

SPoCA Module manual.

Written by Benjamin Mampaey (benjamin.mampaey@oma.be) on 5th March 2010.

Module description.

The SPoCA module is composed of 3 programs :

- spoca.pro which is an IDL procedure. Basically it is just an IDL wrapper to call the 2 other programs, transform pixel Cartesian coordinates into HPC coordinates, and do file cleanup.
- bin/SPoCA_HEK.x is a program written in C++. It detects Active Regions (AR) on a pair of images (171 and 195 A) and output the map of AR.
- bin/Tracking_HEK.x is a program written in C++. It uses several maps of AR produced by bin/SPoCA_HEK.x to track an AR through time.

Content of the tarball.

- rob_spoca_0.1.xml and rob_spoca.xml : xml files describing the module for the pipeline.
- hek.xsd : schema for the xml files above
- spoca.pro : The IDL module (see Module description)
- bin, cgt, classes, dsr, objects, programs, results : folders containing the C++ code (see Module description)
- SPoCA_HEK.mk, Tracking_HEK.mk : make files for the module subprograms (see Module compilation)
- VOevents : directory of Voevents xml files generated by our test data set.

Module compilation.

spoca.pro :

Requisites :

- Solar Soft WCS routines

Compilation :

This procedure can be compiled into a Java-IDL bridge object using the instruction stated in SDO Event Detection System (EDS) API.

Like suggested in the aforementioned document, we have not compiled the IDL procedure so it can be done on location.

bin/SPoCA_HEK.x :

Requisites :

- gcc version 4.4.1
- GNU Make 3.81

Compilation :

```
make -f SPoCA_HEK.mk
```

bin/Tracking_HEK.x :

Requisites :

- gcc version 4.4.1
- GNU Make 3.81

Compilation :

```
make -f Tracking_HEK.mk
```

Remark :

1. If it is needed to move the executables of SPoCA_HEK .x and Tracking_HEK .x to another folder, their path must be updated in the procedure spoca.pro.
2. There is a debug flag at the beginning of the module. Changing it from 0 to 1 allow to print a lot of debug information.

Module Arguments.

Sun images

image1, image2: in, required, type string, images filename of wavelength 171 and 195

Arguments general to all modules (as specified in the document SDO EDS API)

- events: out, required, type string array, see document SDO EDS API
- write_file: in, optional, type boolean, see document SDO EDS API
- error: out, required, type string array, see document SDO EDS API
- imageRejected: out, required, type boolean, see document SDO EDS API
- status: in/out, required, type struct, see document SDO EDS API
- runMode: in, required, type string, see document SDO EDS API
- inputStatusFilename: in, optional, type string, see document SDO EDS API
- outputStatusFilename: in, required, type string, see document SDO EDS API
- numActiveEvents: out, required, type integer, see document SDO EDS API

Arguments specific to the SPoCA module

- output_directory: in, required, type string, folder where spoca can store temporary files (The modules manage the cleanup of old files)
- write_events_frequency: in, required, type integer, number of seconds between events written to the HEK
- spoca_args_preprocessing: in, optional, type string, type of image preprocessing for spoca
- spoca_args_numberclasses: in, optional, type string, number of classes for spoca
- spoca_args_precision: in, optional, type string, precision for spoca
- spoca_args_binsize: in, optional, type string, bin size for spoca
- tracking_args_deltat: in, optional, type string, maximal time difference between 2 images for tracking
- tracking_number_images: in, optional, type integer, number of images to track at the same time
- tracking_overlap: in, optional, type integer, proportion of the number of images to overlap between tracking successive run

Remarks :

1. output_directory is a directory in which the module is going to store intermediate files. So it should have write access permission on that directory. No other program or person should be allowed to write in that directory as it could disturb the module. The module takes care of the cleaning of the files in that directory.

2. All sun images files passed to the module must have the same pixel as sun center, and the same radius, otherwise the module will fail. If this is not the case, I must be told, so I can adapt.
3. `write_events_frequency` is the frequency in seconds to which we write events to the HEK.