Deep Sky Image Calibration, Integration and Processing with PixInsight

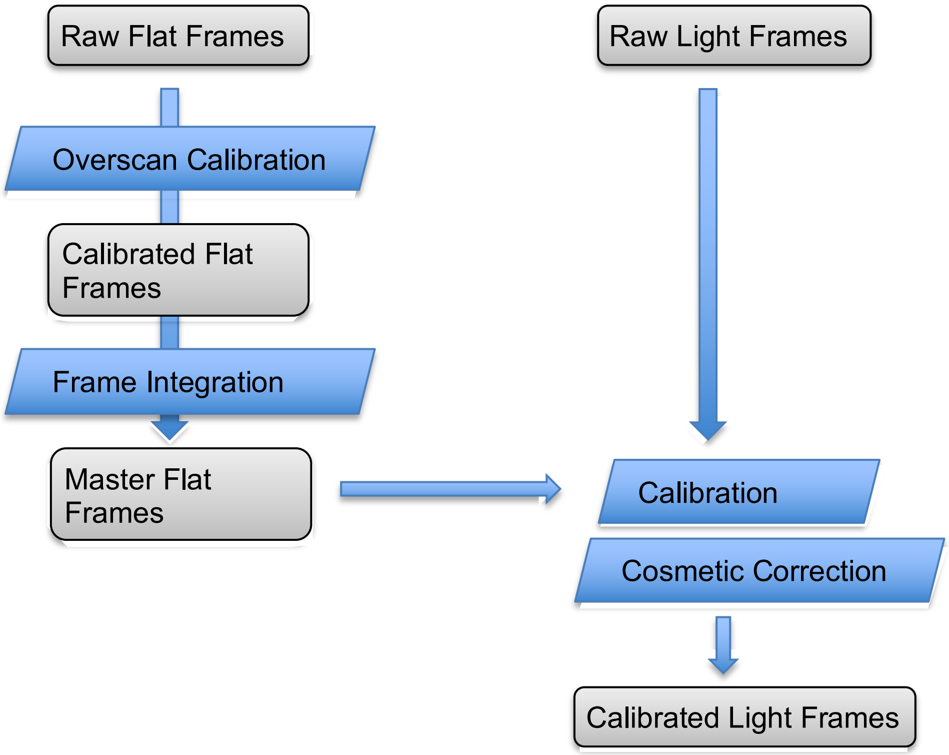
M81, 88x600s, January 2021 & 2022, Celestron 9.25 EdgeHD /f7, QHY268C, Optolong L-Pro Filter, Cornwall (c) Dean Ashton

M27 96X600s, 2021 & 2022, 235mm EdgeHD /f10, QHY268C, Opt. LeNhance Filter, Cornwall (c) Dean Ashton

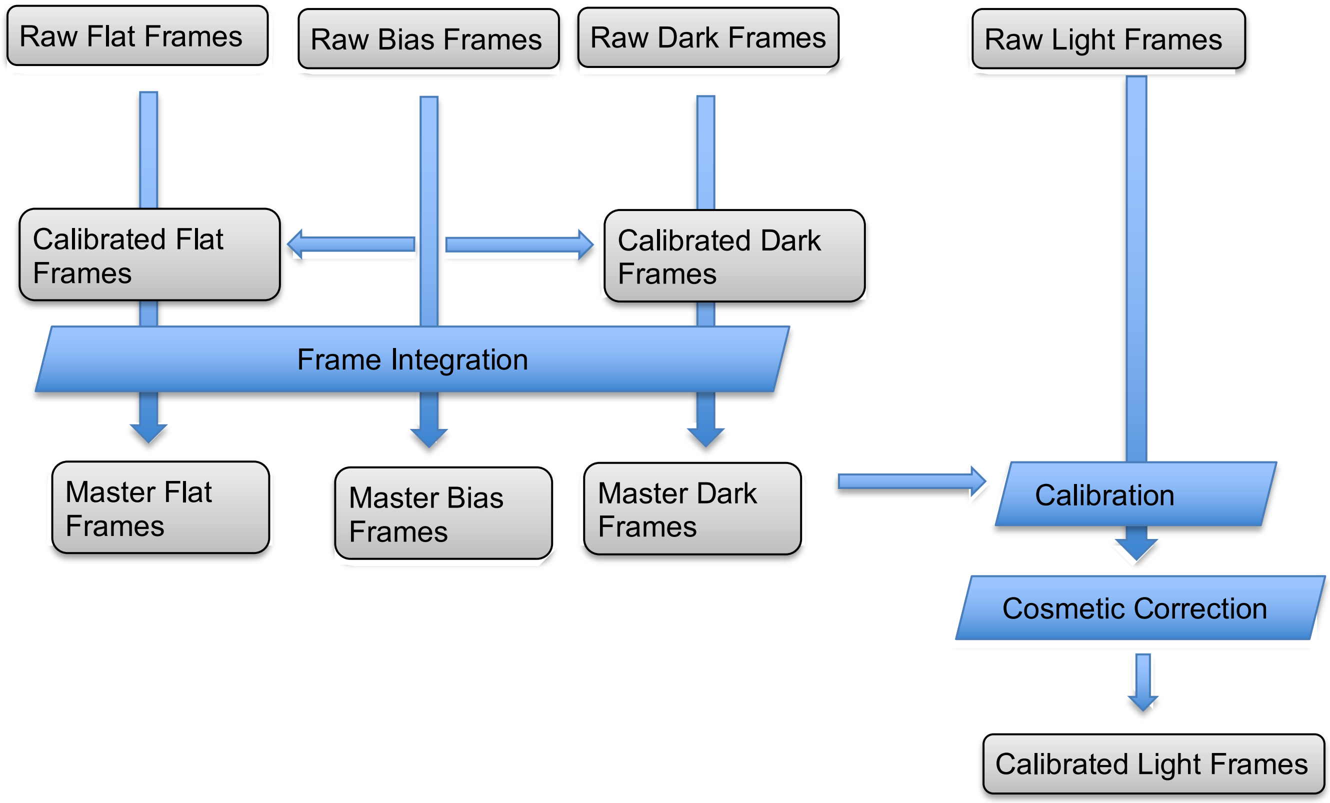
Table of Contents

Deep Sky Image Processing Workflow

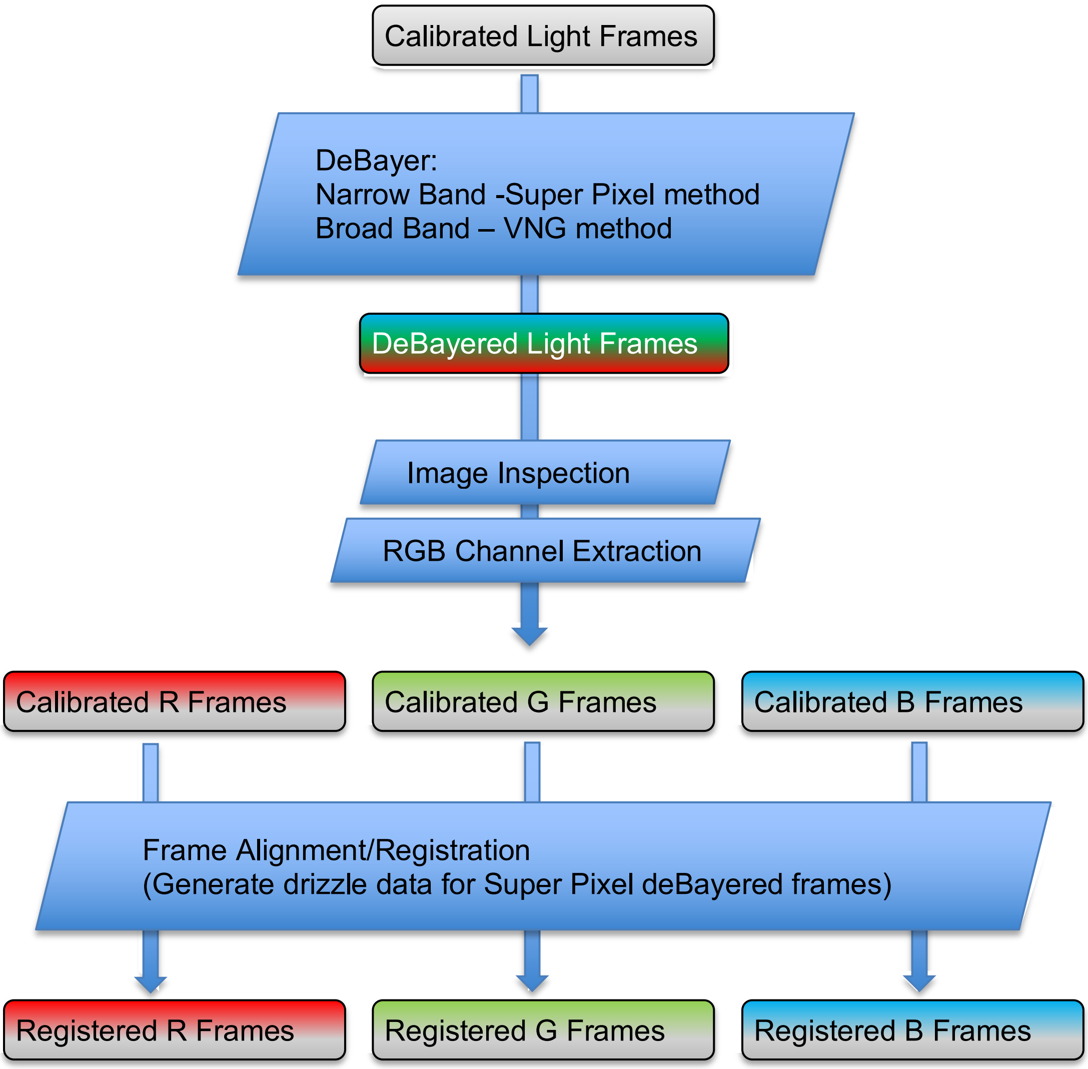
Calibration



OR



DeBayer & Registration



Calibrated Light Frames

DeBayer:
Narrow Band -Super Pixel method
Broad Band – VNG method

DeBayered Light Frames

Image Inspection

RGB Channel Extraction

Calibrated R Frames

Calibrated G Frames

Calibrated B Frames

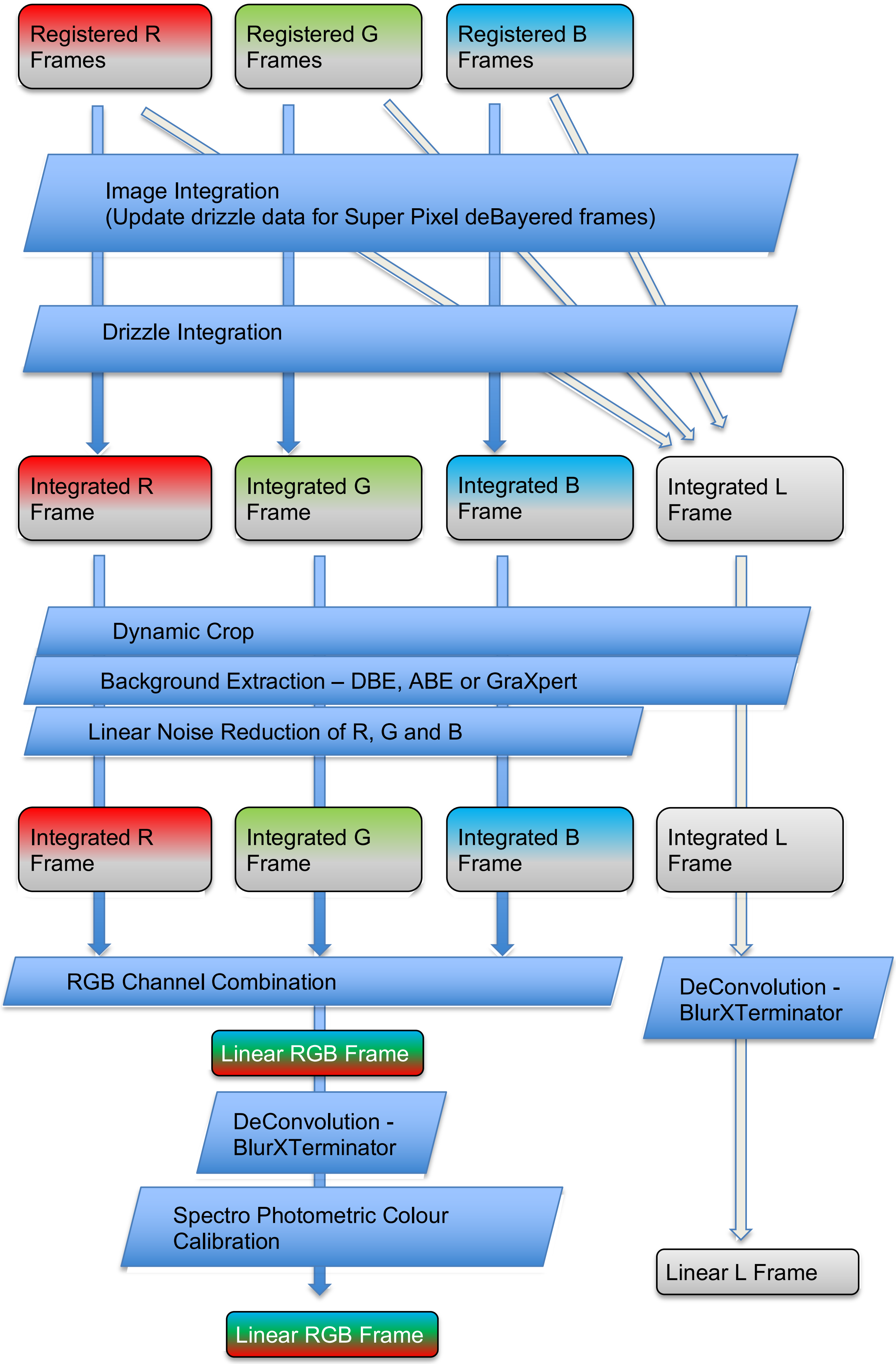
Frame Alignment/Registration
(Generate drizzle data for Super Pixel deBayered frames)

Registered R Frames

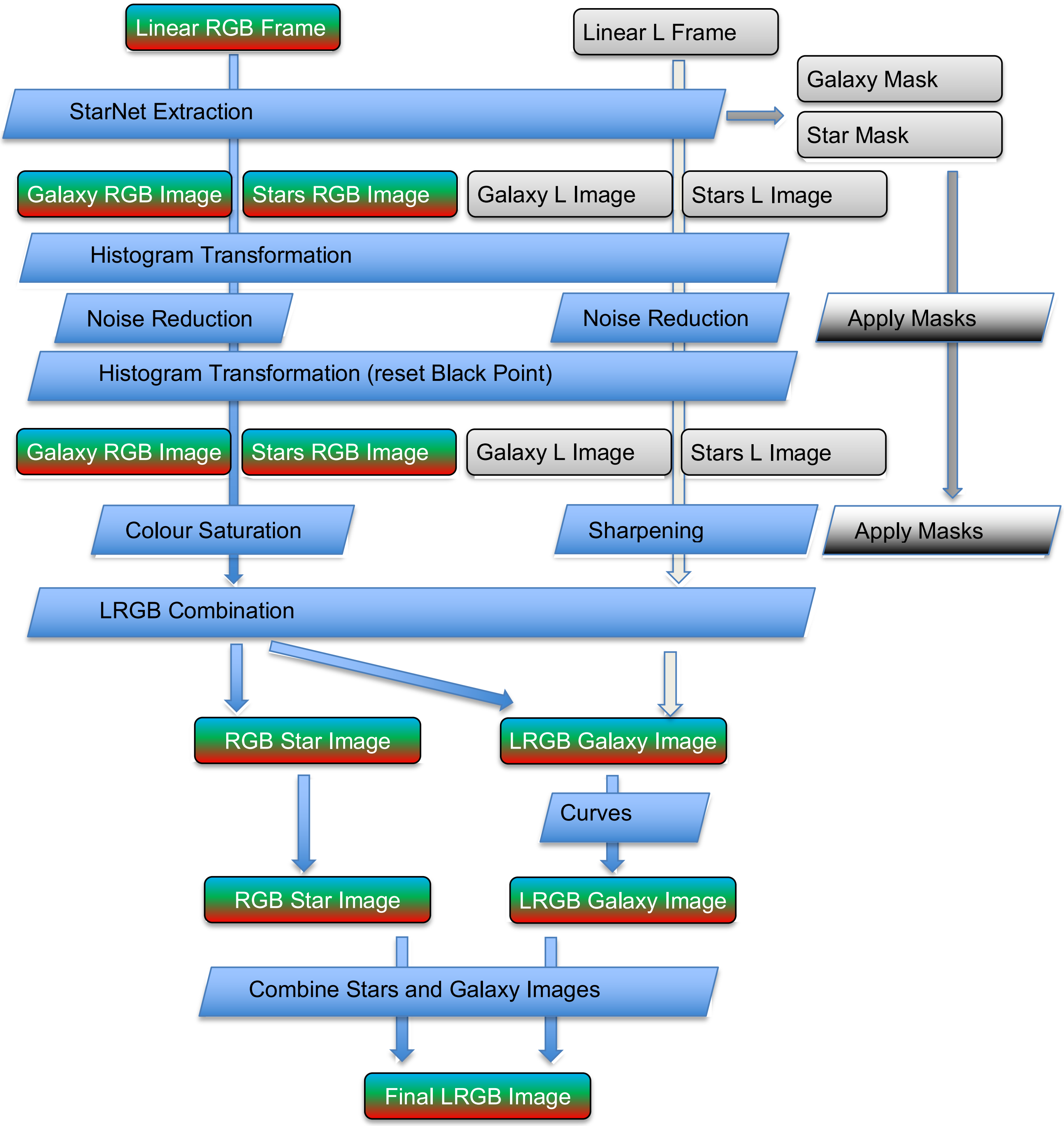
Registered G Frames

Registered B Frames

Image Integration & Initial Linear Pre-Processing

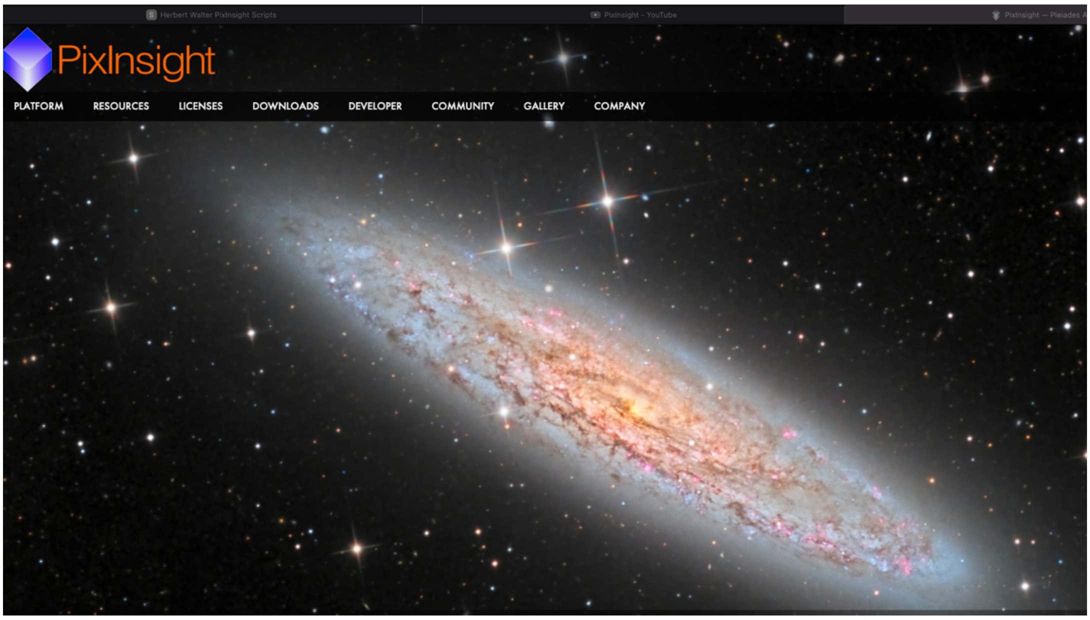


Processing


Linear RGB Frame
Linear L Frame
Galaxy Mask
Star Mask
StarNet Extraction
Galaxy RGB Image
Stars RGB Image
Galaxy L Image
Stars L Image
Histogram Transformation
Noise Reduction
Noise Reduction
Apply Masks
Histogram Transformation (reset Black Point)
Galaxy RGB Image
Stars RGB Image
Galaxy L Image
Stars L Image
Colour Saturation
Sharpening
Apply Masks
LRGB Combination
RGB Star Image
LRGB Galaxy Image
Curves
RGB Star Image
LRGB Galaxy Image
Combine Stars and Galaxy Images
Final LRGB Image

Introduction to PixInsight

https://pixinsight.com



PixInsight is an image processing platform specialized in astrophotography, available natively for FreeBSD, Linux, Mac OS X and Windows operating systems. Pleiades Astrophoto is a software development company based in Spain. The software requires a one-time-only purchased licence.

There are a range of on-line resources to help beginners get to grips with PixInsight. https://www.youtube.com/playlist?list=PL0Nr1Pazdc5-itJshDQVx98a3Snq46Tik is an excellent set of introductory videos. https://pixinsight.com/resources/ provides advanced user resources.

The PixInsight core application provides the infrastructure on top of which external modules can implement processes, image file formats, and their associated user interfaces. The PixInsight Class Library (PCL) is a C++ development framework to build PixInsight modules.

PixInsight modules are special shared libraries (.so files on FreeBSD and Linux; .dylib under OS X; .dll files on Windows) that communicate with the PixInsight core application through a high-level API provided by PCL. Along with a core communication API, PCL includes a comprehensive set of image processing algorithms, ranging from geometrical transformations to multiscale analysis algorithms, most of them available as multithreaded parallel implementations.

Installing Third-Party Scripts

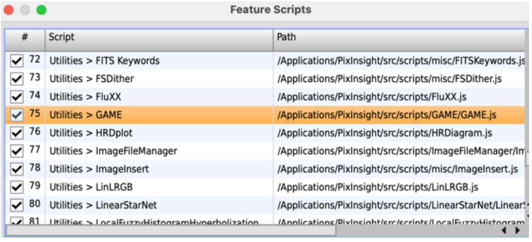
Third-party scripts can be downloaded from a number of authors. Most of these third party scripts are provided free of charge. The script needs to be copied to <Applications><PixInsight><src><scripts>

A number of useful scripts can be obtained at 'Astrophotography by Herbert Walter' http://www.skypixels.at/pixinsight_scripts.html#game

GAME Script

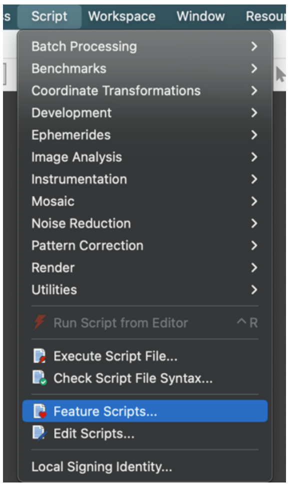
One example is the GAME script. GAME is an interactive graphical Editor for the creation of elliptical and multipoint areas used for masking galaxies.

Once the script is downloaded into the PixInsight <scripts> folder, use the <Scripts><Feature Scripts> window to enable the script.



Feature Scripts

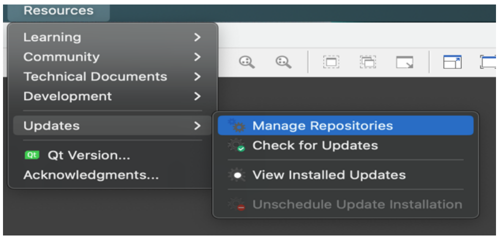
#	Script	Path
☑ 72	Utilities > FITS Keywords	/Applications/PixInsight/src/scripts/misc/FITSKeywords.js
☑ 73	Utilities > FSDither	/Applications/PixInsight/src/scripts/misc/FSDither.js
☑ 74	Utilities > FluXX	/Applications/PixInsight/src/scripts/FluXX.js
☑ 75	Utilities > GAME	/Applications/PixInsight/src/scripts/GAME/GAME.js
☑ 76	Utilities > HRDplot	/Applications/PixInsight/src/scripts/HRDiagram.js
☑ 77	Utilities > ImageFileManager	/Applications/PixInsight/src/scripts/ImageFileManager/Im
☑ 78	Utilities > ImageInsert	/Applications/PixInsight/src/scripts/misc/ImageInsert.js
☑ 79	Utilities > LinLRGB	/Applications/PixInsight/src/scripts/LinLRGB.js
☑ 80	Utilities > LinearStarNet	/Applications/PixInsight/src/scripts/LinearStarNet/Linear
☑ 81	Utilities > LocalFuzzyHistogramHyperbolization	/Applications/PixInsight/src/scripts/LocalFuzzyHistogram

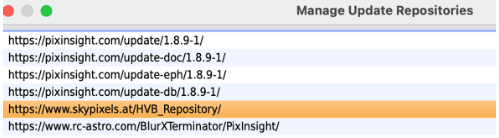


PSF Image Creator

To ensure that the third-party script is automatically updated with other PixInsight updates, use the <Resources><Updates><Manage Repositories> windows to add the repository address.

For Hartmut Borneman scripts, including PSF Image Creator, the address is https://www.skypixels.at/HVB_Repository/





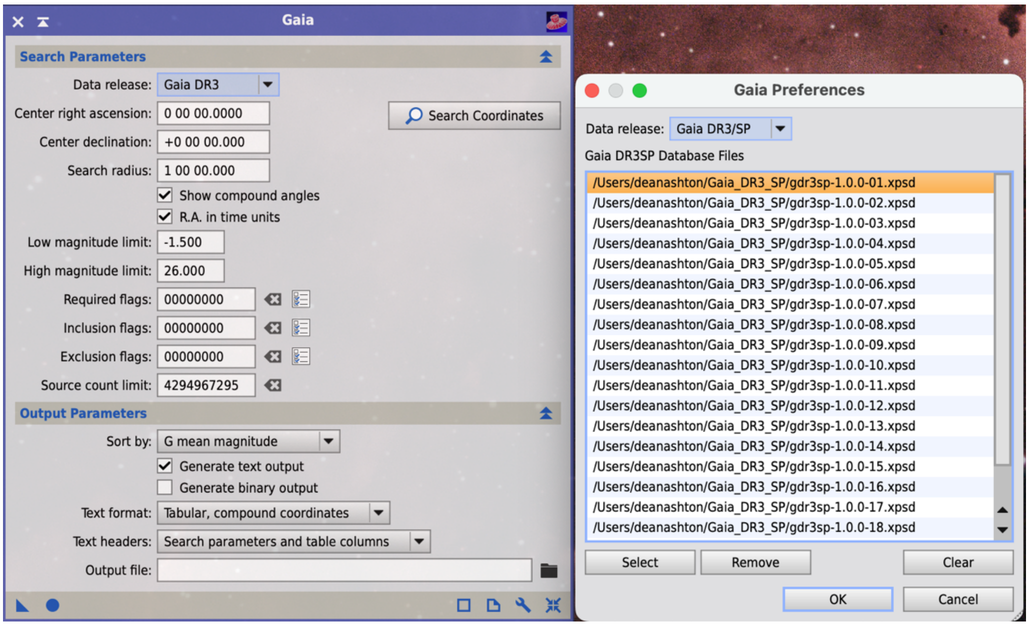
Manage Update Repositories

https://pixinsight.com/update/1.8.9-1/
https://pixinsight.com/update-doc/1.8.9-1/
https://pixinsight.com/update-eph/1.8.9-1/
https://pixinsight.com/update-db/1.8.9-1/
https://www.skypixels.at/HVB_Repository/
https://www.rc-astro.com/BlurXTerminator/PixInsight/

Installing the Gaia DR3 Dataset for Image Plate Solver Script and Spectro-Photometric Colour Calibration

https://www.cosmos.esa.int/web/gaia/dr3 Gaia Data Release 3 (Gaia DR3) was released on June 13, 2022. This Gaia release includes mean BP/RP spectra for more than 219 million point sources. This is the first massive, accurate spectrophotometric catalogue in the history of astronomy, with unprecedented homogeneity and consistency thanks to the fact that it is based on measurements made from space, without the varying conditions of the atmosphere. Thanks to this vast database, the PixInsight Spectro-Photometric Colour Calibration (SPCC) process implements colour calibration by reproducing the exact characteristics of the equipment we use to acquire our images: the same filter transmission curves, the same sensor quantum efficiency curves, and the precise spectra of the selected white reference.

A full description of the download of the GDR3 data and its use in the PixInsight Spectrophotometric Colour Calibration process, is provided at https://pixinsight.com/doc/docs/SPCC/SPCC.html Downloading the GDR3 data onto a local PC avoids the need for an internet connection when carrying out the SPCC process. The GDR3 data set is required for the Image Plate Solver Script. The GDR3/SP data set is required for the Spectrophotometric Colour Calibration Process.



Gaia

Search Parameters

Data release: Gaia DR3
Center right ascension: 0 00 00.0000
Center declination: +0 00 00.000
Search radius: 1 00 00.000
☑ Show compound angles
☑ R.A. in time units
Low magnitude limit: -1.500
High magnitude limit: 26.000
Required flags: 00000000
Inclusion flags: 00000000
Exclusion flags: 00000000
Source count limit: 4294967295

Output Parameters

Sort by: G mean magnitude
☑ Generate text output
□ Generate binary output
Text format: Tabular, compound coordinates
Text headers: Search parameters and table columns
Output file:

Gaia Preferences

Data release: Gaia DR3/SP

Gaia DR3SP Database Files

/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-01.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-02.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-03.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-04.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-05.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-06.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-07.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-08.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-09.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-10.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-11.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-12.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-13.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-14.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-15.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-16.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-17.xpsd
/Users/deanashton/Gaia_DR3_SP/gdr3sp-1.0.0-18.xpsd

Once the relevant data set is downloaded, the Gaia Process is used to set the location of the data set for use by PI processes and scripts. Select the ‘spanner’ icon on the Gaia Process to open the Gaia Preferences window. Select the relevant data sets and select OK.

RC Astro BlurXTerminator

BlurXTerminator https://www.rc-astro.com/software/bxt/ is an AI-powered deconvolution tool designed specifically for astronomical images. It is available as a plug-in process module for PixInsight only. BlurXTerminator requires a one-time purchase from the RC Astro website.

GraXpert

GraXpert https://www.graxpert.com is a freely available application that is designed to remove light gradient artefacts from Astro-photographs. It can be used as a stand-alone application or as a script add-on ion PixInsight.

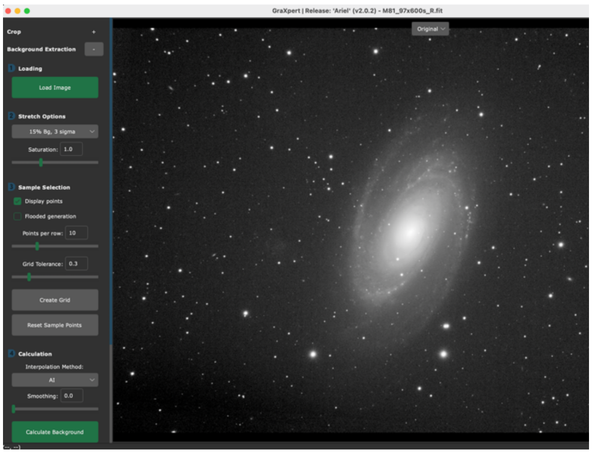
To use Graxpert within PI first download the application from https://www.graxpert.com and run it as a stand-alone application.

Note that as a stand-alone application GraXpert works on 16 or 32-bit FITS, TIF, PNG or JPG files.

For the 'Interpolation Method' select AI and download the AI files.

Note that there is no need to select sky sample points in AI mode.

Now switch off GraXpert.



15% Bg, 3 sigma

Saturation: 1.0

☑ Display points

□ Flooded generation

Points per row: 10

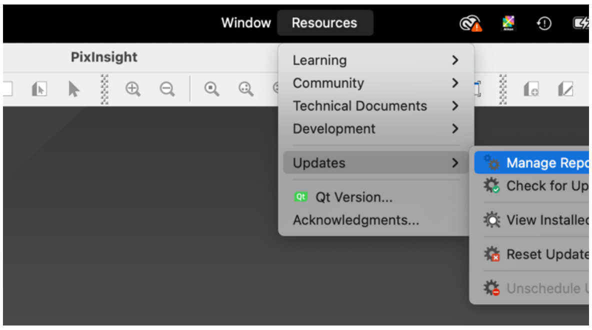
Grid Tolerance: 0.3

Interpolation Method: AI

Smoothing: 0.0

In PixInsight select **<Resources><Manage Repositories>**

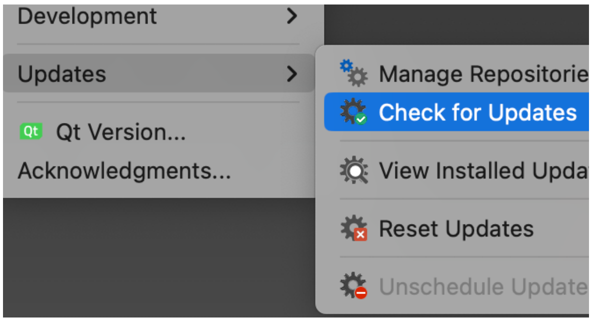
Add https://www.ideviceapps.de/PixInsight/Utilities/ to the repository list.



Now select **<Resources><Check for Updates>**

The repository should now update with a new collection of tools under **<Scripts><Toolbox>**.

The **GraXpert** script if found in the <Toolbox>.



If the installed Toolbox scripts do noyt appear in the scripts drop-down menu list, it may be because the 'feature scripts' is pointing at the wrong directory. If this is the case:

1. Select <Scripts><Feature Scripts>
2. Click on the <Add> button.
3. Navigate to /Applications?PixInsight/src/scripts/
4. This should scan and update the folder. The toolbox should now appear in the <Scripts> drop-down menu.

Generalised Hyperbolic Stretch

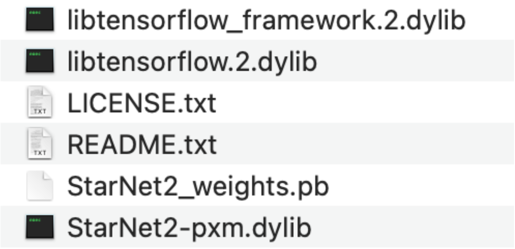
- From within Pixinsight go to: Resources>Updates>Manage Repositories
- In the Manage Update Repositories dialog, click the Add button
- In the URL box type the GHS repository information as follows: https://www.ghsastro.co.uk/updates/
- In the Add Pixinsight Repository dialog, click the OK button

- In the Manage Update Repositories dialog, click the OK button

Installing StarNet on a Mac-Silicon PC

PixInsight includes a StarNet https://www.starnetastro.com star-removal process as a ‘plugin’. At the time of writing this manual, StarNet2 was not working for MacBookPro A1/A2 chips, with the PI download. The process is reported to install automatically on Windows OS and older Mac chips. This is fixed A1/A2 chips by:

1. From the StarNet website, https://www.starnetastro.com download the PixInsight plugin for MacOs. This will provide the StarNet version 2 plugin files.

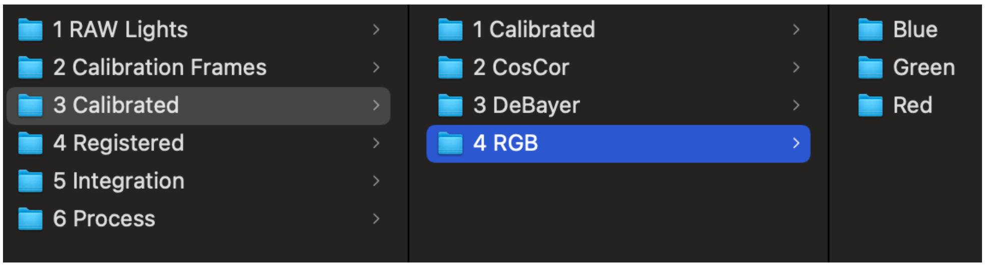


libtensorflow_framework.2.dylib
libtensorflow.2.dylib
LICENSE.txt
README.txt
StarNet2_weights.pb
StarNet2-pxm.dylib

2. Move the existing libtensorflow and libtensorflow_framework files from the PixInsight/Bin directory to a backup directory.
3. Copy the following files to the PixInsight/Bin directory:
   i. StarNet2_weights.pb
   ii. StarNet2-pxm.dylib
   iii. LICENCE
   iv. libtensorflow.2.dylib
   v. libtensorflow_framework.2.dylib
4. Run PixInsight
5. Go to <Processes><Modules><Install Modules> and select <Search>. StarNet2 should appear in the list. Select <Install>.
6. If StarNet2 does not appear in the process list, copy the libtensorflow.2.dylib and libtensorflow_framework.2.dylib files to /Application/Pixinsight?pixinsight.app/Contents/Frameworks/
7. Repeat go to <Processes><Modules><Install Modules> and select <Search>. StarNet2 should appear in the list. Select <Install>.
8. The StarNet2 process module should now be available to use.
9. It is likely that this issue will be resolved soon within PI and these additional steps will not be required.

Organised File Structure

Deep sky image calibration, integration and processing is multi-layered and can seem complex. To manage and simplify the process it is essential to create an organised directory structure for the process in advance.



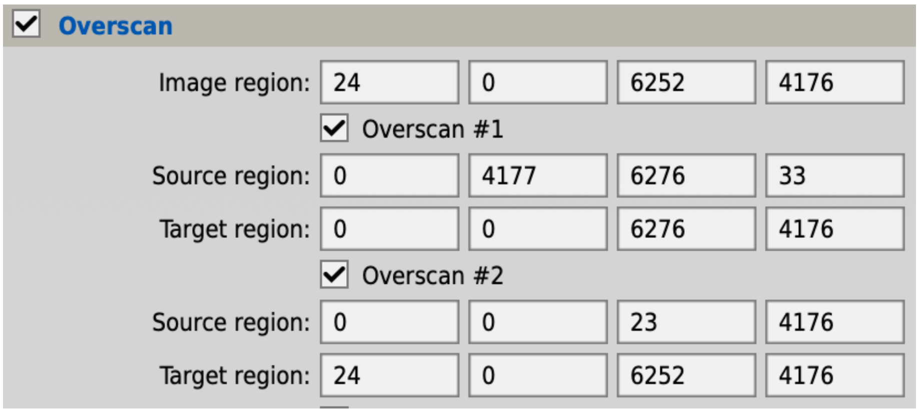
- 1 RAW Lights
- 2 Calibration Frames
- 3 Calibrated
  - 1 Calibrated
  - 2 CosCor
  - 3 DeBayer
  - 4 RGB
    - Blue
    - Green
    - Red
- 4 Registered
- 5 Integration
- 6 Process

Shown here is a recommended directory structure for saving image frames. The structure is ordered to mirror the sequence of steps in the calibration, integration and processing workflow.

Image Calibration

Without a Master Dark

Already described in this manual, is the use of the ‘overscan’ and ‘optic dark’ regions of the QHY268C CMOS sensor to calibrate each individual frame taken for both bias and dark noise.



☑ Overscan

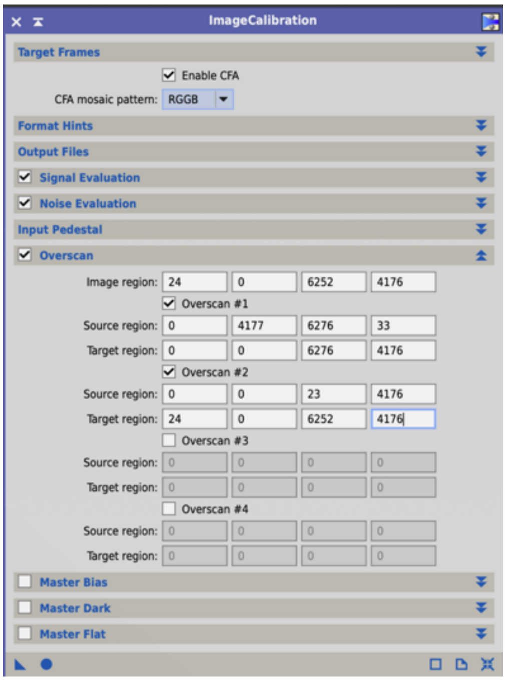
Image region:	24	0	6252	4176
	☑ Overscan #1			
Source region:	0	4177	6276	33
Target region:	0	0	6276	4176
	☑ Overscan #2			
Source region:	0	0	23	4176
Target region:	24	0	6252	4176

Creating a Master-Flat

The *Image Calibration Process* in PixInsight (PI) includes an overscan option in which the pixel co-ordinates for the final calibrated image region, the bias (read) noise overscan data and the dark (thermal) noise data are located on the camera sensor. The calibration process will subtract the mean bias and dark noise levels and crop of the overscan and optic dark areas to leave an individual calibrated frame.

Images are also calibrated for vignetting using a 'master flat frame'. At least 100 individual flat frames are captured using an Aurora Flat Field panel to evenly illuminate the cameras field of view.

The individual flat-fields are then calibrated for bias and dark current using the *Image Calibration Process*.



ImageCalibration

Target Frames

☑ Enable CFA

CFA mosaic pattern: RGGB

Format Hints

Output Files

☑ Signal Evaluation

☑ Noise Evaluation

Input Pedestal

☑ Overscan

Image region:	24	0	6252	4176
☑ Overscan #1				
Source region:	0	4177	6276	33
Target region:	0	0	6276	4176
☑ Overscan #2				
Source region:	0	0	23	4176
Target region:	24	0	6252	4176
□ Overscan #3				
Source region:	0	0	0	0
Target region:	0	0	0	0
□ Overscan #4				
Source region:	0	0	0	0
Target region:	0	0	0	0

□ Master Bias

□ Master Dark

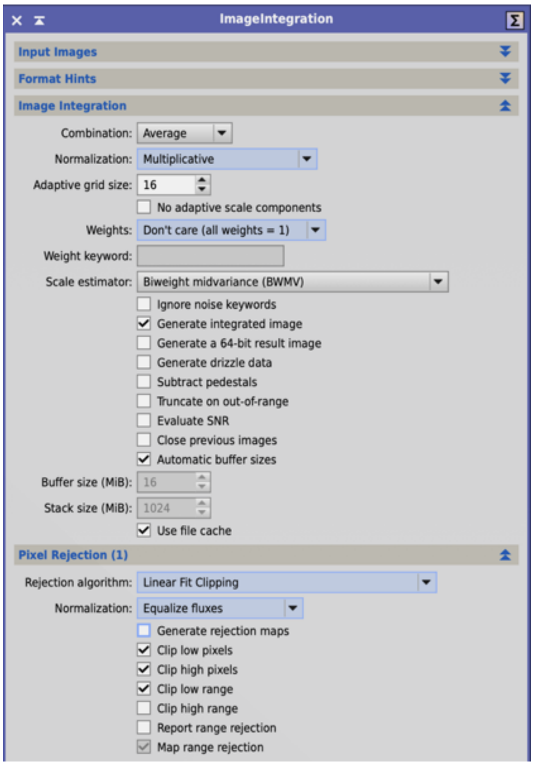
□ Master Flat

The individual calibrated flat-frames are now combined using the *Image Integration Process* to create a single Master-Flat Frame.

Some of the *Image Integration Process* settings are uniquely used for creating an integrated flat-frame:

- *Image Integration* Normalization: Multiplicative
- *Image Integration* Weights: Don't Care (all weights=1)
- *Image Integration* Evaluate SNR: Uncheck
- *Pixel Rejection (1)* Normalization: Equalize Fluxes

Once Created, the Master-Flat is saved in 16-bit FITS format to be used to calibrate the light-frames.



ImageIntegration

Input Images

Format Hints

Image Integration

Combination: Average

Normalization: Multiplicative

Adaptive grid size: 16

□ No adaptive scale components

Weights: Don't care (all weights = 1)

Weight keyword:

Scale estimator: Biweight midvariance (BWMV)

□ Ignore noise keywords

☑ Generate integrated image

□ Generate a 64-bit result image

□ Generate drizzle data

□ Subtract pedestals

□ Truncate on out-of-range

□ Evaluate SNR

□ Close previous images

☑ Automatic buffer sizes

Buffer size (MiB): 16

Stack size (MiB): 1024

☑ Use file cache

Pixel Rejection (1)

Rejection algorithm: Linear Fit Clipping

Normalization: Equalize fluxes

□ Generate rejection maps

☑ Clip low pixels

☑ Clip high pixels

☑ Clip low range

□ Clip high range

□ Report range rejection

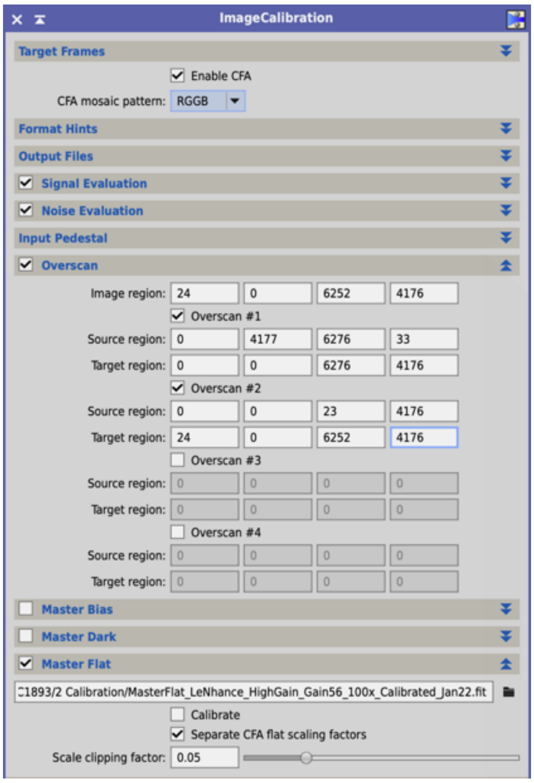
☑ Map range rejection

Calibration of Light-Frames

The individual light-frames are calibrated using the *Image Calibration Process.*

The overscan settings provide the bias and dark noise calibration. Loading the calibrated Master-Flat frame into the process provides the flat-field calibration.

Enable CFA to allow separate flat field calibration of the RGB channels in each individual image.



ImageCalibration

Target Frames

☑ Enable CFA

CFA mosaic pattern: RGGB

Format Hints

Output Files

☑ Signal Evaluation

☑ Noise Evaluation

Input Pedestal

☑ Overscan

Image region:	24	0	6252	4176
☑ Overscan #1				
Source region:	0	4177	6276	33
Target region:	0	0	6276	4176
☑ Overscan #2				
Source region:	0	0	23	4176
Target region:	24	0	6252	4176
□ Overscan #3				
Source region:	0	0	0	0
Target region:	0	0	0	0
□ Overscan #4				
Source region:	0	0	0	0
Target region:	0	0	0	0

□ Master Bias

□ Master Dark

☑ Master Flat

:1893/2 Calibration/MasterFlat_LeNhance_HighGain_Gain56_100x_Calibrated_Jan22.fit

□ Calibrate

☑ Separate CFA flat scaling factors

Scale clipping factor: 0.05

Calibration With a Master Dark

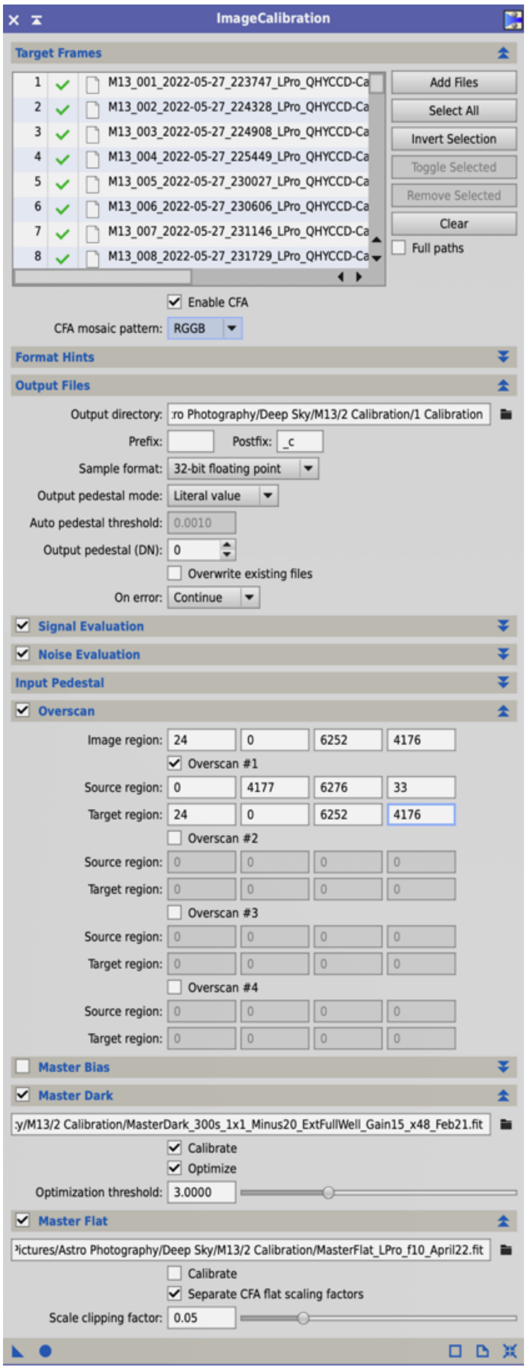
Image calibration without a master-dark frame will not correct for any subtle variations or banding of sensitivity they might exist in the CMOS sensor. Analysis of dark frames using the QHY268C's sensor demonstrates only the very slightest of banding. This may not normally be an issue. However, this can be compensated for by using a full Master-Dark frame in the calibration.

Master-Dark frames are created by taking a set of dark frames ta the same TEC temperature setting, exposure time and gain setting as the light frames to be calibrated. Dark frames are integrated to create a Master-Dark using the same settings as for light frames but with the 'Evaluate SNR' option unchecked.

During image calibration, the overscan settings are adjusted to only apply the 'bias' data to the light frames.

If the individual dark frames were not calibrated before integration to a master-dark, the calibration can also be checked for the master-dark at this stage.

All other aspects of the calibration are carried out as it is for 'calibration without a master-dark.



Target Frames

1 ✓ M13_001_2022-05-27_223747_LPro_QHYCCD-Ca
2 ✓ M13_002_2022-05-27_224328_LPro_QHYCCD-Ca
3 ✓ M13_003_2022-05-27_224908_LPro_QHYCCD-Ca
4 ✓ M13_004_2022-05-27_225449_LPro_QHYCCD-Ca
5 ✓ M13_005_2022-05-27_230027_LPro_QHYCCD-Ca
6 ✓ M13_006_2022-05-27_230606_LPro_QHYCCD-Ca
7 ✓ M13_007_2022-05-27_231146_LPro_QHYCCD-Ca
8 ✓ M13_008_2022-05-27_231729_LPro_QHYCCD-Ca

□ Full paths

☑ Enable CFA

CFA mosaic pattern: RGGB

Format Hints

Output Files

Output directory: :ro Photography/Deep Sky/M13/2 Calibration/1 Calibration

Prefix: Postfix: _c

Sample format: 32-bit floating point

Output pedestal mode: Literal value

Auto pedestal threshold: 0.0010

Output pedestal (DN): 0

□ Overwrite existing files

On error: Continue

☑ Signal Evaluation

☑ Noise Evaluation

Input Pedestal

☑ Overscan

Image region:	24	0	6252	4176
☑ Overscan #1				
Source region:	0	4177	6276	33
Target region:	24	0	6252	4176
□ Overscan #2				
Source region:	0	0	0	0
Target region:	0	0	0	0
□ Overscan #3				
Source region:	0	0	0	0
Target region:	0	0	0	0
□ Overscan #4				
Source region:	0	0	0	0
Target region:	0	0	0	0

□ Master Bias

☑ Master Dark

:y/M13/2 Calibration/MasterDark_300s_1x1_Minus20_ExtFullWell_Gain15_x48_Feb21.fit

☑ Calibrate

☑ Optimize

Optimization threshold: 3.0000

☑ Master Flat

ictures/Astro Photography/Deep Sky/M13/2 Calibration/MasterFlat_LPro_f10_April22.fit

□ Calibrate

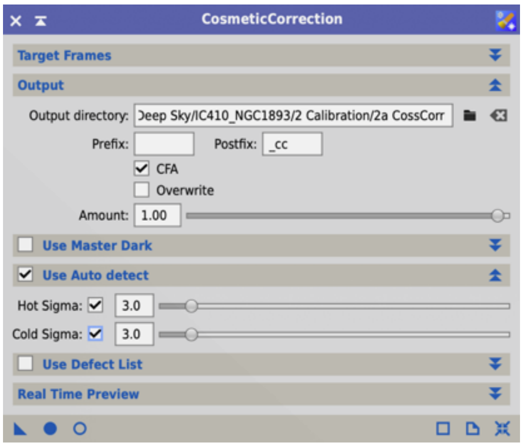
☑ Separate CFA flat scaling factors

Scale clipping factor: 0.05

Cosmetic Correction

The final stage in the image calibration workflow is the *Cosmetic Correction Process*. Cosmetic Correction in PI is a tool to detect 'hot' or 'cold' pixels in each frame and replace the value of the pixel with the average value of the surrounding pixels.

Although image dithering and stacking will remove random 'hot' or 'cold' pixels, the cosmetic Correction process does improve the overall quality of the individual light-frames as well as the final stacked integrated image.



CosmeticCorrection

Target Frames

Output

Output directory: Deep Sky/IC410_NGC1893/2 Calibration/2a CossCorr

Prefix: Postfix: _cc

☑ CFA

□ Overwrite

Amount: 1.00

□ Use Master Dark

☑ Use Auto detect

Hot Sigma: ☑ 3.0

Cold Sigma: ☑ 3.0

□ Use Defect List

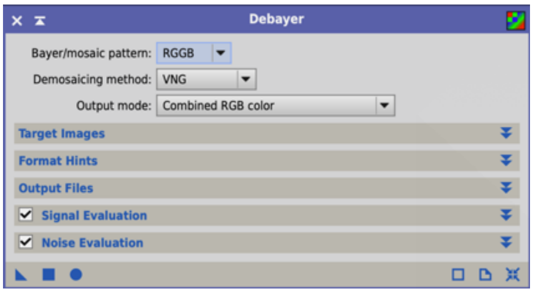
Real Time Preview

Image Debayer and Registration

RGB Image Debayer

The QHY268C camera has a RGGB (Red, green, green, blue) ordered Bayer matrix of pixels. The light-frames are captured as RAW monochrome images. The *Debayer process* assigns the correct colour to each pixels data to restore the colour information in each image.

For RGB images using the Optolong LPro filter, the default 'Demosaicing method' VNG (Variable Number of Gradients) is always used.



Debayer

Bayer/mosaic pattern: RGGB

Demosaicing method: VNG

Output mode: Combined RGB color

Target Images

Format Hints

Output Files

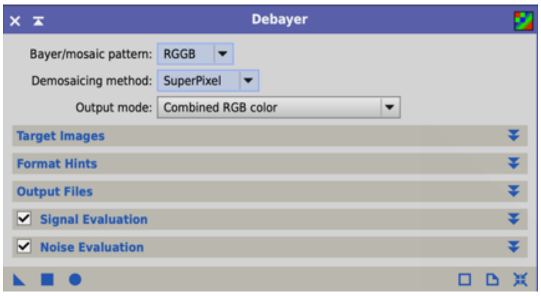
☑ Signal Evaluation

☑ Noise Evaluation

Narrow Band Image Debayer

For narrow-band images, taken using the Optolong LeNhance filter, for example, the 'Demosaicing method' SuperPixel may give better results.

The advantage of the SuperPixel method is that it only uses the signal in each RGB channel associated with the corresponding colour sensitive value pixel. This gives a pure signal value for the Debayered signal value that does not include the noise associated with the empty pixels for that RGB channel.



Debayer

Bayer/mosaic pattern: RGGB

Demosaicing method: SuperPixel

Output mode: Combined RGB color

Target Images

Format Hints

Output Files

☑ Signal Evaluation

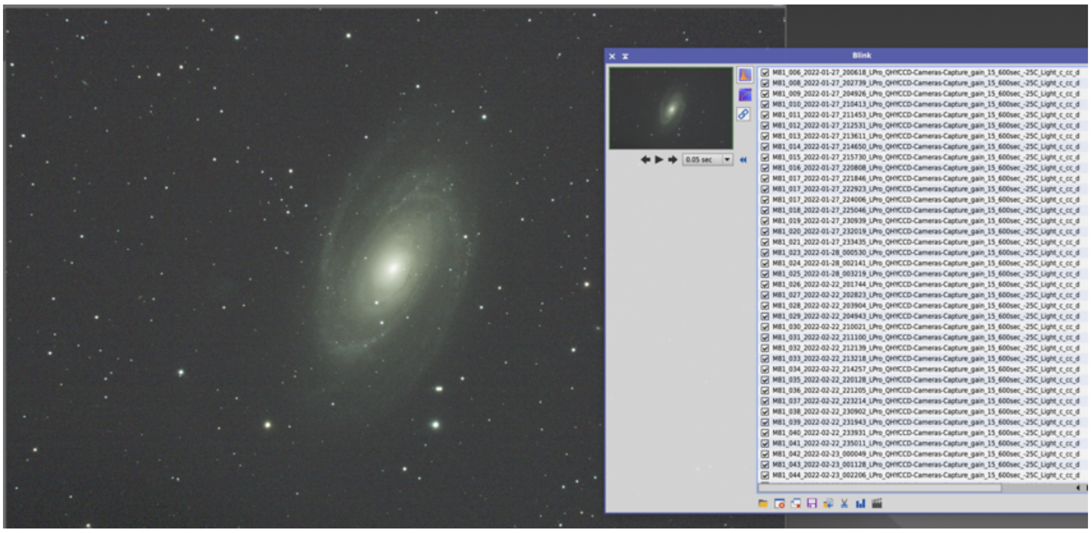
☑ Noise Evaluation

The resulting image has used 2x2 (4) pixels to provide RGB channels. A consequence of this is that the image produced is a quarter of the size of the original image.

Following the Debayer process, we have a complete set of light-frames that have been calibrated and Debayered. Our calibrated images are now ready for inspection and analysis.

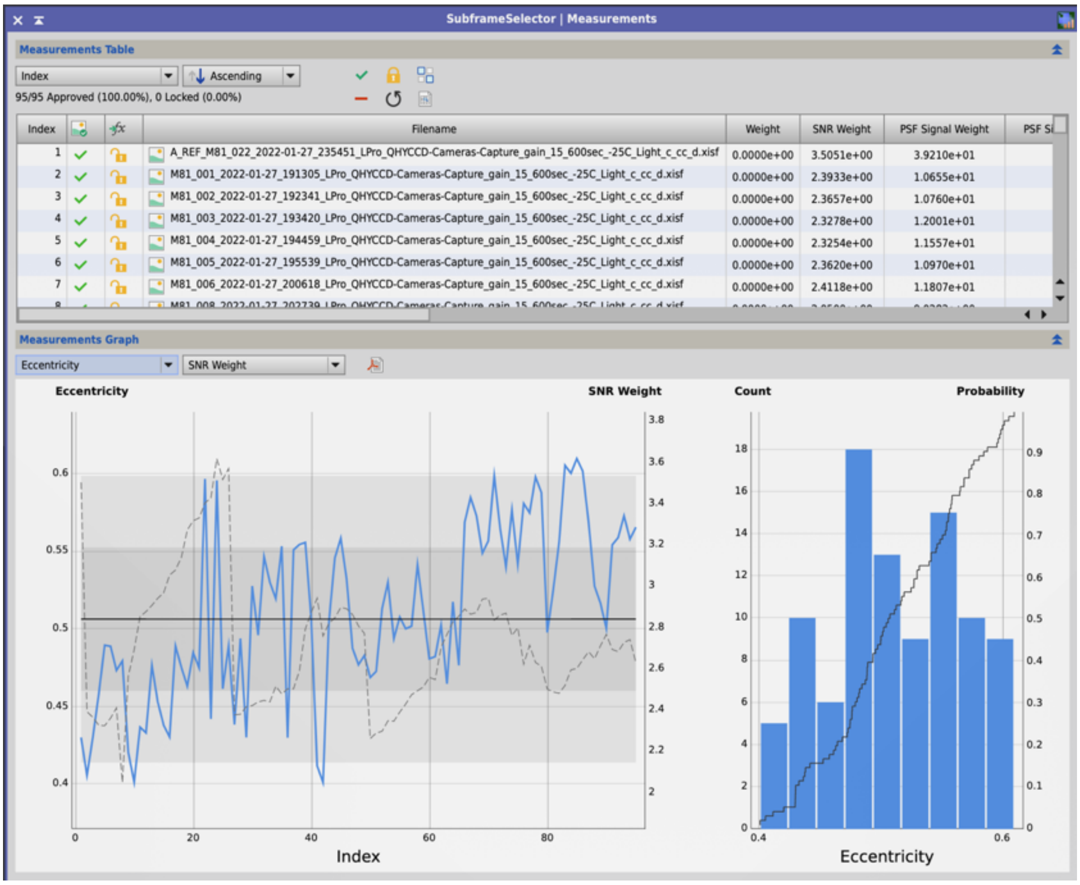
Image Inspection and Analysis

The PI *Blink Process* is used to visually inspect the quality of each calibrated and debayered light frame. Any poor quality light-frames are removed at this stage.



```
☑ M81_006_2022-01-27_200618_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_008_2022-01-27_202739_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_009_2022-01-27_204926_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_010_2022-01-27_210413_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_011_2022-01-27_211453_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_012_2022-01-27_212531_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_013_2022-01-27_213611_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_014_2022-01-27_214650_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_015_2022-01-27_215730_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_016_2022-01-27_220808_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_017_2022-01-27_221846_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_017_2022-01-27_222923_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_017_2022-01-27_224006_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_018_2022-01-27_225046_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_019_2022-01-27_230939_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_020_2022-01-27_232019_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_021_2022-01-27_233435_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_023_2022-01-28_000530_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_024_2022-01-28_002141_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_025_2022-01-28_003219_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_026_2022-02-22_201744_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_027_2022-02-22_202823_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_028_2022-02-22_203904_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_029_2022-02-22_204943_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_030_2022-02-22_210021_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_031_2022-02-22_211100_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_032_2022-02-22_212139_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_033_2022-02-22_213218_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_034_2022-02-22_214257_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_035_2022-02-22_220128_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_036_2022-02-22_221205_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_037_2022-02-22_223214_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_038_2022-02-22_230902_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_039_2022-02-22_231943_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_040_2022-02-22_233931_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_041_2022-02-22_235011_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_042_2022-02-23_000049_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_043_2022-02-23_001128_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
☑ M81_044_2022-02-23_002206_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
```

The *Subframe Selector Process* provides an analysis of each individual calibrated light frame.



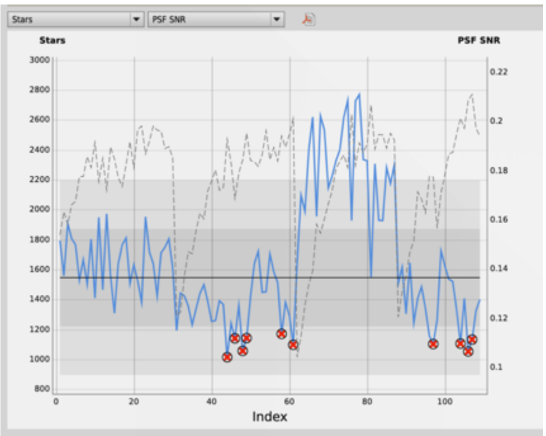
95/95 Approved (100.00%), 0 Locked (0.00%)

Index			Filename	Weight	SNR Weight	PSF Signal Weight	PSF Si
1	✓	🔓	A_REF_M81_022_2022-01-27_235451_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	3.5051e+00	3.9210e+01	
2	✓	🔓	M81_001_2022-01-27_191305_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	2.3933e+00	1.0655e+01	
3	✓	🔓	M81_002_2022-01-27_192341_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	2.3657e+00	1.0760e+01	
4	✓	🔓	M81_003_2022-01-27_193420_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	2.3278e+00	1.2001e+01	
5	✓	🔓	M81_004_2022-01-27_194459_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	2.3254e+00	1.1557e+01	
6	✓	🔓	M81_005_2022-01-27_195539_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	2.3620e+00	1.0970e+01	
7	✓	🔓	M81_006_2022-01-27_200618_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	0.0000e+00	2.4118e+00	1.1807e+01	
8			M81_008_2022-01-27_202739_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d.xisf	[illegible]	[illegible]	[illegible]	

The sub-frame selector process analysis is used to compare each individual light frame for:

- PSF SNR (signal to noise ratio) weight – high is best
- Number of STARS – high is best
- FWHM full width half maximum star size) - low is best
- Star eccentricity – low is best.

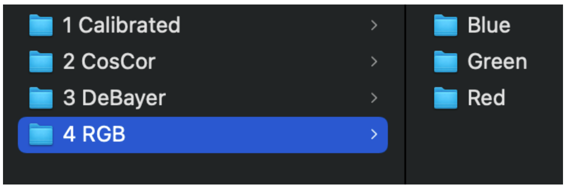
From this analysis poor quality frames can be identified for rejection. These will be frames with low SNR and low star count.



From this analysis, a **reference frame** is also identified that optimises the highest SNR weighting with the lowest FWHM and greatest number of stars identified.

The file name of the best quality reference name is appended with A_REF_ so that it is easilly identified for the *image registration* and *image integration* process to follow.

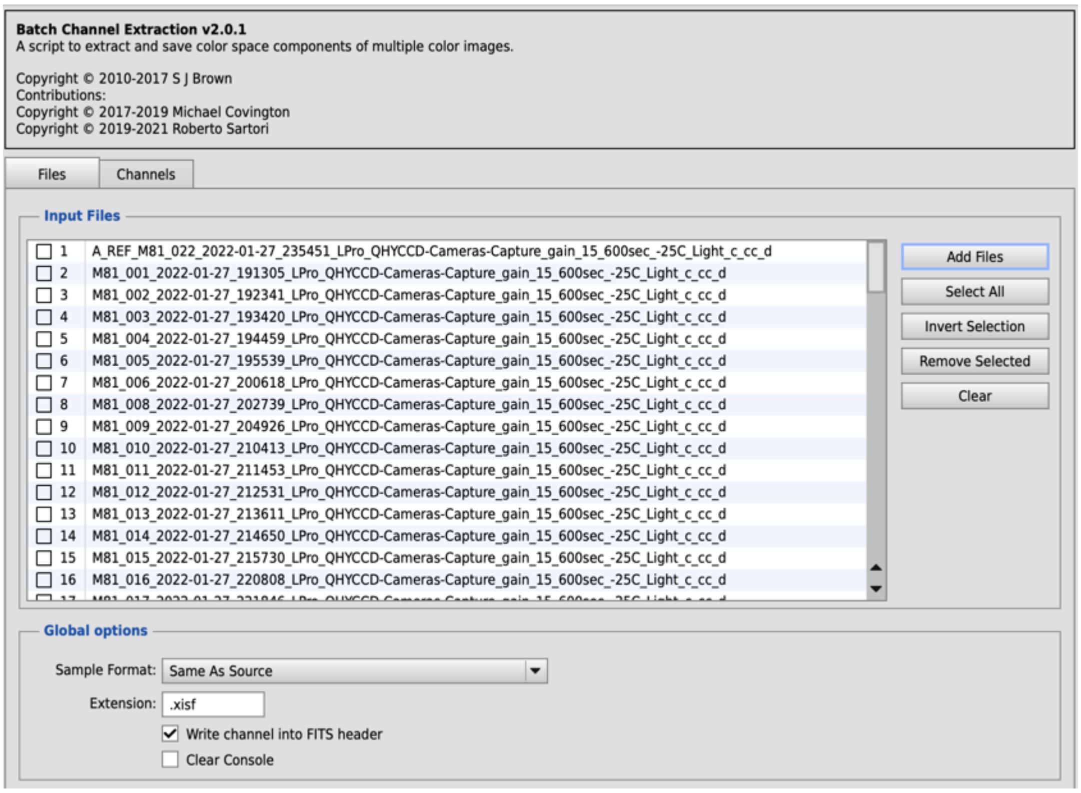
RGB Channel Extraction



The calibrated and debayered light-frames are now 'channel extracted' to separate the RGB components as separate images. It is the separate R, G and B channel images that will then be registered or aligned. It is important that directories have been prepared before this process to save the separate R, G and B channel images and following that the separate registered/aligned R, G and B channel images.

RGB channel extract is carried out using the PI *Batch Channel Extraction* script.

- On the <Files> tab of the script, load all of the calibrated and debayered images.
- On the <Channels> tab of the script, firsty select <RGB-R>, <RGB-G> and <RGB-B> options. Then enter the destination directory for the images extracted for each channel to be saved.



Batch Channel Extraction v2.0.1
A script to extract and save color space components of multiple color images.

Copyright © 2010-2017 S J Brown
Contributions:
Copyright © 2017-2019 Michael Covington
Copyright © 2019-2021 Roberto Sartori

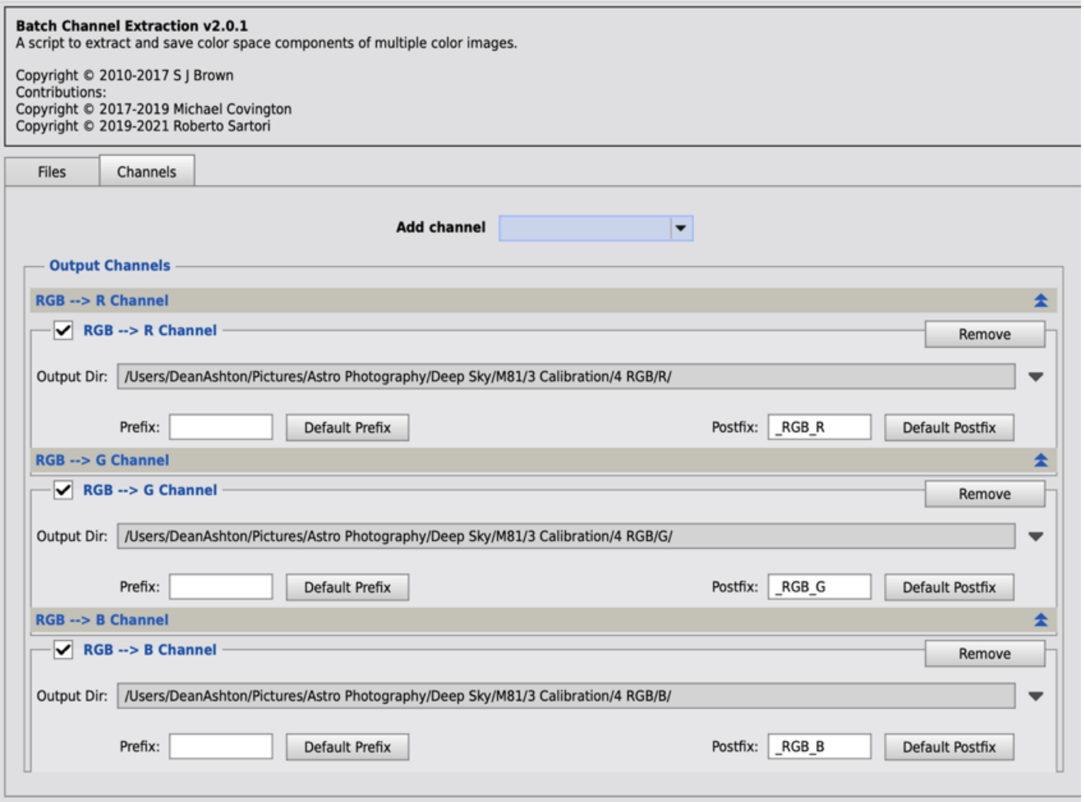
- □ 1 A_REF_M81_022_2022-01-27_235451_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 2 M81_001_2022-01-27_191305_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 3 M81_002_2022-01-27_192341_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 4 M81_003_2022-01-27_193420_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 5 M81_004_2022-01-27_194459_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 6 M81_005_2022-01-27_195539_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 7 M81_006_2022-01-27_200618_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 8 M81_008_2022-01-27_202739_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 9 M81_009_2022-01-27_204926_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 10 M81_010_2022-01-27_210413_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 11 M81_011_2022-01-27_211453_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 12 M81_012_2022-01-27_212531_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 13 M81_013_2022-01-27_213611_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 14 M81_014_2022-01-27_214650_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 15 M81_015_2022-01-27_215730_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d
- □ 16 M81_016_2022-01-27_220808_LPro_QHYCCD-Cameras-Capture_gain_15_600sec_-25C_Light_c_cc_d

Sample Format: Same As Source

Extension: .xisf

☑ Write channel into FITS header

□ Clear Console



Batch Channel Extraction v2.0.1
A script to extract and save color space components of multiple color images.

Copyright © 2010-2017 S J Brown
Contributions:
Copyright © 2017-2019 Michael Covington
Copyright © 2019-2021 Roberto Sartori

Files | Channels

Add channel:

Output Channels

RGB --> R Channel

☑ RGB --> R Channel — Remove

Output Dir: /Users/DeanAshton/Pictures/Astro Photography/Deep Sky/M81/3 Calibration/4 RGB/R/

Prefix: (blank) — Default Prefix — Postfix: _RGB_R — Default Postfix

RGB --> G Channel

☑ RGB --> G Channel — Remove

Output Dir: /Users/DeanAshton/Pictures/Astro Photography/Deep Sky/M81/3 Calibration/4 RGB/G/

Prefix: (blank) — Default Prefix — Postfix: _RGB_G — Default Postfix

RGB --> B Channel

☑ RGB --> B Channel — Remove

Output Dir: /Users/DeanAshton/Pictures/Astro Photography/Deep Sky/M81/3 Calibration/4 RGB/B/

Prefix: (blank) — Default Prefix — Postfix: _RGB_B — Default Postfix

Note that if the original images were taking using the Optolong LeNhance narrow band filter: the R channel will be the Ha data; the G channel will be the Oiii_G data; and the B channel will be the Oiii_B data as well as any Hb data.

Image Alignment or Registration

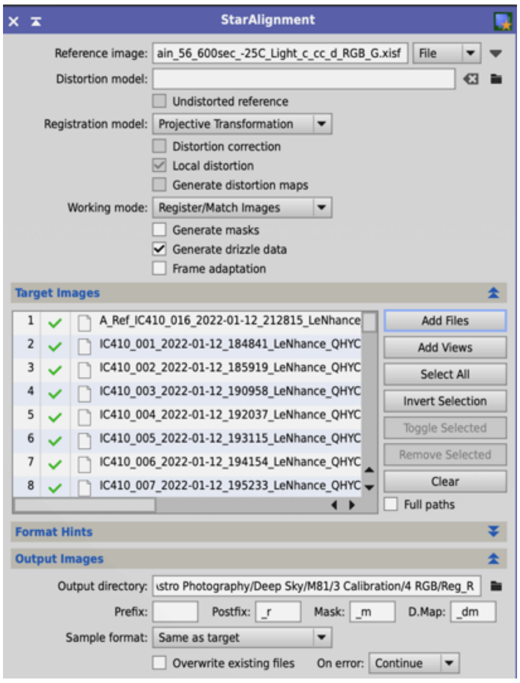
The individual R, G and B channel images are now all aligned to a single reference image using the *Star Alignment Process*.

The reference in mage will normally be the GREEN channel reference image identified at the *Sub-frame Selector Process* stage.

All R channel frame images are aligned to the G channel reference image and saved in the REG_R channel directory.

The process is repeated for the G channel images that are saved in the REG-G channel directory and the process repeated a third time for the B channel images.

The *Star Alignment Process* default settings are normally used.



StarAlignment

Reference image: ain_56_600sec_-25C_Light_c_cc_d_RGB_G.xisf — File

Distortion model:

☐ Undistorted reference

Registration model: Projective Transformation

☐ Distortion correction
☑ Local distortion
☐ Generate distortion maps

Working mode: Register/Match Images

☐ Generate masks
☑ Generate drizzle data
☐ Frame adaptation

Target Images

1	✓	A_Ref_IC410_016_2022-01-12_212815_LeNhance
2	✓	IC410_001_2022-01-12_184841_LeNhance_QHYC
3	✓	IC410_002_2022-01-12_185919_LeNhance_QHYC
4	✓	IC410_003_2022-01-12_190958_LeNhance_QHYC
5	✓	IC410_004_2022-01-12_192037_LeNhance_QHYC
6	✓	IC410_005_2022-01-12_193115_LeNhance_QHYC
7	✓	IC410_006_2022-01-12_194154_LeNhance_QHYC
8	✓	IC410_007_2022-01-12_195233_LeNhance_QHYC

Add Files — Add Views — Select All — Invert Selection — Toggle Selected — Remove Selected — Clear

☐ Full paths

Format Hints

Output Images

Output directory: ıstro Photography/Deep Sky/M81/3 Calibration/4 RGB/Reg_R

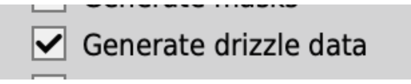
Prefix: (blank) Postfix: _r Mask: _m D.Map: _dm

Sample format: Same as target

☐ Overwrite existing files — On error: Continue

Preparing for Drizzle Integration

If the **VNG** method was used at the *debayer process* stage, then it is recommended that the <**Generate Drizzle Data**> option should be **checked** in the *Star Alignment Process*.



☑ Generate drizzle data

However, if the **SuprerPixel** nethod was used in the *debayer process*, then the image size of the debayered frames will have been reduced to one-quarter of the original image size. The image size **must** be restored to its original resolution through a *Drizzle Integration Process*. To prepare for a Drizzle Integration Process, the <**Generate Drizzle Data**> option should be **checked** in the *Star Alignment Process*.

If the <**Generate Drizzle Data**> option is **checked,** the Star Alignment Process will save a linked drizzle data file for each aligned light frame image as well as the aligned image.

The registered (aligned) image files and if appropriate the ‘drizzle’ data is saved to separate R, G and B directories.

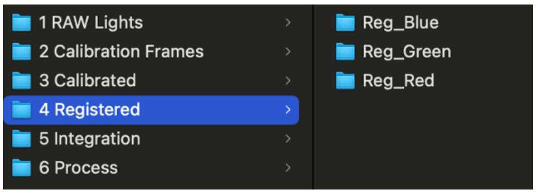
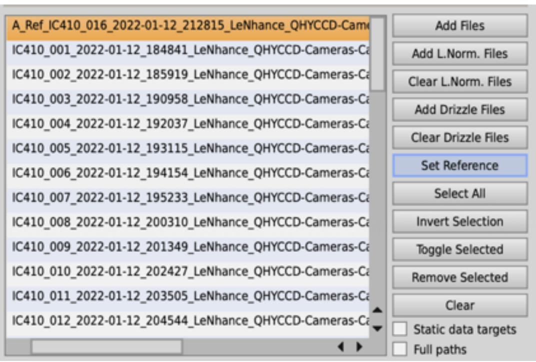


Image Integration

The individual aligned light frames are now stacked or integrated to create the single image frame.

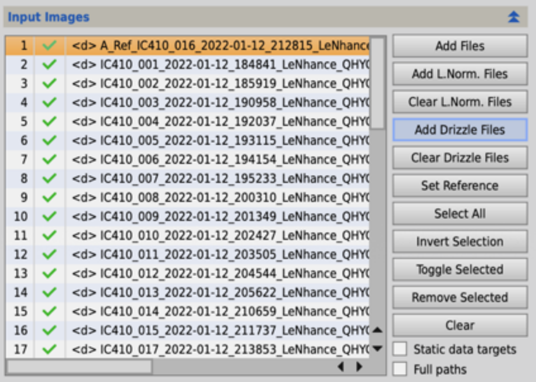
For example, all of the individual R channel images are loaded into the *Image Integration Process*. The reference image is identified, highlighted and the <**Set Reference**> button selected in the process.



A_Ref_IC410_016_2022-01-12_212815_LeNhance_QHYCCD-Came
IC410_001_2022-01-12_184841_LeNhance_QHYCCD-Cameras-Ca
IC410_002_2022-01-12_185919_LeNhance_QHYCCD-Cameras-Ca
IC410_003_2022-01-12_190958_LeNhance_QHYCCD-Cameras-Ca
IC410_004_2022-01-12_192037_LeNhance_QHYCCD-Cameras-Ca
IC410_005_2022-01-12_193115_LeNhance_QHYCCD-Cameras-Ca
IC410_006_2022-01-12_194154_LeNhance_QHYCCD-Cameras-Ca
IC410_007_2022-01-12_195233_LeNhance_QHYCCD-Cameras-Ca
IC410_008_2022-01-12_200310_LeNhance_QHYCCD-Cameras-Ca
IC410_009_2022-01-12_201349_LeNhance_QHYCCD-Cameras-Ca
IC410_010_2022-01-12_202427_LeNhance_QHYCCD-Cameras-Ca
IC410_011_2022-01-12_203505_LeNhance_QHYCCD-Cameras-Ca
IC410_012_2022-01-12_204544_LeNhance_QHYCCD-Cameras-Ca

If 'drizzle integration' is going to be used, then the <**Select Drizzle Files**> is selected and the drizzle files created during the star alignment process also loaded into the *Image Integration Process*.

The image integration process will update the drizzle files with the data required to complete the *drizzle integration process* at the next stage.



Input Images

1	✓	<d> A_Ref_IC410_016_2022-01-12_212815_LeNhance
2	✓	<d> IC410_001_2022-01-12_184841_LeNhance_QHYC
3	✓	<d> IC410_002_2022-01-12_185919_LeNhance_QHYC
4	✓	<d> IC410_003_2022-01-12_190958_LeNhance_QHYC
5	✓	<d> IC410_004_2022-01-12_192037_LeNhance_QHYC
6	✓	<d> IC410_005_2022-01-12_193115_LeNhance_QHYC
7	✓	<d> IC410_006_2022-01-12_194154_LeNhance_QHYC
8	✓	<d> IC410_007_2022-01-12_195233_LeNhance_QHYC
9	✓	<d> IC410_008_2022-01-12_200310_LeNhance_QHYC
10	✓	<d> IC410_009_2022-01-12_201349_LeNhance_QHYC
11	✓	<d> IC410_010_2022-01-12_202427_LeNhance_QHYC
12	✓	<d> IC410_011_2022-01-12_203505_LeNhance_QHYC
13	✓	<d> IC410_012_2022-01-12_204544_LeNhance_QHYC
14	✓	<d> IC410_013_2022-01-12_205622_LeNhance_QHYC
15	✓	<d> IC410_014_2022-01-12_210659_LeNhance_QHYC
16	✓	<d> IC410_015_2022-01-12_211737_LeNhance_QHYC
17	✓	<d> IC410_017_2022-01-12_213853_LeNhance_QHYC

The <Image Integration> and <Pixel Rejection (1)> tabs in the Image Integration Process are set.

- The <**Combination**> method is set to Average.
- The <**Normalization**> method is Additive Scaling
- Check <**Generate 64-bit result image**>
- Only if drizzle integration is going to be used, check <**Generate Drizzle Data**>
- The <Pixel Rejection Algorithm> depends on the number of images being integrated:
    - 3-7 images – percentile clipping
    - 7-10 images – averaged sigma clipping
    - 15-25 images – winsorized sigma clipping or Generalized Extreme Studentized Deviate (GESD)
    - 25+ images – Linear fit clipping or Generalized Extreme Studentized Deviate GESD

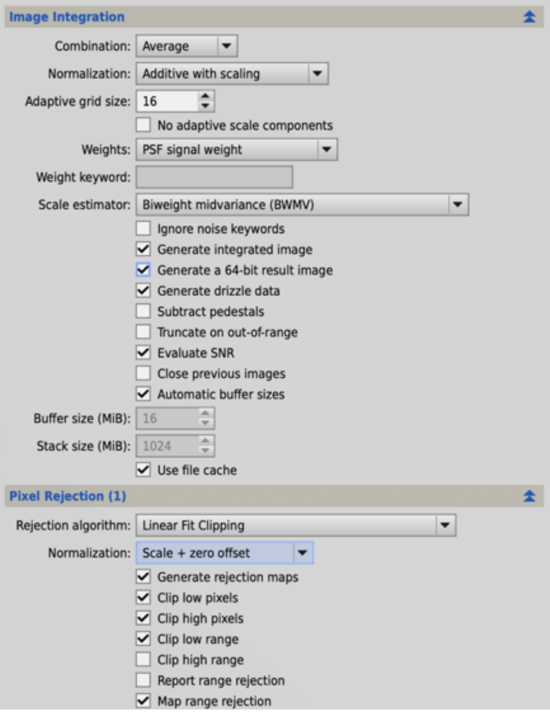


Image Integration

Combination: Average
Normalization: Additive with scaling
Adaptive grid size: 16
□ No adaptive scale components
Weights: PSF signal weight
Weight keyword:
Scale estimator: Biweight midvariance (BWMV)
□ Ignore noise keywords
☑ Generate integrated image
☑ Generate a 64-bit result image
☑ Generate drizzle data
□ Subtract pedestals
□ Truncate on out-of-range
☑ Evaluate SNR
□ Close previous images
☑ Automatic buffer sizes
Buffer size (MiB): 16
Stack size (MiB): 1024
☑ Use file cache

Pixel Rejection (1)

Rejection algorithm: Linear Fit Clipping
Normalization: Scale + zero offset
☑ Generate rejection maps
☑ Clip low pixels
☑ Clip high pixels
☑ Clip low range
□ Clip high range
□ Report range rejection
☑ Map range rejection

The process is repeated to create an integrated R-channel image, G-channel image and B-channel image. These separate integrated R, G and B images will already be aligned to each other because we aligned all of the original individual light frames to the G channel reference image.

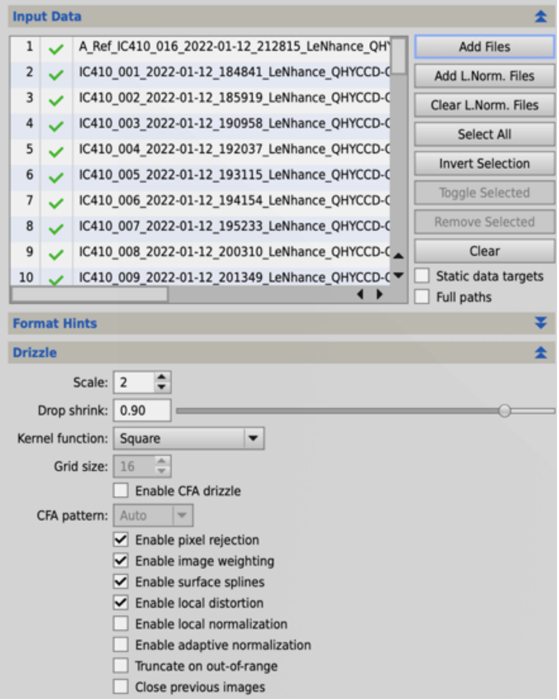
Drizzle Integration

The integration process uses a final optional further step of drizzle integration.

In the *Drizzle Integration Process*, load all of the drizzle files created in the *Star Alignment Process* and updated in the *Image Integration Process*.

The process default setting are normally retained. This final stage will result in integrated R, G and B channel images whose size and resolution are returned to the original camera resolutions.

Drizzle integration of all of the separate R, G and B drizzle date can be used to generate a synthetic luminous image.



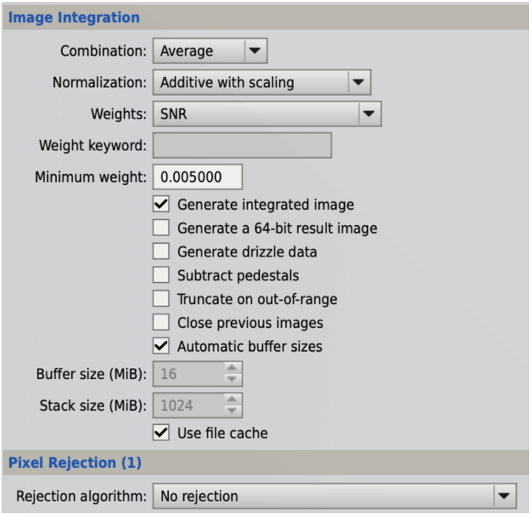
1	✓	A_Ref_IC410_016_2022-01-12_212815_LeNhance_QHY
2	✓	IC410_001_2022-01-12_184841_LeNhance_QHYCCD-C
3	✓	IC410_002_2022-01-12_185919_LeNhance_QHYCCD-C
4	✓	IC410_003_2022-01-12_190958_LeNhance_QHYCCD-C
5	✓	IC410_004_2022-01-12_192037_LeNhance_QHYCCD-C
6	✓	IC410_005_2022-01-12_193115_LeNhance_QHYCCD-C
7	✓	IC410_006_2022-01-12_194154_LeNhance_QHYCCD-C
8	✓	IC410_007_2022-01-12_195233_LeNhance_QHYCCD-C
9	✓	IC410_008_2022-01-12_200310_LeNhance_QHYCCD-C
10	✓	IC410_009_2022-01-12_201349_LeNhance_QHYCCD-C

☐ Static data targets
☐ Full paths

Scale: 2
Drop shrink: 0.90
Kernel function: Square
Grid size: 16
☐ Enable CFA drizzle
CFA pattern: Auto
☑ Enable pixel rejection
☑ Enable image weighting
☑ Enable surface splines
☑ Enable local distortion
☐ Enable local normalization
☐ Enable adaptive normalization
☐ Truncate on out-of-range
☐ Close previous images

Luminous Image

Regardless of whether drizzle integration was used, the process at this stage will have used of our captured light-frame date to create calibrated and integrated red, green and blue channel images (R, G and B). In effect, we have the same result as if we had used a mono camera with R, G and B filters. We don't however have a luminous (L) image. That is overcome by creating a synthetic luminous frame.



Combination: Average
Normalization: Additive with scaling
Weights: SNR
Weight keyword:
Minimum weight: 0.005000
☑ Generate integrated image
☐ Generate a 64-bit result image
☐ Generate drizzle data
☐ Subtract pedestals
☐ Truncate on out-of-range
☐ Close previous images
☑ Automatic buffer sizes
Buffer size (MiB): 16
Stack size (MiB): 1024
☑ Use file cache

Rejection algorithm: No rejection

The usual method to create a synthetic L-frame image is to integrate all of the aligned R, G and B separate images to create a single L or luminous frame. For narrow band images, it may be that a single channel, often the Ha or red channel, to create the L-frame image.

In the Image Integration process select SNR for weights to reduce the contribution of a noisy channel. Pixel rejection should be switched off.

At this stage we have combined all our data into four separate components or images that are aligned: L, R, G and B images.

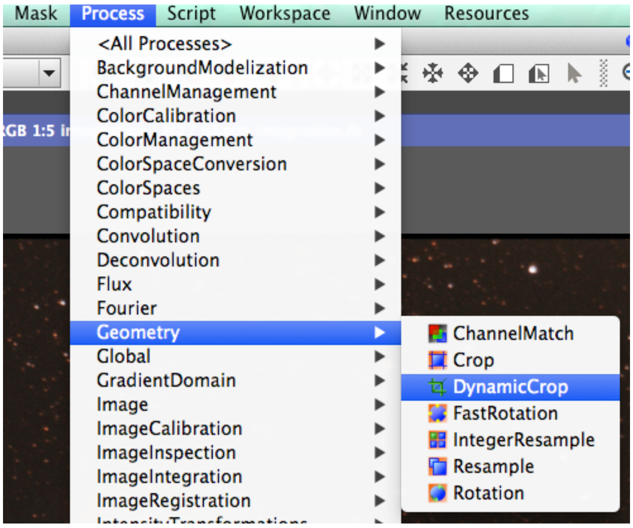
Linear Pre-Processing

Screen Transfer Function (STF)

https://pixinsight.com/doc/tools/ScreenTransferFunction/ScreenTransferFunction.html The PI is used to improve the visibility of a linear image on the screen but without altering its actual pixel data in anyway. All images shown here, during the linear processing workflow, are STF applied images.

Dynamic Crop

Some processes such as crop and background extraction work in what is called a 'dynamic mode'. These are characterised by a green tick to execute ✓. These processes cannot be closed without exiting with the red X.



Edge Artifact Trimming

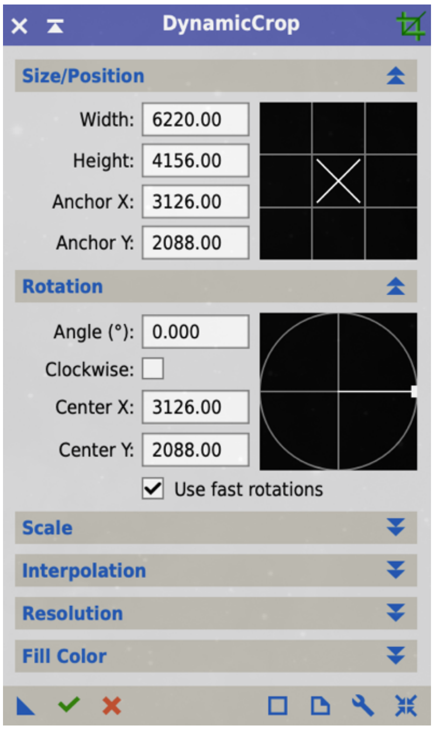
The image integration or stacking process may result in artefacts at the edge of the integrated image.

The Dynamic Crop Process is used to crop away the edges of the images to remove these artefacts.

Under normal conditions, 70-pixels are removed from each edge.

Images taken over multiple observing sessions may require greater edge cropping.

It is essential that an *instance* of the *Dynamic Crop Process* is created so that exactly the same dynamic crop is applied to each of the L, R, G and B images. This will ensure that the separate LRGB image components remain aligned with each other.



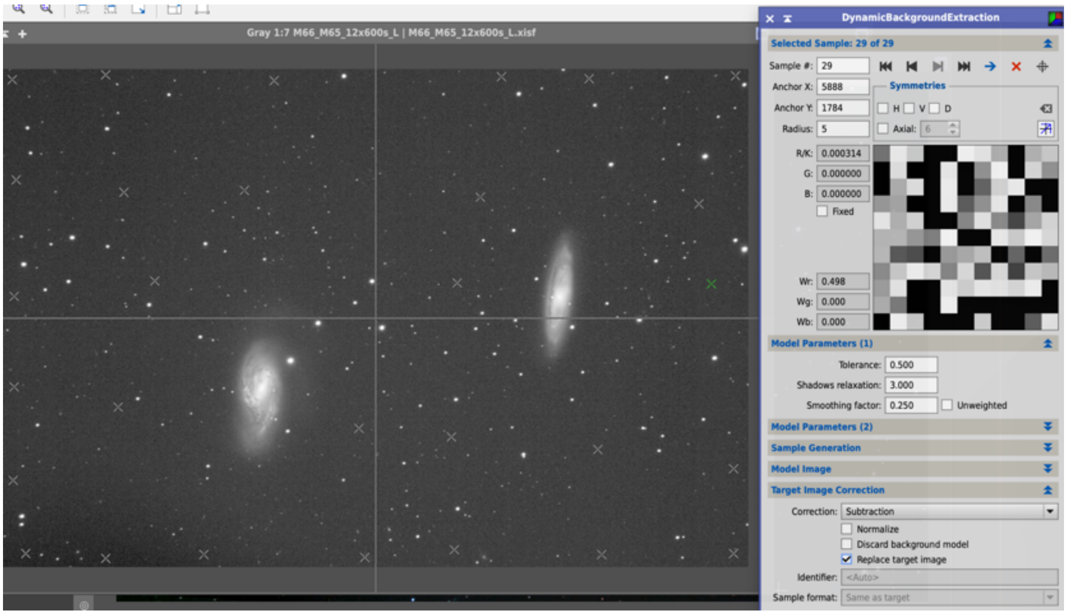
Width: 6220.00
Height: 4156.00
Anchor X: 3126.00
Anchor Y: 2088.00
Angle (°): 0.000
Clockwise: □
Center X: 3126.00
Center Y: 2088.00
☑ Use fast rotations

Background Model and Subtraction

PixInsight has two light gradient removal tools:

- ***Automatic Background Extraction (ABE)Process***
- ***Dynamic Background Extraction (DBE)Process***

As the name implies, the automatic background extraction will automatically analyse an image to identify and extract a light background gradient. The background may include both light pollution and vignetting effects not removed by flat-frame calibration.



Gray 1:7 M66_M65_12x600s_L | M66_M65_12x600s_L.xisf

DynamicBackgroundExtraction

Selected Sample: 29 of 29

Sample #: 29
Anchor X: 5888
Anchor Y: 1784
Radius: 5

Symmetries: □ H □ V □ D
□ Axial: 6

R/K: 0.000314
G: 0.000000
B: 0.000000
□ Fixed

Wr: 0.498
Wg: 0.000
Wb: 0.000

Model Parameters (1)

Tolerance: 0.500
Shadows relaxation: 3.000
Smoothing factor: 0.250 □ Unweighted

Model Parameters (2)

Sample Generation

Model Image

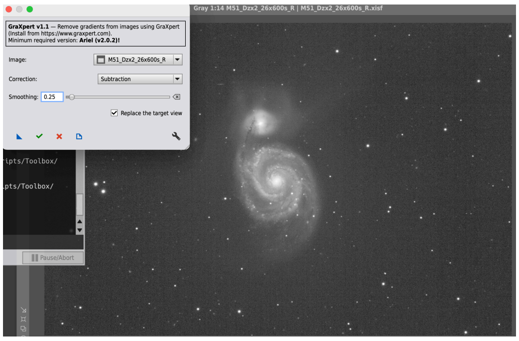
Target Image Correction

Correction: Subtraction
□ Normalize
□ Discard background model
☑ Replace target image
Identifier: <Auto>
Sample format: Same as target

The dynamic background extraction process allows the user to determine the sample points. This has the advantage of ensuring that faint subject areas of the image are not mis-interpreted as being a light gradient artefact.

An alternative to ABE and DBE is to use GraXpert within PixInsight.

The GraXpert AI script is a one-click script that does not require and sample points to be selected to identify and remove light gradients.



Gray 1:14 M51_Dzx2_26x600s_R | M51_Dzx2_26x600s_R.xisf

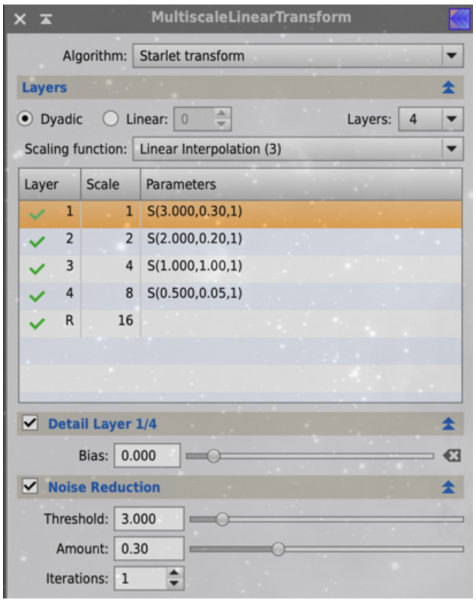
GraXpert v1.1 — Remove gradients from images using GraXpert
(Install from https://www.graxpert.com).
Minimum required version: **Ariel (v2.0.2)!**

Image: M51_Dzx2_26x600s_R
Correction: Subtraction
Smoothing: 0.25
☑ Replace the target view

Differential Linear RGB Noise Reduction

An initial noise reduction is carried out on each of the R, G and B images using the *Multiscale Linear Transform Process*.

The wavelet layer setting for the red and green channel are shown in the image to the right. Higher noise reduction amount levels are used for the blue channel which will inherently be noisier.


MultiscaleLinearTransform

Algorithm: Starlet transform

Layers

● Dyadic ○ Linear: 0 Layers: 4

Scaling function: Linear Interpolation (3)

	Layer	Scale	Parameters
✓	1	1	S(3.000,0.30,1)
✓	2	2	S(2.000,0.20,1)
✓	3	4	S(1.000,1.00,1)
✓	4	8	S(0.500,0.05,1)
✓	R	16	

☑ Detail Layer 1/4

Bias: 0.000

☑ Noise Reduction

Threshold: 3.000

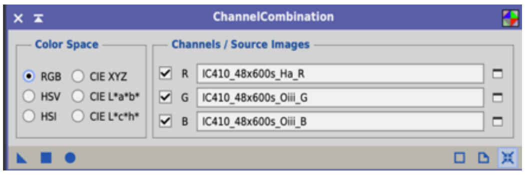
Amount: 0.30

Iterations: 1

RGB Channel Combination

True Colour Broadband Images

The separate R, G and B channel images are combined using the *Channel Combination Process* to create a single colour RGB image.


ChannelCombination

Color Space

● RGB ○ CIE XYZ

○ HSV ○ CIE L*a*b*

○ HSI ○ CIE L*c*h*

Channels / Source Images

☑ R IC410_48x600s_Ha_R

☑ G IC410_48x600s_Oiii_G

☑ B IC410_48x600s_Oiii_B

Simply the red channel image is assigned to the red channel in the channel combination process and the green and blue channel images assigned to their equivalents in the process.

Close to True Colour Narrow Band Images

Images taken with the colour camera and a tri or duo-band filter to selectively capture Ha and Oiii transmissions can also be assigned to the corresponding channel in the channel combination process:

Ha assigned to RED

Oiii green assigned to GREEN

Oiii blue assigned to BLUE

The resulting composite RGB image is referred to as a HOO image.

False Colour Narrow Band Images

Images taken to isolate the traditional emission lines of Sii, Ha and Oiii cannot be assigned to their corresponding colours as both Sii and Ha are in the red channel. In this case a false colour RGB image is created to differentiate between the Sii and Ha components. The process for capturing these components with the QHY268C camera is:

1. Capture Ha and Oiii data using the Optolong LeXtreme filter (7nm pass on Ha and 7nm pass on Oiii).
2. Using the 'channel extraction script' separate the Ha (red channel) and the Oiii (green and blue channels).
3. Capture the Sii using the Optolong Sii 6.5 filter (6nm pass).
4. Using the channel extraction script separate the Sii (red channel).
5. Align and integrate to create linear Ha, Sii, Oiii(green) and Oiii(blue) images.
6. Combine the Oiii(green) and Oiii(Blue) images with PixelMath to create a single Oiii image.

The separate integrated images are now combined with the 'channel combination' process as:

Sii assigned to RED

Ha assigned to GREEN

Oiii assigned to BLUE

The resulting RGB image will be a false colour image also referred to as a SHO image or a Hubble Pallet image.

DeConvolution of L-Image and RGB Image - BlurXTerminator

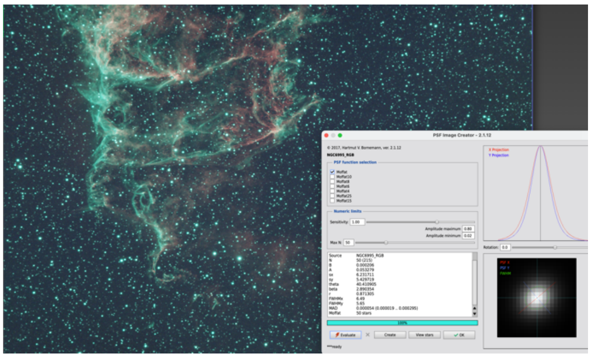
BlurXTerminator is an AI based deconvolution tool designed specifically for astronomical images taken with equipment commonly used by amateur astrophotographers.

BlurXTerminator is applied both to the linear luminous image and the linear, SPCC colour calibrated RGB image.

Deconvolution Process:

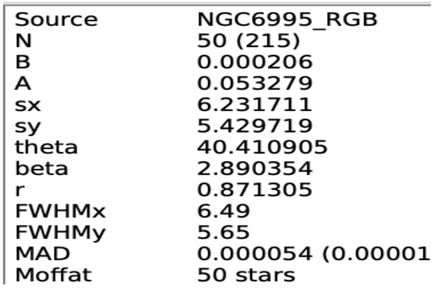
With the RGB or L image open, from <Script><Render> select the PSF Image script.

Select ‘Evaluate’



Note the average FWHM value for the stars in the image.

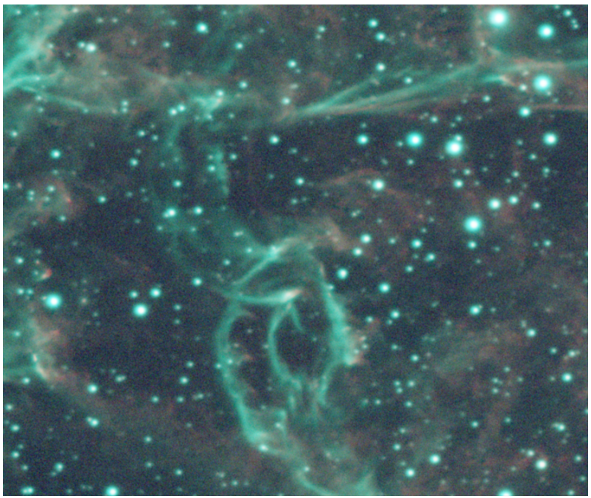
In this example the FWHM is (6.49+5.65)/2 = 6.07



```
Source      NGC6995_RGB
N           50 (215)
B           0.000206
A           0.053279
sx          6.231711
sy          5.429719
theta       40.410905
beta        2.890354
r           0.871305
FWHMx       6.49
FWHMy       5.65
MAD         0.000054 (0.00001
Moffat      50 stars
```

Create a preview area of a small section of the image and create several copies of the preview.

The previews will enable us to test different settings of BlurXterminator before applying it to the whole image.

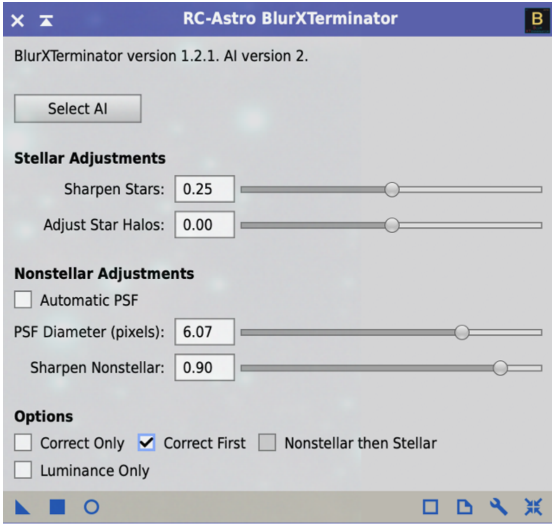


In the BlurXterminator process switch off ‘Automatic PSF’ and input the PSF calculated by the PSF Image script.

Select ‘Correct First’ and apply the process to the preview.

Repeat the process on a separate preview using the ‘Non stellar than stellar’ option for comparison.

It may also be desirable to test with different ‘Adjust Star Halos’ and level of ‘Sharpen Nonstellar’.



RC-Astro BlurXTerminator

BlurXTerminator version 1.2.1. AI version 2.

Stellar Adjustments

Sharpen Stars: 0.25

Adjust Star Halos: 0.00

Nonstellar Adjustments

☐ Automatic PSF

PSF Diameter (pixels): 6.07

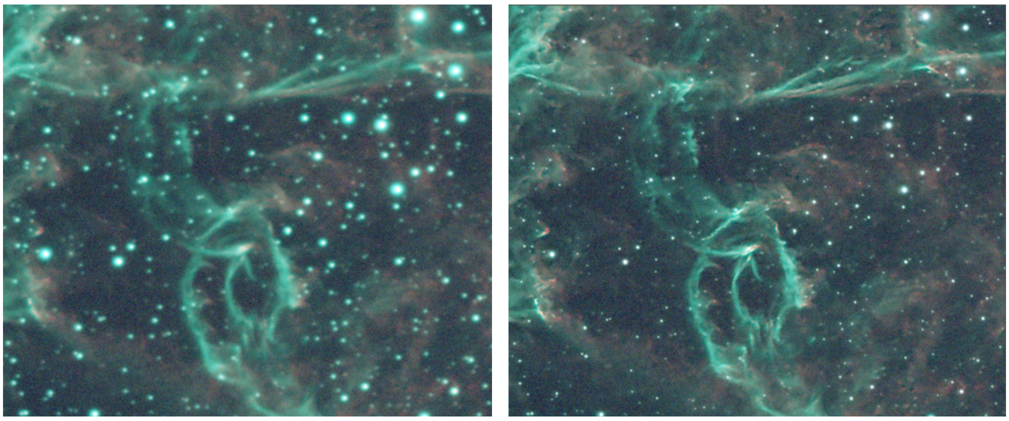
Sharpen Nonstellar: 0.90

Options

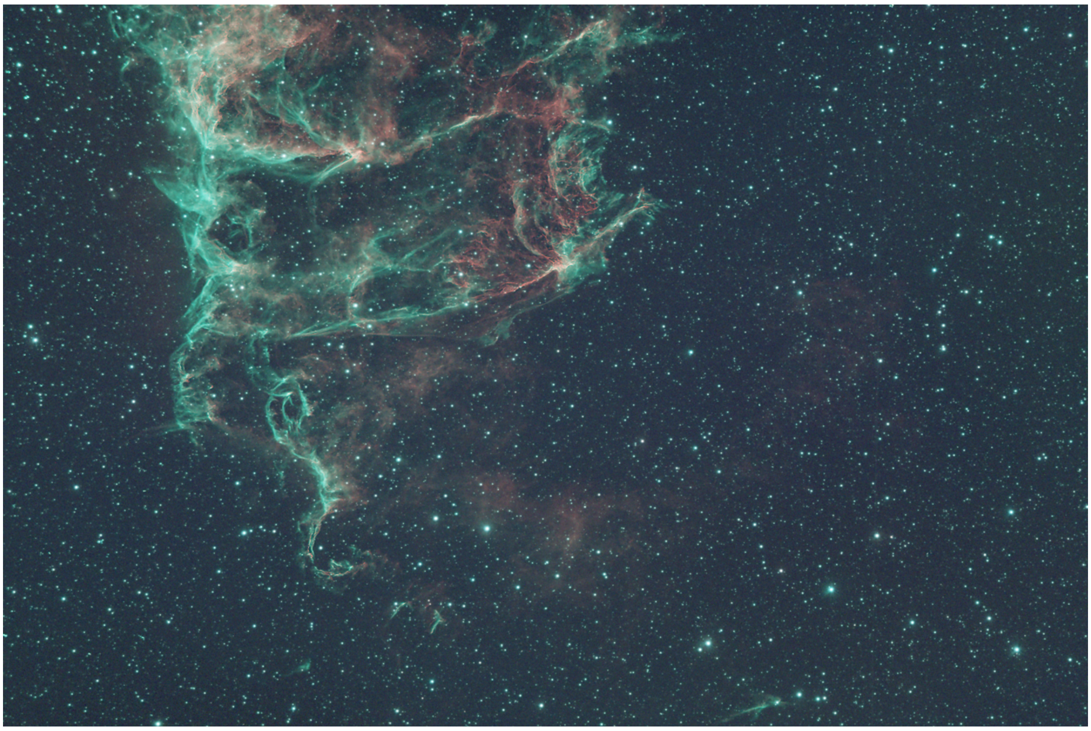
☐ Correct Only ☑ Correct First ☐ Nonstellar then Stellar

☐ Luminance Only

Before BlurXTerminator **After BlurXTerminator**



The BlurXTerminator process is now applied to the whole image to carry out the deconvolution.



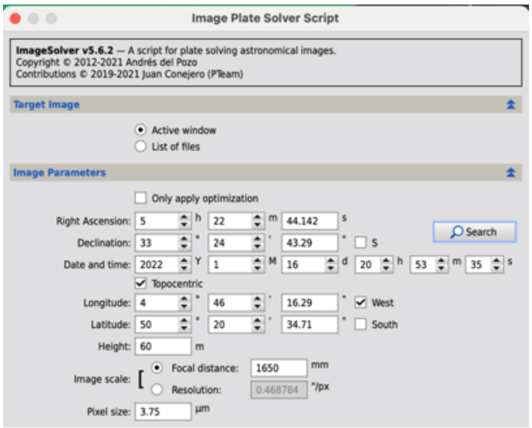
Colour Calibration

Image Plate Solve

The new combined RGB image needs to be colour calibrated. The first step to achieve the colour calibration is to plate solve the RGB image.

The telescope focal length and camera pixel size must be correctly entered into the script.

Note that if drizzle integration was used, the resolution in arcsec/pixel will be reduced by 50% and the pixel size will also be reduced by 50%.



ImageSolver v5.6.2 — A script for plate solving astronomical images.
Copyright © 2012-2021 Andrés del Pozo
Contributions © 2019-2021 Juan Conejero (PTeam)

Target Image

● Active window
○ List of files

Image Parameters

□ Only apply optimization

Right Ascension: 5 h 22 m 44.142 s
Declination: 33 ° 24 ' 43.29 " □ S
Date and time: 2022 Y 1 M 16 d 20 h 53 m 35 s
☑ Topocentric
Longitude: 4 ° 46 ' 16.29 " ☑ West
Latitude: 50 ° 20 ' 34.71 " □ South
Height: 60 m
Image scale: ● Focal distance: 1650 mm
○ Resolution: 0.468784 "/px
Pixel size: 3.75 µm

Pixel Size and Resolution for PixInsight Plate Solving

Broadband Image Debayered using VNG Algorithm QHY268C Camera with 9.25 EdgeHD				
Focal Length	1,650mm		2,350mm	
	Pixel Size µm	Resolution Arcsec/pixel	Pixel Size µm	Resolution Arcsec/pixel
Bin 1x1	3.76	0.468	3.76	0.329
Debayer VNG	3.76	0.468	3.76	0.329
Drizzle x2	1.88	0.234	1.88	0.165

Narrowband Image Debayered using Super Pixel Algorithm QHY268C Camera with 9.25 EdgeHD				
Focal Length	1,650mm		2,350mm	
	Pixel Size µm	Resolution Arcsec/pixel	Pixel Size µm	Resolution Arcsec/pixel
Bin 1x1	3.76	0.468	3.76	0.329
Debayer SP	7.52	0.936	7.52	0.658
Drizzle x2	3.76	0.468	3.76	0.329
Drizzle x4	1.88	0.234	1.88	0.165

The *Image Plate Solver Script* will plate solve the RGB image and save the image RA and Dec. coordinates to the RGB image XISF header.

The plate solve will also confirm the telescope focal length and the field of view covered by the image.

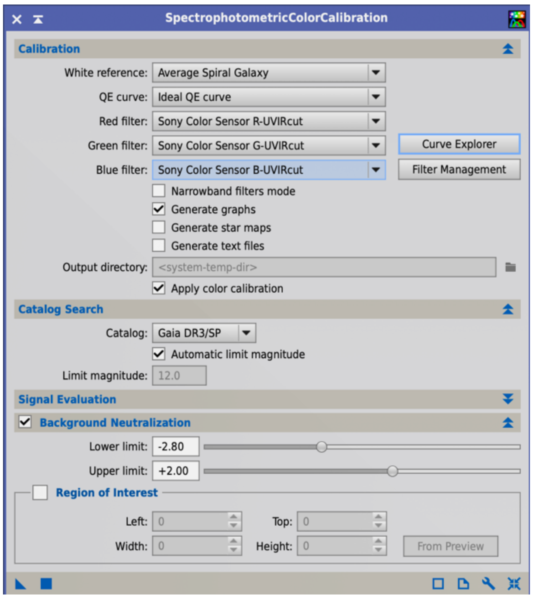
Spectrophotometric Colour Calibration

The plate solved and deconvoluted RGB image can now be colour corrected using the Spectrop*hotometric Color Calibration Process*.

The image parameters are automatically acquired from the image.

The plate solving parameter catalogue is set to Gaia DR3/SP.

A 'preview' frame is drawn on an area of clear background sky in the RGB image. The preview frame is used to set a region of interest to carry out a background neutralisation as part of the colour calibration.



SpectrophotometricColorCalibration

Calibration

White reference: Average Spiral Galaxy

QE curve: Ideal QE curve

Red filter: Sony Color Sensor R-UVIRcut

Green filter: Sony Color Sensor G-UVIRcut

Blue filter: Sony Color Sensor B-UVIRcut

☐ Narrowband filters mode

☑ Generate graphs

☐ Generate star maps

☐ Generate text files

Output directory: <system-temp-dir>

☑ Apply color calibration

Catalog Search

Catalog: Gaia DR3/SP

☑ Automatic limit magnitude

Limit magnitude: 12.0

Signal Evaluation

☑ Background Neutralization

Lower limit: -2.80

Upper limit: +2.00

☐ Region of Interest

Left: 0 Top: 0

Width: 0 Height: 0

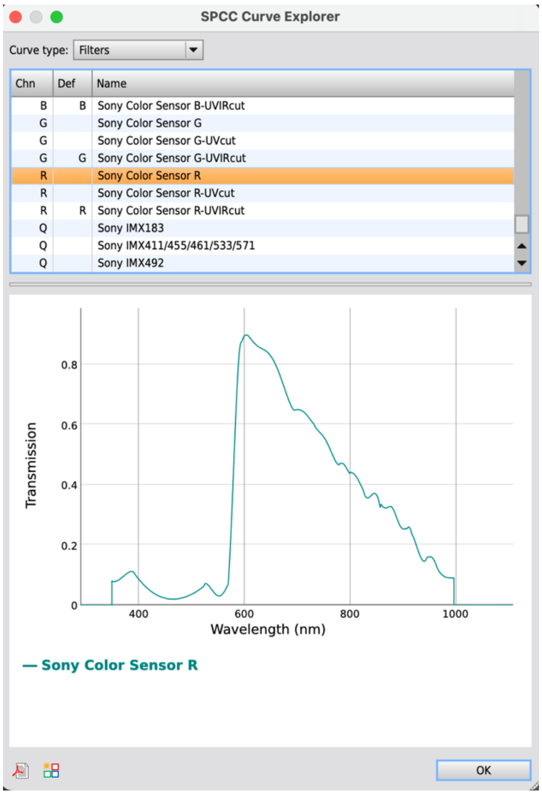
Broadband Colour Calibration - SPCC Curves Explorer

To provide precise colour calibration, for a broad band image, set the 'QE Curve' to 'Ideal QE Curve'.

The QE response for each colour channel is set using the SPCC 'Curve Explorer'.

First set the QE curve for the individual camera pixels.

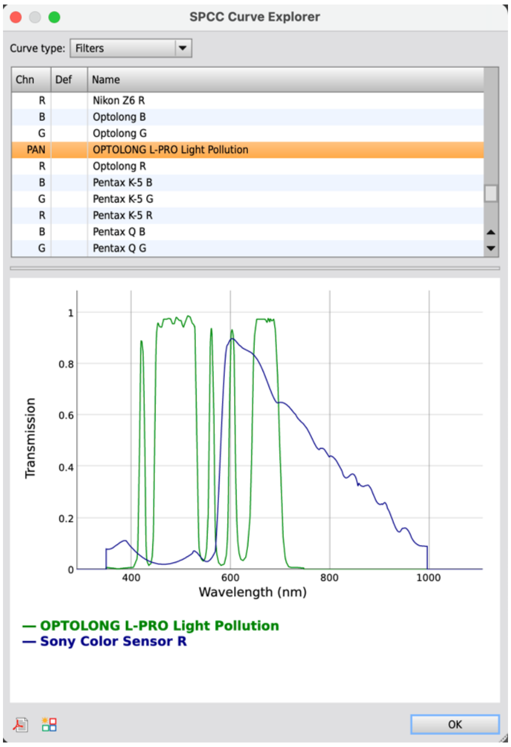
Next, also select the filter that is being used with the camera, in this example the Optolong L-Pro filter.



SPCC Curve Explorer

Curve type: Filters

Chn	Def	Name
B	B	Sony Color Sensor B-UVIRcut
G		Sony Color Sensor G
G		Sony Color Sensor G-UVcut
G	G	Sony Color Sensor G-UVIRcut
R		Sony Color Sensor R
R		Sony Color Sensor R-UVcut
R	R	Sony Color Sensor R-UVIRcut
Q		Sony IMX183
Q		Sony IMX411/455/461/533/571
Q		Sony IMX492

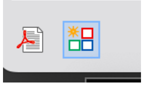


SPCC Curve Explorer

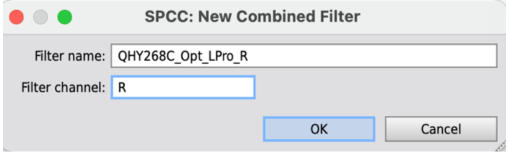
Curve type: Filters

Chn	Def	Name
R		Nikon Z6 R
B		Optolong B
G		Optolong G
PAN		OPTOLONG L-PRO Light Pollution
R		Optolong R
B		Pentax K-5 B
G		Pentax K-5 G
R		Pentax K-5 R
B		Pentax Q B
G		Pentax Q G

Once the combination of pixel and filter is chosen, select the filter combination icon



Give the combined filter a name and assign it to a colour channel.

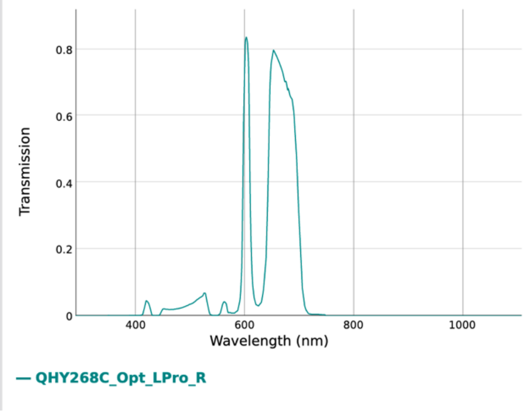


SPCC: New Combined Filter

Filter name: QHY268C_Opt_LPro_R

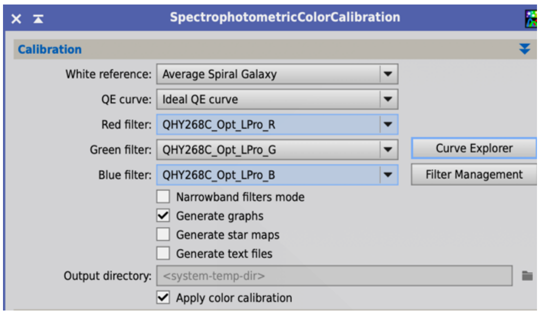
Filter channel: R

The combined filter curve provides and acurate profile of the sensativity for each colour channel to the range of wavelengths of light.



Repeat the process for the green and blue channels.

The individual filters created for each colour channel is applied in SPCC.



SpectrophotometricColorCalibration

Calibration

White reference: Average Spiral Galaxy

QE curve: Ideal QE curve

Red filter: QHY268C_Opt_LPro_R

Green filter: QHY268C_Opt_LPro_G

Blue filter: QHY268C_Opt_LPro_B

□ Narrowband filters mode

☑ Generate graphs

□ Generate star maps

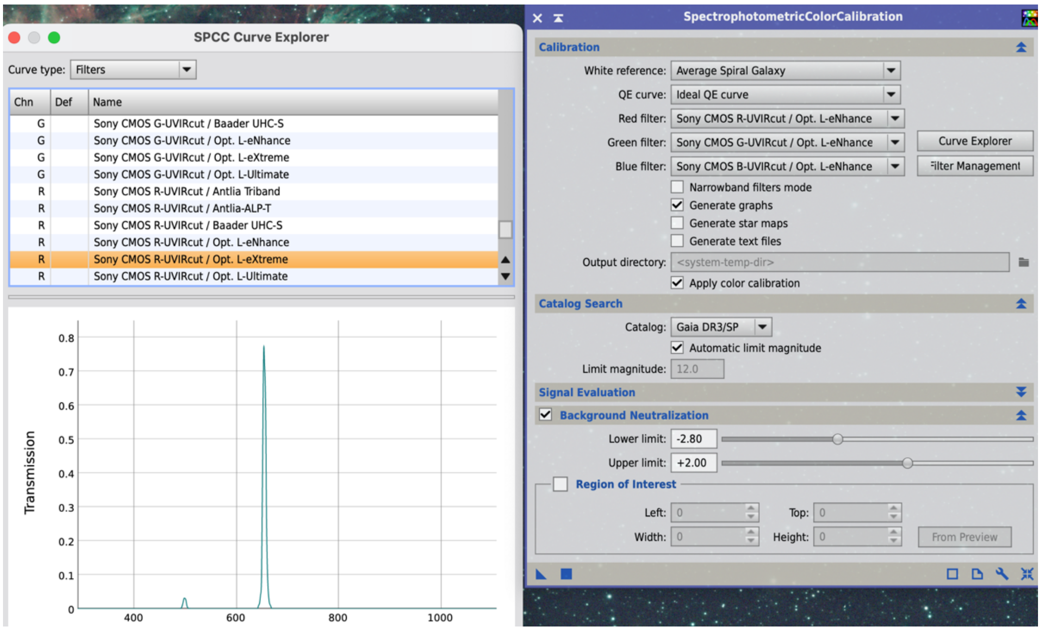
□ Generate text files

Output directory: <system-temp-dir>

☑ Apply color calibration

Narrow Band Image Colour Calibration – Close to True Colour

For narrow band images taken with a tri-band or duo-band filter (such as the Optlong eNhance or eXtreme) then colour calibration can be carried out with the Spectrophotometric Colour Calibration process, in the same way as for a broadband image.



SPCC Curve Explorer

Curve type: Filters

Chn	Def	Name
G		Sony CMOS G-UVIRcut / Baader UHC-S
G		Sony CMOS G-UVIRcut / Opt. L-eNhance
G		Sony CMOS G-UVIRcut / Opt. L-eXtreme
G		Sony CMOS G-UVIRcut / Opt. L-Ultimate
R		Sony CMOS R-UVIRcut / Antlia Triband
R		Sony CMOS R-UVIRcut / Antlia-ALP-T
R		Sony CMOS R-UVIRcut / Baader UHC-S
R		Sony CMOS R-UVIRcut / Opt. L-eNhance
R		Sony CMOS R-UVIRcut / Opt. L-eXtreme
R		Sony CMOS R-UVIRcut / Opt. L-Ultimate

SpectrophotometricColorCalibration

Calibration

White reference: Average Spiral Galaxy

QE curve: Ideal QE curve

Red filter: Sony CMOS R-UVIRcut / Opt. L-eNhance

Green filter: Sony CMOS G-UVIRcut / Opt. L-eNhance

Blue filter: Sony CMOS B-UVIRcut / Opt. L-eNhance

☐ Narrowband filters mode

☑ Generate graphs

☐ Generate star maps

☐ Generate text files

Output directory: <system-temp-dir>

☑ Apply color calibration

Catalog Search

Catalog: Gaia DR3/SP

☑ Automatic limit magnitude

Limit magnitude: 12.0

Signal Evaluation

☑ **Background Neutralization**

Lower limit: -2.80

Upper limit: +2.00

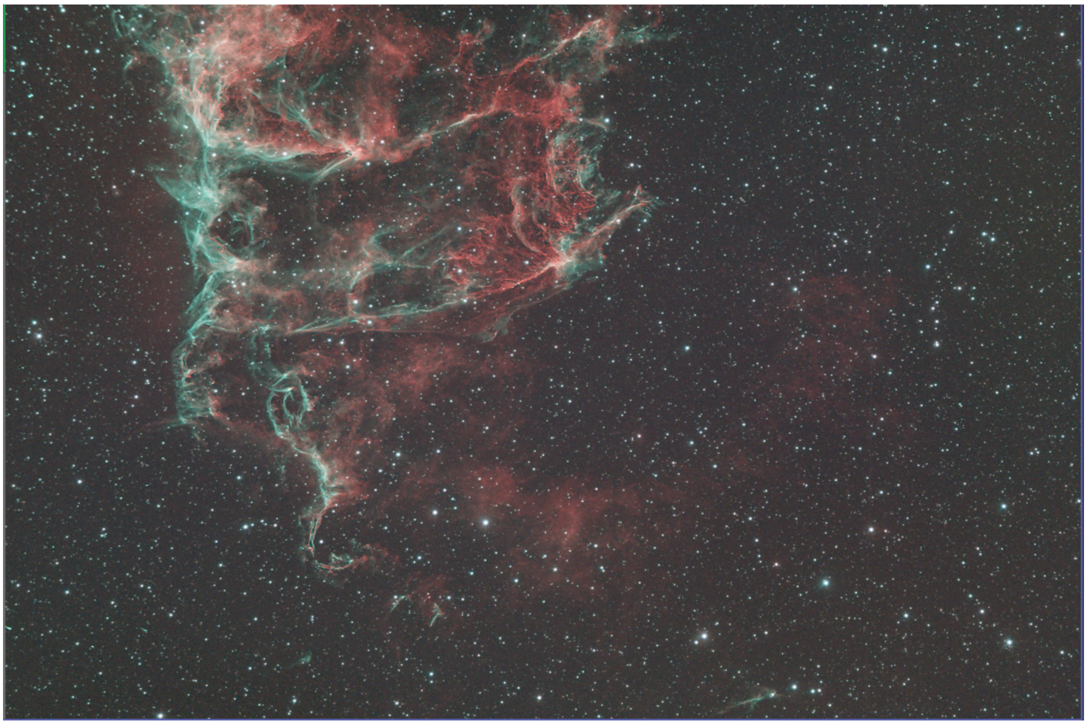
☐ **Region of Interest**

Left: 0 Top: 0

Width: 0 Height: 0

For the filter option in the SPCC process select the appropriate narrowband filter.

In this example an Optolong LeNhance filter was used with the QHY268C camera to produces an Ha and Oiii image.

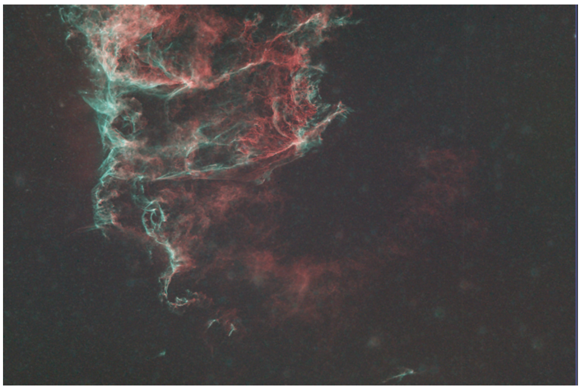


The Ha signal was recorded in the red channel while the Oiii was recorded in both the green and blue channels of the original image. This allows us to colour calibrate to create a close to true colour image with SPCC.

Colour Correction of Nebula

The SPCC process will provide fully colour calibrated stars in the image. The nebula colour is composed of discrete narrow bands of colour and so will approximate true colour. However, the Ha red channel will predominate in the image. This skewing of the image can be further colour corrected as follows.

The first step to improving the colour of the nebula is to separate the colour calibrated stars and nebula using the StarNet process described in the next section. This prevents us undoing the correct colour calibration of the stars.

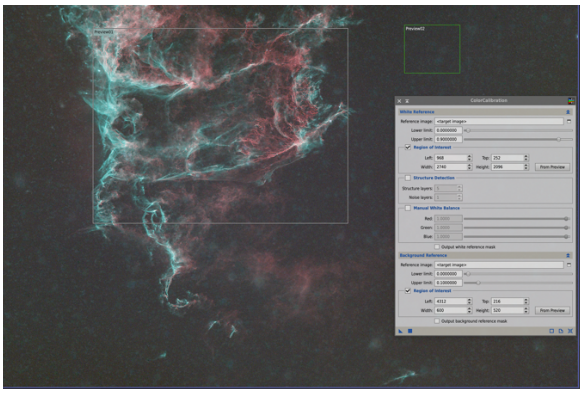


The over-emphasised Ha red channel can now be corrected using the Colour Calibration process.

Switch off ‘Structure Detection’

Create a preview over the nebula and set this as the ‘Region of Interest’ for calibration.

Create a second preview and use this as the ‘Region of Interest’ for ‘Background Neutralisation’.

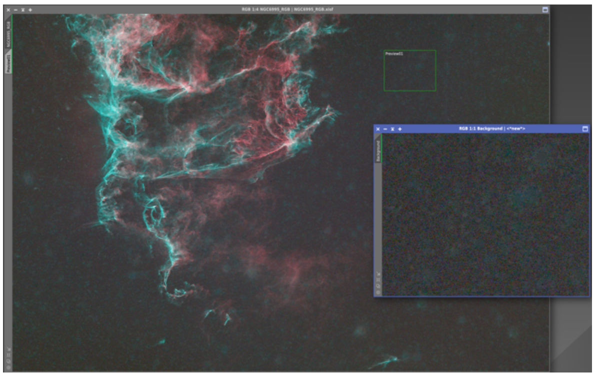


White Reference
Reference image: <target image>
Lower limit: 0.0000000
Upper limit: 0.9000000
☑ Region of Interest
Left: 968 Top: 252
Width: 2740 Height: 2096 From Preview
□ Structure Detection
Structure layers: 5
Noise layers: 1
□ Manual White Balance
Red: 1.0000
Green: 1.0000
Blue: 1.0000
□ Output white reference mask
Background Reference
Reference image: <target image>
Lower limit: 0.0000000
Upper limit: 0.1000000
☑ Region of Interest
Left: 4312 Top: 216
Width: 600 Height: 520 From Preview
□ Output background reference mask

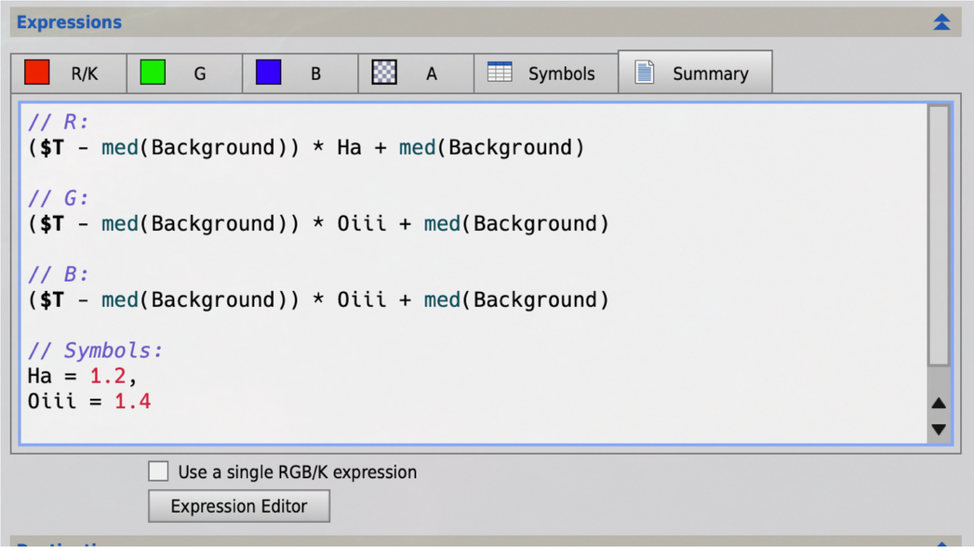
With the nebula colour corrected using the Colour Calibration process we can now further selectively enhance the individual narrow band components.

The first step to the narrowband enhancement is to create a preview of a neutral area of the sky in the image and drag the preview to the PI desktop to create a new image. Rename the new image ‘Background’.

We will use the background to ensure any enhancements are only applied to the nebula and not the background sky pedestal.


RGB 1:4 NGC6995_RGB | NGC6995_RGB.xisf
NGC6995_RGB
Preview01
Preview01
RGB 1:1 Background | <*new*>
Background

In the PixelMath editor, deselect ‘Use a single RGB/K expression’ in order to enter different expressions for each colour channel.



```
// R:
($T - med(Background)) * Ha + med(Background)

// G:
($T - med(Background)) * Oiii + med(Background)

// B:
($T - med(Background)) * Oiii + med(Background)

// Symbols:
Ha = 1.2,
Oiii = 1.4
```

☐ Use a single RGB/K expression

In the Pixel Math Expression Editor set two symbols – Ha = 1.0, Oiii = 1.0

In the red channel of the expression editor use the expression:

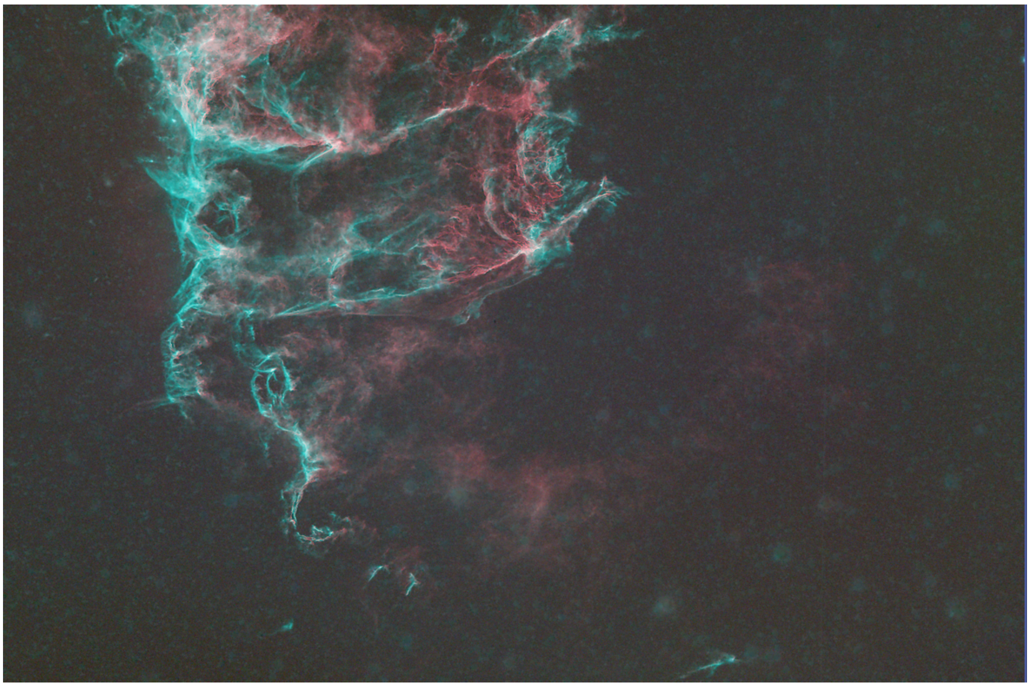
> ($T - med(Background)) * Ha + med(Background)

In each of the green and blue channels use the expression:

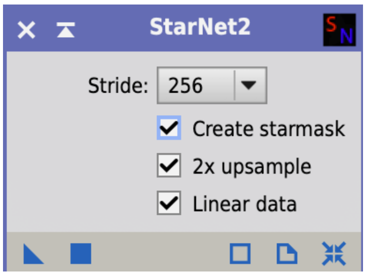
> ($T - med(Background)) * Oiii + med(Background)

Note that ‘*med(Background)’* calculates the background pedestal. In each expression we are first subtracting the background pedestal of each pixel value; enhancing it by the factor Ha or Oiii; and the adding back the background pedestal.

The value of Ha and/or Oiii enhancement is then set in the Symbol value to test the effect on the image. In this example the Ha component is enhanced by 1.2x and the Oiii 1.4x.



Object and Star Separation – StarNet2



StarNet2

Stride: 256

☑ Create starmask

☑ 2x upsample

☑ Linear data

StarNet2 can be used on both linear and non-linear data. Normally we use it on the linear images prior to processing.

The ‘2x upsample’ option improves star recognition and removal by 2x-upsampling the image first and after star extraction down-sampling the image again to its original resolution. This option would not normally be used for data that has been drizzle integrated.

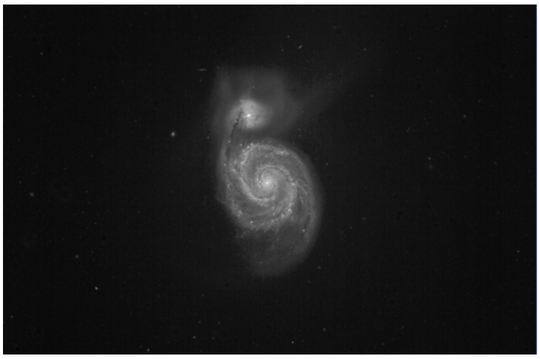
The ’Create starmask’ option creates the separate ‘star’ image as well as the starless image.

This is a calibrated, but otherwise unprocessed, linear image of M51. A PixInsight STF stretch has been applied to the image.

StarNet2 process was applied to the image with the ‘Create starmask’, ‘and ‘Linear data’ options all checked.

The StarNet process successfully removed all of the stars to create a ‘galaxy-only’ linear image.

A clone was created of the galaxy-only image from which a galaxy-mask could be created.

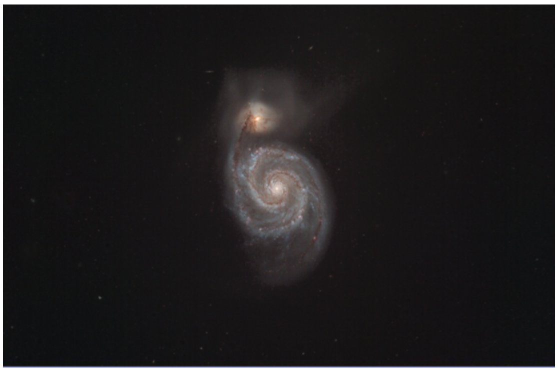


The StarNet process also created a 'star-only' linear image.

Note that the black-point will have already been set by the StarNetV2 'create a stark mask operation'. The black-point should not require further adjustment for the star-only image.

A clone of the star-only image was created from which a star-mask could be created.

The StarNet process was also run on the colour 'RGB' linear image to create a galaxy-only RGB image and a star-only RGB image.



At this stage we have four linear images:

a) Galaxy-only luminous
b) Star-only luminous
c) Galaxy-only RGB colour
d) Star-only RGB colour

Creating Masks

A clone of the luminous Galaxy-Only and Star-Only luminous images can be used to create galaxy and star masks. The masks will be tools that we can use to process our actual luminous and star images as we turn them into a single high dynamic range image.

Clone Stamp Tool

The 'CloneStamp' tool is found in the <Painting> group of processes.

The sole purpose of the clone stamp tool is to correct small-scale defects, residuals or artifacts cosmetically, replacing them plausibly with nearby data.

To copy parts of one image to another image:

- Load the CloneStamp tool.
- Define the receiving image by clicking the top name-bar to turn it blue.
- Click reset in CloneStamp.

Now press Ctrl/Cmd and left-hand mouse button in the sending image.

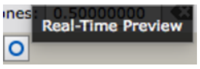
Histogram Transformation (HT)

Histogram Transformation is one of PixInsight's 'Intensity Transformations'. It allows us to stretch the relationship between the actual pixel values in the original image and the output pixel values in a non-linear way. This allows us to brighten dim objects without further brightening bright objects.

Switch off the STF to return the view of the linear image to its actual dark appearance. Launch the Histogram Transformation process.

In the <no view selected> box select the image to apply the histogram stretch to.

Select the 'Real Time Preview' button.



A real time preview window will open to show the effects on our image of adjusting settings in the Histogram Transformation (HT) process.

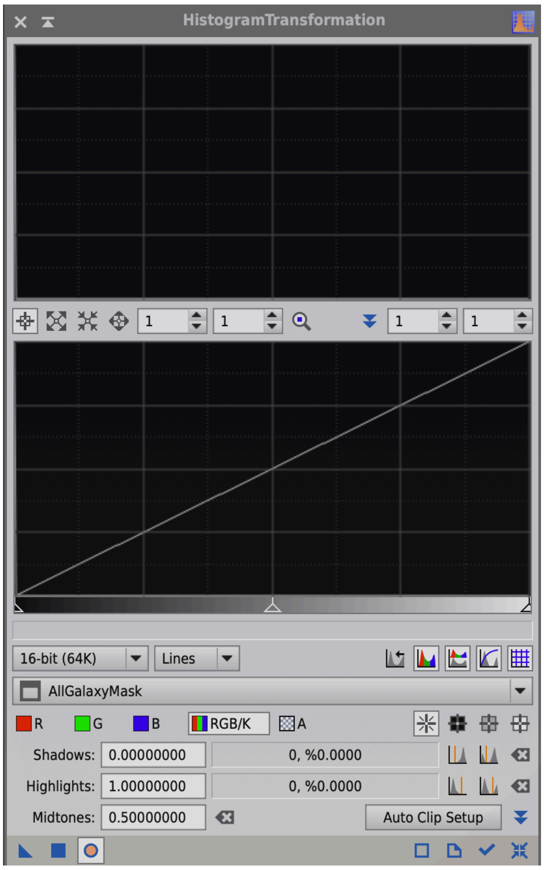
A full description of the process is given at https://pixinsight.com/doc/tools/HistogramTransformation/HistogramTransformation.html

The HT process has two graphical displays. The bottom one shows the relationship between the current pixel values (x-axis) and the new pixel values (y-axis). In the example above the linear diagonal line in the lower graph window demonstrates the linear nature between pixel current values and their corresponding new values.

At the bottom of the graph are three sliders. If the middle slider is moved to the left, the linear relationship between current values and new values will be stretched.

The left hand slider sets the black point. By not moving it we anchor its position so that an empty pixel corresponds with black in the output image.

The right hand slider sets the white point. This ensures that only a completely full pixel corresponds to white in the output image.



AllGalaxyMask

Shadows: 0.00000000 0, %0.0000

Highlights: 1.00000000 0, %0.0000

Midtones: 0.50000000

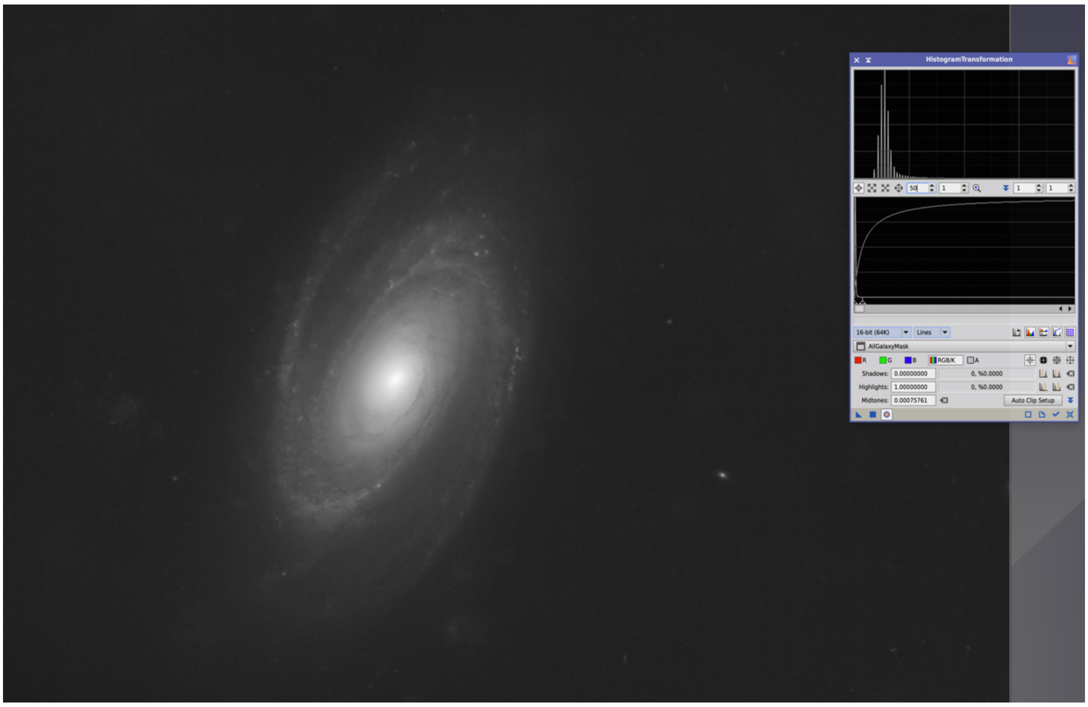
Moving the middle slider will change the overall relationship between input and output values to produce a non-linear relationship.

As we move the middle slider to the left, we can see a number of things happening:

1. The shape of the line in the lower graph is becoming highly curved;
2. In the real time preview more objects are becoming brighter and visible;
3. The upper graph in the HT process window now shows a graphical spike that has broadened and moved to the right.

Dark pixels are on the far left of the chart and bright pixels on the far right. As we moved the middle slider on the lower graph to the left, the spike in the upper graph moved to the right. The spike tells us that most pixels are very dim in this picture but they have brightened as the slider was moved further to the left. This confirms the brightening that was also visible in the real time preview of our image.

There comes a point where we have moved that middle slider so far to the left that we can go no further and yet the image is still not bright enough. This is where PixInsight shows its power as an astronomical image-processing tool. The horizontal zoom value can be increased for the lower input graph. As we zoom-in, the gap between the left and middle sliders is widened so that we can further adjust them to increase the image stretch.



Horizontal zoom: 50 | 1 | 1 | 1

AllGalaxyMask

Shadows: 0.00000000 | 0, %0.0000
Highlights: 1.00000000 | 0, %0.0000
Midtones: 0.00075761

Adjusting the HT sliders allows us to create a non-liner copy of the galaxy-only image that we will use as a template to create a number of masks.

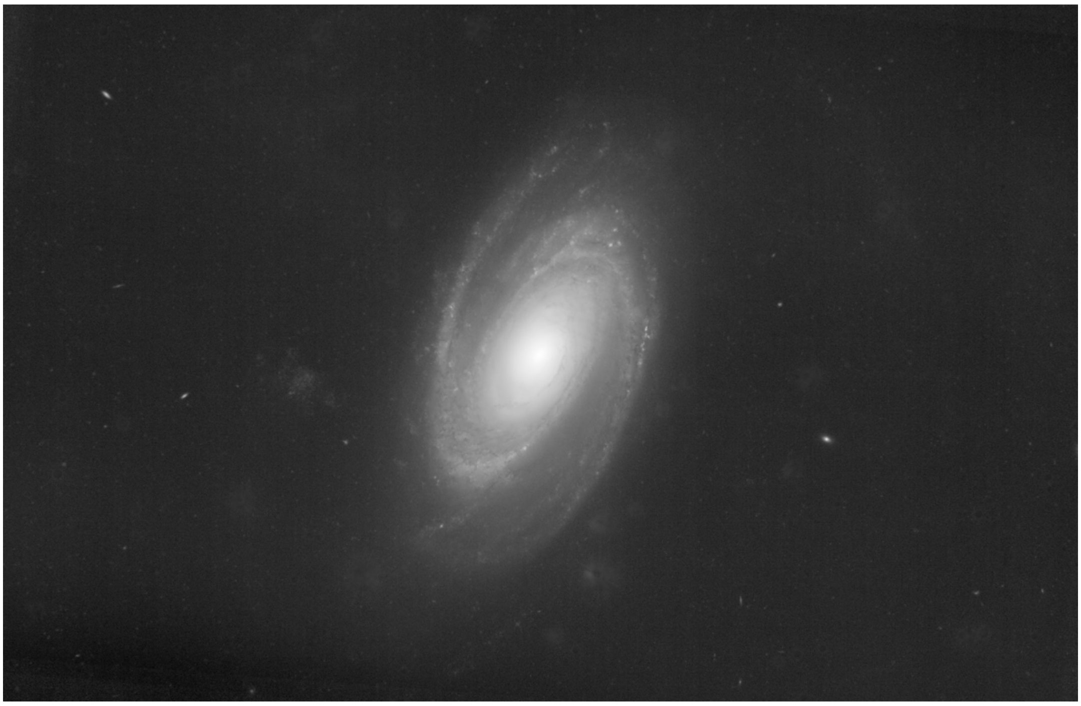
The same application to the stars-only image copy, lets us create a non-linear star-mask.

It is important to remember that these are tools that we are creating to help us process the actual images. These are not our actual images. We will apply masks to our real images to apply a process to a defined part of an image only, the part under the white-area of the mask. The parts of the image under the black-area of the mask will not have the process applied at all. The areas of the image under grey-areas of the mask will have the process partly applied.

Improving Galaxy/Nebula Masks

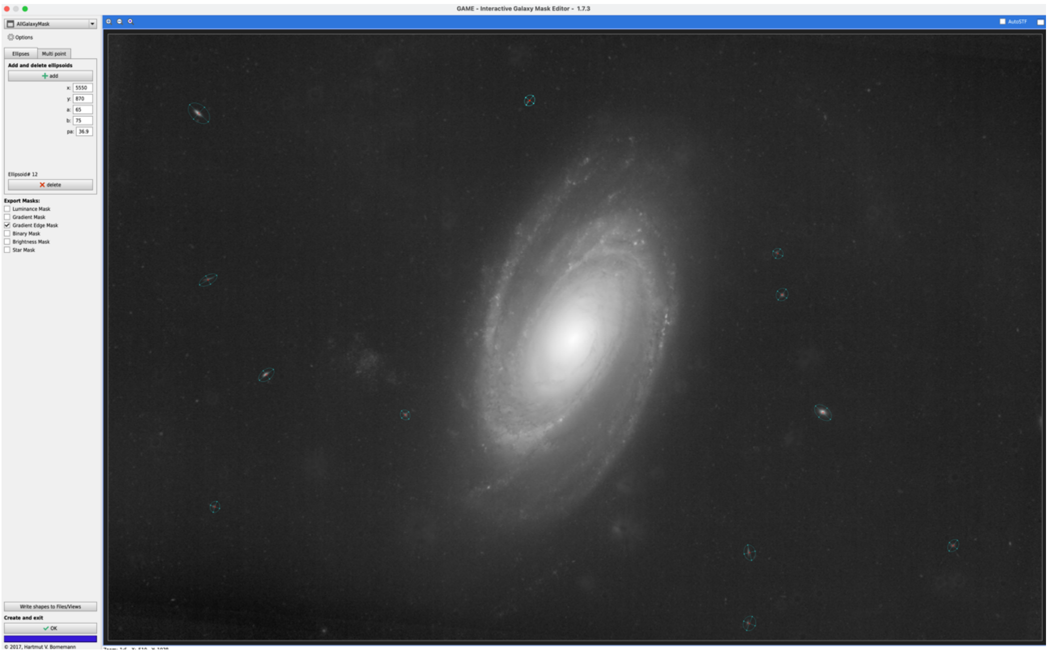
The galaxy or nebula mask will need to be improved and refined to reflect its intended process. The is a key step in the capability and intentions of the astrophotogrophers. The method of improving a galaxy or nebula mask will be tailored to the specific object and what we want to achieve in processing it. There is no pre-set workflow here, the astrophotogrophers has to figure out what they want themselves.

Every example will be different but here we use the real example of an image of galaxy M81.



In this initial galaxy-mask we can see a number of features in addition to galaxy M81:

- Top left – Galaxy PGC 28848, a 16th magnitude galaxy, 69mly away.
- Below PGC28848 – Galaxy PGC 5372321, a 18th magnitude spiral galaxy, 840mly away
- Just to the left of M81 – Galaxy UGC 5336 (PGC 28757) is an irregular dwarf galaxy companion of M81 at 13mly.
- To the left of UGC 5336 – Galaxy PGC 2726822 a 17th magnitude spiral galaxy at 870mly.
- Below UGC5336 – Galaxy PGC 3749345 a magnitude 17 spiral galaxy at 1,300mly.
- Right of M81 – Galaxy PGC 28505 a 15th magnitude spiral at 550mly.
- Numerous other faint and small galaxies.

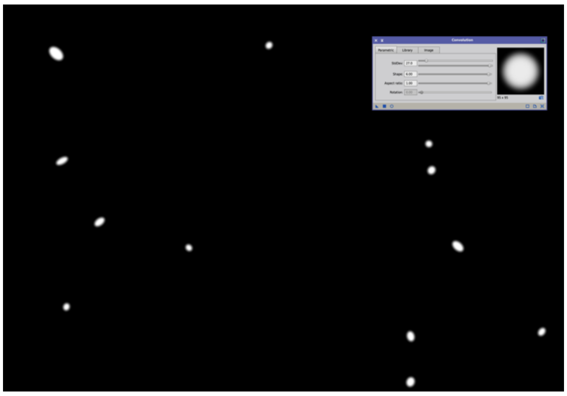

GAME - Interactive Galaxy Mask Editor - 1.7.3
AllGalaxyMask
Options
Ellipses
Multi point
Add and delete ellipsoids
add
delete
Write shapes to Files/Views
Create and exit
OK
AutoSTF
© 2017, Hartmut V. Bornemann

x: 5550
y: 870
a: 65
b: 75
pa: 36.9
Ellipsoid# 12

Export Masks:
- ☐ Luminance Mask
- ☐ Gradient Mask
- ☑ Gradient Edge Mask
- ☐ Binary Mask
- ☐ Brightness Mask
- ☐ Star Mask

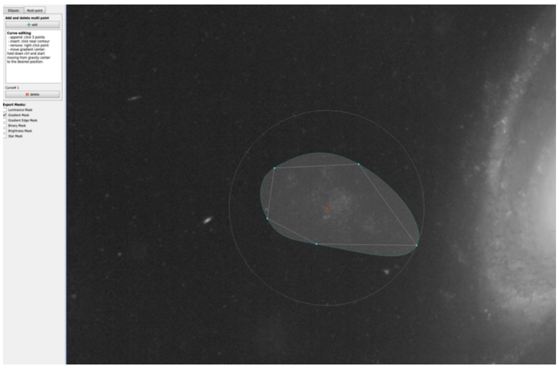
The GAME script is used to create and elliptical patches, with an edge-gradient over each of the identified small galaxies in the image.

The mask created in this way is softened with the convolution process to improve the gradient transition from white to black at the edge of each galaxy mask.



The GAME script can also be used to create a multi-point irregular mask to cover UGC 5336.

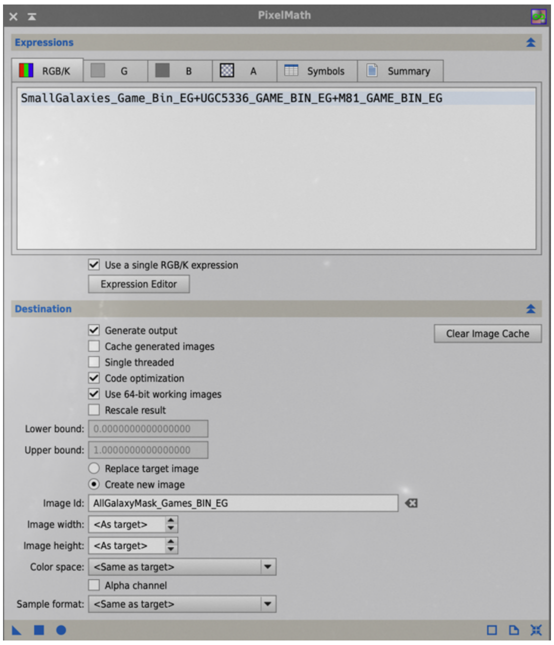
Finally, the process is repeated to create and elliptical mask to cover M81. Each individual mask has its edges ‘softened’ with the convolution tool.



An important feature of these masks is that the back ground is completely black with a pixel value of 0. In other words, completely transparent.

The separate masks created with the GAME script, the small galaxies plus UGC 5336 plus M81, can be combined, using PixelMath, to create a single, new, mask to cover all of the galaxies. The combined mask is shown below.

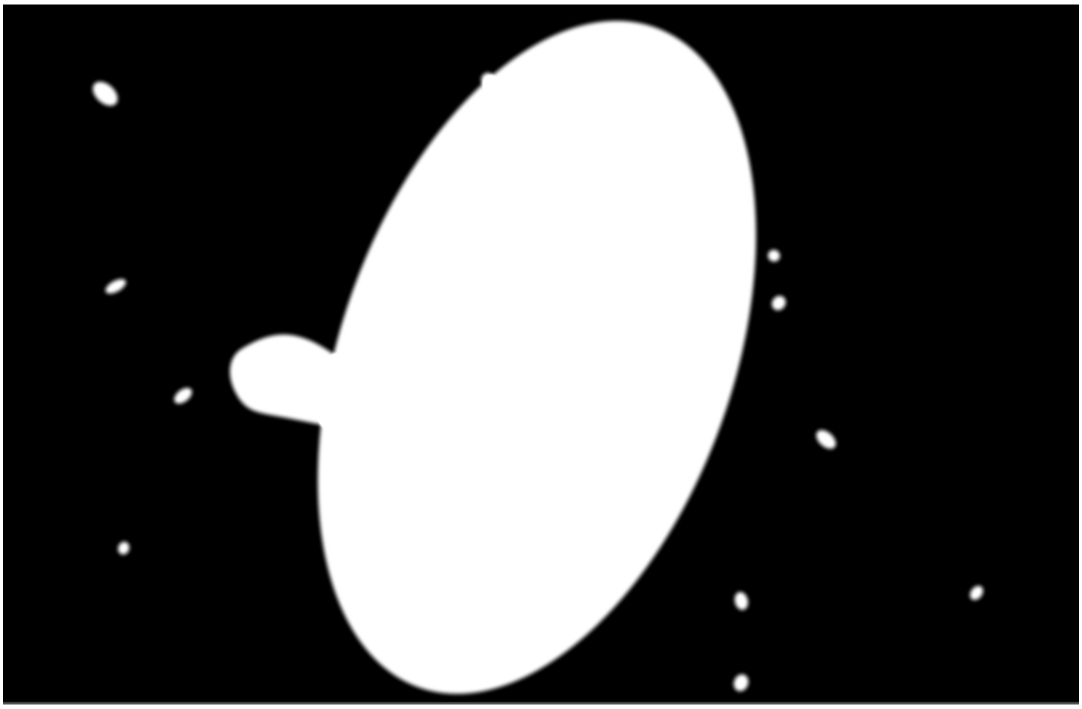
Pixel Math, https://pixinsight.com/doc/legacy/LE/22_pixel_math/pixel_math/pixel_math.html is a PI process that allows mathematical computations to be carried out on images at the level of individual pixels. In this example, the three GAME masks have simply been added together to create the new combined mask.



```
SmallGalaxies_Game_Bin_EG+UGC5336_GAME_BIN_EG+M81_GAME_BIN_EG
```

☑ Use a single RGB/K expression

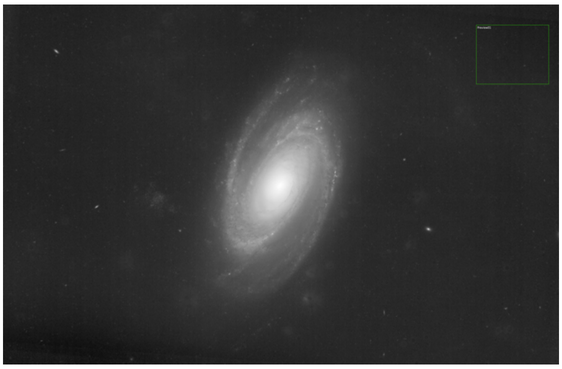
☑ Generate output
☐ Cache generated images
☐ Single threaded
☑ Code optimization
☑ Use 64-bit working images
☐ Rescale result
Lower bound: 0.0000000000000000
Upper bound: 1.0000000000000000
○ Replace target image
● Create new image
Image Id: AllGalaxyMask_Games_BIN_EG
Image width: <As target>
Image height: <As target>
Color space: <Same as target>
☐ Alpha channel
Sample format: <Same as target>



Correct the Galaxy Mask Background

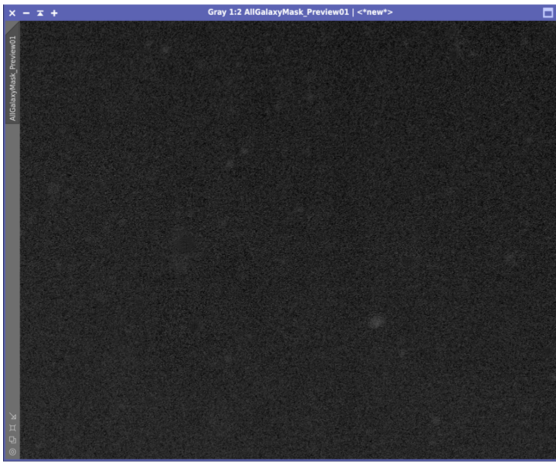
We can now correct processing artefacts in the galaxy mask sky background.

The first step is to create a PI 'preview' are on a reasonably clean and consistent area of the sky background of the galaxy-mask.



The ‘preview’ of the galaxy-mask background can be dragged as a tab of the main image and dropped onto the PI desktop. This creates a separate image ‘Preview 01’.

The ’Combined Galaxy GAME Mask’ is now applied to the original galaxy mask. The mask is arranged to protect the galaxies but allow processing of the background sky only.

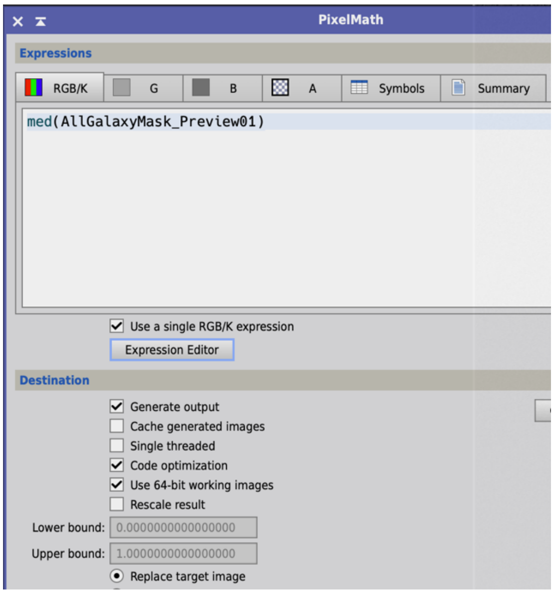


A new Pixel Math expression is used to calculate the median-pixel-value of the original galaxy mask sky background preview that we created a moment ago.

med(AllGalaxyMask_Preview01)

The median pixel value is used, rather than the mean value, because in skewed data distributions (which pixel value distributions usually are) the median value is more stable and less affected by outlier data, than the mean.

The Pixel Math expression is applied to the original galaxy mask.



```
med(AllGalaxyMask_Preview01)
```

☑ Use a single RGB/K expression

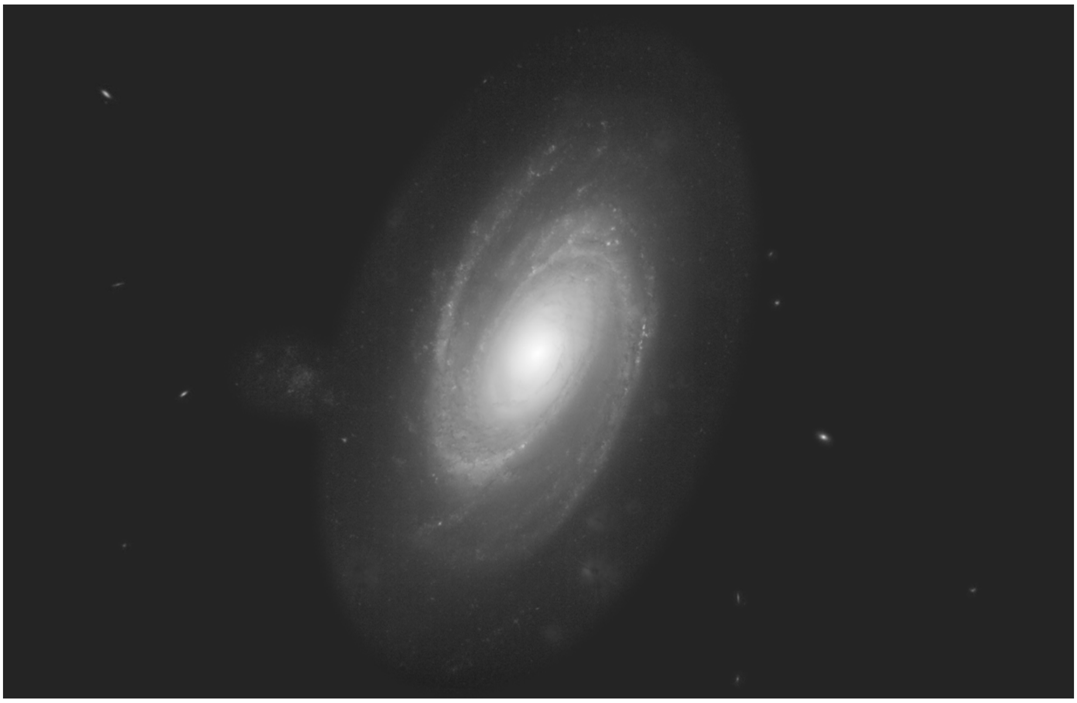
Expression Editor

Destination

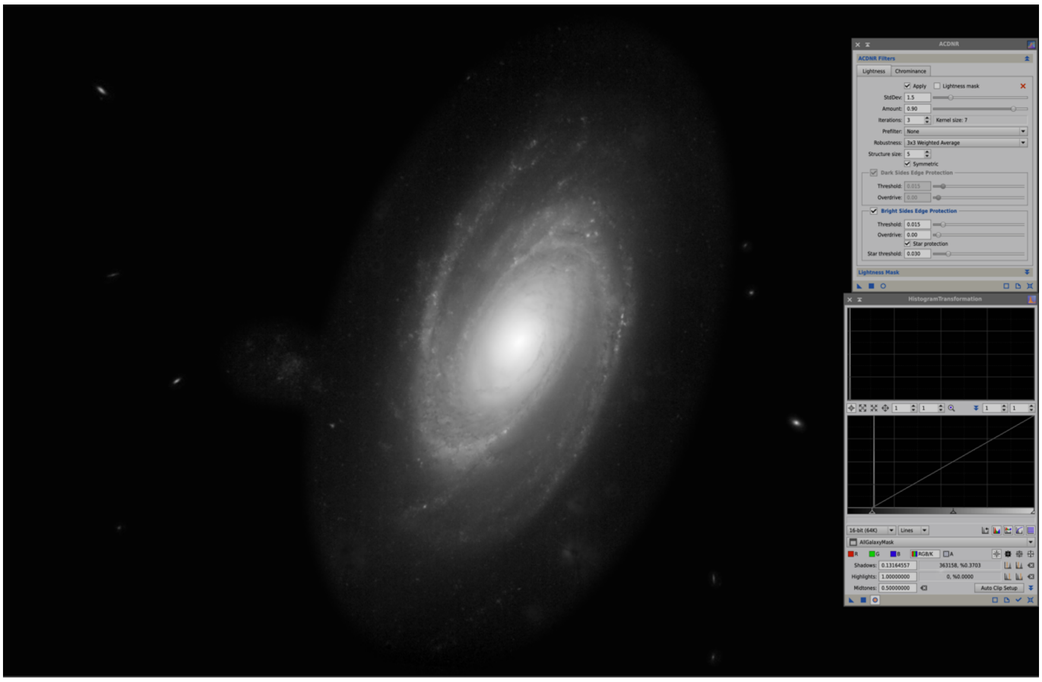
☑ Generate output
□ Cache generated images
□ Single threaded
☑ Code optimization
☑ Use 64-bit working images
□ Rescale result

Lower bound: 0.0000000000000000
Upper bound: 1.0000000000000000

● Replace target image

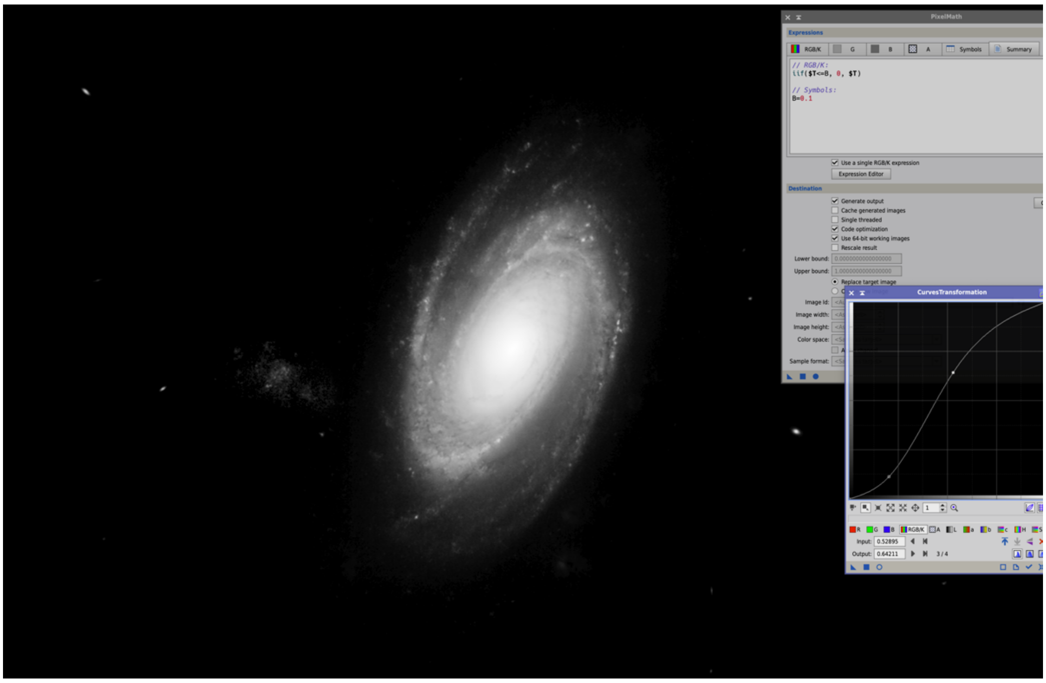


The galaxies are not affected by the Pixel Math expression because they were protected by the GAME mask. The sky background pixels, however, have all had their values changed to the median value of the sky background. The result is that artefacts, such as those created by the original integration, have all been smoothed out.



☑ Apply ☐ Lightness mask
StdDev: 1.5
Amount: 0.90
Iterations: 3 Kernel size: 7
Prefilter: None
Robustness: 3x3 Weighted Average
Structure size: 5
☑ Symmetric
☑ Dark Sides Edge Protection
Threshold: 0.015
Overdrive: 0.00
☑ Bright Sides Edge Protection
Threshold: 0.015
Overdrive: 0.00
☑ Star protection
Star threshold: 0.030

AllGalaxyMask
Shadows: 0.13164557 363158, %0.3703
Highlights: 1.00000000 0, %0.0000
Midtones: 0.50000000



```
// RGB/K:
iif($T<=B, 0, $T)

// Symbols:
B=0.1
```

☑ Use a single RGB/K expression
☑ Generate output
☐ Cache generated images
☐ Single threaded
☑ Code optimization
☑ Use 64-bit working images
☐ Rescale result
Lower bound: 0.0000000000000000
Upper bound: 1.0000000000000000
● Replace target image

Input: 0.52895
Output: 0.64211 3 / 4

With the GAME mask removed, a new Pixel Math expression is applied:

RGB/K iif($T<=B, 0, $T)

Symbol. B=0.1

In Pixel Math, $T refers to the value of the target pixel in the target image. This expression means:

If the value of the target pixel is less than or equal to the value of **B**, then set the target pixel value to **0**, otherwise leave it at the original value.

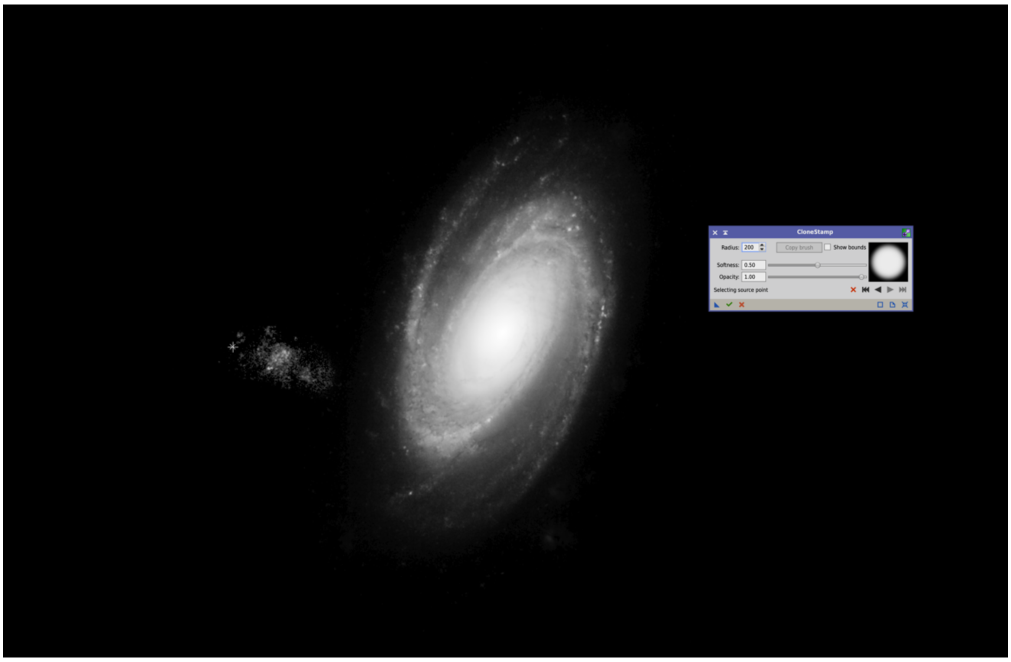
B is a constant whose value is set as a Pixel Math ‘Symbols’.

Pixel Math assigns every pixel a value between 0 and 1. 0 is a pure black pixel. 1 is a pure white pixel. By hovering the mouse over an image in PixInsight, the value of the pixel, **k**, is shown at the bottom of the image frame. This can be used to estimate the brightest pixel value in the image, that we wish to become part of the black background. In this example, that value, assigned was 0.1.

After applying the Pixel Math expression to the image, and applying a PI curve process https://pixinsight.com/doc/legacy/LE/17_curves/curves_transforms/curves_transforms.html the galaxy mask is complete. The background is pure black but all of the galaxies are bright greys or white.

Creating Individual Galaxy Masks

The All-Galaxy-Mask can be duplicated (cloned). The PI Clone Stamp process can be used to ‘paint-out’ the small galaxies to create a M81-Only mask. The same process can be used to create a mask only for the small galaxies and a UGC5336-Only mask.



CloneStamp

Radius: 200

Copy brush

□ Show bounds

Softness: 0.50

Opacity: 1.00

Selecting source point

At this stage we will have a complete set of galaxy masks. It is helpful to create and keep these masks in a PI masks ‘Workspace’ that is separate to the image processing ‘Workspace’. This well help to keep the processing workflow organised and uncluttered.

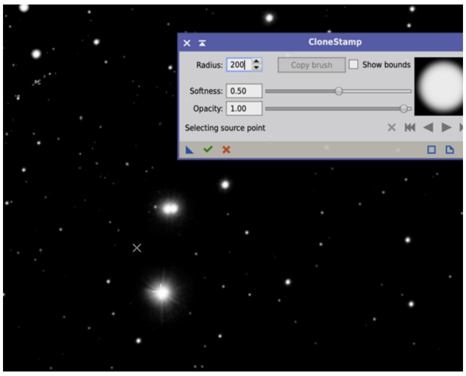
All masks are saved as 64-bit XISF image files.

Improving Star Masks

The original star-mask can also be improved. The first improvement is to create separate masks for the large and small stars.

A mask only consisting of small stars is easily created by making a clone of the full star mass and using the 'Clone Stamp' process to paint out the big stars.

Once a 'Small Star Mask' has been created from a clone of the original star-mask, then a 'Large Star Mask' (a mask of the large stars) can be created using Pixel Math to subtract the 'small star mask' from the original star-mask.



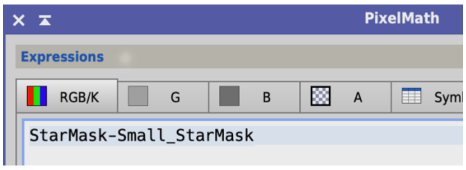
CloneStamp

Radius: 200

Softness: 0.50

Opacity: 1.00

Selecting source point



PixelMath

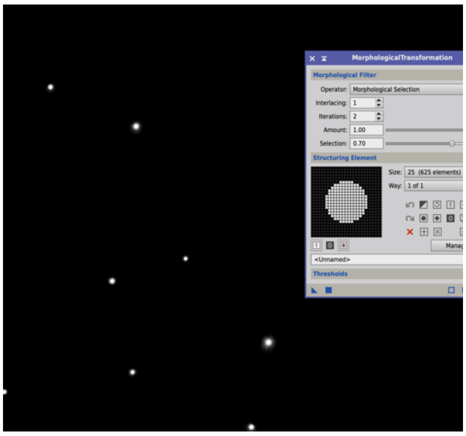
Expressions

```
StarMask-Small_StarMask
```

Each star-mask is further improved by creating a clone of the star-mask and increasing the star sizes in the clone by using the Morphological Transformation Process https://www.youtube.com/watch?v=VeGRDrZWxUQ

Select the 'Morphological Selection' operator and set the 'Selection' slider to 0.75.

Apply the process to increase the star sizes of the cloned mask.



MorphologicalTransformation

Morphological Filter

Operator: Morphological Selection

Interlacing: 1

Iterations: 2

Amount: 1.00

Selection: 0.70

Structuring Element

Size: 25 (625 elements)

Way: 1 of 1

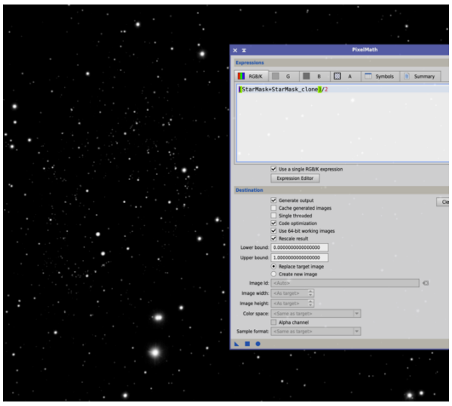
<Unnamed>

Thresholds

Use a Pixel Math expression to combine the original star-mask with the cloned mask with enlarged stars:

(StarMask + StarMask_Clone)/2

This will create stars with the original cores but the enlarged parts as a 50% grey colour.



PixelMath

Expressions

```
(StarMask+StarMask_clone)/2
```

☑ Use a single RGB/K expression

Destination

☑ Generate output

□ Cache generated images

□ Single threaded

☑ Code optimization

☑ Use 64-bit working images

☑ Rescale result

Lower bound: 0.0000000000000000

Upper bound: 1.0000000000000000

● Replace target image

○ Create new image

Image Id: <Auto>

Image width: <As target>

Image height: <As target>

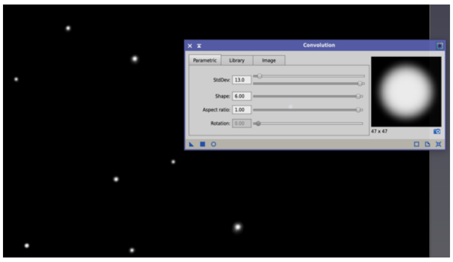
Color space: <Same as target>

□ Alpha channel

Sample format: <Same as target>

Finally apply a convolution to soften or blur the edges of the stars to create a smooth transition between the masked and non-masked areas.

All masks should have a final convolution applied to create a smooth transition of masked areas.



Convolution

StdDev: 13.0

Shape: 6.00

Aspect ratio: 1.00

Rotation: 0.00

47 x 47

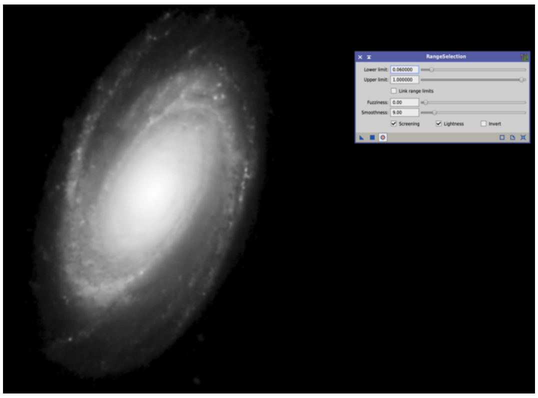
We will now have a set of star-masks.

Creating Range Masks for High Dynamic Range Objects

For very high dynamic range objects, such as globular clusters or galaxies with bright cores, it may be helpful to create a sequence of range masks for that object.

Starting with the M81-Only mask, the 'Range Selection' process is used to increase the 'lower limit'. These sets pixel values below the lower limit setting to a value of 0.

When the 'Range Selection' process is applied a new M81 mask will be created consisting only of the brighter pixel values not excluded by the process.



RangeSelection

Lower limit: 0.060000

Upper limit: 1.000000

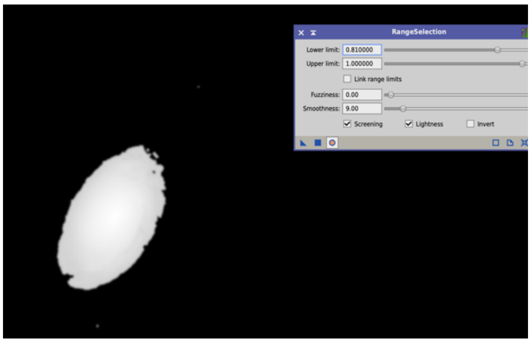
□ Link range limits

Fuzziness: 0.00

Smoothness: 9.00

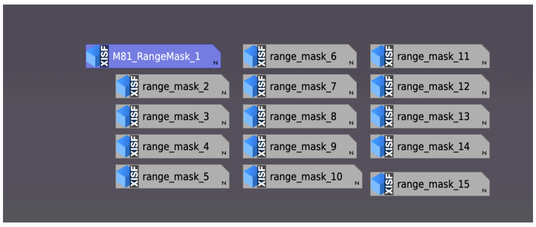
☑ Screening ☑ Lightness □ Invert

The ‘Range Selection’ ‘lower limit’ can then be further raised to create another M81 mask that is constrained further in terms of only the brightest pixels being defined by the mask.

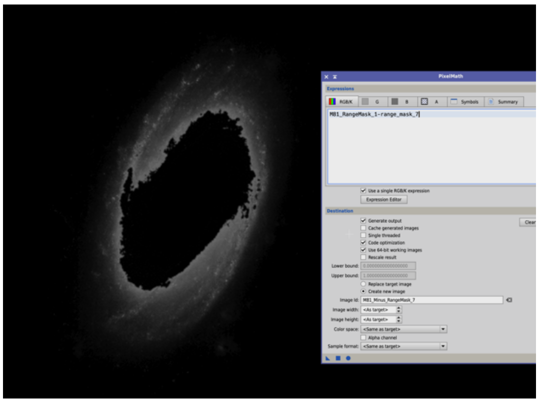

RangeSelection

Lower limit: 0.810000
Upper limit: 1.000000
□ Link range limits
Fuzziness: 0.00
Smoothness: 9.00
☑ Screening ☑ Lightness □ Invert

The process is repeated several times to create a series of range masks that only include an increasing level of pixel value. The final mask will only cover the brightest pixels in the nucleus of the galaxy.


M81_RangeMask_1
range_mask_2
range_mask_3
range_mask_4
range_mask_5
range_mask_6
range_mask_7
range_mask_8
range_mask_9
range_mask_10
range_mask_11
range_mask_12
range_mask_13
range_mask_14
range_mask_15

The range masks can be modified by subtracting each one from the full M81 mask (using PixelMaths) to create a range of masks that will allow process to be applied only to the outer arms or only to the inner core.


PixelMath

Expressions

```
M81_RangeMask_1-range_mask_7
```

☑ Use a single RGB/K expression

Destination

☑ Generate output
□ Cache generated images
□ Single threaded
☑ Code optimization
☑ Use 64-bit working images
□ Rescale result
Lower bound: 0.0000000000000000
Upper bound: 1.0000000000000000
○ Replace target image
● Create new image
Image id: M81_Minus_RangeMask_7
Image width: <As target>
Image height: <As target>
Color space: <Same as target>
□ Alpha channel
Sample format: <Same as target>

Once a bespoke full set of masks have been created for an image, the non-linear processing can begin.

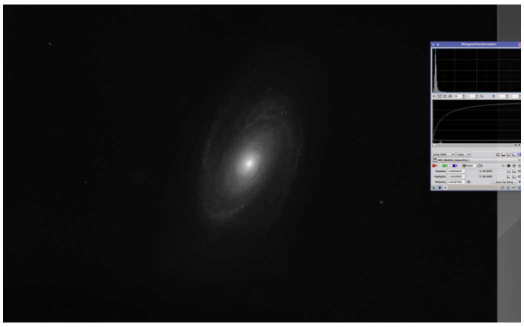
Non-Linear Processing

The Galaxy

Histogram Transformation of the Luminous Image

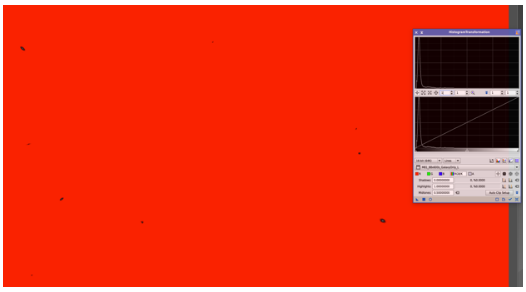
The PI Histogram Transformation (HT) process is used to apply an initial non-linear stretch to the image.

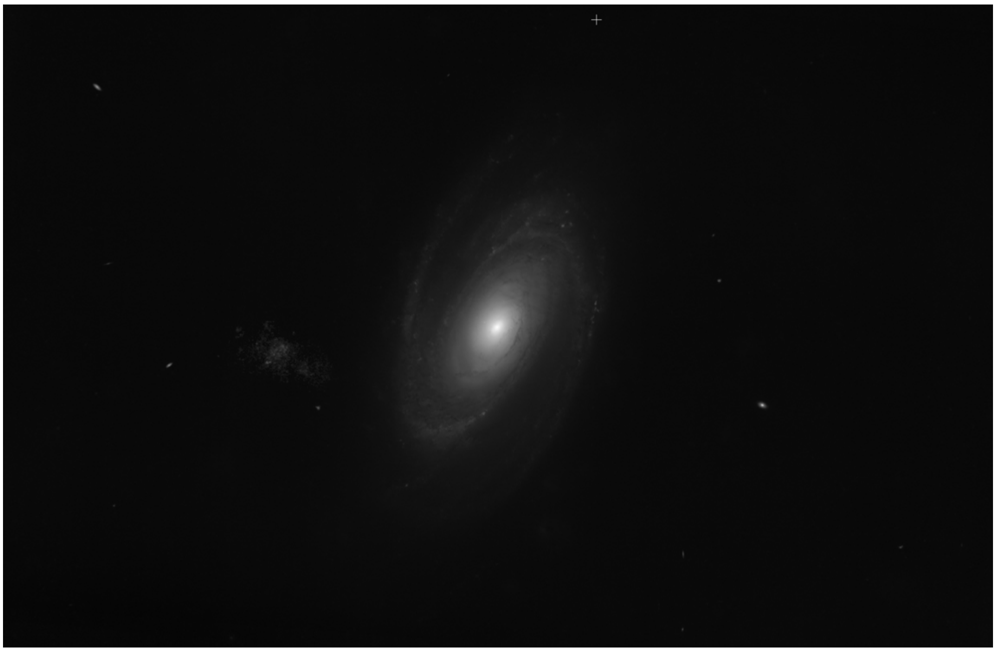
For high dynamic range objects, such as galaxies, the initial stretch is restricted so that the brightest part of the galaxy is not over exposed.



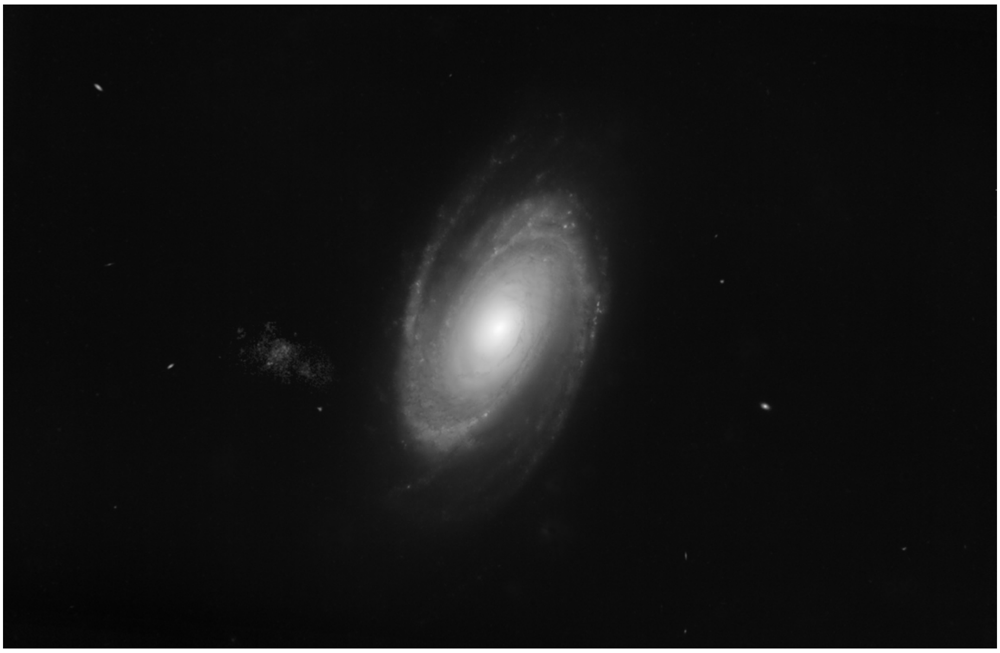
Selective masks can be applied to selectively enhance the brightness of objects. In this case, the small background galaxies are being brought our using the HT process.

The fainter galaxies and the brightest areas of the main galaxy are exposed in this way.





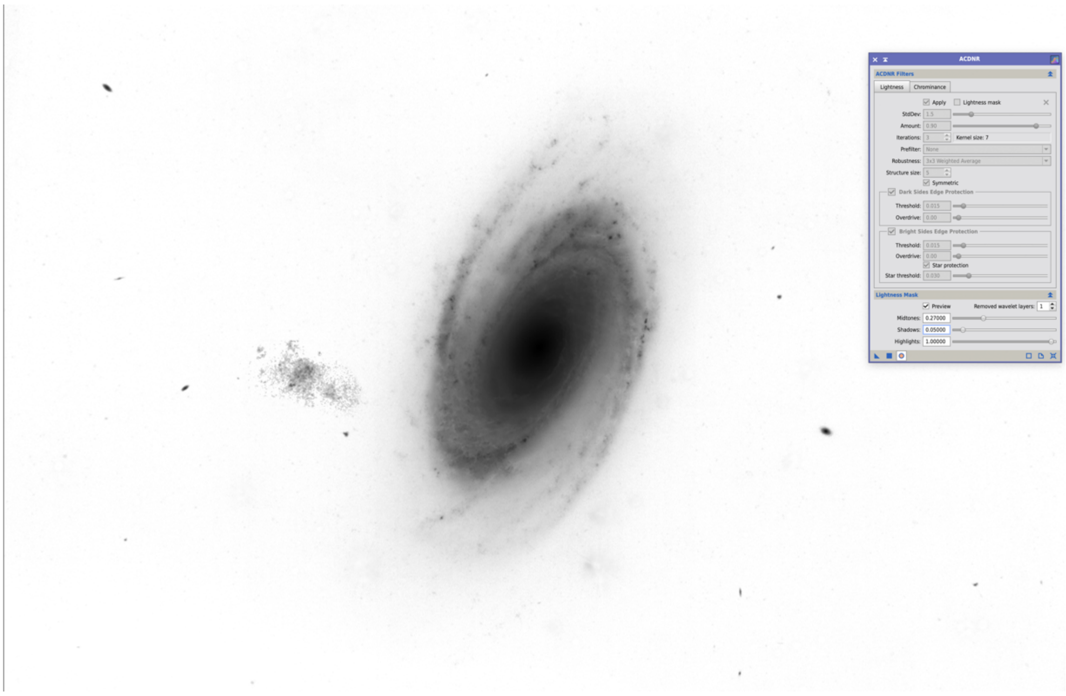
The main galaxy is now further exposed using the sequence of range masks. Selective use of range masks allow the fainter parts of the galaxy to be selectively brightened without over exposing the bright inner core at the same time.



Selective use of range masks allows the fainter parts of the galaxy to be selectively brightened without over exposing the bright inner core at the same time.

Noise Reduction with ACDNR & TGVDenoise of the Luminous Image

https://pixinsight.com/tutorials/nr-comparison/



ACDNR

ACDNR Filters

Lightness | Chrominance

☑ Apply □ Lightness mask

StdDev: 1.5

Amount: 0.90

Iterations: 3 Kernel size: 7

Prefilter: None

Robustness: 3x3 Weighted Average

Structure size: 5

☑ Symmetric

☑ Dark Sides Edge Protection

Threshold: 0.015

Overdrive: 0.00

☑ Bright Sides Edge Protection

Threshold: 0.015

Overdrive: 0.00

☑ Star protection

Star threshold: 0.030

Lightness Mask

☑ Preview Removed wavelet layers: 1

Midtones: 0.27000

Shadows: 0.05000

Highlights: 1.00000

Non-linear noise reduction is applied in turn to the L and RGB images using the ACDNR Process.

For the L image, an ACDNR ‘lightness mask’ is created and applied to the image. This protects the signal areas from the noise reduction algorithm.

For the RGB image, the ‘Chrominance’ tab is used in the *ACDNR Process* without the need for a mask. We are not concerned about the noise reduction removing small scale structure in the RGB image as this information with be provided by the L image when it is combined with the RGB image at a later stage.

The noise reduction will remove or blur small scale structure random noise from all areas of the image not protected by a mask.

A second de-noise algorithm, *TGVDenoise Process* is also applied to each image. The detailed signal in the L image is this time protected by applying a luminous mask. The process is applied to the RGM image without the need for a mask.

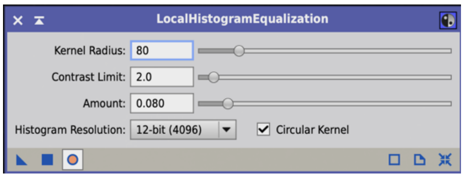
Re-Setting the Black-Point

Following the application of the de-noise processes, the black point in the L and RGB images is reset using the *Histogram Transfer Process*.

At this stage we have four high dynamic range image components, the luminous (L) component and the colour (RGB) component for each of the galaxy/nebula images and the separate star images.

Local Histogram Equalisation of the Galaxy Centre

With selective use of masks to only process the detailed structures in M81, the Local Histogram Equalization process is used to enhance the detailed structures of the galaxy.



LocalHistogramEqualization

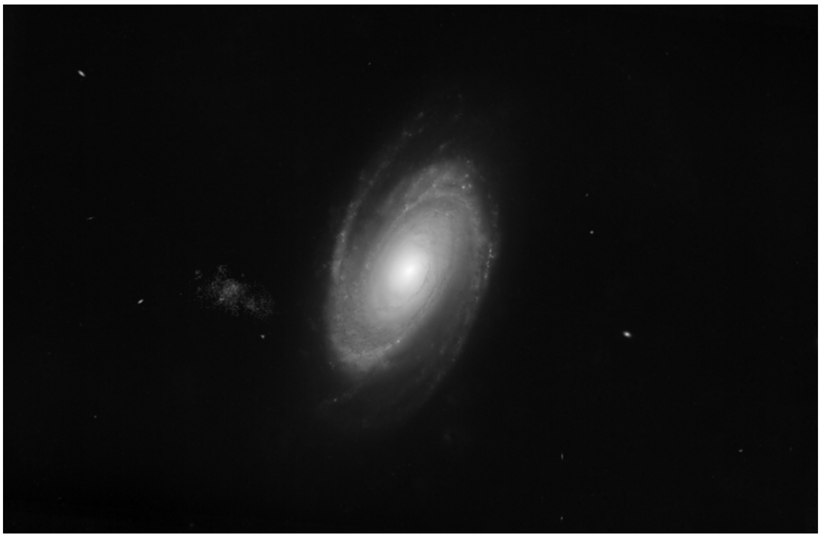
Kernel Radius: 80

Contrast Limit: 2.0

Amount: 0.080

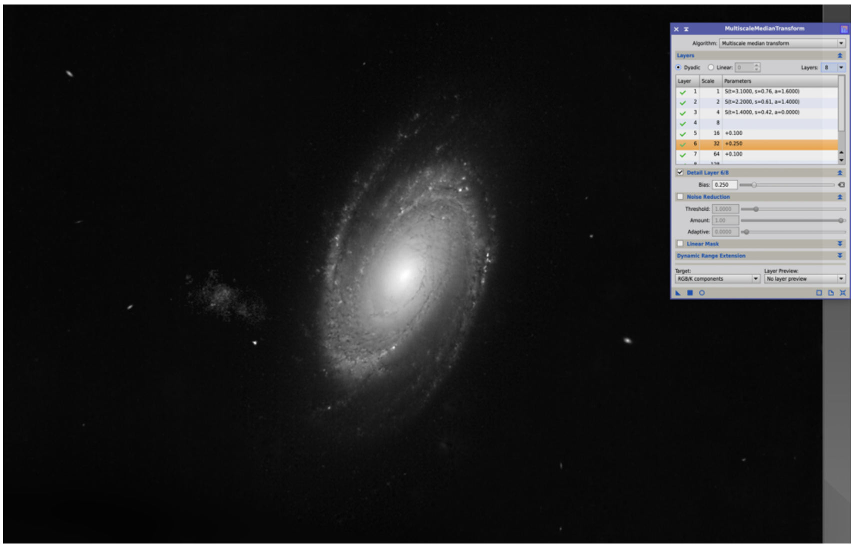
Histogram Resolution: 12-bit (4096)

☑ Circular Kernel



The luminous ‘galaxy-only’ image is now processed to provide a high dynamic range image of M81 and the fainter galaxies.

Multiscale Median Transform to Sharpen the Galaxy



MultiscaleMedianTransform

Algorithm: Multiscale median transform

Layers

● Dyadic ○ Linear: 0 Layers: 8

	Layer	Scale	Parameters
✓	1	1	S(t=3.1000, s=0.76, a=1.6000)
✓	2	2	S(t=2.2000, s=0.61, a=1.4000)
✓	3	4	S(t=1.4000, s=0.42, a=0.0000)
✓	4	8	
✓	5	16	+0.100
✓	6	32	+0.250
✓	7	64	+0.100

☑ Detail Layer 6/8

Bias: 0.250

□ Noise Reduction

Threshold: 1.0000

Amount: 1.00

Adaptive: 0.0000

□ Linear Mask

Dynamic Range Extension

Target: RGB/K components

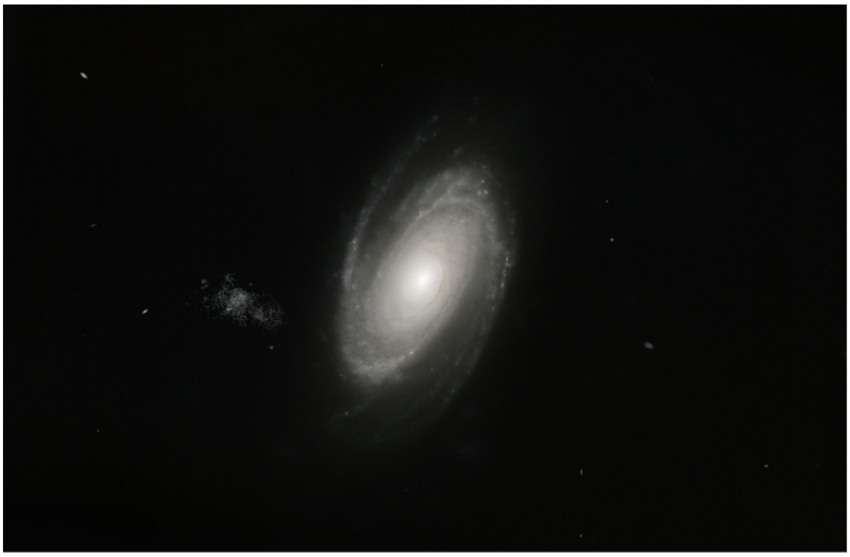
Layer Preview: No layer preview

One final step for the galaxy-only LRGB image is to further sharpen detail in the galaxy using the ‘MultiScale Median Transform’ process.

Use of the ‘Detail’ sliders in the MMT process, for the appropriate wavelet layers, can significantly enhance the detail in the galaxy. Use a mask to ensure that the process is only applied to M81.

Histogram Transformation of the RGB Image

The RGB-Galaxy-Only image is now selectively non-linearly stretched with the Histogram Transformation process and selective use of masks in the same way that the luminous image was stretched.

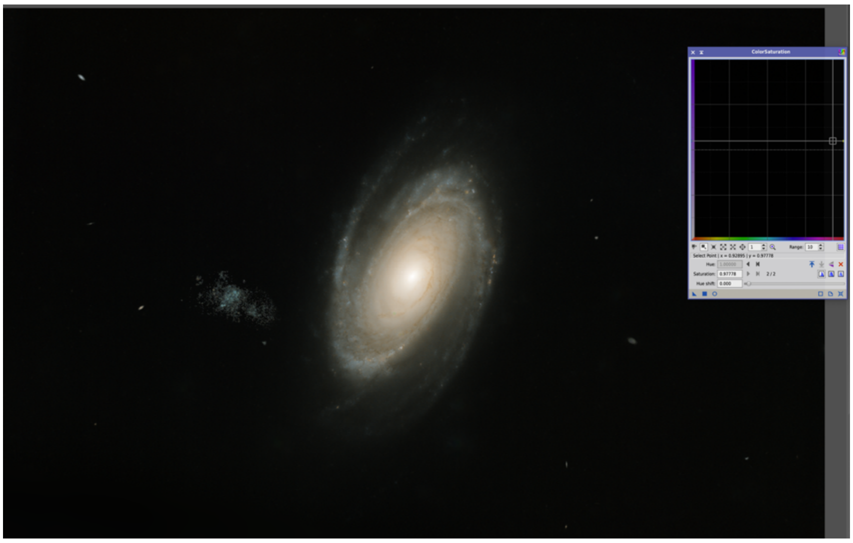


Chrominance noise reduction is applied to the stretched RGB image using ACDNR and/or TGVDenoise. No mask protection is used during noise reduction with the RGB image.

After noise reduction, the HT process is used to re-set the black-point for the RGB image.

Colour Saturation Process

The Colour Saturation process is used to improve the colour saturation of the galaxy. This is an aesthetic and subjective decision. The selective use of masks allows different levels of colour saturation to be applied to different parts of the image.

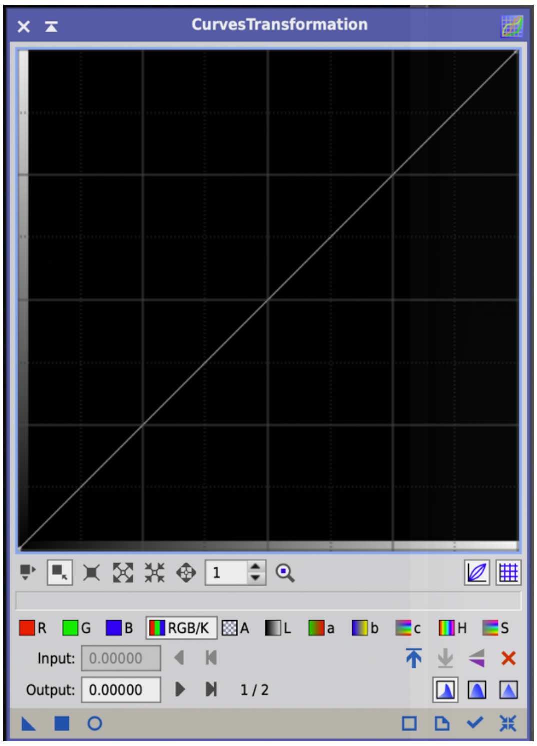


Colour Saturation and Curves Transformation

The 'Curves Transformation' process is a versatile and powerful process for simultaneously adjusting up to 11 different channels or characteristics of the image:

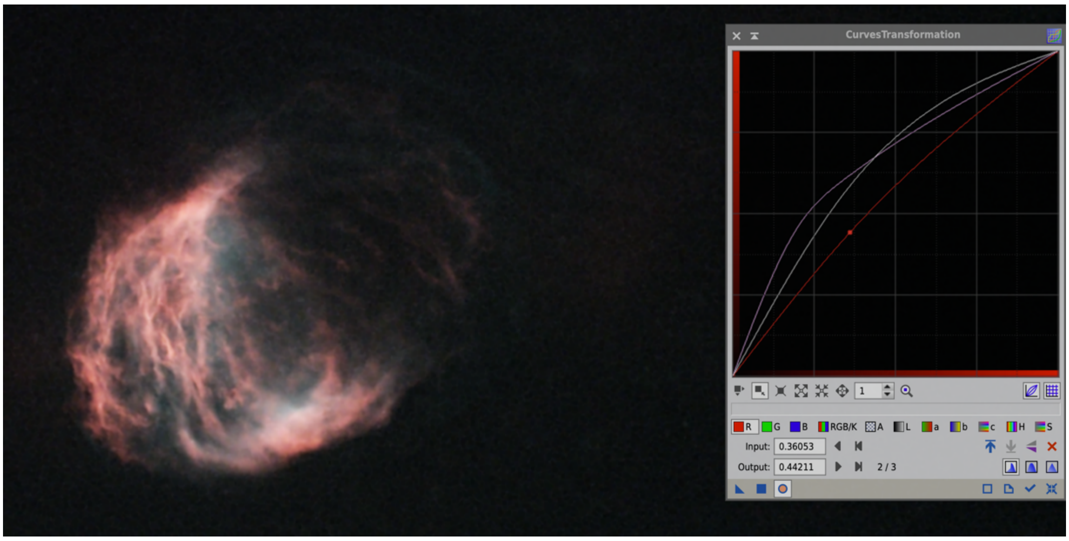
- Red
- Green
- Blue
- RGB/Grey
- Alpha
- CIE L
- CIE A
- CIE B
- CIE C
- Hue
- Saturation

The curves provides a non-linear transformation of the relevant channel.



Input: 0.00000

Output: 0.00000



Input: 0.36053
Output: 0.44211
2 / 3

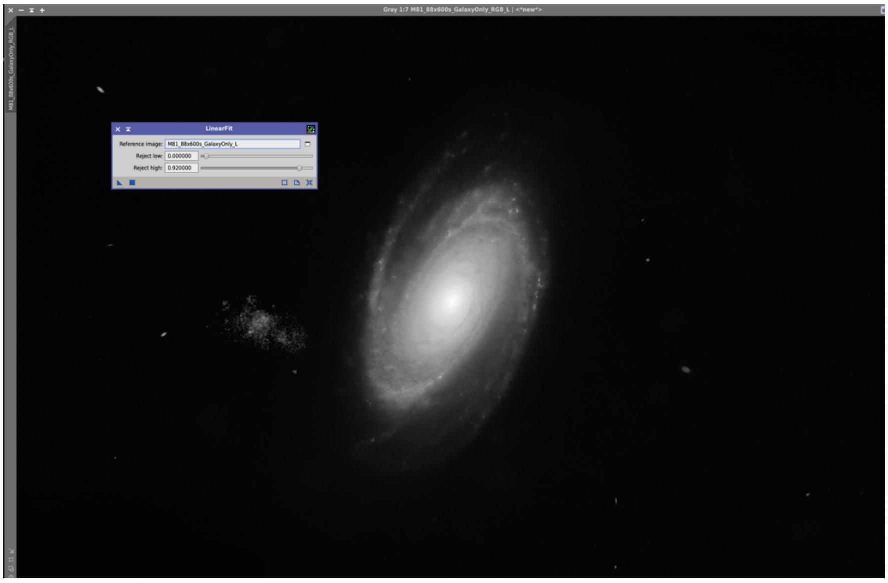
In the example above the red, CIE L and saturation curves have all been adjusted to give the optimum result.

Galaxy LRGB Combination

The separate non-linear processing of the L and RGB components is completed. With the two components as processed the next stage will be to combine them in to a single LRGB image. The LRGB combination consist of three distinct process steps.

Linear Fit

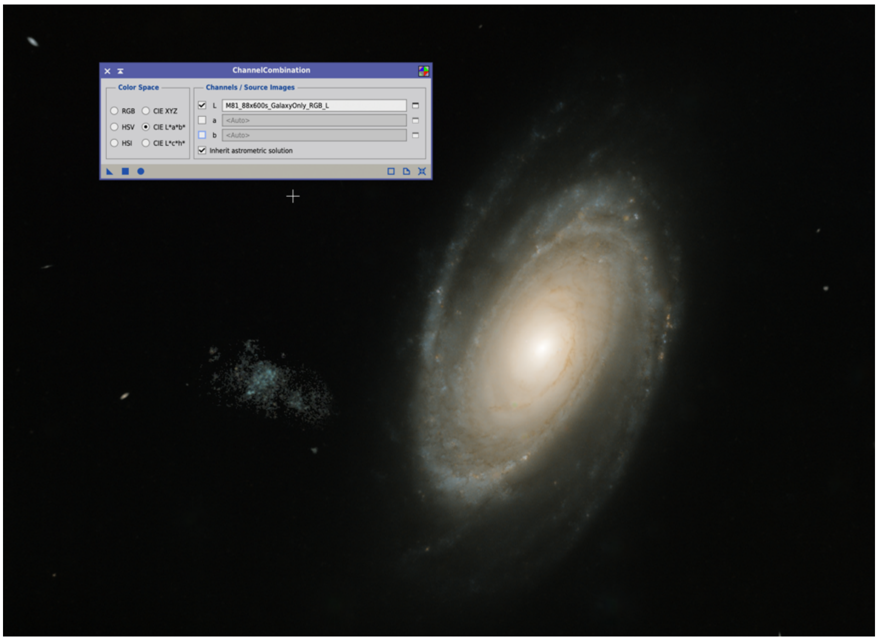
The first two steps of the process aim to secure a match of the overall luminosity levels of the RGB image to the L image.


Gray 1:7 M81_88x600s_GalaxyOnly_RGB_L | <*new*>
M81_88x600s_GalaxyOnly_RGB_L
LinearFit

Reference image: M81_88x600s_GalaxyOnly_L
Reject low: 0.000000
Reject high: 0.920000

Step 1 – A luminous clone of the RGB image is first extracted. The *Linear Fit Process* is used to match the luminosity of the RGB-L clone to the L image.

Channel Combination


ChannelCombination

Color Space

○ RGB ○ CIE XYZ
○ HSV ● CIE L*a*b*
○ HSI ○ CIE L*c*h*

Channels / Source Images

☑ L M81_88x600s_GalaxyOnly_RGB_L
□ a <Auto>
□ b <Auto>
☑ Inherit astrometric solution

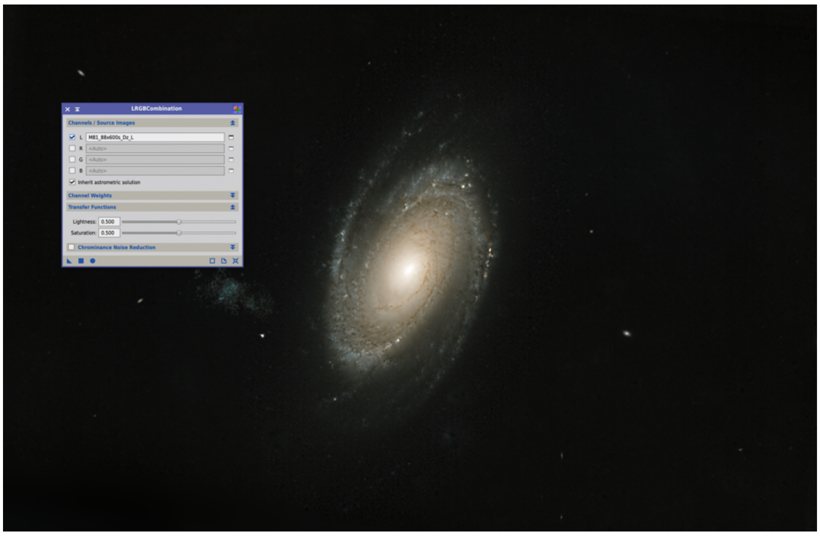
Step 2 – The *Channel Combination Process* is used to re-combine the RGB-L clone with the RGB image.

Once this process has been completed, the RGB-L clone can be deleted.

The overall luminosity of the RGB image will now match that of the L image.

LRGB Combination

Step 3 – The LRGB Process is used to combine the L-component into the RGB-component. This creates a single LRGB image.


LRGBCombination

Channels / Source Images

☑ L M81_88x600s_Dz_L
□ R <Auto>
□ G <Auto>
□ B <Auto>
☑ Inherit astrometric solution

Channel Weights

Transfer Functions

Lightness: 0.500
Saturation: 0.500

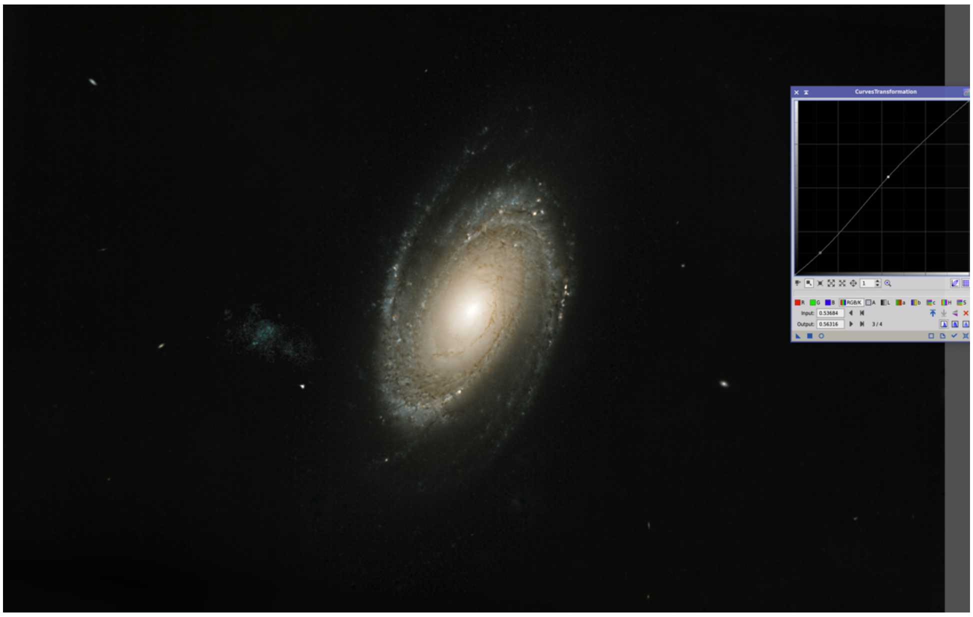
□ Chrominance Noise Reduction

The LRGB combination process is used to combine the L and RGB images of the galaxy or nebula. It is also used to combine the L and RGB images of the stars.

At this stage, we have now an LRGB image of the galaxy/nebula.

Curves Transformation

The final step in processing the data is a Curves Transformation Process of the LRGB image to improve contrast and aesthetics.



Input: 0.53684

Output: 0.56316

Processing the Stars

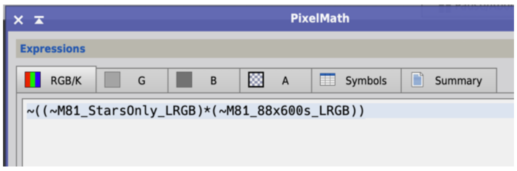
The separate luminous and RGB star-only images are now processed in much the same way as the galaxy-only images.

At the end of this stage we will have a completed stars-only LRGB image alongside our galaxy-only LRGB image.

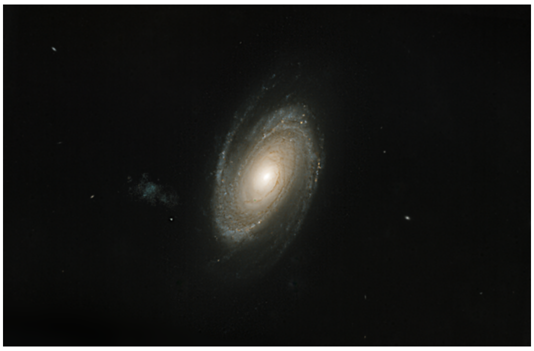
Combining the Galaxy and Stars

The final task is to combine a non-linear and fully processed LRGB star image with the starless galaxy/nebula image.

The range of pixel values in the star image will be between 0 and 1.



```
~((~M81_StarsOnly_LRGB)*(~M81_88x600s_LRGB))
```



We cannot simply add the images together as this would saturate and clip some pixel values losing data. For example, combining a pixel with a value of 0.75 in the star image with a pixel value of 0.5 in the starless image, will create a combined value of 1.25 that will be converted back to a maximum of 1. We will have lost the some of the information.

Image Combination

The PixelMath function will combine a star-only image with a starless image, without the result being able to exceed 1.0

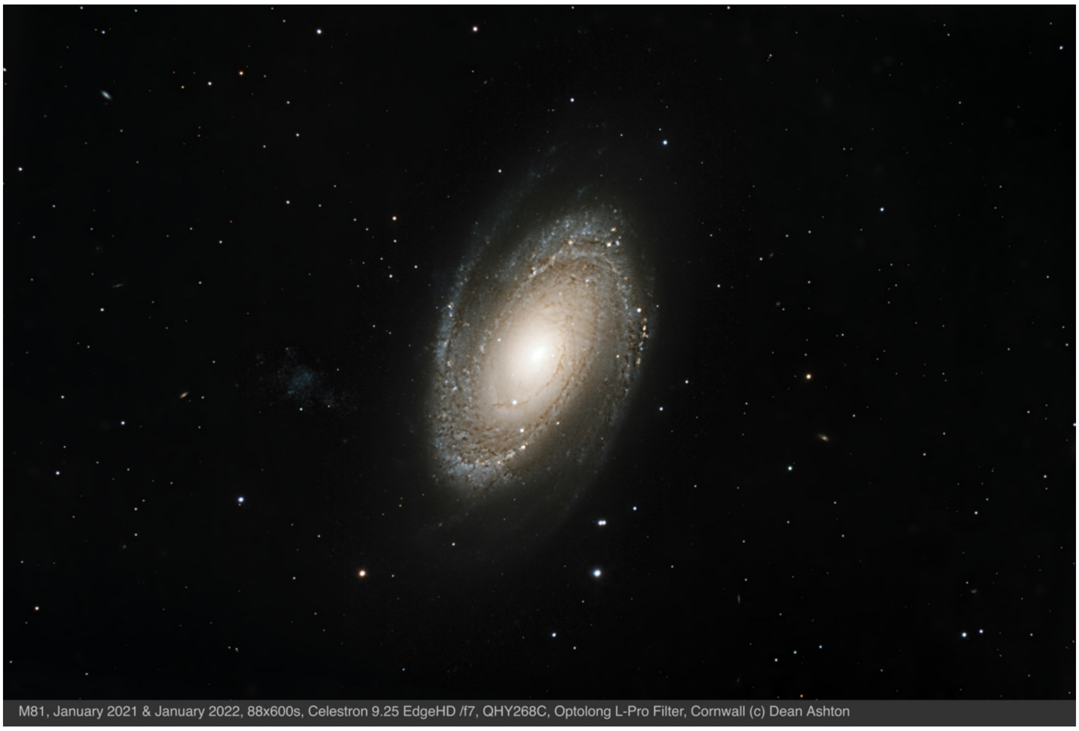
~(~Starless*~StarsOnly)

The ~ function in PixelMath represents the 1-*x* function, where *x* is the operand that the function operates on. This effectively inverts the pixel value. Black (0) becomes white (1) and via versa.

The function is the equivalent of:

1-((1-Starless)*(1-StarsOnly))

'1-Starless' inverts the Starless image pixel values. Bright pixels become dim ones and all values will be between 1 and 0. Multiplying the inverted starless and star-only values will give a smaller value between 0 and 1, I.e. it will be dimmer. Inverting this product value again with 1-(a*b) will cause the dimmer product to now be brighter, but with a value of less than 1.



M81, January 2021 & January 2022, 88x600s, Celestron 9.25 EdgeHD /f7, QHY268C, Optolong L-Pro Filter, Cornwall (c) Dean Ashton

PixInsight PixelMath

Introduction

PixInsight's PixelMath performs a series of pixel-level arithmetic and logical operations between images. Basically, an instance of PixelMath contains a set of references to operand images, the operations that must be performed between them, and some parameters to control how the result image is generated.

PixelMath Values

PixelMath converts the ADC value of each pixel to the corresponding value on a 0-1 scale. 1 equates to a saturated or full pixel, white. 0 equates to an empty pixel, black. The output of all PixelMath functions will be a value in the range 0-1.

Target Image and Operands

The image an instance of PixelMath is applied to is called the *target image* of the process. From PixelMath's point of view, the target image is just a placeholder variable: it is unknown until the process instance is executed, so it isn't included as an actual parameter.

The rest of elements that participate in the expression being calculated are *operands*. Operands can be either images or constant values. For example, if you want to subtract the image identified as 'Image01' from any target image, the required PixelMath instance could be represented as:

<*target*> - Image01

where:

<*target*> identifies the target image, which is unknown until execution.

- stands for the subtraction operator.

Image01 is the identifier of an operand image.

When no image is used for an operand, a different constant value is used for each channel of the target image. For example, if the target is a RGB color image, a PixelMath instance like:

<*target*> * (0.95, 1.00, 0.98)

might be used to apply a linear colour correction to the target image. * stands for the multiplication operator. For grayscale images, a single constant is used.

In addition, target and operand images can be inverted. Inversion always takes place before the operations associated to operands. If p is a target or operand pixel, and inversion has been selected for the target or the corresponding operand image, then

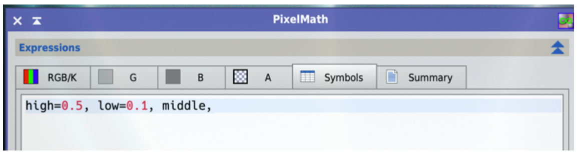
~p = **1 — p**

where **~p** is the inverted pixel used to operate.

Symbols

Symbols in PixelMath are the equivalent of variables in programming language.

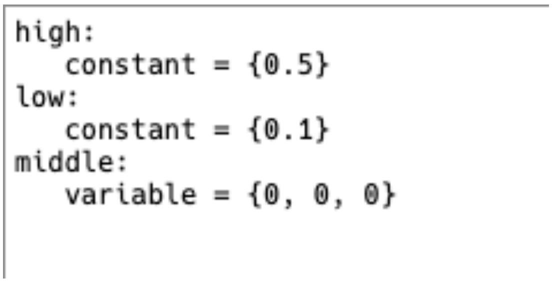
Symbols are defined and initiated in the PixelMath expression editor. The symbols must be separated by a comma.



```
high=0.5, low=0.1, middle,
```

In this example, the symbols ‘high’ and ‘low’ have been defined with a single constant value that will be applied to all three channels (R, G and B).

The symbol ‘middle’ has been initiated with assigning a constant value. PixelMath will interpret this initiation as setting a separate value of 0 to each of the three channels in preparation for a value being set through an expression.



```
high:
   constant = {0.5}
low:
   constant = {0.1}
middle:
   variable = {0, 0, 0}
```

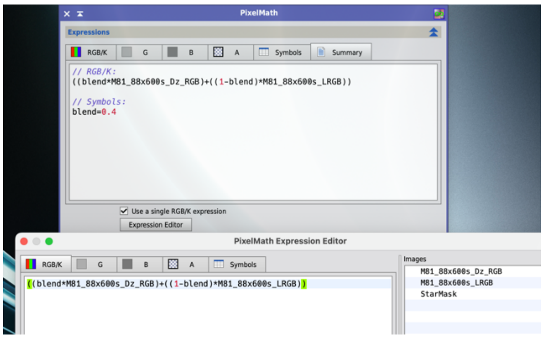
Important built-in symbol, known as metasymbols are:

- **$T** references the target image or target pixel.
- **$A** references the active image in the current workspace.
- **$M** references the mask image currently selected for the target image.

Expressions

In the PixelMath process window the ‘Use a single RGB/K expression’ box is ticked to apply a single expression to all three channels in an RGB channel. If the box is unticked, different expressions can be applied to each channel separately.

The PixelMath expression editor will display all images that are currently open in PixInsight. In this example the following two images and a star-mask are available.

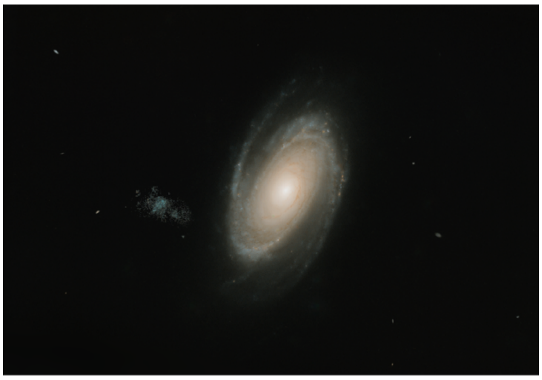


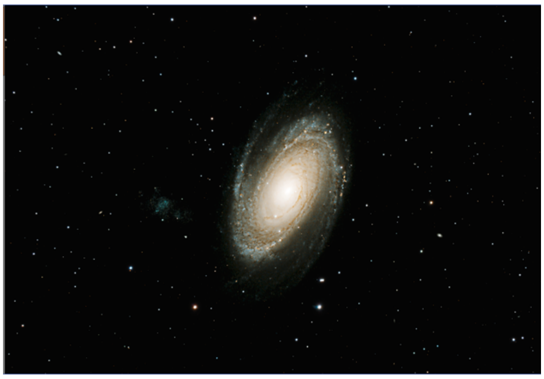
```
// RGB/K:
((blend*M81_88x600s_Dz_RGB)+((1-blend)*M81_88x600s_LRGB))

// Symbols:
blend=0.4
```

☑ Use a single RGB/K expression

```
((blend*M81_88x600s_Dz_RGB)+((1-blend)*M81_88x600s_LRGB))
```

Images
M81_88x600s_Dz_RGB
M81_88x600s_LRGB
StarMask





In this case, the image on the right ‘M81_88x600s_LRGB’ is the **target** image. The image on the left ‘M81_88x600s_Dz_RGB’ will be used as an **operand** image. The symbol **’blend’** has been defined and given a value of 0.4.

An expression has been written to blend the two Images:

((blend * M81_88x600s_Dz_RGB) + ((1 – blend) * M81_88x600s_LRGB))

Symbol: blend=0.4

A star-mask has been applied to the target image to prevent the expression applying to the stars. The expression blends the two images applying 40% of the left-hand image and 60% of the right-hand image to modify the target image.

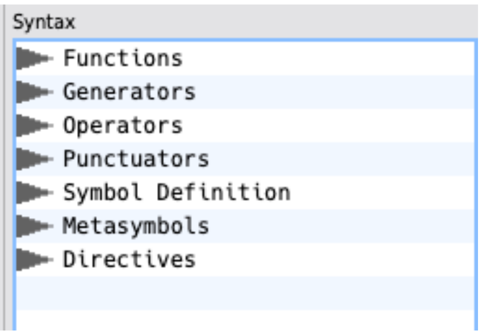
The PixelMath expression can be applied to either replace the existing target image or to create a new image.

The expression above could also have used the metasymbols $T or $A to achieve the same effect.

((blend * M81_88x600s_Dz_RGB) + ((1 – blend) * $T))

Symbol: blend=0.4

The PixelMath expression editor includes the full list of functions, opeartors, punctuators, pre-defined symbols and metasymbols. All of these can be selected from the drop-down lists to reveal their definition or explanation. All of these can be added to an expression by double clicking on them in the drop-down list or simply typing them into the expression.



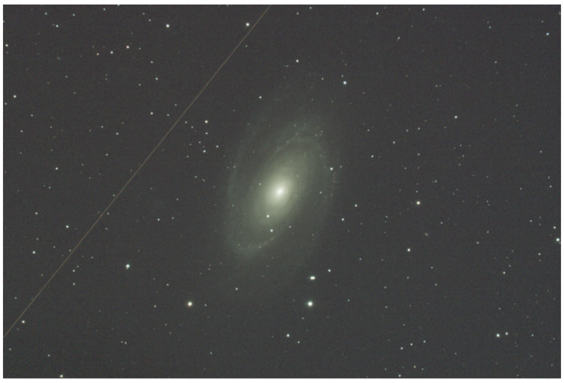
Included in the 'Symbol Definitions' are x() and y() which are the current horizontal x or vertical y coordinates in integer pixels of an image.

Generators

Generators are special functions that can build new images on the fly when a PixelMath expression is executed. Generators are always executed just after all image references, metasymbols and invariant subexpressions have been evaluated, but *before running PixelMath expressions*. This means that generators don't have the concept of *current pixel*, as the rest of PixelMath functions and operators do: a generator works on an entire image as a single step just once, not in a pixel-by-pixel fashion as normal function. This has important consequences that must be taken into account from the syntactic and semantic points of view.

Expression to remove satellite trails using a generator

The following expression is built up to remove bright satellite trails from individual linear images, prior to stacking.



This is a single 600-second exposure of M81 that has been calibrated and de-bayered.

Zooming in on the image in PixInsight, the centre of the satellite trail can be found to start at pixel (x=0, y=3733) and end at pixel (x=3007, y=0)

The aim is to replace the bright pixels in the satellite trail with typical pixel values of the background sky either side of it.

1) The function **d2seg(x1, y1, x2, y2)** returns the distance of a target pixel from the current line defined by x1, y1, x2, y2.
2) The conditional function **iif(condition, if-trues, if-false)** will let us apply a change to pixels only if they are a certain distance from the centre of the line segment. We will define the distance from the line segment by the symbol '**dist**'.

iif(d2seg(3007, 0, 0, 3733) < dist, [change the pixels to match the background], [keep the pixels the same])

OR

iif(d2seg(3007, 0, 0, 3733) < dist, [change the pixels to match the background], $T)

3) If the pixel is closer to the satellite trail line segment than 'dist' (for example 8-pixels, the **generator function** medfilt($T, 49) applies a median filter to the specified image in a 49x49 box around the target pixel.

   iif(d2seg(3007, 0, 0, 3733) < dist, medfilt($T, 49), $T)

   symbol: dist=8

   We now have a complete expression for replacing the satellite trail, defined as line segment 'd2seg' with width 'dist', to replace the pixels with the median value of the pixels in a 49x49 pixel box around the pixels to be replaced.

4) If we apply the expression however, and the satellite trail passes through or close to a bright star, we may get an unwanted effect. To reduces this possibility, we can add an additional condition using && which is the logical AND operator in PixelMath, so that the expression is only applied if the pixel value at $T is less than a specified value.

   iif(d2seg(3007, 0, 0, 3733) < dist && $T<0.12, medfilt($T, 49), $T)

   symbol: dist=8

5) Finally, we can improve the median value of the surrounding pixels by introducing an element of randomness to the values.

imad=mdev($T);

iif(d2seg(3007, 0, 0, 3733) < dist && $T<0.12, (medfilt($T, 49)+((random()*imad)-(random()*imad))), $T)

symbol: dist=8, imad

the 'mdev' function returns median deviation from the median value for the target image. The 'random' function returns a pseudo random value in the range 0-1.

The expression is PixelMath expression is highly effective at removing the unwanted satellite trail without creating artefacts or anomalies in the final image.