Introduction to AstroDrizzle

This is an unofficial document for internal RIAB training purposes only

Roberto J. Avila
Space Telescope Science Institute
3700 San Martin Dr.
Baltimore, MD 21218
Version 1.1

July 2014

Abstract

This document is an introduction to the DrizzlePac software. It was written for the purposes of training new members of the Research and Instrument Analysis Branch (RIAB) at the Space Telescope Science Institute (STScI). You will find a brief introduction to dithering, the history and theory of the Drizzle algorithm, a simple introduction on how to use the DrizzlePac interface within Pyraf, and a set of exercises intended to make trainees familiar with the two main tasks, TweakReg and AstroDrizzle.

For detailed information on everything concerning DrizzlePac, please consult the DrizzlePac website[1] and handbook(Gonzaga et al., 2012).

[1] http://drizzlepac.stsci.edu

Revision History

Version	Date	Editors and Contributors
1.1	July 2014	Roberto J. Avila
1.0	August 2012	Roberto J. Avila

I thank Jennifer Mack, Sara Ogaz and Max Mutchler for their helpful comments.

Send comments and corrections to:
Space Telescope Science Institute
3700 San Martin Dr.
Baltimore, MD 21218
avila@stsci.edu

Contents

1 The Drizzling Concept

1.1 Introduction & History

DrizzlePac is a Python package that contains a suite of tasks used to remove distortion and combine images from the Hubble Space Telescope (HST) imaging cameras. The tasks included in this package provide facilities for aligning images, correcting geometric distortions, removing cosmic rays and other artifacts, and combining images into a single science quality output product.

The primary underlying algorithm, called Drizzle (Fruchter and Hook, 2002), was developed as a way to combine dithered images. Later, MultiDrizzle was developed as a wrapper script that performs all the operations mentioned above. The AstroDrizzle task incorporates fundamental changes in the way geometric distortion and image alignment information is handled. It has been completely rewritten in C and Python and contains capabilities for multiprocessing to speed up computing time.

1.2 Before Drizzling, Dither

Drizzling goes together with another concept of observing and image processing, dithering. Dithering is when you spatially offset the telescope by shifts that are generally smaller than the size of the detector. This places the target at different locations on the chip between the different images. Dithering is done so that detector artifacts (bad pixels, bad columns, chip gaps, blobs in the IR detector, etc.) can be removed during image combination. Sub-integer pixel offsets improve the sampling of the point spread function (PSF). Note that you also get the benefit of multiple images for cosmic ray rejection. Large offsets can also be applied in order to map areas of the sky that are larger than the detector. This is called "mosaicing".

The details of dithering and the different types of dithering strategies is a topic beyond the scope of this document. For more information consult the HST Dither Handbook (Koekemoer et al., 2002)

1.3 Why Drizzle?

Images obtained from the Mikulski Archive for Space Telescopes (MAST) are bias, dark, and flat field corrected using the standard On The Fly Reprocessing (OTFR) pipeline corresponding to each instrument (except WFPC2, which is a static archive). These individual images still contain artifacts that need to be corrected and removed before a stack of images can be combined. An example of these artifacts is shown in figure 1. This is an image taken with the Advanced Camera for Surveys Wide Field Channel (ACS/WFC). This single frame shows cosmic rays, bad pixels, and bad columns.

The imperfection that is not immediately apparent from looking at pipeline products is the geometric distortion. Each instrument has corrective optics to compensate for spherical aberration in the HST primary mirror, but in order to avoid coma distortion the CCD chips are tilted with respect to the focal plane. Because of this tilt the area of the sky covered by each CCD is not a rectangle but a rhombus, with varying degrees of skewness depending

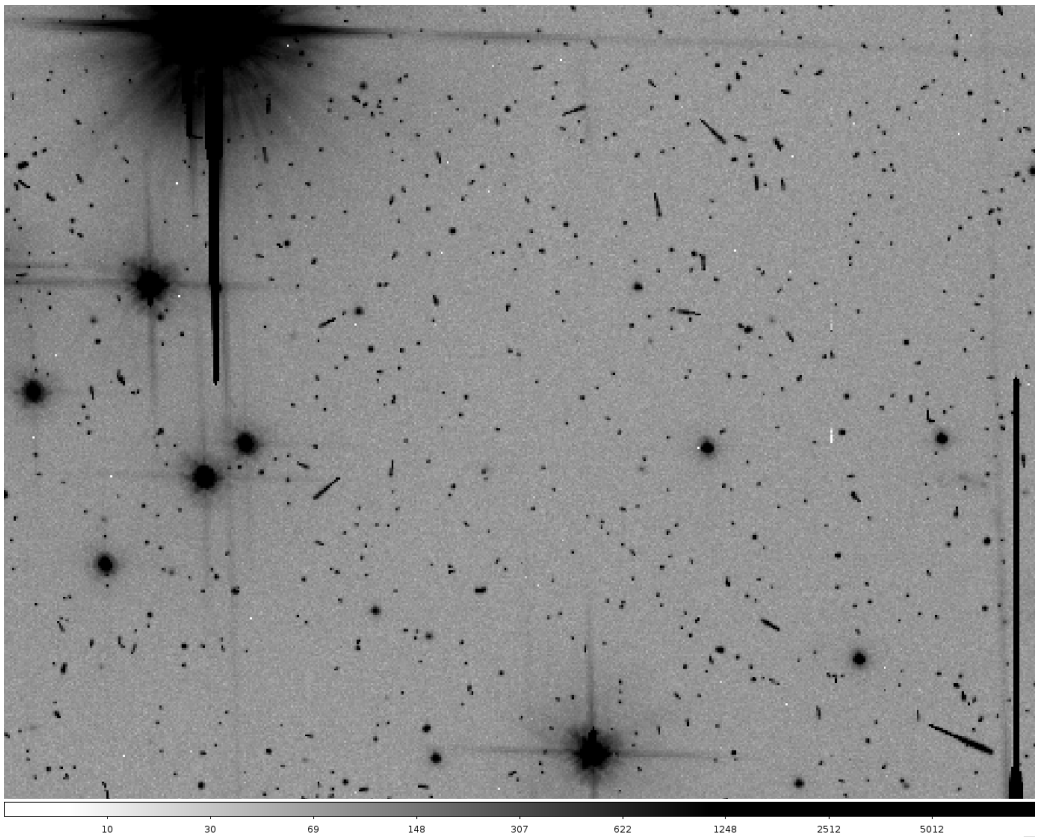


Figure 1: Section of an ACS/WFC image of globular cluster NGC6791 (GO-9815). This image shows cosmic rays, bad pixels, and bad columns.

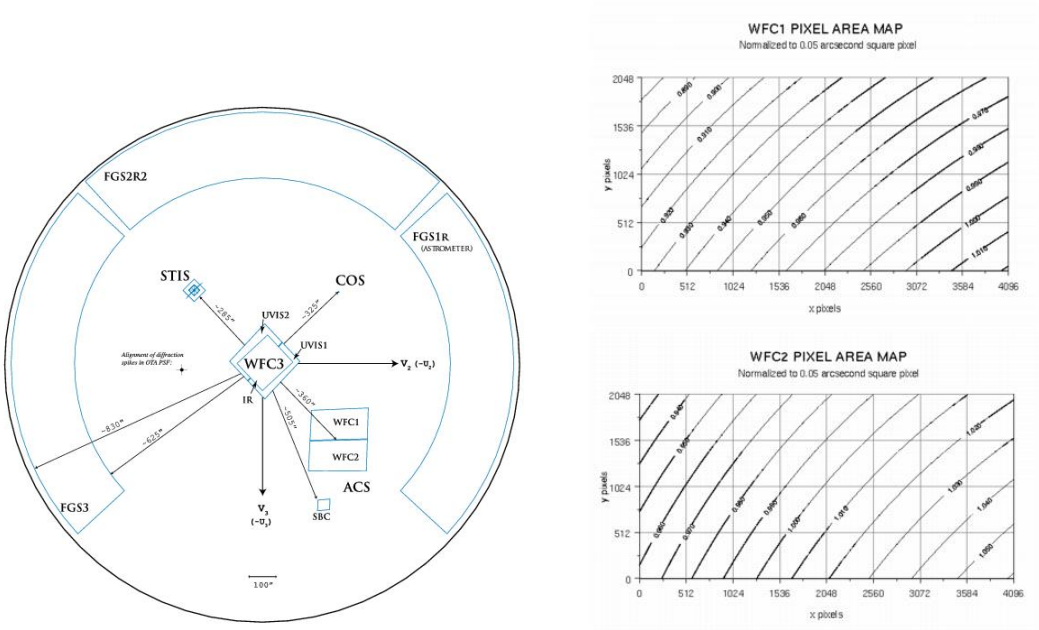
on the instrument (figure 2). This would obviously affect any astrometry measurements obtained using this image.

Additionally, The area of the sky covered by each individual pixel varies across the detector. This introduces an apparent variation in surface brightness across the field of view. Drizzling removes these effects by correcting the distortion and making the sky coverage of every pixel in the output image equal in size.

Note that drizzling is not required and one can still use the individual images for photometric analysis by applying a pixel area map. Pixel area maps contain the effective area of each pixel normalized to the value of the central pixel. This is shown as a contour plot in figure 2. This method is usually applied to single images since drizzling smears the PSF by resampling and convolving it with a kernel.

1.4 The Drizzle Concept

There are many techniques for combining and reconstructing images after they have been convolved with a telescope's PSF and sampled by the detector's pixels. Two of these techniques are called *interlacing* and *shift-and-add*.



(a) HST focal plane

(b) ACS pixel area map

Figure 2: *Left*: Location of the field of view of each instrument in the HST focal plane. Note that the footprints for the imaging cameras are rhombus in shape. *Right*: The pixel area map for ACS/WFC.

In the interlacing method, pixels from the independent input images are placed in alternate pixels on the output image according to the alignment of the pixel centers in the original images. This can only be done if the dithers are well placed. However, due to occasional small positioning errors by the telescope and non-uniform shifts in pixel space across the detector caused by geometric distortion of the optics, true interlacing of images is generally not feasible.

In the shift-and-add method each input pixel is block-replicated onto a finer subsampled grid, shifted into place, and added to the output image. Shift-and-add has the advantage of being able to easily handle arbitrary dither positions. However, it convolves the image yet again with the original pixel, thus adding to the blurring of the image and to the correlation of noise in the image. Furthermore, it is difficult to use shift-and-add in the presence of missing data (e.g., from cosmic rays) and geometric distortion.

In response to the limitations of the two techniques described above, an improved method known formally as variable-pixel linear reconstruction, and more commonly referred to as Drizzle, was developed by Andy Fruchter and Richard Hook (1997), initially for the purposes of combining dithered images of the Hubble Deep Field North (HDF-N). This algorithm can be thought of as a continuous set of linear functions that vary smoothly between the optimum linear combination technique (interlacing) and shift-and-add. This often allows an improvement in resolution and a reduction in correlated noise, compared with images produced by only using shift-and-add.

The degree to which the algorithm departs from interlacing and moves towards shift-and-add

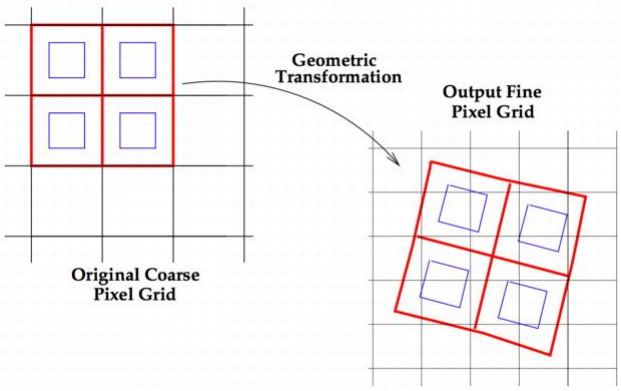


Figure 3: The Drizzling algorithm. Pixels in each individual image are shrunk and then placed on the output grid. The input and output pixels will not align properly so the value of the output pixel is a weighted combination of the input pixels that fall into its area. The dithered images in the rest of the stack would fill in the rest of the information in the output grid shown here. Figure taken from Fruchter and Hook (2002).

depends upon how well the PSF is subsampled by the shifts in the input images. In practice, the degree of interlacing of the Drizzle algorithm is controlled through the use of a parameter called pixfrac, which can be set to values ranging from 0 to 1, that represents the amount by which input pixels are shrunk before being mapped onto the output image plane.

A key to understanding the use of pixfrac is to realize that a CCD image can be thought of as the true image convolved first by the optics, then by the pixel response function (ideally a square the size of a pixel), and then sampled by a delta-function at the center of each pixel. A CCD image is thus a set of point samples of a continuous two-dimensional function. Hence the natural value of pixfrac is 0, which corresponds to pure interlacing. Setting pixfrac to values greater than 0 causes additional broadening of the output PSF by convolving the original PSF with pixels of non-zero size. Thus, setting pixfrac to its maximum value of 1 is equivalent to shift-and-add, the other extreme of linear combination, in which the output image PSF has been smeared by a convolution

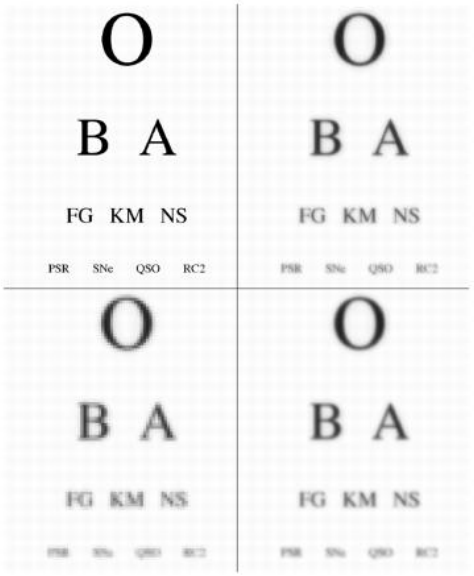


Figure 4: *Top left*: "True image", i.e., the image one would see with an infinitely large telescope. *Top right*: Imager after convolution with the optics of HST and the WFPC2 camera. *Bottom left*: Image after sampling by the WFPC2 CCD. *Bottom right*: Linear reconstruction of dithered CCD images. Figure taken from Fruchter and Hook (2002).

with the full size of the original input pixels. The Drizzle algorithm has also been designed to handle large dithers, where geometric distortion causes non-uniform subsampling across the field and takes into account missing data resulting from cosmic rays and bad pixels.

Some of the instruments under-sample the point spread function (PSF), but if observations are taken using sub-pixel dithering some of the information can be recovered and the PSF reconstructed using the Drizzle algorithm. This will improve the resolution of the image from the native resolution of the instrument (figure 4).

1.5 The Drizzle Process

As explained in section 1.3, before a set of images can be combined, artifacts from the individual input images must masked and geometric corrections need to be applied. AstroDrizzle is the task within DrizzlePac that handles all these procedures. The interface itself, and some of the parameters that the task uses will be explained in the following section.

The task goes through a series of steps to create the final science product. Here we present a rough outline of the steps involved in drizzling images together, although in practice there are many more than those described here:

• The geometric distortion is removed from each input image. These intermediate images are called ***single_sci.fits**. The data quality (DQ) flags of the input images are used to mask bad pixels that should be excluded in the median combination.

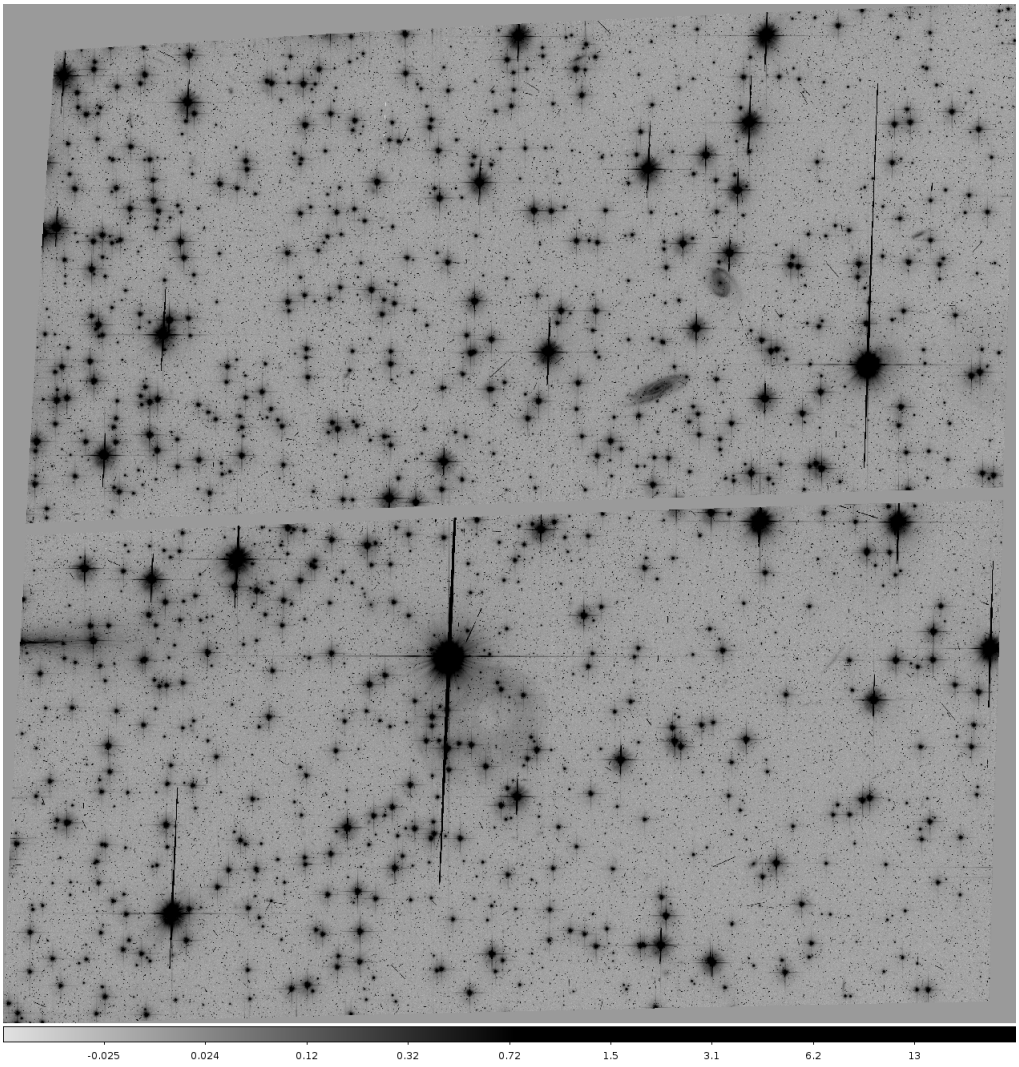


Figure 5: Full frame of a single, undistorted image.

• The undistorted images are combined to create a clean median image.

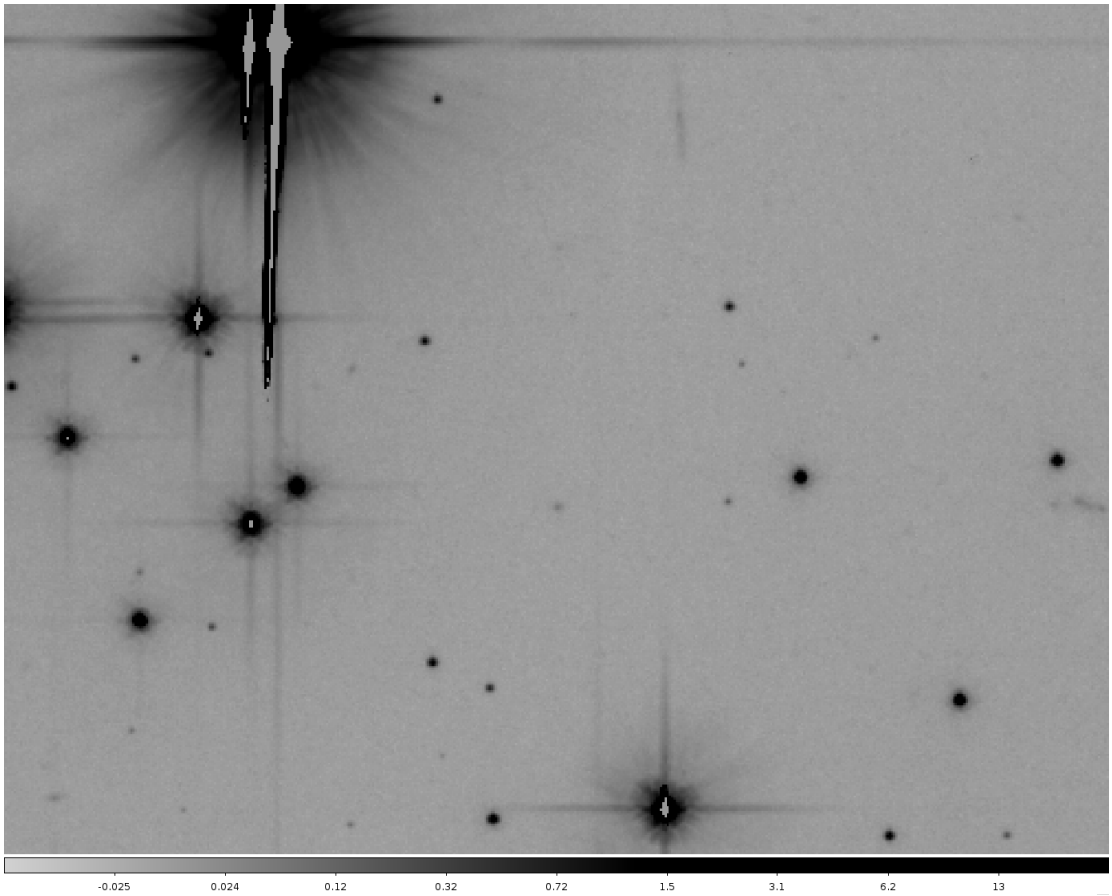


Figure 6: Close up of the median image. This is the same region of the image as in figure 1. In this close-up the diffraction spikes are perpendicular to each other, showing that distortion has been removed.

• The median image is blotted back (reverse drizzled) to the distorted frame of each input image. These intermediate images are called ***_sci#_blt.fits**.

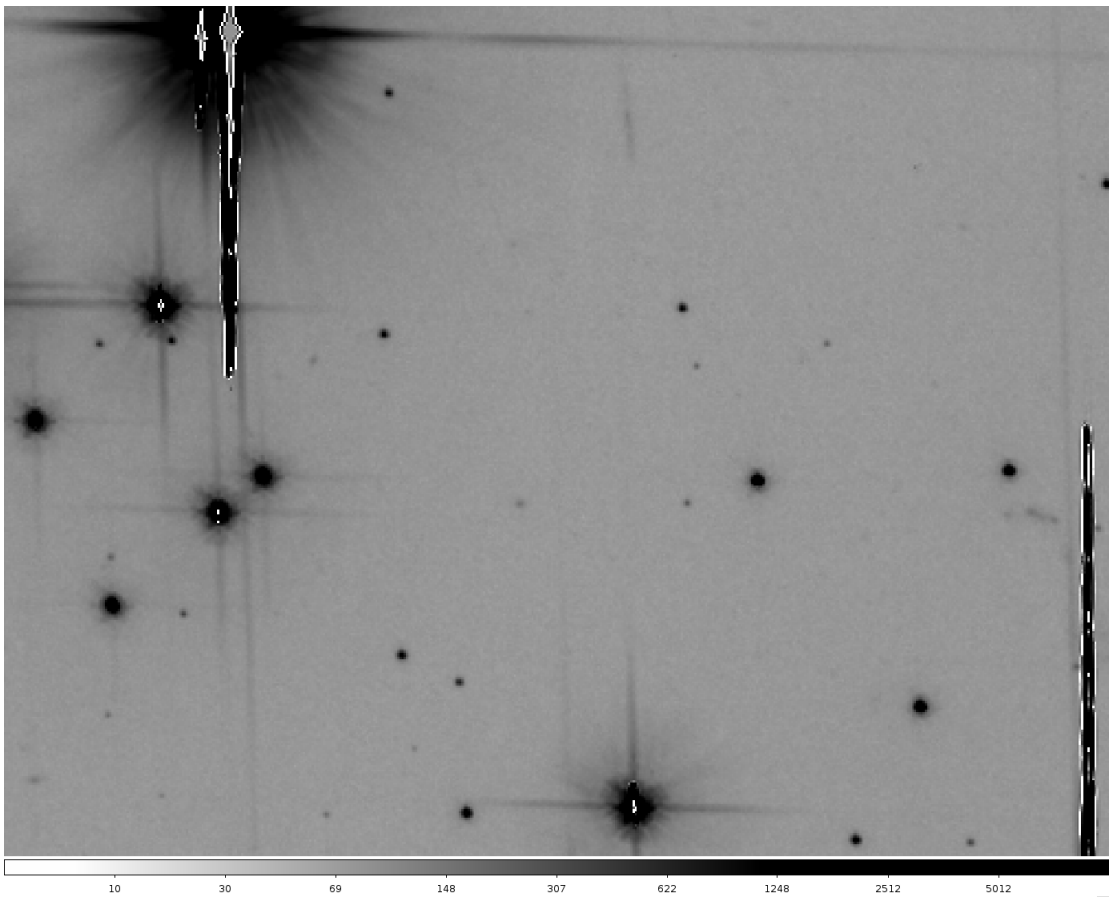


Figure 7: Close up of of a blotted image. This is the same region of the image as in figures 1 and 6. Note that the diffraction spikes are not perpendicular, indicating that this is a distorted image.

• A CR mask is created by comparing each input image to its corresponding blotted image. Cosmic rays are flagged using the following rule:

$|data\_image - blotted\_image| > scale \times deriv + SNR \times noise$

where "scale" and "SNR" correspond to the *driz_cr_scale* and *driz_cr_snr* parameters (Step 6) in the AstroDrizzle task.

The single and CR masks are combined to create a final mask which contains all the pixels that should be excluded from the final combination. Final mask files are called ***_sci#_final_mask.fits**.

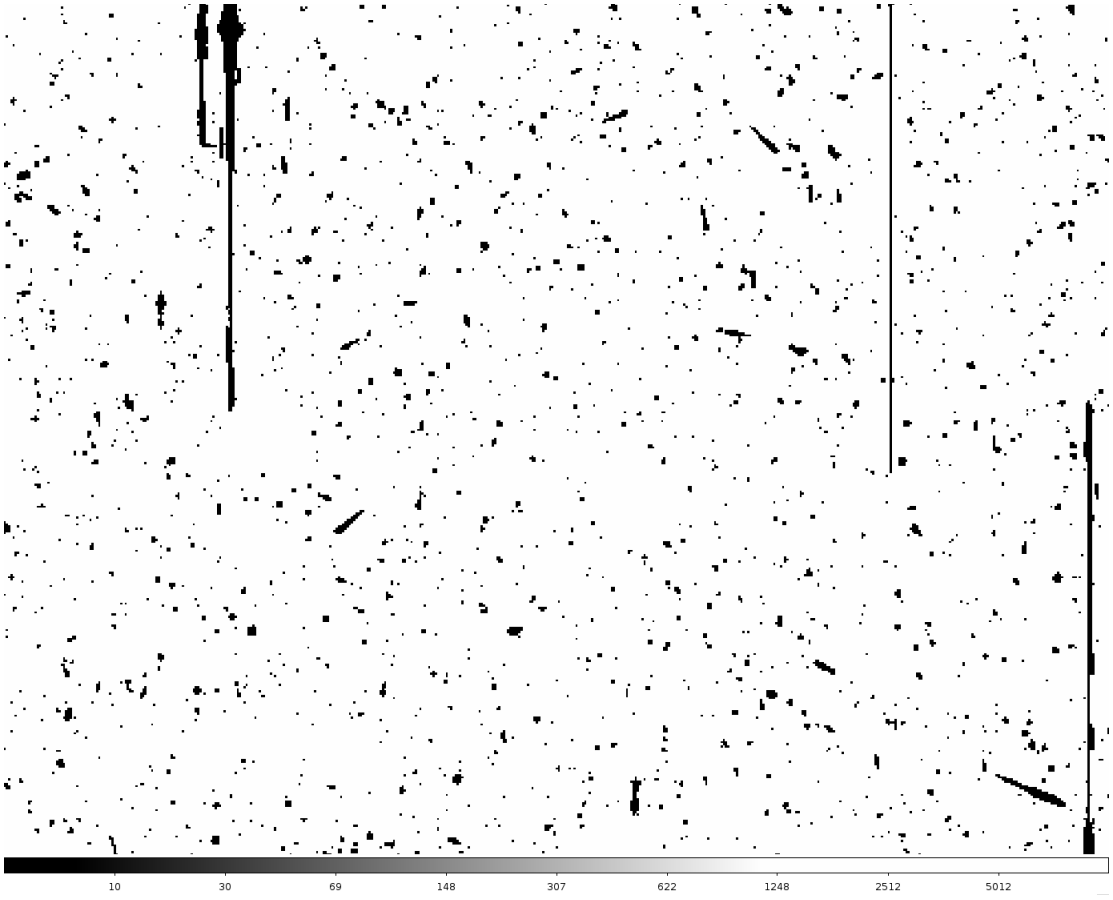


Figure 8: Close up of the final mask file. This is the same region of the image as in figures 1, 6, and 7.

• Using the information in the final mask file, the individual images are drizzled into a final science product.

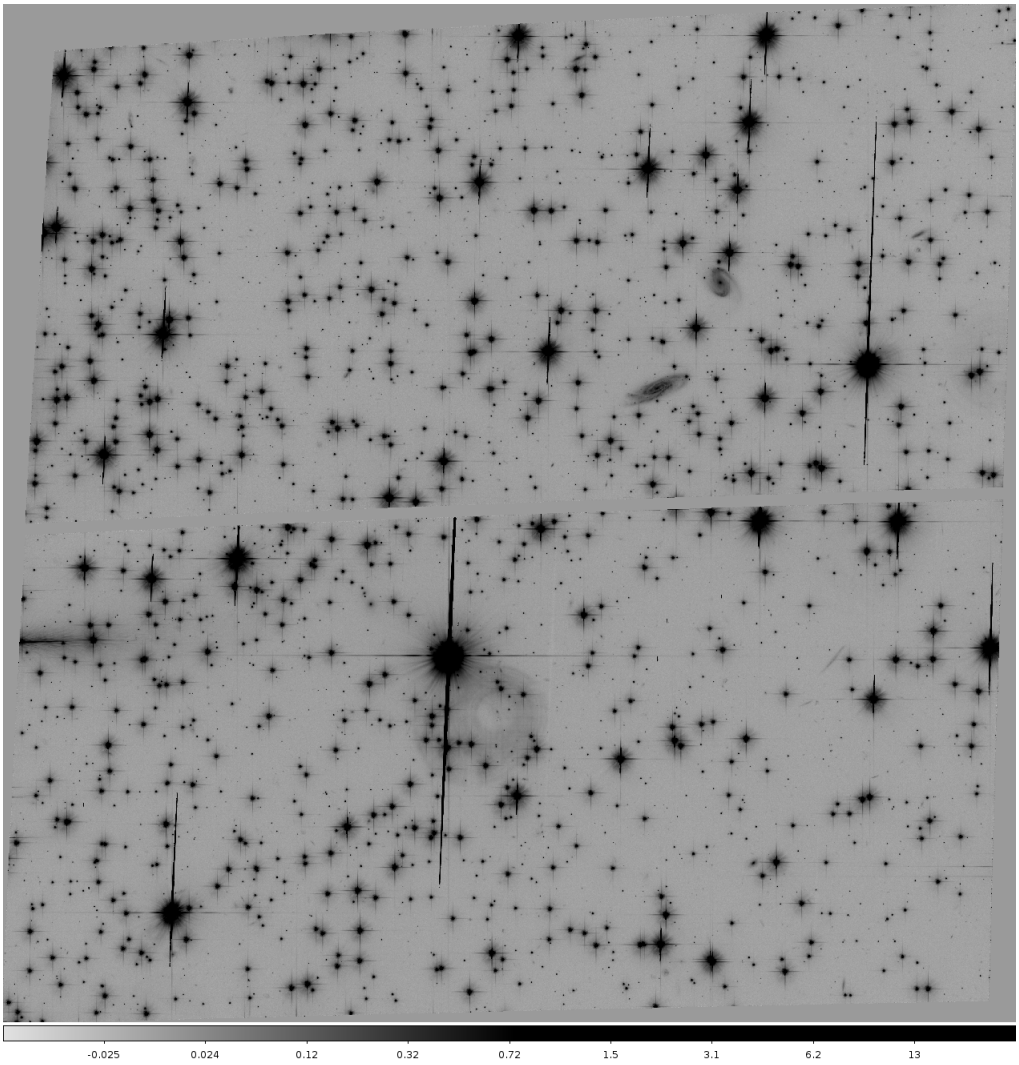


Figure 9: Full frame of the final drizzled product. Displayed with the same stretch and scale as figure 5.

To recap, these are roughly the steps in the Drizzling algorithm:

- Distortion is removed from each individual image
- The undistorted images are combined to create a clean median image
- The median image is blotted back to the distorted frame of each input image
- Mask files are created by comparing input images to blotted images
- Final combined product is made using information from masks and input images

It is during the final step where a user can elect to modify the Drizzle parameters to improve the image resolution. More on this in section 2.4.

2 Using DrizzlePac

2.1 Requirements

Before you begin, you need to install the DrizzlePac package. To do so download and install the SSBX software bundle from the SSB website. That page contains instructions on how to install this software[2].

The data set you will be using for these exercises can be found at '/user/avila/training/ngc6791/'. Table 1 describes the data set. These are ACS images of globular cluster NGC6791 taken in two broadband filters.

File	Date-obs	Filter	Exptime (s)	Postarg_x	Postarg_y
j8ny01svq_flc.fits	2003-07-16	F606W	1142.0	0.0000	0.0000
j8ny01szq_flc.fits	2003-07-16	F606W	1142.0	0.1412	0.0363
j8ny01t5q_flc.fits	2003-07-16	F606W	1185.0	0.0748	0.1476
j8ny01t9q_flc.fits	2003-07-16	F606W	1185.0	-0.0166	0.1070
j8ny01tfq_flc.fits	2003-07-16	F606W	1185.0	0.0665	-0.0114
j8ny01ttq_flc.fits	2003-07-16	F606W	1185.0	0.0581	0.0713
j8ny02yvq_flc.fits	2003-07-17	F814W	1142.0	0.0000	0.0000
j8ny02z1q_flc.fits	2003-07-17	F814W	1142.0	0.1412	0.0363
j8ny02zdq_flc.fits	2003-07-17	F814W	1185.0	0.0748	0.1476
j8ny02zkq_flc.fits	2003-07-17	F814W	1185.0	-0.0166	0.1070
j8ny02ztq_flc.fits	2003-07-17	F814W	1185.0	0.0665	-0.0114
j8ny02zyq_flc.fits	2003-07-17	F814W	1185.0	0.0581	0.0713

Table 1: Header keywords for the images that will be used in the exercises. Note that each filter set was observed on different visits so the alignment will have to be checked (see section 5.1 of the HST Data Handbook (Smith et al., 2011) for details on how to decipher file root names). The *postarg* values indicate how much the telescope was dithered (in arcseconds).

2.2 The Pyraf Interface

DrizzlePac can be accessed through several different ways – Pyraf GUI inteface, Python command line, or through Python scripts. There is no wrong way to do it and you should find the way which best fits your workflow. In this document we will use the Pyraf interface because it allows us to still use some IRAF tasks for examining our images. In addition we will be using the DS9 FITS viewer to display our images. I use DS9 7.0 so there might be slight differences depending on what you have on your computer. For more information on the other ways to access DrizzlePac, read Chapter 5 of the DrizzlePac Handbook (Gonzaga et al., 2012).

Before starting Pyraf you want to ensure you are in the correct SSB environment. To access the SSBX environment simply type `ur_setup common ssbx` in a command line and then `pyraf`.

[2] http://ssb.stsci.edu/ssb_software.shtml

Once you are in the Pyraf command line (denoted by the "−− >" prompt), import the DrizzlePac package as you would any other Python package.

```
 > ur_setup common ssbx
 > pyraf
setting terminal type to xterm...

   NOAO/IRAF PC-IRAF Revision 2.16 EXPORT Thu May 24 15:41:17 MST 2012
      This is the EXPORT version of IRAF V2.16 supporting PC systems.


  Welcome to IRAF.  To list the available commands, type ? or ??.  To get
  detailed information about a command, type 'help <command>'.  To run  a
  command  or  load  a  package,  type  its name.   Type  'bye' to exit a
  package, or 'logout' to get out  of the CL.    Type 'news' to find  out
  what is new in the version of the system you are using.

  Visit http://iraf.net if you have questions or to report problems.

  The following commands or packages are currently defined:

  (Updated on 2014-7-4)

clpackage/:
 adccdrom/      deitab/         language/       optic/          system/
 apropos        esowfi/         lists/          plot/           tables/
 cfh12k/        finder/         mem0/           proto/          ucsclris/
 cirred/        fitsutil/       mscdb/          rvsao/          upsqiid/
 clpackage/     gemini/         mscred/         softools/       user/
 ctio/          gmisc/          mtools/         song/           utilities/
 cutoutpkg/     guiapps/        nfextern/       sqiid/          vo/
 dataio/        images/         noao/           stecf/          xdimsum/
 dbms/          kepler/         obsolete/       stsdas/         xray/
PyRAF 2.2.dev2218 Copyright (c) 2002 AURA
Python 2.7.5 Copyright (c) 2001-2013 Python Software Foundation.
Python/CL command line wrapper
  .help describes executive commands
--> import drizzlepac
The following tasks in the stwcs.gui package can be run with TEAL:
     apply_headerlet          archive_headerlet          attach_headerlet
     delete_headerlet         extract_headerlet         headerlet_summary
    restore_headerlet             updatewcs              write_headerlet


The following task in the fitsblender package can be run with TEAL:
```

```
                                  blendheaders
The following tasks in the drizzlepac package can be run with TEAL:
    astrodrizzle       imagefindpars          pixtopix            pixtosky
     resetbits            skytopix           tweakback           tweakreg
     updatenpol
-->
```

Pyraf lists the tasks available to you from IRAF. After importing DrizzlePac you also get a listing of the tasks included within that package. All the tasks can be accessed using the usual `epar` command followed by the task name. The help files for each task can be accessed from the GUI by selecting '<taskname> Help' from the pull-down menu in the upper right corner of the window. For training purposes, we will concentrate on the two most common tasks, **TweakReg** and **AstroDrizzle**.

2.3 TweakReg

In order for the Drizzle algorithm to work, input images must be properly aligned. For images taken during the same visit, the alignment is usually better than ~ 0.1 pix. In many cases though images come from different visits. In those cases the image alignment could be off as a result of uncertainties in the guide star catalog (as large as 2.0"). When this occurs the images must be aligned before using AstroDrizzle.

To align the images, the world coordinate system (WCS) in the header must be tweaked. The task used to do this is called TweakReg. TweakReg takes input images and aligns them to a common reference frame in the header WCS. You can use an image from your stack as the reference or any image from an external source (e.g. SDSS, Spitzer) that uses the standard conventions for WCS in the headers.

These are roughly the steps that TweakReg goes through to align images:

1. Makes source catalog for the input and reference image using an algorithm similar to **daofind**
2. Use the distortion information of the image to project the pixel position of each object to a sky position
3. Find and match sources common to both images and determine the residual offsets between the two images
4. Determine the offset, rotation, and scale needed to align the images
5. Update the headers with the new tweaked WCS solution

To get a handle of how good the alignment is before we make any changes, display all the images in DS9. Align the images by WCS (from the menu go to Frame → Match → Frame → WCS). As you tab through the images you should see that the alignment between images is already good.

We will align the first two images in the list together. Launch the GUI editor for TweakReg and change the following parameters:

input = j8ny01szq_flc.fits

refimage = j8ny01svq_flc.fits

conv_width = 3.5

threshold = 100.

Notice that as you change parameter values their description in the GUI window changes color. This is a feature to let you know what parameters are not set to their default values.

The first two parameters are the input file and the image to use as reference. The next two parameters are found by pressing the **imagefind Parameters** button. These parameters control how sources will be detected in the images. This field has many stars, but only a few are needed to get a good alignment solution. The *threshold* parameter ensures that we only find bright objects. This will eliminate faint sources which are bad for alignment and false detections from saturated pixels. Be sure to press the **Save & Quit** button when you are finished in the Imagefindpars window.

The search parameters in the main window tell tweakreg how big to make the search radius around each star for a match. For now we will use the default values.

Before executing the task, save the parameter values to a configuration file (***.cfg**). This is done by selecting "Save as..." from the top left pull down menu in the GUI. This will save a text file with the current set of parameters for this task. You should inspect this file with a text editor. Press **Execute** and the task will begin running.

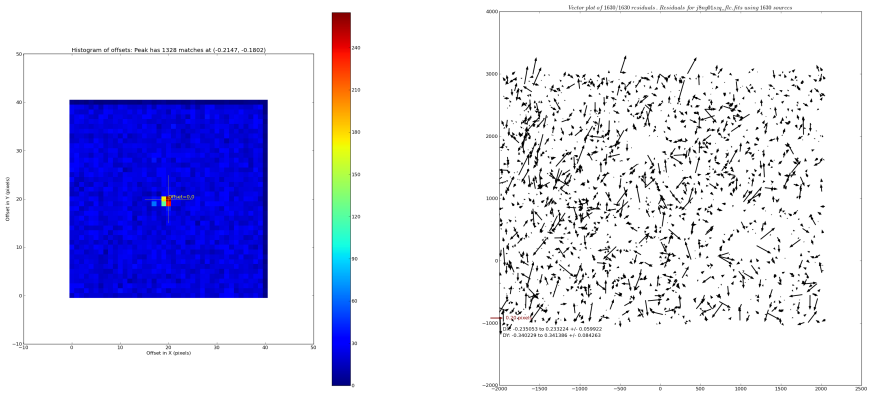
You will see messages from the task telling you what it is doing. All this information will also be written out to the logfile. Once the source detection is finished the task will display 3 figures with diagnostic plots (figure 10).

The first plot is a two dimensional histogram of the *dx* and *dy* found for all the stars. Good stars that match will have very similar offsets, this is the peak you see in the histogram. The value of this peak is what TweakReg uses as the first guess for the offset between the two images. The other two plots show the offset residuals, in two forms, after an analytic solution has been found. The second plot is a vector plot of the residuals. A good solution is indicated when the direction of the vectors is isotropic and their sizes are small (the scale of the vectors is shown in the lower left of the figure). The final plot shows the residuals in *dx* and *dy* with respect to X and Y. You should not see any systematic effects in this plot if you have a good solution. The XRMS and YRMS values should be < 0.1 pix, preferably below 0.05 pix.

This is an acceptable solution. To apply it to your image re-run TweakReg with the parameter *updatehdr=yes*. This will repeat the process you just conducted and at the end update the image header with the new WCS solution.

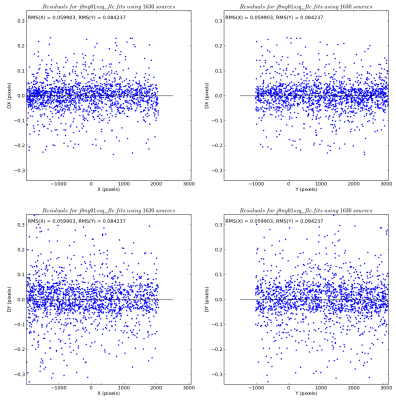
2.4 AstroDrizzle

Now that you have aligned the images you can finally combine them using AstroDrizzle, the workhorse task of DrizzlePac.



(a) 2D histogram

(b) Residual vectors



(c) Residuals

Figure 10: The three diagnostic plots displayed by `tweakreg`. See text for details.

You will need to separate the images into different directories for each filter. Go to your F606W images and bring up the AstroDrizzle GUI. You will see that it contains a large number of parameters. The first set of parameter (before Step 1) control the overall behavior of the task. After that you will see parameters grouped by steps. You can turn any step on or off. Whenever you turn off a step, the parameters that control that section will be grayed out and will not be editable (e.g. Step 3a is turned off by default). Notice that the steps follow the outline of the drizzle process described in section 1.5.

For now we will only change a few parameters:

input = *flc.fits

output = f606w

num_cores = 2

driz_cr_corr = Yes

The *input* parameter accepts wildcards, lists, or @lists just like any other Pyraf task. Here we have changed it to find all the "*flc.fits" files in the current directory. The *output* parameter is just the rootname for final drizzled image. Finally, we have changed *num_cores* so that the task doesn't try to use all the cores in your computer. AstroDrizzle can be memory intensive and if you don't have enough RAM your computer could be slowed down to a crawl. If you want to know what the rest of the parameters do consult the help file from the GUI window.

Press **Execute** and the task will begin running. You will see messages scroll by giving you information on what is going on.

When the task is finished, your directory will contain a new set of images for each input image. They are described below:

sci#_blt.fits : Blotted image for each science extension

sci#_crmask.fits : Cosmic ray mask for each science extension

sci#_final_mask.fits : Mask containing all flagged pixels for each science extension

sci#_single_mask.fits : Mask containing pixels flagged as bad in the input data quality array

single_sci.fits : Undistorted input image

single_wht.fits : Weight map for each undistorted input image

You will also find "f606w_drc" files. These are the final science, weight, and context images. The drizzled image is called "f606w_drc_sci.fits". Open this file in DS9 to see your combined image. This image should look like figure 9. Also, inspect the basic image attributes using the `fits.info()` command included in the astropy package (which replaced pyfits some time ago).

```
--> from astropy.io import fits
--> pyfits.info('f606w_drc_sci.fits')
Filename: f606w_drc_sci.fits
No.    Name         Type      Cards   Dimensions   Format
0    PRIMARY     PrimaryHDU     918   (4216, 4242)   float32
1    HDRTAB      BinTableHDU    591   12R x 291C   [9A, 3A, J, D, D, D, D, D, D, D, D, D,
-->
```

You can see that the image size is 4216x4242 pixels. This size was chosen automatically by the task. You will learn how to control this attribute in the AstroDrizzle exercise. Finally, save this image somewhere it will not be overwritten. You will compare this image the your results from the exercise.

3 Exercises

By now you should have disk space in central storage for personal use (/user/<username>). You will be placing the results of these exercises in this directory. You should keep your working directory for these exercises in your machine since these are big files and you don't want to be loading them to memory from a remote disk. Do not hesitate to ask if you have questions or run into any problems when doing these exercises.

3.1 TweakReg Exercise

Use tweakreg to align all the images to the first image of the stack. The parameters used in section 2.3 should produce good alignments, but you should play with *peakmin* and *peakmax* to speed up the source finding process and with *nclip* and *sigma* to reduce your residual RMS. When you find a solution, save the parameters you used to a text file, update the headers, create a logfile and a shiftfile. Place these files in your results folder.

3.2 AstroDrizzle Exercise

Make a drizzled image for each filter. Start with the default settings and adjust parameters so that your final images have the following characteristics:

1. Both images have the same size and center. See section 7.5.6 of the DrizzlePac handbook for two methods to do this.
2. Rotated so that north is up.
3. Have pixel scale of 0.03"/pix.
4. Have pixfrac = 0.6

Compare the image you made in the previous section to the one you make here. The one from the exercise should be larger but have higher resolution and rotated in a different direction.

Place the two drizzled science images, the configuration files, and the log files in your results folder.

References

Fruchter, A. and Hook, R. N.: 1997, in A. G. Tescher (ed.), *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Vol. 3164 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, pp 120–125

Fruchter, A. S. and Hook, R. N.: 2002, *PASP* **114**, 144

Gonzaga, S., Hack, W., Fruchter, A., and Mack, J.: 2012, *The DrizzlePac Handbook*

Koekemoer, A., Gonzaga, S., Fruchter, A., Biretta, J., Casertano, S., Hsu, J.-C., Lallo, M., Mutchler, M., and Hook, R.: 2002, *HST Dither Handbook*

Smith, E., Boestrom, A., Bushous, H., Casertano, S., Hack, W., Karakla, D., and Lallo, M.: 2011, *Introduction to the HST Data Handbooks*