The Swift Reduction Package - FITS - Users' Manual

by Stefano Covino, 25 Mar 2014, v. ***1.0.0.***

Background
The Swift Reduction Package (hereafter SRP) is a packet of tools supposed to make everyday astronomer's life easier.
For any specific comment the main documentation for **SRP** is the reference source. Here we refer to a sub-package, **SRPAstro.FITS**, devoted to the management of FITS files typically, but not only, obtained by optical/NIR telescopes.

Installation
If you are just updating **SRPAstro.FITS** you just need to download the package from the PyPI archive with:

sudo easy_install -U --script-dir=/usr/local/bin SRPAstro.FITS

provided of course you are connected to the web, and that you want your executable files in "/usr/local/bin".

If you, instead, are installing **SRPAstro.FITS** for the first time or maybe you are upgrading to a new **Python** release, it is likely you need to install many different libraries **SRP** and the related sub-packges relies on.

In principle the command:

sudo easy_install --script-dir=/usr/local/bin SRPAstro.FITS

should again do the job. You might also consider to install the package in a virtual python environment if you do not want to interfere with the system python installation.

However, some of the required libraries can (will) require more concerned actions to allow their installation. In essentially all cases, browsing the web you can quickly find the solution to any problem.

An alternative and strongly advised procedure is to install one of the available open-source self-contained scientific python installations as the Anaconda distribution. Most of the required libraries would then available with no further efforts and SRP is installed smoothly (the Ureka project also deserves consideration).

An alternative and strongly advised procedure is to make a smart use of the various package managers available on many platforms (macports, yum, apt-get, etc.). A possible sequence of operations on Mac OSX is the following:

i) Install SRPAstro (i.e. check the SRP users' manual)
ii) *sudo port install sextractor*
iii) *sudo easy_install --script-dir=/usr/local/bin -U SRPAstro.REM*

while, on other Linux platforms, using yum or apt-get, an analogous sequence should work.
For instance, on a linux-PC running Fedora:

i) Install SRPAstro (i.e. check the SRP users' manual)
ii) sudo yum install sextractor
iii) sudo easy_install --script-dir=/usr/local/bin -U SRPAstro.REM

Do not forget to install this package widely used by several **SRP** tools.

- The ESO-Eclipse package. **Eclipse** is a general purpose reduction system that can very easily be interfaced with regular UNIX commands to run pipelines, etc. The package should run on essentially any UNIX release. The suggested version is the last, 5.0.

Step by step "how to"
These are just examples of what you can do with **SRPAstro.FITS**.

Optical imaging data reduction
Let us assume to have a raw dataset in a directory, now we propose to follow a simple recipe that will allow one to get the final scientific frames with the minimum amount of choices (i.e. the procedure is not completely blind, but almost...).

1. The file list.
   - The first step requires to create a list of the FITS files we want to analyse. This task can be exploited in several different ways. A possible example is: *ls *.fits > filelist.ascii*, you can name the output file as you like. It is also possible to list in the output file, one file per line, FITS files located in different directories.
2. The session name
   - Here you define a common "session name", or a string to prefix most of the files you will create later. This is useful if you have different datasets to reduce and/or analyze in the same directory. The command is **SRPSessionName -n Test**, where of course you can substitute to test any string you like. Please remember that any **SRP** command is executed without parameters will show you a short summary.
3. The FITS keywords
   - This is, probably, the most important task. It is the only one which requires a some level of interaction. Basically, we now choose which are the important FITS keywords to later classify our files (i.e. to separate biases from flat-fields, standards from object frames, etc.). The command is: **SRPKeywords -f myfile.fits**, where "myfile.fits" is a generic FITS file in your dataset supposed to contain all the relevant FITS keywords. Obviously, it

is also assumed that your dataset is homogeneous (as it should be, i.e. no data from different instrument/telescope together, no different observing techniques, etc.). Executing the command now you see on the terminal a list of FITS keywords and you can select or not those you want pressing “y” or any other key but “s”. Pressing “s” you terminate the selection without having to go to the end of the list. Then, in your directory, there is a text file, usually named “TestKeywords.txt” containing the keywords you selected. Nothing prevents you to further modify the file if you need. For very lazy astronomers there is also the possibility to select a pre-selected keyword list available for some combination of telescopes/instruments. An example of this possibility could be **SRPKeywords -p VLTFORSIMA**. In this case you are clearly selecting a set of keywords for VLT FORS1 or FORS2 imaging data.

4. The classification
   - Now we are ready to classify our FITS files. The command is **SRPClassify -i filelist.ascii** and the output is a text file where you can find all files listed in “filelist.ascii” with the content of the selected keywords. Unless you select a different output file name the file with the classification is named: “TestClassify.txt”, where of course “Test” is the selected session name.
5. The selection
   - This is also a very important step. You have to define criteria to classify the input files. There is not a strict sequence to follow. It depends everything on your needs and fantasy. If you have a “classification file” obtained during the previous step and there is a keyword flagging the bias frames you can select only these frames with the command: **SRPSelect -o _bias.txt -k bias** where “bias” is the keyword to be searched in the “classification file”. The output file will be named, in this case, “Test_bias.txt”. The command performs a simple string search, selecting all entries (rows) in the classification files where the keyword can be found. A smart use of this facility can allow you to obtain all file lists you need for any subsequent analysis. Of course this command is provided to help any user, however if you have specific skill with one of the many UNIX character manipulation tools (awk, sed, etc.) you can use them as well. For particularly complex situation you can of course edit manually the classification file and create the output files as required.
6. Frame extraction
   - It frequently happens to need to select an area of a frame to be removed. For instance because the boundary pixels are damaged or for any other reason. This step of course can be required everywhere during a reduction stage. The goal can be exploited with **SRPCut -i filelist.ascii -e 10 10 20 20** where “10 10 20 20” are the distance from the original frame borders in pixels. When available, astrometric information (CRPIX1...) are properly updated.
7. Bias creation

- Assuming you have a list of bias frames you can obtain your final “master bias frame” with **SRPBias -i Test_bias.txt -o _bias.fits** where as usual “Test_bias.txt” is the input file list and the output will be “Test_bias.fits”. The master bias frame is obtained by a 5σ-clipped average of the input frames unless you provide different parameters.

8. Imaging flat-field creation
   - Again, assuming you have a list of flat-field frames you can derive your “master flat-field” frame with **SRPFlatImaging -i Test_flat.txt -b Test_bias.fits -o _flat.fits** where “Test_flat.txt” is the input file list, “Test_bias.fits” is the “master bias frame” and “Test_flat.fits” will be the output frame. The master flat-field is computed by 5σ-clipped average of the input frames unless you provide different parameters.
9. Science frame creation
   - Once you have created your “master bias and flat-field frames” if you have a list of science frames you can apply the correction easily with **SRPScienceFramesImaging -i Test_obj.txt -b Test_bias.fits -f Test_flat.fits** where the meaning of the various parameters should now be trivial. This command also creates an output file list, in this case would be “Test_obj_biasflat.txt” to be used for subsequent analyses. The output files will be named “originalname_biasflat.fits”.
10. Frames alignment
    - If you need you can try to align scientific frames with **SRPAlignImaging -i Test_Di.txt** where again “Test_Di.txt” is the input file list obtained, for instance, by the use of **SRPSelect**. The commands produces output FITS files with the extension “shift” and an output file list named “Test_Dishift.txt”. These files should now be aligned. Of course, being a blind procedure, in case of very noisy frames of with a peculiar lack of bright targets the procedure may easily fail. In addition, it works only with frames all of the same size and with no rotation (i.e. pure translations).
    - For more complicated cases, input frames of different sizes, etc. there is a powerful alternative: **SRPImageMapping -i Test_Di.txt**. This command generates an ouput frame with roto-traslation information with respect a reference frame (the first of the list). This is a relatively long task, but will allow you to then choose a set of objects to be analyzed in all your frames consistently. Output is:
      - filename X_shift Y_shift Rotation_angle X_rotation_centre Y_rotation_centre FWHM Common_area Number_of_matched_stars Comment

      The rototraslation function is:
      - def rotoTrasla ((X,Y), x0=0.0, y0=0.0, alpha=0.0, xrotcent=0.0, yrotcent=0.0):
        - x = X-xrotcent
        - y = Y-yrotcent
        - xn = x0 + x*math.cos(math.radians(alpha))-y*math.sin(math.radians(alpha))

        - yn = y0 + x*math.sin(math.radians(alpha))+y*math.cos(math-.radians(alpha))
        - NX = xn+xrotcent
        - NY = yn+yrotcent
        - return NX,NY
    - The automatic image mapping tries to find a solution applying different approaches. However there will be for sure cases too difficult to manage. You can try to find a solution "by hand" with **SRPRotoTransla -i inputfits.-fits -r reffits.fits**. The solution consists in the roto-traslation parameters to move the object coordinate of the input fits file to the reference frame of the reference fits file. The coordinate center is at the center of the frames.
    - Finally you can create a set of aligned frames, if you applies **SRPImageMapping**, with **SRPRTAlignImaging -i inputfiles.txt**. This command allows also to generate exposure maps to be used later.
11. Frames average
    - If you have more frames now aligned by means of one of the steps previously described, you might want to derive an average of all these frames. You can try **SRPAverage -i inputfiles.txt -o outave.fits**, which implies all frames have the same size.
    - Else you can try **SRPAdvAverage -i inputfiles.txt -o outave.fits**, which processes frames of different sizes and can compute a sigma-clipped average. In addition, in conjunction with **SRPRTAlignImaging,** it can manage exposure maps.
12. Frame astrometry
    - Once you have obtained one or more final scientific frames it is usually very important to compute an astrometric solutions for them. You can do that with **SRPAstrometry -i inputframe.fits -o outframe.fits**. The script tries to get information from the file headers, however in a lot of cases you have to provide more reliable information, i.e. frame orientation, pointing, etc.

Imaging data analysis

1. Frames photometry
    - You can carry out a photometric analysis by means of SExtractor. The first step requires to create the parameter files needed by SExtractor to compute the photometry. The command is **SRPPhotParSet -p parset**, where “parset” can be a set adapted to a specific instrument or a generic set of parameter files.
    - Then, if you want to obtain photometry for most of the sources in your frame the command should be like this: **SRPPhotometry -g 5 -s 30000 -i Test_Dishift.txt**, where “Test_Dishift.txt” is the list of files to be processed, “-g 5” is the gain to be used for error estimate, “-s 30000” is the saturation value. The command creates a text output file for each input frame with the extension “_photom.dat”. The format of the output files are “Id X Y RA DEC magap emagap sky fmax mag emag FWHM” where “Id” is the identi-

fication code for each studied obiect, “X Y” are the position in pixels on the frame, “RA DEC” are the sky coordinate, if available, of the same object, “magap emagap” are the are the aperture magnitudes in the requested aperture for one second exposure and the 1-б error, “mag emag” the integrated magnitude and 1-б error for one second exposure and “FWHM” the full width at half maximum in pixel for the object. This command is just an interface to SExtractor. Objects too close to the frame boundaries are automatically removed. For more detailed off-line analysis, the command creates (or uses if already available) a set of parameter files that can be modified according to your needs. Then, it is enough you run again **SRP-Photometry** and the new input parameter set will be used. One more comment: given the need to provide an automatic tool, in case your frame shows a very variable background the automatic star-finding routine and background estimator may very likely fail. Please be aware: automatic pipelines are not a substitute for a skilled brain.

2. Photometry calibration
    - Unless you have derived your photometry with a pre-computed zero-point, it is possible to compute the zero-point for each frame you are studying provided you know the magnitude for at least an object in the field with **SRPZeroPoint -i instrmag.txt -l 1 4 5 6 7 14 -c calib.txt -C 1 2 3 4 5 -t 1.5**. In case you are analyzing standard star frames you can also get calibrated magnitudes with **SRPQuery**.

Optical spectroscopy data reduction

Again let us assume to have a raw dataset in a directory. Steps from 1 to 7 of the optical data reduction are still perfectly adequate.

1. Spectroscopy flat-field creation
    - Assuming you have a list of raw flat-field frames you can derive your “master flat-field” frame with **SRPFlatSpectroscopy -i Test_flat.txt -b Test_bias.fits -o _flat.fits** where “Test_flat.txt” is the input file list, “Test_bias.fits” is the “master bias frame” and “Test_flat.fits” will be the output frame. The flat-field is obtained exactly as for imaging. However, the spectrum of the flat-field lamp is removed dividing the flat-field frame with a frame created computing an average of the all spectrum raw (dispersion is assumed to be along the horizontal axis). Then an artificial image is created with the same size of the original flat-field and after the division we have the final normalized flat-field suitable for spectroscopy.
- Spectra extraction
    - If you have obtained a 2D FITS frame you can quickly extract your spectra with **SRPSpectralExtraction -i inputframe -s 65 85 -u 100 150 -l 0 50 -m**. You can choose to compute the sky with a median or a sigma-clipping algorithms. The extracted spectra are a sum of the selected pixel window.

File format management.

1. DAOPHOT output file conversion
    - This command allows to convert **DAOPHOT** output photometric files to a more readable format. With, for instance, **SRPDao2Sky -f filename.als -v -S** you convert a PSF photometry **DAOPHOT** file to the **ESO-Skycat** format.
2. GAIA-Photom output file conversion
    - This command allows to convert **GAIA-Photom** output photometric files to a more readable format. With, for instance, **SRPGAIA2Sky -f filename.dat -v -S** you convert your photometry to the **ESO-Skycat** format.
3. FITS spectra to ASCII conversion
    - 1D FITS spectra can be converted to ASCII files with **SRPFitsSpectrum2ASCII -f spec.fits**.

FITS files management

1. FITS header management
    - FITS header reading or even writing can be performed with **SRPFitsHeaders -f filefits.fits**.
2. FITS file statistics
    - Statistics (mean, standard deviation, median and maximum value) about FITS files can be computed with **SRPFitsStats -i fitsfile.fits**.
3. File filtering
    - If required it is possible to apply a median filter to a set of FITS frames with **SRPImageFiltering -i filelist.txt**.
4. Conversion to/from WCS coordinates can be carried out with **SRPWCSPixel**.
    - You can compute positions on a frame with known astrometry or convert pixel positions to astronomic positions with **SRPWCSPixel -c 2 3 -t data.txt -w file.fits -s**.
5. FITS extensions
    - Several FITS files are produces with extensions. You can quickly see and extract them with **SRPFitsExtension -i fitsfile.fits -e**.
6. FITS file composition
    - If you have several FITS images and you need to compose a frame putting individual subframes at given positions in a grid you can try with: **SRPFitsComposer -i filelist -o outfile.fits**. filelist is a text file with a simple syntax: filename xpos ypos, one entry per line. xpos and ypos are pixel shift wrt the reference position.

List of commands

1. **SRPAdvAverage**
    - Its purpose is to obtain an average frame from all the input frames.
    - SRPAdvAverage [-v] [-h] [-e] -i arg1 -o arg2 [-s arg3 arg4] [-x arg5]
        -w Weight for exposure time
        -i Input FITS file list
            -s Sigma-clipping levels (left right)
            -x Input FITS exposure map file list
            -o Output FITS file

The exposure maps, if available, allow to compensate areas less exposed.

2. **SRPAlignImaging**
   - Its purpose is to align different frames on a common reference defined by the first frame processed.
   - SRPAlignImaging [-h] -i arg1 [-v]
     -i is the list of FITS images to align

   This routine is a wrapper to the "xcorr2d" ESO-Eclipse command.
3. **SRPAstrometry**
   - Its purpose is to compute an astrometric solution for any image.
     SRPAstrometry -i arg1 [-c arg2 arg3] [-h] [-m arg4] [-N] -o arg5 [-O]
     [-p arg6 arg7] [-P arg8 arg8] [-r arg10] [-t arg11 arg12] [-v]
     [-x arg13 arg14]
     -c Reference for equatorial coordinates
     -f Frame size (arcmin)
     -i Input FITS file
     -m Max rms (arcsec) for an acceptable solution
     -n Number of objects to analyze (source catalog)
     -N Use 2MASS catalogue
     -o Output FITS file
     -O Use USNO-A2 catalogue
     -p Reference for pixel coordinates
     -P Pointing coordinates
     -r Rotation angle (deg)
     -x Increment per pixel [e.g. -1.0 1.0] (arcsec/pix)
4. **SRPAverage**
   - Its purpose is to obtain an average frame from all the input frames.
   - SRPAverage [-v] [-h] -i arg1 -o arg2
     -i Input FITS file list
     -o Output FITS file
5. **SRPBias**
   - Its purpose is to obtain a bias frame by means of a plain average of the input frames.
   - SRPBias [-h] -i arg1 -o arg2 [-s arg3] [-v]
     -i is the ascii file containing the list of FITS files to be processed.
     -m median rather than sigma-clipped average.
     -o is the name for the output BIAS file.
     -s sigmal level (default 5)
     The output BIAS file is obtained by a 5σ-clipped average of the input files.
6. **SRPClassify**
   - Its purpose is to extract information from the FITS headers to be used for subsequent classification.
   - SRPClassify [-h] [-v] -i arg1 [-k arg2] [-o arg3]
     -i is a file with a list of FITS file to analyse

-k is a file with the keyword to read
-o is the output file.

The script extracts from a set of FITS files information coded in their headers.

7. **SRPCut**
   - Its purpose is to extraxct subimages from a frame.
   - SRPCut -e arg1 arg2 arg3 arg4 [-h] -i arg5 [-o arg6] [-v]
     -e indicates the distances in pixels from frame border (leftx, lowy, rightx, upy)
     -i is the input file list.
     -o is the optional suffix for output files.
     The script trims a frame extracting a subframe with the given limits preserving astrometric information.
8. **SRPDao2Sky**
   - Its purpose is to convert photometric files generated by DAOPHOT to other more friendly formats.
     SRPDao2Sky [-h] [-v] [-a arg1] [-e arg2] -f arg3 [-S] [-z arg4 arg5]
     -a Airmass of the analyzed frame
     -e Exposure time (sec) for frame(s)
     -f Input Daophot (.ap,.als) file
     -S ESO-Skycat output
     -z Zero point and error for photometry
9. **SRPFindingChart**
   - Its purpose is to draw nice (hopefully) finding-charts.
   - SRPFindingChart [-c arg1 arg2] -f arg3 [-h] [-l arg4] [-o arg5 arg6] [-r arg7]
   - [-s arg8] [-t arg9] [-v]
     -c Image cuts (min max)
     -f Fits file name
     -l Object label
     -o Object coordinates (RAH:RAM:RAS DECD:DECM:DECS)
     -r Object circle radius (arcsec)
     -t Finding-chart title
     -s Finding-chart size (arcmin)
10. **SRPFitsComposer**
    - Its purpose is to create composed FITS frames with individual FITS subframes located in a grid.
      SRPFitsComposer [-e arg1] [h] -i arg2 -o arg2 [-v]
      -e FITS file extension
      -i Input FITS file list
      -o Output FITS file
11. **SRPFitsExtension**
    - Its purpose is to create independent FITS files from each extension present in the original one.
      SRPFitsExtension [-h] [-e] -i file [-n ext] [-p] [-v] [--version]
      -e Extract extensions
      -i Input FITS file list or single FITS file

-n Plane or extension number to extract
-p Extract planes

12. **SRPFitsHeaders**
    - Its purpose is to read and/or write FITS headers.

SRPFitsHeaders [-h] [-e ext] -f file [-k key] [-o file]
[-n newkey newkey newkey] [-v] [--version]
-e FITS extension
-f Input FITS file
-k Keyword to be searched for
-o Output FITS file
-n New keyword (key value comment)

13. **SRPFitsSpectrum2ASCII**
    - Its purpose is to convert a Fits 1D spectrum to an ASCII file.
    - SRPFitsSpectrum2ASCII -f arg1 [-h] [-v]
      -f Input FITS file list or single FITS file
      Convert a FITS spectrum to an ASCII file
14. **SRPFitsStats**
    - Its purpose is to compute basic statistics for FITS files.
    - SRPFitsStats -i arg1 [-h] [-r arg1 arg2 arg3 arg4] [-v]
      -i Input FITS file list or single FITS file
      -r Select a subregion in pixel (leftx bottomy rightx uppery)
      Returns mean, standard deviation, median and maximum value
15. **SRPFlatImaging**
    - Its purpose is to produce a flat field frame for imaging.
    - SRPFlatImaging -b arg1 [-h] -i arg2 -o arg3 [-v]
      -b is the BIAS/DARK/SKY file to be subtracted
      -i is the list of files to be processes
      -m median rather than sigma-clipped average.
      -o is the output FITS file name
      -s sigmal level (default 5)
      It is obtained by a 5σ positive clipped average of the input files.
16. **SRPFlatSpectroscopy**
    - Its purpose is to produce a flat field frame for spectroscopy.
    - SRPFlatSpectroscopy -b arg1 [-h] -i arg2 -o arg3 [-v]
      -b is the BIAS/DARK/SKY file to be subtracted
      -i is the list of files to be processes
      -m median rather than sigma-clipped average.
      -o is the output FITS file name
      -s sigmal level (default 5)
      The flat-field is obtained by a 5σ positive clipped average of the input files. Then the flat field lamp spectrum is removed dividing by the average response on the whole frame along the spatial direction.
17. **SRPGAIA2Sky**
    - Its purpose is to convert photometric files generated by the GAIA-Photom package to other more friendly formats.

SRPGAIA2Sky [-h] [-v] [-a arg1] [-e arg2] -f arg3 [-S] [-z arg4 arg5]
-a Airmass of the analyzed frame
-e Exposure time (sec) for frame(s)
-f Input GAIA photom file
-S ESO-Skycat output
-z Zero point and error for photometry

18. **SRPImageFilter**
    - Its purpose is to apply a median filter to a set of images.
    - SRPImageFilter [-h] -i arg1 [-m arg2] [-v]
      -i file of list if files to be processed.
      -m size of median filter.
      The output files are produced applying a median filter of given size.
19. **SRPImageMapping**
    - Its purpose is to derive rototraslation parameters for a set of images.
    - SRPImageMapping [-v] [-h] [-f] -i arg1 [-l arg2] [-m arg3] [-n arg4] [-o] [-p] [-t]
      -f Filter reference object by means of their FWHM
      -i Input FITS file list
      -m Minimum number of stars in common area for matching (default 5)
      -n Number of objects for matching for matching search (default 10)
      -o Save files with object positions
      -t Force pure translation
      -p Integer pixel shift for pure translation
      -r Maximum tolerance (pixel, default 0.0)
      -l Search deepness (default 2, same paramater as for ESO-eclipse peak)
      The script identifies common objects by means of a triangle match.
20. **SRPKeywords**
    - Its purpose is to select which FITS header entries have to be used for classification.
    - SRPKeywords [-h] -f arg / -p arg [-v] arg
      -f is the name of a FITS file to be used as a teplate for FITS keyword selection
      -p is a set of pre-chosen FITS headers for several instrument/telescope combinations.
      If you are not sure, try with any letter and you will be prompted with a list of the available combinations.
21. **SRPPhotometry**
    - Its purpose is to perform aperture photometry for most of the source in the frame.
    - SRPPhotometry [-e arg1] [-g arg2] [-h] [-H arg3 arg4] -i arg5 [-r arg6] [-s arg7] [-S] [-v] [-z arg8 arg9]
      -i Input FITS file list or single FITS file
      -g Gain (e-/ADU) for error estimate in photometry

-s Saturation level (ADU) for frame(s)
-e Exposure time (sec) for frame(s)
-S ESO-Skycat output
-r Radius (pixel) for aperture photometry
-z Zero point and error for photometry
-H FITS file header for exposure time, duration, airmass
and filter [default: MJD-OBS EXPTIME AIRMASS FILTER]

22. **SRPPhotParSet**
    - Its purpose is to create a set of SExtractor parameter files.
    - SRPPhotParSet [-h] [-g / -p arg] [-v]
      -g Generic SExtractor parameter set
      -p Pre-selected SExtractor parameter sets
23. **SRPRotoTransla**
    - Its purpose is to apply a roto-translation with parameters provided by the user.
    - SRPRotoTransla [-v] [-h] [-f] -i arg1 -p arg2 arg3 arg4 -r arg5
      -f Filter for FWHM value
      -i Input FITS file
      -p Rototraslation parameters (x0,y0,ang [deg])
      -r Reference FITS file
24. **SRPRTAlignImaging**
    - Its purpose is to align different frames on a common reference defined by the first frame processed and basing on roto-traslation parameters.
    - SRPRTAlignImaging -i arg1 [-v] [-x]
      -i Input FITS file list
      -x Generate exposure maps
      The exposure maps can then be used to generate average files with compensated exposures.
25. **SRPScienceFramesImaging**
    - Its purpose is to apply bias and flat-field correction to a list of frames.
    - SRPScienceFramesImaging -b arg1 -f arg2 [-h] -i arg3 [-v]
      -b Input BIAS FITS file or value
      -f Input FLAT FITS file or value
      -i Input science FITS file list
26. **SRPSourceFinder**
    - Its purpose is to found sources in a frame.
    - SRPSourceFinder -e/-n -f arg1 [-h] [-m arg2] [-S] [-t arg3] [-v]
      -e Eclipse algorithm (default)
      -f FITS file
      -m Minimum number of connected pixel (for native only)
      -n Native algorithm
      -S Skycat output
      -t Threshold for pixel selection
27. **SRPSpectralExtraction**
    - Its purpose is to extract spectra from 2D frames.

SRPSpectralExtraction [-h] [-c clip value] [-e ext] -i file
[-l pixel pixel] [-m] -s pixel pixel [-v]
[--version] [-u pixel pixel]
-a Automatic spectrum location
-c clip value and sky computed by sigma clipping
-e FITS extension (default 0)
-i Input FITS 2D spectral frame
-l Lower sky window (pixel)
-m Sky computed by median operator
-s Spectrum window (pixel)
-u Upper sky window (pixel)

28. **SRPWCSPixel**
    - Its purpose is to convert coordinates to pixel on a specific frame.
    - SRPWCSPixel -c arg1 arg2 [-d] [-h] [-j arg3] -t arg4 [-s] [-v] -w arg5
        -c Columns for coordinates [i.e. 2 3]
        -d Decimal degree data in input [i.e. 152.54166 -10.16944]
        -j Number of header lines to jump
        -s Sexagesimal data in input [i.e. 10:10:10 -10:10:10]
        -t Table containing data to convert
        -w FITS file with WCS solution
29. **SRPZeroPoint**
    - Its purpose is to compute magnitude zero-point with instrumental and catalogue data.
    - SRPZeroPoint [-a arg1] -c arg2 -C arg3 arg4 arg5 arg6 arg7 [-h] -i arg8 -I arg9 arg10 arg11 arg12 arg13 arg14 [-t arg15] [-v] [-z arg16]
        -a Extinction coefficient (mag/airmass)
        -c File with catalogue magnitudes
        -C Column positions for Id RA DEC Mag eMag
        -i File with instrumental magnitudes (at 1s)
        -I Column positions for Id RA DEC Mag eMag Airmass
        -t Maximum tolerance for object association (arcsec)
        -z Zero-point for instrumental magnitudes

Bugs, comments, etc.

Of course, as already stated, any contribution from anyone is welcome. In case you find bugs, have improvements to suggest, would like to contribute to the code, etc. Please send an e-mail to Stefano Covino, stefano.covino@brera.inaf.it. We can not promise to take into account all your comments, but we will anyway try to improve the package to meet your needs.