Montage Baseline Background Correction

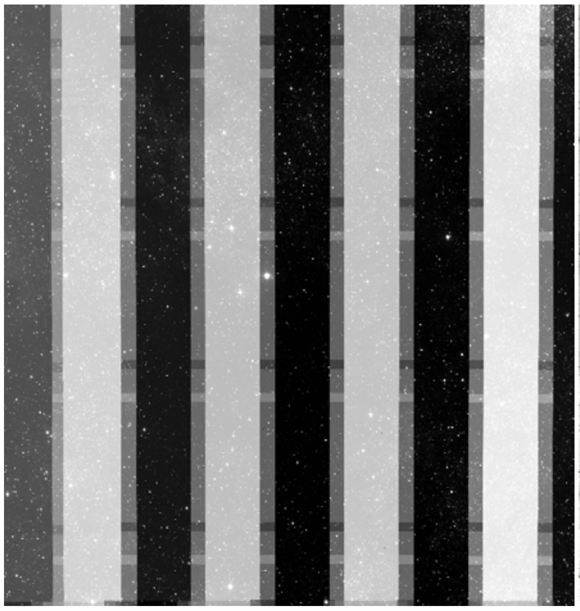
2MASS Background Correction

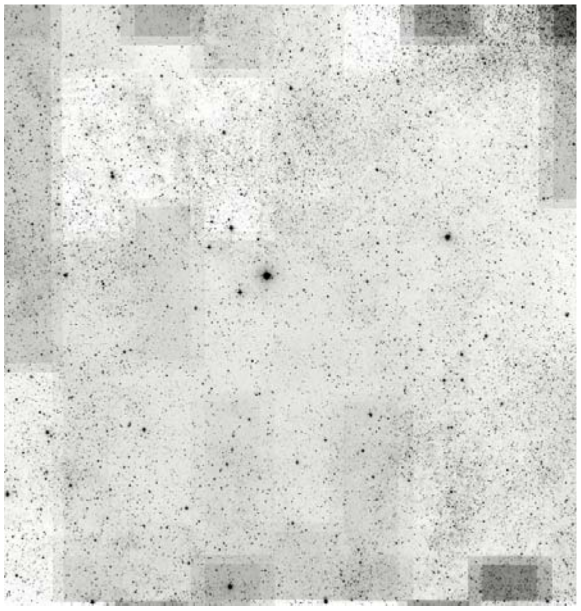
The 2MASS images have a great deal of image-to-image background variation. This is a result of difference between and variability in the airmass through which the observations were take.

Our baseline background correction algorithm attempts to compensate for this by performing a lower-envelope least squares plane fit to the individual images, hoping that by bringing all image background to zero, it will be possible to mosaic them together with a minimum of seams.

The first image below shows the data coadded with no background removed. The second shows the results of this baseline correction algorithm. The baselining for some of the smaller image fragments around the periphery is less stabe (given the smaller amount of data to work with).

The algorithm does remove the gross background differences, though there are clearly finer adjustments that require analysis of the overlap regions.


Raw coadd


Coadd after baseline background correction

2MASS Baseline Software

2MASS baseline background removal was implemented using three discrete, standalone programs (in ANSI C). These have been tested on a Sun Ultra 5 workstation (Solaris 2.8). The modules are:

`mImgtbl`	Extracts the FITS header geometry information from a set of files and creates an ASCII image metadata table from it used by several of the other programs.
`mFitplane`	Fit a plane (excluding outlier pixels) to an image. Output is the plane parameters (used by mBackground)
`mBackground`	Remove a background from a single image (planar has proven to be adequate for the images we have dealt with).
`mFlattenExec`	Run `mFitplane` and then `mBackground` on all the images in the metadata table

All these programs are run entirely from the command line and all return a structured response for which we have parsing code. Given no arguments, each program responds with a short usage summary. Each also has built-in debugging that can be turned on, though as usual with debugging output, it is mainly useful to the originally develper.

2MASS Background Correction Procedure

The general procedure for using these program should be obvious from the above descriptions. The example packaged with the code has a subdirectory for the code, one for the images to be background corrected and one for the corrected images:

```
                  |- 980702s-j0910009.fits
                  |- 980702s-j0910009_area.fits
                  |        .
|- uncorrected -|        .
|                 |        .
|                 |- 980814s-j0500009.fits
|                 |- 980814s-j0500009_area.fits
|
|
|               |-
|- corrected -|                    (initially empty)
|               |-
```

```
          |
  2massgb -|
          |            |- mFitplane.c
          |- Montage -|- mBackground.c      (background removal
programs)
          |            |- mFlattenExec.c
          |
          |
          |- Imgtbl -|- mImgtbl.c           (image metadata summary
program)
          |
          |
          |        |- include
          |        |
          |        |        |- mtbl
          |- lib -|        |- svc            (source code for various
                   |- src -|- coord           off-the-shelf libraries)
                            |- cfitsio-2.300
                            |- wcstools-3.0.5
```

The software can be built from the top of the directory tree using the rudimentary Makefile found there (there are more complete Makefiles in each of the source directories).

The sequence of commands used for the timing test is as follows:

```
               make
                cd uncorrected
                ../bin/mImgtbl . images.tbl
                ../bin/mFlattenExec images.tbl ../corrected -d
```

The `-d` option on the last command turns on debugging, which includes the times (wall clock) each program takes.

2MASS Timing Results

Below are the times for a few of the images in the test set when run on a Sun Ultra 5 workstation (Solaris 2.8). Some of the images in the test set are fairly small (the ones around the periphery in the image above) and run faster but these are typical of the time to correct full images:

```
   [mFitplane ./2mass-atlas-980702s-j0910044.fits]
   [mBackground ./2mass-atlas-980702s-j0910044.fits
    ../corrected/./2mass-atlas-980702s-j0910044.fits
    6.37174e-05  9.85556e-03  1.14975e+02   1331.22    855.05]
```

```
Time: 11 seconds

-------------------------------
[mFitplane ./2mass-atlas-980702s-j0920233.fits]
[mBackground ./2mass-atlas-980702s-j0920233.fits
 ../corrected/./2mass-atlas-980702s-j0920233.fits
-1.33085e-03 -3.16278e-03  3.13931e+02   1030.57    709.37]

Time: 14 seconds

-------------------------------
[mFitplane ./2mass-atlas-980702s-j0920245.fits]
[mBackground ./2mass-atlas-980702s-j0920245.fits
 ../corrected/./2mass-atlas-980702s-j0920245.fits
-2.99051e-03 -6.01130e-03  3.15664e+02   1028.48    -37.81]

Time: 13 seconds

-------------------------------
[mFitplane ./2mass-atlas-980702s-j0920257.fits]
[mBackground ./2mass-atlas-980702s-j0920257.fits
 ../corrected/./2mass-atlas-980702s-j0920257.fits
-1.80820e-03 -3.03723e-05  3.18274e+02   1027.69   -873.17]

Time: 13 seconds

-------------------------------
[mFitplane ./2mass-atlas-980702s-j0930021.fits]
[mBackground ./2mass-atlas-980702s-j0930021.fits
 ../corrected/./2mass-atlas-980702s-j0930021.fits
 2.12610e-04  6.97836e-04  1.13398e+02    678.59   -668.63]

Time: 14 seconds

-------------------------------
[mFitplane ./2mass-atlas-980702s-j0930044.fits]
[mBackground ./2mass-atlas-980702s-j0930044.fits
 ../corrected/./2mass-atlas-980702s-j0930044.fits
-1.13888e-02  5.72135e-03  1.17063e+02    676.03    869.91]

Time: 13 seconds

-------------------------------
```

The average time per image is approximately 13 seconds. Since there are 5834400 images in the 2MASS archive, this would translate into approximately 7.8E7 CPU seconds or 2.4 CPU years total processing.

DPOSS Background Correction

The DPOSS background correction used to date uses a similar approach though the details differ a fair amount.

DPOSS images are quite large (23500 x 23500 pixels / 2 bytes per pixel: total 1.1 Gbyte) and cover more than 6 degrees on the sky. There are a total of 879 DPOSS plate locations, in 3 filters, or 2637 images total.

The background can have appreciable curvature and the data for the individual images have been fit here by a second order polynomial. No lower envelope processing was used; the result therefore requires no iteration but will be affected by bright regions in the image (*i.e.,* non-background pixels are used in the fitting).

The quadratic form used was:

```
    I = I0 + A*x + B*y + C*x*x + D*x*y + E*y*y.
```

where I is the image pixel intensity and (x,y) are the pixel coordinates (I0, A, B, C, D, and E are the free parameters to be fit). For the test image, these coefficients are printed below:

```
   I0 = 4752
   A  =    6
   B  =  162
   C  =  -99
   D  =  -39
   E  = -169
```

DPOSS Background Correction Procedure

The code for performing the above procedure can be obtained by contacting Montage. The steps in building and running are:

- Untar the file
- `cd flatten`
- look at the Makefile for the C compiler, it is currently set to "`gcc -O`".
- `make`
- look at the script "`flatten`". It needs a directory called "`plates`" for input and another "`platesflat`" for output. Also change "`file`" to be the one you want to flatten.
- `flatten`

Given the size of the DPOSS images, no data is included in the tar file. Please contact the Montage project to arrange for example data.

DPOSS Timing Results

This backgrounding code was run on a single processor of an HP Superdome -- the processor is PA8600 at 550 MHz. Processing took 660 seconds for each plate, or about 20 days for the whole 3 Tbyte DPOSS.