Req 6.3.6

Title:

Dedithering

Objective:

Combine all images of a pointing into a dedithered image and check the quality of the processed result.

Most science applications of OmegaCAM combine data of series of dithered (jittered) observations of a single pointing. The co-addition step maximizes the signal to noise ratio per pixel in the output image, and combines the background subtracted image data, using variance weighting. Since image artifacts (hot, cold, saturated pixels, cosmic-ray events, and satellite tracks) have zero-weight, the co-added image will be much "cleaner" than the input images. A dedithered weight image records the sum of the input weights for each pixel, and is proportional to the reciprocal of the variance in the dedithered image.

By co-adding onto a simple coordinate system—characterized by the projection (Tangential, Conic-Equal-Area), reference coordinates, reference pixel, and pixel scale—the distortions recorded by the astrometric solution are removed from the images.

The consortium proposes to define a standard grid on the Southern sky. Such a standard grid could involve a standardized set of field centers, at for instance 1 degree separation, and a standard set of pixels of for instance 0.2 arcsec size defined with respect to the fieldcenters. See Appendix A6 for a possible definition.

Standardization will result into similar grids of the output images for independent observations (different epochs, different filtering etc) for the same region of sky, without any further administrational burden, facilitating direct image comparison on pixel-to-pixel level without additional interpolations.

Since different input images may have a different zeropoint, the dithering process applies a scaling factor to each input image as well as each weight image.

Note that the current weighting scheme is based on maximizing the signal-to-noise ratio per pixel in the dedithered image. However, when images with different seeing are combined, adding images with bad seeing may lead to a degradation of the signal-to-noise ratio per object, even when the signal-to-noise-ratio per pixel is increased. Therefore, a seeing-based weight-adjustment may be desirable (TBD)

Swarp is a package specifically designed to handle these operations.

The size of the output image is adjusted automatically to completely cover the area covered by the input images. Alternatively, a maximum size of the output image may be defined, which will clip data lying outside this predefined region.

In order to validate the result of the image pipeline, **seq.– 636** should finish with a Quality Check.

A catalog of the dedithered image is obtained in order to measure the PSF variation over the FoV of **OmegaCAM**. This measurement should be compared to the PSFs of the input data (**seq.– 634**) to provide a check on the entire coaddition process. The degradation of the PSF should not exceed the expected degradation due to astrometric uncertainty.

In addition it may be necessary to compare the positions and photometry of the catalogs of the input images to catch possible systematic errors in the relative astrometry and relative photometry.

Finally, the catalog of the dedithered image can be used to check the astrometry w.r.t an external catalog, and to compare the image sensitivity to the one predicted by the ETC. These checks provide end-to-end sanity checks of the entire observation and reduction process.

Inputs:

The input for dedithering is an arbitrary large list of **SeqFile– 634** $Astrometrically\ calibrated\ science\ data$ with photometry and seeing present in the descriptors, and
SeqFile– 633 $Individual\ weight\ images$

output coordinate system (projection, reference coordinates, pixel scale) - (most likely fixed in the code itself - TBC)

Outputs:

SeqFile– 636 $Dedithered\ image$ including statistics and seeing
SeqFile– 636W $Dedithered\ weight\_image$
SeqFile– 636cat $Catalog$
sensitivity description

Estimated time needed:

Regridding: 1 min./CCD Coaddition: Approximately 2 min. for 5 science frames

Recipe:

Note that this recipe only performs the co-addition part of **seq.– 636.**

```
Coadd -i <regridded_frames> [-t target]

regridded_frames : Regridded science frames as produced by the Re-
duce recipe
target           : Filename of the resulting coadded image (string)
```

Needed functionality:

image - coaddition
image - statistics
catalog - source extraction
catalog - seeing
image - background subtraction

CA:

1. Compute FLUXSCALES necessarily to normalize all input data to a single zeropoint.
2. Resample science data to a new grid. A suggested grid is given in Appendix 6.
3. Coadd input data with **Swarp**.
4. Determine statistics of output image.
5. Run **Sextractor** on the output image.
6. Use the median of the half-light radius (FLUX_RADIUS) as a measure of the seeing.
7. Determine the PSF variation over the FOV from the resulting catalog, and compare this to the seeing of the input images (determined in **seq. 634** *)*.

CAP:

```
Constants:
  SWARP_REGRID_CONFIG : Swarp regridding configuration parameters
and their
                        values in case they differ from the Swarp
defaults.
    COMBINE         = 'N'
    CENTER_TYPE     = 'MANUAL'
    MEM_MAX         = 384
    PIXELSCALE_TYPE = 'MANUAL'
    WEIGHT_TYPE     = 'MAP_WEIGHT'
  SWARP_COADD_CONFIG : Swarp coaddition configuration parameters
and their
                       values in case they differ from the Swarp
```

```
defaults.
    COMBINE          = 'Y'
    COMBINE_TYPE     = 'WEIGHTED'
    MEM_MAX          = 384
    RESAMPLE         = 'N'
    WEIGHT_TYPE      = 'MAP_WEIGHT'

# regridding (done in Reduce recipe, on individual science image)
regridded_frame = Swarp(science_frame,
                        SWARP_REGRID_CONFIG,
                        CENTER=RA, DEC,
                        PIXEL_SCALE=PIXELSCALE,
                        FSCALE_DEFAULT=science_frame.get_fluxscale())

# coaddition (done in Coadd recipe, on multiple regridded images)
# regridded_frames could be a list of images from a dither Observ-
ing Block
coadded_image = Swarp(regridded_frames,
                      SWARP_COADD_CONFIG,
                      WEIGHT_SUFFIX='.weight')

coadded_image.statistics()

catalog = coadded_image.sextractor(weight=coadded_image.weight)
coadded_image.seeing = catalog.get_seeing()
output coadded_image, coadded_weight
```