ommodmap

February 1, 2016

Abstract

Creates and applies a 'modulo-8' fixed pattern noise tile.

1 Instruments/Modes

Instrument	Mode
OM	FAST
OM	IMAGING

2 Use

pipeline processing	yes
interactive analysis	yes

3 Description

This task corrects a given OSW image for 'modulo-8' spatial fixed-pattern noise that results from the OM centroiding algorithm. If there are sufficient counts in the OSW image the mod-8 tile can be constructed and applied to the OSW. Both the mod-8 corrected OSW image and the mod-8 map are outputs of the task.

The following functionality is currently not available:

If the s/n of the OSW image is too poor (typically because the OSW image was obtained in FAST mode and is too small to provide good statistics), then an alternative IMAGE mode image that encompasses the FAST mode window is used to construct the mod-8 tile. If no such alternative image is available then the task is aborted.

3.1 The origin of the mod-8 fixed pattern noise

The useable imaging area of the OM CCD detector is 256 by 256 physical (detector) pixels. Individual photon events are centroided to one eighth of a physical CCD detector pixel by the OM electronics.

The resulting OM imaging area is therefore 2048 by 2048 pixels. However, the centroiding algorithm distributes the photons unevenly within each physical CCD pixel, corresponding to 8 by 8 unbinned image pixels or a single mod-8 tile, as though the pixels are smaller near the edge of the tile.

3.2 Creating the mod-8 map

A box of size *nbox* is first stepped across the image and within the box a sigma clipping algorithm is used to mask out sources and any surrounding pixels affected by coincidence loss. Then the average of all remaining mod-8 tiles within a sliding box of size *nbox* is computed and used to produce the mod-8 map.

3.3 Correcting the image

The problem relates to the distribution of events, not to changes in the sensitivity, and so dividing the science image by the mod-8 map, whilst providing a simple cosmetic solution, would compromise the photometry of the image. Instead the algorithm resamples the image to give each image pixel equal area within a mod-8 tile.

For each mod-8 tile in the mod-8 map, the pixel x-boundaries are determined such that within each row of pixels the counts/unit pixel area is the same for each pixel, as illustrated in the left hand panel of Figure 1. This is then taken as the spatial layout of the pixel x-boundaries in the same mod-8 tile in the science image and the science image is resampled to a grid of evenly spaced pixels. Then the y-boundaries of each row are determined such that each row has the same number of counts/unit pixel area, as illustrated in the right hand panel of Figure 1. This is then taken as the spatial layout of pixel y-boundaries in the same mod-8 tile in the science image, which is then resampled to a grid of evenly spaced rows.

The same procedure is then repeated with the order reversed (remapping in y followed by remapping in x) and the final corrected science image is made from an average of the two resamplings. This is to ensure that the redistribution of photons is performed in an identical fashion in x and y.

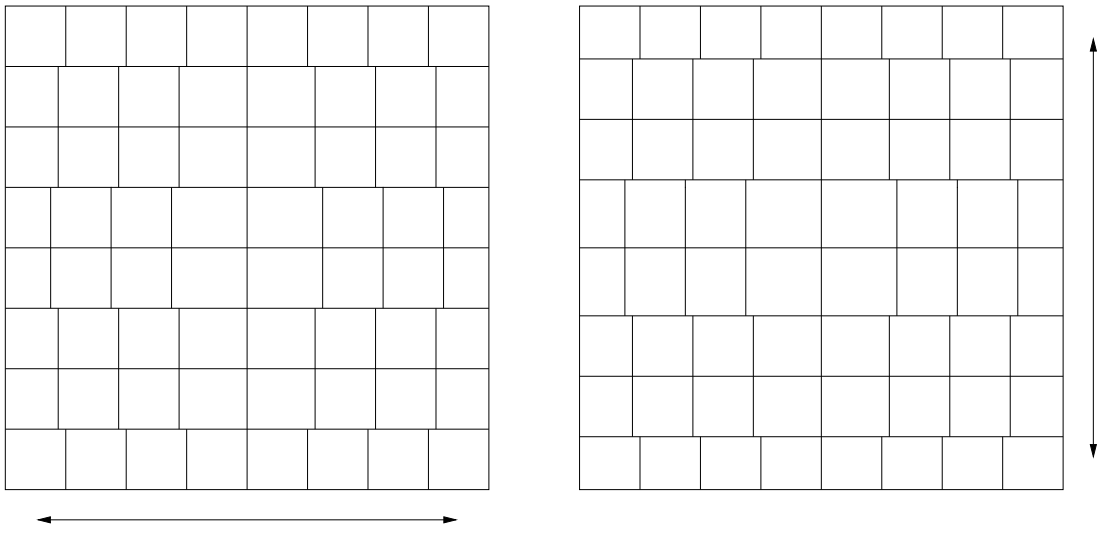

Figure 1: Schematic of the pixel boundary determination and resampling in x and y

The threshold for applying the mod-8 mapping algorithm is currently established to be 1000 mean background counts from the nBox * nBox area, i.e. the mean background value times $nBox^2$ must exceed 1000 counts. The motivation for doing that was due to the fact that the mod-8 noise up to 10 to 20% assumes that in order to improve the image we have to have an accuracy of about 3%, which corresponds to mean*$nBox^2$ ¿ 1000 counts. The algorithm is not applied to images with poor statistics.

4 Parameters

This section documents the parameters recognized by this task (if any).

Parameter	Mand	Type	Default	Constraints
set	yes	string	none	

Name of OM OSW FITS image file

mod8product	yes	logical	false	

Create an output image of the modulo-8 map.

mod8set	yes	string	none	

Name of output product OM OSW modulo-8 tile

outset	yes	string	none	

Name of modulo-8 noise corrected OM OSW FITS image file

nbox	no	integer	16	

Size of sliding box in units of 8 (unbinned) image pixels (corresponding to one CCD pixel)

nsig	no	integer	3	

Significance level for sigma clipping

5 Errors

This section documents warnings and errors generated by this task (if any). Note that warnings and errors can also be generated in the SAS infrastructure libraries, in which case they would not be documented here. Refer to the index of all errors and warnings available in the HTML version of the SAS documentation.

allocError *(fatal)*
Can't allocate memory for internal array

badFileMode *(fatal)*
Invalid datamode 'xxxxxx' in set 'xxxxxx' - the value of the datamode keyword is not as expected

paramError *(fatal)*
Mod 8 product requested but no filename provided for output product

deAllocError *(warning)*
Failed to release memory for *name*
corrective action:

noMod8Noise *(warning)*
The binning factor for the image equals or exceeds the scale of the mod-8 noise pattern and so it is averaged out - the image contains no mod-8 noise pattern.
corrective action: No mod-8 noise correction is necessary or applied

lowMod8Noise *(warning)*
Too few counts in image to determine and thus remove mod8 noise
corrective action: No mod-8 noise correction is applied

6 Input Files

1. Intermediate FITS image file (Output from OMFLATFIELD)

7 Output Files

1. OM OSW modulo-8 map (if requested)
2. Modulo-8 corrected OSW FITS image (goes to OMDETECT)

8 Algorithm

```
subroutine ommodmap

   read in task parameters

   open OM OSW Image file

   determine file type from datamode in primary header

   read in the science window keywords from the primary header

   calculate the image binning

   extract window data from MODES binary table extension

   extract quality array

   extract OM OSW Image
```

```
   if modulo-8 noise is not lost in binning

      calculate good pixels to be used in producing the mod8 Tile Map using a
      sigma clipping method

      calculate size of modulo-8 tile

      calculate size of sliding box

      loop over OM OSW Image in steps of modulo-8 tile

         loop over each pixel in modulo-8 tile

            use sliding box to calculate mean pixel values

         end loop

         use mean pixel values to calculate modulo-8 tile

         insert modulo-8 tile into modulo-8 map

      end loop

   end if

   correct OM OSW Image for modulo-8 noise

   create output OM OSW Image file

end subroutine ommodmap
```

9 Comments

10 Future developments

- Cases involving poor statistics need to be dealt with more rigorously.
- Use of an alternative image for reduction of cases with poor statistics needs to be implemented.

References