0 + - 1). One plot is shown.
for each channel. For illustration, only the HH channel is shown. The beginning and end of an image in range usually have small values. However, you could also have a very bright nadir line. This particular image has land and sea. On the surface of the ocean, there is a minimum of large values; just what you would expect on the ocean. If something is peculiar or out of place, you could check it out. Saturation would also be evident with flattened peaks in the large extreme values.
Finally, there is the Doppler centroid plot. Ideally, the base should be approximately zero. The peaks are caused by contrast, not motion.

After the plots are save as a postscript file by Gnu2PS.sh, you don’t need to keep the Doppler and extreme files.

If you want to alter the density of the display grids, you can alter the values of `georef_grid_rg`, `doppler_grid_rg` and `satur_grid_rg` in the configuration file.
PRINTOUTS

Print out the **highlighted** files. The rest are optional

From the signal directory:

- `ancgain.ps`
- `plt.ps`
- `about.bite`
- `ancdat.txt`
- `ancendat.txt`
- `hh.hdr`
- `hv.hdr`
- `vh.hdr`
- `vv.hdr`

From the pro directory:

- `show1.plt.ps`
- `show2.plt.ps`

From the noise directory:

- `noiseimage.jpg`

From the precal directory:

- `polgasp.ps`
- `s2.tiff`
- `asar#_l#p#_calibrators.jpg`
- `asar#_ARCname_phaseplot.jpg`
- `asar#_CORNERname_phaseplot.jpg`

From precal/genCALresults

- `about`

From the calcheck / cal directory:

- `s2.tiff`
- `hhrc.hdr (abridged)`
- `chasp1c.conf`
- `test.plt.ps`

From your PC:

- **Image Parameter Sheet**
- **Calcheck spreadsheet** (calibration lines only)

*About the abridged file*

The regular COASP header file is very long because of all the georeferencing. Copy this to a file with a different name (e.g. `l2p3short`). Edit the file by removing information, but be careful. You want to keep the 4 corner coordinates and you want to keep the altitude.
OTHER UTILITIES

TransposeCoasp and transposeChasp

These programs do what they say. They will take a standard COASP image and transpose it so that it looks like a PolGASP image. COASP and CHASP images have a large vertical dimension. This is the azimuth and it runs from top to bottom. The transposed image has a large horizontal dimension and the azimuth runs from left to right. The header files are also changed. If an image were called l3p9hh.rc, after transpose it would be called l3p9transposedhh.rc. The original files are left intact. The example shown below is repeated for each channel.

**transposeCoasp**

**************************
Transpose COASP Image
**************************

Enter image base filename (ex. 'l1p2'): l3p9

Transposing HH channel...

***********************
Base filename: l3p9
Channel hh
Image name: l3p9hh.rc
Number samples: 4028
Number lines: 8192
Number blocks: 0
Number leftover lines: 8192
New img name: l3p9transposedhh.rc

**Extraction Programs**

The MATLAB program extRCdata1_4_4.m that extracts COASP data and places it in a MATLAB matrix has already been discussed (page 52 - 54). This program has 2 sisters, extPolData1_4_3.m and extRawData1_4_1.m. They are for extracting PolGASP and RAW signal data respectively. They have the same basic execution as extRCdata1_4_4.m.

There are also C programs that extract portions of image files but do not store them as matrix files but leave them in their native format. There is extRCdata2 and extPolData for COASP and PolGASP data. Their operation is straightforward.

A good one for RC data is extrc. Type it in and you get this response.

pigpen 56% extrc
Usage: extrc -f <base_name> [-a <first_line>] -A <lines> [-r <first_sample>] [-R <samples>]
( -f to specify the input filename base e.g. l2p2 )
( -a to specify the first range line for extraction i.e. azimuth offset )
( -A to specify the number of range lines to extract i.e. new azimuth length )
( -r to specify the first range sample for extraction i.e. range offset )
( -R to specify the number of range samples to extract i.e. new range length )
( -h|H|? to print this info )
Failed in getCommandLineArgs.
It is self-explanatory. An example line is:

```
extrc -f 12p3 -a 26000 -A 30000 -r 1 -R 4096
```

If you specify a number of range lines > the number of range lines in the file, it defaults to the number of range lines in the file.
SAR images of moving objects can be challenging to process and interpret. One objective for SAR imagery is ship detection. Ships in open water are usually moving and this causes problems with focusing and can create azimuth ambiguities. CHASP was developed to optimize focusing of the ships and to determine velocity and heading. It does this by working on a small image “chip”, usually 256 range lines by 8192 azimuth lines.

*NOTE*: At the time of this writing, all CHASP processing is done in a Linux environment. Also, all SAR images must be processed in COASP without azimuth compression and the Endianess must be compatible with the machine you are working on. This guide assumes you are in a Linux environment and that all required executables are in your bin directory.

If you don’t have the benefit of a ship detection program or procedure, it is fairly easy to spot ships visually with a bit of practice. In order to get an image, use the following command:

**ShowImg.sh l#p#hh rc**

The image will be long and narrow. Grab an edge and stretch out the image. It could look something like this.
You won’t see any azimuth ambiguities in an image without azimuth compression. All ships generally appear as long thin objects. An exception is the ship in the centre of this image.

A ship detection program would have given you coordinates. To do it manually, set your cursor over the centre of any ships. Click the centre button of the mouse and the coordinates will appear at the top or bottom of the image. Record these values.

As mentioned, the ship chip should be 256 range lines by 8192 or 4096 azimuth lines. The 256 range lines can vary and sometimes must vary when you want to exclude another ship or land. The azimuth lines must be 4096 or a multiple of 4096 or CHASP will not work. An easy way to find the chip coordinates is the shell script `ezeshipchip`. The arguments are the centre range line and the centre azimuth line. As an example:

```
ezeshipchip 1914 13344
Range: 1786 - 2041
Azimuth: 9248 - 17439 (8K)
Azimuth: 11296 - 15391 (4K)
```

The results are the coordinates to use for the chip, including results for 4096 azimuth lines, if you wish to use that. These values must be entered into the configuration file.

CHASP cannot be run without a configuration file. Certain parameters such as Doppler centroid (DC) and velocity offsets are changed automatically as CHASP progresses. For a template, take an existing configuration file, rename it and edit it. Keep the extension “.conf”. This is an example:

```
; WorkOrder
;
inpu$\text{data_{file}} 122p2ext

\text{channels} 4

\text{prf_{file}} 122p2prf.mat

\text{first\_line} 1

\text{last\_line} 4096

\text{first\_sample} 1

\text{last\_sample} 256

\text{geoid\_over\_ellipsoid} 0

\text{reference\_range} 1

\text{spread\_loss} 0

\text{georef\_grid\_rg} 0

\text{doppler\_grid\_rg} 64

\text{azimuth\_focus} No

\text{azimuth\_align} No

\text{2D\_frame} No

\text{compensate\_radial} No

\text{window\_shape} 2.8

\text{multi\_look} No

\text{coherence} No

\text{mag\_coherence} No

\text{norm\_coherence} No

\text{spatial\_size} No

\text{pgm\_out} No

\text{pgm\_floor} -40.000000000

;

; Adaptive algorithm
;

\text{auto\_focus} No

\text{range\_migration} No

\text{auto\_fit} No

\text{track\_range}

\text{track\_azim}
```
bunch_a
bunch_r

phase_order 4
auto_correct No

prf_over_v
relative_dc_offset 0.2499999851
relative_v_offset 0.0062380214
doppler_looks 1
full_bandwidth 0.8000000119
look_bandwidth 0.8000000119
look_offset 0.0000000000
look_overlap 0.0000000000
correlate_looks_len 0
locate Yes
measure_contrast No

window_range 32
window_azim 256
profile No

System parameters

wavelength 0.0565646000
range_shift_hh 0
range_shift_hv 0
range_shift_vv 1.25
range_shift_vh 1.25
azimuth_shift_hh 0
azimuth_shift_hv 0
azimuth_shift_vv 0.5
azimuth_shift_vh 0.5
range_slant_spacing 4.0
saw_delay 13.289e-06
boresight_offset_hs -1.585
boresight_offset_vs -1.585
boresight_offset_hp -3.330
boresight_offset_vp -3.330
antenna_ang_min -73.0
antenna_ang_inc 1.0
azimuth_beamwidth 3.05

Processor parameters

processors 1
block_lines 4096
parts 1
channels_together 1
interp_size 11
interp_fractions 16
alpha 0.85
change_anc_endian Yes
change_input_endian Yes
change_output_endian No

Exchange parameters

phase_term_0 0.000000e+00
phase_term_1 0.000000e+00
phase_term_2 -7.478885e-05
phase_term_3 -7.263318e-08
phase_term_4 4.229004e-11
range_migration No
measure_contrast No
The actual file is longer as there are several multilook parameters at the end. Don’t be intimidated. For basic CHASP, most of the parameters don’t have to be changed. Before you begin CHASP, you must change the following.

**input_data_file**
Use the correct base file name and remove “ext”.

**prf_file**
The right PRF file must be in your working directory and be indicated in this file.

**change_output_endian**
Make sure this is “No”. Otherwise you won’t see your results.

---

**WRAP**

CHASP is initially run using an executable called “wrap”. The syntax is:

```
wrap -c configuration file
```

```
wrap -c 370S2.conf
```

You are given the processing status from the COASP header file. If the file has any of these parameters:

+SHIFT+FOCUS+UNMOC,

it can’t be used for CHASP processing

Is COASP status +MOC+RAD(3)+ANT+CAL ok? (y/n): y

Next, the chip parameters are entered. Just remember that the first and last lines are the **azimuth** parameters. The first and last samples are the **range** measurements.

Chip Parameters for l1p6

```
1. First line           (    1)  
2. Last line            ( 8192) 
3. First sample         (    1)  
4. Last sample          (  256) 
```

Enter # to modify (0 exits to review) : 1

Enter first line: **9248**

Chip Parameters for l1p6

```
1. First line           (    1)  
2. Last line            ( 8192) 
3. First sample         (    1)  
4. Last sample          (  256) 
```

Enter # to modify : **2**

AND CONTINUING ON

```
1. First line           ( 9248) 
2. Last line            (17439) 
3. First sample         ( 1786) 
```
4. Last sample (2041)

Enter # to modify (0 exits to review) : 0
memBytes 67108864

This tells you that the program has successfully extracted an image chip. If you get an error message, you must correct the problem before continuing. You will end up with an image chip that will be displayed on your monitor. It will look a lot like this.
Grab an edge and stretch it out to ensure that there are no other ships or any land, especially in the corners of the chip. Sigma nought measurements of the ocean are made in the corners.

Is this a good chip? (y/n): y
After confirming the chip, CHASP will be processing only the extracted chip. All the files will have “ext” in their file names (e.g. l1p6exthchasp.hdr). In the configuration file, the input_data_file will appear as l1p#ext. If at any time after this point, you interrupt processing, you can pick up where you left off. Just remember to add –ext to the wrap command line.

```
wrap -c 370S2.conf –e
```

Otherwise, you will have to start at the beginning of the process. The next step is to determine the offset of the doppler centroid. If you already know this, you can select “1” and carry on.

```
Doppler Parameters for l1p6ext

1. Best relative DC offset   ( 0.0000)
2. Show ghost results       (n)

Enter # to modify : 2
memBytes 67108864
```

This message will come up many times during CHASP processing, since CHASP is executed many times. Always be alert for an error message. There are “break” points in the program so that you can go back to where the error occurred and not have to go through the entire program again.

The program will display 2 images of the chip in the HH and HV channels. Often you can see a ghost image of the ship. At the same time, it will display an intensity profile of the chip in all 4 channels. You will see a variation of this. This is an HH channel image. The HV image will have the same format.
You will then be asked this question. From the images and the profile, it is obvious that there is an image ghost (ambiguity) present. The answer will be “y”. The –0.5 is the Doppler centroid offset. This offset proceeds in increments of 0.1 until it reaches 0.5. You must answer honestly at each increment. There could be times that the ghost cannot be seen in the images. That is when you rely on the profile. When the profile appears like this, you answer “n”. The profile may not always appear like this. There may be 2 small symmetrical or nearly symmetrical peaks on both sides of the main peak. In that case you would also answer “n”.

Is there a nonsymmetric ghost 200.7 away for -0.50? (y/n): y
memBytes 67108864

Is there a nonsymmetric ghost 200.7 away for -0.40? (y/n): y
memBytes 67108864

AND CONTINUING ON

Is there a nonsymmetric ghost 200.6 away for 0.10? (y/n): n
memBytes 67108864

AND CONTINUING ON

Is there a nonsymmetric ghost 200.7 away for 0.50? (y/n): y

You are now given the option of repeating the profiles if you are not sure of your result or you would like to see the images with more or less intensity.
Would you like to repeat this with a lower threshold? (y/n): **n**
Would you like to repeat this with a higher threshold? (y/n): **n**
Would you like to repeat this with the same threshold? (y/n): **n**

1 8192 256 in l1p6exthhchasp.hdr

The Doppler centroid offset is probably the value determined above. But there is the possibility that the offset is the measured value plus 1.0 or the measured value minus 1.0. The next part of the program uses several statistical parameters to determine the best value.

It begins with a rotated image of the chip and the ship with the Doppler centroid applied. You are asked if the ship is straight. It usually is. In order to move on, the **Track range** and **Width range** must be selected and answered. This is the minimum that can be done. If the ship is close to the centre in the azimuth direction and there is good contrast with the background, you can enter –1 for both values. This allows the program to determine where the ship is and perform measurements.

*NOTE*: Always enter 4 for the **Phase order**.

If there is poor contrast, it is best to measure the track range. Place the cursor on the image and click the centre button of the mouse to get the mean range. The range width is the total number of range pixels you want measured. Usually, this is between 10 and 50 depending on the size of the ship. Sometimes, to avoid other ships, the ship will end up near the end of the chip. If this is the case, then **Track azimuth** should be entered. It is usually not necessary to enter the azimuth width.

*NOTE*: If you enter parameters other than –1, these same parameters should be repeated for this same ship for the rest of the program. These values should also be used in view123, which is run after this program.

Is this track straight? (y/n): **y**

Tracking Parameters for l1p6ext

1. Track range: ( -1)
2. Width range: ( -1)
3. Track azimuth: ( -1)
4. Width azimuth: ( -1)
5. Phase order: ( 0)

Enter # to modify (-1 breaks): **1**

Enter target range (-1 to allow search): **-1**

Tracking Parameters for l1p6ext

1. Track range: ( -1)
2. Width range: ( -1)
3. Track azimuth: ( -1)
4. Width azimuth: ( -1)
5. Phase order: ( 0)
Enter # to modify (-1 breaks): 2

Enter target width in range (-1 to allow search): -1
Tracking Parameters for llp6ext
1. Track range ( -1)
2. Width range ( -1)
3. Track azimuth ( -1)
4. Width azimuth ( -1)
5. Phase order ( 0)

Enter # to modify (0 exits menus, -1 breaks) : 5

Enter phase order for fitting (max is 4): 4
Tracking Parameters for llp6ext
1. Track range ( -1)
2. Width range ( -1)
3. Track azimuth ( -1)
4. Width azimuth ( -1)
5. Phase order ( 4)

Enter # to modify (0 exits menus, -1 breaks) : 0
memBytes 67108864

A graphic representation of the azimuth walk will be displayed. You have to decide if it is linear and if it is consistent. If it is non-linear, there is no way to accurately determine the ship’s velocity. Sometimes, it is a judgment call. The example below is non-linear but is consistent.
Azimuth path assessment

1. Linear  (y)
2. Consistent  (y)
3. Repeat  (n)

Enter # to modify (-1 breaks): 1

Is this path linear :  (y/n): n
Azimuth path assessment

1. Linear  (n)
2. Consistent  (y)
3. Repeat  (n)

Enter # to modify (-1 breaks): 2

Are all tracks consistent :  (y/n): y
Azimuth path assessment

1. Linear  (n)
2. Consistent  (y)
3. Repeat  (n)

Enter # to modify (0 exits menus, -1 breaks) : 0

Next, is a graphic display of the range walk. You must determine if it is sloped. If there is a slope, use the mouse to get the coordinates of each end. If it slopes from upper left to lower right, it is a negative slope. If the slope is from lower left to upper right, it is a positive slope. A negative slope must be indicated with a minus sign before the slope value in the Walk. Sometimes, there is more than one slope. You could average them out or measure the main one. Again, a judgment call.
Range path assessment

1. Sloped          (n)
2. Walk            (0.00)
3. Repeat          (n)

Enter # to modify (-1 breaks) : 1

Is this path sloped: (y/n): n

Range path assessment

1. Sloped          (n)
2. Walk            (0.00)
3. Repeat          (n)

Enter # to modify (-1 breaks) : 2

Enter range walk (include sign): 0

Range path assessment

1. Sloped          (n)
2. Walk            (0.00)
3. Repeat          (n)

Enter # to modify (0 exits menus, -1 breaks) : 0

Exit this menu and let the program run. It can take several minutes. The following messages will appear.

memBytes 67108864
memBytes 67108864
memBytes 67108864
memBytes 67108864
memBytes 67108864

Next, a series of correlated peaks will appear for each channel. You have to decide whether they are aligned or not. This takes some practice. The peaks are never perfectly aligned. With practice, you will know when to say yes or no. Don’t worry too much if you think you may have been mistaken. The answers have some influence, but they do not uniquely determine the final outcome.
Are peaks aligned? (y/n): y
Are peaks aligned? (y/n): y
memBytes 67108864
memBytes 67108864
memBytes 67108864
As mentioned above, the Doppler centroid offset can differ from the initially determined value by 1.0 or –1.0. The above section is repeated two more times for a Doppler centroid with offsets of –1.0 plus the initially determined value and 1.0 plus the initially determined value. The procedure is the very same as detailed above, starting from:

Is this track straight? (y/n): y

Tracking Parameters for l1p6ext

At the final end of the processing, this table will come up:

Final Parameters for l1p6

1. DC offset                   (  0.10)
2. DC ambiguity                (     0)
   range slant score           (     1)
   look correlation HH score   (     1)
   look correlation HV score   (     1)
   look correlation VV score   (     1)
   look correlation VH score   (     1)
   contrast HH score           (     1)
   contrast HV score           (     1)
   contrast VV score           (     1)
   contrast VH score           (     1)
   entropy HH score            (     1)
   entropy HV score            (     1)
   entropy VV score            (     1)
   entropy VH score            (     1)
3. V offset                    (0.0079)
   track V offset              (-0.0002 +/- 0.0025)
   multi-look V offset         (-0.0240 +/- 0.0531)
   contrast HH V offset        (0.0079)
   contrast HV V offset        (0.0079)
   contrast VV V offset        (0.0079)
   contrast VH V offset        (0.0079)
   entropy HH V offset         (0.0079)
   entropy HV V offset         (0.0079)
   entropy VV V offset         (0.0079)
   entropy VH V offset         (0.0079)

Enter # to modify (0 exits menus) : 0
It is a good idea to save this table. The easy way is to copy and paste it into a created document. If not, record the DC offset, the DC ambiguity and the V offset. Ideally, the values under DC ambiguity should all be 1. If there are zeroes, make a note of them. If there is more than one zero, it could be the result of bad data or bad processing. If you forget to take notes, the relevant values can be found in wrap.log.

You will end up with a number of header files with the format: l#p#ext??chasp.hdr.0
l#p#ext??chasp.hdr.1
l#p#ext??chasp.hdr.-1

The files contain several parameters and statistics about the processed image. The relevant files will have the same extension as the DC ambiguity. All the header files can be automatically saved later on. The last step in wrap is the program asking if you want to delete all the images that have come up during the program and are covering your monitor. Use the program XV if you really want to save any of them. Then answer “y”.

Clean all xv? (y/n): y
Ciao. Program finished successfully.
**VIEW123**

View123 is a program to create the best possible focus of a CHASP processed image. It is used mainly for ships. There will be 3 images displayed. Note that these images have square pixels to create the best possible display. As part of the procedure, save the “before” and “after” images of the ship. It is not essential to do this. If you do not save any of the images, only the final image will be saved at the end. It will be in the native CHASP format and have the extension “.ppm”. If you want to save the images, one good way is to right click on the image after it appears. It will bring up the XV GUI. Then click on **save** and chose any of the formats that **XV** supports.

There are 2 false colour options that can be used with view123. The upper and lower case of the letter “P” designate them. (–p and –P). If you don’t use either modifier, the images will be black and white. -P will give you the most colour. –p will give you less colour but more contrast. You can decide which is best for the features you want to see. It doesn’t take long to run view123. If you aren’t happy with the first results, just try again.

After checking the image status, it displays a nominal image of the ship before any CHASP processing was done.

```
view123 -c 370S2.conf -P
```

Is COASP status +MOC+RAD(3)+ANT+CAL ok? (y/n): y
memBytes 67108864
As is often the case, a ghost image (azimuth ambiguity) is also visible. This is a large ship. So some detail is visible, but not well focused.

Got nominal images? (y/n): y
memBytes 67108864

In the next image, the Doppler centroid and velocity offset will be applied.

Got DC&DV images? (y/n): y

Before the 3rd image, you will be asked form the same parameters as in WRAP. If you applied any parameters other than the defaults, they should be applied here. The optimum azimuth walk and range walk chosen for this image will be displayed. If the azimuth walk is not linear, an accurate velocity of the ship cannot be calculated. The azimuth walk is averaged out and the resulting line function is applied to the image. This usually gives the best result, but not always. Sometimes there is very little difference between the last 2 mages.

Tracking Parameters for 1lp6ext

1. Track range (   -1)
2. Width range (   -1)
3. Track azimuth (   -1)
4. Width azimuth (   -1)
5. Phase order (    4)

Enter # to modify (-1 breaks): 1
You will be asked to rate the 3 images. They can be rated from 1 (bad) to 4 (excellent). Sometimes, two images will be equal and require the same score. The image that will get used is the one that requires the least amount of processing. The configuration file will be changed depending on your selection.

### Rating for l1p6ext

<table>
<thead>
<tr>
<th>1. Nominal</th>
<th>(0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH Contrast</td>
<td>(20.82)</td>
</tr>
<tr>
<td>HV Contrast</td>
<td>(40.04)</td>
</tr>
<tr>
<td>VV Contrast</td>
<td>(13.52)</td>
</tr>
<tr>
<td>VH Contrast</td>
<td>(32.72)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. DC&amp;DV offset</th>
<th>(0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH Contrast</td>
<td>(14.95)</td>
</tr>
<tr>
<td>HV Contrast</td>
<td>(21.16)</td>
</tr>
<tr>
<td>VV Contrast</td>
<td>(10.90)</td>
</tr>
<tr>
<td>VH Contrast</td>
<td>(22.30)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Motion compensated</th>
<th>(0)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH Contrast</td>
<td>(19.86)</td>
</tr>
<tr>
<td>HV Contrast</td>
<td>(37.09)</td>
</tr>
<tr>
<td>VV Contrast</td>
<td>(16.54)</td>
</tr>
<tr>
<td>VH Contrast</td>
<td>(37.01)</td>
</tr>
</tbody>
</table>

Enter # to modify (-1 breaks): 3

Enter grade for 'new way' (1 bad, 2 fair, 3 good 4 excellent): 3

AND CONTINUING ON

### Rating for l1p6ext

<table>
<thead>
<tr>
<th>1. Nominal</th>
<th>(2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH Contrast</td>
<td>(20.82)</td>
</tr>
<tr>
<td>HV Contrast</td>
<td>(40.04)</td>
</tr>
<tr>
<td>VV Contrast</td>
<td>(13.52)</td>
</tr>
<tr>
<td>VH Contrast</td>
<td>(32.72)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. DC&amp;DV offset</th>
<th>(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH Contrast</td>
<td>(14.95)</td>
</tr>
<tr>
<td>HV Contrast</td>
<td>(21.16)</td>
</tr>
<tr>
<td>VV Contrast</td>
<td>(10.90)</td>
</tr>
<tr>
<td>VH Contrast</td>
<td>(22.30)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>3. Motion compensated</th>
<th>(3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>HH Contrast</td>
<td>(19.86)</td>
</tr>
<tr>
<td>HV Contrast</td>
<td>(37.09)</td>
</tr>
<tr>
<td>VV Contrast</td>
<td>(16.54)</td>
</tr>
<tr>
<td>VH Contrast</td>
<td>(37.01)</td>
</tr>
</tbody>
</table>

Enter # to modify (0 exits menus, -1 breaks): 0

Clean all xv? (y/n): y

Ciao. Program finished successfully.
This table can also be saved. Copy and paste it into the same document as the Final Parameters from CHASP processing.

By this time, you will have a great many files in your working directory. Which one you want to save and / or keep is entirely up to you.

At DRDC there is a directory structure where all the results of CHASP processing are kept. Gathering together all the relevant files into an easily accessible comprehensive data structure is a complex multi-step task. To avoid errors and a lot of keystrokes, shell scripts were created. This guide will explain what happens with the shell scripts used at DRDC. Some of the files and results are for specific purposes at DRDC. You can alter the shell script or create your own if you are going to do this often.

**ezechaspro**

The main script is called **ezechaspro**. There are 3 arguments: the line number, pass number and the DC ambiguity number.

*NOTE*: When the DC ambiguity number is –1, the relevant header files will have the extension of “.hdr.-1”. The file names must be changed so that the last extension is a positive number (e.g. .hdr.2). The shell script will not accept a negative number as an argument. The number can be changed back when the script is complete.

**ezechaspro 1 6 0**

Answer the prompts.

Enter image asar number (e.g. 358):
370
Enter image Part number (e.g. 3 for P3):
0
Enter Ship Name or Number (no S before number):
1
What is the name of the configuration file?
370S1.conf

Then, the configuration file will come up in a text editor. Editing is optional. This is useful for changing the format of the output file to big-endian. At this point, another shell script “**Keep.sh**” is called. It selectively gathers and stores all the profiles and plots generated by WRAP. It stores the log files of WRAP and view123. The current configuration file is stored along with the previous configuration files used before view123 was run. This file is labeled as “filename.conf.wrap”. If you want to run view123 again, or run CHASP with both Doppler and velocity offsets, this is the file to use. Just remove the “wrap” extension. All the header files including header files with all DC ambiguities are stored.

**Keep.sh $tarname $cname**;
Headers found: 4 4 4
Profile plots found: 44
Track plots found: 4+4 4+4 4+4
Multi-look plots found: 20 20 20
Doppler plots found: 4 4 4
Archiving:
A long list of filenames will scroll by.

wrap.log
view123.log
370S1.conf
370S1.conf.wrap
l1p6exthhchasp.hdr.0
l1p6exthhchasp.hdr.1
l1p6exthhchasp.hdr.-1
l1p6exthvchasp.hdr.0
l1p6exthvchasp.hdr.1
l1p6exthvchasp.hdr.-1
l1p6extvhchasp.hdr.0
l1p6extvhchasp.hdr.1
l1p6extvhchasp.hdr.-1
l1p6extvvchasp.hdr.0
l1p6extvvchasp.hdr.1
l1p6extvvchasp.hdr.-1
track
prof_-0.00_0
prof_-0.00_1

AND CONTINUING ON

doppler_2.1
doppler_2.-1
doppler_3.0
doppler_3.1
doppler_3.-1

The directory files are listed and the script pauses. Wait for the cursor to start flashing and hit return. You will see the following. A postscript file and a new directory are created. Files are copied and moved to the new directory. The files in the new directory are displayed.

Done! See file track1.plt.ps
370_P0_S1_results.tar.gz  370S1.conf  370_S1_summary
370_P0_S1_results.tar.gz  l1p6exthh.rc  l1p6exthvchasp.hdr.0
370S1.conf  l1p6exthh.rc  l1p6exthvchasp.hdr.0
370_S1_summary  l1p6exthhchasp.hdr.0  l1p6exthv.rc
1lp6after_big.jpg  l1p6exthv.rc  l1p6extvhvchasp.hdr.0
1lp6before_big.jpg  l1p6exthvrc.hdr  l1p6prf.mat
1lp6DCDV_big.jpg  1lp6ext.ppm  track
1lp6ext_before.jpg  l1p6extvhchasp.hdr.0  track1.plt.ps
1lp6exthhchasp.hdr.0  l1p6extv.ch
memBytes 67108864
off

The script will pause for a few seconds while CHASP is run on the files copied into the new directory. A shell script, ezeheadisp is called to extract information for a special purpose file. Another shell script, ezehead2tarX is called to open the tar file and add the header files from the CHASP processing to the tar archive.

SEE l1p6_header
Enter name of tar file (not gzipped)
370_P0_S1_results.tar
l1p6exthhchasp.hdr
l1p6exthvchasp.hdr
l1p6extvhvchasp.hdr
chasp2.log
The program **transposeChasp** is called to transpose the CHASP image for other programs.

```
************************************************
Transpose CHASP Image
************************************************

Enter image base filename (ex. 'l1p2'): l1p6ext

Transposing HH channel...

************************************************
Base filename: l1p6ext
Channel hh
Image name: l1p6exthh.chasp
Number samples: 256
Number lines: 8192
Number blocks: 0
Number leftover lines: 8192
New img name: l1p6exttransposedhh.chasp

AND CONTINUING ON

transposeCoasp finished.

The files are displayed again. Two more directories are created and files are moved, removed or copied. The CHASP files are all in the main directory. Under this is a directory called “rcfiles”. It contains the extracted COASP files that the CHASP files were created from. It also contains the configuration file, so that CHASP can be run again under different parameters. The other directory is called “info”. It contains the tar archive, images of the ship, header and summary files. The PRF file for the whole line/pass is also saved and stored. This directory structure is then moved to specific locations in our network so that they are kept together and can be accessed easily.

Pressing return two more times should cause the script to complete.

*NOTE*: If **transposeChasp** does not run, it normally means that CHASP itself did not run properly. You will have to look into the chasp2.log file to find what happened and correct it. Before running the script again, check carefully to see where all the files are. Temporary directories are created and some critical files will be there. Bring all the files you need into your main working directory before running the shell script again.
References

<table>
<thead>
<tr>
<th>1. ORIGINATOR</th>
<th>(The name and address of the organization preparing the document. Organizations for whom the document was prepared, e.g. Centre sponsoring a contractor's report, or tasking agency, are entered in section 8.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Terry M. Potter Consulting</td>
<td></td>
</tr>
<tr>
<td>206 Chipmunk St., Limoges, Ont. K0A 2M0</td>
<td></td>
</tr>
<tr>
<td>2. SECURITY CLASSIFICATION</td>
<td>(Overall security classification of the document including special warning terms if applicable.)</td>
</tr>
<tr>
<td>UNCLASSIFIED</td>
<td></td>
</tr>
<tr>
<td>3. TITLE</td>
<td>(The complete document title as indicated on the title page. Its classification should be indicated by the appropriate abbreviation (S, C or U) in parentheses after the title.)</td>
</tr>
<tr>
<td>CV-580 Polarimetric SAR data processing software: A user's guide to COASP and CHASP</td>
<td></td>
</tr>
<tr>
<td>4. AUTHORS</td>
<td>(last name, followed by initials – ranks, titles, etc. not to be used)</td>
</tr>
<tr>
<td>Potter, T.M.</td>
<td></td>
</tr>
<tr>
<td>5. DATE OF PUBLICATION</td>
<td>(Month and year of publication of document.)</td>
</tr>
<tr>
<td>August 2007</td>
<td></td>
</tr>
<tr>
<td>6a. NO. OF PAGES</td>
<td>(Total containing information, including Annexes, Appendices, etc.)</td>
</tr>
<tr>
<td>12</td>
<td></td>
</tr>
<tr>
<td>6b. NO. OF REFS</td>
<td>(Total cited in document.)</td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>7. DESCRIPTIVE NOTES</td>
<td>(The category of the document, e.g. technical report, technical note or memorandum. If appropriate, enter the type of report, e.g. interim, progress, summary, annual or final. Give the inclusive dates when a specific reporting period is covered.)</td>
</tr>
<tr>
<td>Contract Report</td>
<td></td>
</tr>
<tr>
<td>8. SPONSORING ACTIVITY</td>
<td>(The name of the department project office or laboratory sponsoring the research and development – include address.)</td>
</tr>
<tr>
<td>Defence R&amp;D Canada – Ottawa</td>
<td></td>
</tr>
<tr>
<td>3701 Carling Avenue</td>
<td></td>
</tr>
<tr>
<td>Ottawa, Ontario K1A 0Z4</td>
<td></td>
</tr>
<tr>
<td>9a. PROJECT OR GRANT NO.</td>
<td>(If appropriate, the applicable research and development project or grant number under which the document was written. Please specify whether project or grant.)</td>
</tr>
<tr>
<td>9b. CONTRACT NO.</td>
<td>(If appropriate, the applicable number under which the document was written.)</td>
</tr>
<tr>
<td>W7714-0207-16/001/SV</td>
<td></td>
</tr>
<tr>
<td>10a. ORIGINATOR'S DOCUMENT NUMBER</td>
<td>(The official document number by which the document is identified by the originating activity. This number must be unique to this document.)</td>
</tr>
<tr>
<td>10b. OTHER DOCUMENT NO(s).</td>
<td>(Any other numbers which may be assigned this document either by the originator or by the sponsor.)</td>
</tr>
<tr>
<td>DRDC Ottawa CR 2007-188</td>
<td></td>
</tr>
<tr>
<td>11. DOCUMENT AVAILABILITY</td>
<td>(Any limitations on further dissemination of the document, other than those imposed by security classification.)</td>
</tr>
<tr>
<td>Unlimited</td>
<td></td>
</tr>
<tr>
<td>12. DOCUMENT ANNOUNCEMENT</td>
<td>(Any limitation to the bibliographic announcement of this document. This will normally correspond to the Document Availability (11). However, where further distribution (beyond the audience specified in (11) is possible, a wider announcement audience may be selected.)</td>
</tr>
<tr>
<td>Unlimited</td>
<td></td>
</tr>
</tbody>
</table>
For several years, DRDC Ottawa has been acquiring and processing airborne polarimetric radar imagery from the Environment Canada Convair-580 aircraft. This guide describes how to use the various software packages to take polarimetric signal data from an 8-mm mammoth tape and create useful end-products for further analysis. This process uses software written for DRDC and software originally written by CCRS. They are combined into a logical data flow and there are several shell scripts that speed up the processing. A directory structure is described to make the greatest use of these shell scripts.

The processing begins by describing how to get the signal data off the tape, and then separating the image data from the noise data and downloading 4 different polarization channels. The data are checked and the noise data evaluated. Other required data sources include inertial navigation data from the aircraft and a differentially processed GPS data set. These are processed and checked with quality control software. Then the main processor, COASP is run to process the noise and then produce an image over a calibration site. This is followed by collection of data from the available calibration devices, data quality assessment, and use of a program that produces the calibration parameters. All the calibration parameters are then verified. Once an image is produced, a program called CHASP is used for a detailed analysis of moving ocean targets. CHASP consists of two main software packages; there is a complete description of how to use them. All the processes are described in a logical order.

**PolSAR, Data processor**