Introduction

CCDStack is a tool for processing raw CCD data to produce high quality analysis data and presentation images. CCDStack is designed to yield optimal results and makes use of several advanced concepts and features. The user interface facilitates intuitive processing with advanced visual aids for image registration, data rejection, and image presentation.

If you are new to CCDStack then see first use

Check for recent updates and upgrades:

http://www.ccdware.com/downloads/updates.cfm

First use

Nearly all processes are accessable via the main menu. All menu options are documented in this Help file.

Load your first image via file menu

Several thing will happen the first time you load an image:

- camera manager will detect a new camera and ask you to verify or change the settings. If you want to overide the default values then make any changes desired, otherwise just press the OK button.
- several sub-windows/forms will appear, jumbled up in the upper left corner. Drag each form to a place on the screen that you like. CCDStack will remember where you put it (you can always move it again later). Each of these forms may be removed (via the usual Windows method of clicking the symbol on the right top border) and reinstated via the windows menu.

To process your first stack

Using a group of images of the same target:

1. load images
2. calibrate images
3. save calibrated images
4. register the stack
5. normalize the stack
6. data reject (use Poisson sigma-reject)
7. combine via mean
8. save FITS
9. set DDP and sharpen
10. save JPEG

Read Concepts from beginning to end

Concepts

The concepts section can be read from top to bottom starting here.

first concept

Image

Image data

The image data data is a 32-bit floating-point array that is the underlying base data for the image. Most of the data procedures operate on the image data but most of the image display functions do not affect the image data (e.g. adjust display).

Display bitmap

The display bitmap is the immediate source of the image seen on your screen, though it is not the screen image itself. The display bitmap consists of 8-bits/color and is derived from the image data primarily via adjust display. Note that adjust display has no effect on the underlying image data (e.g. DDP or sharpen does not modify the underlying 32-bit image data).

The magnification control resizes the display bitmap to create the screen image. Show pixels magnifies the display bitmap by integer pixels (i.e. magnifies the actual pixels), otherwise the display bitmap is resized via interpolated zoom view (bi-cubic resampling).

If magnification is 0.5x or less then the image data is automatically binned to create the display bitmap. A binned display bitmap is smaller than the image data and Save display bitmap saves that binned display bitmap. (E.G. magnification 0.33x produces a bitmap that is 1/9 the area of the original image).

Display bitmap values (shades of gray or color) are controlled by adjust display, via scaling algorithms: linear, gamma, DDP, and gamma-DDP. Several of the adjust display controls have built-in slider acceleration, which should become obvious with use. To initially scale an image, first select DDP or linear/gamma via DP box then press the auto scale button. Next, adjust the sliders to fine-tune the display. Optionally, sharpen the display bitmap by checking the sharpen box.

Source files

CCDStack accepts image-file formats FITS, SBIG, TIFF, JPEG, and BMP. The program will read these formats without regard to the actual file name (e.g. an SBIG or FITS file named “IMG.001” is OK). Depending on format, CCDStack reads 8, 16 and 32 bit monochrome files to create 32-bit floating-point image data and 24, 48, and 96 bit color files to create three 32-bit floating-point data structures.

CCDStack accepts color FITS and "one shot color" (Bayer) FITS and SBIG formats.

Output files

CCDStack writes two different types of image files: image data and display bitmap. Save data creates a 16 or 32 bits/color file from the 32 bits/color image data. Save display bitmap creates an 8 bits/color file from the current display bitmap. Save screen as JPEG saves the current screen as a “screen capture”.

Color

CCDStack handles color images using a psuedo-Lab scheme. The image data consists of a Luminance (L) 32-bit floating-point 2-dimensional array and two color-ratio 32-bit floating-point arrays. These psuedo-Lab values are converted to RGB for the display bitmap, output files, and color controls. "One shot color" (Bayer pattern) are calibrated in Bayer format then should be converted to R,G,B for normal processing.

Image information

Information about the image can be obtained via: file header information, image manager, and selection information.

next concept

previous concept

Image stack

An image stack is a group of individual exposures framing the same object or field. Image stacks are typically used to accumulate long total exposure times that would otherwise be impractical or impossible in a single exposure. Such individual exposures are registered/aligned, normalized, data reject and combined to create a composite image.

Image stacks can be processed to remove undesirable artifacts (such as cosmic rays or airplane trails) from the composite image. CCDStack implements several data rejection procedures to detect artifacts, pixel by pixel, and reject those pixels in the image combine

Create a stack

To create a stack, open more than one image. The status bar (bottom left of window) displays the number of images in the stack. Various menu items, procedures and options are activated according to the number of images (e.g. median combine requires at least 3 images).

Navigate thru the stack

To view or process a particular image, use Select image or Image Manager.

The image displayed is “this” or the “top image”.

The status bar (bottom left of window) displays pixel values for the top image.

This image, all images, included images

Many procedures can be applied to this or all images.

This image is the image currently on the screen (on “top”).

All images refers to all included images unless otherwise indicated. Use Image Manager to view or change the include status of an image (color images cannot be included).

The concept of included images is used to restrict operations to a subset of the stack. For example, a stack of exposures will automatically be included and all new images made from combining those images will be not included (default). This prevents the combined images from being included in any subsequent combine. Also, certain exposures can be temporarily not included for some combines (e.g. to test the effects of omitting sub-par images).

Image size

CCDStack requires that all images within the stack be the same size. If differently sized source images are loaded then all images in the stack are adjusted to the largest size. If images are different by an integer magnification factor then the smaller image is assumed to be binned and is enlarged (unbinned) to match the larger image (useful for LRGB). If images are differently sized by non-integer magnification factor then the smaller images are centered within an enlarged canvas without magnification change.

Large stacks

CCDStack uses several large data structures for each image and this imposes limits on the number of images that can reside in computer memory. When the number of images exceeds a threshold (depending on image size) then CCDStack starts caching the images, placing many of the structures in temporary files on the hard drive. Cache status is shown in the status bar (bottom left of window). Caching may noticeably slow performance.

64 bit operating systems (Vista 64, XP-64) have no need for image caching and they impose no practical limit on stack size (other than the speed and RAM of this machine and your patience).

Cache control and memory warnings are accessible via Settings.

Memory problems can be minimized by the following practices:

- Load all images at once; do not load individually or in small batches (the program tries to predict the memory needed by how many images are opened at once).
- After processing a large stack, it is advisable to quit the program and restart it before processing another stack.
- If the set of images to be processes is very large then consider breaking it up into smaller batches for separate processing then combine the results of each batch in a final step.
- Use Windows Task Manager ([ctrl][alt][del]) to monitor VM use by CCDStack. If VM exceeds 1,000,000k then the program will soon become unstable so you should save your work, quit and restart.

next concept

previous concept

Camera Manager

The camera manager is used to store and reference three important qualities that are used for some of the procedures. In particular, Poisson statistics require these qualities to be accurate (e.g. Poisson reject).

Most image acquisition software place a camera name in the FITS header. If the FITS header has no camera designation then the "default" camera is used.

The camera manager is automatically shown whenever a new camera is detected and can be accessed via “edit” menu.

next concept

previous concept

Calibration

Raw CCD images are calibrated to remove systematic artifacts from the camera and lens. Calibration consists of dark subtraction and flat fielding. Dark subtraction removes the pixel-to-pixel variation due to dark-current accumulation. Flat fielding compensates for variations in illumination across the CCD and pixel-to-pixel variations in sensitivity. Bias frames can facilitate the application of both darks and flats.

Although calibration is usually necessary, it enviably imparts some noise into the image and it is desirable to minimize the increased noise by using high quality master dark, master flat and master bias. It is also desirable to understand the calibration process in order to produce the highest quality images.

Dark Frame and Dark Subtraction

“Dark current” is the name for electrons that accumulate in the CCD pixels due to thermal effects. A “dark frame” is made from one or more “dark exposures”. A dark exposure is a CCD image taken with the camera shutter closed, so that the CCD receives no light.

Every pixel in a CCD exposure starts out at its bias level and accumulates dark electrons over time at a fairly uniform rate. However, the rate varies from pixel-to-pixel so pixels increasingly diverge from one another over time, which results in a noisy “salt and pepper” look. Dark subtraction reduces this apparent noise by removing the dark current from each pixel in the image. Dark subtraction should be the first calibration performed, prior to flat fielding, normalization, registration, etc.

Each pixel in a dark exposure represents an instance of the CCD’s inherent dark current rate. The instance only approximates the “true” rate because of uncertainty due to Poisson noise. Additionally each dark exposure contains readout noise, which for most pixels is considerably greater than the Poisson noise. These noises increase the uncertainty of the calibrated image, so it is desirable to minimize the dark frame’s noise. This is done by taking and combining several exposures. Averaging N same-exposure-time dark exposures decreases the dark frame noise by sqrt(N).

In addition to unwanted noise, dark exposures often contain undesirable outlier artifacts, particularly cosmic ray hits. Outliers can be removed from the dark frame by taking several exposures and applying a data rejection method when combining. Use the calibration procedure in CCDStack to create high quality master darks.

The dark electron accumulation rate depends on the temperature of the CCD, which makes it desirable to operate the CCD at the coldest temperature practical. It is important to match the temperature of the dark frame with the image. Although it is possible to calibrate via a different-temperature dark using nonlinear adaptive subtraction.

If the dark frame matches the image’s temperature and exposure-time then simply subtract the dark frame from the image. If the dark frame matches the image’s temperature but does not match exposure-time then calibrate via linear adaptive dark subtraction. If temperatures do not match then calibrate via nonlinear adaptive subtraction.

Dark frames are generally stable over long time periods, depending on the camera. Thus it is useful to maintain a dark library containing high quality master darks for a few exposure times at each operating temperature that you use.

CCDStack does not restore the pedestal after a dark subtract: calibrated image pedestal = 0.

Simple Dark Subtraction

If the light exposure and dark frame are same-temperature and same-exposure-time then the dark subtraction for each pixel is:

Image_pixel_out = Image_pixel_in – Dark_pixel;

Adaptive Dark Subtraction

If the light exposure and dark frame are taken with different exposure times then it is necessary to scale the dark frame to account for the differences. It is necessary to subtract the bias from the dark frame in order to correctly scale the dark frame. The scaled dark subtraction for each pixel is:

Image_pixel_out = Image_pixel_in – (Dark_pixel – Bias_pixel)*scaling_factor;

RMS Dark Subtraction

CCDStack uses an iterative best-fit algorithm to derive a scaling factor that produces minimal pixel-to-pixel differentials (RMS) in the subtracted image. The derived scaling factor is displayed in the information form.

Time Scaled Dark Subtraction

The Adaptive scaling factor = Image_exp_time / Dark_exp_time

Creating a master dark

Take as many dark frames as possible, using the same exposure time and temperature for each frame. Use create master dark

to combine the dark frames.

Set the camera at the desired, sustainable temperature. It is advisable to cap the camera and keep it in an area with low ambient light to avoid light leakage onto the CCD. Also, dark exposures should not be taken too soon after a light exposure because CCDs can retain dim image “ghosts” that fade with time (electrons bleeding from the substrate). A refrigerator is a good place for taking darks in the daytime.

Dark Library

Because dark frames are long lived and because they require time and effort to produce, it is useful to save and reuse them. Master darks should be stored in a “library”. CCDStack facilitates use of dark and bias libraries by controlling the “file open” and “save” dialogs associated with dark frames and by suggesting descriptive file names for the master frames.

The dark library should contain a master dark for each CCD, binning state, and operating temperature. It is not necessary to have master darks for every exposure-time that you use because adaptive subtract compensates for different exposure times. But if you commonly use certain exposure-times then make master darks for those exposure-times.

Flat Field

Flat fielding compensates for uneven optical illumination and pixel-to-pixel QE variations. A “flat frame” is an exposure of an evenly illuminated object, such as the twilight sky at zenith or an evenly illuminated screen. A series of exposures are combined to make a master flat Each exposure should have an average ADU of about 60% full well capacity (commonly about 30,000 ADU).

Dark/Bias subtraction

Typically, flat exposures are very short and it is unnecessary to subtract a matched dark frame. A bias frame or the pedestal or average bias will usually work very well. CCDStack calibration facilitates 2 different methods for dark-subtracting flats:

Perform dark|bias subtract during the creation of the master flat
or
Specify bias/pedestal for a non-dark-subtracted flat in calibration form

Both methods work fine. It is suggested that you choose one method and stay with it to avoid confusion.

Flat field – auto bin/unbin

During calibration, the flat is automatically binned or unbinned to match the image frame.

It is recommend that flats be taken without camera binning, even for binned images because an unbinned flat usually contains intrinsically higher S/N due to fixed camera gain and bit-depth. E.G. for a fixed-gain camera, a 2x2 camera-binned flat with average 50,000 ADU/pixel has the same total S/N as an unbinned flat with average 12,500 ADU/pixel; likewise an unbinned flat with average 40,000 ADU has the same total S/N of a 2x2 binned flat with average 160,000 ADU (which exceeds 16-bit number-space).

Master flats

Although a single flat exposure will usually produce acceptable results, a master flat made from multiple flat exposures will produce superior results. It is usually easy to take multiple flat exposures and it is also easy to create a master flat.

Flat library

Due to small changes in camera or optical conditions, flat frames are usually specific to a limited session of images. A common practice is to take fresh flats for every imaging session, in which case it is useful to keep images and their associated master flats in the same folder/directory. For very stable systems it may be permissible to use static master flat frames, in which case it is useful to keep master flats in a special flat library.

To facilitate a flat library, check "use static flat lib" in settings

To facilitate flats and images in the same-directory, uncheck "use static flat lib"

Bias Frame

A bias frame is a very short dark exposure (too short to accumulate significant dark current), which defines the base-line ADU for each pixel. A bias frame is used in adaptive dark subtraction. Bias frames are also commonly used to dark-subtract flat frames because flat exposure times are typically very short.

Master bias

It is advisable to use a master bias in order to minimize the effects of readout noise. For many cameras, it is not necessary to maintain temperature-specific master biases. Thus it may be acceptable to create and use a single master bias. If the average pixel value changes more than 10-20 ADU for bias frames of different temperatures then it may be useful to maintain temperature-specific master bias frames.

CCDStack facilitates a bias library, which can be the same folder/directory as the dark library.

Pedestal / average bias

Raw CCD images from the camera usually have a constant added to each pixel ADU. Also, some CCD software packages add a constant to dark-subtracted images. This constant is called the pedestal (or bias). Typically, a pedestal is used to avoid negative ADU, which cannot be represented by a 16-bit unsigned integer (a common CCD format).

CCDStack does not add a pedestal to the image data. Nor is a pedestal added to output files, except for 16-bit color FITS and TIFF, which are given a 100 ADU pedestal. If necessary, use pixel math to add a constant to the image data.

Because a bias frame has no significant light signal or dark current, the average pixel ADU of a bias frame will approximate the pedestal and the two can be used interchangeably. To measure the average bias – open a bias frame and use the mouse to drag a large rectangle within the image. The mean ADU of that rectangle will be shown in the information window

CCDStack needs to know the pedestal or average bias or bias frame for flat fielding and adaptive dark subtraction. Also, images calibrated outside of CCDStack may contain a pedestal that affects procedures such as sigma-reject. Use camera manager to inform CCDStack of any pedestals.

next concept

previous concept

Registration / Alignment

Usually, the first step for combining exposures is to register (align) the stack. If the exposures are pre-aligned (i.e. the scope or camera were not moved between exposures) then skip to normalization.

It is beneficial to deliberately offset (dither) individual exposures by a few pixels. This practice avoids arithmetically additive dark-frame noise and improves flat-field quality. But it is necessary to register the offset exposures.

Also, it may be necessary to register images affected by field rotation caused by mount polar alignment error.

Base (Reference) Image

The top image at the start of registration becomes the base image, which serves as the reference to which all other images are registered. The base image is not transformed or resampled. It is a good practice to select the sharpest image as the base image.

Registration steps

Registration consists of two distinct operations:

1. determining the alignment transform.
2. apply the transform by resampling the images.

Alignment Methods

The following methods are available to determine the alignment transforms:

- star snap - uses star centroids to calculate the transform(s)
- FFT – automatic registration. Does not rely on stars.
- manual - full manual control over shift, rotation, and magnification

The transform results are shown in a screen overlay, which allows visual verification prior to resampling. Blink the stack to check all alignments.

When the transforms look satisfactory they may be applied via resampling.

next concept

previous concept

Resampling / Interpolation methods

Image transforms (shift, rotate, magnify/resize) require that the original images be resampled using one of the following methods for interpolating pixels in the transformed images.

The optimal method depends on several things, including:

- Stack vs. Single image
- Sampling
- Number of images in stack
- Data rejection considerations
- Type of image(s) – e.g. stars vs. planets

The method descriptions below give some guidelines.

"Which one should I use?"

A good default method for registering stacks is Nearest Neighbor.

A good general default method for transforming single images is Bicubic B-Spline.

Nearest Neighbor

Pixels from the source image are transformed to the nearest point in the resampled image. Nearest Neighbor produces sub-pixel errors between 0-½ pixels, which can result in small jagged or jumpy artifacts in the resampled image, though they usually disappear in a combined image. These artifacts are generally more pronounced in rotated or magnified alignments.

Nearest Neighbor is primarily used for aligning a stack to produce a combined image. It is usually not appropriate for resampling a single image. Also, Nearest Neighbor should generally not be used on under-sampled images or small stacks, but it often works very well for stacks of well-sampled images (blinking the stack may look weird but the combined image may be very good).

For non-magnified transforms there is a one-to-one pixel correspondence and thus noise statistics are preserved in the resampled image. And because Nearest Neighbor does not combine source-image pixels there is no smearing of hot or cold pixels in the resampled image, which has definite advantages in data rejection.

Bi-linear

Pixels in the resampled image are imputed from a small matrix of pixels in the source image. Bi-linear interpolation produces sub-pixel accuracy and works very well on non-magnifying and non-rotating transforms.

Bicubic B-Spline

Bicubic B-Spline works well for magnification and/or rotation transforms. This method produces the smoothest result with the fewest artifacts (no ringing) but it also imparts a slight blur to undersampled images.

Quadratic B-Spline

A B-Spline variant. The B-Splines are related but use different weights and have subtle differences in results. Quadratic B-Spine produces sharper results than Bicubic but it has an increase in potential artifacts.

Bicubic Spline 16

Pixels in the resampled image are imputed from a matrix of 16 pixels from the source image. Useful for undersampled images. Bicubic Spline 16 produces a sharper result than Bicubic B-Spline at the expense of some small artifacts.

Bi-cubic Spline 32

Pixels in the resampled image are imputed from a matrix of 32 pixels from the source image. This produces similar results as Spline 16 with slightly higher fidelity, but it takes longer to process.

Gauss

Applies a Gaussian weighted function.

Mitchell

An efficient algorithm most suitable for enlarging.

Cosine

Similar to, but much simpler and faster than sinc. This method preserves resolution without producing too many artifacts.

Lanczos/sinc 64

The "sine cardinal" (sinc) resampling function using a matrix of 64 input pixels per output pixel. This function preserves fine detail but produces some ringing (small artifacts). Because this algorithm processes many pixels it runs slowly.

Lanczos/sinc 256

The "sine cardinal" (sinc) resampling function using a matrix of 256 input pixels per output pixel. This function preserves the finest detail but produces some ringing (small artifacts). Because this algorithm processes many pixels it runs very slowly.

Preserve flux

Preserve flux adjusts the reampled pixel's ADU so that the total ADU of the image remains the same. The average pixel ADU will increase with magnification <1 or decrease with magnification > 1. Otherwise the average resampled pixel ADU remains similar and the total ADU increases with magnification <1 or decrease with magnification > 1.

Not preserving flux affects the meaning of camera gain.

Flux is automatically preserved for Registration/Alignment resampling.

Performance Note

Rotation greater than 45deg runs slower than rotations of 45deg or less. So if you are registering images taken on different sides of the meridian with GEM (“flipped”) then you should pre-rotate some of them 180 deg (via “Edit”; “Transform”; “Rotate”; “180 deg”) before registration. You don’t have to do that (you can rotate them within the registration procedure) but pre-rotating them speeds-up the registration “apply”.

next concept

previous concept

Normalization

Normalizing the stack results in all images having similar ADU values for corresponding pixels (and area and features). Normalization mathematically compensates for variations in sky background, sky transparency, exposure times and so on. Such compensation is often necessary to produce optimal data rejection and image combines.

CCDStack performs 3 types of normalizations:

- scalar (slope) - each pixel is multiplied by a constant.The constant is automatically computed for each image based on user selected area. Scalar normalization adjusts for exposure time and sky transparency, but not for varying levels of sky brightness. Use scalar to normalize flats when making a master flat via data-rejection.
- offset (intercept) - a constant is added to each pixel. The constant is automatically computed for each image based on user selected area. Offset adjusts for varying sky background levels.
- both (slope and intercept) - each pixel is transformed by both offset and scalar constants. The constants are automatically computed for each image with an option to specify selected areas.

Reference image

One image is the reference image to which all others are normalized. The current top image at the start of normalization is used as the reference image (to select a different image for the reference, cancel normalization, navigate to the desired image and restart normalization).

Auto normalize

Auto normalize performs a slope and intercept normalization via histogram fitting of a center portion of the image or selected image area.

Control normalization

Control normalization performs any of the 3 types of normalization based on user selected areas.

Verify normalization

The scalar and offset values calculated for each image are displayed in the information window. To verify normalization – check "apply to all" in adjust display and blink the stack.

If the Auto normalization produces poor results then repeat it, selecting an area containing a broad ADU range with minimal saturation or use Control normalization.

Repeated normalizations are OK.

The stack should be registered prior to normalization to get accurate results.

Weights

Weights are calculated from normalization as the inverse of the scalar factor. The combine procedures use these weights to optimize the resulting S/N.

Changes in gain (due to different cameras in the same stack or variable gain due to binning) are factored into the weights so that all of the images are normalized on the electron level. The gain of the reference image (top image) is used to normalize the other gains (if different) and this affects the weights.

For example, after normalization the average ADU of a 5-minute exposure will approximate the average ADU of a 10-minute exposure (that’s what normalization does). If these images are summed without weights then the 10-minute exposure will contribute the same S/N as the 5-minute exposure, thus resulting in sub-optimal S/N. But a weighted sum preserves this difference to produce optimal S/N.

The weights are shown in image manager. It is possible to override a weight by entering a new weight in the image manager form. One possible reason to modify a weight might be to decrease the impact of images with poorer resolution.

next concept

previous concept

Data rejection

Data rejection is used to remove or attenuate undesirable artifacts in the image(s). CCDStack implements a variety of data rejection procedures to detect undesirable pixels and mark them as rejected. Rejected pixels are graphically displayed for each image so that the effects of data rejection procedures are obvious.

Rejected pixels are replaced by imputed values during the image combine or by the impute rejected pixels process.

Rejected pixels and Missing data

Regected pixels are generated by the data rejection procedures below. These pixels are marked as rejected in a special data structure so the actual pixels in the image data remains unchanged. Thus the rejected pixel's original value is not lost, even though that value is not used in an image combine.

MissingValue is assigned to pixels that have no meaningful value, such as pixels in the offset border region of a registered image. Rejected pixels can be converted to MissingValue via set rejects to MissingValue.

Assessing the effects of rejections

CCDStack graphically shows the effects that any rejection procedure has on any image. Rejected pixels are tinted red. The

Single image data rejections

Single image rejections only use data from the image itself:

- Clear
- Reject hot/cold pixels
- Reject blooms
- Reject range
- Clear range
- Grow
- Freehand draw
- Impute rejected pixels
- Set rejects to MissingValue

Single image rejections can be simultaneously applied to all included images.

Pixel stack data rejection procedures

Pixel stack data rejections operate on pixel stacks, which are a collection of pixels from a single location in all images in the stack. These procedures determine rejection based on the statistics of the pixel stack only (i.e. near by pixels within an image have no influence). There are basically 2 different types of stack rejections: Sigma-reject and clip.

The importance of prior normalization

For all stack based rejections is imperative that the stack first be properly normalized. Otherwise some images may be over-rejected and others may be under-rejected. If rejection rates are markedly different for no obvious reason (e.g. different exp-times) then it may be beneficial to re-normalize the stack then repeat the rejection procedure. Normalization automatically clears rejected pixels.

Sigma-reject

- STD sigma-reject
- Poisson sigma-reject
- Linear factor reject

These procedures calculate the average value for each pixel stack and compute a "sigma" error-term based on the particular method (STD, Poisson or linear) and the user specified factor. If any of the pixels in the stack deviate from the average by more than the error-term then the pixel with the greatest deviation is rejected and removed from the pixel stack. The process is then repeated (calculating a new average and error-term) until all pixels are within the specified error. Each process is an iteration and the user can specify a limtied number of iterations.

For equal-time exposures, sigma-rejections should usually produce approximately equal rejection percentages for all images, except for the registration reference image, which will have more rejections unless the images were registered via nearest neighbor (due to the smoothing of bi-linear and bi-cubic resampling).

Sigma-reject is fundamentally different from a median or clip in that the rejected pixels have no effect on the final value. Median or clipped pixels do affect the final value by shifting the selection.

Clip

- Clip min/max

Clip rejects the highest valued (maximum) pixels and/or the lowest valued (minimum) pixels in the pixel stack. Clip is a hybrid of median and mean and offers high S/N associated with mean combined with the data cleaning of median. Median is simply a most-restricted clip (e.g. clipping 1 Max; 1 Min from a stack of 3 images is the same thing as median) and mean is the least restricted clip (0 Max; 0 Min).

next concept

previous concept

Combine

These procedures combine all included images in the stack to create a new image. The new image will have include status set to “No” so that it will not be included in any subsequent combines (unless desired, in which case set include status to “Yes”).

Sum

A new image is created from the weighted sum of all included images:

Pr = sum over all i [Pi*Wi]

- Pr = pixel in result image
- i = included image
- Pi = pixel in image i
- Wi = weight for image i

If Pi = MissingValue then Pi is imputed from the non-missing pixels in the other images. If all Pi are missing then Pr = missing.

If the stack is not normalized then all weights = 1 (unless they were changed via image manager) and this procedure acts as a simple non-weighted sum.

Mean

A new image is created from the weighted mean of all included images.

Pr = (sum over i [Pi*Wi]) / (sum over i [Wi])

- Pr = pixel in result image
- i = included image
- Pi = pixel in image i
- Wi = weight for image i

If Pi = MissingValue then Pi is imputed from the non-missing pixels in the other images. If all Pi are missing then Pr = missing.

If the stack is not normalized then all weights = 1 (unless they were changed via image manager) and this procedure acts as a simple non-weighted mean.

Weighted mean and weighted sum are subtly different and Sum/N will not necessarily equal Mean. This is because Mean is based on the normalization reference image, whereas the Sum is independent of the normalization reference.

Median

A new image is created from the un-weighted median of all included images:

Pr = middle value of all non-missing Pi

- Pr = pixel in result image
- i = included image
- Pi = pixel in image i

Weights are not used and pixels with MissingValue are not imputed, they are removed from the calculation.

Median should be preceded by normalization in order for it to function properly.

Data rejection is not necessary (though it is acceptable) because median is a de-facto data-rejection algorithm. Because the median is un-weighted it should not be used on stacks containing different exposure-times or other differences that result in significantly different normalization weights.

Minimum

A new image is created from the minimum of all included images:

Pr = min of all i [Pi]

- Pr = pixel in result image
- i = included image
- Pi = pixel in image i

Weights are not used and pixels with MissingValue are not imputed, they are removed from the calculation.

Minimum should be preceded by normalization in order for it to function properly.

Maximum

A new image is created from the maximum of all included images:

Pr = max of all i [Pi]

- Pr = pixel in result image
- i = included image
- Pi = pixel in image i

Weights are not used and pixels with MissingValue are not imputed, they are removed from the calculation.

Maximum should be preceded by normalization in order for it to function properly.

next concept

previous concept

Color

CCDStack handles color images using a psuedo-Lab scheme. The image data consists of a Luminance (L) 32-bit floating-point 2-dimensional array and two color-ratio 32-bit floating-point arrays. These psuedo-Lab values are converted to RGB for the display bitmap, output files, and color adjustments.

Color images are always not included. Thus it is not possible to directly perform most stack procedures on the color image. A color image can be separated into R,G,B and L components, which can then be included.

Creating a color image

Creating an RGB color image from Red, Green, Blue filtered images:

1. Process the red, green, and blue images separately: calibrate, register, normalize, data reject, and combine each set of color images independently.
2. Start a fresh instance of CCDStack (or remove all images) and load the final red, green, and blue images
3. Register the R,G,B images.
4. Select Color, create from the main menu. Enter approximate filter ratios (they can be fine-adjusted later). Don't worry about actual exposure times because CCDstack accounts for that when applying the ratios.
5. After the initial color image is created the program prompts for set background This will neutralize background hueand set the background level used to adjust ratios. It will optionally desaturate colors for pixels below or near the background average.
6. The adjust color form is used to make fine adjustments. Closing the form applies the factors to the image data.

Creating an LRGB color image from Luminance and Red, Green, and Blue filtered images:

1. Process the red, green, and blue images separately: calibrate, register, normalize, data reject, and combine each set of color images independently.
2. Start a fresh instance of CCDStack and load the final L, R, G, and B images. Note that binned RGB images will be automatically unbinned to match an unbinned L (i.e. don’t worry about different binning, the program will figure it out).
3. Register the R,G,B images, using L as the reference/base image.
4. Select Color, create from the main menu. Enter approximate filter ratios (they can be fine-adjusted later). Don't worry about actual exposure times because CCDstack accounts for that when applying the ratios.
5. After the initial color image is created the program prompts for set background . This will neutralize background hueand set the background level used to adjust ratios. It will optionally desaturate colors for pixels below or near the background average.
6. The adjust color form is used to make fine adjustments. Closing the form applies the factors to the image data.

Creating an L+RGB color image from Luminance and Red, Green, and Blue filtered images:
L+RGB adds the RGB into the Luminance of an LRGB.

1. Process the luminance: calibrate, register, normalize, data reject, and combine to create the LumMaster.
2. Process the red, green, and blue images separately: calibrate, register, normalize, data reject, and combine each set of color images independently. Use the LumMaster as the registration base (reference image) for each set (do not include it in the data reject or combine).
3. Start a fresh instance of CCDStack and load the final L, R, G, and B images and create the RGB color image: Select Color, create from the main menu. Do NOT specify the Luminance.
4. Extract a grayscale image from the RGB image (Menu: Color; Convert; retain the color image).
5. Add the extracted grayscale to the LumMaster (Menu: Process; File Math). Optionally, use Pixel Math to adjust the relative strengths of each image prior to add.
6. Replace the RGB image Lum.

The adjust color form is used to make fine adjustments. Closing the form applies the factors to the image data

Adjust colors

The adjust color form is used to change RGB ratios, offsets and saturation. The result of changes are shown on screen, but the underlying image data is not actually modified until the apply button is pressed (thus closing the form without pressing the apply button will revert to the image before the form opened).

If possible, set background should be performed prior to adjusting colors in order to avoid hue drift.

R,G,B sliders are used to adjust color factors (ratios) and offsets. Color ratios are applied as a factor offset by the background ADU via slope and intercept (background is the intercept and factor determines slope). Color offsets only affect hue, not luminance values, so the offsets are automatically normalized to have a total R+G+B offset = zero.

White/gray balance adjusts RGB ratios or offsets so that each color has the same average value within the selection area. This adjustment is applied to fators (ratios) or offsets based on the choosen mode. This procedure can be used to determine color ratios based on a G2 star or star fields or spiral galaxies (most of which have an overall white aperarence).

Background ADU is used for scaling RGB ratios (factors). The Background ADU is the "pivot point" used to apply ratio changes. This prevents hue drift when changing ratios (factors). With the correct ADU entered, the background hue will be largerly unaffected by ratio changes. Background ADU is automatically set by the set background button or it can be entered manually (type-in the ADU).

Set background does the following, based on a user selected area:

1. Computes and applies R,G,B offsets to remove background color (due to sky pollution)
2. The desaturate background option removes color from the background and near-background.
3. The background agerage ADU is placed in the Adjust colors form. This number is used as the pivot point when adjusting RGB ratios, which eliminates hue-drift (e.g. the background color does not significantly change for different RGB ratios).

Add/Replace Luminance

This is used to convert RGB to LRGB or to replace the Luminance in an LRGB. Because the new Luminance likely has a different background it is advisable to set background after applying the new Luminance frame.

One shot color / Bayer color

Raw FITS and SBIG images from "one shot color" cameras should be calibrated then converted to component Red, Green, and Blue frames for normal processing. Alternately "one shot color" images can be input as 16-bit color TIFF then converted to component Red, Green, and Blue frames for normal processing.

Suggested method for processing "one shot color" Bayer pattern images:

1. load the Bayer pattern FITS or SBIG frames
2. Calibrate (dark subtract and flat field)
3. Save the calibrated images
4. Convert Bayer to Red (do not retain Bayer frames)
5. Register, Normalize, Data Reject, Combine (sum), save
6. Clear, load the calibrated Bayer frames (from #3)
7. Convert Bayer to Green, repeat step 5 using green
8. Repeat step 6 and 7 for Blue
9. clear, load the final Red, Green, Blue images and create color

next concept

previous concept

Sharpening

Deconvolution

Deconvolution is a process that sharpens the image via algorithms developed by Benoit Schillings. Two types are implemented: Positive Constraint and Maximum Entropy. The deconvolution algorithms work best with well-sampled resolution (FWHM > 3 pixels).

Deconvolution uses a selected star to model the point-spread-function (PSF).

The Maximum entropy algorithm generally requires between 100 to 300 iterations to produce optimal results. Too few iterations result in under-sharpening (it can actually produce some blurring). Too many iterations result in over-sharpening with harsh artifacts. This deconvolution works best within a limited range of brightness. Therefore specify a “maximum ADU” near the upper limit of interesting features. Specifying a too high maximum ADU results in sub-optimal sharpening and takes longer to execute.

The Positive Constraint algorithm generally requires between 10 and 100 iterations to produce optimal results.

Drag-out a rectangle via the mouse to limit the deconvolution to the selected area.

At the start of deconvolution a new image is created, which consists of a rescaled copy of the top image. Deconvolution usually takes a long time to complete. After completion the image is rescaled to match the ADU range of the original image.

Unsharp Mask

Unsharp mask is a method to sharpen and increase fine-scale contrast. A blurred (unsharp) mask is combined with the image to produce the sharpened image.

CCDStack implements two different methods of combining the image and mask: DDP and non-DDP (traditional):

CCDStack provides the following parameters to control the application of the mask:

- DDP or non-DDP method
- mask width
- radius
- strength
- maximum differential

next concept

previous concept

Blur

CCDStack implements 4 kernel filters that blur and smooth. Each filter operates on a circular matrix of pixels with the user specified diameter or radius.

Gaussian

This function approximates the blurring effects of the atmosphere and can be used to increase an image's FWHM to match other images (e.g. for RGB combine of images with disparate FWHM). The Gaussian function width can be specified as Standard Deviation or FWHM.

Median

Each pixel in the result image is selected as the median (mid value) of all pixels in the source image that are within the circular radius specified.

Mean

Each pixel in the result image is computed as the mean (average value) of all pixels in the source image that are within the circular radius specified.

Mode

Each pixel in the result image is selected as the mode (most common value) of all pixels in the source image that are within the circular radius specified. Round to nearest bins the pixel values prior to selecting the mode. This filter can be used to extract the background from an image, which can then be subtracted from the image to remove all "gradients" and flat errors. But it will also remove any diffuse nebula.

Note that custom kernel filters can be implemented in Pixel Math Compile.

Open

CCDStack reads files in the following formats:

- 32 bit/color floating point FITS
- 16 bit/color unsigned integer FITS
- 32 bit/color integer FITS
- 16 bit/color signed integer FITS
- SBIG uncompressed or compressed
- TIFF 8 or 16 bit/color
- JPEG
- BMP

CCDStack does not require that the file name suffix match the format (e.g. a FITS file named “image.001”).

Save data

CCDStack saves image data in the following formats:

- 32 bit floating point FITS
- 16 bit unsigned integer FITS
- 32 bit integer FITS
- 16 bit signed integer FITS
- 16 bit TIFF - scaled or raw

All of these formats can be used to save color images, in which case the bit depth is stated per color.

16 bit unsigned FITS and 16 bit raw TIFF files have a 100 ADU pedestal added to each pixel.

16 bit TIFF files can be written as:

- scaled - the underlying 32 bit data is scaled and sharpened according to the current display parameters to produce a full range 16 bit image.
- raw - the underlying 32 bit data is written as is (no scaling) with an added 100 ADU pedestal.

This

Saves the top image.

All

Saves all included images in the stack using the names in image manager (these names will usually be the original file names).

Option to append a suffix to the file names (e.g. image01_calibrate).

Option to over-write all or prompt for each.

Sequence

Saves all included images in the stack using the specified prefix and a sequentially numbered suffix.

Suffix number starts with specified suffix, unless there is no suffix number, in which case numbering starts at 001.

The images are saved in the sorted sequence of image manager save display Bitmap

Saves the top image display bitmap.

JPEG

TIFF (8 bits/color)

save Screen as JPEG

Captures and saves the current screen, including magnification and cropping.

Remove this image

Removes the top image from the stack. Save the image before removing, if appropriate.

remove all images Except this

Except this Removes all images from the stack except for the top image. Image are removed regardless of include status.

remove All images (clear)

Removes all images from the stack. This clears the program, which can be beneficial when operating on large stacks.

Exit

Save images before exiting, if appropriate.

Crop

Crops all images in the stack (regardless of include status).

Use the mouse to define crop area or enter pixel coordinates.

Expand Canvas

Enlarges the area of all images in the stack regardless of include status.

Does not magnify or resample the images.

Images are placed in the center of the enlarged canvas and surrounded with pixels set to MissingValue.

Duplicate

Creates a new image in the stack that is a copy of the top image.

Resize

Resizes all images in the stack (regardless of include status).

Bin 2z2

Each pixel in resulting images is the sum of 4 adjacent pixels from the original images.

The summed pixels preserves flux, but if you want to preserve the average ADU/pixel then use pixel math to divide by 4.

The resulting images are ½ the size of the original images.

Bin 3z3

Each pixel in resulting images is the sum of 9 adjacent pixels from the original images.

The summed pixels preserves flux, but if you want to preserve the average ADU then use pixel math to divide by 9.

The resulting images are 1/3 the size of the original images.

Specify Scale

Specify the magnification or the new image dimensions.

Select the interpolation method.

Preserve flux

Checking preserve flux will adjust the ADU so that the total ADU of the image remains the same and the average pixel ADU will increase with magnification <1 or decrease with magnification > 1.

If this option is unchecked then the average pixel ADU remains similar and the total ADU increases with magnification <1 or decrease with magnification > 1.

Rotate

Rotate 180 degrees

Transposes the pixels 180 degrees (no resampling).

Rotate 90 degrees clockwise

Transposes the pixels 90 degrees clockwise (no resampling) for all images.

Rotate 90 degrees counter-clockwise

Transposes the pixels 90 degrees counter-clockwise (no resampling) for all images.

Specify Rotation Angle

Specified images are rotated by the specified rotation angle.

Checking expand canvas will enlarge the image size to avoid corner clipping. the resulting blank areas are set to MissingValue.

Select the interpolation method for resampling. Bi-cubic B-spline is recommend for rotations.

Mirror

Flip Horizontal

Mirror transposes the pixels left-to-right (no resampling) for all images.

Flip Vertical

Mirror transposes the pixels top-to-bottom (no resampling) for all images.

Camera Manager

If you do not know any of these values then accept the default numbers that appear in the entry.

Camera

Camera name defaults to the Header Tag but can be changed by the user.

Header Tag

The value of FITS Header keywords [“CAMERA”, “INSTRUME”].

Most image acquisition software place a camera name in the FITS header. If the FITS header has no camera designation then the "default" camera is used.

Bin

Separate entries are required for each binning because some cameras change gain when binned.

Gain

Gain is used to convert ADU (Analog to Digital Unit) to electrons:

```
gain * ADU = electrons
```

Several procedures rely on Poisson statistics, which require calculations based on electrons.

Readout noise

Readout noise is expressed in electrons.

Pedestal

The pedestal should approximately equal the average bias level. If you do not know the camera pedestal then measure the median pixel value for a short dark exposure and use that number.

Most CCD cameras produce frames with an elevated average-bias or pedestal. Additionally, many CCD processing software add a pedestal. In both cases, this is done to avoid negative numbers.

CCDStack handles negative numbers and uses pedestal = 0 (e.g. nothing is added to an image after dark subtraction).

One Shot Color

Check this box for Bayer pattern one-shot color cameras.

Allow Header information to override Camera parms

When this box is checked, any camera parameters (Gain, Readout noise, and Pedestal) that are present in the FITS Header will overide the values specified in Camera Manager. If the box is unchecked, then Camera Manager values are used regardless of FITS Header data.

Settings

Check Web for Updates

Option for CCDStack to check a web server for new version. It is necessary to give CCDStack permission to access the web, so it may be necessary to modify firewall and security settings.

Files

use static Flat lib

Check this to maintain a static flat lib. see concepts.

Otherwise CCDStack will first look for flats in the current image lib.

write minimal FITS header (this session only)

Checking this causes CCDStack to write all FITS files with a minimum number of parameters / keywords.

CCDStack stores many parameters in the FITS header, some of which may not be recognized by other software. Some software may malfunction or refuse to read FITS files with unexpected keywords, in which case use this option to create FITS files that should be readable by any FITS capable programs. Also note that some software will only read certain bit depths, e.g. 16-bit unsigned FITS.

This option is only applied to the current session. The next time you start CCDStack or remove all the option is repealed.

JPEG Quality

This determines the compression and quality of JPEGs saved.

Range 1 to 100:

- lower number = lower quality with smaller file size
- higher number = higher quality with larger file size

Directory for temp files:

Specify the desired directory for temporarily storing cached images.

Warning: Vista users should avoid specifying directories in "Program Files" to avoid problems with UAC.

clean-up (delete) CCDStack temporary files

Use this to clean-up temporary files created by large stacks via caching. Normally these files are removed at the end of a session but an interrupted session (e.g. memory error crash) may leave temporary files on the disk.

Do not press this button if there is another CCDStack session in progress at the same time. Close all other sessions first.

Large Stacks

These settings control how CCDStack handles large stacks. Stack size is measured as the total number of pixels (number of pixels per image times number of images).

Enable Cache

64 bit operating systems (Vista 64, XP-64) have no need for image caching and there is no practical limit on stack size (other than

the speed and RAM of this machine and your patience).

32 bit operating systems (Vista 32, XP) impose a severe memory limit on .NET applications so CCDStack can be set conserve memory by caching image files to disk when the image is not on top. It is recommended that the default cache settings be used.

Cache Threshold

Caching is used when the stack size (in mega-pixels) exceeds this value. Note that the stack size is displayed on the bottom left border, which also indicates caching (when in use).

Cache 3-d Buffer Size

Some operations require pixels from every image in the stack but cached stacks only keep one image in memory. So CCDStack processes portions of each image. Cache 3-d Buffer Size is the total size of the portion (image-portion * number of images).

Maximum Stack Size

Even with caching there is still a practical limit on stack size. Exceeding the limit is likely to exceed the memory limit.

Memory Warning

Large stacks naturally consume large amounts of memory (RAM). .NET applications running in 32 bit operating systems (Vista 32, XP) have an effective Virtual Memory (VM) limit of about 1 gigabyte but become potentially unstable above 850 megabytes. These limits apply regardless of the amount of physical RAM installed.

64 bit operating systems (Vista 64, XP-64) have no practical VM limit so this warning should be disabled.

Note: The amount of Virtual Memory (VM) consumed by CCDStack is shown in parentheses on the bottom left border.

System Priority

Use to set CCDStack's main thread priority.

If you are simultaniously using camera software it may be useful to set CCDStack to a low priority in order to avoid slow camera downloads.

Also, you can set a low priority while running a large deconvolution in the background.

Calibrate

Calibrate – Calibration Settings

see Concepts

Adaptive dark subtract

Flat frame - dark/bias subtraction

Flat frame - auto bin/unbin

The bias/pedestal is only used for adaptive dark subtraction and/or subtracting from a master flat made from unsubtracted frames.

Set "include" status to calibrate a sub-set of the stack.

Make Master Dark

see Concepts

1. Take a series of dark exposures using the same exposure time and camera temperature
2. Select “Process”, “Calibrate”, "make master dark"

Make Master Flat

see Concepts

1. Take a series of flats exposures (need not be equal exposure times)
2. Select “Process”, “Calibrate”, "make master flat"

Make Master Bias

see Concepts

1. Take a series of very short dark exposures.
2. Select “Process”, “Calibrate”, "make master bias"

Data Reject

Procedures

See concepts and procedures.

Show Rejected Data

Turn this on to continue showing rejected pixels (even when the Data Reject Procedures form is closed). Showing rejected pixels may slow screen performance.

Select again to turn-off.

Data Rejection procedures

see concepts

clear before apply checkbox

If the clear before apply box is unchecked then several different procedures may be applied additively in succession. Otherwise, this option will clear pixel rejection status before applying the procedure (except for pixels with MissingValue, which are always rejected).

restrict to selection checkbox

Data reject procedures are applied to the entire image unless you drag-out a selection rectangle then check the restrict to selection checkbox. Use this option to treat a limited area within the image (e.g. apply a more severe rejection to an airplane trail).

Restrict to selection is also useful to preview and test various methods and parameters because it will execute faster than including the entire image.

undo/redo

Most of the data rejection procedures allow one level of undo/redo. This makes it easy to try-out a possible rejection procedure without ruining the current state.

Single Image data rejections

Clear

Resets pixel reject status to not-rejected, except that MissingValue pixels always remain rejected.

Reject hot/cold pixels

Each pixel within an image is compared to the filtered-average of neighboring pixels. Pixels that are different from the filtered-average by more than (filtered-average) / (Strength) are rejected. E.G. a 500 ADU pixel with neighbors filtered-average of 400 ADU will be rejected when Strength >= 4 (because 400 / 4 = 100 = abs(500 - 400)).

The upper limit parameter is a cutoff. Pixels with values greater than the upper limit will not be rejected. This parameter may be used to restrict operation to background only.

Reject blooms

Star blooms are rejected but most star cores are not rejected (use Reject Range to reject the star cores).

There are basically 2 ways to remove blooms:

1) Reject and impute the bloomed areas. If all of the images in the stack contain similar blooms then the bloomed areas should be rejected and imputed.

2) Reject and replace the bloomed areas with real data from non-bloomed (or cross-bloomed) images. Reject the blooms then set rejected pixels to missing value, then register and normalize prior to combining.

Bloom rejection should be performed prior to registration. Otherwise, using this procedure on a registered image may also require a subsequent grow in order to reject near-by pixels affected by resampling the blooms.

The saturation parameter sets the lowest ADU value within the bloom area. A default setting of 40,000 is good for most 16-bit cameras.

Reject range

Rejects all pixels having ADU within the specified range.

Reject range can be used to blend images based on ADU. For example, to blend 2 images:

1. Decide on the crossover value X for the 2 images: lower valued pixels will primarily come from image 1 and higher valued pixels will mostly come from image 2.
2. Navigate to image 1 (from which the lower valued pixels will come) and reject range > X+50 and < 1,000,000 (any high value larger than the brightest pixel).
3. Navigate to image 2 (from which the higher valued pixels will come) and reject range > 0 and < X-50
4. Combine the 2 images. Note that there is a X +-50 ADU overlap where the combined image uses pixels from both images.

Clear range

Clear range un-rejects pixels with ADU values within the specified values.

This can be used to remove undesired bright-pixel rejections from a Poisson sigma-rejection of a stack of images with dissimilar FWHM.

Grow

Rejects all pixels near a previously rejected pixel. Number of pixels specifies the grow radius. If a previously rejected pixel lies within the radius then the pixel is rejected. It is often useful to restrict the area for applying grow.

Some typical uses for grow:

- Enlarge bloom rejections in images that were registered prior to bloom rejection.
- Expand rejection of airplane or satellite trail
- Crop border overlay areas

Freehand draw

Press the mouse button and slowly draw over the image to reject pixels.

Erase un-rejects pixels (unless they are missing value, which cannot be un-rejected).

Pen diameter sets the width and height of the drawing tool.

Impute rejected pixels

Rejected pixels (including MissingValue) are replaced with imputed values based on non-rejected neighboring pixels.

A Gaussian matrix is used to impute the value.

Width is the Gaussian function width (STD = 2.34 FWHM) used to construct the matrix.

Iterations specifies the number of times to perform the procedure (large continuous areas of rejected pixels will be progressively imputed via more iterations. E.G. specifying 3 iterations is the same thing as specifying 1 iteration and pressing the apply button 3 times.

Some uses for impute rejected pixels are to apply it after:

- Bloom removal
- Remove hot/cold pixels
- Remove airplane/satellite trails in a single image via freehand draw

Set rejects to missing value

Converts all rejected pixels to MissingValue. The original ADU for converted pixels is lost.

Some uses of this conversion:

- Preserve the data reject map using 32-bit FITS.
- Preserve bloom rejections; convert prior to registration
- Make the current data reject map permanent (subsequent clearing will not un-reject converted pixels).

Stack data-reject procedures

see concepts

Find sigma multiplier based on percentage in top image

Sigma and linear reject procedures have the option to auto-calculate the factor (multiplier) based on the percentage of rejected pixels in the top image. Check top image % to use this feature. CCDStack will hunt for a factor that produces the specified percentage. Sometimes it takes awhile for the program to find the appropriate factor. Sometimes it is not possible to find a factor, in which case the program will report an error and suggest re-normalizing the stack.

STD sigma-reject

Computes the mean and standard deviation of the pixel stack. If any pixels lie outside the bounds of [mean +/- (std*multiplier)] then the most outlying pixel is rejected and the process is reiterated until no pixels lie outside the bounds or the specified maximum number of iterations has been performed.

The standard deviation statistic is most effective when there are many images and it is somewhat unreliable for stacks of less than 8-10 images, though it is somewhat usable for as few as 3 images.

The STD sigma-multiplier must be adjusted based on the number of images in the stack. Use a small restricted area to quickly test various factors or specify a rejection percentage for the top image.

Poisson sigma-reject

Computes the mean and predicted noise of the pixel stack. The noise is predicted from Poisson and readout noises. If any pixels lie outside the bounds of [mean +/- (noise*multiplier)] then the most outlying pixel is rejected and the process is reiterated until no pixels lie outside the bounds or the specified maximum number of iterations has been performed.

Poisson noise is the intrinsic noise of the light signal and is mathematically related to the signal intensity. Thus it is possible to predict the noise for any signal. If one or more values in the pixel stack lie outside the calculated Poisson noise range then those pixels are rejected.

Poisson noise sigma = sqrt((ADU-pedestal)*gain+readout^2)

Gain and readout are from the camera manager. Pedestal is from the FITS header or camera manager.

For a normal, un-resampled image:

- Sigma factor = 1 produces approximately 14% rejection due to normal noise distribution.
- Sigma factor = 2 produces approximately 1.5% rejection due to normal noise distribution.
- Sigma factor = 3 produces approximately 0.1% rejection due to normal noise distribution.

It is advisable to use a sigma factor between 2 and 3 for most cases. Use a lower, more-aggressive sigma factor to treat a restricted area for the subtle effects of airplane trails and the like.

Poisson sigma-reject can be applied to a stack that contains as few as 2 images. If there are only 2 pixels remaining in the stack and they lie outside the sigma bounds then the high-value pixel is rejected (because most outliers and are positive - cosmic rays, airplanes, etc.).

Images with disparate FWHM exhibit an odd behavior with Poisson sigma-reject. Star shapes of mismatched-FWHM have different intensity profiles that can exceed sigma bounds. This will result in star-center rejections and/or rejections in the star peripheries. These effects can be limited via the following:

1) Set a low number of iterations (1 or 2).

2) Process sub-stacks based on resolution.

3) Try linear factor reject or STD sigma-reject instead of Poisson.

Linear factor reject

Computes the mean of the pixel stack. If any pixels lie outside the bounds of [mean +/- (mean*factor)] then the most outlying pixel is rejected and the process is reiterated until no pixels lie outside the bounds or the specified maximum number of iterations has been performed.

This procedure operated similar to Poisson sigma-reject except “sigma” is linear instead of square root. Thus the higher ADU values have larger bounds and are less likely to be rejected. Linear factor reject is most aggressive in background areas.

Appropriate linear factors vary by image characteristics. Use a small restricted area to quickly test various factors or more quickly set a factor based on percentage

Linear factor reject can be applied to a stack that contains as few as 2 images. If there are only 2 pixels remaining in the stack and they lie outside the bounds then the high-value pixel is rejected (because most outliers and are positive - cosmic rays, airplanes, etc.).

Clip min/max

Rejects the highest valued (maximum) pixels and or lowest valued (minimum) pixels in the pixel stack. This procedure works best on stacks containing numerous images.

The number parameters specify how many pixels to clip from the pixel stack. Min + Max should be at least 1 less than the total number of images; otherwise every pixel in every image will be rejected. Usually Min and Max numbers should be equal, though there are situations that benefit from non-symmetrical clipping (e.g. clip a 2-image stack: 1 Max; 0 Min).

Poisson fit to this image

Computes the mean and predicted noise of this image. The noise is predicted from Poisson and readout noises. If any pixels lie outside the bounds of [mean +/- (noise*multiplier)] then the most outlying pixel is rejected and the process is reiterated until no pixels lie outside the bounds or the specified maximum number of iterations has been performed.

See Poisson sigma-reject for more information about Poisson calculations.

show rejected data

Exiting the data reject form will hide rejected pixel, but any pixels rejected at that point remain rejected and will be treated as such for image combine.

Turn-on "show rejected data" to show rejected pixels when the data reject procedures form is closed.

Pixel Math

Pixel math applies mathematical operations to pixels in the top image or all included images.

There are 2 methods available to perform pixel math: form and compiler.

Pixel Math Form

Use this form to add, subtract, multiply, divide, or raise to a power (exponential) the pixels by the specified constant.

Compiler

Beginners

1. Construct a statement via menu[Statements -> Single statement]
2. Replace the statement's number with your desired constant
3. Apply the statement via 'Execute' menu

Pixel math operations on a MissingValue result in MissingValue. Also, pixel math operations that produce impossible numbers (e.g. divide by zero) result in MissingValue.

Advanced users

C# statements will be processed for each pixel in the image(s)

Syntax is very similar to C++, Java, JavaScript, etc. (most simple statements are identical for these languages). Statements must be terminated with semicolon; all words are case sensitive invalid C# statements will abort the compile your code should modify the predefined variable:

> pixel = ADU for each pixel

the following predefined variables are available for reference only (do not assign):

> y = Row variable used to loop thru the image(s)
> x = Column variable used to loop thru the image(s)
> image[y,x] = input pixel ADU (32 bit float) prior to any statements

for information on available Math methods see http://msdn.microsoft.com

see Microsoft for more information on C#.

Sample Code

```
pixel = pixel + 100; // add 100 adu to each pixel
pixel += 100; // add 100 adu to each pixel

pixel = pixel-100; // subtract 100 from each pixel
pixel -= 100; // subtract 100 from each pixel

pixel = pixel*2; // multiply each pixel by 2
pixel *= 2; // multiply each pixel by 2

pixel = pixel/2; // divide each pixel by 2
pixel /= 2; // divide each pixel by 2

pixel = Math.Pow(pixel, 2); // square each pixel

pixel = Math.Sqrt(pixel) * 100; // square-root is multiplied by 100 to avoid quantization noise

pixel = 100*Math.Pow(pixel,0.3); // apply gamma 0.3
pixel += x/100; // creates a 1% horizontal gradient (or removes an existing gradient)
pixel += (x+y)/200; // creates a 1% 45deg gradient (or removes an existing gradient)

// simple kernel filter:
pixel = image[y,x] * 0.2 +
image[y-1,x] * 0.2 +
image[y+1,x] * 0.2 +
image[y,x-1] * 0.2 +
image[y,x+1] * 0.2;
```

Expert Users

Selecting 'Expert shell' from the 'Statements' menu will place a PixelMath Class template into the textbox.

There are 2 marked insertion points in the 'Process2dImage' method code for you to place persistent variables and looped expressions.

You may create and call your own methods within the PixelMath Class.

If the compile is successful then 'Process2dImage' is called by CCDStack.

It is recommended that you pre-debug your code via Visual Studio. Otherwise use "//" to comment-out all new lines and add them one-at-a-time in logical sequence.

File Math

The top image is arithmetically modified by a selected image.

The selected image must have the same dimensions (width, height) as the stack.

Add

The selected image is added to the top image, pixel by pixel.

Subtract

The selected image is subtracted from the top image, pixel by pixel.

Divide by

The selected image is divided into the top image, pixel by pixel.

Multiply

The selected image is multiplied by the top image, pixel by pixel.

Deconvolution

See concepts.

Select Method

Select Positive Constraint or Maximum Entropy

Select star

Single-click (left button) on a suitable star. The star should be well formed; not near saturation; uncrowded; and have typical FWHM. It is permissible to select another star before applying the deconvolution. Only the last star selected is used in the deconvolution.

Number of iterations

Maximum Entropy generally requires between 100 to 300 iterations to produce optimal results. Too few iterations result in under-sharpening (it can actually produce some blurring). Too many iterations result in over-sharpening with harsh artifacts.

Positive Constraint generally requires between 10 to 200 iterations to produce optimal results.

Maximum ADU (Maximum Entropy only)

Specify a maximum ADU near the upper limit of interesting features. Specifying a too high maximum ADU results in sub-optimal sharpening and takes longer to execute.

Restrict to selection

Just prior to applying the deconvolution, drag-out a rectangle via the mouse to limit the deconvolution to the selected area. This may be used to more quickly test different parameters. It may also be used to limit the deconvolution to an object within the image.

Apply

At the start of deconvolution a new image is created, which consists of a rescaled copy of the top image. Deconvolution usually takes a long time to complete. After completion the image is rescaled to match the ADU range of the original image.

Remove Gradient

This procedure adds a counter-gradient to the image based on sample areas.

Select background areas

Single-click (left button) to select an area.

Different methods for computing the counter-gradient are used, based on the number of selected areas:

1. nothing occurs when only one area is selected
2. two areas define a linear gradient running between the two points (centers of the areas)
3. three areas define a tilted plane that passes thru the three points
4. four or more areas define a complex space where every point is weighted by a psudo-gravitational model.

Up to 24 areas may be selected.

Create Scaled Image

This procedure creates a new image in the stack based on the top image and its current scaling parameters (controled by adjust display).

The display functions (DDP, Gamma, sharpen, etc.) normally affect only the display bitmap and do not affect the underlying image data, but this is a way to apply those functions to the image data.

The rescaled image data consists of 32-bit floating point data that ranges from 0 to 65,535 ADU, which will exactly fit within 16 bit unsigned integers. Thus the image may be saved as 16 bit FITS or TIFF for further processing in other software.

these inter-image procedures operate on stacks

Registration / Alignment

see concepts

Base (Reference) Image

The current top image at the start of registration is used as the reference image. This base image serves as the reference to which all other images are registered (the base image is not transformed or resampled). To select a different image for the reference, cancel registration, navigate to the desired image and restart registration.

Registration screen

During registration the screen shows an overlay of two images: a moveable top image combined with the reference image. The overlaid top image shows the current geometric transformation, which is not (yet) applied to the underlying image data or display bitmap. It is useful to blink the stack to verify the current geometric transformations prior to applying the transformations.

Dragging the mouse (hold right-click while moving) during registration shifts the top image. Use the coarse/fine control in manual registration to control the strength of the mouse drag.

Show pixels is automatically turned off at the start of registration and should not be turned back on until registration has completed.

Reset all / this

Resets image(s) to their original state.

Star snap

Star snap attempts to align the image(s) to the reference image by matching one or more reference star(s). The images should be approximately pre-aligned so that the chosen reference stars are aligned to within about 20 pixels. If necessary, use manual to roughly pre-align the images.

select/remove reference stars

Press the "Select/remove reference star" button then single click on star(s) to select reference star(s) within the base image The last star selected becomes the center of alignment rotation and magnification. Re-click a reference star to remove it.

Selecting only one reference star results in only [x,y] shifts (no rotation or magnification). Two or more reference stars result in a shift, rotation and magnification transform. Three or more reference stars produce an average of the transforms.

For a successful star snap:

- Avoid saturated stars
- Avoid very dim stars
- Avoid crowded star areas (especially double stars)
- Select reference stars that are separated by at least 100 pixels
- In most cases it is only necessary to select 2 or 3 reference stars
- One reference star is sufficient if no camera rotation and good polar alignment

To align images that are significantly rotated, do a 2 stage registration:

1. select 2 close-by stars (near the center of rotation), align.
2. select new reference stars that are well separated, align again.

auto-select reference stars

Press the button to generate reference stars. Use the "Select/remove reference star" button to alter the selection.

Remove all reference stars

Press this button the clear all reference stars and reset the rotation/scale center to the center of the image.

align all

Press this button to snap all included images.

align this

Press this button to snap the top image.

If the program cannot match reference stars then an error message will indicate a problem. Use the mouse-drag (hold left button while moving) or manual mode to approximately align the image(s) and/or select different reference stars; then snap again.

You should examine the stack before applying the registration (blink it). There are times when the algorithm may select a wrong star or cosmic ray, etc. If an image does not properly snap then: reset; use the mouse or manual mode to approximately align the image(s) and/or select different reference stars; then snap again.

FFT Automatic Alignment

FFT automatic alignment is based on work by Reddy and Chatterjii (IEEE transactions on Image Processing, vol 5, no 8), which relies on the Fourier shift theorem (De Castro, 1987). The central portions of each image are transformed to an FFT array and processed differently according to the options chosen. FFT automatic alignment works on images with or without stars (e.g. planetary, lunar, or terrestrial). Because it has limited rotational accuracy, FFT can be used to pre-align images for a subsequent star snap.

shift only

The images are align-shifted [x,y] with sub-pixel accuracy. No rotation or scale (magnification) alignments are performed. This requires less computation and is faster than the other options. The procedure can tolerate alignment errors up to several hundred pixels (depending on the image size).

Shift and rotate

The images are aligned without changing scale (magnification). The rotation-alignment accuracy is within 1/3 degree, which may not be sufficient for large rotated images. This procedure is computationally intense and takes longer than “shift only”.

Shift and rotate

The images are aligned, accounting for variations in rotation, scale (magnification) and shift. Scale differences greater than 1.4x will probably fail to align. Rotation-alignment accuracy is within 1/3 degree, which may not be sufficient for large rotated images. This procedure is computationally intense and takes longer than “shift only”.

Manual alignment

Use the mouse or arrow keys to shift the image and use the sliders to rotate and magnify. It is also possible to directly type-in the transforms (press Tab or click on other text-box to effect the change).

It is rarely necessary to use manual alignment for exact alignment. Manual alignment is mostly useful for approximately pre-aligning images prior to a star snap.

Strength

Sets the gradations used for shift, rotation, and magnification.

Center of rotation / magnification

At the start of registration, alignment rotation and magnification are centered at the center of the image. Star snap changes the alignment center is to the last chosen reference star. To reset the alignment center to the center of the image, clear all reference stars (button in Star Snap Tab).

Note that the [x,y] offsets displayed are always relative to (top, left) and thus may appear strange for rotated or magnified transforms.

Apply registration

Applying registration irreversibly changes the image data, so you should first blink the stack to verify that all images are properly aligned.

Quitting registration (closing the form) without Applying will result in no changes to the images (i.e. not aligned). Re-starting registration (select from Registration from Stack Menu) will remember any prior alignments, so if you accidentally quit before applying then re-start and apply.

To register the images it is necessary to resample the original image to interpolate pixels in the resulting image. CCDStack offers several interpolation methods. The subject of interpolation is complicated and the best method for a given set of images depends on several factors as discussed in interpolation methods. A good general default is Bicubic B-Spline.

Normalization

see concepts.

The stack should be registered prior to normalization.

Auto normalize

Auto normalize performs a slope and intercept normalization via histogram fitting of a center portion of the image or selected image area.

To select an area for normalization, use the mouse to drag a rectange before pressing "OK". Select an area containing a broad ADU range with minimal saturation (e.g. a cross-section of a galaxy). Try to include some sky background within the selection area.

Control normalization

Scalar normalization

Scalar (slope) normalization applies multiplicative factors to each image being normalized and primarily adjusts for exposure time and sky transparency. Use scalar to normalize flats when making a master flat via data-rejection.

Offset normalization

Offset (intercept) normalization applies additive constants to each image and primarily adjusts for sky background.

Both

Most real imaging situations benefit from applying both scalar and offset adjustments (slope and intercept).

Combine

seeconcepts

Sum

A new image is created from the weighted sum of all included images.

see concepts

Mean

A new image is created from the weighted mean of all included images.

see concepts

Median

A new image is created from the un-weighted median of all included images:

see concepts

Minimum

A new image is created from the minimum of all included images:

see concepts

Maximum

A new image is created from the minimum of all included images:

see concepts

Color

see concepts

Create

Select the Red, Green, Blue, and optional Luminance images.

All images are automatically unbinned to the largest image size (e.g. binned R,G,B images will be enlarged to match an unbinned Luminance).

Set the initial filter ratios. These can be fine-tined later.

see concepts for detailed instructions.

Adjust colors

Adjust color is used to change the RGB ratios, offsets and saturation. The result of ratio changes are shown on screen, but the underlying image data is not actually modified until apply is pressed. See concepts

The RGB sliders and controls operate according to the mode selected:

Factor mode

In Factor mode, the RGB sliders adjust the color ratios. The ratios are based on the original filter ratios entered at the time of image creation.

White/gray balance adjusts RGB ratios so that each color has the same average value within the selection area. This procedure can be used to determine color ratios based on a G2 star or star fields or spiral galaxies (most of which have an overall white appearance).

In factor mode, double-click the area above the Red, Green, or Blue slider to normalize the RGB ratios so that the selected color ratio = 1.0.

The Save factors button stores the current RGB ratios for the next color create.

Offset mode

In Offset mode, the RGB sliders adjust the ADU offsets.

The offsets are always normalized to sum = zero (after releasing the slider) so they only affect hue, not luminance.

White/gray balance adjusts RGB ADU offsets so that each color has the same average value within the selection area. This procedure can be used to determine and set the color offsets of sky background. However, it is recomended that sky background be treated via set background

The Set Offsets = 0 button resets all RGB offsets to zero

Background

Background ADU is used for scaling RGB ratios (factors). The Background ADU is the "pivot point" used to apply ratio changes. This prevents hue drift when changing ratios (factors). With the correct ADU entered, the background hue will be largely unaffected by ratio changes.

Background ADU is automatically set by the set background button or it can be entered manually (type-in the ADU).

The Set background button:

1. Computes and applies R,G,B offsets to remove background color (due to sky pollution)
2. Sets Background ADU
3. The desaturate background option removes color from the background and near-background.

Add/replace Luminance

The luminance data will be replaced with the selected image. It is recommended to set background after replacing the luminance in order to maintain the correct Background ADU.

Convert

Converts the color image to any combination of Red, Green, Blue, Grayscale (Luminance).

Check Retain Color image(s) to keep the original color image (otherwise it will be removed).

Convert Bayer

Converts One-Shot-Color Bayer pattern into any of the following:

- Color image
- Red
- Green
- Blue
- Grayscale (Luminance)

Bayer Pattern

Different cameras use different arrangements of the colored pixels that make up the Bayer patter. There are 4 possible configurations.

Check Retain Bayer image(s) to keep the original Bayer frame (otherwise it will be removed).

Windows

Most of the sub-windows/forms are accessable via the Windows main menu.

(e.g. if you close a sub-window then you can re-open it via Windows menu).

Adjust Display

Adjust the display parameters. These controls are only applied to the display bitmap and do not affect the underlying image data (i.e. the controls are non-destructive).

background and maximum sliders are elastic and the maximum slider is progressive (play with them to see what that means).

gamma applies exponential scaling. gamma = 1.0 results in non-exponential scaling (linear scaling when DDP is not checked).

the DDP checkbox applies the "Digital Development Process".

DDP is the DDP-factor applied within the DDP equation.

apply to all identically scales all images in the stack. (If the images are not normalized then this may result in undesirable results). While the apply to all box is checked, any changes made to the slifers will be applied to all images.

auto acale applies algorithmic scaling to the top image. A different algorithm is used for DDP vs. non-DDP. Auto-scaling is based on the visible area within the window (auto-scaling a zoomed-in view results in different scaling than a zoomed-out view). If apply to all is checked when auto scale button is pressed then all images in the stack are auto-scaled and apply to all is turned off (because each image will probalby have different scale values).

invert reverses the scaling results (e.g. black becomes white)

sharpen opens the sharpen form and applies unsharp-mask to the image (or all images if apply to all is turned on).

Magnification

Use to size/resize the screen display (zoom in/out).

Size to Window box automatically sizes the image so that it always fits within the main window.

Show Pixels checkbox forces the display to show whole pixels (no interpolation).

If the magnification is less than 0.5x then CCDStack automatically bins the raw data for computing the display bitmap.

zoom to selection will fit the #selected area# within the window.

Select image

The < > buttons navigate backward or forward thru the image stack.

The name of the top image (the visible image) appears on the top left border.

#Image Manager# provides an alternate method of navigating thru the stack.

Blink

Blink automatically cycles the images forward.

While blinking is in effect, the << >> buttons control blink speed.

If there are at least 2 included images in the stack then only included images are blinked (thus you can blink a subset of images by setting include via #image manager#). Otherwise all images in the stack are blinked.

Image Manager

The image manager is an interactive list of all images in the stack, including some information about each image. The list includes image name, include status, date and time, exposure length, filter, FWHM, and weights.

The file list may be sorted by any field (ascending or descending) by mouse-clicking the top label of any column (e.g. to sort by name – click the name tab). Click a second time to reverse the sort order.

Select an image by clicking the row of the desired image.

Use the <Del> key (delete) to remove images from the stack. Select image(s) to be removed by clicking the row button. To select all images between 2 images in the list - select the first image, press [shift] key while selecting the last image. To individually select several images - press and hold [ctrl] key during selections.

Name

To change an image name, mouse-click the name-box and type-over to change the name. Or save the image using the new name.

Include

The include column contains Y or N to indicate if that image should be included in procedures (e.g. only included images are combined). To change the include status of an image – double-click the include box or type-in Y or N (upper or lower case). Color images cannot be included (include status always = N).

Date Time

Date and time of the exposure, taken from the FITS headers.

Exposure

Exposure time taken from the FITS headers. Time is usually expressed in seconds (depending on the camera software). Combined images display the sum or mean exposure time (depending on combine method).

Filter

Filter name is taken from the FITS headers. Mouse-click the Filter-box and type-over to change the filter name.

FWHM

This field displays the FWHM (in pixels) for the last star #selected#. If the images are closely registered/aligned (and not cached) then every FWHM is refreshed. Otherwise only the top image FWHM is refreshed. FWHM = 999 means no star has been selected in that image.

Weight

This shows the statistical weight used for combining images. Weights are assigned when the stack is normalized An image weight can be changed via mouse-click and type-over. Decreasing the weight lowers the contribution from that image in the combined image. Increasing the weight raises the contribution from that image in the combined image. One reason for changing weights might be to favor sharper images or decrease the impact of blurrier images.

File Header Information

Displays information from the file header (FITS header).

Keywords related to image size and bit-depth are not displayed. (All images are 32-bit floating point and the image size is displayed at the bottom left of the main window).

Full Screen

Displays the image in the full screen, with no visible forms.

Press any key to return to normal display.

File procedures input files; output files; and remove images from the stack.

Mouse Actions

Image Pan

To pan across an image that is magnified larger than the screen - right-click-hold and drag.

Select Area

Left-click-hold and drag a rectangle wihtin the image to create a selection area. Statistics for the selected area will be displayed in the information window.

To remove the area selection:

- single-click anywhere on the image
- close the information window
- change the magnification
- pan across the image
- load or create a new image
- apply a process (e.g. blur, deconvolve)

Select Star

double click on a star to display the FWHM in the information window and image manager

Information Window

The information window appears automatically when selecting an area or star.

The most recently selected item is displayed at the top of the information window. Prior selections scroll down, with the first selection at the bottom.

If an area selection is active (the selection rectangle is visible on the screen) then navigating to a new image will update the information window with data from the new top image.

Closing the information window clears the information.

Selected Area

The following information is displayed for a selected area:

Image File

Name of the image measured.

rectangle

The dimensions of the selected area.

diagonal

The length of the rectangle's diagonal. This is also the pixel distance between the 2 points selected with the mouse.

pixels

Number of pixels contained in the selected area.

rejected

Number of rejected pixels within the selected area.

mean

The mean (average) value of the non-rejected pixels within the selected area.

STD

The Standard Deviation of the non-rejected pixels within the selected area.

S/N

simple pixel S/N = mean / STD

median

The median (middle value) of the non-rejected pixels within the selected area.

int mode

The integer mode is the most common value (rounded to integer) of the non-rejected pixels within the selected area. This is often a good measure of the background intensity because it is unaffected by uncrowded stars.

min

The minimum (lowest) value of the non-rejected pixels within the selected area.

max

The maximum (highest) value of the non-rejected pixels within the selected area.

R:G:B ratio

The average color of the non-rejected pixels within the selected area.

Selected star

The following information is displayed for a selected star:

Image File

Name of the image measured.

centroid

The pixel coordinates of the calculated centroid.

FWHM

Measured Full Width Half Max. The FWHM algorithm interpolates sub-pixel rings around the centroid to produce a conservative and robust measure of reasonably sampled stars. The FWHM for under-sampled stars may be overly conservative (slightly greater than would be measured via other methods).

background

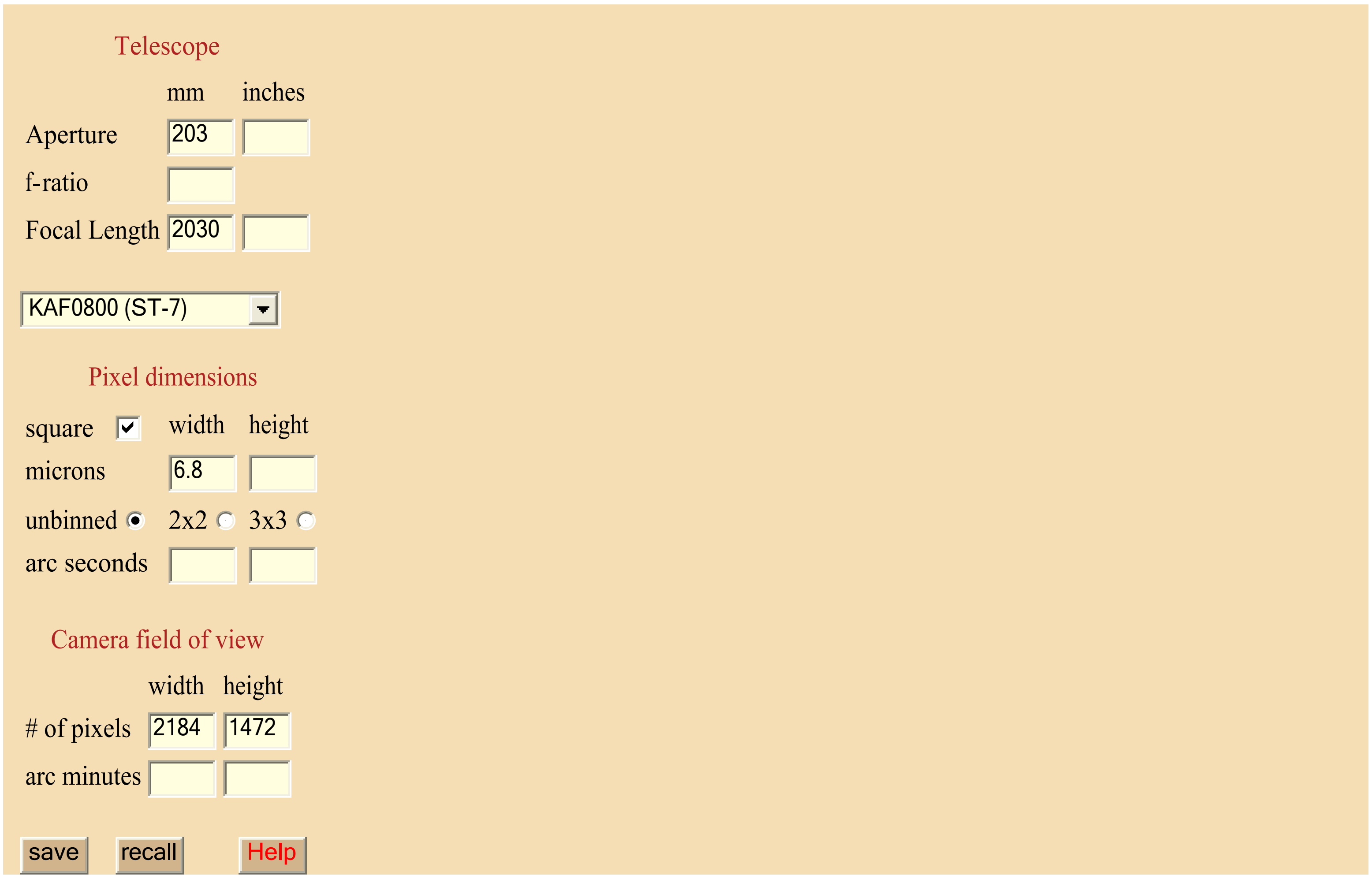
The interpolated background value calculated by the FWHM algorithm.

maximum

The sub-pixel interpolated value at the centroid. (Often is not equal to the brightest pixel).

flux within halfMax

The sum of pixels within a radius of the Half Max distance of the FWHM. The flux is calculated to sub-pixel accuracy via interpolation.



Telescope

	mm	inches
Aperture	203	
f-ratio		
Focal Length	2030	

KAF0800 (ST-7)

Pixel dimensions

square ✓	width	height
microns	6.8	
[✓ unbinned]	2x2	3x3
arc seconds		

Camera field of view

	width	height
# of pixels	2184	1472
arc minutes		

MissingValue is a special pixel value that denotes unknown value. MissingValue is assigned to pixels that have no meaningful value, such as pixels in the offset border region of a registered image.

MissingValue pixels are given special treatment in several procedures, which essentially makes them "transparent". For example, substitute values are automatically imputed for MissingValue pixels in image combine

Pixels with MissingValue are always rejected. Rejected pixels can be converted to MissingValue, which is useful for such things as bloom rejection

CCDStack uses a 32-bit constant for MissingValue (-33,333). MissingValue cannot be represented in 16-bit or 8-bit images, so to preserve MissingValues in output files use 32bit FITS (float or integer).

Pixels with MissingValue are shown as dark gray on the screen and the reported pixel value = "MV".

Normalization

see concepts.

The stack should be registered prior to normalization.

Reference image

One image is the reference image to which all others are normalized. The current top image at the start of normalization is used as the reference image (to select a different image for the reference, cancel normalization, navigate to the desired image and restart normalization).

Auto normalize

Auto normalize performs a slope and intercept normalization via histogram fitting of a center portion of the image or selected image area.

Control normalization

#Control normalization# performs any of these types based on user selected areas:

- Scalar normalization applies multiplicative factors to each image being normalized and primarily adjusts for exposure time and sky transparency. Use scalar to normalize flats when making a master flat via data-rejection.
- Offset normalization applies additive constants to each image and primarily adjusts for sky background.
- Both - most real imaging situations benefit from applying both scalar and offset adjustments (mathematically speaking, this is linear fitting via slope and intercept). It is recommended that you use “both” normalization to prepare astronomical images for median or pixel rejection.

Verify normalization

The scalar and offset values calculated for each image are displayed in the #information window#. To verify normalization – check #apply to all# in #adjust levels# and #blink# the stack. If the normalization is inadequate then try again using a different area or switch from #auto# to #control#. Repeated normalizations are OK.

Weights

The #combine# procedures use statistical weights to optimize the resulting S/N. These weights are calculated from normalization as the inverse of the scalar factor.

For example, after normalization the average ADU of a 5-minute exposure will approximate the average ADU of a 10-minute exposure (that’s what normalization does). If these images are summed without weights then the 10-minute exposure will contribute the same S/N as the 5-minute exposure, thus resulting in sub-optimal S/N. But a weighted sum preserves this difference to produce optimal S/N.

The weights are shown in #image manager#. It is possible to override a weight by entering a new weight in the #image manager form#. One possible reason to modify a weight might be to decrease the impact of images with poorer resolution.

next concept

Sharpen Form

This form is opened when the adjust display sharpen box is checked.

DDP method

The blur mask is applied within the DDP function (simplified):

```
sharpPix = Pix /(MaskPix+DDP)
```

Non-DDP (traditional) method

the blur mask is applied as:

```
sharpPix = (Pix-MaskPix)*Strength + Pix
```

CCDStack uses a Gaussian blur to create the mask, with control over the blur function.

mask width

The width (STD in pixels) of the Gaussian function. This is the main sharpening control. The optimal width for a given image depends on sampling, S/N, and the contrast power-spectrum of objects in the image. Smaller value produce finer sharpening with less contrast enhancement. Larger values produce greater contrast enhancement but may also produce undesirable artifacts such as “panda eyes” around stars (these artifacts can be limited by some of the other controls).

radius

Number of pixels over which the Gaussian function is applied. The radius should generally be set greater than the mask width (>2x width is a good default). Setting a smaller radius can limit undesirable artifacts, such as “panda eyes”.

strength

Strength only applies to non-DDP method (see above). threshold (bitmap) If a pixel’s bitmap value (0-255) is less than the specified threshold then the pixel is not sharpened. Use this to avoid sharpening low S/N areas such as the background. A typical setting is 75-100.

maximum differential

A differential is calculated for each pixel based on the sharpened value divided by the original value. If that differential exceeds the specified maximum then the pixel is not sharpened. Use this to limit artifacts such as “panda eyes”. A good default setting is 1.2-1.4.

Applying an unsharp mask

Check the sharpen box in the adjust display form to apply sharpening to the display bitmap. This does not alter the underlying image data.

To apply a non-DDP style Unsharp mask to the image data, press the “apply permanently” button in the sharpen settings form.

The only way to apply a DDP-type unsharp mask to the image data is to create a rescaled image using the desired sharpening, as set in the above settings accessible via adjust display

Check for recent updates and upgrades:

http://www.ccdware.com/downloads/updates.cfm