Technical Report SSB 2003-001

Providing Updated Image Shifts to PyDrizzle

Warren Hack, Jennifer Mack, Richard Hook
May 6, 2003

ABSTRACT

Proper image registration during image combination relies on accurately determining the offsets between images. These offsets may take many different forms depending on how they are measured. This reports describes conventions which PyDrizzle Version 4.0 (and later) will use when interpreting offsets between images that are being processed. The format of an ASCII file will be described as one method for providing the offsets to PyDrizzle. We will describe updates to PyDrizzle which support additional image shifts in the association table. Finally, an example demonstrates how these elements work together to provide an improved output product.

Introduction

The ability of PyDrizzle to combine dithered observations relies on the accuracy of the WCS information in the headers to properly align the input images. Unfortunately, there are times when those WCS values are inaccurate resulting in a mis-alignment of the final output drizzled product. Derivation of the shifts between images based on the position of objects in the data provides a way to verify or refine the shifts based solely on the WCS header values. Methods such as image source matching or cross-correlation techniques can independently determine the offsets between images allowing for extremely accurate alignment when drizzling the images together. This proposal outlines a set of conventions which can be implemented to allow PyDrizzle to use independently computed offsets between images. These conventions will be implemented for use by PyDrizzle Version 4.0 and later.

Computing Shifts

This report will begin with a definition of a few key terms related to computing image offsets and the conventions used to represent those shifts.

relative shift:

the total shift derived between an undistorted input image and an undistorted reference image.

delta shift:

the residual shift found after applying the shift computed from the header WCS values.

input frame:

undistorted WCS frame obtained by applying the distortion model to an input, distorted image **without** applying any additional rotations or scale changes.

output frame:

undistorted WCS frame obtained by applying the distortion model to a distorted input image **including** additional rotations and/or scale changes.

pixel shift:

shift computed as the simple difference between the X/Y pixel position from the input image minus the X/Y pixel position of the same source from the reference image. These X/Y positions can be derived from images in either the input or output frame and can be either relative shifts or delta shifts.

arcsecond shift:

shift computed as the difference between the RA/Dec position of a source from an input image minus the RA/Dec position given for the same source from the reference image. These positions MUST be determined after correcting for any distortion in the image. PyDrizzle then directly applies this shift to the CRVAL position of the input image.

NOTE: These arcsecond shifts are NOT the same as angular offsets on the sky, and do not require any 'cos(Dec)' adjustments as they are only changes in the RA/Dec of the reference pixel.

shift file:

ASCII file containing the separately computed shifts for a set of images used as input to the *buildasn* task for updating the offsets provided in the ASN table.

The definitions for the input and output frames provided here correspond to the definitions used by 'drizzle' allowing for identical use of shifts for both tasks.

Relative and Delta shifts

Delta shifts can be determined by separately 'drizzling' the input images onto a common WCS frame, just as *MultiDrizzle* does during the 'Driz_separate' stage. Fitting can then be done on either sources from the separate images or with cross-correlation techniques to derive the residual offset between the observations. For clarification, these shifts will be in the 'input' frame when the common output WCS frame has the same orientation and scale as the inputs. Alternatively, the delta shifts would be in the 'output' frame if any rotation and/or scaling gets applied during the drizzling.

Relative shifts, on the other hand, could be determined using any number of methods. Shifts computed in the 'input' frame could be determined by finding targets in each input image prior to correcting for distortion, applying the default distortion model to X/Y positions and fitting those results with lists for the same targets derived from the other images. 'Output' frame shifts could be computed by drizzling each input to separate output images with the same rotation and scaling, but without the same RA/Dec for the center. The resultant images could then be cross-correlated or target lists could be matched to compute the relative shifts.

Shift Units

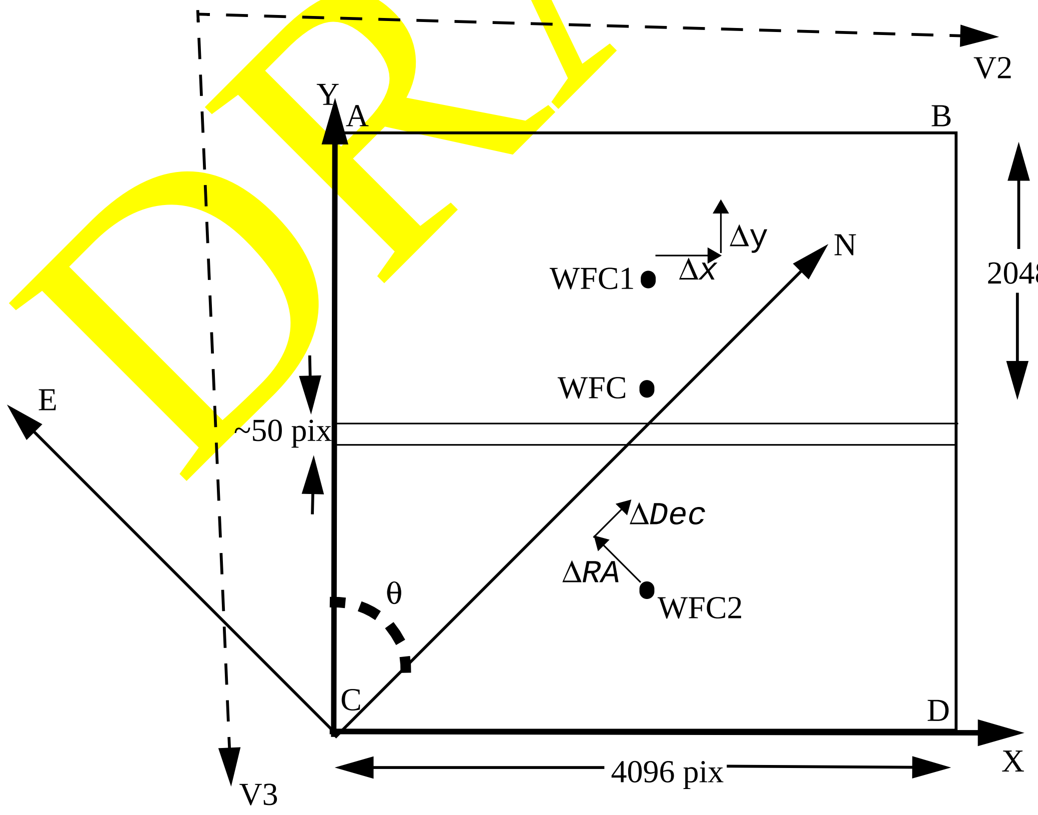
The computed shifts can be specified in units of either *pixels* or *arcseconds*, with PyDrizzle interpreting the shifts accordingly. *Arcsecond shifts* come as a natural result of using RA/Dec coordinates of image sources for computing the shifts. This would eliminate any ambiguity with the reference frame since the RA/Dec coordinates are fixed on the sky. *Arcsecond shifts* get interpreted as shifts with respect to the reference pixel RA/Dec values, so they are interpreted as offsets in direction of increasing RA and Dec. For example, the IRAF task 'xy2rd' can be used to determine the RA/Dec of a star in an image based on the WCS in it's header. If the 'true' position known from another source/image had RA which was larger in value and a Dec which was smaller, then the *arcsecond shift* would need to be positive in RA and negative in Dec to bring the measured position into agreement with the 'true' position.

P*ixel shifts* are interpreted with respect to each image's reference position in the X/Y pixel frame and require a specification of 'input' or 'output' for the frame. *Pixel shifts* given in terms of the 'output' frame designate that the shifts were measured from images's that were already drizzled onto an image which has the same WCS (*i.e.*, plate scale, orientation) as the final output product. PyDrizzle requires specification of the image which defined the 'output' WCS used for computing these 'output' pixel shifts so that they can be converted back to the simple 'input' frame (without scale or orientation changes). This allows PyDrizzle to apply these shifts to any arbitrary output frame regardless of the frame used to compute the shifts in the first place. *Pixel shifts* given as 'input' shifts specify that the shifts were computed based on images which have been distortion-corrected without any additional scale or orientation changes. PyDrizzle then does a final conversion of the shifts based on the WCS of the final output product to account for scale and orientation changes.

The relationship between *pixel shifts* and *arcsecond shifts* can be seen in Figure 1 for the WFC, where both types of shifts are shown for an observation with ORIENTAT=45. It can be clearly seen from this figure that shifts given in units of arcseconds translate to shifts in different directions than shifts given in pixels, underscoring the need to carefully specify the units in the shifts file or ASN table.

Most individual observers have developed their own way to compare images and compute offsets with scripts/tasks with which they are familiar. The conventions described here should support the majority of forms used by observers making it a simple matter to incorporate the results from most shift computations into PyDrizzle.

Figure 1: Relationship between pixel shifts and arcsecond shifts for an observation with ORIENTAT of 45^{o}.



Sign Conventions

Regardless of the form of shifts that are computed, it should be kept in mind that PyDrizzle and 'drizzle' both work with shifts from the observers perspective. In particular, the shifts should be in terms of how the targets in the image need to move to result in precise registration, rather than on how the telescope would need to move. The computation of any shift requires taking the difference between two values; namely, the position from the input image and the position from a reference image. The order of the difference then dictates what sign the shift will have.

PyDrizzle assumes that the values provided by the user in the ASN table were computed as ’image - reference’. For example, if using *geomap* to compute the fit between two source target lists, this corresponds to having the current image’s values in the first two columns of the input table and the reference image in the last two columns (as described in the *geomap* help file).

Use of Shifts in Association Tables

Association (ASN) tables can support either *delta shifts* or *relative shifts* given in units of either ’pixels’ or ’arcseconds’. Table 1 lists the names of the columns for the relative and delta shifts. *Relative shifts* get stored in columns ’XOFFSET’ and ’YOFFSET’ to denote the fact that the shifts represent the offset of the image from the reference image. *Delta shifts* get stored in columns ’XDELTA’ and ’YDELTA’. The units defined for these columns record whether these shifts are given in terms of pixel shifts or shifts in arcseconds of RA/Dec.

In addition, the columns ’ROTATION’ and ’SCALE’ can account for any additional rotations or scale changes found when computing the final shifts. The additional rotation given in the ‘ROTATION’ column can specify how much should be added to the image’s ORIENTAT to get proper alignment. Finally, if there are noticeable scale changes not accounted for in the distortion model for a particular image, the plate scale ratio of expected/measured can be supplied in the ‘SCALE’ column for use by PyDrizzle.

Table 1. Summary of Shifts supported in ASN table

Shift Type	Column Names	Units Supported
relative	XOFFSET, YOFFSET	pixels, arcseconds
delta	XDELTA, YDELTA	pixels, arcseconds

PyDrizzle reads in all columns from the ASN table. If *relative offsets* are provided in the shift file, it will populate the ASN table columns for the OFFSET with the values and populate the DELTA values with all zeroes. Similarly, if *delta shifts* are provided in the shift file, those values will be used to populate the DELTA columns and the OFFSET columns will be filled with zeroes. If shifts are given in both the OFFSET and DELTA columns, the shifts given in the OFFSET column will always be applied ignoring any values in the DELTA columns. This convention used by *buildasn* eliminates any ambiguity as to which values are applied to the data. Editing the ASN table directly will then allow the user to further update the offsets using either *delta* or *relative* shifts.

The ASN table can either be created directly using standard STSDAS tools (such as *tedit* or *tcreate*) or using the *buildasn* task provided as part of the PyDrizzle package. The *buildasn* task accepts ASCII files containing the shift information in a simple, specific format as a *shift file*. The same *shift file* can also be provided as input to *MultiDrizzle* after independently determining improved shifts. The *shift file* uses the following format:

```
# units: pixels          Values: pixels (default), arcseconds
# frame: input           Values: input (default), output
# form: relative         Values: relative (default), delta
# reference: <filename>  Only needed when 'frame == output'
filename xshift yshift [rotation [scale]]
```

The first lines in the file provide information regarding the form of the shifts provided in the file, and only those with non-default values are necessary. The list of values for each line (as shown above) are not a required part of the file, although if present will not introduce any problems. If there are no rotation or scale changes, then the rotation and scale columns do not need to be created at all. However, if there are scale changes to be applied, a rotation must also be provided in order to maintain the proper format for each entry.

This file can contain shifts for files other than those contained in the ASN table. In that case, only those entries found in the ASN table will be used to update the table. This allows one file to be generated for a whole set of observations represented by multiple association tables.

Example

These principles can best be understood through their application to an actual set of dithered observations. This example will be based on a set of three ACS/WFC images taken from a CR-SPLIT=2, 9-point dither pattern taken as part of the ACS calibration program 9018 in the F435W filter. The filenames of the calibrated images used for this example are:

FILENAME	ORIGINAL ROOTNAME
f435w_01_flt.fits	j8c0a1abq
f435w_05_flt.fits	j8c0a1ajq
f435w_18_flt.fits	j8c0a2d6q

The pixel position and coordinates for a single star common to all three images indicate the sense of the shifts between the images. The coordinates were derived using *xy2rd* on the distorted input images, while the pixel positions were obtained using *imcntr* with *cbox=15*. The last two columns are the delta's computed as 'image - reference' with the first image serving as the reference. These can be compared to the actual values computed by PyDrizzle after taking into account the distortion.

f435w_01	2633.25	770.98	0:22:47.86	-72:03:11.2	0.	0.
f435w_05	3513.51	1582.40	0:22:47.79	-72:03:11.0	880.26	811.42
f435w_18	1752.48	1712.77	0:22:47.87	-72:03:11.5	-880.77	941.79

Since these pixel positions were derived from the distorted input images, they are unsuitable for determining the overall shifts between the images. Instead, they are given to illustrate the general direction and magnitudes of the shifts between the images.

Improved image shifts were derived in the input frame for these observations by Jennifer Mack. The images were drizzled onto separate output images, each with the same WCS as the first image in the ASN table using *MultiDrizzle*. Targets were identified in each of the drizzled outputs and *geomap* was used to fit common targets. This resulted in a derivation of *delta shifts* since the shifts indicated by the input headers was already accounted for in the initial drizzling. The original shift file used for this association only reported the *delta shifts*.

```
# form: delta
f435w_01    0.00        0.00
f435w_02   -0.06689    -0.06178

f435w_03    1.22879     0.07208
f435w_04    1.25398     0.06622
f435w_05    2.54834     0.22263
f435w_06    2.55290     0.21753

f435w_07   -1.21249    -0.11886
f435w_08   -1.24936    -0.14741
f435w_09   -2.40082     0.10166
f435w_10   -2.41300     0.10193

f435w_11   -1.96196    -0.87050
f435w_12   -2.03822    -0.84855
f435w_13   -2.78908    -1.42002
f435w_14   -2.85048    -1.36502

f435w_15    0.48301     1.01453
f435w_16    0.37425     0.88224
f435w_17    1.91038     2.34576
f435w_18    1.88253     2.30274
```

PyDrizzle comes with a set of utility tasks separate from the primary drizzling task, including *buildasn* for creating an ASN table. *Buildasn* runs on its own and generates an association table based on files in the working directory or files listed in an ASCII file. It can also take the shifts from a shift file and populate the appropriate columns in the ASN table. The shifts reported in the ASN table upon using *buildasn* are:

	XOFFSET (pixels)	YOFFSET (pixels)	XDELTA (pixels)	YDELTA (pixels)
f435w_01	0.0	0.0	0.0	0.0
f435w_05	0.0	0.0	2.54834	0.22263
f435w_18	0.0	0.0	1.88253	2.30274

The relative shifts were then computed by PyDrizzle by updating its understanding of the input WCS for each image by the ’delta’ shifts and computing a new overall shift. At no time does PyDrizzle update the headers of any input image based on these shifts, but instead, performs all the updates internally in its handling of the WCS information. The reference image does not end up with a shift of (0.,0.) in Table 2 as the images are centered on the combined output from all the images by default, not on center of the first input image. The relative shifts, if provided in a shift file, could equally have been expressed relative to the 01 image such that the 01 image would have a relative shift of (0.,0.). Either convention will be handled transparently to the user, and would not affect the computations by PyDrizzle.

This ASN table produces shifts computed by PyDrizzle, along with the offsets expected from the headers are shown in Table 2.

Table 2. Computed shifts using delta shifts compared to expected shifts.

Image rootname	XSH	YSH	Expected XSH	Expected YSH
f435w_01	40.3898115609	460.076363919	0.0	0.0
f435w_05	-879.127891188	-381.553062752	-880.	-880.
f435w_18	879.921338297	-459.194790072	880.	-880

The computed shifts are offset relative to the expected shifts because PyDrizzle centers the final output image and computes all shifts relative to that center position.

The final combination by PyDrizzle uses this ASN table to produce what appears to be a properly combined, distortion-corrected product. The coordinates of the star in the example are now:

```
f435w_drz    3531.22    1774.63   0:22:47.90  -72:03:11.2
```

in an output image of 5979 x 5152 pixels.

Summary

PyDrizzle Version 4.0 and the *buildasn* task have both been modified to support these options for the ASN table and shift file. These conventions should allow users to compute refined image shifts and use them in a simple manner with PyDrizzle to produce properly aligned, combined images. Future development work will rely on this mechanism to have automated routines deduce the shifts between sets of images and pass those results to PyDrizzle using these conventions. This work will form the basis for a flexible and simple way to combine images with proper registration using manual, and eventually, automated systems.

Acknowledgement

We would like to recognize the critical contributions made by several people who have made this capability possible. In particular, Anton Koekemoer provided the initial idea of tweaking the shifts computed by PyDrizzle and managed the overall effort to implement and test the Multidrizzle code. Andrew Fruchter provided considerable into how these shifts should be applied and managed by PyDrizzle. Max Mutchler rigorously tested these capabilities uncovering many issues which needed to be fixed in the implementation. Finally, all the Dither Working Group members provided invaluable experience and support for this effort. It is due to their contributions that registration of dithered observations can be managed so easily and robustly with the shift file documented here.