The SolarSoft WCS Routines: A Tutorial

William Thompson
Adnet Systems, Inc.
NASA Goddard Space Flight Center

March 17, 2010

1 Introduction

The FITS World Coordinate System (WCS) (Greisen and Calabretta, 2002) is a formal system for embedded coordinate information within FITS files. At its simplest level, the WCS is an extension of the coordinate system described in the original FITS paper (Wells *et al.*, 1981) with some ambiguities resolved, and with a structure that allows the system to be extensible to handle more complicated kinds of data. The most important of these extensions is the system of projections for spherical coordinates (Calabretta and Greisen, 2002), which itself has been adapted for solar coordinates (Thompson, 2006). Another extension has been published for spectral coordinates (Greisen *et al.*, 2006), and extensions are in the works for distortions (Calabretta *et al.*, 2010) and time axes (Rots *et al.*, 2010).

The details of the WCS are covered by the above references, and won't be repeated here. Instead, this document will outline some of the basics of reading and manipulating the WCS information inside FITS headers using SolarSoft. We'll also give examples of how to use the WCS software to accomplish some common tasks.

2 A basic WCS FITS header

One of the powerful features of the WCS is that it's specifically designed to handle multi-dimensional data, with all the complicated interactions between axes that can occur. The WCS also handles the more complicated forms of FITS data, such as binary tables. However, a large amount of solar data are stored as simple two-dimensional FITS images, and we will restrict most of our discussion to such files. Figure 1 shows a simple FITS header for a hypothetical solar image with the minimal information needed to extract the coordinates of each pixel. We will examine each of the components of this header.

The first five keywords are standard for every FITS file, and simply state that the file contains a 1024×1024 floating-point image, while DATE-OBS gives the observation date. The remainder of the keywords describe the coordinates of the pixels in the image. Many of the keywords in Figure 1—i.e. CTYPEj, CRPIXj, CRVALj, and CDELTj—go back to the original FITS paper (Wells *et al.*, 1981). The WCSNAME, CUNITj, and PCj_i keywords were added as part of the WCS. Each keyword is described below:

WCSNAME: This keyword is not really needed, but is recommended for documentation purposes. It simply gives the name of the coordinate system being used for the data.

```
SIMPLE  =                    T /
BITPIX  =                  -32 /Real*4 (floating point)
NAXIS   =                    2 /
NAXIS1  =                 1024 /
NAXIS2  =                 1024 /
DATE-OBS= '2009-01-01T00:00:00.000' /
WCSNAME = 'Helioprojective-Cartesian' /
CTYPE1  = 'HPLN-TAN'           /Solar-X
CTYPE2  = 'HPLT-TAN'           /Solar-Y
CRPIX1  =              512.50 /Center of detector
CRPIX2  =              512.50 /Center of detector
CRVAL1  =             100.753 /
CRVAL2  =            -23.5210 /
CDELT1  =              3.0000 /
CDELT2  =              3.0000 /
CUNIT1  = 'arcsec  '           /
CUNIT2  = 'arcsec  '           /
PC1_1   =           1.00000000 /
PC1_2   =           0.00000000 /
PC2_1   =           0.00000000 /
PC2_2   =           1.00000000 /
END
```

Figure 1: Example of a basic WCS-compliant FITS header for a two-dimensional solar image.

CTYPEj: The axis labels. The WCS uses a "4-3" format for the axis labels, with the first four characters describing the axis itself, and the last three characters giving the projection used for the axis. Here the labels "HPLN" and "HPLT" stand for helioprojective-cartesian longitude and latitude respectively, also known as the Solar X and Y axes. The "TAN" code stands for the gnomonic projection, which is the projection common to most simple cameras, and has the property that the coordinates scale as the tangent of the angle away from the reference point.

CRPIXj: The reference pixel. For this example, we chose the center of the array. Note that this works out to 512.5 along each axis, because pixels in FITS files are always numbered from 1–N, rather than starting from 0 as is done in IDL.

CRVALj: The coordinate values at the reference pixels. Some data providers, such as SOHO/EIT, choose to set the CRPIXj values to represent Sun center, in which case the CRVALj values are always zero.

CDELTj: The plate scale, here 3 arc seconds per pixel.

CUNITj: The units along each axis. Greisen and Calabretta (2002) give detailed instructions on how to format units strings. Note, in particular, that the units are case sensitive. This allows one, for example, to distinguish between "mm" for millimeters and "Mm" for megameters.

PCj_i: The transformation matrix to be applied to the data. In this example, it is assumed that the data are aligned with solar north, and thus the transformation matrix is simply the unity matrix, with ones along the diagonal, and zeroes everywhere else. For simple 2D images, a non-trivial transformation matrix usually represents a rotation, which in the older FITS system would have been handled with a CROTA2 keyword. The WCS considers the use of CROTAj keywords *deprecated*, and does not allow CROTAj and PCj_i keywords to appear in the same header.[1] However, one is still allowed to use CROTA2 *instead* of the PCj_i matrix for simple FITS files, even though that use is discouraged.

There are, of course, many other keywords which one can add to the header. For example, the units of the data values themselves can be given with the BUNIT keyword. However, the WCS is only concerned with the coordinates along the data axes, and other keywords won't be discussed here unless they affect the coordinates calculation.

2.1 Ephemeris keywords

The WCS paper on solar coordinates (Thompson, 2006) gives a number of keywords which can be used to describe the observer position. These are particularly important for observations made from locations other than the Earth environment, such as from STEREO or Solar Orbiter. The most important ephemeris keywords are listed in Table 1. Additional keywords exist for describing the position of the spacecraft in various heliophysical coordinate systems (e.g. HAEX_OBS, HAEY_OBS, HAEZ_OBS for Heliocentric Aries Ecliptic).

[1] Note that STEREO/SECCHI headers use the PCj_i keywords, but for documentation purposes also include a CROTA keyword without an axis number. This follows the letter of the WCS prescription, if not necessarily the spirit.

Table 1: Principal keywords used for ephemeris information in solar FITS headers.

Keyword	Description
DSUN_OBS	Distance from Sun center in *meters*
HGLN_OBS	Stonyhurst heliographic longitude
HGLT_OBS	Stonyhurst heliographic latitude
CRLN_OBS	Carrington heliographic longitude
CRLT_OBS	Carrington heliographic latitude

3 Extracting coordinate information

The basic routine for interpreting the WCS information in FITS headers is `FITSHEAD2WCS`. This routine parses the information in the header into a structure which is then used by all the other routines. The basic calling sequence is:

```
wcs = fitshead2wcs( header )
```

`FITSHEAD2WCS` should work for most FITS image headers, even those which don't follow the WCS formalism. Common older systems are recognized, and appropriate assumptions are made. For example, headers without embedded ephemeris information are generally assumed to be taken from Earth, except for SOHO images where appropriate adjustments are made.

Some FITS files will contain more than one coordinate system embedded within them. These alternate coordinate systems are differentiated by appending a single letter code from "A" to "Z" after each of the keyword names. For example, the following command is used to extract celestial coordinates in right ascension and declination from STEREO/SECCHI images:

```
wcs = fitshead2wcs( header, system='A' )
```

The routine `WCS_GET_COORD` calculates the coordinates for each pixel. In its simplest form, it will return the coordinates for every pixel in the image. For example, the command

```
coord = wcs_get_coord( wcs )
```

when applied to the FITS header in Figure 1 would return an array with the dimensions (2,1024,1024). The first dimension is always equal to the number of axes, in this case 2. There is obviously considerable redundancy in the coordinates array for this particular example, but there wouldn't be if the roll angle was not zero.

One can also extract the coordinates for specific pixels. For the example in Figure 1, the call

```
coord = wcs_get_coord( wcs, [511.5, 511.5] )
```

would return the reference value (100.753, -23.5210). The SolarSoft WCS routines always use IDL notation for pixel locations, which are one less than the values in the FITS headers.

The inverse process is performed through the routine `WCS_GET_PIXEL`. For example, the command

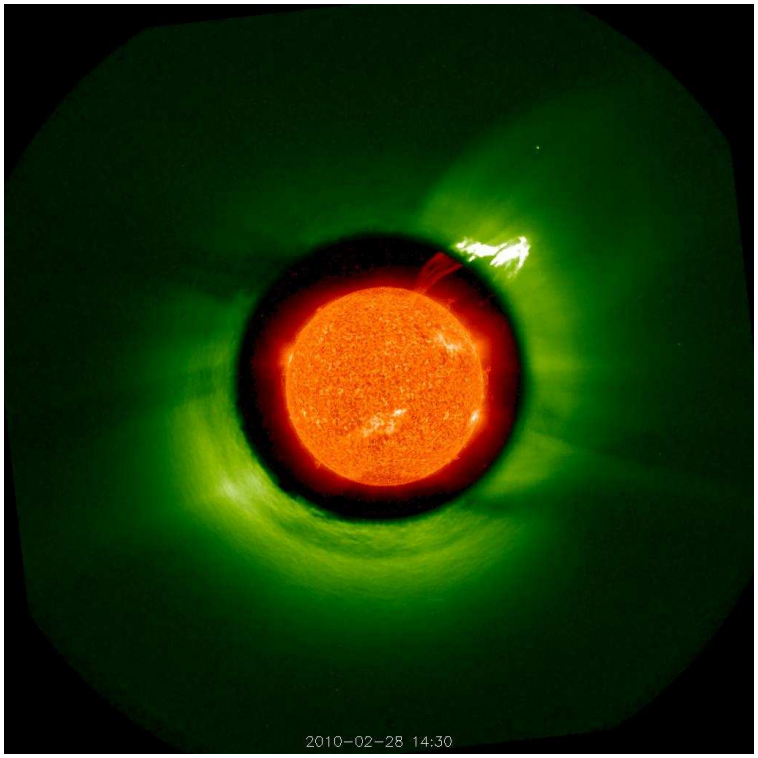

Figure 2: Images from the STEREO-Ahead EUVI and COR1 telescopes combined using WCS software. The combined image shows a prominence eruption that occurred on February 28, 2010.

```
pixel = wcs_get_pixel( wcs, [0, 0] )
```

will return the (IDL) pixel position of Sun center.

3.1 Example: Combining two images

Figure 2 shows an example of using WCS software to combine images from the STEREO EUVI and COR1 telescopes. The basic commands used to register the two images to each other were:

```
wcs_cor1 = fitshead2wcs( header_cor1 )
wcs_euvi = fitshead2wcs( header_euvi )
coord = wcs_get_coord( wcs_cor1 )
pixel = wcs_get_pixel( wcs_euvi, coord)
new_euvi = reform( interpolate( euvi, pixel[0,*,*], pixel[1,*,*] ))
```

These commands rescaled the EUVI image to match the coordinates of the COR1 image. The COR1 pixels inside the occulter could then be replaced with the corresponding pixels from the EUVI image.

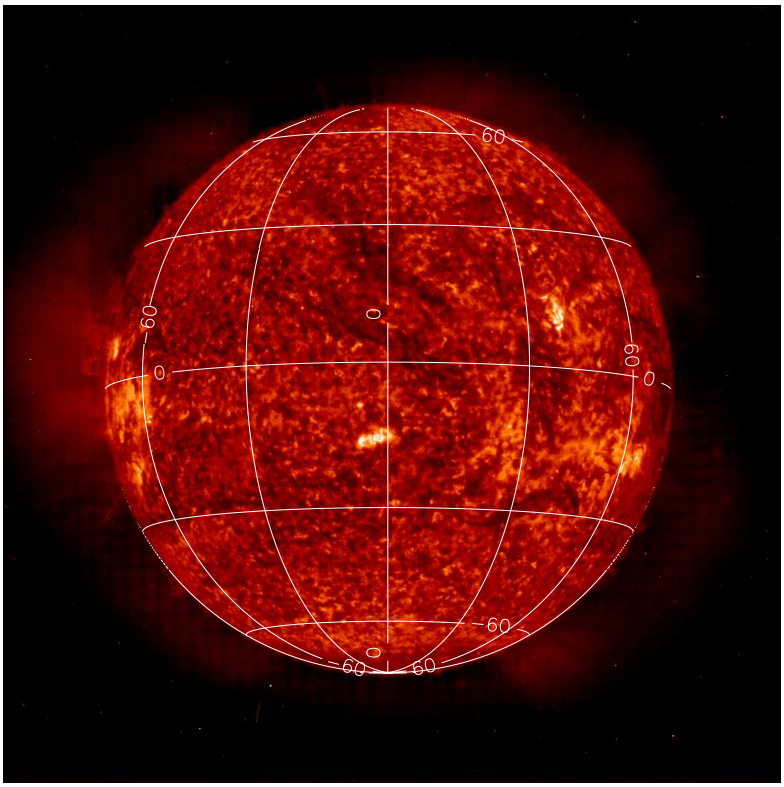

Figure 3: EIT 304 Å image with Stonyhurst heliographic longitude and latitude meridians overplotted.

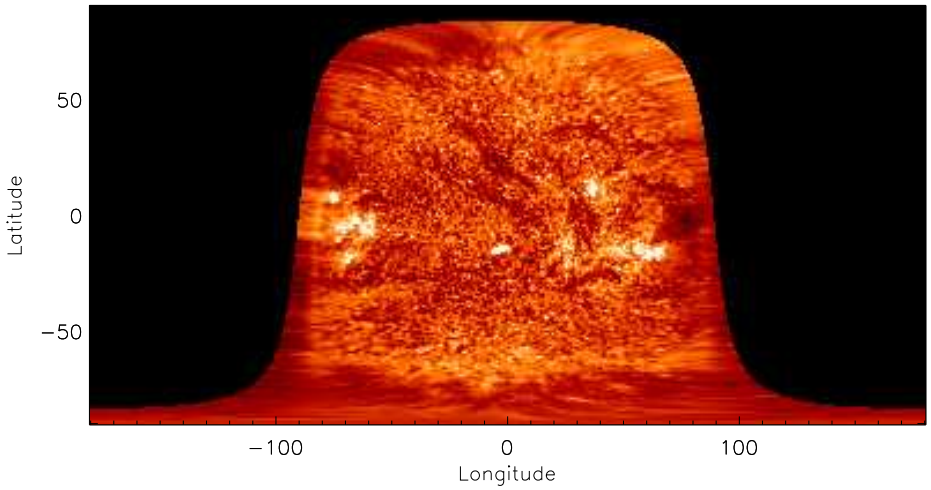

Figure 4: Stonyhurst heliographic map formed from an EIT 304 Å image.

4 Converting coordinates

The WCS library in SolarSoft includes routines to allow one to convert between various solar image coordinate systems. Figure 3 demonstrates the use of the routine `WCS_CONVERT_FROM_COORD` to calculate Stonyhurst heliographic coordinates for an EIT image. The commands used were:

```
wcs = fitshead2wcs( header )
coord = wcs_get_coord( wcs )
wcs_convert_from_coord, wcs, coord, 'hg', lon, lat
```

The inverse operation is performed by the routine `WCS_CONVERT_TO_COORD`. Figure 4 demonstrates the use of this routine to generate a Stonyhurst heliographic map. The commands used were:

```
lon = (findgen(361)-180) # replicate(1,181)
lat = replicate(1,361) # (findgen(181)-90)
wcs = fitshead2wcs( header )
wcs_convert_to_coord, wcs, coord, 'hg', lon, lat
pixel = wcs_get_pixel( wcs, coord )
new_image = reform( interpolate( image, pixel[0,*,*], pixel[1,*,*] ))
```

The same operations can be performed for Carrington heliographic coordinates simply by adding the keyword `/CARRINGTON` to the conversion calls.

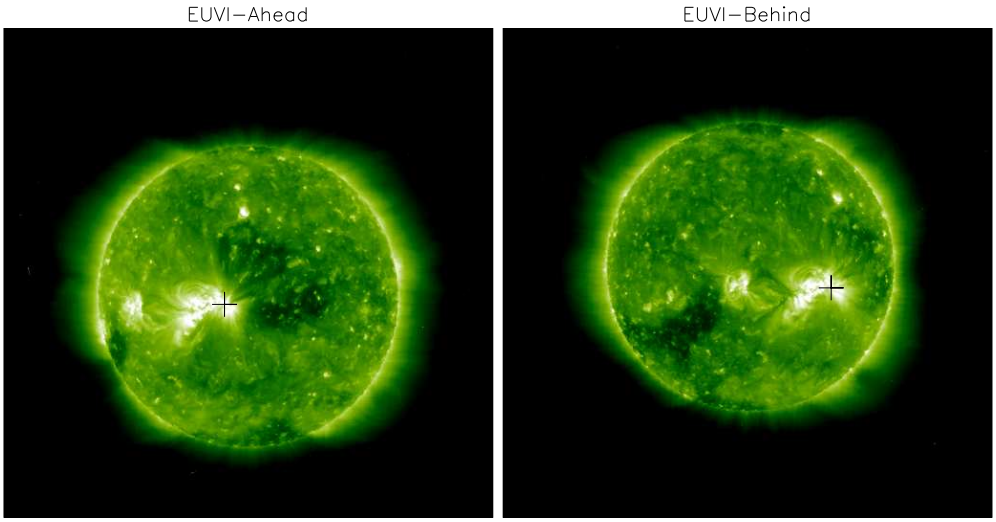

Figure 5: Sample EUVI images from the STEREO *Ahead* and *Behind* spacecraft. The plus symbols mark the same point on the surface as seen in both images, calculated using the WCS information in the FITS headers.

4.1 Example: Comparing observations from different viewpoints

Figure 5 demonstrates the use of the WCS conversion routines to convert pixel positions on a STEREO-A image into the equivalent position on a STEREO-B image. This is done by using heliographic coordinates as an intermediate step. The commands used were:

```
wcs_a = fitshead2wcs(header_a)
wcs_b = fitshead2wcs(header_b)
coord_a = wcs_get_coord(wcs_a, pixel_a)
wcs_convert_from_coord, wcs_a, coord_a, 'HG', hgln, hglt
wcs_convert_diff_rot, wcs_a, wcs_b, hgln, hglt
wcs_convert_to_coord, wcs_b, coord_b, 'HG', hgln, hglt
pixel_b = wcs_get_pixel(wcs_b, coord_b)
```

Note the use of the routine `WCS_CONVERT_DIFF_ROT` to take into account any difference in time between the two observations. (This example was taken from Thompson and Wei (2010).)

4.2 Example: Plotting the limb position

In Figure 6 we show an example of using the routine `WCS_LIMB` to calculate the position of the solar limb as seen from SOHO onto a STEREO-Ahead image. The commands used were:

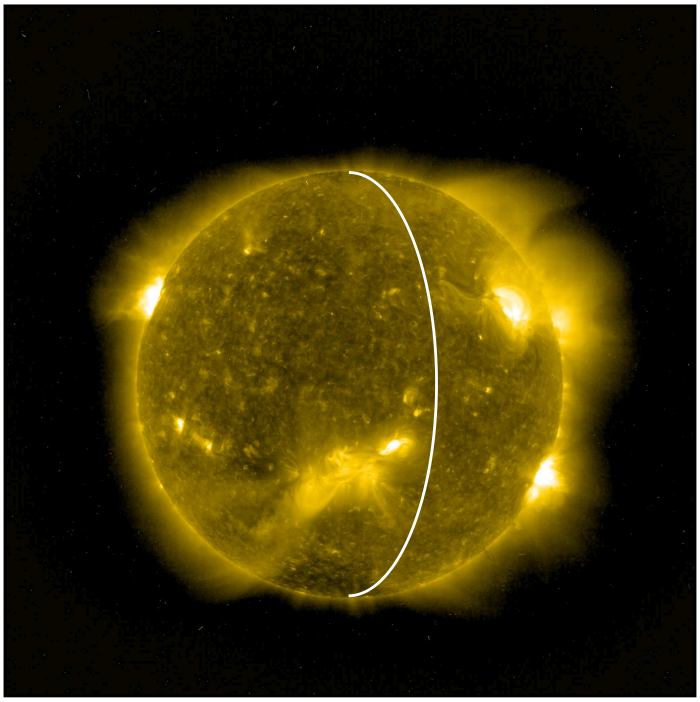

Figure 6: STEREO EUVI-A 284 Å image with the solar limb as seen from SOHO overplotted.

```
wcs_soho = fitshead2wcs( hdr_soho )
wcs_euvi = fitshead2wcs( hdr_euvi )
wcs_limb, wcs_soho, lon, lat
wcs_convert_to_coord, wcs_euvi, coord, 'hg', lon, lat
pixel = wcs_get_pixel( wcs_euvi, coord )
```

5 Interaction with SPICE

The STEREO mission uses the SPICE software from JPL to handle their ephemeris information. The SPICE software can be combined with WCS to accomplish some common tasks. Figure 7 shows how SPICE and WCS can be used to identify planets in the STEREO images. The basic commands used to accomplish this were:

```
wcs = fitshead2wcs( header )
coord = get_stereo_lonlat( header.date_obs, 'Ahead', /degrees, $
                           system='HPC', target='Earth' )
pixel = wcs_get_pixel( wcs, coord[1:2] )
```

and so on for the other planets. (The outer planets include the word "barycenter" as part of their name, e.g. "Jupiter barycenter".) The coordinate systems for the HI2 telescopes are distinctly non-linear, but this is accommodated within the WCS software.

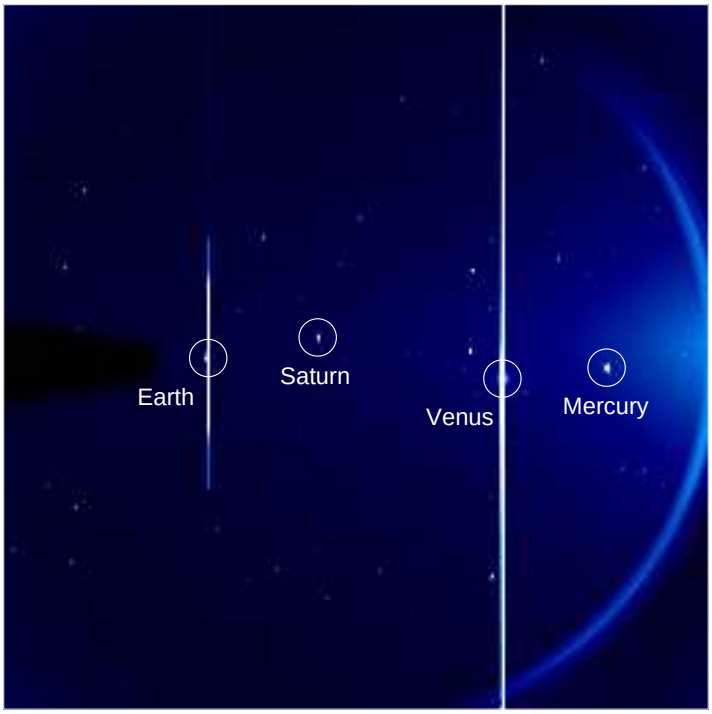

Figure 7: Planets visible in the field of view of the STEREO HI2-A telescope on 20 May 2009, identified using a combination of SPICE and WCS software.

6 Simulating WCS structures

The WCS software is designed to work with FITS headers. There are tasks, however, where one wants to manipulate the data using viewpoints or projections for which one doesn't have a FITS file. The routine `WCS_2D_SIMULATE` was designed to enable such tasks. The "2D" in the procedure name signifies that the procedure is restricted to two-dimensional image or map data, and does not support additional dimensions such as spectral axes or time. Most of the WCS routines do not have this restriction.

Figure 8 demonstrates the use of `WCS_2D_SIMULATE` to simulate what an active region would look like if one were looking directly down upon it. On the left is the original STEREO EUVI-A image, and on the right is the simulated image. The commands used to accomplish this were:

```
wcs0 = fitshead2wcs( header0 )
wcs1 = wcs_2d_simulate(512, cdelt=wcs0.cdelt, dsun_obs=1.5e11, $
                       date_obs=header.date_obs, hgln_obs=88, hglt_obs=-20)
coord1 = wcs_get_coord( wcs1 )
wcs_convert_from_coord, wcs1, coord1, 'hg', lon, lat
wcs_convert_to_coord,   wcs0, coord0, 'hg', lon, lat
pixel = wcs_get_pixel( wcs0, coord0 )
image1 = reform( interpolate( image0, pixel[0,*,*], pixel[1,*,*] ))
```

where the suffix "0" is used for the original image, and "1" is used for the simulated image.

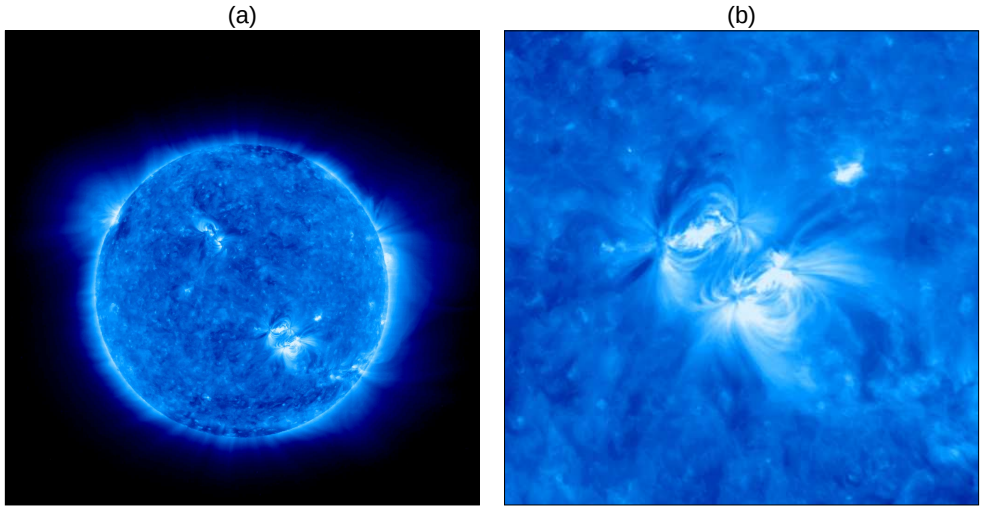

Figure 8: (a) EUVI-A 171 Å image from 6 March 2010. (b) Simulated view from a point directly above the active region.

7 Spectral coordinates

As stated earlier, one of the enhancements applied to the WCS was support for spectral coordinates (Greisen *et al.*, 2006). This system is supported by the WCS software in SolarSoft. At its simplest, the spectral coordinate part of WCS is implemented simply by using the appropriate four-letter type code for the spectral axis, e.g. `CTYPE3='WAVE'` for wavelength, or `CTYPE3='FREQ'` for frequency. There are also some optional projections which can be applied to spectral axes, and which are supported within SolarSoft.

In particular, the "grism" (GRI) projection can be used to model non-linearity in the dispersion relation. The projection is based on a small set of grating and prism parameters to cover a wide variety of spectrometers. In principal, one can characterize the dispersion relation simply by putting in known spectrometer parameters. However, some spectrometer data require that the grism parameters be adjusted to best match the measured dispersion relation. The SolarSoft routine `WCS_FIT_GRISM` is intended to help determine the proper grism parameters to characterize the data.

Figure 9 demonstrates the use of `WCS_FIT_GRISM` for SOHO/CDS data. The plotted symbols represent the residual error for a series of spectral lines when a linear dispersion model is subtracted. It's evident that the CDS dispersion relation is not quite linear. In the CDS software, this is handled by applying a quadratic relationship, but one can also attempt to fit the relationship with the grism function. The resulting fit is demonstrated by the curve in Figure 9. This fit was accomplished with the following commands:

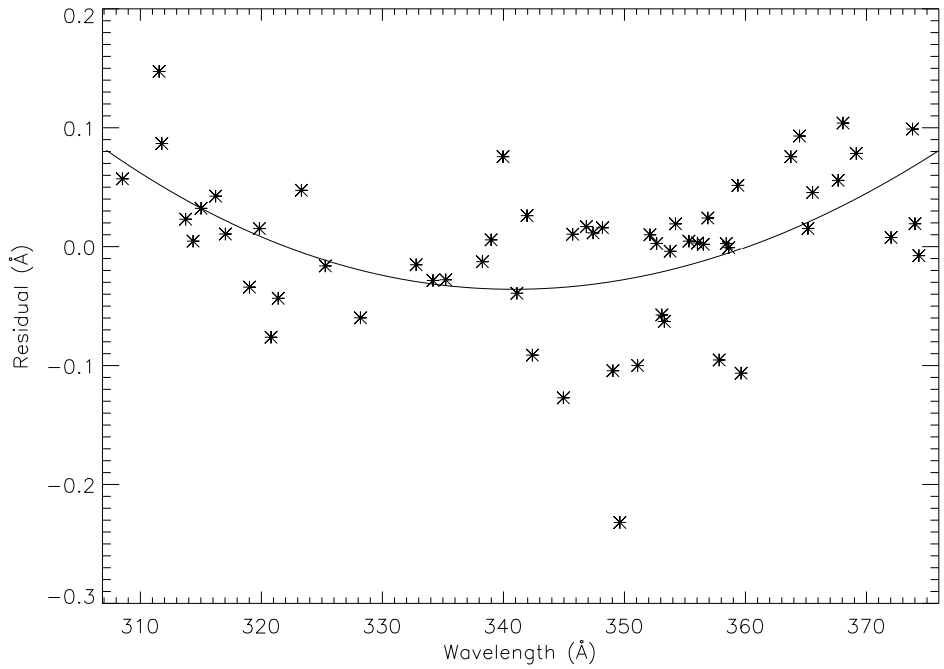

Figure 9: Estimated residuals of the CDS/NIS1 dispersion function for a series of spectral lines, relative to a linear fit. Also shown is a fit to the data using the grism projection.

```
param = [307.2, 0.07, 4.0d6, 0.55, 0, 0]
wcs_fit_grism, pix, wave, param, coord_type='wave', cunit='Angstrom', $
          lambda=[0.1, 0.01, 0, 0.1, 0, 0], max_iter=1000
```

The parameters of the fit have the following nominal meanings, in order:

- The wavelength λ_r at the reference pixel (CRVALj).
- The pixel spacing at the reference pixel (CDELTj).
- The value $Gm/\cos\epsilon$, where G is the grating ruling density in lines per meter, m is the grating order, and ϵ is the out-of-plane angle (held constant).
- The value $n_r \sin\alpha$, where n_r is the zeroth order refractive term in the relationship $n(\lambda) \approx n_r + n'_r(\lambda - \lambda_r)$, and α is the angle of incidence.
- The value $n'_r \sin\alpha$, where n'_r is the first order refractive term (unused).
- The angle θ between the reference ray and the camera axis (unused).

However, the fitted values may not match their physical meanings. What is important is deriving a set of parameters which reproduce the observed dispersion function.

The keyword `LAMBDA` in the call to `WCS_FIT_GRISM` specifies the initial parameter search size for the reiterative fitting technique. Setting the search size to zero for any parameter holds that parameter constant. For CDS, the fit was accomplished by holding the grating parameter constant,

```
CTYPE1  = 'WAVE-GRI'           /
CUNIT1  = 'Angstrom'           /
CRPIX1  =              1.00000 /
CRVAL1  =            307.20592 /
CDELT1  =          0.069664977 /
PV1_0   =            4000000.0 /Grating ruling density
PV1_1   =                    1 /Grating order
PV1_2   =            34.560024 /Fitted angle of incidence
PV1_3   =              1.00000 /Zeroth order refractive index
PV1_4   =              0.00000 /First order refractive index
PV1_5   =              0.00000 /Out of plane angle
PV1_6   =              0.00000 /Angle to camera axis
```

Figure 10: Keywords to implement the fitted CDS grism function. Actual CDS FITS headers may contain slightly different values.

setting the last two parameters to zero, and fitting only the parameters CRVALj, CDELTj, and the expression $n_r \sin\alpha$. Other spectrometers may require a different mix of parameters. Experimentation is recommended to find the best set of parameters which meet the requirements.

Once a set of fitted parameters has been determined, these parameters must then be decomposed into the set of parameters used for the grism projection, which are G, m, α, n_r, n'_r, ϵ, and θ. Figure 10 shows sample header keywords which would implement the grism function shown in Figure 9. Generally speaking, one needs to make some assumptions to accomplish this. For example, it was assumed that $n_r = 1$ in order to derive α.

8 Miscellaneous routines

Although we don't intend to list all the routines within the WCS library, the following are a few selected routines that haven't been touched on yet, and which might be of interest:

WCS_GET_TIME: Returns time values (e.g. `DATE-OBS`) from the WCS structure.

WCS_AU, WCS_RSUN: Returns the values used within the WCS software for an astronomical unit, and the solar radius, in meters. One can also use the `UNITS` keyword to use other units, e.g. `wcs_rsun(units='km')`.

WCS2FITSHEAD: Converts a WCS structure back into a FITS header.

MAP2WCS, WCS2MAP: Converts between WCS and SolarSoft map structures.

References

Calabretta, M., Valdes, F., Greisen, E. W., and Allen, S. (2010). Representations of distortions in FITS world coordinate systems. *Astron. Astrophys.* to be submitted.

Calabretta, M. R. and Greisen, E. W. (2002). Representations of celestial coordinates in FITS. *Astron. Astrophys.*, **395**, 1077–1122.

Greisen, E. W., Calabretta, M., Valdes, F., and Allen, S. (2006). Representations of spectral coordinates in FITS. *Astron. Astrophys.*, **446**, 774–771.

Greisen, E. W. and Calabretta, M. R. (2002). Representations of world coordinates in FITS. *Astron. Astrophys.*, **395**, 1061–1075.

Rots, A. H., Bunclark, P. S., Calabretta, M. R., Allen, S. L., Manchester, R. N., and Thompson, W. T. (2010). Representations of time coordinates in FITS: Time and relative dimension in space. *Astron. Astrophys.* to be submitted.

Thompson, W. T. (2006). Coordinate systems for solar image data. *Astron. Astrophys.*, **449**, 791–803.

Thompson, W. T. and Wei, K. (2010). Use of the FITS World Coordinate System by STEREO/SECCHI. *Solar Phys.*, **261**, 215–222.

Wells, D. C., Greisen, E. W., and Harten, R. H. (1981). FITS - a Flexible Image Transport System. *Astron. Astrophys. Supp.*, **44**, 363.