Correction of sensor image errors

Please have a look at the following notes:

- If you execute all correction methods, you have to keep the following order:

  1. Defective pixels correction
  2. Dark current correction
  3. Flat-field correction

- To correct the complete image, you have to reset the **AOI** (right-click "**Restore Default**" on each size value [width and height] in "*Setting → Base → Camera*").

- You have several possibilities to save the correction data. The chapter "*GUI → wxPropView → How to work with wxPropView → Storing and restoring settings*" in the camera manuals describes the different ways.

- All corrections available in "*Setting → Base → ImageProcessing*" are

  1. executed host based and need processor load.
  2. Are available for all MATRIX VISION cameras.

- With mvBlueCOUGAR-X, it is possible to execute the flat-field correction on the camera. This document will describes this in the flat-field correction section. As a general rule: if a camera supports camera-based corrections, you will find the controls in "*Setting → Base → Camera → GenICam*". If you have any questions about the implementation pipeline of the camera-based corrections, please contact us.

1. Defective pixels

Due to random process deviations, not all pixels in an image sensor array will react in the same way to a given light condition. These variations are known as blemishes or defective pixels.

There are two types of defective pixels:

1. leaky pixel (in the dark)
2. cold pixel (in standard light conditions)

1.2. Leaky pixels

These are pixels that produce a higher read out code than average when the sensor is not exposed. They manifest themselves as bright dots on a dark background. Leaky Pixels are defined as pixels that have a read out code that deviates more than T_{leak} ADC codes from the mean as shown in figure 1. An example of a potential cause of a Leaky Pixel is a high leakage current of the pixel's photo-diode.

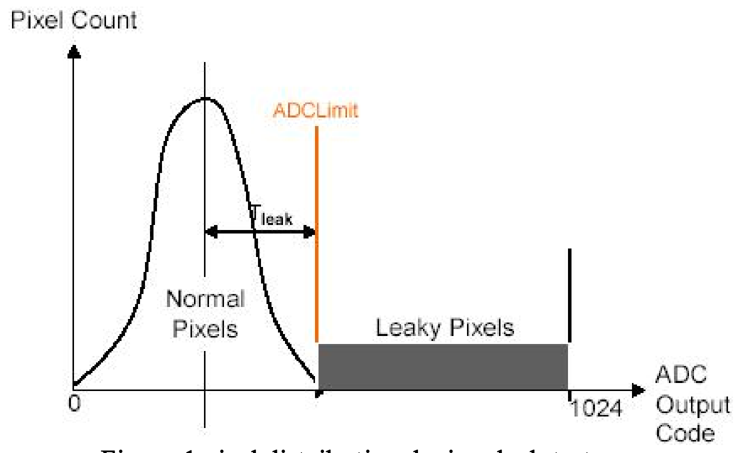


Figure 1 pixel distribution during dark tests

1.2.1. Blemish specification

The total number of defective pixels found in the array are gain and integration time dependent. To find the maximum number of leaky pixels for a given operating condition the following procedure should be followed:

$GT_{Product}$ = Linear Gain Setting * Integration time

1.2.2. Leaky pixels correction (theory)

By using neighbor pixels in a different way, you could replace the defective pixels. Available replace methods are:

- 3 x 1 Average
- 3 x 3 Median

1.2.3. Leaky pixels correction (practice)

To correct the leaky pixels following steps are necessary (see Figure 3):

- $GT_{Product}$ = 360 msec (**Gain_dB** = 0 dB, exposure time (**Expose_us**) = 360 msec)
- Shade the lens completely
- (Filter-)*Mode = Calibrate leaky pixel*
- Snap an image (**Acquire** [*Acquisition Mode = SingleFrame*])
- Save the correction data via "*Action → Capture Settings → Save Active Device Settings*"
  (Settings are saved in the registry)

The filter checks:

```
Pixel > LeakyPixelDeviation_ADCLimit
```

All pixels above this value have a high leakage current.

1.3. Cold pixels

Illuminated blemish tests are performed under standard light conditions. The light level is chosen in such a manner that the pixels produce relatively high output readings (e.g. 70% of the saturation level). Cold Pixels are pixels that produce a lower read out code than average when the sensor is exposed to light. They manifest themselves as lower intensity dots when compared to a uniformly illuminated background. These pixels can be detected by exposing the sensor to light, and detecting the pixels that deviate more than a percentage T_{cold} from the mean, as shown in Figure 2.
A example of a potential cause of a Cold Pixel is a dust particle on top of the pixel.

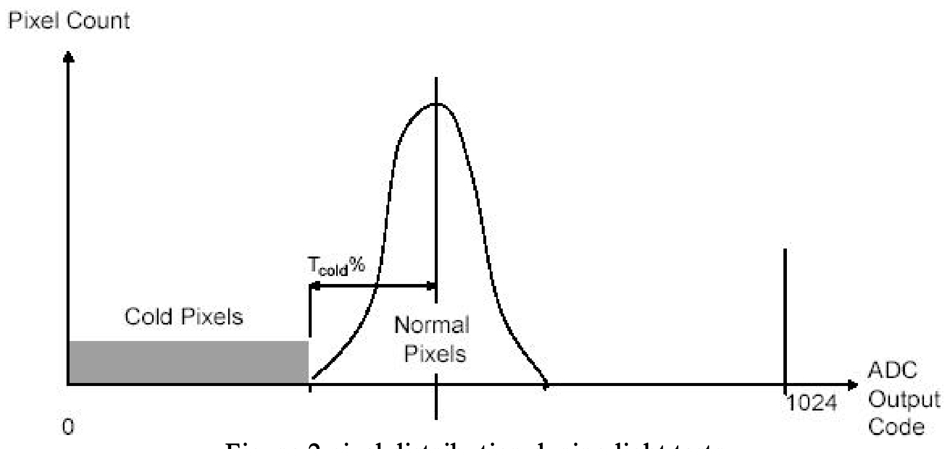


Figure 2 pixel distribution during light tests

1.3.1. Cold pixels correction (theory)

Accordant to the leaky pixels correction the cold pixels are replaced by using the neighbor pixels. There are also different possibilities which neighbors you can use.

1.3.2. Cold pixel s correction (practice)

To correct the cold pixels following steps are necessary (see Figure 3):

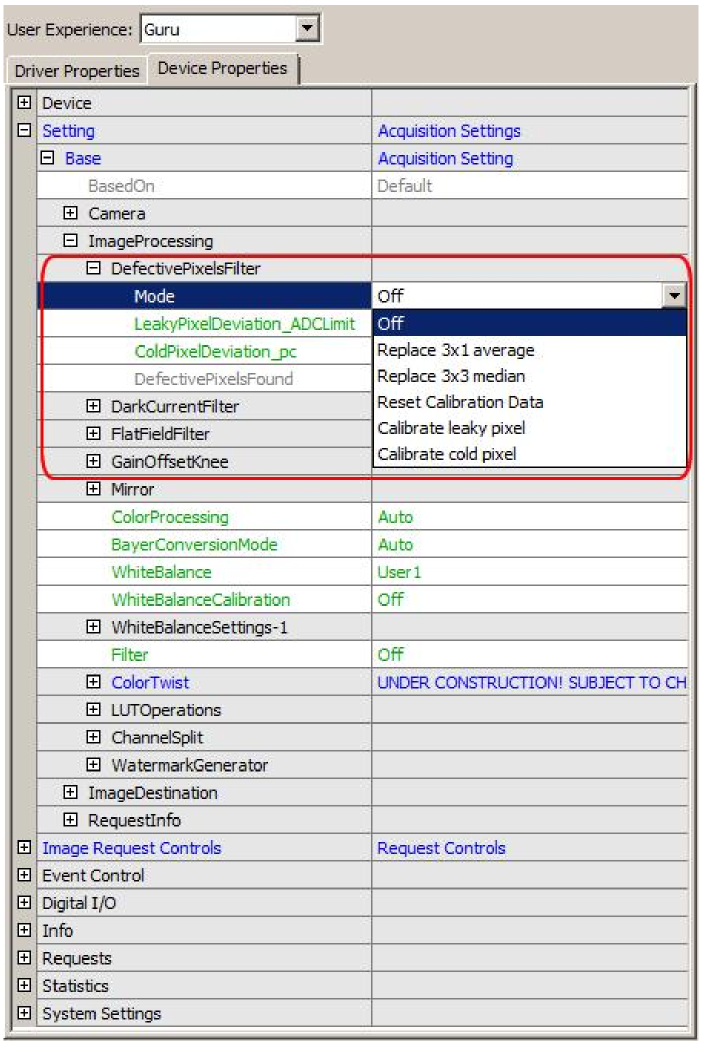
- Uniform sensor illumination approx. 50 - 70 % saturation (which means an average gray value between 128 and 180)
- (Filter-)Mode = Calibrate cold pixel
- Snap an image (**Acquire** [Acquisition Mode = SingleFrame])
- Save the correction data via "*Action → Capture Settings → Save Active Device Settings*"
  (Settings are saved in the registry)

The filter checks:

```
Pixel < Tcold (default value: 15 %)
```

T_{cold} = deviation of the average gray value (**ColdPixelDeviation_pc**)

All pixels below this value have a too weak dynamic.



User Experience: Guru

Driver Properties | Device Properties

⊞ Device	
⊟ Setting	Acquisition Settings
⊟ Base	Acquisition Setting
BasedOn	Default
⊞ Camera	
⊟ ImageProcessing	
⊟ DefectivePixelsFilter	
Mode	Off
LeakyPixelDeviation_ADCLimit	Off
ColdPixelDeviation_pc	Replace 3x1 average
DefectivePixelsFound	Replace 3x3 median
⊞ DarkCurrentFilter	Reset Calibration Data
⊞ FlatFieldFilter	Calibrate leaky pixel
⊞ GainOffsetKnee	Calibrate cold pixel
⊞ Mirror	
ColorProcessing	Auto
BayerConversionMode	Auto
WhiteBalance	User1
WhiteBalanceCalibration	Off
⊞ WhiteBalanceSettings-1	
Filter	Off
⊞ ColorTwist	UNDER CONSTRUCTION! SUBJECT TO CH
⊞ LUTOperations	
⊞ ChannelSplit	
⊞ WatermarkGenerator	
⊞ ImageDestination	
⊞ RequestInfo	
⊞ Image Request Controls	Request Controls
⊞ Event Control	
⊞ Digital I/O	
⊞ Info	
⊞ Requests	
⊞ Statistics	
⊞ System Settings	

Figure 3 defective pixel filter

2. Dark current

Dark current is a characteristic of image sensors, which means, that image sensors also deliver signals in total darkness. This effect is provoked by warmness, which creates charge carriers spontaneously. This signal overlays the image information.
The production of dark current depends on two circumstances:

1. **Exposure time**
   The longer the exposure, the greater the dark current part. I.e. using long exposure times, the dark current itself could lead to an overexposed sensor chip.
2. **Temperature**
   By cooling the sensor chips the dark current production can be highly dropped (approx. every 6 °C the dark current is cut in half).

2.1. Dark current correction theory

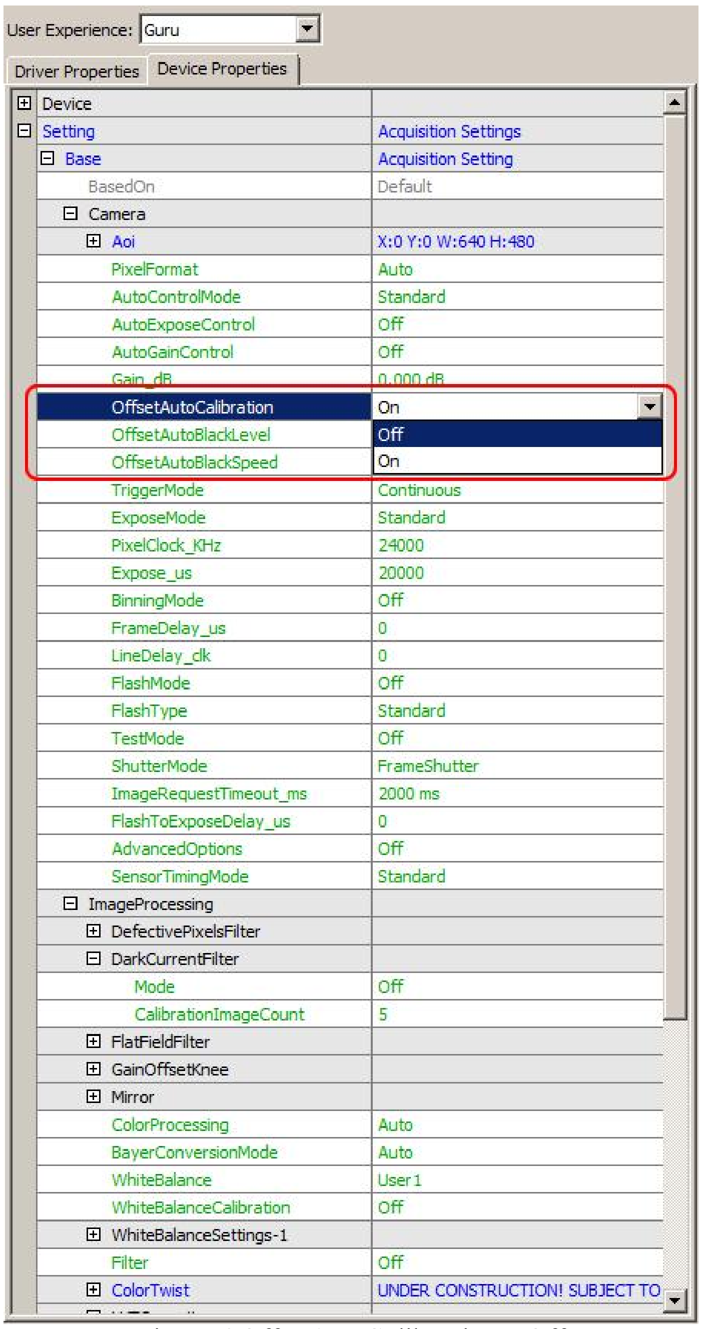
The correction happens pixel wise, i.e. the first pixel (e.g. top left of the image corner) of the original image will be corrected with the analogous pixel of the adjustment images.
As the dark current is repeatable, you can use either a dark current image ahead or after the original image and remove it pixel wise. To get a better result it is necessary to snap the original and the dark current images with the same exposure time and temperature. After the correction the signal of the snapped object remains.

Note: Dark current snaps (like other snaps as well) are generally endued with noise. This noise shows the unsafeness of a quantity to be measured, like it appears everywhere in physics. As the difference of two uncorrelated noise sources never is null, likewise the difference of two dark current images is not null. As a rule noise remains.

2.2. Dark current correction practice

- Shade lens
- OffsetAutoCalibration = Off (see Figure 4)
- Change **Offset_pc** until you'll see an amplitude in the histogram (see Figure 5)
- Set exposure time according to the application
- (Filter-)*Mode = Calibrate* (see Figure 6)
- Start a live acquisition (**Acquire** [*Acquisition Mode = Continuous*])
- Save the correction image via "*Action → Capture Settings → Save Active Device Settings*"
  (Settings are saved in the registry)

The filter snaps a number of images and averages the dark current images to one correction image.



User Experience: Guru

Driver Properties | Device Properties

⊞ Device	
⊟ Setting	Acquisition Settings
⊟ Base	Acquisition Setting
BasedOn	Default
⊟ Camera	
⊞ Aoi	X:0 Y:0 W:640 H:480
PixelFormat	Auto
AutoControlMode	Standard
AutoExposeControl	Off
AutoGainControl	Off
Gain_dB	0.000 dB
OffsetAutoCalibration	On
OffsetAutoBlackLevel	Off
OffsetAutoBlackSpeed	On
TriggerMode	Continuous
ExposeMode	Standard
PixelClock_KHz	24000
Expose_us	20000
BinningMode	Off
FrameDelay_us	0
LineDelay_clk	0
FlashMode	Off
FlashType	Standard
TestMode	Off
ShutterMode	FrameShutter
ImageRequestTimeout_ms	2000 ms
FlashToExposeDelay_us	0
AdvancedOptions	Off
SensorTimingMode	Standard
⊟ ImageProcessing	
⊞ DefectivePixelsFilter	
⊟ DarkCurrentFilter	
Mode	Off
CalibrationImageCount	5
⊞ FlatFieldFilter	
⊞ GainOffsetKnee	
⊞ Mirror	
ColorProcessing	Auto
BayerConversionMode	Auto
WhiteBalance	User1
WhiteBalanceCalibration	Off
⊞ WhiteBalanceSettings-1	
Filter	Off
⊞ ColorTwist	UNDER CONSTRUCTION! SUBJECT TO

Figure 4 OffsetAutoCalibration = Off

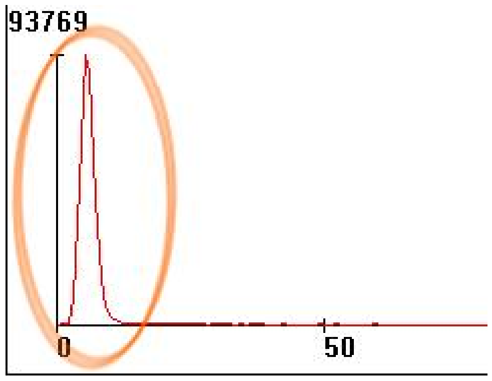
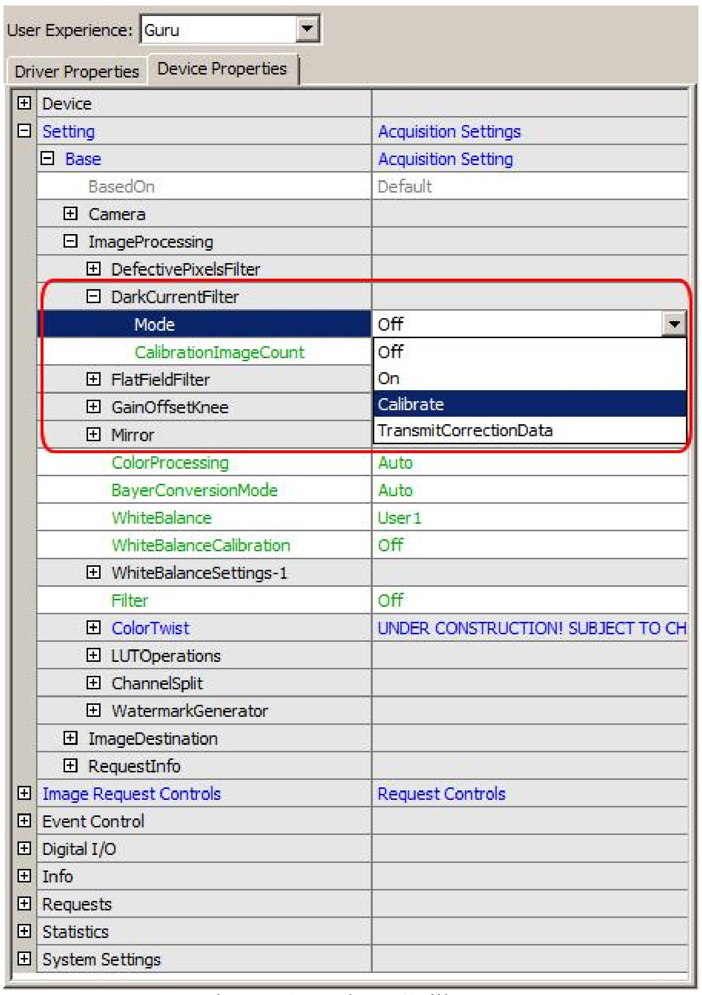


Figure 5 histogram output of a software example


User Experience: Guru

Driver Properties | Device Properties

⊞ Device	
⊟ Setting	Acquisition Settings
⊟ Base	Acquisition Setting
BasedOn	Default
⊞ Camera	
⊟ ImageProcessing	
⊞ DefectivePixelsFilter	
⊟ DarkCurrentFilter	
Mode	Off
CalibrationImageCount	Off
⊞ FlatFieldFilter	On
⊞ GainOffsetKnee	Calibrate
⊞ Mirror	TransmitCorrectionData
ColorProcessing	Auto
BayerConversionMode	Auto
WhiteBalance	User1
WhiteBalanceCalibration	Off
⊞ WhiteBalanceSettings-1	
Filter	Off
⊞ ColorTwist	UNDER CONSTRUCTION! SUBJECT TO CH
⊞ LUTOperations	
⊞ ChannelSplit	
⊞ WatermarkGenerator	
⊞ ImageDestination	
⊞ RequestInfo	
⊞ Image Request Controls	Request Controls
⊞ Event Control	
⊞ Digital I/O	
⊞ Info	
⊞ Requests	
⊞ Statistics	
⊞ System Settings	

Figure 6 Mode = Calibrate

3. Variations of sensitivity

Strictly speaking: each pixel of a sensorchip is a single detector with its own properties. Particulary this pertains to the sensitivity as the case may be the spectral sensitivity. The difference of the sensitivity - depending on the sensor - can be more than 10 %. This is too much for being ignored, the differences have to be corrected. To solve this problem, a perfect equal background as a flat-field is snapped, which will be used to correct the original image.

3.1. Flat-field correction theory

A flat-field is a snap with the future focus (sharp) and an uniform lighten up area. Between the snap of the flat-field of an object, it is not allowed to change the sensor chip and the optic. To reduce errors while doing the flat-field correction, a saturation between 50 % and 75 % of the flat-field in the histogram is convenient.

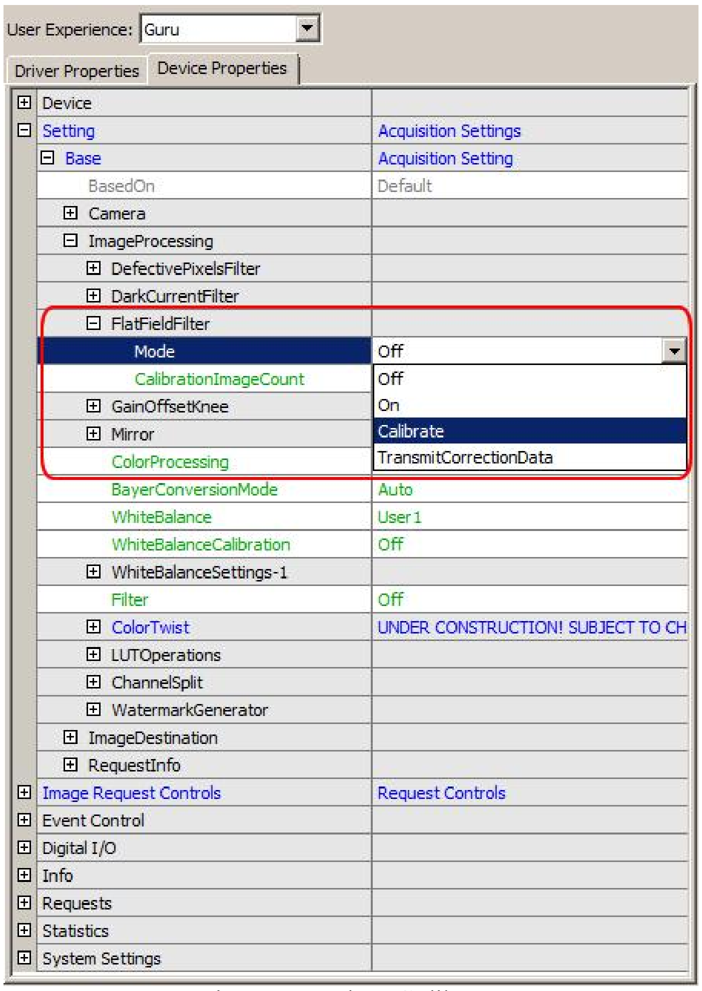
Afterwards the snapped image of the object is divided by the flat-field image and it is normalised to the averaged pixel value of the flat-field (flat-field correction). For example, if a pixel of the sensor chip is covered by 50 % through dust, the measured luminosity is reduced both in the image of the snapped object and in the flat-field. The division by the small luminance value of the flat-field helps to lift the luminance value of the object image to the level the image would have without dust.

Before the division you have to subtract a dark current image from the image of the object as well as from the flat-field, in the case a dark current image was taken, too.

3.2. *Flat-field correction practice*

- Uniform sensor illumination approx. 50 - 70 % saturation
- (Filter-)*Mode = Calibrate* (see Figure 7)
- Start a live acquisition (**Acquire** [*Acquisition Mode = Continuous*])
- Save the correction image via "*Action → Capture Settings → Save Active Device Settings*"
  (Settings are saved in the registry)

The filter snaps a number of images and averages the flat-field images to one correction image.



User Experience: Guru

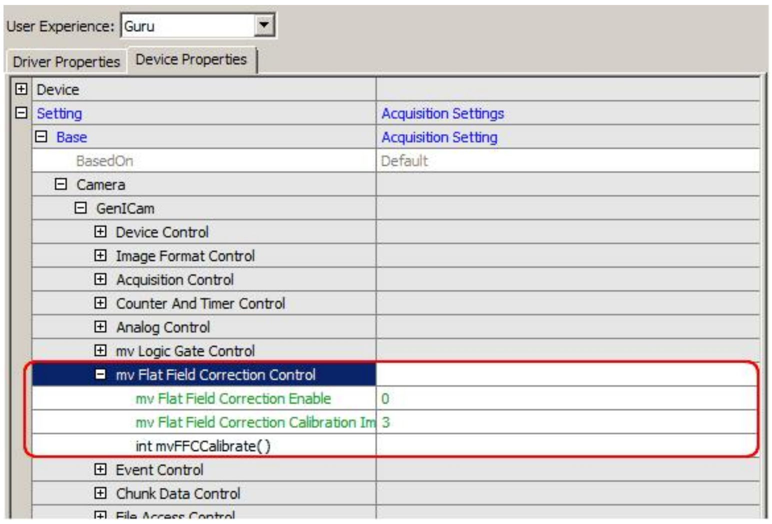
Driver Properties | Device Properties

⊞ Device	
⊟ Setting	Acquisition Settings
⊟ Base	Acquisition Setting
BasedOn	Default
⊞ Camera	
⊟ ImageProcessing	
⊞ DefectivePixelsFilter	
⊞ DarkCurrentFilter	
⊟ FlatFieldFilter	
Mode	Off
CalibrationImageCount	Off
⊞ GainOffsetKnee	On
⊞ Mirror	Calibrate
ColorProcessing	TransmitCorrectionData
BayerConversionMode	Auto
WhiteBalance	User1
WhiteBalanceCalibration	Off
⊞ WhiteBalanceSettings-1	
Filter	Off
⊞ ColorTwist	UNDER CONSTRUCTION! SUBJECT TO CH
⊞ LUTOperations	
⊞ ChannelSplit	
⊞ WatermarkGenerator	
⊞ ImageDestination	
⊞ RequestInfo	
⊞ Image Request Controls	Request Controls
⊞ Event Control	
⊞ Digital I/O	
⊞ Info	
⊞ Requests	
⊞ Statistics	
⊞ System Settings	

Figure 7 Mode = Calibrate

3.2.1. *Camera-based flat-field correction*

With the GigE Vision camera family mvBlueCOUGAR-X, it is possible to execute the flat-field correction in the FPGA of the camera. Additionally, you can save the correction image on the camera, however, this depends on the sensor's resolution. You can have a look in "*Setting → Base → Camera → GenICam*" to check, if the specific camera model supports the camera-based correction (e.g. "*mv Flat Field Correction Control*"):



User Experience: Guru

Driver Properties | Device Properties

⊞ Device	
⊟ Setting	Acquisition Settings
⊟ Base	Acquisition Setting
BasedOn	Default
⊟ Camera	
⊟ GenICam	
⊞ Device Control	
⊞ Image Format Control	
⊞ Acquisition Control	
⊞ Counter And Timer Control	
⊞ Analog Control	
⊞ mv Logic Gate Control	
⊟ mv Flat Field Correction Control	
mv Flat Field Correction Enable	0
mv Flat Field Correction Calibration Im	3
int mvFFCCalibrate( )	
⊞ Event Control	
⊞ Chunk Data Control	

Figure 8 Camera-based flat-field correction

In the chapter "*Use Cases → Using real-time flat field correction (FFC)*" you will find a detailed description, how to use the camera-based flat-field correction.