DOING YOUR OWN DATA REDUCTION

This chapter is about redoing the basic data reduction step which corrects event positions for aspect and detector distortions, applies the wavelength mapping to Spectrometer data, and remaps events to sky positions in Deep Survey data. This is done by using the event pipeline task **euvred.cep** and the EUVE DS/S reference data to convert the time-ordered event list in *table0.tab* to one or more QPOE files.

There are two main reasons why GO's will want to rerun **cep**. First, if there are problems with unpacking the FITS encoded QPOE files in your IRAF environment, running the pipeline on your own IRAF system will produce a set of locally compatible QPOE files. Second, as new knowledge about the operation of the instruments becomes available, some GO's may profit by reprocessing their data with a new reference data set or revised version of the pipeline. The event pipeline is now capable of producing QPOE files for any or all of the DS/S detectors: short, medium, and long wavelength Spectrometers, and the Deep Survey.

New versions of the reference data are announced to all GO's, and may be obtained or from the EGODATA directory in the CEA *ftp* site. Some versions of the reference data may require a newer version of the IRAF/EUV software than that supplied on the GO data tape. The software and installation instructions are available in the *ftp* directory */pub/software/euv1.#.#*.

If the reference data changes, do I also need new software?

New versions of the EUVE reference data do not automatically require a newer version of the IRAF/EUV software than that supplied on the GO data tape. The message announcing the new reference data version should also indicate which version(s) of the software can be used. The IRAF/EUV software and the compatible versions of the reference data will not in general have the same version number. If you are unsure what software version to use with a particular data set, it is usually best to use the latest available. The latest software version can always be obtained from the CEA/EUVE *ftp* site, in the directory /pub/software/euv1.#.#.

5.1 Running The EUVE Event Pipeline, CEP

The task **cep** (for comprehensive event pipeline) implements the steps of wavelength correction and aspect correction, and makes QPOE files with two-dimensional linear world coordinate systems from the event list in *table0.tab*. It processes by reading each event as a line from the table, processing the event, and writing to a separate QPOE file for each of the detectors. **cep** uses aspect reports from the telemetry table *table1.tab* and instrument data from a reference data set. Each data tape comes with a copy of the reference data used to produce the products on the tape.

5.1.1 Preparation

Most of the time required of the user in running **cep** goes into preparation. Once the task starts, although it may take quite a while, the user need only check its progress occasionally. This section contains instructions and checklists for getting ready to run the pipeline.

cep needs to know the names and locations of several tables and the reference directory it will use. Each of the pointers it uses to get this information has a default value. If used with the standard directory setup described in Chapter 0 of the *EGO Center Data Products Guide*, the default values are correct, with one minor exception (noted below). This is the easiest arrangement to work with. Confirm that each of the required items below is in place before running **cep**, and that all files have been converted from FITS format to their internal IRAF formats[1].

Preparing to run **cep**

- ☐ The reference data should be in a subdirectory of the working directory called *reference*, and should include the file called *reference*. These data are placed in the proper directory when the tarfile of reference data is read from the data tape with **softools.rtar**.

  Make sure sufficient disk space is available. By default, **cep** will make a temporary file for each QPOE, in the working directory. Each file is the size of the finished QPOE, which can be 10 – 60 Mgb. All four DS/S files together roughly equal the size of *table0.tab*. So there must be at least twice this much space (for both at once) in the working directory. To use another location for the temporary files, search for lines in the *reference* file which have the form:

  ```
  TempDir ./
  ```

  and change them to point to the desired location, e.g.:

  ```
  TempDir path.to/lotsoroom/
  ```

- ☐ The file called *pipeline* should be moved from the *reference* directory, where it is normally placed by **rtar**, to the working directory where **cep** is to be run.

[1] See section 0.3 of the *Data Products Guide* for instructions on FITS conversion.

- The telemetry tables *table0.tab, table1.tab,* and *table7.tab* should be in a subdirectory of the working directory called *observation*. They are normally written in this location when read from the tarfile of observation/engineering files with **rtar**.
- The file called *observation* should be read from tape into the *observation* directory.
- The user should check the information in the table *catalog.tab* to make sure the target coordinates and units are correct for the current application.
- **cep** always creates QPOE files called *sw.qp, mw.qp, lw.qp,* and *ds.qp*. If these files already exist, it will either write over them or fail at the very end of processing (depending on how IRAF clobber options are set). Be sure to remove or rename old QPOE files from the working directory before running **cep**.

Figure 5-1 shows a schematic of the directory structure for running **cep**, with the required tables and ASCII files. Use this figure as a map to check the file locations before running **cep**.

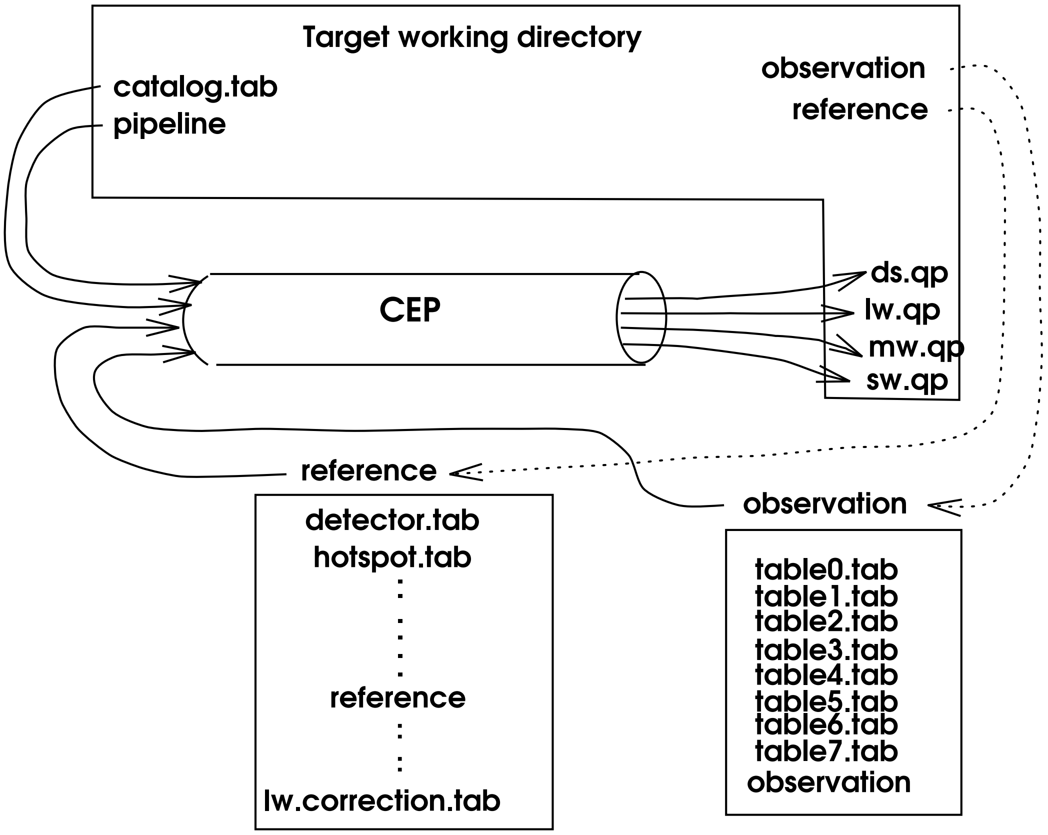

Figure 5-1: Schematic of directories and files for running the event pipeline **cep**. Existing versions of QPOE files should be moved from the working directory.

5.1.2 Processing scripts control cep

The parameter file for **cep** is shown in figure 5-2; it is deceptively simple. This complex task uses its two parameters as pointers to other ASCII files which contain descriptions and locations of the telemetry tables, the reference data and the subprograms **cep** uses at each stage of the reduction.

```
eu> lpar cep


database = "pipeline"      Processing text database name
pipeline = "defpipe"       Pipeline record name
   (mode = "al")
```

Figure 5-2: Parameters for the task **cep** with default values.

The first parameter, `database`, points to the file called *pipeline*. The file contains a number of pipeline scripts which can be used to control processing by **cep** and produce different kinds of QPOE files. The scripts may be specific to a certain version of the reference data, so it is important always to use the version of the pipeline file that was bundled with the reference data you are using. If you have not already done so, move the file to the working directory where **cep** will be run. The second parameter, `pipeline`, indicates *which* of the scripts in the file is to be used. In our example, `pipeline` is set to `defpipe`, which is an alias for the default GO processing pipeline.

The default processing script

cep will start reading file pointed to by `database` (usually *pipeline*) and will expect its processing directions to start with a line containing the value of `pipeline` (usually "defpipe"). The listing below shows the pipeline script used to process observations at the EGO Center. Comments have been omitted, and users may wish to read the actual file for more detail. Most GO's who run **cep** on their data will use this pipeline script. Notes on the other scripts in the *pipeline* file and their uses are given later in this section.

The three lines starting with `begin defpipe` under the banner, "Highest Level Indirection to Pipeline," define `defpipe` as a shorter name for the script `Spectrometer Wavelength Calibration Plots`, which begins under the banner "Wavelength Calibration Image." The first lines of the actual script tell **cep** where to find the observation data and reference data. The string `RefPath/Reference` must give the correct path to the *reference* file, which is a directory to of reference data files and processing commands. Likewise, the string `ObsPath/Observation` must be the path to the *observation* file, in the directory with the telemetry tables. These fields need not be changed if the user has created the standard directory and file setup shown in figure 5-1. To use some other location for the *reference* or *observation* directories and files, change the values of `RefPath`, `Reference`, `ObsPath`, and `Observation` in *pipeline*.

One can see the general flow of the processing by looking over the rest of the script, under `ProcessList`, Events are read from the telemetry; W,S, and Z pulse-heights are converted to detector X,Y coordinates[2]. Subpixel randomization is done to remove binning distortion effects (described in section 4.2.1 of the *EGO Center Data Products Guide*). After wavelength mapping and aspect correction, events are sorted using several `Copy` commands, and written to the QPOE files. **cep** translates each line of the `ProcessList` into lower-level processing commands using definitions in the *reference* file. These commands in turn point to the subroutines which are executed by **cep**. We will not describe pipeline scripts in any more detail here, though users can read section 4.2.1 of the *EGO Center Data Products Guide*, which describes the steps in the standard script.

```
######################################################################
#
#               Highest Level Indirection to Pipeline
#
######################################################################

begin   defpipe
        Type    Indirect
        Name    Spectrometer Wavelength Calibration Plots

######################################################################
#
#               Top Level Pipelines
#
######################################################################


#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
#
#                       Wavelength Calibration Image
#
#+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

begin   Spectrometer Wavelength Calibration Plots
        Type            Pipeline
        RefPath         reference
        Reference       reference
        ObsPath         observation
        Observation     observation
        ProcessList
                Read Spectrometer Photon Events
                Convert Spectrometer WSZ to XY
                Copy Spectrometer X to S
                Copy Spectrometer Y to Z
```

[2] Events are telemetered in WSZ format, starting on March 1, 1995. Prior observations were in XY format, and did not require the conversion.

```
Randomize Spectrometer X SubPixel Position
Randomize Spectrometer Y SubPixel Position
Correct Spectrometer Wavelength
Source Spectrometer Aspect
Aspect Spectrometer Correction
Correct Spectrometer Wavelength Aspect
Correct Spectrometer Imaging Aspect
Copy Spectrometer S to TYPE
Copy Spectrometer Z to FLAT
Copy Spectrometer X to W
Copy Spectrometer Y to S
Write Spectrometer Photon Events
EOL
```

Target coordinates for aspect correction

The target position on the sky is needed to perform aspect correction. The position is specified in a table called *catalog.tab*. When the pipeline gets to the step `Aspect Spectrometer Correction` it looks up the table in the section of the pipeline file starting at the line `begin SourcePosition`, listed below. Under `SourcePosition`, the entry `Catalog` specifies the table name, which happens to be *catalog*. IRAF automatically supplies the *.tab* extension. **cep** is instructed to look in the first row (`Column` = ROW and `Value` = 1) and get the RA from the column "RA" and the Dec from the column "DEC". The RA can be in either hours or degrees; **cep** looks at the units field of the table column to determine which is used.

```
######################################################################
#
#                       Source Position Record
#
######################################################################

begin   SourcePosition
        Type    Starcat
        Catalog catalog
        Column  ROW
        Value   1
        ColumnRA        RA
        ColumnDEC       DEC
```

A copy of the catalog table used in EGO Center processing is supplied on each data tape, in the fourth tarfile. Using **tlcol** on this copy of *catalog.tab* will show the columns, formats, and units that are required in the file, as well as the content seen with **tread**. An alternate name, target priority and HISTORY card are shown in the header, and the values are printed after the column formats:

```
tt> tlcol catalog
```

```
# catalog.tab
ID              CH*15       %-15s  ""
RA              R          %15.7g  degrees
DEC             R          %15.7g  degrees
exposure        I            %11d  ""
priority        CH*4         %-4s  ""
name            CH*15       %-15s  ""
HISTORY  t  Created Thu 17:21:36 08-Dec-94
       WD1254+223        12.95064        22.03194              0.
             0.      500000     1A             ALGOL
```

This listing can be used with **tcreate** to make a new catalog table, if one is needed.

Telemetry tables and the *observation* file

cep does not use all the telemetry tables in processing event data; only a few of the tables need to be transferred from the data tape to disk just to run the pipeline. The subset of tables required to process most observations is only *table0.tab*, the events, and *table1.tab*, which has the aspect quaternion components. GO's whose data were telemetered in WSZ mode (observations made after March 1, 1995) will also need *table7.tab*.

cep gets information about the format of the telemetry tables from the file *observation*, which is in the same directory with the tables. Users should not change this file; it is created by the program that builds the tables and is not meant to be modified.

Example 5.1 : Rerunning the pipeline with new reference data

We will describe a situation in which A new release of the EUVE reference data has been announced by the EGO Center, and this new data contains important changes which the GO can take advantage of by rerunning the pipeline. This is a straightforward repeat of the processing that creates the standard EUVE QPOE files.

The GO first obtains the new version of the reference data. In the target directory with the original QPOE files, she removes the old files from the *reference* directory (she still has a copy on the data tape). She starts *ftp* by typing:

```
% ftp ftp.cea.berkeley.edu
```

After changing to binary mode, she uses the `get` command to copy the new tarfile from */pub/reference/egodata1.#* to her *reference* directory. She checks the *README* file to confirm that she already has the correct version of the **euv** package installed. She then unpacks the tarfile in IRAF with **rtar** and converts the various FITS files to IRAF images or ST tables with the **tables** task **fitsio.strfits**.

The GO did not have her telemetry on disk, so she mounts the data tape and positions the tape drive at the beginning of the sixth tarfile, which contains the engineering telemetry tables. She transfers to the target directory, and reads the telemetry files *table0.FITS* and *table1.FITS*, and the file *observation* from the tape with **rtar**. **rtar** creates the directory *observation* and places the FITS files in it. The GO converts the telemetry tables to ST format with **strfits**.

The GO moves any existing QPOE files to the names *filenamew_old.qp*, to prevent them from being overwritten. The old files are also backed up in ST tables. She moves the file *pipeline* from the *reference* directory to the target directory.

The GO checks the size of the file *table0.tab* in *observation*. **cep** will need roughly twice this much space in the target directory for the temporary files and finished QPOE files. The GO decides to use a system temporary storage space for the temporary files, so she must edit the *reference* file. She opens the file and goes to the section labeled " Primitives for writing photon events," and changes the lines `TempDir ./` to `/tmp1/user/target/` (with the terminating /).

Using **tread**, the GO examines the contents of the file `catalog.tab`, in the target directory, and makes sure that the values in the columns RA and DEC are the same as those recorded in the observation log.

At this point, she can run the default pipeline by simply typing:

```
eu> cep pipeline defpipe
```

cep prints a message to the screen as it finishes reading each 10% of the telemetry events. Since **cep** processes each event as soon as it is read, one can estimate how long it will take to finish the processing by multiplying the time from startup to the first message by 10. It is often several hours for a typical observation. After all the events are read and processed, the message `sorting...` appears on the screen, as the QPOE files are sorted. When **cep** finishes, it writes the sorted QPOE files in the working directory with the usual names, and deletes the temporary files.

5.1.3 Other processing options with cep

There are now several other options for running the event pipeline, which produce different subsets of the QPOE files. These options can be applied by running **cep** with a different value for the parameter `pipeline`. Scripts for the first two options below are included in the *pipeline* file on all GO data tapes. The option for creating QPOE files with event pulse heights is included with version **1.5.1** of the software.

Single Spectrometer QPOE

It is possible to run **cep** with a script which makes a QPOE file for only one Spectrometer channel. This provides economy of both disk space and run time for GO's who must rerun the pipeline, but know that they have interesting data in only one of the detectors. The scripts for creating single Spectrometer QPOE's are called `SW Spectrometer Wavelength Image`, `MW Spectrometer Wavelength Image`, and `LW Spectrometer Wavelength Image`. Each script reads through all of *table0.tab*, but processes events from only one detector. Make a new QPOE file (after all required preparations are done) by typing:

```
eu> cep pipeline 'SW Wavelength Calibration Image'
```

To make QPOE files for two of the three Spectrometer channels, it is necessary to run **cep** twice, so those who want two new QPOE files, and have sufficient disk space, will probably still want to run the standard pipeline, and discard the unwanted QPOE(s) after processing.

5.1.3.1 Remapped QPOE's for imaging instruments

Scripts for making remapped images for all the EUVE imaging instruments, the three all-sky survey Scanners and the Deep Survey instrument, are also supplied in the standard *pipeline* file. At present, the Deep Survey is the only imaging instrument for which data are available in ST table format. Every GO event table (*table0.tab*) produced with version **1.5** or later of the IRAF/EUV software and **1.9** or later of the reference data includes the Deep Survey events. If it is necessary to rerun **cep** to create a new DS QPOE file, GO's can use the script `DS Remapped Image`:

```
eu> cep pipeline 'DS Remapped Image'
```

Pulse height QPOE files for WSZ observations

Starting March 1, 1995, the default mode of operation for EUVE is to telemeter event data in WSZ format, rather than converting the signals to detector X,Y locations on board the satellite. In WSZ mode each event is transmitted as the wedge, strip and zigzag pulses from the detector anode. This means that events are recorded by three pulse heights in *table0.tab*, not the two detector coordinates used for X,Y data. GO's whose data was taken after 3/1/95, and processed with version **1.10** of the reference data should have this new version of *table0.tab*. Refer to the cover letter included with the tape, or use **tlcol** to look for the WSZ columns in *table0*. Telemetry tables for earlier observations do not contain the WSZ information for events. The format of the standard EUVE QPOE files has *not* changed; X,Y coordinates are now calculated in the course of processing by **cep**.

The change to recording events in WSZ mode is advantageous for observers, since pulse height information is preserved in the raw data, while no other information is lost. Energetic particle events that have pulse heights outside the photon distribution can be removed later by filtering on event pulse height. Pulse height filtering is implemented by rerunning **cep** on the WSZ-format telemetry to create a QPOE file with pulse height as one of the event attributes. Once a pulse height QPOE file has been made, the event list can be sorted into an image using a QPOE/IO expression to filter on pulse heights and screen out unwanted events. This is expected to produce a substantial improvement in S/N for low-signal spectra in the short and medium wavelength Spectrometer channels. See the EUVE Electronic News, vol. **5**, no. 2b, for test statistics[3].

All GO's who receive telemetry in the WSZ format can make pulse height QPOE files with the software and reference data supplied on the data tape. The *pipeline* file on the data tape includes new processing scripts for producing pulse height QPOE's with **cep**. The operation requires version **1.5.1** of the IRAF/EUV software, and version **1.10** of the EUVE reference data. The *observation* file is created by the telemetry translation task **euvred.gettm**, and the copy from the data tape with the WSZ-format data should be used to ensure the version is correct.

The *pipeline* file in the version **1.10** reference data includes the script, `Spectrometer Wavelength Pulse Height Image`, which create pulse height QPOE files for all three Spectrometers. The user runs **cep** with this as the value for `pipeline`:

```
eu> cep pipeline 'Spectrometer Wavelength Pulse Height Image'
```

The *pipeline* file also has scripts called `SW/MW/LW/DS Wavelength Pulse Height Image`, which create pulse height QPOE's for each DS/S detector.

5.2 Rerunning CEP to Correct Aspect Problems

As discussed in section 1.3, the aspect record in telemetry may sometimes have errors, because of pointing problems. Also, if the target coordinates used to remap the data in **cep** are in error, or at all different from those used to point the telescope, the results will be the same as if a systematic error had occurred in the aspect reporting. In all these cases, **cep** must be rerun on the data with some kind of "corrected" input to force the correct remapping of events. Aspect errors that change over time are addressed by rerunning **cep** with a "corrected" aspect table. The table can be generated with the task **aspcorr**, as described in section 1.3.2 and example 1.3. We complete the application of aspect-corrected mapping here with the rerunning of **cep** to make the corrected QPOE file.

Related topics

◇ See the material in section 1.3 for more information on diagnosing aspect errors. and the corrections that can be applied.

[3]CEA Web pages, at the URL in section D.2.

Example 5.2 : Reprocessing an observation of the Moon

Recall example 1.3 in Chapter 1, which demonstrated the creation of a revised aspect table for an observation of the moon. The moon was moving through the field of view, and a table of apparent positions in the Spectrometer field was used to add compensating offsets to the aspect components in *table1.tab* with the task **aspcorr**. To produce QPOE files with the corrected event positions, the event pipeline must be rerun with the new table as the aspect reference.

Since **cep** expects the table to be named *table1.tab*, the original table must be moved from *observation/table1.tab* to *observation/table1old.tab* and the new aspect table copied from *newtable1.tab* to *observation/table1.tab*. The pipeline uses only the aspect columns, `Aspect(T,W-Z)`, so this change will be transparent to **cep**.

The GO moves the original QPOE files to different names first, to keep them for comparisons with the new files. IRAF's handy global substitution syntax saves a little typing:

```
eu> rename sw.qp sw_old.qp

eu> ^sw^mw^g

eu> ^mw^lw^g
```

He then makes the other required preparations and reruns **cep**to produce only the LW QPOE file. Since he wishes to do other processing IRAF, he puts the task in the background:

```
eu> cep pipeline 'LW Wavelength Calibration Image' >&cepmess &
```

The messages **cep** usually displays on `STDOUT` will be redirected to the file *cepmess*.

The image created from the new long wavelength QPOE is shown in figure 1-19, and may be compared to figure 1-16. The corrected picture shows each spectral feature centered in Y at the position of a boresight spectrum. The height of each feature is determined by the size of the moon in the field of view, and the width by the collimator response ($\approx 20'$ FWHM).

5.2.1 Correcting constant aspect error

Some cases of error in the aspect report involve constant offsets between the actual pointing direction and the aspect reported. This may cause the spectra to be displaced from boresight in wavelength and/or position, and to be slightly blurred. A constant aspect error can occur for any (or a combination) of several reasons:

- **Telescope flexure** – The structure that supports the DS/S instrument appears to flex slightly in some orientations of the satellite. This changes the orientation of the DS/S instruments to the spacecraft attitude control system (ACS), producing an unrecorded offset between the telescope pointing direction and the ACS coordinate system. As a result, the aspect reports do not have the assumed relationship to the DS/S look direction, whether the telescope is pointed directly at the target, or not, and imaging is degraded.

- **Guide star interference** – A start tracker may lock onto another star near the intended guide star, or become stuck in an intermediate position, while reporting that it has settled on the guide star. Again a constant aspect offset can result.

- **Incorrect coordinates** – If the coordinates used to point the telescope and reduce the observation are in error, or if the target is an extended object with an EUV source of unknown location, this may produce an off-boresight spectrum.

As described in chapter 4 of the *EGO Center Data Products Guide*, **cep** compares the aspect report for each event to the target position in *catalog.tab*, figures the difference, and applies this factor to the wavelength and imaging angle (or the sky coordinates, for DS data) as a correction. This correction scheme uses the aspect information to remap the events as if the telescope were pointed with the target perfectly on-axis. The usual pointing errors reported in the aspect are within about $\pm 20 - 30$ seconds. All events that occur during each aspect reporting period (1 MF) are given the same correction.

Small, unreported errors in aspect occur in many observations because of pointing or coordinate offsets. However, the problems caused by constant aspect offsets are systematic and relatively easy to correct by rerunning the pipeline with an adjusted source position. This is an easier way to add the necessary compensating factor to the event adjustments calculated in **cep**; because the compensation is always the same, it is added only once, to the coordinates in *catalog.tab*, instead of having to be calculated for each aspect report (the job done by **aspcorr**).

cep *always* produces a Deep Survey image with the *catalog.tab* coordinates at the center. If the source is not really on-axis, but the aspect reports reflect only the normal jitter, the image will appear not only off-center, *but at a different location, in the World Coordinate System applied by* **cep**. This situation is depicted in figure 5-3. The telescope is actually pointed in the direction $\vec{\mathfrak{p}}$, but the aspect reports locations near the intended pointing $\vec{\mathfrak{t}}$. After processing with **cep**, the image appears up at the apparent location $\vec{\mathfrak{t}}'$ in the remapped coordinates.

To correct for an unreported offset, the coordinates in *catalog.tab* should cause **cep** to compute coordinate corrections that compensate for both the recorded aspect jitter and the offset. This is equivalent to running **cep** with the *apparent coordinates* of the target from the remapped DS image replacing the coordinates in *catalog.tab*.

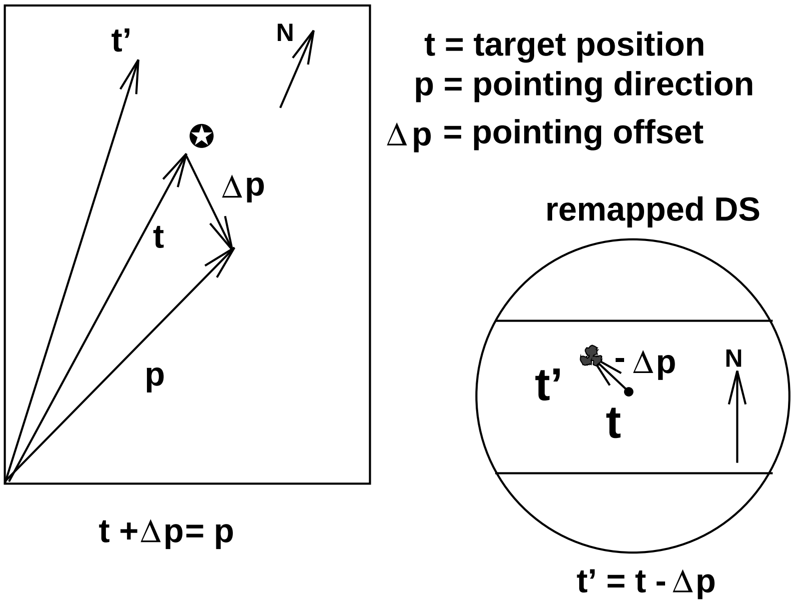

Figure 5-3: Incorrect imaging due to coordinate/pointing offset. The offset $\Delta\vec{p}$ causes the target image to be mapped to the location $\vec{t}'$ in the Deep Survey detector. The remapped DS image always has north along the positive Y axis.

5.2.2 Finding Deep Survey image centroids

Some GO's will find it helpful to produce apparent coordinates by making centroids from the Deep Survey QPOE file. If there is a systematic, constant pointing offset, make a single image for the entire observation. If multiple centroids are needed, use QPOE time filters to create several images for different periods of the observation. For example, with a high Lexan count rate, the pairs of start and end times from the filter macro string in *ds_gt.tab* might be dumped and edited to make several files, each with valid times in a segment of the observation. Then each file, *orbits1_3, orbits4_6*, etc., can be used to make a separate image for those valid times:

```
eu> qpmkim 'ds.qp[time=(@orbits1_3),rect=[1026:1048,1009:1031]]' @img1_3
applycor- detector=ds

eu> ^1_3^4_6^g
```

etc., where each file *img#_#* has an image made from the start : end times of three orbits. Determine values for the `rect` filter, including a margin of background, from displays or plots of the QPOE file. If the observation count rate is limited, it may not be possible to generate frequent DS centroids, so experiment first.

The task **proto.imcntr** can then be used to find the centroid of each image along the rows and columns. A sample parameter file for **imcntr** is shown in figure 5-4.

```
pr> lpar imcntr

      input = "img*.imh"      Image names
     x_init = 1037.           Approx x position of star
     y_init = 1020.           Approx y position of star
  (cboxsize = 23)             Size of extraction box
      (mode = "ql")
```

Figure 5-4: Sample parameter file for using **imcntr** to find the centroid of a DS image.

The value of `cboxsize` should be an odd number. **imcntr** requires an estimate of the centroid as a starting point, such as one might read from the cursor location in a 2-d image display. The following runs the task on each of the images made by **qpmkim** above, directing the output to an ASCII file called *centroids*:

```
pr> imcntr img* 1037 1020 > rawcents
```

Check the output centroids against the displayed images, and if necessary, remove any extra characters before using the *rawcents* file as input for the next step.

cep uses coordinates in decimal degrees, and **aspcorr** needs the apparent coordinates in hours (RA) and decimal degrees (Dec), so the centroids output by **imcntr** need to be converted from logical coordinates (image section pixels) to celestial coordinates. The orientation of the DS image in celestial coordinates is given in the WCS system placed in the image header by **cep**. The Y axis of the remapped image always corresponds to north on the sky. Use the task **xray.xspatial.wcscoords** to convert centroids to RA and Dec using the WCS in the DS image header:

```
xs> wcscoords ds 11 rawcents ofile=dscents
```

wcscoords uses the WCS in the reference image *ds.imh*, and an internal code (11) to perform a logical-to-WCS conversion on the pairs of X,Y coordinates in the input file (*rawcents*) and places the converted coordinates in the file *dscents*. Recall that the coordinates in an image made from a section of another image, in this case the partial DS images input to **imcntr**, are the logical coordinates, referenced to the section boundaries. See the **wcscoords** help page for a complete list of the conversion codes.

To correct for a systematic aspect error, where the pointing offset is constant, the remapped DS image coordinates from a centroid of the entire observation are used in place of the original target coordinates in *catalog.tab*, as in next example. If multiple DS centroids have been created, to map a drift in aspect, the coordinates can be used to create a simple position table for input to **aspcorr**, as in section 1.3.2.

Example 5.3 : Reprocessing an observation to correct for flexure

Suppose that a GO displays the Deep Survey detector remapped image for an observation of FK Aqr, and finds it is fuzzy and not well-centered in the field of view. He locates the approximate center of the image with the cursor, and uses **wcscoords** to find that the image position from the internal image WCS:

```
xs> wcscoords ds 5 STDIN

 :          (Terminate interactive list with <cntl>-D)

 1023 1021


 Convert image pixels to sky coordinates


 1023.  1021.  22:38:41.6 -20:37.19.5
```

These coordinates correspond to an RA and Dec of (339.6732,-20.6221) in degrees: about 35.8 seconds from the requested position of (339.6829,-20.6199). Next the GO checks the target coordinates in the observation log and the copy of *catalog.tab* from the data tape, to make sure that his requested coordinates were indeed used to point the telescope and run the pipeline.

He then inspects the table *mkaspectds.tab*, which contains the satellite aspect reports from *table1.tab*, converted to celestial coordinates. The aspect report suggests the satellite reported it was pointing just over half an arc minute away from the target, and jittering systematically around that spot. The GO spot-checks some rows at the beginning, middle and end of the aspect file, making sure that the aspect reports change by no more than $\approx 30-40$ seconds about this position– the normal aspect jitter. So there is a constant pointing offset, but the pointing and the aspect report are stable. The problem can be fixed fairly easily, by finding the apparent position of the source in the field of view, and inserting this position in *catalog.tab*, causing **cep** to perform the correct remapping, event with to incorrect aspect.

The GO first determines the image centroid more precisely, using **imcntr**, and converts them to degrees with **wcscoords**:

```
xs> imcntr ds 1024 1021 cbox=21

 [ds] x:  1023.117 y:  1021.105
```

```
xs> wcscoords ds 5 STDIN
  ⋮          (Terminate interactive list with <cntrl>-D)

 1023.117 1021.105

 Convert image pixels to sky coordinates

 1023.117 1021.105 22:38:41.6 -20:37.23.6
```

He then converts the coordinates to decimal hours and degrees, and inserts them into *catalog.tab* with **tedit**. Now he is ready to reprocess the data using the default pipeline in **cep** and the new version of *catalog.tab*.

A combination of the methods in this section and section 1.3.2 may be necessary to correctly reduce observations with, for example, telescope flexure *and* roll angle changes, or conditions that produce bad affect reporting over a fraction of the exposure.