Drizzling dithered ACS images - a demonstration

Max Mutchler, Anton Koekemoer, & Warren Hack

Space Telescope Science Institute
3700 San Martin Drive, Baltimore, Maryland 21218
mutchler@stsci.edu, koekemoer@stsci.edu, hack@stsci.edu

Abstract. Since the 1997 version of this poster, dithering and drizzling have evolved from an advanced form of WFPC2 observing and data reduction, to the norm for ACS. We demonstrate the reduction of a typical dithered ACS dataset using the latest drizzling methods.

1. Introduction

The Drizzle task (Fruchter & Hook, 2002) is available in the IRAF/STSDAS dither package. PyDrizzle (Hack & Jedrzejewski, 2002) is a PyRAF wrapper for drizzle, which allowed drizzle to be incorporated into the ACS calibration pipeline. PyDrizzle uses the drizzle task to correct for geometric distortion, in order to produce an image which is photometrically and astrometrically correct across the image's entire field-of-view. Multidrizzle (Koekemoer, 2002) encapsulates the processes of building association tables, rejecting cosmic rays (even for singly-dithered observations), producing object catalogs, refining shift measurements, and producing a drizzled combination of the input images. We use a set of F814W (I-band) images of the "Tadpole" galaxy UGC 10214 (HST/ERO program 8992, PI Holland Ford) to illustrate the use of these tools. This example can be reproduced with the software and data available via the websites listed at the end of this document.

2. Pointing patterns and data associations

The shifts for small-scale dither or large-scale mosaic pointing patterns can be specified in a Phase II HST observing proposal using either POS TARG special requirements, or pattern parameter forms (Mutchler & Cox, 2001). When pattern forms are used, the entire pointing pattern is automatically associated (except for the largest WFC mosaic patterns), and the standard calibration pipeline is then able to process the dataset more completely. However, this demonstration illustrates how any set of data, which may be either partially or completely unassociated, can be associated post-facto, and reprocessed. Our sample dataset employs a POS TARG dither to shift across the gap between the two WFC chips. The ACS-WFC-DITHER-LINE pattern parameter form would produce the same results (see Appendix A).

3. Pipeline processing: CALACS and PyDrizzle

Association tables are used to process datasets which are related, such as cosmic-ray split (CR-SPLIT) exposures, dithered exposures, or data from different programs/epochs. Our sample dataset includes CR-SPLIT associations, but we need to produce one table which associates the entire pointing pattern. Download the following association tables (*asn.fits), and the combined/cleaned (*crj.fits) images that PyDrizzle will use as input:
Drizzling dithered ACS images (demo version)

ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54030_asn.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54031_crj.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54040_asn.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54041_crj.fits

Merge the two tables, and edit to create a unique (numbered) MEMTYPE for the CR-SPLIT pairs, and add a PROD-DTH row for the combined output, as follows:

```
c1> tmerge *asn.fits pipeline_asn.fits append
c1> tedit pipeline_asn.fits
```

<table>
<thead>
<tr>
<th>#</th>
<th>MEMNAME</th>
<th>MEMTYPE</th>
<th>MEMPRSNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>J8CW54P9Q</td>
<td>EXP-CR1</td>
<td>yes</td>
</tr>
<tr>
<td>2</td>
<td>J8CW54PDQ</td>
<td>EXP-CR1</td>
<td>yes</td>
</tr>
<tr>
<td>3</td>
<td>J8CW54031</td>
<td>PROD-CR1</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>J8CW54PPQ</td>
<td>EXP-CR2</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>J8CW54PTQ</td>
<td>EXP-CR2</td>
<td>yes</td>
</tr>
<tr>
<td>6</td>
<td>J8CW54041</td>
<td>PROD-CR2</td>
<td>yes</td>
</tr>
<tr>
<td>7</td>
<td>f814w</td>
<td>PROD-DTH</td>
<td>no</td>
</tr>
</tbody>
</table>

Alternately, association tables can be built using buildasn (Hack & Jedrzejewski, 2002). This association table can be used to re-run both CALACS and PyDrizzle. We will only run PyDrizzle here, but re-running CALACS may be necessary to recalibrate your data, and/or to produce the flat-fielded (*flt.fits) images needed to run multidrizzle (see Appendix B).

Start PyRAF and define your reference file directory, where the distortion correction table (IDCTAB) resides. Load the STSDAS dither package, run PyDrizzle (see parameters in Appendix C) using your new association table, and display the output image:

```
> pyraf
--> set jref = '/data/cdbs7/jref/'
--> stsdas
dither
--> pydrizzle pipeline_asn.fits bits=8578
--> display f814w_drz.fits[sci,1] z1=0 z2=5
```

4. Multidrizzle processing

To download the latest version of multidrizzle, contact Anton Koekemoer (koekemoe@stsci.edu). See detailed instructions on Anton’s webpage (listed below). As input, we will use existing flat-fielded images (*flt.fits) which were created by exposure lines with no CR-SPLIT special requirement, i.e. there is only one exposure at each dither pointing. Download the following images to your working directory. There are additional F814W exposures (see Appendix A), but we will use only two here:

ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54p8q_flt.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54plq_flt.fits

Move to your working directory, make an input image list, and set up multidrizzle. Specify tweakshift1=yes to refine the shifts:
Drizzling dithered ACS images (demo version)

> cd /data/mymachine1/demo/
> ls *flt.fits > input.list
> source /data/wallaby1/multidrizzle/setup
> pyraf
--> pyexecute('/data/wallaby1/multidrizzle/multidrizzle_iraf.py',
          tasknames='multidrizzle')
--> unlearn multidrizzle
--> multidrizzle output='f814w' filelist='input.list' tweakshift1=yes

Although we haven't specified singleCR=yes above, the tweakshift1 step automatically runs the SExtractor version of it. This is a relatively crude rejection method, which is used mainly to produce cleaner object catalogs for shift measurement (see Appendix E), and the resulting masks are not (by default) used in the final drizzle. Specifying singleCR_include=yes would include these masks in the final drizzle (see Appendix D). Since we are using only two input images here, this would be a way to reject cosmic rays in the gap overlap regions. While this may produce a better result cosmetically, a more reliable result would be achieved by including additional input frames.

Multidrizzle (using buildasn) creates the following association table, with the delta-shifts and delta-rotations determined by tweakshift1 stored in additional columns:

```
# Table f814w_twk1_asn.fits[1] Tue 20:02:49 05-Nov-2002
# row MEMNAME MEMTYPE MEMPRSNT XOFFSET YOFFSET ROTATION
#    arcsec arcsec degrees
1    j8cw54p8q EXP-DTH yes  0.  0.  0.  
2    j8cw54plq EXP-DTH yes -0.064946 0.013180 0.0094
3    f814w_twk1 PROD-DTH no  0.  0.  0.  
```

Display the drizzled output: the individual frames, and the final drizzled science (sci) and weight (wht) images:

--> display j8cw54p8q_flt_single_sci.fits[0] 1 z1=0 z2=5
--> display j8cw54plq_flt_single_sci.fits[0] 2 z1=0 z2=5
--> display f814w_sci.fits[0] 3 z1=0 z2=5
--> display f814w_wht.fits[0] 4 zr+ zs+

5. Further resources available via the web

The following web resources provide background on dithering and drizzling, and the sample data which can be used to reproduce this demonstration, if desired:

AC05 drizzling: www.stsci.edu/hst/acs/analysis/drizzle/
Andy Fruchter: www.stsci.edu/~fruchter/dither/dither.html
PyDrizzle: stdas.stsci.edu/pydrizzle/
Multidrizzle: www.stsci.edu/~koekemo/multidrizzle/
PyRAF: pyraf.stsci.edu/
SExtractor: terapix.iap.fr/soft/sexttractor/
AC05 ERO data: archive.stsci.edu/hst/acsero.html
AC05 ERO release: opposite.stsci.edu/pubinfo/pr/2002/11/pr-photos.html
Figure 1. The final multidrizzle output image of the Tadpole galaxy UGC 10214, using only two of the available F814W exposures as input.

A "draft version" of this document was available as a handout during the workshop. Due to page limitations, the supplemental Appendices in this "demo version" will not appear in the published workshop proceedings, but this version is available via the ACS drizzling webpage (listed above).

References

6. Appendix A: The pointing pattern

The dither pattern used to produce the Tadpole dataset was the POS TARG equivalent of the ACS-WFC-DITHER-LINE pattern. This pattern is defined in ACS Instrument Science Report 2001-07 (Mutchler & Cox, 2001), the HST Phase II Proposal Instructions, and is selectable within the HST Proposal Editor (PED).

<table>
<thead>
<tr>
<th>Pattern_Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pattern_Number: 1</td>
</tr>
<tr>
<td>Primary_Pattern:</td>
</tr>
<tr>
<td>Pattern_Type: ACS-WFC-DITHER-LINE</td>
</tr>
<tr>
<td>Pattern_Purpose: DITHER</td>
</tr>
<tr>
<td>Number_Of_Points: 2</td>
</tr>
<tr>
<td>Point_Spacing: 3.001</td>
</tr>
<tr>
<td>Coordinate_Frame: POS-TARG</td>
</tr>
<tr>
<td>Pattern_Orient: 85.28</td>
</tr>
<tr>
<td>Center_Pattern: NO</td>
</tr>
</tbody>
</table>

POS TARG equivalent: 0.248, 3.001
Three F814W exposures were obtained on April 1, and another three were obtained on April 9, 2002. We used the two exposures marked with an asterisk (*) for the multidrizzle example (but the others could also be included):

```
ni> hselect *flt.fits[0] $I,FILTER2,EXPTIME,RA*,DEC*,POSTARG* yes
j8cw04eq0_flt.fits  F814W  180  2.41553625E+02  5.54310944E+01  0.000  0.000
j8cw04eq1_flt.fits  F814W  610  2.41553625E+02  5.54310944E+01  0.248  3.001
j8cw04eq2_flt.fits* F814W  810  2.41537833E+02  5.54275944E+01  0.000  0.000
j8cw54p8q_flt.fits* F814W  180  2.41537833E+02  5.54275944E+01  0.000  0.000
j8cw54p8q_flt.fits F814W  610  2.41537833E+02  5.54275944E+01  0.248  3.001
j8cw54p8q_flt.fits* F814W  810  2.41537833E+02  5.54275944E+01  0.248  3.001
```

7. Appendix B: Running CALACS to produce flat-fielded images

You may need to generate the flat-fielded (*flt.fits) images, if they don’t already exist, which are used as input for multidrizzle. This is the association table we created (above) for running CALACS and PyDrizzle:

```
# Table pipeline_asn.fits[1] Tue 09:48:18 19-Nov-2002
# row MEMNAME MEMTYPE MEMPRSNT
# 1 J8CW54P9Q EXP-CR1 yes
2 J8CW54PDQ EXP-CR1 yes
3 J8CW54031 PROD-CR1 yes
4 J8CW54PPQ EXP-CR2 yes
5 J8CW54PTQ EXP-CR2 yes
6 J8CW54041 PROD-CR2 yes
7 f814w PROD-DTH no
```

Download the input raw images (*raw.fits) which correspond to the EXP-CR rows in the association table:

```
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54p9q_raw.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54pdq_raw.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54ppq_raw.fits
ftp://archive.stsci.edu/pub/ero/tadpole/j8cw54ptq_raw.fits
```

In addition to setting your reference file (jref) directory, the photometric calibration step (PHOTCORR) requires the following directories to also be defined (can include them in your .setenv file):

```
> setenv jref /data/cdbs7/jref/
> setenv mtab /data/cdbs1/mtab/
> setenv crotacomp /data/cdbs1/comp/ota/
> setenv cracscomp /data/cdbs1/comp/acs/
```

Change the following calibration switches in the image headers, and run CALACS:

```
cl> set jref = '/data/cdbs7/jref/'
cl> hedit *raw.fits[0] RPTCORR,CRCORR OMIT
cl> hedit *raw.fits[0] EXPSCORR PERFORM
cl> calacs pipeline_asn.fits quiet=no
```
Now, you could run multidrizzle to combine all the F814W images obtained by this observing program: with the flat-fielded (*flt.fits) images you have generated here, and the existing flat-fielded images listed in the previous Appendix.

8. Appendix C: PyDrizzle parameters

```bash
--> lpar pydrizzle

input = Input data: ASN table or single image
(output = ) Output drizzle image
(kernel = square) Shape of kernel function
(units = cps) Units for output image (counts or cps)
(pixfrac = 1.0) Linear size of drop in input pixels
(rotate = no) Rotate output drizzle product?
(orient = 0.0) Orientation angle of North in drizzled output
(psize = ) Linear size of output pixels (arcsec)
(ra = ) right ascension output frame center
(dec = ) declination output frame center
(xsize = ) Size of output frame X-axis (pixels)
(ysize = ) Size of output frame Y-axis (pixels)
(use_mask = yes) Use mask files created from input files?
(bits = 0) Integer mask bit values considered good
(wt_scl = exptime) Weighting factor for input data image
(fillval = 0.0) Value assigned to undefined output points
(idckey = ) Key for selecting IDC table (idctab/cubic/trauger/None)
(clean = yes) Remove temporary files?
(save = no) Keep individual drizzle output files?
(build = yes) Create multi-extension output file?
(version = 2.0 (14-Mar-2002)) Date of Installation
(mode = al)
```
9. Appendix D: Multidrizzle parameters

By default, `do_all=yes` is specified, in which case multidrizzle automatically performs all the steps except for single-image cosmic ray rejection (step 3) and shift refinement (steps 4a, 7a), which must be explicitly turned on by setting them to "yes". Keep copies of your original flat-fielded images (*.flt.fits) in another directory, and make new copies of them in your working directory each time you re-run multidrizzle, since the script does modify them (e.g. changes header keywords, updating the dq file, etc).

```bash
--> lpar multidrizzle
```

- `output=` Rootname for output drizzled products
- `(suffix = flt.fits)` Suffix of the input files
- `(filelist = )` HST Instrument used
- `(refimage = )` File for logging the script commands
- `(inst = ACS/WFC)` Do all the steps?
- `(do_all = yes)` Use new or old ACS distortion coefficients?
- `(restart = )` Create context image during final drizzle?
- `(coeffs = new)` Remove temporary files?
- `(context = no)` Create static bad-pixel mask from the data?
- `(static = no)` Name of (optional) input static bad-pixel mask
- `(staticfile = )` Value of good pixels in the input static mask
- `(static_goodval = 1.0)` Type of sky subtraction
- `(skysub = no)` Header keyword containing sky value
- `(skytipe = quadrants)` Interval width for sky statistics
- `(skytype = quadrants)` Sky correction statistics parameter
- `(skywidth = 50.0)` Lower limit of usable data for sky (in DN)
- `(skyname = SKYSUM)` Upper limit of usable data for sky (in DN)
- `(skywidth = 50.0)` Type of single-image CR rejection
- `(skytype = ringmedian)` Threshold value for single-image CR rejection
- `(singleCR = no)` Drizzle onto separate output images?
- `(singleCR_include = no)` Shape of kernel function
- `(singleCR_type = ringmedian)` Linear size of output pixels...
- `(singleCR_threshold = 10.0)` Linear size of drop in input pixels
- `(driz_separate = no)` Rotation to be applied (degrees anti-clockwise)
STEP 4A: REFINE THE SHIFTS
(tweakshift1 = no) Refine shifts after carrying out CR rejection?
(tweakstop1 = no) Stop after doing tweakshifts?
(tweaklimit1 = 0.1) Threshold shift for repeating CR rejection

STEP 5: CREATE MEDIAN IMAGE
(median = no) Create a median image?
(median_newmasks = yes) Create new masks when doing the median?
(combine_type = minmed) Type of combine operation
(combine_reject = minmax) Type of rejection
(combine_nsigma = 6 3) Significance for accepting min instead of med
(combine_nlow = 0) minmax: Number of low pixels to reject
(combine_nhigh = 1) minmax: Number of high pixels to reject
(combine_grow = 1.0) Radius (pixels) for neighbor rejection

STEP 6: BLOT BACK THE MEDIAN IMAGE
(blot = no) Blot the median back to the input frame?

STEP 7: REMOVE COSMIC RAYS WITH DERIV, DRIZ_CR
(driz_cr = no) Perform CR rejection with deriv and driz_cr?
(driz_cr_snr = 2.5 2.0, 3.0 2.5, 3.0 2.5) Driz_cr.SNR parameter
(driz_cr_scale = 1.2 0.7, 1.8 1.2) Driz_cr.scale parameter

STEP 7A: REFINE THE SHIFTS
(tweakshift2a = no) Refine the shifts based on the _cor files?
(tweakstop2a = no) Stop after doing tweakshifts?
(tweaklimit2a = 0.1) Threshold shift (in pixels) for repeating CR rejection
(tweakshift2b = no) Second iter: Refine shifts based on _cor files?
(tweakstop2b = no) Second iter: Stop after doing tweakshifts?
(tweaklimit2b = 0.1) Second iter: Threshold shift (pixels) for repeating CR rejection

STEP 8: COMBINED DRIZZLED IMAGE
(driz_combine = no) Perform final drizzle image combination?
(final_kernel = square) Shape of kernel function
(final_scale = 1.0) Linear size of output pixels
(final_pixfrac = 1.0) Linear size of drop in input pixels
(final_rot = 0.0) Rotation of input image (anti-clockwise)
10. Appendix E: Object catalog

The following is an excerpt (first and last 10 rows) of one of the object catalog files which was produced by \texttt{SExtractor} during the \texttt{tweakshift1} step:

\texttt{> more j8cw54plq_flt_single_cln.cat}

\begin{verbatim}
# 1 NUMBER  Running object number  [pixel]
# 2 X_IMAGE  Object position along x  [pixel]
# 3 Y_IMAGE  Object position along y  [pixel]
# 4 MAG_BEST  Best of MAG_AUTO and MAG_ISO [mag]
# 5 FLAGS  Extraction flags
# 6 A_IMAGE  Profile RMS along major axis  [pixel]
# 7 B_IMAGE  Profile RMS along minor axis  [pixel]
# 8 ELONGATION  A_IMAGE/B_IMAGE
# 9 FWHM_IMAGE  FWHM assuming a gaussian core  [pixel]
# 10 CLASS_STAR  S/G classifier output
# 11 IMFLAGS_ISO  FLAG-image flags OR’ed over the iso. profile
# 12 NIMAFLAGS_ISO  Number of flagged pixels entering IMFLAGS_ISO
# 13 X_WORLD  Barycenter position along world x axis  [deg]
# 14 Y_WORLD  Barycenter position along world y axis  [deg]

1 34.192 40.095 20.4371 003 3.16 2.46 1.286 1.43 0.12 000000
2 43.138 60.990 17.0514 018 13.41 12.30 1.090 22.96 0.00 000000
3 1694.976 27.430 21.7188 019 4.90 2.11 2.320 3.44 0.00 000000
4 1685.031 26.428 20.3739 018 7.29 3.43 2.127 0.00 000000
5 862.815 6.383 23.6987 000 0.76 0.45 1.702 3.44 0.00 000000
6 1676.247 25.530 23.0709 003 1.78 1.17 1.523 1.37 0.79 000000
7 450.426 25.763 25.1220 000 0.68 0.47 1.449 1.11 0.84 000000
8 682.409 30.171 25.2583 000 0.69 0.46 1.497 2.83 0.87 000000
9 1486.703 45.678 20.9066 000 1.49 1.32 1.133 3.72 0.07 000000
10 45.230 51.089 20.9248 000 1.35 1.19 1.128 1.92 0.02 000000

....

1567 3011.509 3840.138 22.8621 000 0.91 0.68 1.343 3.70 0.00 000000
1568 3814.369 4159.231 21.9100 000 2.00 1.24 1.623 7.45 0.07 000000
1569 2263.454 3905.367 21.7797 002 1.57 0.98 1.595 7.41 0.03 000000
1570 1392.789 3861.166 25.1730 002 0.71 0.56 1.282 1.60 0.49 000000
1571 3433.512 3857.490 22.5082 002 1.90 1.47 1.296 8.46 0.00 000000
1572 4084.333 4101.871 20.4312 002 10.40 1.06 9.841 7.02 0.03 000000
1573 3009.756 3829.495 22.6403 000 1.39 0.70 1.989 5.86 0.00 000000
1574 680.254 3936.003 25.0393 000 0.71 0.63 1.127 1.68 0.68 000000
1575 3059.231 3863.803 25.0711 000 0.71 0.55 1.287 1.79 0.93 000000
1576 3800.190 3902.589 22.3915 000 2.05 1.13 1.805 5.30 0.03 000000
\end{verbatim}