Software Development at DIFFRACTION

ASTRONOMY + SCIENTIFIC IMAGING SOLUTIONS

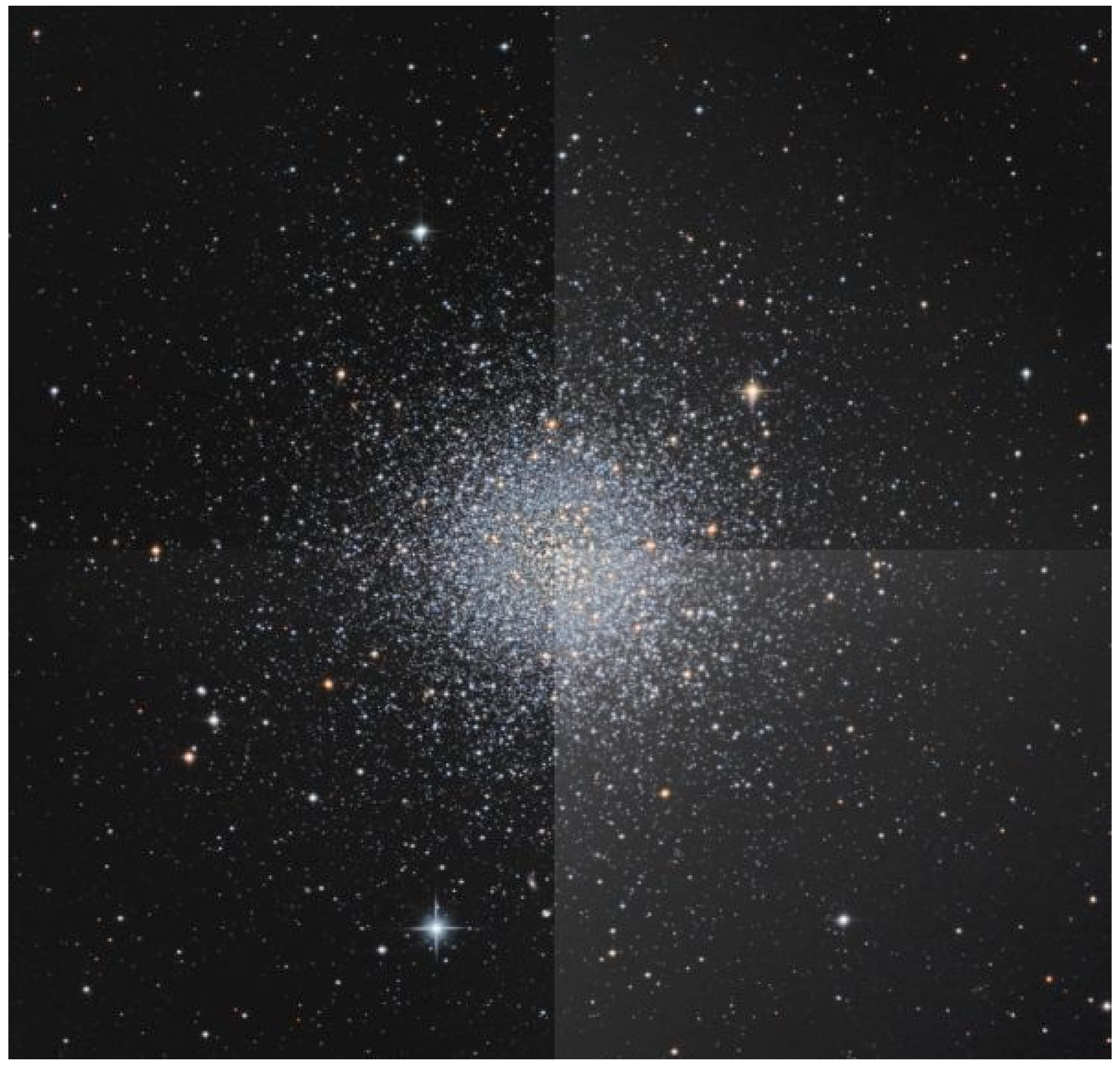
Theodore Grosson
University of Victoria

NTCO

University of Victoria

MaxIm DL Artifact Removal

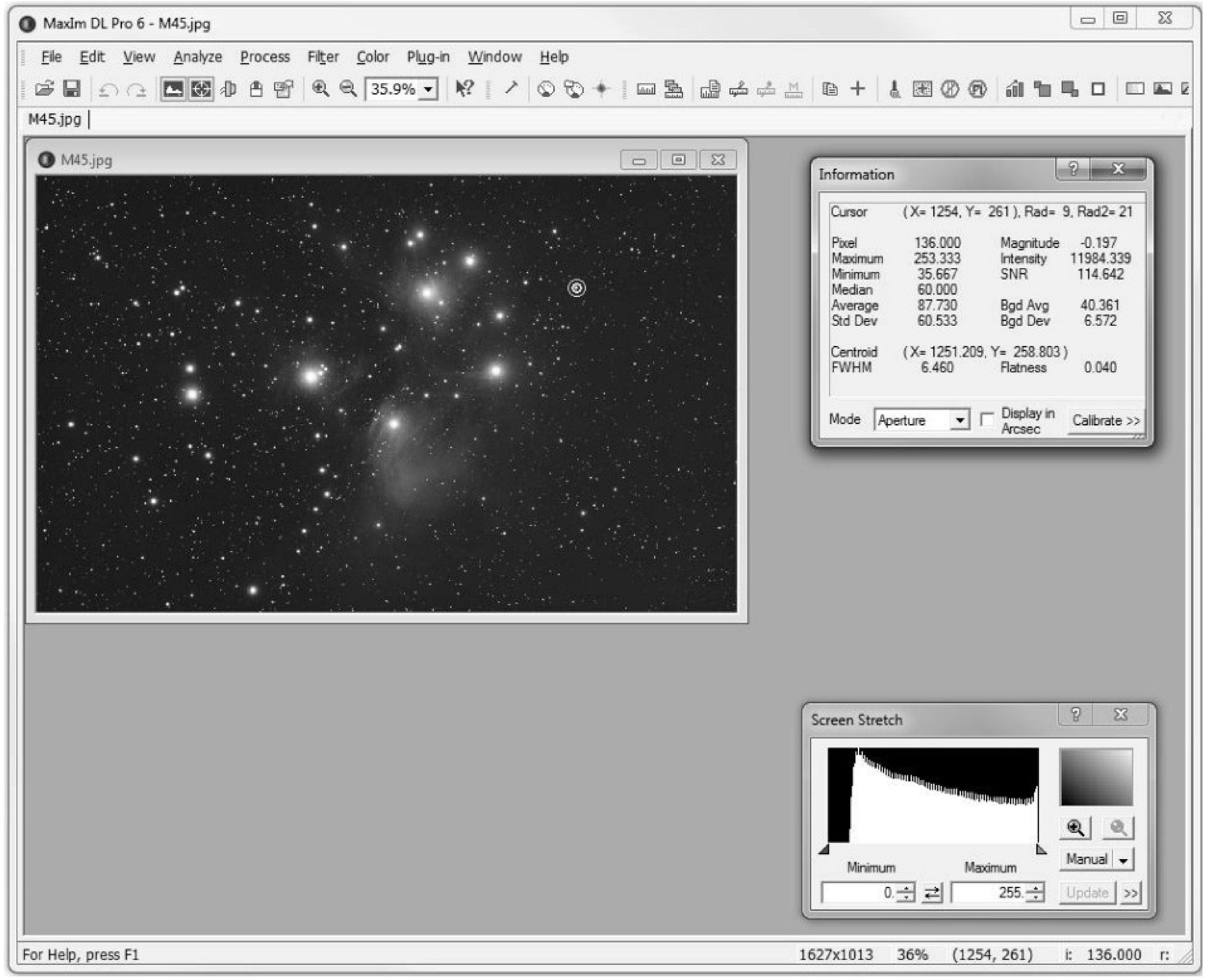
Differing nonlinearities between sensor segments in certain cameras not corrected by flat-fielding



Simulated dark– and flat–corrected image before applying nonlinearity calibration

Algorithm developed and implemented in MaxIm DL to correct issue

Sequence of flat images to calibrate nonlinearities



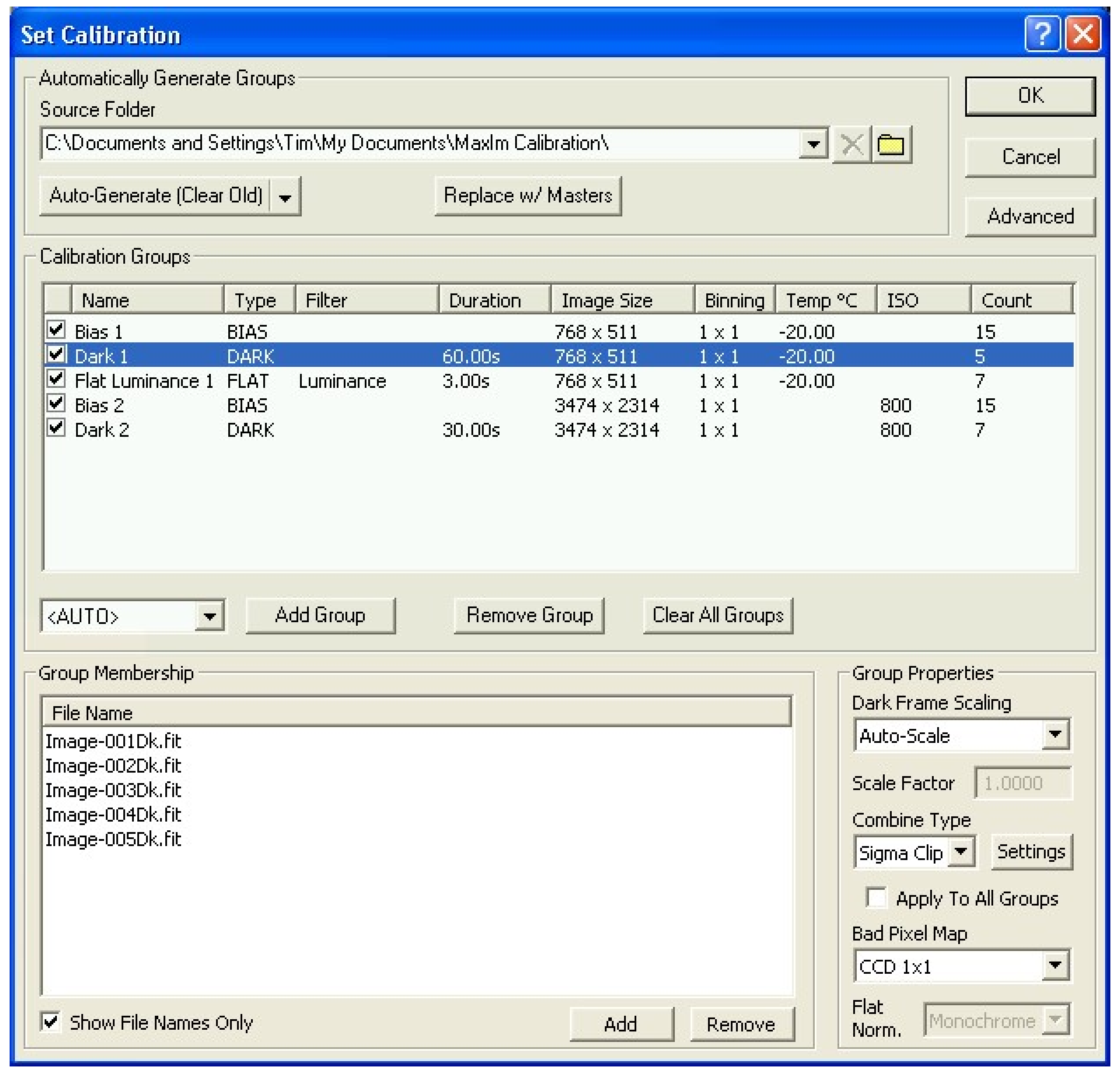
Cursor (X= 1254, Y= 261), Rad= 9, Rad2= 21

Pixel	136.000	Magnitude	-0.197
Maximum	253.333	Intensity	11984.339
Minimum	35.667	SNR	114.642
Median	60.000		
Average	87.730	Bgd Avg	40.361
Std Dev	60.533	Bgd Dev	6.572

Centroid (X= 1251.209, Y= 258.803)
FWHM 6.460 Flatness 0.040

Mode: Aperture
□ Display in Arcsec
Calibrate >>

MaxIm DL main screen



Automatically Generate Groups

Source Folder: C:\Documents and Settings\Tim\My Documents\MaxIm Calibration\

Calibration Groups

	Name	Type	Filter	Duration	Image Size	Binning	Temp °C	ISO	Count
☑	Bias 1	BIAS			768 x 511	1 x 1	-20.00		15
☑	Dark 1	DARK		60.00s	768 x 511	1 x 1	-20.00		5
☑	Flat Luminance 1	FLAT	Luminance	3.00s	768 x 511	1 x 1	-20.00		7
☑	Bias 2	BIAS			3474 x 2314	1 x 1		800	15
☑	Dark 2	DARK		30.00s	3474 x 2314	1 x 1		800	7

<AUTO>

Group Membership

File Name
Image-001Dk.fit
Image-002Dk.fit
Image-003Dk.fit
Image-004Dk.fit
Image-005Dk.fit

☑ Show File Names Only

Group Properties
Dark Frame Scaling: Auto-Scale
Scale Factor: 1.0000
Combine Type: Sigma Clip
□ Apply To All Groups
Bad Pixel Map: CCD 1x1
Flat Norm.: Monochrome

MaxIm DL window implementing correction algorithm

Python Wrapper for DLAPI

Simplified DL device interface in Python for the modern astronomer

Object-oriented Python with self-contained instances

Internal error handling

```
 {c++}
#include <dlapi.h>
#include <iostream>
int main()
{
    dl::IGatewayPtr pGateway = dl::getGateway();
    pGateway->queryUsbCameras();
    if (pGateway->getUsbCameraCount() <= 0)
        return 1;
    dl::ICameraPtr pCamera = pGateway->getUsbCamera(0);
    pCamera->initialize();

    try
    {
        if (!pCamera) throw std::logic_error("No camera selected")
        dl::IPromisePtr pPromise = pCamera->queryStatus();
        dl::IPromise::Status result = pPromise->wait();
        if (result != dl::IPromise::Complete)
        {
            char buf[512] = {0};
            size_t lng = 512;
            pPromise->getLastError(&(buf[0]), lng);
            pPromise->release();
            throw std::logic_error(std::string(&(buf[0]), lng));
        }
        pPromise->release();
        return pCamera->getStatus();
    }
    catch (std::exception &ex)
    {
        throw std::logic_error(std::string("Cannot query the camera status: ") + ex.what() );
    }

    ...
}
```

SWIG

```
#Python3
import PyDLAPI

gateway = PyDLAPI.getGateway()
camera = gateway.getCamera()
camera.initialize()
sensor = camera.getSensor()

sensor.setSubframe(<params>)
sensor.startExposure(<params>)

image = sensor.getImage()

gateway.close()
```