Req 5.4.6

Title:

Flat-field - master flat

Objective:

Determine the master flatfield, to be used to correct for the pixel-to-pixel gain variation in the raw image data.

Three different measures of the variation in the gain are available: the dome flat (**req.**542), the twilight flat (**req.**543) and the night-sky flat (**req.**544).

A suitable choice of the final master flatfield, based on a combination of one or more of these flatfields, will be implemented.

A method whereby the master dome flat is used to measure the pixel-to-pixel (small-scale) variation, and the master twilight flat is used to measure the large scale variation, would provide a first-order approximation of the master flatfield. These spatial frequencies are separated using a Fourier technique. Night-sky flats are created from raw science or standard data, flat-fielded with this master flatfield and can be used to improve the quality of the master flat.

This master flatfield could then be used to flatfield the science and standard images in the image pipeline.

Experience at FORS has shown that a suitable combination of twilight and night-sky flats provided the best determination of the gain variation.

The master flatfield is proportional to the inverse variance in the flatfielded data and can therefore be used to build a weight image. Individual weight images (**seq.– 633**) assign a weight of zero to bad pixels (**req.**522, **req.**535), saturated pixels, pixels that have a relative gain outside a user defined range, cosmic-ray events and satellite tracks.

Fulfilling or fulfilled by:

Additional data reduction of flat fields obtained by **req.**542 543 544.

When performed/frequency:

daytime, daily

Inputs:

CalFile– 522 *Hot pixel map*
CalFile– 535 *Cold pixel map*
CalFile– 542 *Master Domeflat frame*
CalFile– 543 *Master Twilightflat frame*
CalFile– 544 *Nightsky flat frame*

Outputs:

CalFile– 546 *Master flatfield*

The master flatfield is used in **seq. 632** *Trim, debias and flatfield* and **seq. 633** *Construct individual weights.*

Estimated time needed:

Observation: None. Reduction: 3 min./CCD.

Priority:

essential

Recipe:

```
Master_Flat -d domeflat -t twilightflat [-n nightskyflat] [-c cold]
[-h hot]
```

```
domeflat     : the master domeflat

twilightflat : the master twilightflat

nightskyflat : the nightskyflat

cold         : the cold pixel map

hot          : the hot pixel map
```

Needed functionality:

image - cleaning (eclipse.image_clean_deadpix_median)
image - edge mirroring (eclipse.image_mirror_edges)
image - generation (eclipse.image_gen_lowpass)
image - arithmetic (eclipse.image_mul, eclipse.image_div)
image - statistics (eclipse.iter_stat)
image - FFT (eclipse.image_fftw)

CA:

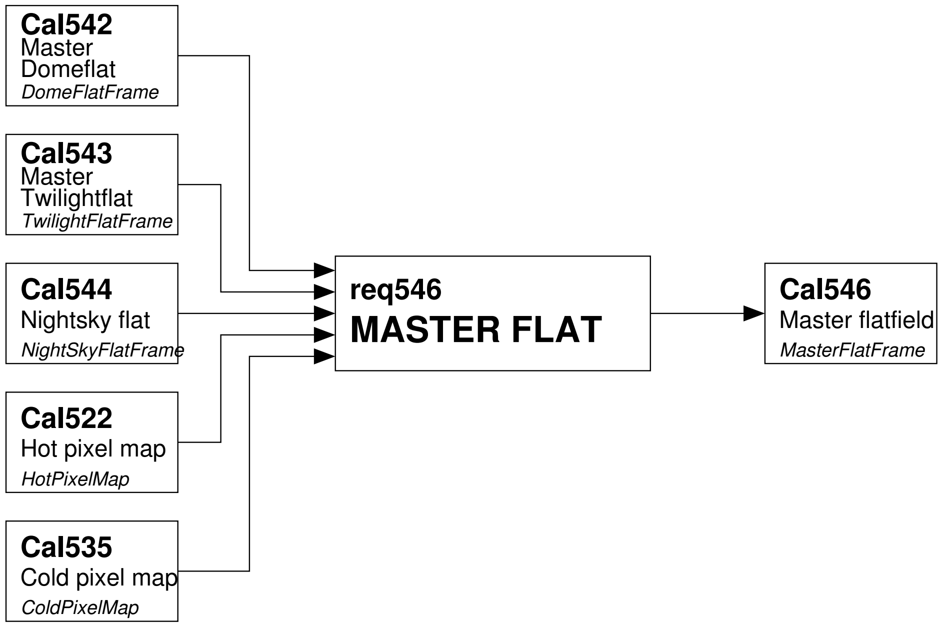

Fig 5.4.6 Dataflow and object class names (in small italic font) for req546

Process (make):

Low spatial frequencies are extracted from the master dome and master twilight flats by the process indicated below. The high spatial frequencies of the dome flat are obtained by dividing the dome flat by its low spatial frequency components. The low spatial frequencies of the twilight flat are then multiplied by the high spatial frequencies of the dome flat.

Low spatial frequencies are extracted as follows:

1. All bad pixels in input images are replaced by the median value of the pixels in a box around the bad pixel.
2. To reduce problems with Fourier filtering near image edges the size of the image is increased by mirroring the edges and corners.
3. A two-dimensional array is created containing the equivalent of a circular Gaussian convolution function in Fourier space (taking into account the quadrant shift introduced by the Fourier transform).
4. The Fourier transform of the image is multiplied by the Gaussian filter.
5. The image is transformed back, and the mirrored regions removed.
6. The resulting image is normalized, excluding bad pixel values.

Verification (verify):
TBD

CAP:

```
Constants:
  DIG_FILTER_SIZE     : standard deviation of Gaussian used to fil-
ter
                        value: 9.0
  MIRROR_XPIX         : width of region in x to mirror
                        value: 75
  MIRROR_YPIX         : width of region in y to mirror
                      : value: 150
  MEDIAN_FILTER_SIZE  : size of box around bad pixels to median
filter
                        value: 36

filter = eclipse.image_gen_lowpass(DIG_FILTER_SIZE)

twilightflat = eclipse.image_clean_deadpix_median(twilightflat,
MEDIAN_FILTER_SIZE)
twilightflat = eclipse.image_mirror_edges(twilightflat, MIRROR_XPIX,
MIRROR_YPIX)
twlt_comp = eclipse.image_fftw(twilightflat, direction=-1)
twlt_comp = eclipse.image_mul(filter, twlt_comp)
twlt_comp = eclipse.image_fftw(twlt_comp, direction=1)
stats = eclipse.stat(twlt_comp, pixmap=mask)
twlt_comp = eclipse.image_div(twlt_comp, stats.avg_pix)

domeflat = eclipse.image_clean_deadpix_median(domeflat, MEDIAN_FILTER_SI
domeflat = eclipse.image_mirror_edges(domeflat, MIRROR_XPIX, MIR-
ROR_YPIX)
dome_comp = eclipse.image_fftw(domeflat, direction=-1)
dome_comp = eclipse.image_mul(filter, dome_comp)
dome_comp = eclipse.image_fftw(dome_comp, direction=1)

dome_comp = eclipse.image_div(domeflat, dome_comp)
masterflat = eclipse.image_mul(twlt_comp, dome_comp)
stats = eclipse.stat(masterflat, pixmap=mask)
masterflat = eclipse.image_div(masterflat, stats.avg_pix)
```

```
if nightskyflat:
    masterflat = eclipse.image_mul(masterflat, nightskyflat)
```