mosaic_mask.pl

David Makovoz, 04/30/04

Version 1.2

Table of Contents

1 Overview

The software in this package performs interpolation and co-addition of FITS masks images. Input masks are interpolated onto a common grid and coadded into two products: a data cube mosaic and a single plane mosaic. The procedure is designed to preserve the bit information in the input mask images. Each data plane in the data cube mosaic carries information about a specific bit in the input masks. In the single plane mosaic each bit corresponds to the input mask bit and is set for a particular pixel if enough of the input masks have that bit set for that pixel.

1.1 Input Image Requirements

The input images have to be in the FITS format. The data type of the images are short integers, i.e. `BUNIT` = 16. The following keywords are required in the headers of the input files for the software to work: `BITPIX`, `NAXIS`, `NAXIS1`, `NAXIS2`, `CRVAL1`, `CRVAL2`, `CRPIX1`, `CRPIX2`, `CTYPE1`, `CTYPE2`. `CD`-Matrix elements or/and `CDELT1`, `CDELT2`, `CROTA2` should be also present.

The following projections are implemented so far: "`SIN`", "`TAN`", "`ZEA`"(Lambert's zenithal equal-area), "`ARC`" (zenithal equidistant), and "`STR`" (stereographic).

There are no practical restrictions on the size of sky covered by the set of images in order to be processed by the mosaicker. The only nominal restriction is that the set of images should only cover one hemisphere (a solid angle $< 2\pi$ Sr).

2 Input and Output

2.1 Input Data

The script *mosaic_mask.pl* requires input images and a namelist (configuration) file. Table 1 lists the names of the input files for *mosaic_mask.pl*. These names can be set in the namelist file, or on the command line; some of them have a default value. Except for the namelist, the names of the input files can be specified using a relative or the absolute path. The command line settings override the namelist settings.

Input File	Default Name	Namelist name	Command line option	Required
Namelist *	mosaic_mask.nl	N/A	-n	y
List of input images	mask_stack.txt	`MASK_STACK_FILE_NAME`	-I	y
FIF file name	-	`FIF_FILE_NAME`	-F	n

Table 1 Input data for *mosaic_mask.pl*. *- namelist needs to be in `cdf/` sudirectory.

2.2 Namelist – Configuration file

The namelist contains several blocks of various parameter settings, input image names, and running options. Most of the parameter settings for the modules are in the corresponding blocks delineated by "&" following by the capitalized module name in the beginning and "&END" at the end of the block. Several parameters affecting more than one module are set outside of the individual modules' blocks. Also, the locations of the output final and intermediate products are set in the namelist file.

Table 2 lists the names of the modules, along with their purpose and namelist triggers. To run a module, its trigger should be set to 1.

Module	Namelist trigger	Purpose
fiducial_image_frame	`run_fiducial_image_frame`	Compute Fiducial Image Frame (FIF)
mosaic_int_mask	`run_mosaic_interp_mask`	Interpolate input images to the FIF
mosaic_coadd_mask	`run_mosaic_coadd_mask`	Co-adds interpolated images into tiles
mosaic_combine_mask	`run_mosaic_combine_mask`	Combines co-added tiles into a mosaic image

Table 2 Modules, their namelist triggers, and purpose.

The intermediate and final products are written in several subdirectories. The names of the subdirectories can be configured in the namelist file. Table 3 lists the default names of the output subdirectories, the keywords used in the namelist, and all the products written in the subdirectory. `OUTPUT_DIR` can be specified as a relative or absolute path.

Subdirectory set in Namelist	Default name	Output files
`OUTPUT_DIR`	`./`	FIF.tbl, fif.tbl, copy of the namelist, and all other subdirectories listed below
`INTERP DIR`	`Interp`	Interpolated images
`COADDER DIR`	`Coadd`	Co-added images
`COMBINER DIR`	`Combine`	Mosaic images

Table 3 Output directories and the products written there.

The subdirectories are created by *mosaic_mask.pl*. For the complete list of products, see Table 5.

Table 4 lists all the parameters, along with their default values, if any, a short description, and the name of the module(s) using this parameter

Parameter Name	Description	Default	Module
`USE_REFINED_POINTING`	Switch to use refined pointing keywords (int)	0	*mosaic_int_mask*
`MOSAIC_PIXEL_SIZE_X` `MOSAIC_PIXEL_SIZE_Y`	The size in degrees of mosaic pixel in the *x*- (*y*-) directions. `MOSAIC_PIXEL_SIZE_X` should be negative according to the convention that `CDELT1` < 0. Takes precedence over `MOSAIC_PIXEL_RATIO_X(Y)` (float)	-	*mosaic_int_mask*
`MOSAIC_PIXEL_RATIO_X` `MOSAIC_PIXEL_RATIO_Y`	The ratio of the input pixel *x*- (*y*-) size to the mosaic pixel *x*- (*y*-) size. (float)	1	*mosaic_int_mask*
`Mask_Use_BitPattern`	The bit pattern of the bits to be included in the coadded image(short)	-	*mosaic_coadd_mask*
`Edge_Padding`	The padding in arcsec added to all four sides of the FIF(int)	0	*fiducial_image_frame*
`CROTA2`	If set, this is the orientation of the FIF. If not set the program will compute the optimal orientation (float or char 'A')	-	*fiducial_image_frame*
`TILEMAX_X` `TILEMAX_Y`	The suggested size in pixels of the coadded tiles in the x- or y- directions (int)		*mosaic_coadd_mask*
`MIN_NUMBER`	Minimum number of images in the stack with a particular bit set in order for that bit to be set in the output single plane coadded mask* (int)	1	*mosaic_coadd_mask*
`MIN_FRACTION`	Minimum fraction of images in the stack with a particular bit set in order for that bit to be set in the output single plane coadded mask* (float)	0	*mosaic_coadd_mask*

Table **4** The processing parameters for *mosaic_mask.pl*. The shaded fields are for the parameters set outside of the individual module blocks. *Both conditions for `MIN_NUMBER` and `MIN_FRACTION` have to be met in order to set the bit in the single plane coadded mask.

2.3 Other Options

1. Switch `create_mosaic_cube`. If set the mosaic data cube is produced. The default is 0.
2. Switch `create_singleplane_mosaic`. If set the single plane mosaic is produced. The default is 0.
3. Switch `NICE`. If `NICE` =1 all the modules called by the script with "nice 19." The default is 0.
4. Switch `save_namelist`. The namelist used in the current run is always copied to the output directory. By default the name of the namelist is not changed. By setting `save_namelist` = 1 in the namelist the namelist copied to output directory will be given a unique name, which is created by appending the namelist name to the time of execution. For example, if you ran "`mosaic_mask.pl -n myname.nl`" at 12:32:53, then the namelist will be copied to the output directory as `12h32m53s_myname.nl`. The default is 0, in which case the file is copied as `myname.nl`.
5. Switch `delete_intermediate_files`. If `delete_intermediate_files` = 1 is set in the namelist the products of all the modules run this time will be deleted except for the last module. The default is 0.

3 Mosaicking

3.1 Fiducial Image Frame (FIF) Computation

This step is optional. If a table with the fiducial image frame had been created previously it can be used by supplying its name in the namelist file using `FIF_FILE_NAME` keyword.

This step creates a unified grid coordinate system that is used for creating a mosaic image. Given a list of input images, the perl script *fiducial_image.pl* creates a list of pointing parameters - `CRVAL1, CRVAL2, CRPIX1, CRPIX2, CROTA2, CDELT1, CDELT2, NAXIS1, NAXIS`. Run subsequently, module *fiducial_image_frame* generates the pointing parameters for the bounding region of a minimal size that encloses the input images. Even if the input images use the CD matrix convention, the mosaic image (and by extension the interpolated images) will use the set of keywords `CDELT1`, `CDELT2`, and `CROTA2`, since the mosaic image is undistorted.

Two namelist parameters are used by this module. `Edge_Padding` specifies the size of the margin in arcsec padded around the FIF on all four sides. `CROTA2`, if set, specifies the orientation of the FIF. If `CROTA2` = A, then the orientation of the FIF is found by averaging the twist angles of the input images. If `CROTA2` is not set the program will compute the optimal orientation. The product of this step is the `FIF.tbl` file.

3.2 Image Interpolation

Image interpolation is performed by the module mosaic_int_mask. The processing consists of three steps:
1. Refine the FIF based on the actual mosaic pixel size.

2. Compute the frame of the output interpolated image.
2. Compute the output pixel values.

For the details on Step1 and Step 2 see Spitzer_Mosaic.

Step 3
The value of each output interpolated pixel is set to the result of the logical OR of the values of all the input pixels overlapping the output pixel (see Figure 1).

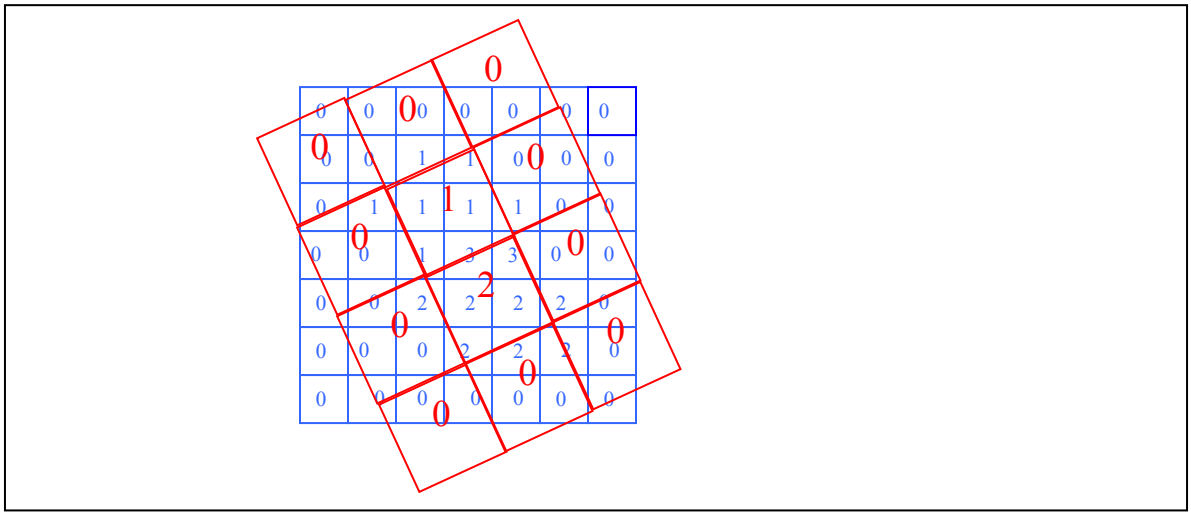

Figure 1 Red input pixels project their values on the blue output pixels. Any output pixels that overlap the input pixel with value = 2 are set to 2, the output pixels overlapping the input pixel with value =1 are set to 1, the output pixels overlapping both input pixels with value 1 and 2 are set to 1 || 2 = 3. Any other output pixels that overlap input pixels with value = 0 only are set to 0. The output pixels that overlap no input pixels are set to 0.

3.3 Co-addition

The module mosaic_coadd_mask performs coadding of interpolated masks. The size, orientation and pixelation of the co-added image are defined by the FIF. The result of co-addition is saved in a single image if its size is not too big. If it is, then the output should be split into tiles. See Spitzer_Mosaic for the details on how tile sizes are computed.

Two mosaic images can be produced. One image is a data cube and is produced if `create_mosaic_cube = 1` is set in the namelist. The second one is a single plane image and is produced if `create_singleplane_mosaic = 1` is set in the namelist. At least one output should be requested.

The namelist parameter `Mask_Use_BitPattern` specifies the bit pattern of the bits to be included in the coadded image. The number of planes in the output data cube is equal to the number of bits set in `Mask_Use_BitPattern`. E.g. if `Mask_Use_BitPattern` = 41 = 1 + 8 + 32, then bits # 0, 3, and 5 are coadded into data planes 1, 2, and 3. Only three planes are written in the data cube in this case.

The pixel value in each plane of the data cube output is equal to the number of pixels in the stack of the interpolated masks with the corresponding bit set.

Here is an example of bit setting for six images: + means the bit is set, - means the bit is not set.

Interpolated Image #	Bit0/plane0	Bit1/plane1	Bit2/plane2	Bit3/plane3
1	+	-	-	-
2	+	+	-	-
3	+	-	-	+
4	+	-	-	-
5	+	-	-	-
6	+	+	-	-
Output	6	2	0	1

The output is saved as a data cube, where each data plane is of data type *byte*, i.e. `BUNIT` = 8. This poses a restriction on the number of images that can be recorded in the coadded image, it can not be greater than 255. However this number is deemed to be sufficiently big for most applications. In the unlikely case that this number exceeds 255 the output will be set to 255.

The single plane image is saved as a short integer (`BUNIT` = 16) image. Only the bits set in `Mask_Use_BitPattern` are set in the this output image. A bit in the single plane mosaic mask is set if two conditions are satisfied:

1. The number of pixels in the stack of the interpolated masks with the corresponding bit set is greater or equal to `MIN_NUMBER`.
2. The fraction of pixels in the stack of the interpolated masks with the corresponding bit set is greater or equal to `MIN_FRACTION`.

The above parameters are set in the namelist. By default `MIN_NUMBER` is set to 1 and `MIN_FRACTION` is set to 0, so that a bit in the coadded pixel is set so long as there is at least one image in the stack of the interpolated images where this bit is set in the corresponding pixel.

This information about the bits used is written in the headers of the output files. The following keywords are written into the header of the data cube for each bit #*I* written in the plane #*J*

`PLNJBIT` = *I* /bit number stored in the *J*'th plane

Using the example above:

`PLN1BIT` = 0 /bit number stored in the first plane

`PLN2BIT` = 3 /bit number stored in the second plane

`PLN3BIT` = 5/ bit number stored in the third plane

The following keyword is written in the header of both data cube and single plane output images (the number 41 is for the example above):

`USEBIT` = 41/Use_Bit_Pattern – bits used in coaddition

4 References

All of the documents referenced in this document can be obtained from the Spitzer Science Center website, http://spitzer.caltech.edu/SSC/. They include:

1. Spitzer_Mosaic

5 Appendices

5.1 Short description of the intermediate and final products

The following table lists the names of the intermediate products of *mosaic_mask.pl* based on the assumption that the input images have names `<input_image_name>.fits`.

Description	Naming convention	Default Location
Fiducial Image Frame	`FIF.tbl, fif.tbl`	`./`
Geometry table, listing the size of the interpolated images and their offsets with respect to the FIF	`interpolated_images.tbl`	`Interp/`
Interpolated images	`interp_<input_image_name>.fits`	`Interp/`
Co-added Image(s) - Tiles	`coadd_cube_Tile_#_Image.fits` `coadd_single_Tile_#_Image.fits`	`Coadd/`
Tile geometry file, listing the size of the tile(s) and their offsets with respect to the FIF	`coadd_cube_tiles.tbl`	`Coadd/`
Tile-input_image association file; lists the input images used in co-adding each tile	`coadd_cube_tile_bcd.txt`	`Coadd/`
Combined co-added data cube image	`mosaic_cube.fits`	`Combine`
Combined co-added single-plane image	`mosaic.fits`	`Combine`

Table 5. A list of the intermediate and final products of *mosaic_mask.pl*.