MultiDrizzle: An Integrated Pyraf Script for Registering, Cleaning and Combining Images

Anton M. Koekemoer, Andrew S. Fruchter, Richard Hook,[1] Warren Hack

Space Telescope Science Institute, 3700 San Martin Drive, Baltimore, MD 21218

Abstract. We present the new PyRAF-based *'MultiDrizzle'* script, which is aimed at providing a one-step approach to combining dithered *HST* images. The purpose of this script is to allow easy interaction with the complex suite of tasks in the IRAF/STSDAS *'dither'* package, as well as the new *'PyDrizzle'* task, while at the same time retaining the flexibility of these tasks through a number of parameters. These parameters control the various individual steps, such as sky subtraction, image registration, 'drizzling' onto separate output images, creation of a clean median image, transformation of the median with *'blot'* and creation of cosmic ray masks, as well as the final image combination step using *'drizzle'*. The default parameters of all the steps are set so that the task will work automatically for a wide variety of different types of images, while at the same time allowing adjustment of individual parameters for special cases. The script currently works for both ACS and WFPC2 data, and is now being tested on STIS and NICMOS images. We describe the operation of the script and the effect of various parameters, particularly in the context of combining images from dithered observations using ACS and WFPC2. Additional information is also available at the *'MultiDrizzle'* home page: http://www.stsci.edu/~koekemoe/multidrizzle/

Introduction

The *MultiDrizzle* task is designed to provide a seamless, integrated approach to using all the various tasks in the IRAF/STSDAS *dither* package to register, clean, and combine dithered images. It has quite a few parameters but in principle can be run very simply from the PyRAF command line, specifying only the output filename and an input file list, e.g.:

```
multidrizzle output='outputfilename' filelist='files.lis'
```

The other parameters can be specified on the PyRAF command line or alternatively can be edited using the standard IRAF 'epar' mechanism before running the task. It is designed to carry out the following steps, either in a single pass or alternatively by selecting various steps individually:

1. StaticMask   - Identify negative bad pixels, based on examining all the images, and include them in the dq file
2. SkySub   - Sky-subtract each frame
3. Driz_Separate   - Drizzle the input images onto separate, registered outputs (using shifts computed from the headers)
4. Median   - Create a median image from the separate drizzled images
5. Blot   - Blot the median image back to the original input frames
6. Driz_cr   - Use each blotted image to create a derivative image, and compute CR masks
7. Driz_Combine   - Do the final drizzle combination

Here we describe the details of the parameters involved in running each of these steps.

---

[1]Space Telescope European Coordinating Facility, Karl-Schwarzschild-Str. 2, Garching, D-85748, Germany

1. Create the Static Mask

Parameters:

staticfile	=		Name of (optional) input static bad-pixel mask
static_goodval	=	1.0	Value of good pixels in the input static mask

Output files:
   modified dq array in the original input files

This step goes through each of the input images, calculates the r.m.s value for each chip, and identifies pixels that are below the median value by more than 5 times the r.m.s. This is aimed at identifying pixels that may have high values in the dark frame that is subtracted during calibration, but may not necessarily have high values in the images, thus the subtraction gives them strongly negative values. Such pixels are not always flagged in the dq file, hence this step allows them to be identified. Sometimes such pixels fall on bright objects so they would not be negative, but instead would be positive although lower than surrounding pixels. However, if the images are dithered then they should land on blank sky at least some of the time, in which case they will appear negative and will be flagged.

2. Perform Sky Subtraction

Parameters:

skytype	=	'single|quadrants'	Type of subtraction (e.g., amplifier quadrants)
skyname	=	'SKYSUM'	Header keyword containing sky value
skywidth	=	50.0	Interval width for sky statistics
skystat	=	'mode|mean|median'	Sky correction statistics parameter
skylower	=	−50.0	Lower limit of usable data for sky (in DN)
skyupper	=	200.0	Upper limit of usable data for sky (in DN)

Output files:
   modified science array in the original input files

This can subtract either the entire chip (skytype = 'single') or specific regions corresponding to each of the four individual amplifiers on the ACS/WFC chips (skytype = 'quadrants'). The other parameters correspond directly to those in the *sky* task in the *dither* package, and are passed to it exactly as they are specified here.

3. Create Separate Drizzled Images

Parameters:

driz_sep_outnx	=		Output image x-size
driz_sep_outny	=		Output image y-size
driz_sep_kernel	=	'square|point|gaussian|turbo|tophat'	Drizzle kernel
driz_sep_scale	=	1.0	Size of output pixels
driz_sep_pixfrac	=	1.0	Size of 'drop'
driz_sep_rot	=	0.	Rotation (anticlockwise)
driz_sep_fillval	=	INDEF	Value for undefined pixels

Output files:
   *_single_sci.fits (drizzled output image for each input image)
   *_single_wht.fits (weight image corresponding to each drizzled image)

This task drizzles the input images onto separate output images. By default it uses the *drizzle* 'turbo' kernel, and *drizzle* parameters of pixfrac = 1 and scale = 1. These can be changed; for example masks can be substantially improved by specifying a smaller value of scale (e.g., 0.5 or 0.66), with the trade-off being larger images (their size increases as the inverse square of the value of scale), and increased computation time. The shifts used here are calculated from the image headers by *PyDrizzle*.

4. Create the Median Image

Parameters:

median_newmasks	=	yes	Create new masks?
combine_type	=	'minmed|average|median'	Type of combine operation
combine_reject	=	'minmax|ccdclip|crreject|avsigclip'	Type of rejection
combine_nsigma	=	6 3	Significance for min. vs median
combine_nlow	=	0	Number of low pixels to reject
combine_nhigh	=	1	Number of high pixels to reject
combine_grow	=	1.0	Radius for neighbor rejection

Output files:
  output_med_*.fits
  *_single_wht_maskhead.pl

This creates a median image from the separate drizzled input images, allowing a variety of combination and rejection schemes. If combine_type is set to 'median' or 'average', then the routine calls the IRAF task *imcombine*, passing to it the values of combine_reject (usually expected to be 'minmax'), along with combine_nlow and combine_nhigh (the number of low and high pixels to reject) and combine_grow, the amount by which flagged pixels can grow. If median_newmasks = 'yes', then pixels are flagged using the static bad pixel masks. If this parameter is 'no' then the task will simply use whatever masks are specified in the 'BPM' header keyword of each image (which you could create yourself). In general, however, it is recommended to use the static bad pixel masks that are generated by default.

   If combine_type is set to 'minmed', then this task will use a slightly more sophisticated algorithm than the ones in *imcombine*, to create a cleaner combined image. The basic concept in this case is that each pixel in the output combined image will be either the median or the minimum of the input pixel values, depending on whether the median is above the minimum by more than a certain number of sigma. An estimate of the 'true' counts is obtained from the median image (after rejecting the highest-valued pixel), while the minimum is actually the minimum unmasked ('good') pixel. This algorithm is designed to perform optimally in the case of combining only a few images (3 or 4), where triple-incidence cosmic rays often pose a serious problem for more simplified median combination strategies. It performs the following steps:

1. Create median image, rejecting the highest pixel and applying masks

2. Use this median to estimate the true counts, and thus derive an r.m.s.

3. If the median is above the lowest pixel value by less than the first value in combine_nsigma, then use the median value, otherwise use the lowest value.

If combine_grow > 0, repeat the above 3 steps for all pixels around those that have already been chosen as the minimum, this time using a lower significance threshold specified as the second value in combine_nsigma. This is very successful at flagging the lower-S/N 'halos' around bright cosmic rays that were flagged in the first pass.

5. Blot Back the Median to the Frame of the Original Images

Parameters:
  [none]

Output files:
  *_blt.fits

This takes the median image and uses the *dither* package task *blot* to apply the geometric distortion and transform it back to the reference frame of each of the original individual input images, in preparation for the subsequent step of cosmic-ray rejection.

6. Create Cosmic Ray Masks

Parameters:

`driz_cr_snr`	=	'3.0 2.5'	driz_cr.SNR parameter
`driz_cr_scale`	=	'1.2 0.7'	driz_cr.scale parameter

Output files:
   * _blt_deriv.fits
   * _cor.fits
   * _crderiv.pl

This uses the original input images, the blotted images, and the derivative of the blotted images (created using the *dither.deriv* task) to create cosmic ray masks (using the *dither.driz_cr* task), stored as separate files, which can later be combined with other masks. This step also creates a '_cor' image, where bad pixels are replaced with pixels from the blotted median image. These relatively clean '_cor' images can be used to determine shifts, if desired.

7. Perform Final Drizzle Combination

Parameters:

`final_outnx`	=		Output image x-size
`final_outny`	=		Output image y-size
`final_kernel`	=	'square|point|gaussian|turbo|tophat'	Drizzle kernel
`final_scale`	=	1.0	Size of output pixels
`final_pixfrac`	=	1.0	Size of 'drop'
`final_rot`	=	0.	Rotation (anticlockwise)
`final_fillval`	=	INDEF	Value for undefined pixels

Output files:
  output_sci.fits
  output_wht.fits

This takes the original input images, together with the final cosmic ray masks, and drizzles them all onto a single output image. The standard *drizzle* parameters of `kernel`, `scale`, `pixfrac` and `rot` can be specified for this task. By default the pixel scale of the output image is 1, but feel free to experiment with other options (e.g., when combining at least 4 sub-pixel dithered images, `scale` = 0.5 and `pixfrac` = 0.7 yields a sharper output PSF).

Future Enhancements

The next major enhancement to this script will consist of the ability to iteratively refine the shifts between images. This involves the use of image-based catalogs, or alternatively cross-correlation techniques, to directly determine the shifts from the data in the images. Prototype versions of these techniques are being successfully implemented, and a robust version will be included in a subsequent public release of *MultiDrizzle*.