Working with Data From the Sun Watcher with Active Pixel System and Image Processing (SWAP)

Laurel Rachmeler & Dan Seaton
PROBA2 Science Center
Royal Observatory of Belgium
Brussels, Belgium

Version 1.0
July 2013

Basic info

Both SWAP (Sun Watcher using Active Pixel System detector and Image Processing) and LYRA are instruments on PROBA2 (PRoject for Onboard Autonomy 2), a satellite that is part of ESA's in-orbit Technology Demonstration program.

SWAP provides images of the solar corona at about 17.4 nm, a bandpass that corresponds to a temperature of roughly 800,000 K, with a cadence of 1 image per 1-2 minutes, and field of view of 54 arcmin × 54 arcmin.

We encourage you to read the two SWAP instrument papers, which go into some detail about SWAP and SWAP data:

D.B. Seaton et al, 2013, *The SWAP EUV Imaging Telescope Part I: Instrument Overview and Pre-Flight Testing* (http://adsabs.harvard.edu/abs/2013SoPh..286...43S)

J.P. Halain et al 2013, *The SWAP EUV Imaging Telescope. Part II: In-flight Performance and Calibration* (http://adsabs.harvard.edu/abs/2013SoPh..286...67H)

Get the routines

There are SWAP and LYRA routines available through SolarSoft that are specifically designed to use our data to its full potential.

First you need to make sure that you have SolarSoft and a recent version of IDL installed and running on your computer. If you are installing SolarSoft for the first time, make sure you select the SWAP and LYRA packages! Detailed instructions for configuring SolarSoft are available at
http://www.lmsal.com/solarsoft/ssw_whatitis.html

If you already have SolarSoft and IDL installed but don't have our packages, within SSWIDL you can run:

```
IDL> ssw_upgrade, /swap, /lyra, /loud, /spawn
```

You will also need to add 'swap' and 'lyra' to SSW_INSTR environment in the file where you configure your SolarSoft.

Where to find the data

For a quick look, you can see images or movies of the SWAP data on the PROBA2 website:

http://proba2.oma.be/data/SWAP

You can also access SWAP images at http://www.helioviewer.org/ (SWAP images will soon available via the Jhelioviewer application).

The PROBA2 website has direct links to the level-0 (raw) and level-1 (calibrated) FITS files that you can download individually.

The ***best*** way to get the data is to use the SWAP object in SSWIDL.

The SWAP Object

In IDL, an object is a data construct, which includes a number of methods (functions and procedures) to operate on its data. Objects are useful because they bring together the data and the tools to use the data. The SWAP Object brings together SWAP images, metadata, maps, and processing routines.

In IDL, objects use the ‘`->`’ operator to call their methods. Among other things, the SWAP Object may be used to search for, download, read, and display SWAP data.

What follows is a non-exhaustive list of some of the things you can do with SWAP Objects followed by notes on prepping and using SWAP data.

Getting started

To use the SWAP Object, you first have to initialize it in IDL. Note that you can call the object whatever you want, and use multiple objects with different names simultaneously. Here we'll just refer to it as '`swap`'.

```
IDL> swap = obj_new('swap')
```

The command above initializes a new object of the type '`swap`'. Once your object is defined, you can use various methods to operate on the object. The basic syntax is usually one of the following:

```
IDL> object -> procedural_method, arg1, arg2, ...

IDL> result = object -> functional_method( arg1, arg2, ... )
```

The ‘set’ method

The ‘set’ method is used to set the basic behavior of the SWAP Object. You should do this before looking for the data so the object knows what kind of data you want, where to look for it, and what to do with it.

```
IDL> swap -> set, filter=filter, local=local, prep=prep
```

The input options for ‘set’ are:

- `filter`: `'lv0'` or `'lv1'`, depending on whether you wish to search for level-0 or level-1 data. The default value is `'lv0'`
- `local`: If `local = 0`, the object searches for data online. If `local = 1`, the object searches for data in your working directory. The default value is `local = 0`.
- `prep`: If `prep = 0`, the object does not attempt to calibrate the downloaded data when it is read. If `prep = 1`, the object will calibrate any level-0 data when it is read using the ‘read’ method.

If `prep = 1`, you can control the calibration steps that are applied using the ‘set_p2sw’ method. This method takes the same keywords as the standalone p2sw_prep procedure, which is discussed later in this tutorial on page 13. For example:

```
IDL> swap -> set_p2sw, /norm, /float, /despike, /outfits
```

The ‘list’ method

The ‘list’ method searches for and returns a list of full-path SWAP filenames based on the given input and the options you used in the ‘set’ method.

Using the ‘list’ method with no arguments returns the file path for the latest image

```
IDL> files = swap -> list( )
```

Specifying a particular date returns all of the filenames for that date

```
IDL> files = swap -> list( timerange = 'YYYY-MM-DD' )
```

Specifying a single date and time returns the file path for the file closest to that time

```
IDL> files = swap -> list( timerange = 'YYYY-MM-DD HH:MM:SS' )
```

Specifying a range of dates and times returns the paths for all the files in that range

```
IDL> files = swap -> list( timerange =
       ['YYYY-MM-DD HH:MM:SS', 'YYYY-MM-DD HH:MM:SS'] )
```

The ‘list_index’ method

The ‘list_index’ method retrieves file metadata without downloading each individual file. This method, which takes no arguments, returns a data structure containing metadata from the FITS header from every file returned by your most recent search with the ‘list’ method.

```
IDL> index = swap -> list_index( )
```

This method is useful when you wish to search for particular files from a long list without first downloading every file in its entirety. (This is particularly useful when you have a slow internet connection and need specific types of SWAP files.)

More information about headers can be found on page 19 of this tutorial.

You can also use the ‘list_index’ method to recover the header metadata from files that have been read into the SWAP Object.

The ‘copy’ and ‘read’ methods

The ‘copy’ and ‘read’ methods work together to download data from the web and load it into the SWAP Object. Once you have obtained a list of files, you can retrieve them from the web using the ‘copy’ method.

```
IDL> swap -> copy, filelist = files
```

The `filelist` keyword takes a string array containing the a list of file paths, such as the one supplied by a ‘list’ method. Once copied, you can read the files from your local disk using the ‘read’ method.

```
IDL> swap -> read, filelist = files
```

Where, again, you must provide a list of files to be read. Once you read files into the object you can work on them, extract SWAP images, make maps and movies, using the methods discussed later in this tutorial.

The ‘getdata’ method

The ‘getdata’ method provides a convenient way to pass the data from files read into the SWAP Object to an external variable or procedure in IDL. Using ‘getdata’ with no arguments returns all the data that was last read into the object.

```
IDL> data = swap -> getdata( )
```

‘getdata’ can also handle the keywords `filelist` and `timerange`, but the results of keywords can be a little confusing, so we only recommend using it without arguments to return data corresponding to the current object state.

Prepping the data

As we mentioned before, the SWAP Object is the best and most versatile tool for obtaining SWAP data, and we recommend you retrieve SWAP data that way if possible.

If you are starting with uncalibrated level-0 files, there are two ways to calibrate the data once you have copied it to your local system.

If you used `prep = 1` in the ‘set’ method, the object will automatically calibrate the data when you use the ‘read’ method, with the settings applied with the ‘set_p2sw’ method (see page 7).

However, you can also prep files directly using the p2sw_prep SSWIDL procedure. When the SWAP Object calibrates the data, it calls p2sw_prep internally using the configuration specified with the ‘set_p2sw’ method. You can also run p2sw_prep on any level-0 SWAP files on your local system without using the SWAP Object. Because this approach my be more transparent and familiar to novice SWAP data users, we use it in the discussion on the following pages.

p2sw_prep.pro

p2sw_prep is the procedure used to calibrate SWAP data. (If you set the `prep=1` keyword in the SWAP Object, the object calls the p2sw_prep procedure.) This procedure is complex, so we recommend referring to the program header, which can be found at http://proba2.oma.be/SSW/proba2/swap/idl/prep/p2sw_prep.pro for a complete description of all keywords and behaviors. To see the header within SSWIDL, you can type:

```
IDL> xdoc, 'p2sw_prep'
```

The simplest possible way to call p2sw_prep on one or several local level-0 files is:

```
IDL> p2sw_prep, filelist, header, data
```

input: string array of level-0 files

output: prepped header structure

output: cube of prepped data

This command performs the following steps:

- Image transformations: solar north is up, sun is in the center, and the sun is round
- dark current subtraction and bad pixel removal
- flat field correction

Prep to level-1

We produce level-1 data at the PROBA2 Science Center, with the following call, which will also write the resulting FITS files to your disk:

```
IDL> p2sw_prep, filelist, header_lv1, data_lv1, /float, $
     /normalize, /despike, /outfits, /verbose
```

- `/float` will change the data from integer to float values, a critical step if you plan to normalize the files by exposure time, as we do here
- `/normalize` will normalize to exposure time so the output intensity units are DN/s
- `/despike` removes cosmic ray hits and other artifacts
- `/outfits` saves a calibrated file in FITS format
- `/verbose` is a useful keyword that prints run-status information

The basic calibration steps—image transformations, dark subtraction, and flat field correction—are also applied.

The `set_p2sw` object method calls the `p2sw_prep` program within the SWAP Object and takes all of the same keywords (see page 7).

A note on calibration updates

The PROBA2 Science Center has periodically updated p2sw_prep and the associated calibration files, such as the dark current model, to improve the quality of our calibrated images. As a result, it is possible your level-1 file does not have the most recent calibration applied to it, especially if you prepped it locally a while ago. If you have any doubt that your calibration is out of date, you may wish to upgrade to the latest version of our SSWIDL routines and re-run p2sw_prep on level-0 data. The level-0 data is never modified.

Generally, the P2SC team can tell if your file is fully up-to-date by inspecting the FITS headers for that file. So you can also contact swap_lyra@oma.be if you have questions about the quality of your calibrated images.

Other common keywords in p2sw_prep

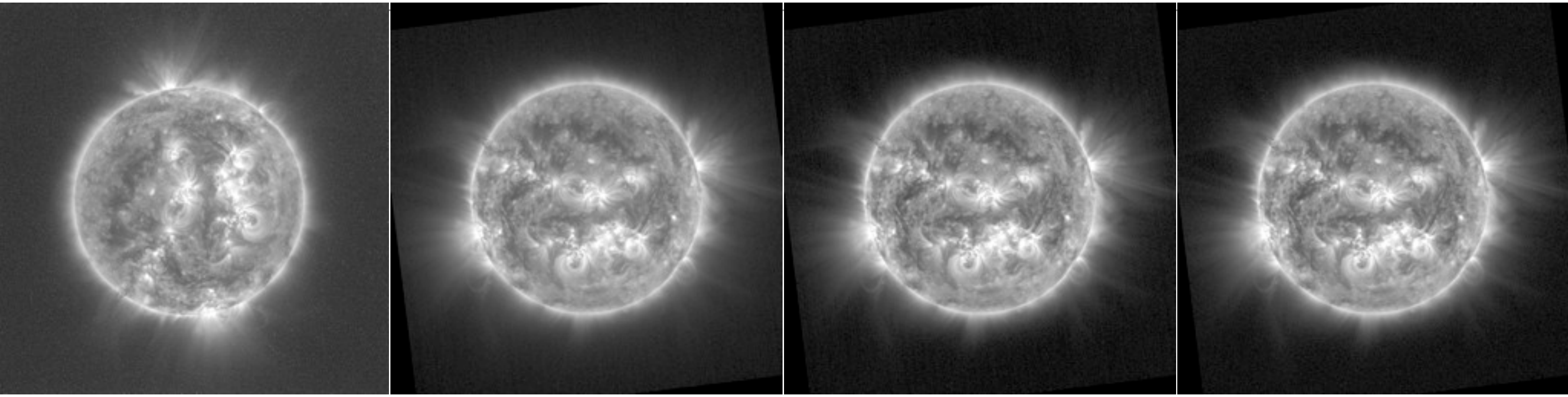
Here we list some of the optional keywords to p2sw_prep, which may be useful to you.

- `/psf_correct` applies a point-spread-function deconvolution to the data to remove stray light.
- `/remove_horizontal` improves slight horizontal banding that is sometimes visible after PSF deconvolution. Note that setting this keyword automatically sets `/psf_correct` as well.
- `/permit_negatives` allows negative values in the resultant data. Negative pixel values are generally undesirable, but this keyword may be useful for those who wish to do careful statistical analyses of SWAP data values.
- `/no_transform` turns off all image transforms: sun centering, rotation to north-up, and pixel aspect ratio anisotropy correction

We do not list all of the keywords here. Complete documentation for all p2sw_prep.pro keywords appears in the file headers for p2sw_prep and its associated routines.

Example: custom calibration options

As an example, here are a few images of the same observation processed with different keywords as input to p2sw_prep.

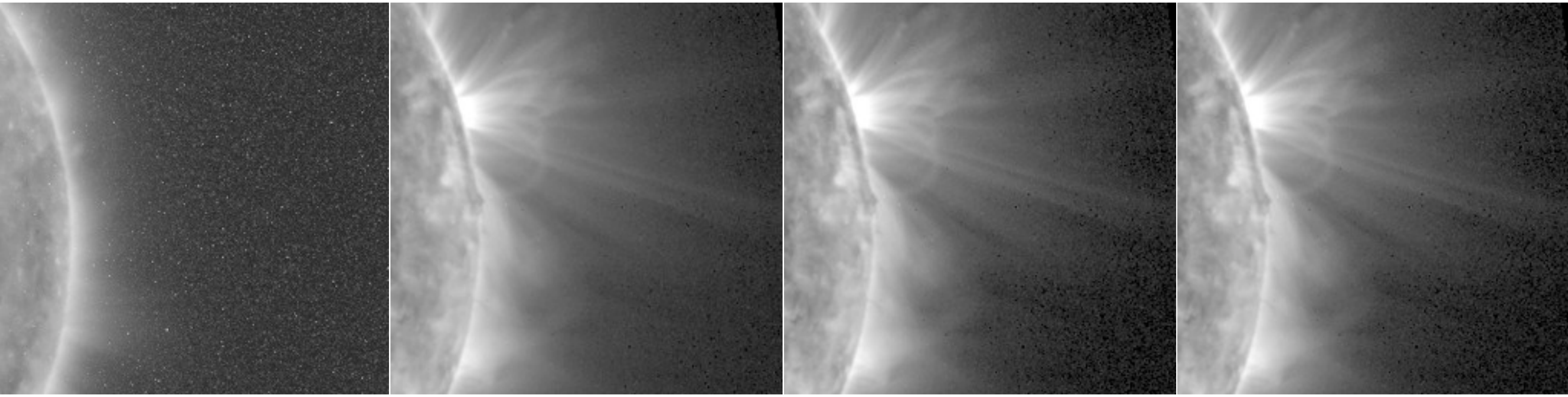


level-0

level-1

level-1 + PSF deconvolution

level-1 + PSF deconvolution + remove horizontal bands



Note on order of prepping

It is possible to partially calibrate SWAP images, save the results, and complete calibration later using a second run of p2sw_prep. However, calibration steps must be applied in the correct order, and many steps lead to irreversible changes in the data that make later calibration impossible. Attempting to re-run p2sw_prep on partially or completely calibrated images may lead to errors or unexpected results. We recommend that a new calibration sequence always begin with uncalibrated, level-0 files.

On the other hand, the internal structure of p2sw_prep guarantees these steps are always applied in the correct order. The order in which optional keywords are set has no effect on the order in which operations are performed, so when you call p2sw_prep, the keywords can be written in any order and the results will always be consistent.

SWAP FITS headers

The FITS header for SWAP files contains a lot of information. A description of most of the header keywords can be found at http://proba2.oma.be/data/SWAP/level0.

Additionally, comments describing the contents of each FITS keyword value appear in the file headers. Read these headers directly, using headfits for instance, to view these comments. Note that these comments are not preserved when the headers are loaded into a data structure using mreadfits, p2sw_prep, or the ‘list_index’ method.

Special Notes:

> In some versions of SSWIDL, mreadfits and thus p2sw_prep, do not handle the standard ‘DATE-OBS’ FITS keyword correctly when converting the header to a data structure. As a result, the date_obs field in the resulting header structure can contain incorrect metadata for some of the files. To avoid difficulties, use the date_d$obs field, which is generated correctly, instead of date_obs
>
> SWAP FITS files use two special keywords, BZERO and BSCALE, that help to minimize file sizes while preserving data integrity. To ensure correct handling of these fields, read the files using an up-to-date version of mreadfits or p2sw_prep. If you wish to suppress the use of these FITS keywords, set the /no_compress_fits keyword when running p2sw_prep.

Working with a header structure

The FITS header information returned with p2sw_prep, the ‘list_index’ method (page 9), and mreadfits is an IDL structure. To see the information contained in this header use

```
IDL> help, header, /str
```

It is easy to return the value associated with a header keyword. Here is an example with the ‘filename’ keyword if you have read in the header information for one file.

```
IDL> FNAME = header.filename
IDL> help, FNAME
FNAME           STRING    = 'swap_lv1_20130621_014014.fits'
```

If your header is an array of structures, which would happen if you read in multiple files instead of a single file, you can select the header information for a single file as you would any array.

```
IDL> help, header
HEADER           STRUCT    = -> <Anonymous> Array[3]
IDL> help, header.filename
<Expression>    STRING    = Array[3]
IDL> help, header[1].filename
<Expression>    STRING    = 'swap_lv1_20130621_020614.fits'
```

Displaying SWAP data

Generally speaking, the dynamic range of SWAP images is too large to be displayed clearly on most standard computer screens, so some processing of the images is necessary to obtain high-quality results when displaying SWAP images.

In the following example we raise the image to the 0.25 power, which reduces the overall dynamic range, allowing the image to be displayed clearly. Other techniques that produce good results include taking the log of the image, taking the square root of the image, and using histogram equalization on the images.

Raising images to the 0.25 power generally produces good looking results. The scaling used in the following example is probably a good starting point for displaying a typical image, but you should experiment to find the technique that works best for your research goals.

Example: display the most recent level-1 image

- Create a SWAP object
```
IDL> swap = obj_new( 'swap' )
```
- Look for level 1
```
IDL> swap -> set, filter = 'lv1'
```
- Identify, copy, and read the latest FITS file
```
IDL> swap -> latest
```
- get the data
```
IDL> data = swap -> getdata( )
```
- open an IDL window
```
IDL> wdef, 0, 1024, 1024
```
- display the image
```
IDL> tv, bytscl( data^0.25, 0.5, 4.2 )
```

You are now displaying the most recent level-1 SWAP image!

Special data

Four times an orbit, the PROBA2 spacecraft rolls 90 degrees (roughly once every 25 minutes). This maneuver takes a few minutes, and frequently results in a few motion-blurred images. The p2sw_prep procedure will attempt to automatically filter these blurred images from your calibrated data. This may result in brief data gaps, or occasionally, an unintentionally blurred image in your final results. Check your data carefully to be sure you have obtained satisfactory results.

PROBA2 occasionally conducts planned off-points in order to study extended off-limb features. These off-pointed images are not automatically re-centered during calibration. If you need sun-centered images from an off-point campaign, setting the `/force_sun_center` keyword in p2sw_prep will override the internal checks for off-points.

PROBA2 also occasionally carries out other calibration operations, resulting in unusual images. If you need to apply calibration steps to these unusual images, you can use the `/force_calibrate` keyword in p2sw_prep.

Creating a SWAP mosaic

By combining multiple off-pointed SWAP images with different pointing, it is possible to produce a mosaic image with a wider field of view than SWAP's standard 54×54 arcmin images. Normally, the easiest way to achieve this is to identify a time range during which there was an off-point and pass this to the mosaic program, which will download, prep, and combine the images obtained in that range. A simple example looks like this:

```
IDL> mosaic = swap_mosaic(['2012-11-14 01:15:00','2012-11-14 04:00:00'], $
                          /verbose, /outfits)
```

- `mosaic` is the final processed mosaic image
- The primary argument is a two-element string array containing the time range of images to be ingested for processing
- `/outfits` causes the program to save the files it preps as it prepares the mosaic.
- `/verbose` causes the program to print details about what it is doing during processing.

If you have already run the program and set the `outfits` keyword, you can quickly generate a new mosaic by omitting the `timerange` argument and setting the `file_list` keyword equal to an array containing the names of the level-1 files to use in generating the mosaic.

```
IDL> mosaic = swap_mosaic(file_list = list_of_files, /verbose, /outfits)
```

Additional special options are described in the header of swap_mosaic.pro, which can be accessed inside SSWIDL using the xdoc procedure.

Useful to know

We hope that the SWAP data has been useful for your research. SWAP has an open data policy and we encourage you to publish your results using this data. However, we ask that you please read through http://proba2.oma.be/data/termsOfUse which has some recommendations for how to acknowledge the data, etc.

If you are interested in working with the PROBA2 team, please consider applying for our Guest Investigator program which provides funding for scientists to come to the Royal Observatory of Belgium to conduct some of their research. Please see the PROBA2 website for the deadline for the latest call for proposals: http://proba2.oma.be/science/guestInvestigatorProgram.