地 平 线
Horizon Robotics

Journey J3 ISP Tuning Guide

V0.6

2020-06

Legal Disclaimer

This documentation information is intended only to assist system and software users with horizon products.This document does not expressly or implicitly authorize others to design or manufacture any integrated circuit based on this document information.

The information in this document is subject to change without notice.While this document ensures to the extent possible the accuracy of the content, all statements, information and Suggestions contained in this document do not constitute any warranty, representation or warranty, express or implied.

All information in this document is provided "as is".Horizon makes no warranties, representations or warranties, express or implied, as to the merchantability, fitness for any particular purpose and non-infringement of intellectual property rights of any third party.Horizon shall not be liable for any damages arising out of the use of the Products, including but not limited to direct or indirect damages.

Buyer and other parties that are developing products based on Horizon Products (collectively referred to as "Customer") understand and agree that Customer is responsible for independent analysis, evaluation and judgment in designing applications for the Products.Users are solely responsible for the security of their applications (and all Horizon products used in their applications) and ensure compliance with all applicable regulations, laws and other requirements.The "typical" parameters provided in the ground line product profile and product specifications may vary from application to application, and actual performance may vary over time.All work parameters, including "typical" parameters, must be validated by the user himself against each user application.

Customer agrees to indemnify Horizon and its representatives in full for any claims, damages, costs, losses and/or liabilities arising out of customer's unauthorized use of Horizon products or non-compliance with the terms of this Note.

Revision Information

The following revisions have been made to this document

Version	Date	Change
0.6	2020-6-30	First release

Contents

List of Figures

List of Tables

1 J3 ISP introduction

1.1 Overview

This document describes a test plan for tuning and verifying still/video image quality of theoverall ISP system. The test plan covers a range of factors that may impact image quality, including CMOS sensor, optics structure and and J3 image processing pipeline. Suitable tuning improvement plan should be taken basing on HW and project requirement.

Horizon Journey J3 Image processor includeing the following important technologies:

- Iridix, Local Tone-mapping technology；
- Sinter, spatial noise reduction technology；
- Temper temporal（frame-to-frame）noise reduction technology；

Image quality testing should be performed at regular milestones throughout the project. Allmandatory tests should be run at least once during the project.

Testing should be carried out in an appropriate test environment that can be completely sealed from all external light sources.

The objective image quality testing described in this document should always be supplemented by subjective real-world testing for the most accurate analysis of image quality. Images should be compared against a reference camera.

Subjective testing should be directed at a range of scenes applicable to the final application of the camera. In particular, testing should be carried out under the types of lighting typical in the intended environment. The following sections of this document describe the tuning tests, procedures of how to execute the tests, how to analyze the results and how to set parameters in the ISP.

Horizon Journey J3 supports three ISP/capture modes: Linear, Native-WDR-Lin and FS-Lin.Unless otherwise stated, this document refers only to Linear mode.

ISP Pipeline

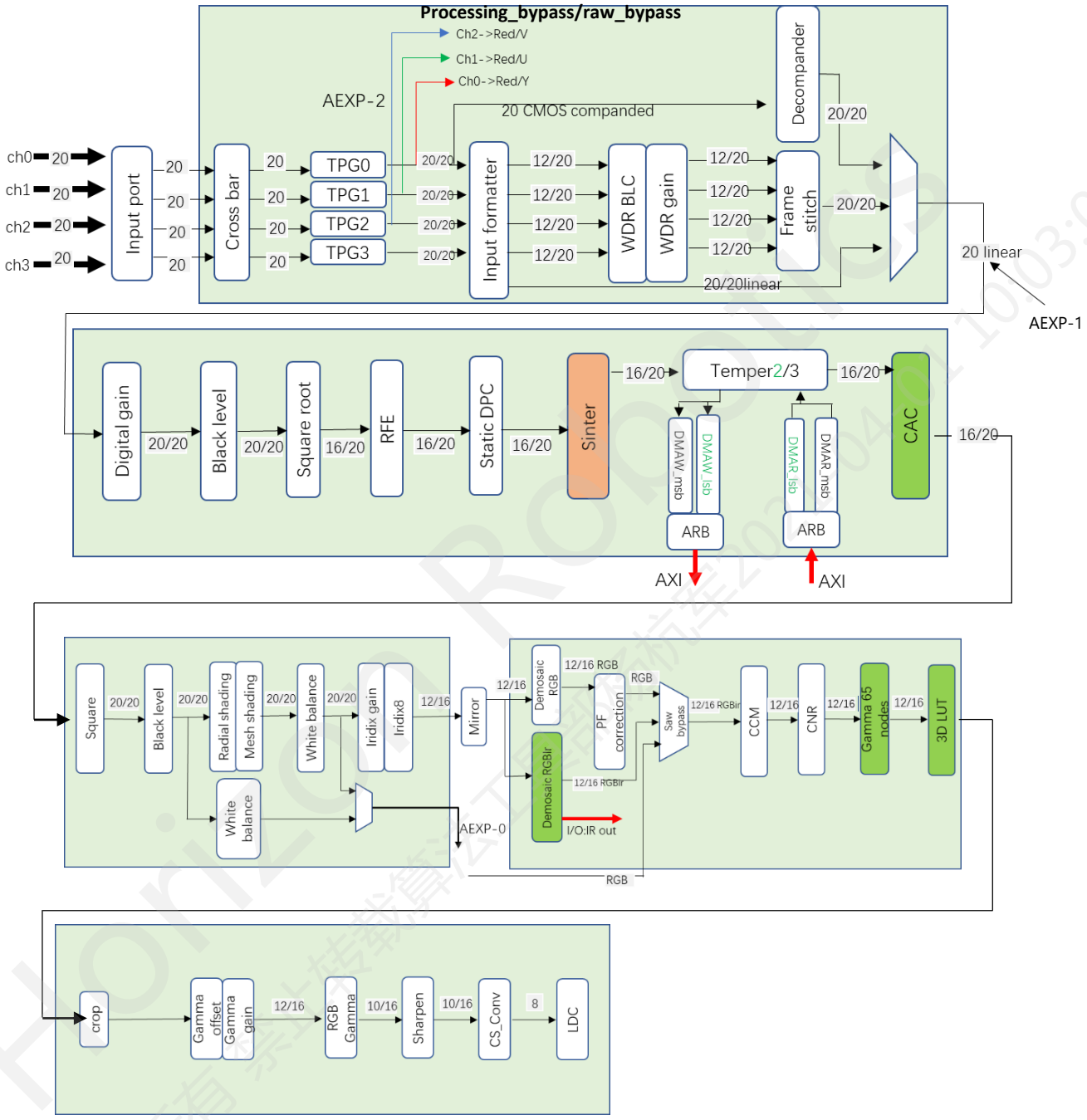


Figure 1.1-1 ISP Pipeline

ISP pipeline introduction：

1.Green means the the block can be moved from pipeline；

2.Red-thick line means data output；

3.Temper block is divided Temper2 和 Temper3，Temper2 only use DMA_LSB which marked green；

4.m/n menas m and n mode can be switched：

- m: means current ISP block bit deepth.
- n: means input n-bits will bypass current blcok and output n-bits directly .
- bypass logic as below:

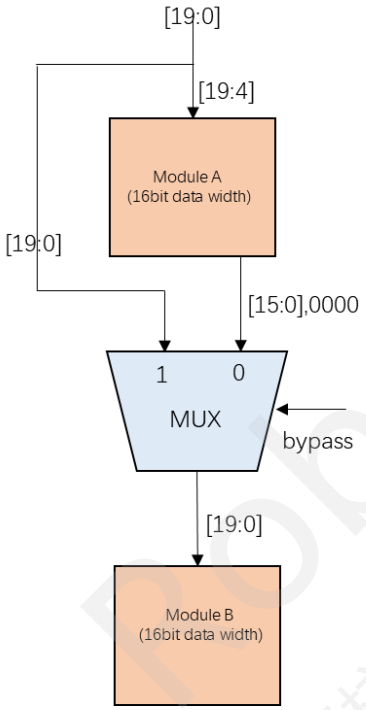


Figure 1.1-2 bypass logic

1.2 Statistics Pipeline

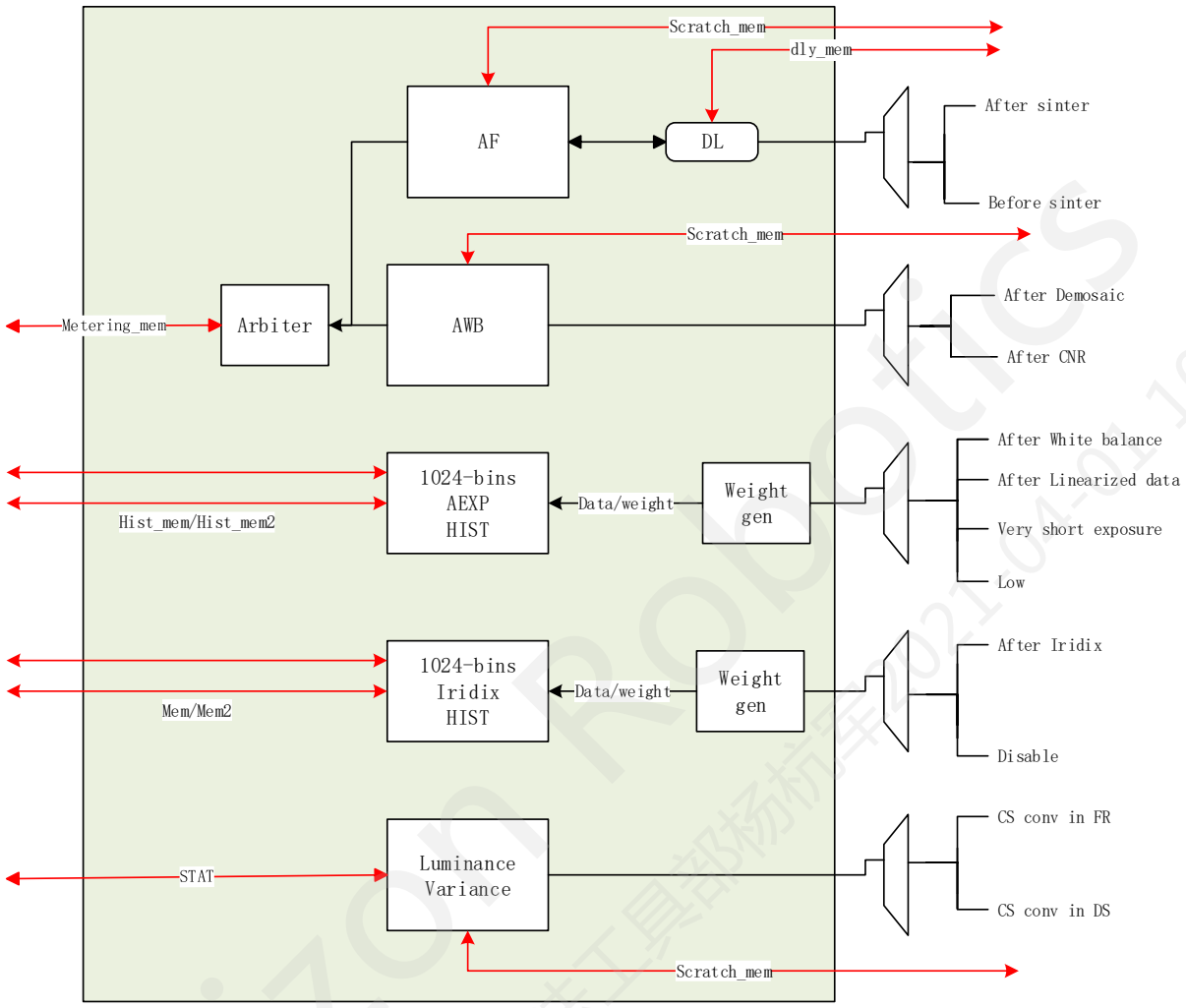


Figure 1.2-1 Statistics Pipeline

2 ISP Tuning Overview

2.1 Tuning procedure

2.1.1 Tuning equipment

ISPs intended for outdoor applications, including automotive, should be tuned in a range of settings appropriate to the intended use.

The following equipment is recommended for tuning the ISP for typical interior applications:

- 24 X-Rite Color chart
- Edmund optics Opal diffusing glass
- ISO12233 chart
- ISO15739 chart
- Two stand light boxes which lumimance can be tuned
- Tri-pod
- Standard CT light box
- Studio scene
- Illumination photometer
- DNP
- 36steps HDR tester

2.1.2 Tuning in Phases

The ISP calibration cycle process comprises three stages.

In the first stage, the imaging sensor is characterized.

In the second stage, the algorithmic modules of the ISP are tuned. According to the imaging sensor characteristics.

In the third stage, comprehensive system testing and fine-tuning are performed.

Three tuning steps as below:

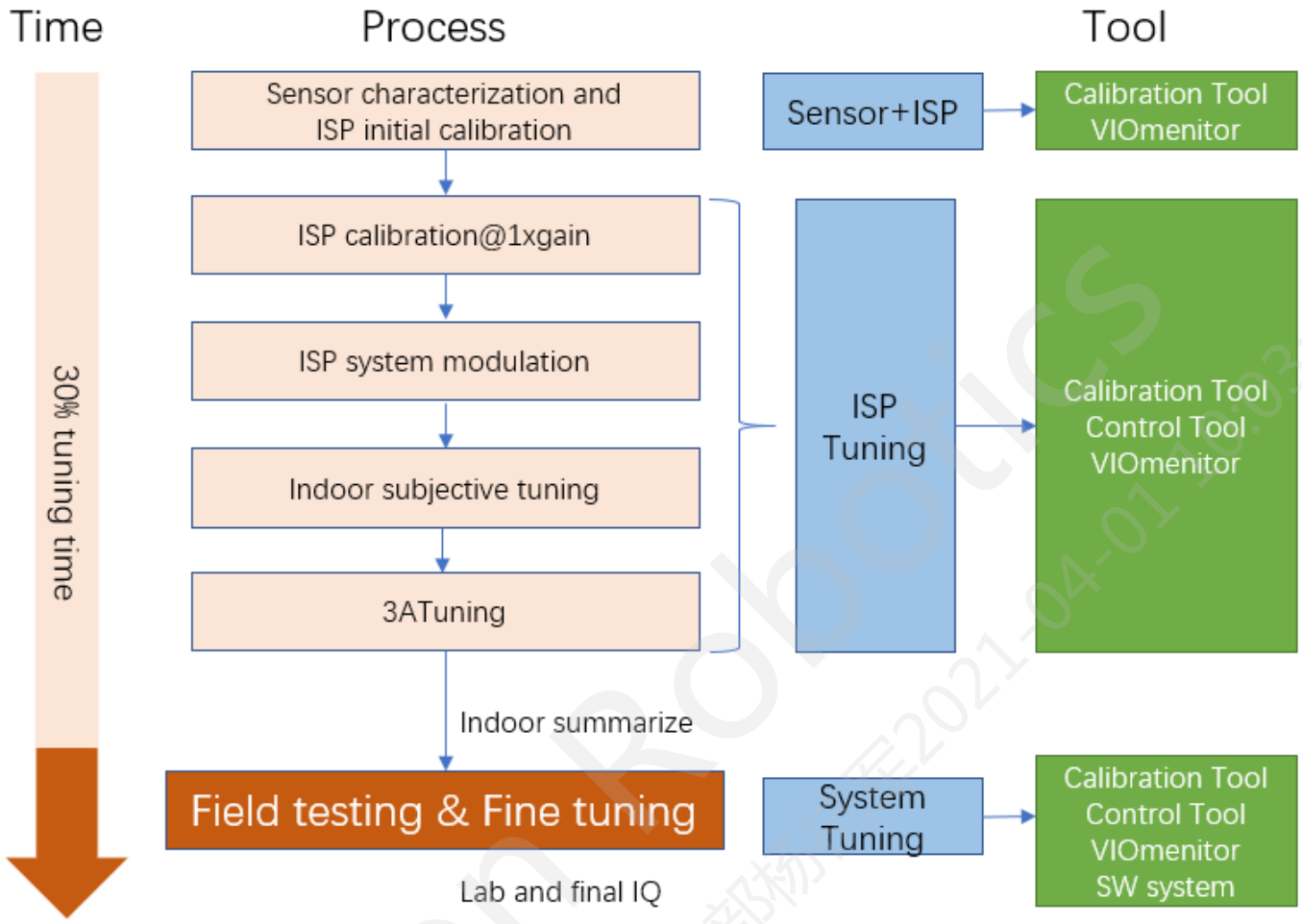


Figure 2.1-1 ISP tuning process

2.1.3 Phase 1-Sensor Dependent Calibration

Initial tuning primarily uses *Calibration Tool* in controlled conditions to characterize the sensor. *Control Tool* is used to prepare the system for capture. As this stage focuses on characterizing the sensor, the order in which modules are tuned is not fully representative of the order of modules in the hardware pipeline. Modules not required for sensor characterization are described later.Static raw data can be captured using Hobot player tool, refer to《Horizon X3 Tuning tool introduction》

Sensor and optics characterization uses the following modules:

- Black Levels.
- De-Companding (frontend/FE) LUTs. For WDR PWL mode
- Noise profile (NP).
- Green equalize (GE)
- Dynamic dead pixel correction (Dynamic DPC)

- Auto-exposure (AE)
- Lens shading correction (LSC)
- Auto-white balance (AWB)
- Gamma
- Color correction matrix (CCM)
- Chromatic aberration correction (CAC)
- Purple fringe correction (PF)

2.1.4 Phase 2-Algorithem

The second tuning phase depends on the sensor and algorithms used and the adjustmentof parameters for a range of lighting conditions.

This phase is mainly carried out using Control Tool, refer to 《Horizon X3 Tuning tool introduction》. The end result should producean imaging system capable of accurately capturing images in defined conditions. The blocks are:

- Stitching
- Sinter（temperal noise reduction）
- Demosaic
- Iridix（Local Tone Mapping）
- Color noise reduction（CNR）
- Sharpening
- ISP modules

2.1.5 Phase 3-Fine tuning

This phase allows for identification of areas where the previously tuned parameters are failing, resulting in poor or unacceptable image quality. Typically, this will involve testing under a variety of luminance levels as a minimum. This and further testing is done under controlled conditions, evaluating the system's ability to capture a range of real world scenes (appropriate to the intended application). Analysis of the captured images will require objective and subjective evaluation. For cases where image quality is particularly poor, parameter tuning is reviewed and adjusted before restarting the test.

ISP modules requiring modulation tuning in phase three:

- Sinter strength
- Temper strength
- CNR strength

- Iridix strength
- CCM saturation and strength
- Demosaic
- Sharpening strength

Note: Throughout the three phases, be cautious when altering parameters for a module after initial tuning. Any change will be reflected throughout the following modules with respect to either color reproduction or noise relative to detail, as seen in Figure 2.1-2. The solution to this is to carry out tuning in cycles. For example, if the black level needs to be corrected, as it is the first module in the pipeline and affects both color and noise/detail, all other modules will require re-tuning.

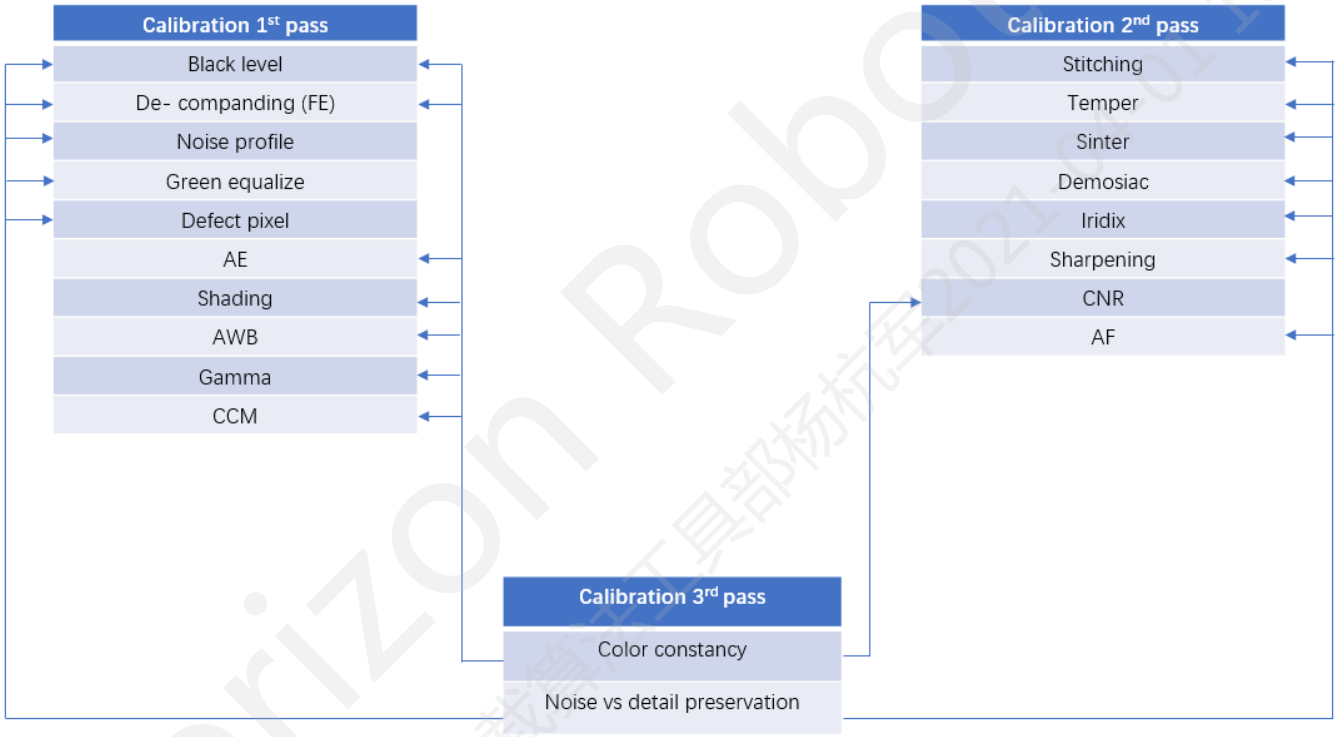


Figure 2.1-2 Tuning cycles

2.2 System Requirement for ISP tuning

Before an ISP can be tuned, the following information and requirements are needed:

2.2.1 ISP Information

- ISP architecture.
- The order of ISP modules and versions in the pipeline.
- ISP mode to be tunned：Liner，Native-WDR-lin，FS-lin.
- Input data type and bit depth.
- Sensor Color pattern mode.
- Whether need OTP parameter.
- Final performance target.

2.2.2 Information about Sensor configuration

- If it's wide dynamic range (WDR), the allowed exposure ratios and limitation of the sensor.
- If it's WDR, a datasheet on the sensor for decompander function.
- Resolution.
- CMOS sensor output bit depth.
- Frame rate.
- Sensor raw data structure.

2.2.3 Information about Optics and Sensor to be tuned

- Estimated variation from unit to unit (for shading, black levels and white point).
- Estimated black level variation with respect to temperature and analogue gain.
- IR-cut filter characteristics.
- Ensuring the sensor driver is correctly programmed, the sensor ISP is disabled andthe correct black level is set in the initialization sequence.

2.2.4 Tuning files, parameters and modulation LUTs

The tuning process is based on finding the optimum values for a range of control parameters for each ISP module. The parameters can be found in the tuning files and through calibration and control tools. Calibration parameters are contained in the following firmware files:

File name	Description
dynamic_calibrations.c	Contains ISP modulation tables
static_calibraton.c	Contains static tables which need no real time modification and are calibrated offline through *Calibration Tool*.
acamera_hardware_sensor.h	Contains HW registers in tuning processing

2.2.5 Tuning Tools (calibration/Control tool/Hobot player)

Horizon J3 ISP run performance tuning with three tools, static calibration tool、Control Tool and Hobot player。

2.2.5.1 Calibration Tool

Calibration Tool use to make initial calibration through simmulation with raw data to generate static calibration settings.Calibration tool application refer to《Horizon X3 Tuning tool introduction》,Calibration tool interface as Figure 2.2-1：

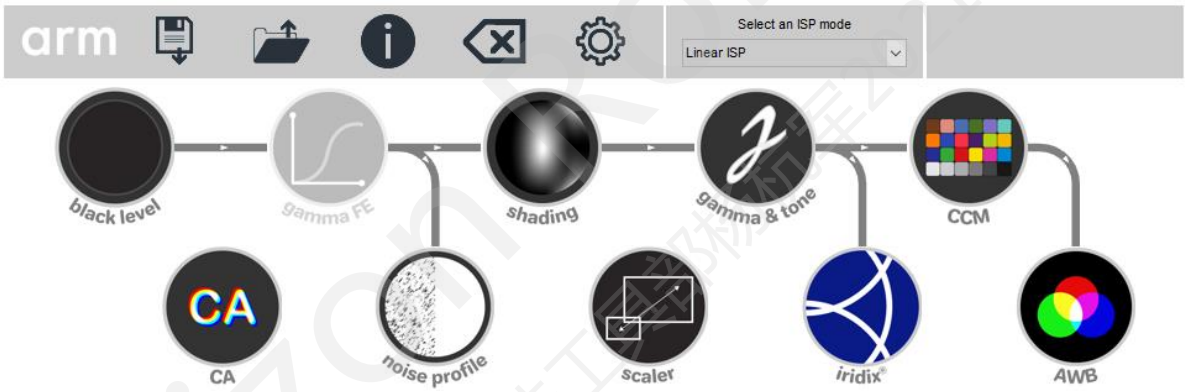


Figure 2.2-1 Calibration Tool

When using Calibration Tool to make calibration should be caution: Some modules simulation should run basing on prerequisite that prior modules need finished. After calibration will generate parameters with .json type which can be transferred to .c type through Control tool.

2.2.5.2 Control Tool

Control Tool is used to modify dynamic parameters of blocks, previewing performance can be found by hobotplayer tool. Before using control tool shuold confirm static calibrationand is uploaded .json to control tool. After finishing dynamic tuning with control tool can save the parameters with .json type. Control Tool interface as Figure 2.2-2.

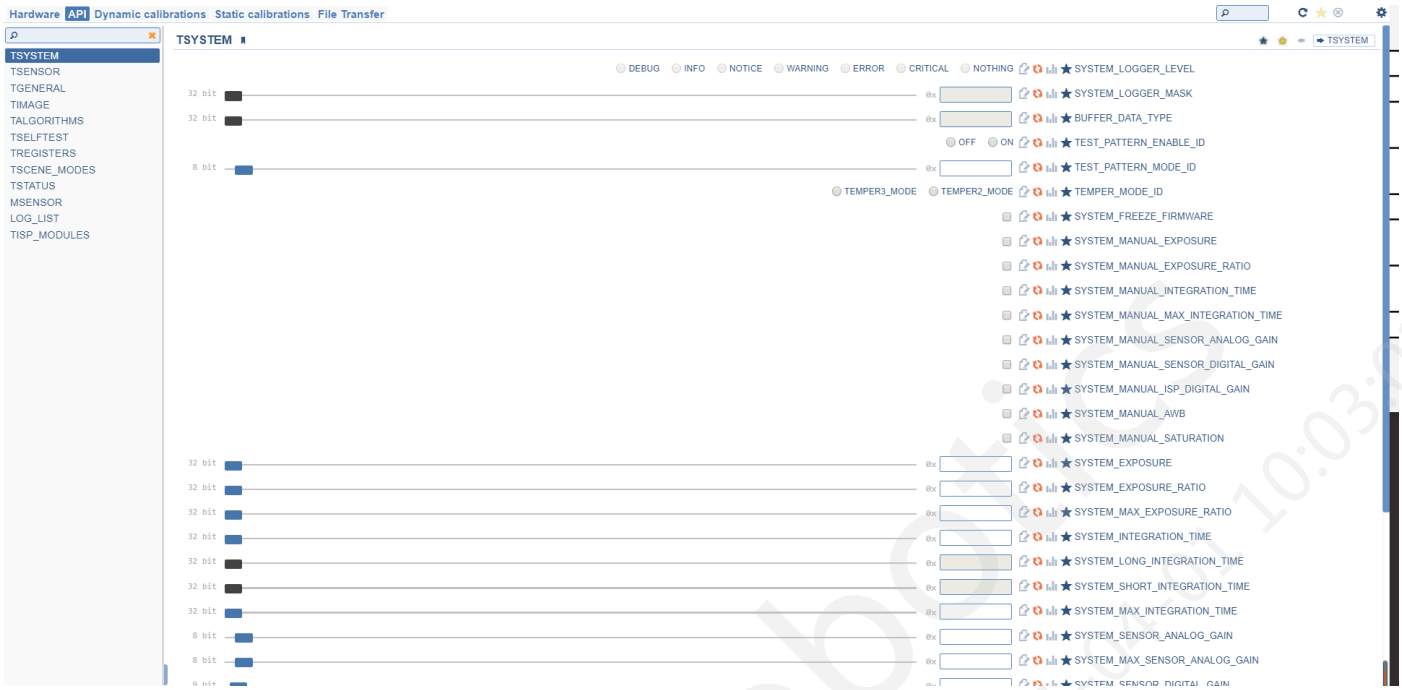


Figure 2.2-2 Control Tool

2.2.5.3 Hobot player

Hobot player is a applcation tool whichh can be used to preview,also used to save different image data, eg.YUV，BMP，JPG.Hobot player aslo can save Raw data.Using Hobot playerto save proper raw data for calibraion with Calibration tool.Refer to 《Horizon X3 Tuning tool introduction》. Hobot player dash board as Figure 2.2-3. Hobot player integrated record video function.

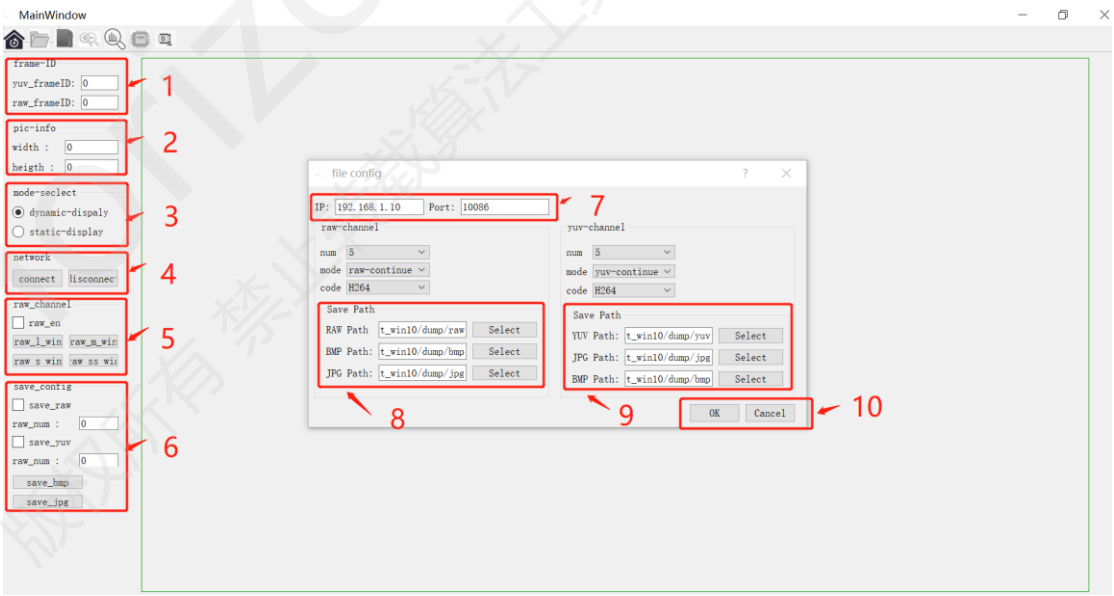


Figure 2.2-3 Hobot player Dash board

Items	Function	Description
1	Image frame-id	
2	Image size information	
3	Network transfer/display switch	
4	Network connection/Break	
5	Open raw display	
6	Image savingconfig	
7	Set network and Port	XJ3 IP
8	Set RAW saving path	
9	SetYUV saving path	
10	Confirm config	

Table 2.2-1 Hobot player functions Table

3 ISP Tuning Modules

3.1 Auto-Exposure (AE)

3.1.1 Overview

Auto exposure is to accurately restore the brightness of the current scene by adjusting the brightness of the image. Although AE plays a secondary role in characterizing the characteristics of the Sensor, you can use the Control Tool to adjust the AE in three steps.

- During phase one - when debugging other modules, AE can be set manually. This depends on the exposure requirements of each module, and the specific requirements will be explained in detail in the debug documentation of each module.
- During phase two - Refer to the objective index requirements, and correct the AE parameters of most scenes in the laboratory.
- During phase two - fine-tune the AE parameters to suit all scenes. The main adjustment is based on the parameters of the dynamic change of the gain.

3.1.2 Tuning Theory

AE is adjusted by Control Tool, there are a series of predefined parameters in Control Tool. AE is mainly driven by firmware parameters. During the adjustment of AE, the hardware registers must not be changed. The following is a brief description of the key parameters:

Key firmware parameters in API:

API parameters are in the "API" tab of Control Tool.

Parameter name	Source	Description
AE_mode_id	TALGORITHMS	Switch exposure mode auto/manual)
AE_gain_id	TALGORITHMS	Control total system gain
AE_exposure_id	TALGORITHMS	Returns exposure value
AE_roi_id	TALGORITHMS	Select the ROI for AE statistics
AE_compensation_id	TALGORITHMS	Offset exposure control
Enable_antiflicker	TSYSTEM	Enables antiflicker
Antiflicker frequency	TSYSTEM	Defines antiflicker frequency
Max_sensor_[A/D]_gain	TSYSTEM	Limits maximum [A/D] gain (sensor)
Manual_sensor_[A/D]_gain	TSYSTEM	Enable/disable manual control (sensor)

Sensor_[A/D]_gain	TSYSTEM	Manually set [A/D] gain (sensor)
Max_isp_digital_gain	TSYSTEM	Limits maximum digital gain (ISP)
Manual_isp_digital_gain	TSYSTEM	Enable/disable manual control (ISP DG)
ISP_digital_gain	TSYSTEM	Manually set digital gain (ISP)
Manual_integration_time	TSYSTEM	Enable/disable manual integration time
[L/S]_integration_time	TSYSTEM	Set [L/S] integration time
Max_integration_time	TSYSTEM	Limits maximum integration time
Manual_exposure	TSYSTEM	Enable/disable manual exposure control
Exposure	TSYSTEM	Manually set system exposure

Table 3.1-1 AE API Parameters

Note1: [A/D] actually represents two parameters, which respectively represent "analogue" and "digital".

[L/S] stands for "long" and "short" respectively.

"AE_compensation_id" is simply referred to as "AE_comp".

Note 2: The value under the TSYSTEM option corresponds to the value in the Dynamic calibration parameter cmos_control.

The Gain value in the above table is derived from the formula $2^{(m/32)}$. "M" is the Gain value displayed in the Control Tool, and the formula output value is the actual applied Gain value (expressed in multiples: 1x, 2x, 3x, etc.). The following table lists the common values in the lookup table:

Gain Input Value（m）	Gain Output
0	1x
32	2x
64	4x
96	8x
128	16x
160	32x
192	64x
224	128x

256	256x

Table 3.1-2 Gain Input and Output

Key firmware parameters in static calibrations

Static calibration parameters exist in the "static_calibrations.c" file and the "Static Calibration" option in the Control Tool.

Parameter Name	Parameter Description
EVTOLUX_ev_lut	EV value for conversion to lux
EVTOLUX_lux_lut	Lux value for conversion to EV

Table 3.1-3 AE Static Calibration Parameters

Key firmware parameters in dynamic calibrations

Dynamic calibration parameters exist in the "dynamic_calibrations.c" file and the "Dynamic Calibration" option in the Control Tool.

Parameter Name	Source	Parameter Description
N1 /AE_convergance	AE_Control	Adaptation time of AE in frames
N2 /AE_LDR_Target	AE_Control	LDR AE mean value target. This should match the 18% gray value of the output gamma
N3 /AE_tail_weight	AE_Control	Highlight weighting to avoid clipping
N4 /%_Max_Clipped_Long	AE_Control	WDR mode only: maximum percentage of clipped pixels allowed in long exposure. 0 = 0%, 128 = 50%,255 = 100%
N5 /Time_Filter_ER	AE_Control	WDR mode only: Time filter for exposure ratio
N6 /%_of_highlights_pixels	AE_Control	Control for highlights: percentage of pixels that should be below Hi_Target_Prc
N7 /Hi_Target_Prc	AE_Control	Control for highlights: highlights percentage target for the tail of the histogram
N8 /Enable_Iridix_global_Gain	AE_Control	1:0 enable,disable iridix global gain.This will disable the control for highlights logic
N1 /AE_HDR_Target	AE_control_HDR_target	AE minimum target value for HDR scenes. This is modulated by gain.
AE_correction	AE_correction	LUT defining AE_comp values that will offset the EV value calculated by AE in U1.6 format as EV*AE_comp where AE_comp value of 128 = 1
AE_exposure_correction	AE_exposure_correction	LUT defines lux in terms of EV

Refer to all "TSYSTEM" API parameters	CMOS_control	*Refer to all "TSYSTEM" API parameters*

Table 3.1-4 AE Dynamic Calibration Parameters

AE_correction and AE_exposure_correction work together. These must be set to contain the same number of nodes.

AE algorithm and controls

The AE algorithm obtains new EV values by analyzing histogram statistics and combining the calibrated AE target values. In order to take into account that both low and high dynamic scenes can be accurately exposed, the algorithm will dynamically adjust the AE target value. For example, in low dynamic range scenes, the AE target value will be adjusted based on the AE_LDR_target (18% gray) obtained from the calibration; for high dynamic scenes, the algorithm will be based on the "%_of_highlights_pixels" and "Hi_Target_Prc" values in the dynamic calibration parameters The AE target value is dynamically adjusted so that the calculated AE target value is not lower than the value in AE_HDR_Target. By adjusting in this way, you can retain the highlights in the high-dynamic scene, and then the local tone mapping engine (Iridix) controls the exposure of the low-light parts, thereby restoring the details of the low-light parts.

The HDR target value needs to be calibrated according to the dynamic range of the Sensor, and the Iridix engine restores the content of the low-light area.

AE_correction and AE_exposure_correction are responsible of adjusting the EV value calculated by the AE algorithm. This allows controlling the brightness according to lux levels given an EV value without having to change AE target tunings.

The AE convergence time can be adjusted by the AE_convergence parameter, which determines the number of frames to be adjusted when switching between two scenes with different brightness. Figure 3.1-1 illustrates how AE_convergence works. In this example, the value of 9 gives 9 frames over which AE can converge.

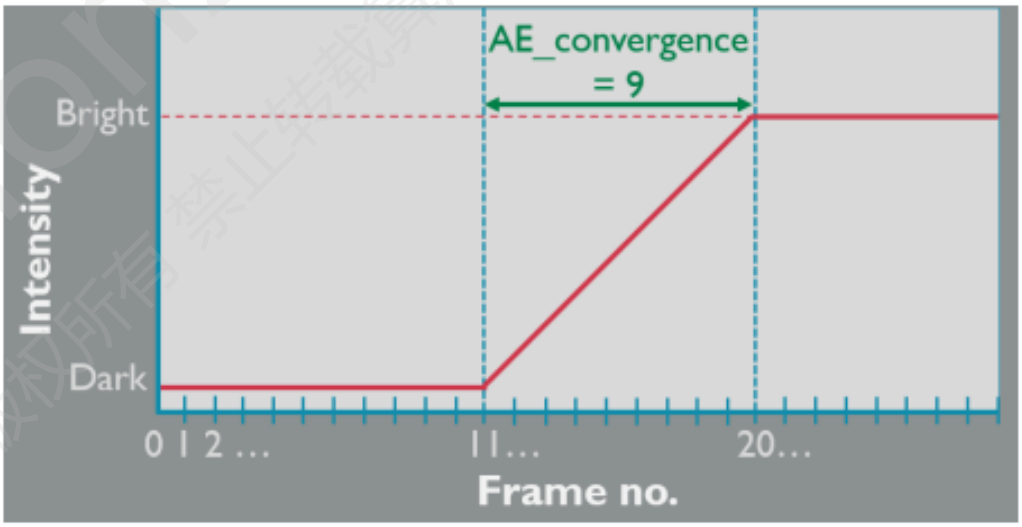


Figure 3.1-1 AE convergence

Figure 3.1-2 shows the design method of the AE_Control parameter. AE_tail_weigt is rarely used in Linear mode, but in all WDR modes, in order to avoid pixel overflow, it is recommended to adjust this parameter.

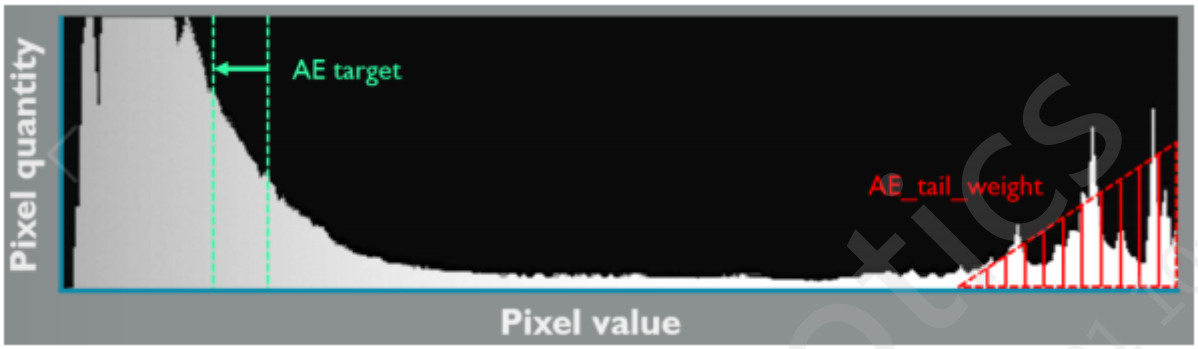


Figure 3.1-2 AE_target and AE_tail_weight example

If AE_tail_weight is not used, set it to 0. 0 means that the weights of the bins of the histogram are the same. Greater than 0 means that the bins between the end of the histogram and AE_tail_weight are given a higher weight. For example, setting AE_tail_weight to 11 means applying higher weight to the 11 bins at the end of the histogram, which will shift the AE_target to the left, thus avoiding the problem of highlight pixel overflow.

Note: The maximum value of AE_tail_weight is equal to the total number of histogram bins. Taking the 8-bit histogram as an example, the maximum value is 256. The recommended value of AE_tail_weight is less than 10% of the maximum value.

3.1.3 Tuning during phase one

In the first stage of Tuning, Black Level and AWB modules need to set the exposure value that meets the conditions (for specific values, please refer to the detailed description of each module). In the tuning process of some front-end modules of ISP pipeline, the requirements for AE are not high, and the accurate exposure value may not be reached. For this kind of module that does not have high requirements for AE, you can enable Manual AE through the Control Tool to control the brightness of the picture.

3.1.4 Fine tuning AE phase two

After completing the minimum prerequisites listed in Table 3.1-5. AE should be tuned as one of the initial steps of the second phase of tuning.

Prerequisite	Status / Value
Sensor driver	Fully tested
Black level	Tuned
Lens shading	Tuned

Iridix	Disabled
Autolevel	Disabled
Initial gamma	Set (Rec. 709 is recommended)
Set AE_compensation	128
Set AE_correction	128
Set AE_exposure_correction	25000
Set initial AE_LDR_Target	Objectively chosen according to the output gamma chosen

Table 3.1-5 AE phase two prerequisites

Other considerations:

- The value of AE_LDR_Target is within the acceptable range. When AE is applied, the average output value of the image histogram as Figure 3.1-3. The histogram mean should be the same as the brightness of patch 22 in ColorChecker or equal to 48% of the maximum pixel value. For example, to use Rec.709 type S-curve to enhance the contrast, the AE_LDR_Target value represented by the 10-bit histogram is around 256.An example of this is presented in Figure 3.1-3.

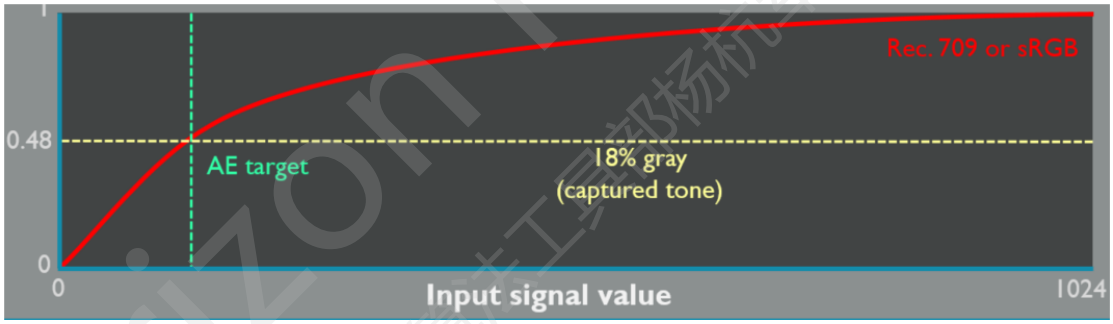


Figure 3.1-3 Correctly tuned AE target

- Adjust the Gamma curve to S-curve, which can effectively improve the contrast. But in the process of adjusting the Gamma curve, it must be ensured that the curve inflection point (the corresponding coordinate of AE_LDR_Target on the curve) cannot be changed. If the inflection point is changed, the value of AE_LDR_Target needs to be readjusted to ensure the same output value. See Figure 3.1-4 below:

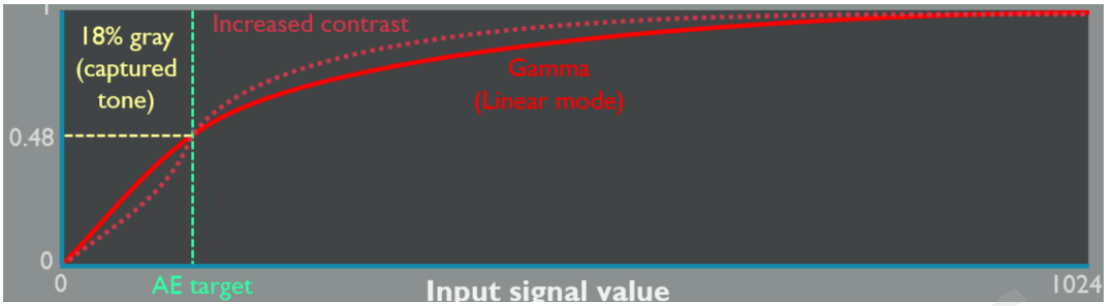


Figure 3.1-4 Correctly tuned AE_LDR_Target and contrast alterations

Note: It is recommended to set different AE_LDR_Targe and Gamma to compare the effect of the picture and find the optimal exposure and contrast of the system.

Tuning Procedure

See Figure 3.1-5 below:

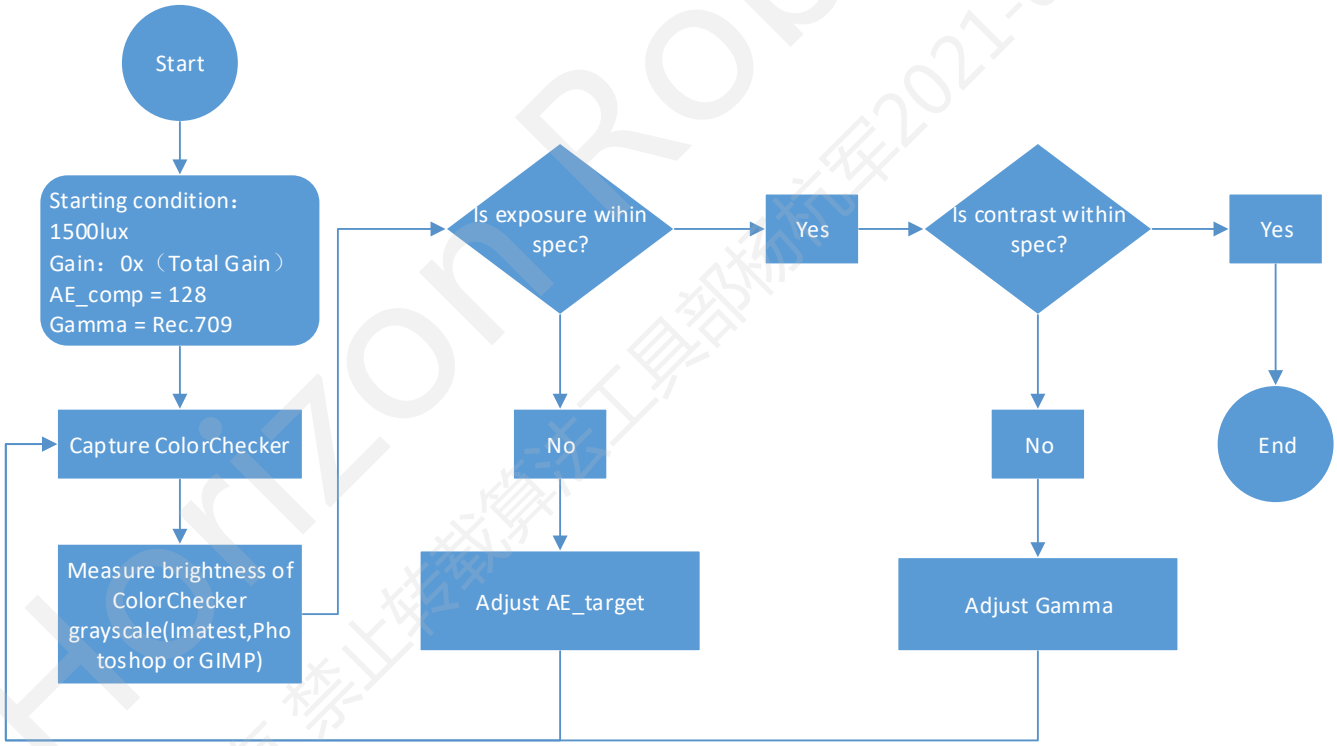


Figure 3.1-5 Flowchart of phase 2 AE Tuning

Check the initial performance of AE in a laboratory environment. Put ColorChecker in 1500lux environment, make sure ColorChecker accounts for 30%-50%(as shown in Figure 3.1-6). Take a picture and analyze the gray value through Imatest's ColorChecker function, or you can manually analyze the gray value in ColorChecker through Adobe Photoshop or GIMP.

Figure 3.1-6 ColorChecker Framing

- **Using Imatest Analyze:**

The Imatest function will output "exposure error" and "density response and curve fit" measurement indicators (as shown in the red box in Figure 3.1-7f-stop). The f-stop value is acceptable from -0.25 to +0.25. It is recommended that the measured f-stop value be between -0.1-0.

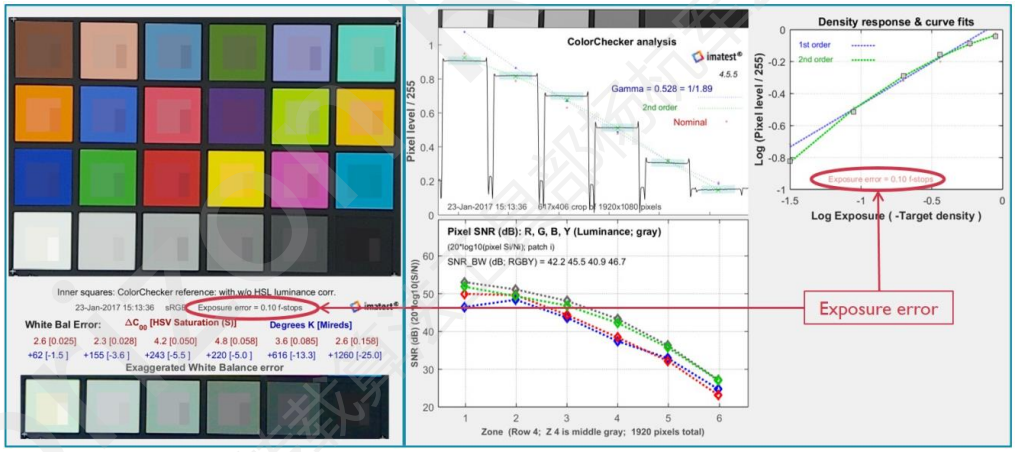


Figure 3.1-7 imatest exposure error plot

- **Using Photoshop or GMIP Analyze：**

Use the ROI selection tool to select a portion of each gray block in ColorChecker. For each patch, after defining the selected area, use the intensity/luminance histogram to calculate the mean of the selected area. Specific as Figure 3.1-8：

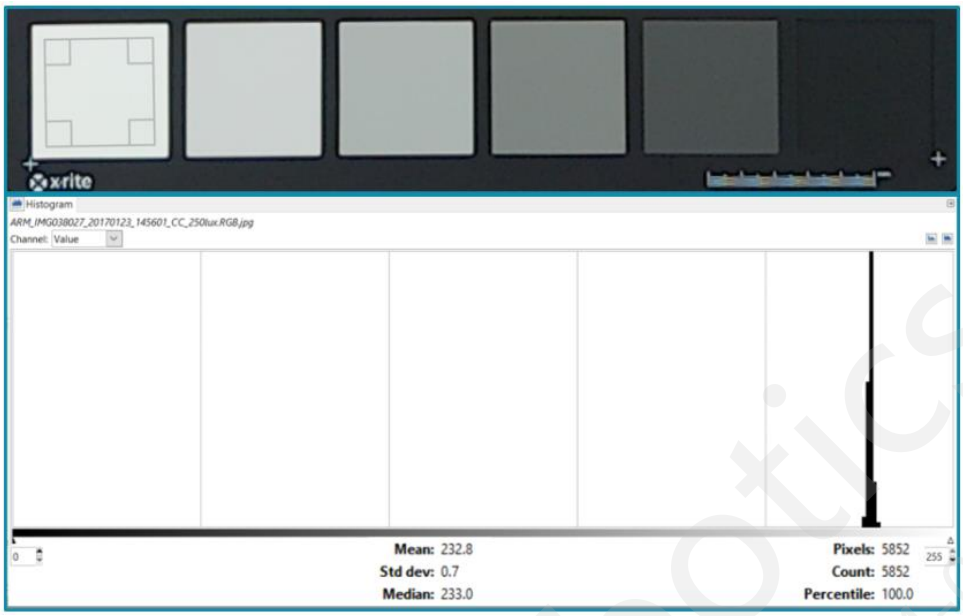


Figure 3.1-8 Pixel value measurement of grayscale patch

For the 8-bit histogram, the acceptable range of the average value of each gray block pixel is shown in Table 3.1-6. Generally speaking, it is expected that the average value of gray pixels measured can be in the range of -10-+10 of the ideal value. The best state is that the picture is slightly underexposed, and the difference from the ideal value is between -5 -0.

Patch No.	Ideal	minimum	maximum
19	240	230	250
22	122	112	132
24	40	30	50

Table 3.1-6 Recommented mean pixel values of ColorChecker grayscale patches

Note: Patch 22 reflects the value of 18% gray

In the case of no pixel overflow, making the actual value fall within the target range value can be done by adjusting AE_LDR_Target. Decreasing the AE_LDR_Target value reduces exposure, while increasing AE_LDR_Target increases exposure. When adjusting AE_LDR_Target, it is necessary to repeat the process of taking pictures and analyzing pictures. This is mainly to ensure that the system is operating within limits. Repeat the adjustment of AE_LDR_Target until the desired exposure error/pixel value is obtained. Once a pixel overflow occurs, the AE algorithm will use AE_HDR_Target as a starting point to find the maximum average value without pixel overflow. The final AE average target will fall between AE_HDR_Target and AE_LDR_Target. Set Enable_Iridix_global_Gain to 1 to enable the IRIDIX_GDG function, because it will compensate for the average AE reduced by reducing the exposure. The IRIDIX_GDG function uses AE_LDR_Target, Hi_Target_Prc, Hi_Target_Prc to calculate the gain value that the IRIDIX_Global_Gain module needs to use, so that the exposure can reach a state that can take into account the appropriate dynamic range before entering the iridix module. In the IRIDIX_GDG function, Hi_Target_Prc and Hi_Target_Prc have the same meaning as the corresponding variables in AE.

The above gives an overview of AE control and the factors that affect the AE algorithm. It seems that there are many parameters that need to be adjusted, but only some of these parameters need to be adjusted. The remaining AE parameters can use the default parameters. Only when there are special requirements that require further adjustment, adjustments are made based on the default parameters.

Example of default parameters in Linear mode:

```
static uint32_t _calibration_ae_ae_control[] = {
    RT, // AE convergence
    RT, // LDR AE target
    0, // AE tail weight
    0, // Max percentage of clipped pixels for long exposure
    0, // Time filter for exposure ratio
    100, //bright percentage of pixels
    99, // (hi_target_prc)
    1, // enable | disable iridix global gain
};

static uint16_t _calibration_ae_control_HDR_target[][2] = {
    //{gain , HDR target}
    {min gain *256, RT },
    {...* 256, RT },
    {max gain * 256, RT },
};
```

Example of default HDR_target parameters in Linear mode:

```
static uint16_t _ calibration_ae_control_HDR_target[][2] = {
    {0 * 256, 139},
    {1 * 256, 139},
    {2 * 256, 139},
    {3 * 256, 187},
    {4 * 256, 236},
    {5 * 256, 236},
    {6 * 256, 236},
    {7 * 256, 236}
};
```

Example of default AE parameters in WDR mode:

```
static uint32_t _calibration_ae_control[] = {
    RT, // AE convergence
    RT, // LDR AE target
    16, // AE tail weight
    77, // Max percentage of clipped pixels for long exposure
    15, // Time filter for exposure ratio
    100, //bright percentage of pixels
    99, // (hi_target_prc)
    1, // enable | disable iridix global gain.
};
```

```
static uint16_t _calibration_ae_control_HDR_target[][2] = {
    //{gain , HDR target}
    {min gain *256, RT },
    {...* 256, RT },
    {max gain * 256, RT },
};
```

Example of default HDR_target parameters in WDR mode:

```
static uint16_t _calibration_ae_control_HDR_target[][2] = {
    {0 * 256, 11},
    {1 * 256, 11},
    {4 * 256, 25},
    {7 * 256, 25},
    {9 * 256, 40},
};
```

3.1.5 Fine tuning AE phase three

Laboratory studio scene debugging

After completing the first and second phases of AE debugging, the third phase of subjective debugging can be carried out in the actual light box of the laboratory. During the third stage of the commissioning process, exposure can be increased and decreased according to needs or preferences. The purpose of this is to restore the actual scene as realistic as possible and improve the subjective acceptance of the output picture. In low light conditions, the balance between clarity and noise is particularly prominent. AE should be guided by specific needs, mainly because the entire camera system is mainly based on practical applications, and different application fields such as mobile phones, surveillance and industrial cameras have their own special requirements.

The third stage of debugging mainly includes two LUT tables, namely AE_exposure_correction and AE_correction. The two LUT tables implement the function of dynamically changing AE under different illuminances. The main process of debugging is shown in Figure 3.1-9.

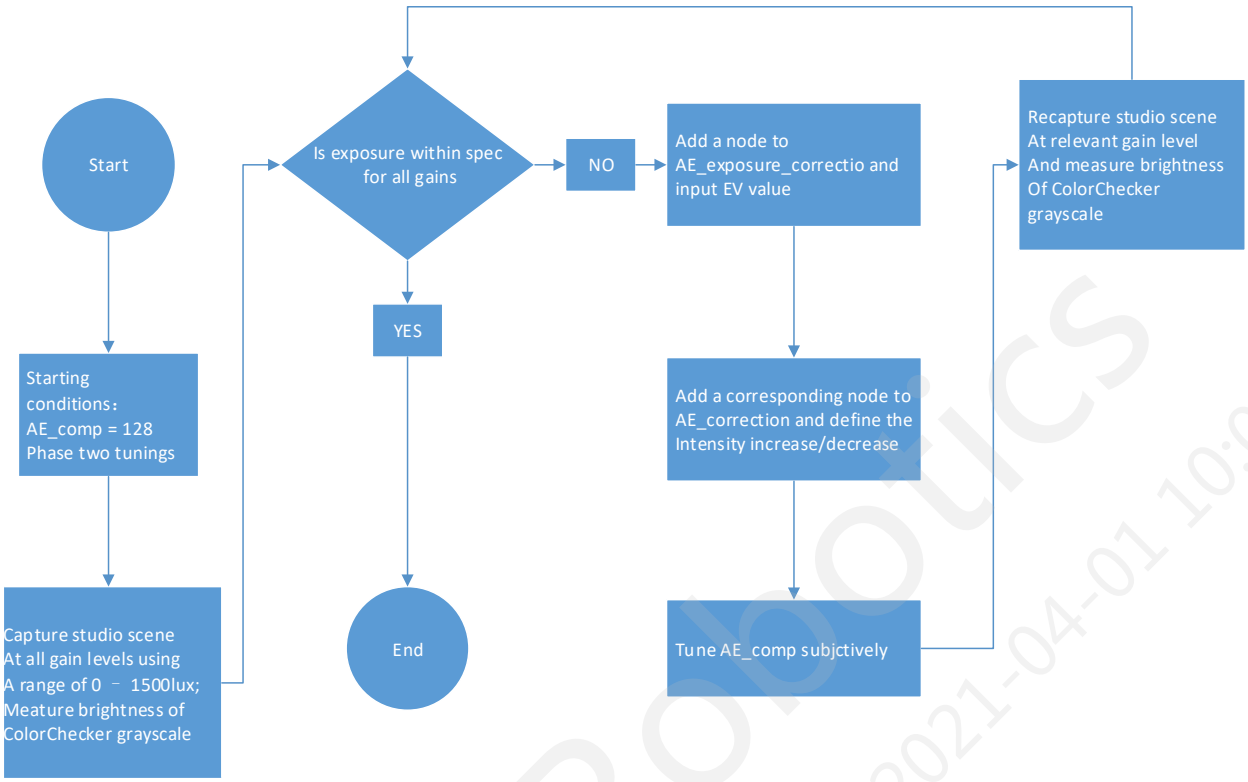


Figure 3.1-9 Flowchart of Phae three AE Tuning

First, a verification test needs to be performed in the studio scene light box. The light box should try to simulate the actual environment. The light box needs to contain an image test card, different colors and different textures. The brightness control can be Tcontrolled by a light box that can adjust the brightness. The illuminance range should include different brightness segments, which can be roughly divided into high brightness (above 3000 lux), medium brightness (100-3000 lux) and low brightness (0-100 lux). Under different brightness conditions, take photos separately and use the tools mentioned in the second stage (Imatest, Photoshop and GIMP) to analyze and confirm whether the exposure value at the current brightness is within the acceptable range of demand.

For the illuminance that AE cannot meet the requirements, you need to recalibrate according to the following rules:

1. Some systems have special requirements for low light, which requires additional debugging for specific needs. For example, surveillance cameras require that in low light, more image information can be retained in the picture; while items such as mobile phones and drones only need to be able to take beautiful pictures in low light. For special needs such as these, additional debugging work is required for low light. The specific debugging process is as follows Figure 3.1-10：

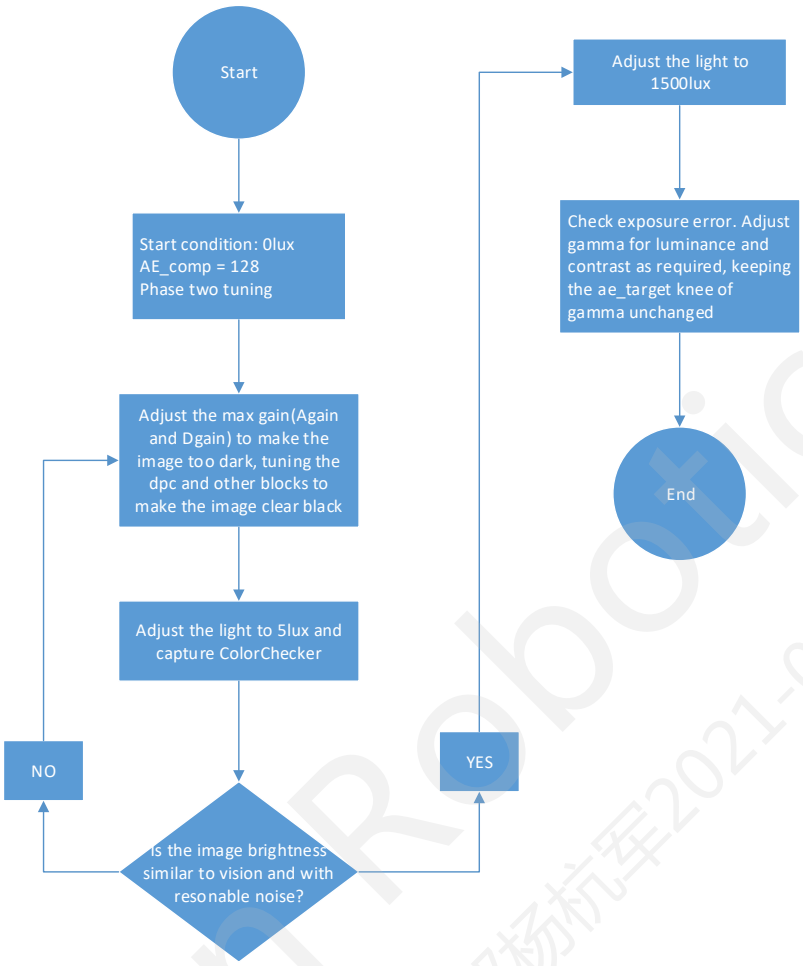


Figure 3.1-10 Flowchart of phase 3 AE Tuning

Reduce the brightness of the test environment to 0lux. Due to the limitation of Max_sensor_[A/D]_gain, the output image may be completely black. Then increase the brightness to 5lux, and modify the maximum value of Gain according to the subjective judgment of the ambient brightness. Adjust the brightness to 0lux again and observe the brightness of the output picture at this time. Adjust the Gain value repeatedly until the details and noise in the picture can meet the actual needs.

In the application case of security cameras, I would like to improve the low-brightness exposure to retain more details in the picture. In the case where the low light meets the demand, it is necessary to confirm whether the current parameters affect the effect of the high light scene.

2. In AE_correction, increase the node corresponding to AE_exposure_correction, and increase/decrease the average value of pixels (the reference value is 128) to meet the demand for brightness.

3. Tuning AE_comp. When calibrating AE_comp at low brightness, you need to reduce the maximum Gain (Max_sensor_[A/D]_gain) before modifying AE_comp. 128 is the reference value and will not have any impact on AE. Under most conditions, AE_comp maintains the reference value under the illumination of 100-3000 lux; illuminance above 3000 lux increases the value of

AE_comp; illuminance below 100 lux reduces the value of AE_comp appropriately. After debugging the AE_comp values under all illuminances, the graph shown in Figure 3.1-11 can be obtained.

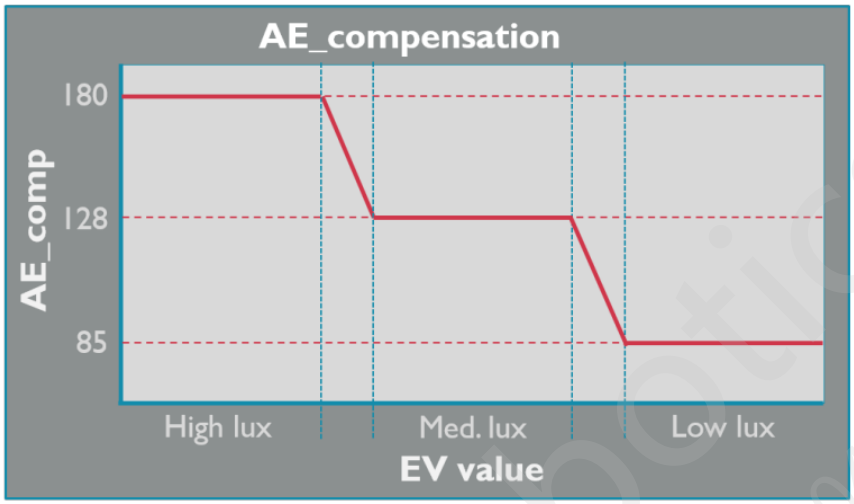


Figure 3.1-11 AE_comp vs EV plot

4. As Figure 3.1-11 shown，five nodes were created to define different illuminance ranges, which is far from enough, and further verification of AE under different illuminances is needed. For AE verification under each illuminance, the brightness value of each gray block needs to be analyzed through Imatest/Phototop/GIMP. If you still find that there is a value that falls outside the exposure range during the test, you need to repeat the above calibration steps. In the process of optimization, it is impossible to take into account all scenarios, and there will be problems of overlapping or conflicting parameters and nodes. In this case, it is necessary to choose according to specific needs.

Note: When adjusting the AE parameters, in order to avoid changing the previous parameters in the look-up table in the module, the adjustment should be adjusted in units of cycles (brightness-wise or Gain-by-bright). In the third stage, commissioning should start with the lowest gain of all relevant system parameters, not just the AE module. Then increase the gain and adjust all relevant system parameters on a gain-by-gain basis.

Field Testing

After all the laboratory commissioning is completed, it is finally necessary to perform commissioning in a real scene. This is mainly to discover the difference between the laboratory conditions and the actual application environment of the equipment. Before analyzing the debugging effect in the laboratory, you need to actually take photos covering a variety of scenes, and you need to capture pictures under various lighting and weather conditions to compare the color and texture reproduction performance. Use the reference camera to debug and shoot the same scene at the same time, then you can compare the debug camera image with the reference camera image, you can more intuitively find the characteristics of the debug camera and the reference machine and

the direction that needs to be optimized.

When subjectively analyzing the collected images, it is necessary to focus on the failure of laboratory debugging. It may be caused by poor exposure or excessive noise. The image output from the reference camera is regarded as the lowest acceptable quality, by continuously iterating the previously debugged parameters until the scene range meets the standard.

Additional tuning information

If there is still more noise in the image, you can try to move the dynamic range to the highlight area to reduce the exposure. This can be achieved by setting the HDR target to a higher value, but lower or equal to the LDR target. If noise is not a problem, it is best to use iridix to compensate and correct low-brightness areas while retaining the highlight information.

AE statistics are taken from the area defined by AE_ROI_ [X / Y] [1/2]. These four parameters define the start (AE_ROI_ [X / Y] 1) and end (AE_ROI_ [X / Y] 2) coordinates. From these two coordinates, the required statistical ROI can be generated, from which the AE statistics can be derived. By default, the ROI selection is set to full frame, but other commonly used RoIs are center weighted and spot metered. The process is shown in Figure 3.1-12.

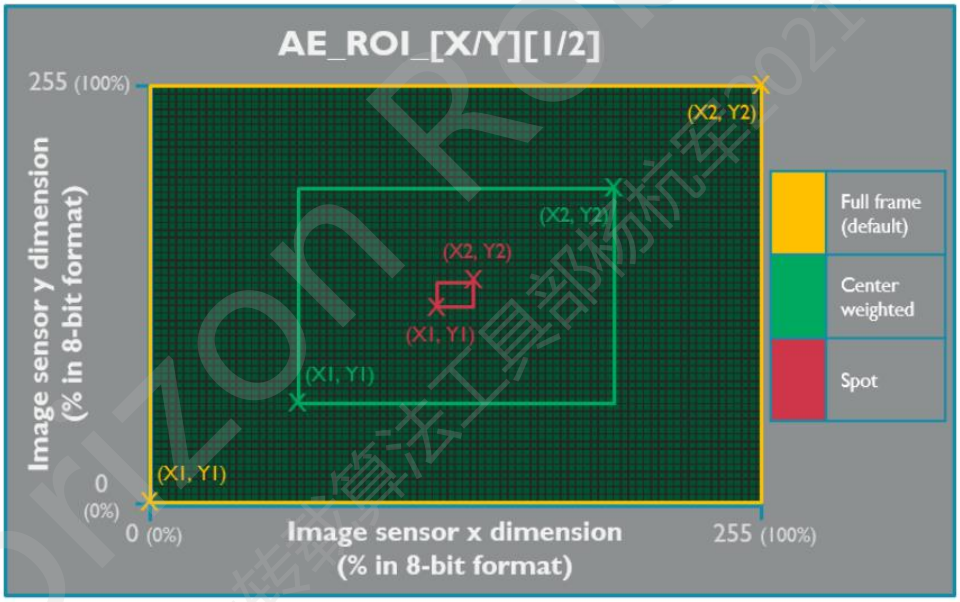


Figure 3.1-12 AE_ROI_[x/y][1/2] example

The area mapping depends on the grid settings, usually in the form of 16x16 or 32x32 grids, but this is determined by the system. Therefore, there is no default value for AE ROI. Figure 3.1-12 demonstrates the setting of 8-bit ROI.

3.2 Gamma

3.2.1 Overview

This module encodes the output color space gamma, adjusting the intensity ratio between input and output. In turn, it is possible to adjust the desired image contrast with this module. The tuning process itself consists of the following:

- During phase 1 - Gamma tuned in lab conditions for the majority of scenes, using Calibration Tool.
- During phase 2 - Gamma adjusted with respect to AE_target to improve image contrast (if necessary).
- During phase 3 - Gamma fine tuned so contrast suits all scenes (if necessary).

3.2.2 Tuning theory

Individual gamma LUTs are applied for each of the three color channels (R, G, B). LUTs consist of 129 evenly spaced nodes, where linear interpolation is applied by the hardware between these. Each data value is a 12-bit unsigned number, so it is expected that gamma[0]=0 and gamma[128]=4095, with the other node values defining the gamma correction curve.

Key hardware registers

Hardware registers are found on the relevant source page listed in the "Hardware" tab of Control Tool.

Register name	Source	Description
Enable	Gamma FR	Enable/disable gamma fr (full resolution /1st output pipeline)
Bypass_fr_gamma_RGB	Top	Bypass gamma fr
Bypass_ds_gamma_RGB	Top	Bypass gamma ds (down scaled / 2nd output pipeline)
Data	Gamma_rgb_mem	Gamma LUT for RGB correction

Table 3.2-1 Gamma hardware registers

Key firmware parameters for static calibrations

Static calibration parameters are found both in the "static_calibrations.c" file, and on the relevant page listed in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
Gamma	LUT for gamma

Table 3.2-2 Gamma static calibration parameters

3.2.3 Gamma & tone window

When you open the gamma & tone module, a window like Figure 3.2-1Figure 3.2-1 will pop up.

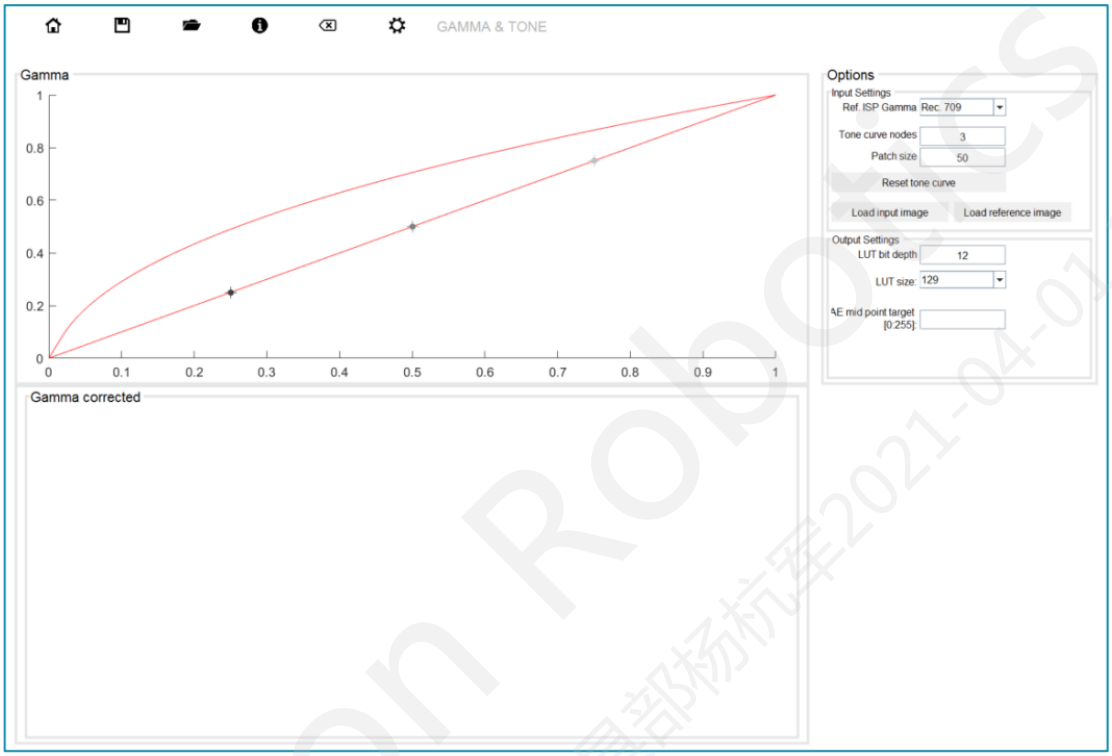


Figure 3.2-1 Gamma & tone window

3.2.4 Window features

Table 3.2-3Table 3.2-3 List of all regions and their functions in the gamma & tone window lists the function of each region.

Region	Sub-region name	Function
Gamma	-	Interactive tone curve plot
Gamma corrected	-	Displays input and reference images
Options	Input – Ref. ISP gamma	Choose a gamma curve which will be used as a basis for interactive tone adjustment

	Input – Tone curve nodes	Set the number of tone curve control points
	Input – Patch size	Set patch size
	Input – Reset tone curve	Reset the tone curve to the basic gamma
	Input – Load input image	Load an image from the sensor
	Input – Load reference image	Load a reference image (e.g. jpeg) from a benchmark camera
	Output – LUT bit depth	Set the bit depth of the LUT
	Output – LUT size	Set the number of LUT nodes
	Output – AE midpoint target	Set the value used for AE_target (refer to "ISP Tuning Guide")

Table 3.2-3 List of all regions and their functions in the gamma & tone window

3.2.5 Prerequisties

Correct image format and ISP mode set in the Home GUI and Settings window. In most cases, initial tuning (during phase 1) of gamma simply requires the "Ref. ISP gamma" to be chosen, before verifying settings, saving and continuing with other modules. However, gamma is often revisited while tuning auto-exposure (AE) during phase 2 of tuning to enhance contrast and tone. The procedure listed here is primarily for the latter, but also includes the relevant information for initial calibration. Refer to "ISP Tuning Guide" for a more detailed explaination about tuning phases.

3.2.6 Calibration images

When using gamma to adjust tone and contrast, it's useful to input a reference image. This not only helps visualize the tuning process, but also provides access to additional tuning information. This simply requires an image of a ColorChecker chart or grayscale as shown in Figure 3.2-2.

Figure 3.2-2 input ColorChecker image

3.2.7 Keyboard shortcuts

There are a couple of keyboard shortcuts worth noting:

- Space bar – Pressing the "space bar" affects the input image viewed in the "Gamma corrected" region of the ROI. Each time it is pressed, the image will toggle between a representation of the image with the default gamma, and that with the modified s-curve gamma. This will help to view the effects of modifying the gamma curve on contrast and tone more clearly.
- r – The "r" key has a similar function to the "space bar" when pressed. Here, the view will toggle between input image with the modified s-curve gamma and a loaded reference image. This is primarily used to visually compare current tone and contrast to that of a reference device.

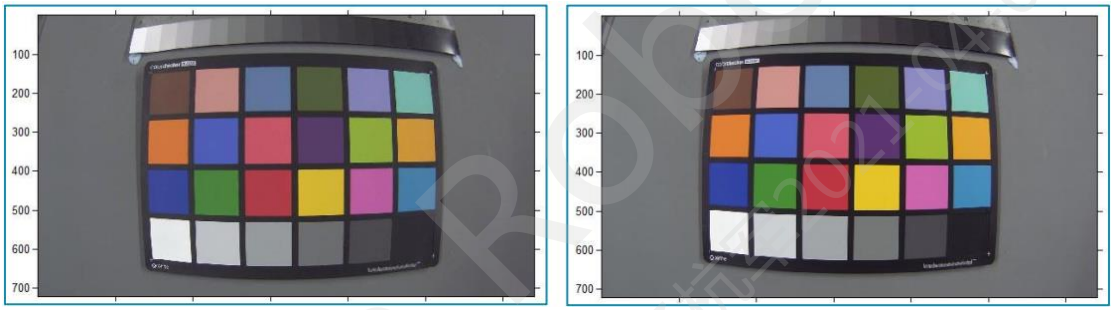
3.2.8 Calibration procedure

1. Open the gamma calibration module from the home window of Calibration Tool. A window similar to Figure 3.2-1 should now be visible.
2. Select the required ISP gamma for calibration using "Ref. ISP gamma". Out of the sRGB and Rec. 709 options the latter is the default choice and should be used if unsure. Alternatively, the "load from file…" option allows the user to load in a custom gamma.
3. Set the "LUT bit depth", "LUT size" and "AE mid point target" out of the options given for each.
4. Choose and set the number of tone curve nodes using the option provided. Note that the

amount is dependent on user judgement. Here, more nodes give the user greater control when shaping the curve. However, using fewer nodes can be beneficial as images are likely to look more natural from smoother transitions.

This value must be set before altering the tone curve. This is because each time this value is changed, any modifications to the curve will be reset. Note that the curve can be manually reset at any point by clicking the “Reset tone curve” button.

5. Load [an] input image using the button provided. Initially, this should be a ColorChecker or grayscale as previously mentioned. This can be used to measure the tone response. Additionally, several use-case images (one at a time) could be loaded in to assess the tone response in real scenes. If required/desired, a reference image can be loaded in using the “Load reference image” button, for comparison between two devices. This will help to align the tone responses, where the view between these can be toggled using the “R” key as described earlier.



(a)-Input image. (b)-Reference image.

Figure 3.2-3 Image comparison, input/reference

Note: In terms of scene content, the input and reference images should be as similar as possible.

6. Adjust the tone curve according to customer requirements or visual specification targets. This is done by clicking and dragging each of the nodes in the “Gamma” region of the GUI. It’s often the case that calibrations are carried out using visual judgement. But, these can also be verified by measuring the luminance and RGB values of ColorChecker patches. Clicking in the center of a patch will show an area of pixels that are being averaged to give these values, while displaying them to the user. The size of this patch can be adjusted using the “Patch size” parameter.

   Note: Make sure the area being averaged is within the patch border, otherwisecalculated values will be incorrect.

   This verification method allows for direct patch-to-patch comparison against a customer specification/target. These are usually defined for patches #19 and #24, or #13, #14 and #15. If the values do not meet the criteria, then the tone curve should be altered via the relevant nodes.

   Some general tips for this procedure follow:

a) **Dark regions** – Darkest areas of the image are influenced by the left-most region of the tone curve. Visually, these dark regions must be distinguishably darker than the rest of the time. This will help to enhance contrast.

b) **Mid tones** – Mid-grays are controlled using the middle region of the tone curve. These tend to influence the "overall" impression of tone, that is, if an image looks too dark or bright as a whole. Hence, if an image looks dark, mid tones should be increased, while a reduction is required if an image looks too bright.

c) **Bright regions** – The right-most region of the curve affects bright regions. It's important to avoid clipping the whites, while colors appear rich and bright.

   Ultimately though, there should be a good contrast range between dark and bright regions.

d) **System dependency** – Specified requirements and system characteristics can be more important than some of the criteria mentioned above. This normally involves either increasing contrast, or hiding noise:

   i. Increasing image contrast – This is done by applying a "typical" s-curve, where dark regions are reduced and bright regions are increased. This creates a sharper contrast as more pixel values are confined to these regions.

   ii. Hiding noise – If excess noise is occurring in bright regions, it may be preferable to clip certain bright values to avoid this. Alternatively, if the excess noise is visible in darker regions then reducing the curve in dark areas should suitably conceal this.

Note: Using the keyboard shortcuts (described earlier) can be very useful during this adjustment process. When combining the effects of toggling between images and measuring luminance and RGB values, tuning the curve becomes much easier.

7. Once gamma and tone match the specification/criteria, the parameter values can be saved using the save button. Figure 3.2-4 depicts one case of a tuned gamma and tone curve. Note that the shape of the curve may vary between systems.

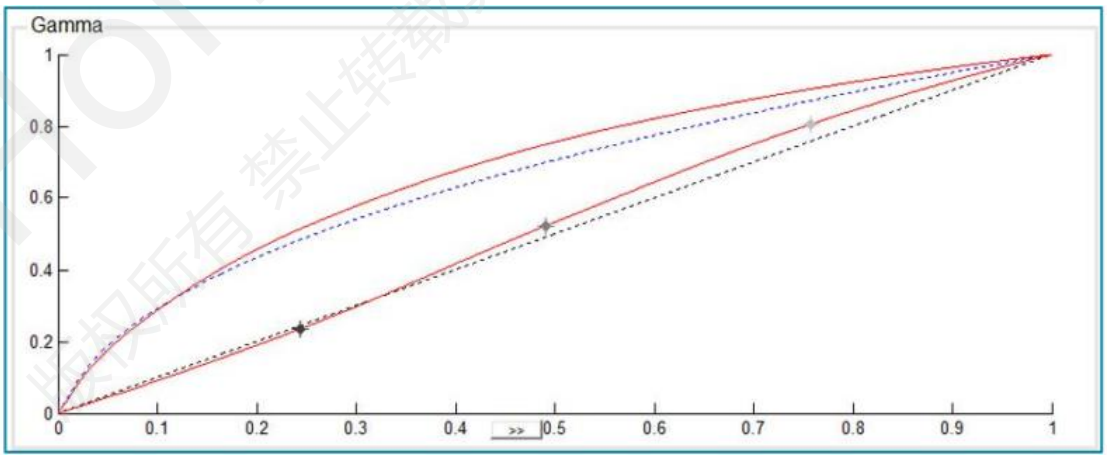


Figure 3.2-4 Tuned gamma and tone curve.

Additional tuning information

When gamma is changed, there are various modules that are affected such as AE, CCM and iridix, hence it is key that the affected modules be revisited and verified. For CCM, this means a full retune. On top of that, if AE_target is changed during gamma tuning, this will need to be accounted for in the AE module.

3.3 Auto-White Balance (AWB)

3.3.1 Overview

The AWB module is responsible of achieve color constancy as image sensor's response to neutral tones depends on scene illumination. These are most noticeable in neutral tones such as white and gray. AWB algorithms must deal with a wide range of lighting conditions and light spectrum to avoid undesirable color casts. This makes AWB to be one of the most challenging modules to tune and test.

Tuning procedure is described below:

- Phase one - Tune AWB parameters using Calibration Tool.
- Phase two -Fine tune AWB parameters to suit different lightings and modulating of parameters. During phase two, the AWB EVTOLUX LUT is tuned and enabled.
- Phase three - During this phase, AWB is fully tested and fined tuned for the scenes noticed with poor colour performance under some lighting conditions. Mix light and color appearance model tuning is also done during this phase.

3.3.2 Tuning Theory

Key firmware parameters for static calibrations

Static calibration parameters are in the "static_calibrations.c" file, and in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
EVTOLUX_ev_lut	EV data for EV to lux calculation
EVTOLUX_lux_lut	Lux data for EV to lux calculation
Light_src	Coordinates of extra illuminants
RG_pos	Red/green positioning of all illuminants
BG_pos	Blue/green positioning of all illuminants
Sky_lux_th	Lux value for natural light/sky
WB_strength_mult	Additional weighting applied to sky
Color_temp	Range of color temperatures in mired
CT_rg_pos_calc	Red/green value of given color temp

CT_bg_pos_calc	Blue/green value of given color temp
CT[65/40/30]pos	D[65/40/30] illuminant position in “Color_temp”

Table 3.3-1 AWB Static calibration parameters

Note: [65/40/30] indicates there are actually three parameters present. One for each of the “D65”, “D40” and “D30” illuminants.

Color_temp, CT_rg_pos_calc, CT_bg_pos_calc and CT[65/40/30]pos all work together. Color_temp defines color temperatures in mireds. CT_rg_pos_calc and CT_bg_pos_calc must contain the same number of nodes as Color_temp. They list the red/green and blue/green ratios for each color temperature. The value of CT[65/40/30]pos is based on the node position of the illuminant’s color temperature in Color_temp. This makes it possible to calculate color temperature for any illuminant.

Note: Mired is a measure of color temperature, given by the formula:

$$M = 1000000/T$$

where M is the mired value and T is the color temperature in kelvin.

Key firmware parameters for dynamic calibrations

Dynamic calibration parameters are in the “dynamic_calibrations.c” file, and in the ”Dynamic Calibrations” tab of Control Tool.

Parameter name	Description
EVTOLUX_probability_enable	Enable/disable use of lux probability

Table 3.3-2 AWB Dynamic calibration parameters

Note: During initial calibration, as the EVTOLUX LUT is not tuned, set EVTOLUX_probability_enable to 0.

3.3.3 Tuning during phase one

Prerequisites

Parameter name	Description
Black level	Tuned
Lens shading	Tuned
CCM	Tuned
GAMMA	Tuned
Manual exposure	-0:25 < exposure error < 0:25 (f-stops). This can be verified through Imatest.

Table 3.3-3 AWB phase one prerequisites

3.3.3.1 Calibration tool introduction

Calibration Tool automatically calculates most AWB parameters from a set of ColorChecker chart images. There are a large number of potential lighting conditions, the aim at this stage is to account for as many as possible in the lab environment.

To tune AWB, open Calibration Tool and click the AWB icon. A GUI similar to Figure 3.3-1 will appear.

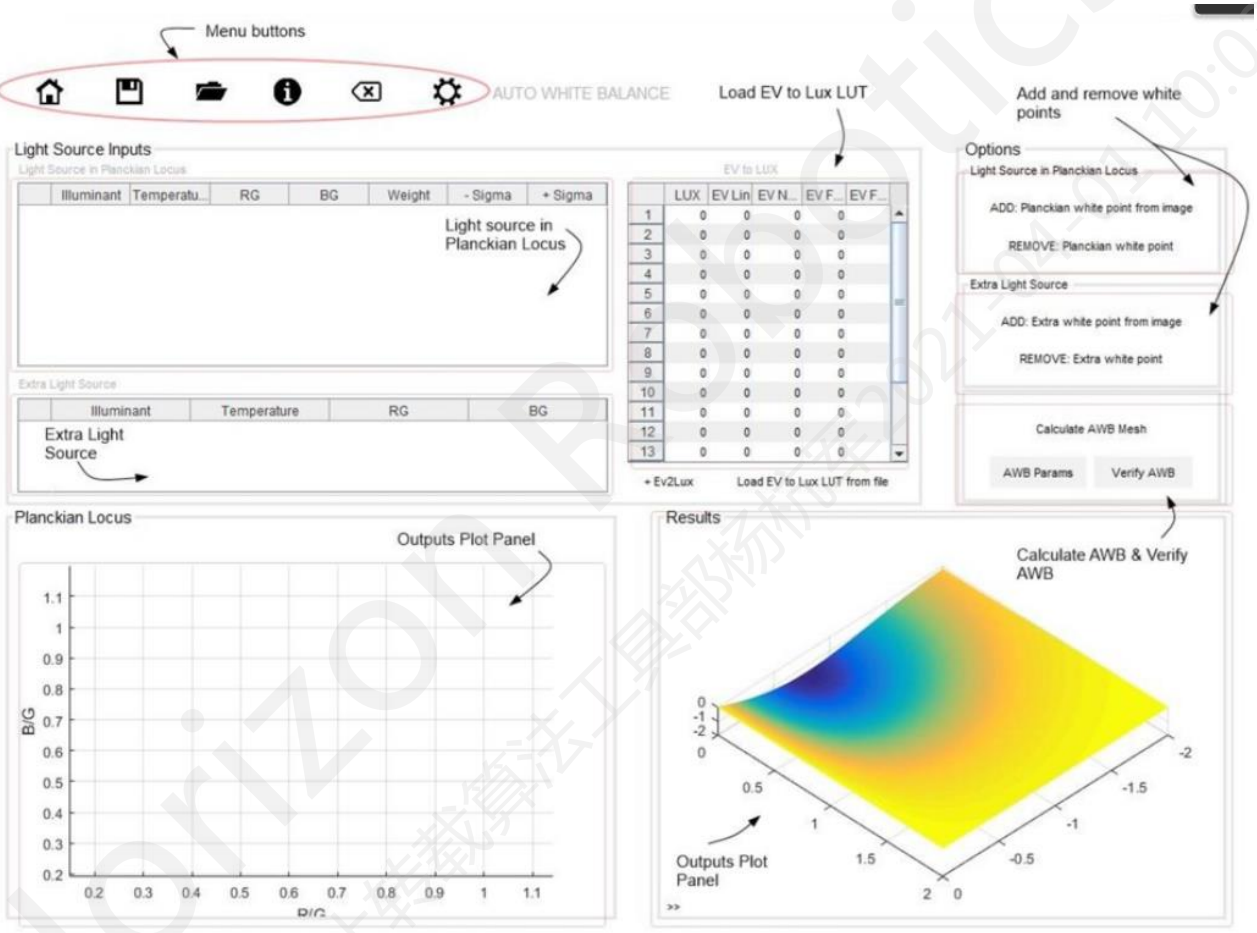


Figure 3.3-1 AWB GUI

Window features

Region	Sub-region name	Function
Illuminant Inputs	Illuminant on Planckian Locus	Table of Planckian illuminant data
	Extra illuminant	Table of non-Planckian illuminant data
	EV to lux	Table of EV to lux conversion look-up values

	New row	Open dialog for adding EV to lux data to table
	Clear table	Reset the EV to lux data
	Load EV to lux file	Import a full EV to lux LUT from a *.dat* file
Planckian Locus	-	Plot of illuminant data on Planckian curve
Options	Illuminant on Planckian Locus	Add/remove a Planckian illuminant to the Planckian Locus
	Extra Illuminant	Add/remove a non-Planckian illuminant [near] to the Planckian Locus
	Calculate AWB Mesh	Recalculate AWB parameters from current data
	AWB Params	Quick access to all AWB fine-tuning parameters
	Verify AWB	Input an image to verify AWB performance, refer to “AWBVerify_info” documentation for more information
Results	-	Displays the output AWB probability mesh
	>> button–Undock plots	Undock current plots from window – Allows for rotation, zooming, printing or saving the figure

3.3.3.2 Calibration images

Figure 3.3-2 Correct ColorChecker framing.

Capture a correctly framed ColorChecker chart under as many of the illuminants listed in table as possible. D50 and CWF illuminants are essential, and you must include at least 5 illuminants. LEE filters make it possible to create a wider range of color temperatures.

Note: Measure the color temperature of each illuminant using a colorimeter such as the Minolta CL-200A.

Illuminant	Color temperature	Planckian/non-Planckian
10K	10000K	Planckian
D75	7500K	Planckian
D65	6500K	Planckian
D50	5000K	Planckian
CWF	4150K	Non-Planckian (Extra)
D40	4000K	Planckian
TL84	3800K	Planckian
TL83	2800K	Planckian
A	2800K	Planckian

Horizon	2300K	Planckian

3.3.3.3 Tuning process

1. Open the AWB module of the ISP calibration Tool.
2. Add all your ColorChecker images to the Planckian Locus plot. There are two options for doing this:

   A. Automatic (recommended) – Use the "ADD: Planckian white point from image" button to process up to 10 Planckian illuminant images. Next, use the "ADD: Extra white point from image" to process the rest of your images.

   For each image, complete the popup shown in Figure 3.3-3 will appear, with a name for the illuminant and its measured color temperature.

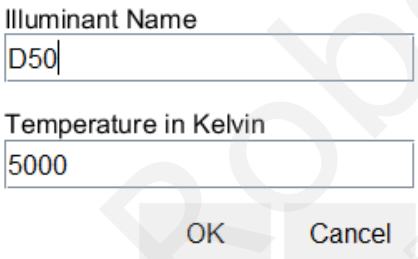


Figure 3.3-3 Illuminant dialog box.

   Click "OK". The ColorChecker patch selection window will open. Use this to align the 24 squares with the patches of the chart.

   B. Manually – Right click the Planckian Locus plot to add a point (Planckian or extra illuminant) at the mouse position as shown in Figure 3.3-4.

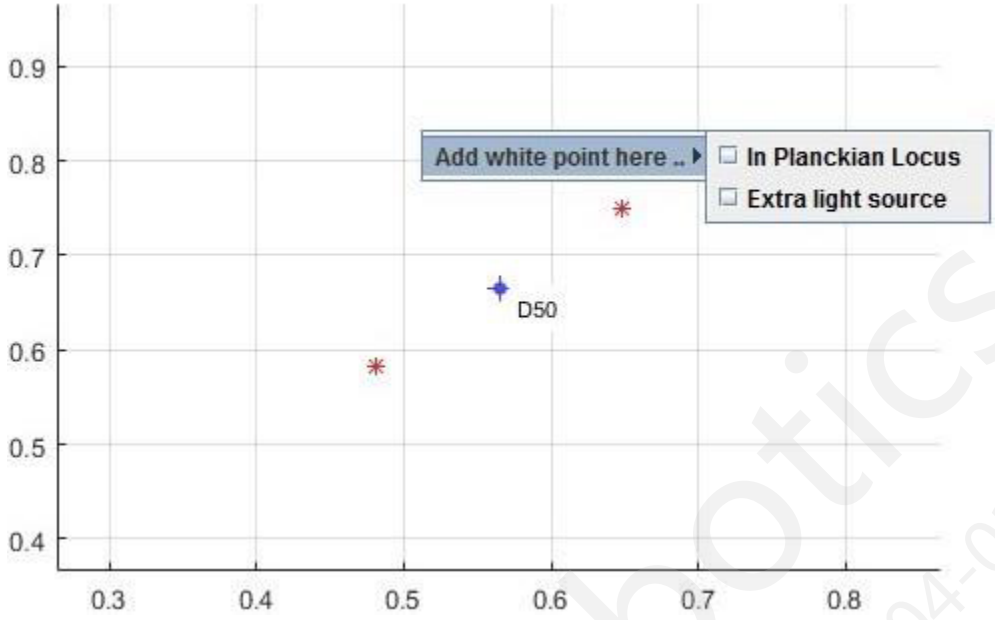


Figure 3.3-4 Manual addition of white point.

Depending on the sensor used, the illuminants falling in the Planckian Locus may vary. It's down to the users discretion to decide which points should be in the Planckian and which should be "extras".

3. Once a point is defined in the "Planckian Locus" plot (using R/G, B/G coordinates), it's data will be shown in the "Illuminant Inputs" region. Each of these points can still be adjusted, which may be necessary during AWB verification. This is required when verification of AWB shows inconsistencies between the captured image and the AWB for the illuminant captured. There are two methods of correction:

   A. In the "Illuminant Inputs" region, parameter values for each input color temperature can be adjusted manually. This is done by simply typing the new value into the relevant "RG" or "BG" cell, overwriting the previous value.

   B. Each point in the "Planckian Locus" plot can be selected and dragged using the cursor, but if you want to make precise adjustments, edit the r/g and b/g coordinates in the table instead.

4. In the case that two illuminants land very closely together, one can be deleted. Click "REMOVE Planckian/Extra white point" and select the illuminant you want to delete. Figure 3.3-5 shows the Planckian locus plot after all illuminants have been added.

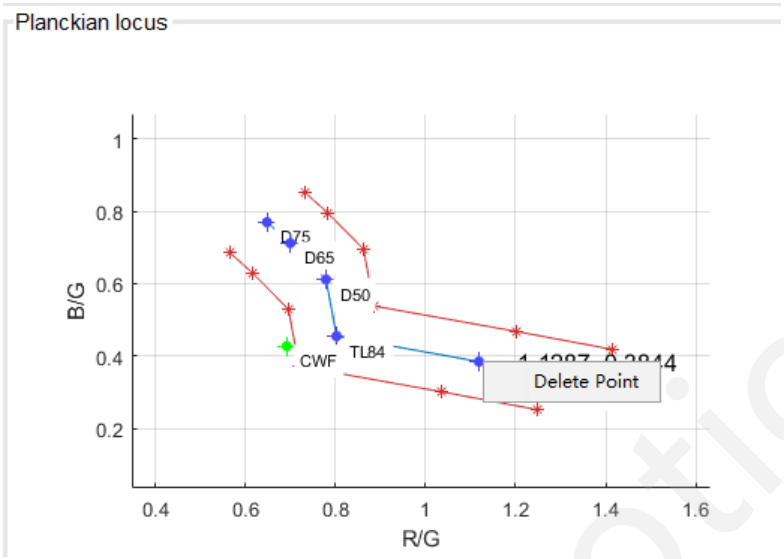
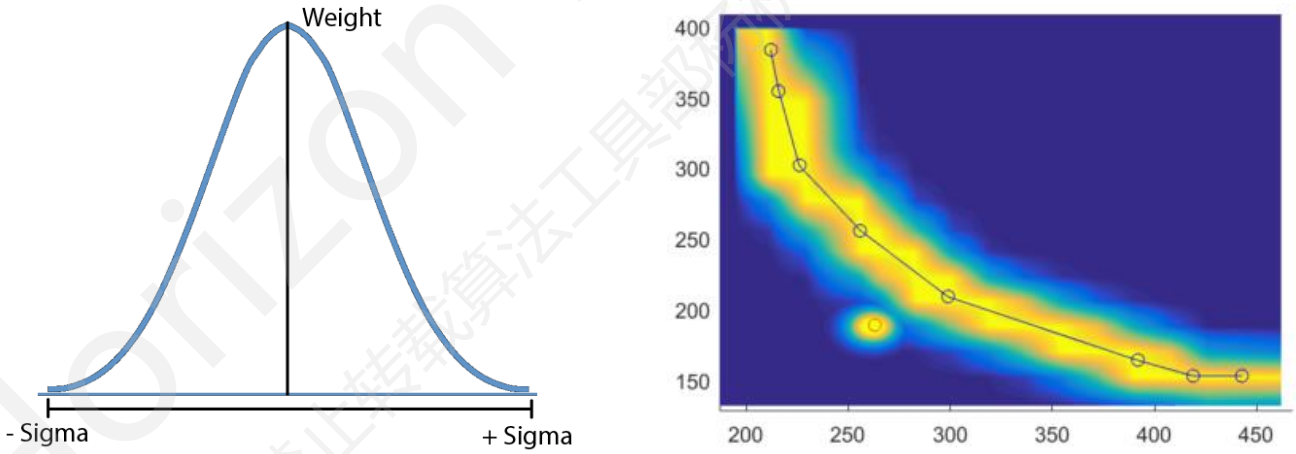


Figure 3.3-5 Planckian Locus curve after addition of white points.

5. The probability mesh is calculated using a truncated Gaussian distribution around these points (see Figure 3.3-6(a)). Looking at Figure 3.3-5, the highest point of the distribution is represented by the blue and green points. The red lines show where the spread of the distribution cuts off. A pseudo color plot of the probability mesh is shown in "results".

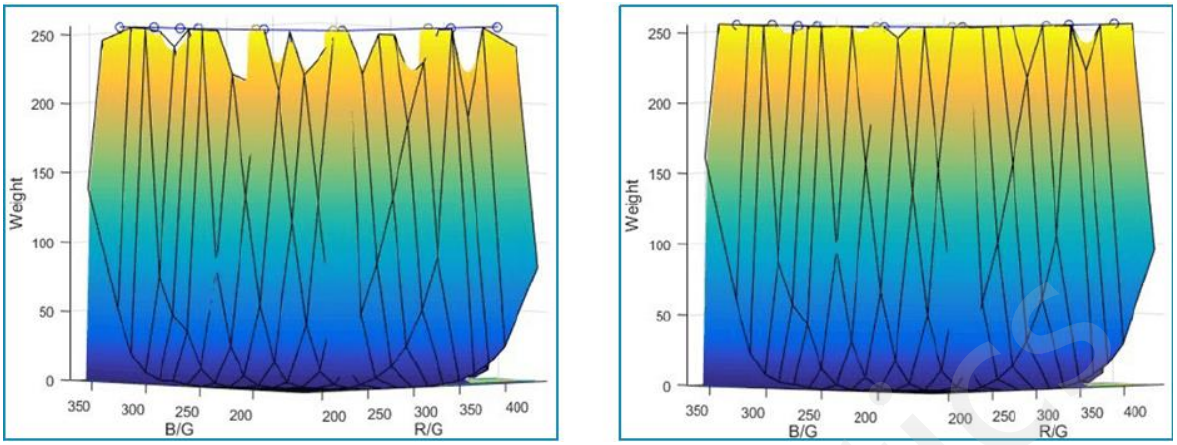


（**a**）– Gaussian distribution and parameter relationship.

（**b**）– Probability mesh in "Results".

Figure 3.3-6 Gaussian distribution of illuminant examples.

The generated probability mesh will need some manual correction. Select the ">>" undock button to show the "Results" figure. Rotate the view to see the side-on view of the plot. If this looks similar to Figure 3.3-7 (a), where weight drops between illuminants, continue with step 6. If it looks similar to Figure 3.3-7(b), skip step 6 and continue with step



（**a**）– Poor probability mesh.　　（**b**）– Tuned probability mesh.

Figure 3.3-7　Probability mesh plots.

6. Adjust the probability mesh is done using the following parameters:

   A. **Weight** – The height of the Gaussian distribution. This should be increased if the probability does not reach 100% (255). Note that this is the case in Figure 3.3-7(a). After correct alteration of weight parameters for relevant illuminants, the probability mesh should look similar to Figure 3.3-7(b).

   B. **- Sigma** – This parameter sets the lower limit of the Gaussian distribution. Along with "+ Sigma", the width of the probability mesh can be altered.

   C. **+ Sigma** – This parameter sets the upper limit of the Gaussian distribution. Along with "- Sigma", the width of the probability mesh can be altered.

7. Initial AWB tuning is complete. Save the calibration for now, and continue with EV to lux calibration after you have tuned AE (auto-exposure).

3.3.3.4　EV to Lux calibration

1. Position the camera in front of a dimmable illuminant tile, capable of producing <10 to >20,000 lux. Also make sure the entire image frame is covered (see Figure 3.3-8). Do not move the camera or illuminant during the EV to lux calibration procedure.

Figure 3.3-8 Correct EV to Lux illuminant framing.

2. Now position a lux meter at the same height and distance from the illuminant as the image sensor. Now take a reading, to measure the illuminance at the sensor (in lux). Adjust the dimming of the illuminant until the reading matches the target illuminance (see table below)
3. Allow a few seconds for the camera exposure to stabilize and record the Exposure Value (EV). In some systems this is called EV_log2 in the TCALIBRATIONS page in the API tab of Control Tool. In others this is called STATUS_INFO_EXPOSURE_LOG2_ID in the TSTATUS page.
4. Repeat steps 2 and 3 to cover the full range of lux values
5. Add the measured values to the AWB EV to lux LUT. This can be done in two ways:
   a. **Manually** – Enter the data one at a time using the New row button
   b. **Import a file** – Input a .dat file using the "Load EV to Lux file" button. This file must be in text format, containing space-delimited columns. The first column contains the lux data, and the second column contains EV values.

LUX	EV
20480	
10240	
5120	
2560	
1280	
640	
320	

160	
100	

Table 3.3-4 EV to lux LUT

Lux values do not have to match the ones listed in Table 3.3-4, but use these as a guide. Note the exponential spacing which gives extra data in the dark range.

The Calibration Tool can accept up to 18 nodes. Meaning, that for the setup shown in Table 3.3-4, additional lux levels can be added to increase precision if preferable. This is most useful for documenting the bend in the EV to Lux curve (see Figure 3.3-9).

6. To view the EV to Lux curve generated, select the ">>" undock plots button. This will provide a plot similar to Figure 3.3-9.

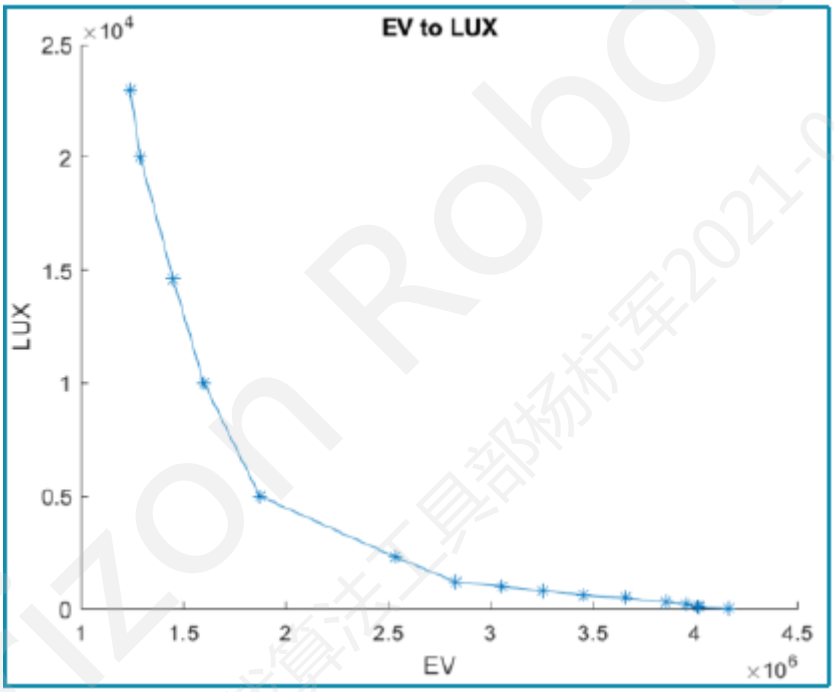


Figure 3.3-9 EV to Lux undocked plot.

Note: Adjusting AE after calibration of the EV to Lux conversion values will make them redundant. In turn, this means EV to Lux values will require recalibrating using the previously described process.

3.3.3.5 AWB verification

1. At this stage, all calibration data can be verified by selecting the "Verify AWB" button. Then follow the "AWBVerify_info" documentation provided in the new window for a full explanation of how to use this feature.
2. If verification shows that AWB is performing correctly, there is no need to change any settings. This means the user can skip to step 3. However, if there are some AWB failures,

these now need to be rectified through fine-tuning.

For each AWB failure, the verification process would have output “Red gain”, “Blue gain” and the color temperature of the illuminant. This will demonstrate where the point is estimated to land. From this feedback, the user can offset the necessary points. This is required until each failing image captured under a particular illuminant lands correctly on the probability mesh curve.

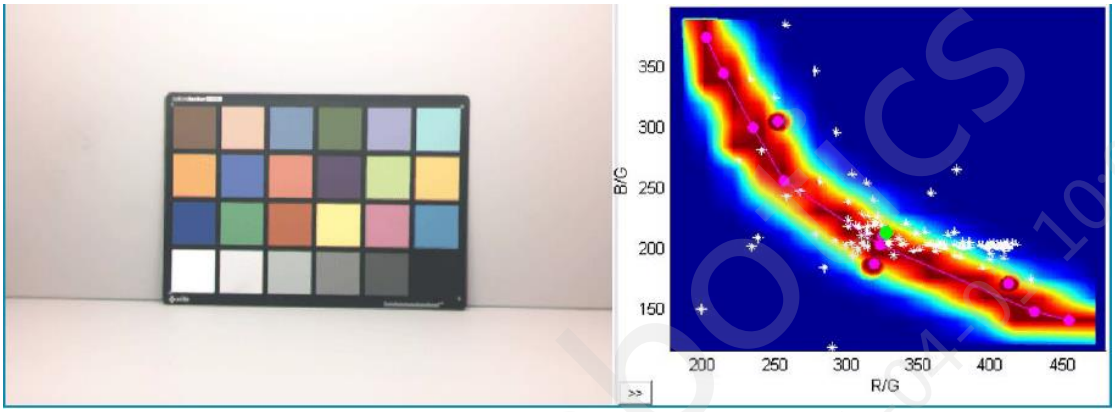


Figure 3.3-10 Failing AWB image and probability mesh.

Figure 3.3-10 shows an example of poor AWB performance. It can be seen in the image that the background has a pink tint, when this should be gray/white. Looking at the probability mesh curve, the estimated color temperature is “2841K”, even though TL84 (around 3,800K) was the illuminant used. This is amplified by the fact that most white points are falling outside of the curve.

Various adjustments can be made to fix this issue:

a) Firstly, run multiple images through the verification process from the failing illuminants. For the illuminants that are constantly failing, it’s likely the data point needs to be adjusted. Adjust the “RG” and “BG” values of the illuminant to move the position closer to the green data points using that illuminant.

   It’s possible that the “weight”, “- Sigma” and “+ Sigma” parameters have poor values. Undocking the verification plot will allow for better judgment of this. But if this is the case, the values should be adjusted as described in Calibration procedure – step 6.

b) Exposure can affect the outcome of colors and thus, the generation of the AWB probability mesh. If set too high, some of the patches will be overexposed or clipped. This could cause excessively high saturation, causing the failure. If set too low, some of the patches will be underexposed, this can also cause failures.

   Using the ColorChecker module in Imatest will output the “exposure error”. This can be used to evaluate exposure performance as described in the “AE” section of “ISP Tuning Guide”. Note that the ColorChecker image used should be captured under the same illuminant as that which is failing (the same image is preferable). If exposure is poor, adjust it, as documented in the same section.

c) If methods a) and b) does not improve the calibration outcome, check the EV2LUT.dat file used. Ensure that the values for this are up to date and accurate. If errors are observed, recalibrate the LUTs and reload the new file.

d) By now, if errors are still being observed, revert back and check the calibration for the shading module. If these are poor/incorrect, this will affect AWB and should be retuned.

Note: If the shading module requires recalibrating, other modules (as well as AWB) will be affected. Meaning each affected calibration module will need retuning. Refer to "ISPcalTool_info" on the home window for more info on affected modules.

e) The final method worth trying if AWB is still failing is to try and offset the points to their "actual" value. This can be done by altering the parameters of the points. If there are multiple illuminants in a small region of the curve, deleting one/some of these may be beneficial. Especially in the case of "extra" illuminants which land on the Planckian Locus.

3. Click the Save button to store your calibration and return to the home window.

3.3.3.6 Tuning verification using Imatest

Run ColorChecker chart images through the "Colorcheck" option in Imatest. This provides color information in a series of diagrams. Focusing particularly on the "Color Analysis" window, take the hue saturation value (HSV) for patches 20-22 and average them (see Figure 3.3-11). This can be used to judge AWB quality with reference to Table 3.3-5. Note the performance depends on image sensor quality and customer/application specifications.

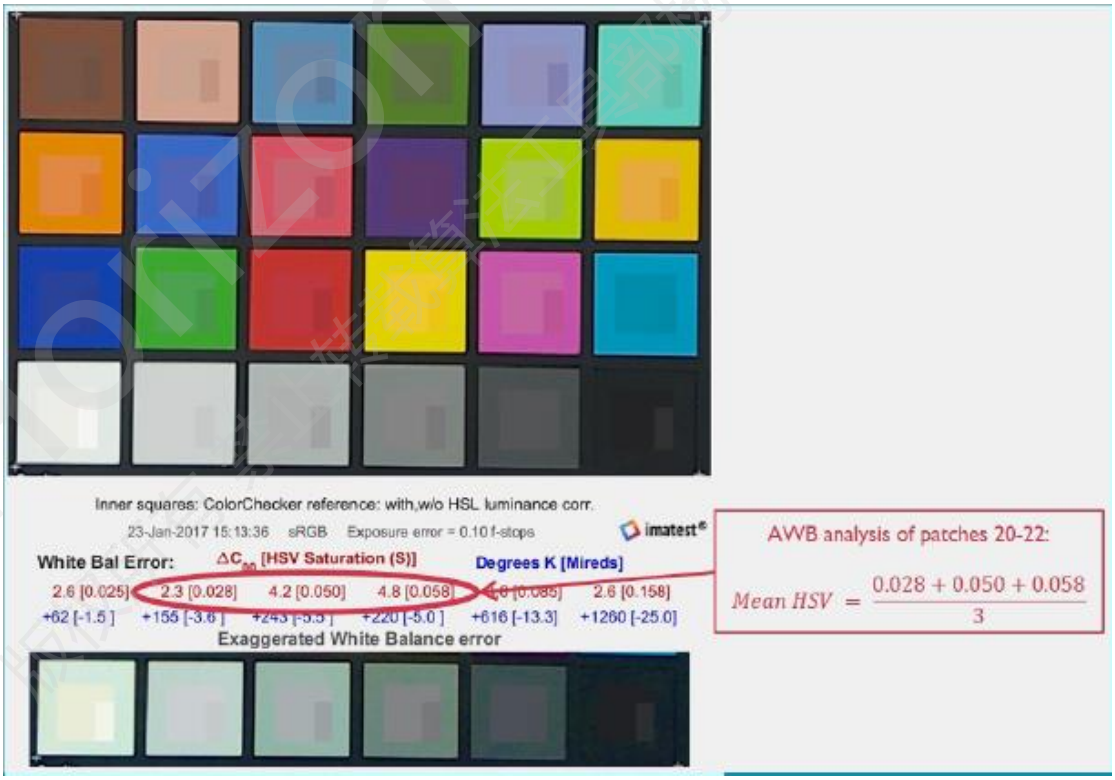


Figure 3.3-11 Imatest color analysis window showing patch 21 HSV.

AWB quality	6,500K (HSV)	4,000K (HSV)	2,700K (HSV)
Great	<0.025	<0.035	<0.030
Good	0.025 - 0.050	0.035 - 0.060	0.030 - 0.065
Fair	0.050 - 0.100	0.060 - 0.110	0.065 - 0.160
Poor	>0.100	>0.110	>0.160

Table 3.3-5 AWB HSV performance.

3.3.4 Tuning during phase two

Prerequisites：

Tuning continues in phase two after completing phase one prerequisites from Table 3.3-3.

Tuning procedure：

The EVTOLUX LUT helps distinguish between natural and artificial illumination. This is created in Calibration Tool under the AWB module. Instructions to create the LUT are in the AWB calibration guide used in phase one and two.

3.3.5 Fine tuning AWB phase three

Mix lighting AWB and Color appearance model tuning could be done during phase three, the tuning methodology refers to ISP calibration tool guide – AWB chapter.

3.3.5.1 AWB params window

When you open the AWB module, click the "AWB params" button. A window like Figure 3.3-12 will pop up.

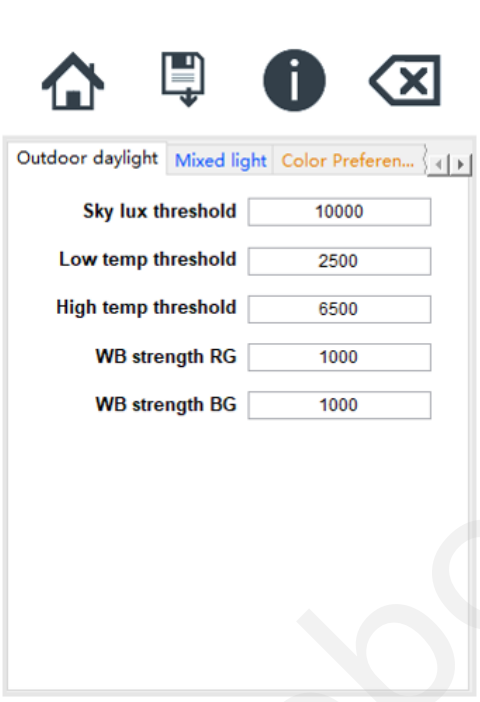


Figure 3.3-12 AWB params window

3.3.5.2 Window features

Table 3.3-6 lists the function of each tab.

Tab	Function
Outdoor daylight	Parameters for detecting and handling naturally illuminated scenes
Mixed light WB	Parameters for mixed light (multi-illuminant) correction
Color Preference Model	Parameters for balancing subjective beauty and objective accuracy

Table 3.3-6 List of all tabs and their functions in the AWB params window

3.3.5.3 AWB params window

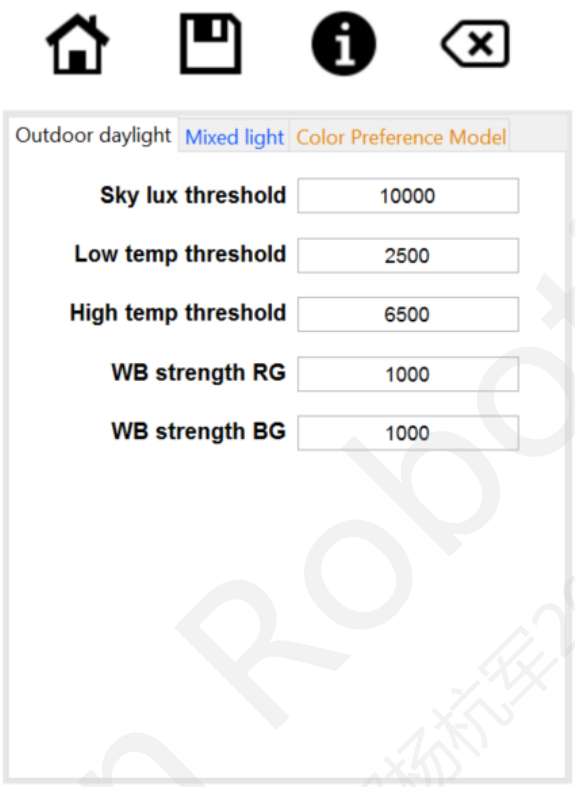


Figure 3.3-13 Outdoor daylight tab

Naturally illuminated (sky-lit) scenes have a limited range of possible CCTs. The outdoor daylight tab handles this.

If the scene illuminance (in lux) is greater than Sky lux threshold, only the statistics that represent a CCT between Low temp threshold and High temp threshold are used.

The performance can be controlled further by adjusting WB strength RG and WB strength BG which are gains applied to the statistics. A value of 1000 means 1x gain.

3.3.5.4 Tuning mixed light WB tab

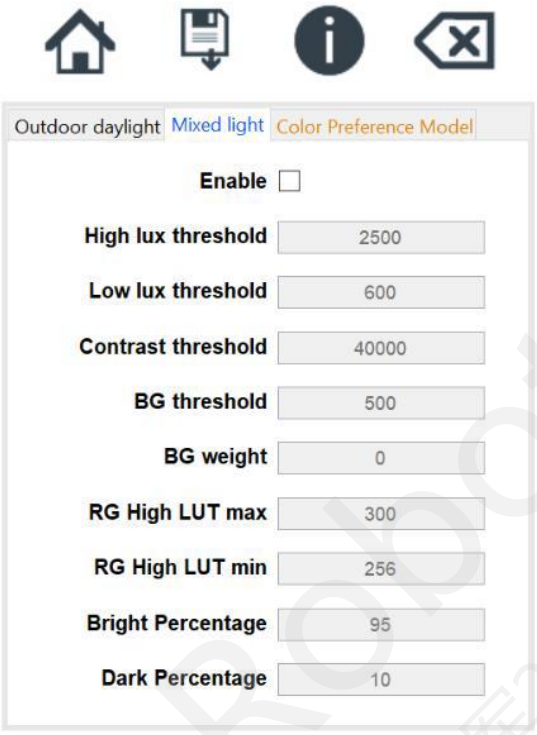
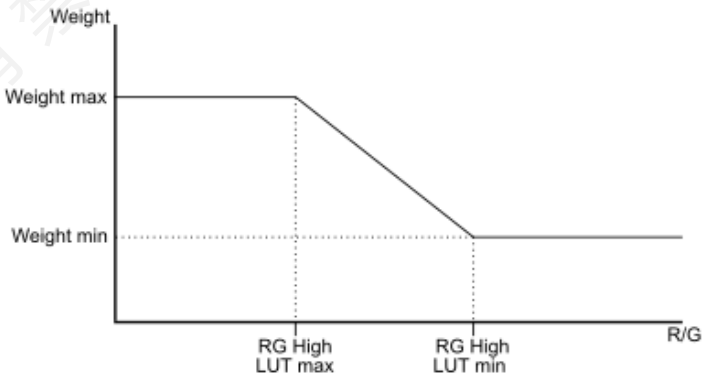


Figure 3.3-14 Mixed light WB tab

The AWB algorithm will apply mixed light compensation when these conditions are met:

- Scene illuminance > **Low lux threshold**
- Scene illuminance < **High lux threshold**
- Histogram contrast > **Contrast threshold**

Histogram contrast is controlled by **Bright Percentage** and **Dark Percentage** to determine the strength of output contrast. Mixed light compensation manipulates the weight of statistics based on their R/G and B/G ratios. Statistics with B/G > **BG threshold** will be given a weight equal to **BG weight**. Statistics are also weighted based on their R/G value as shown below:



3.3.5.5 Tuning color preference model tab

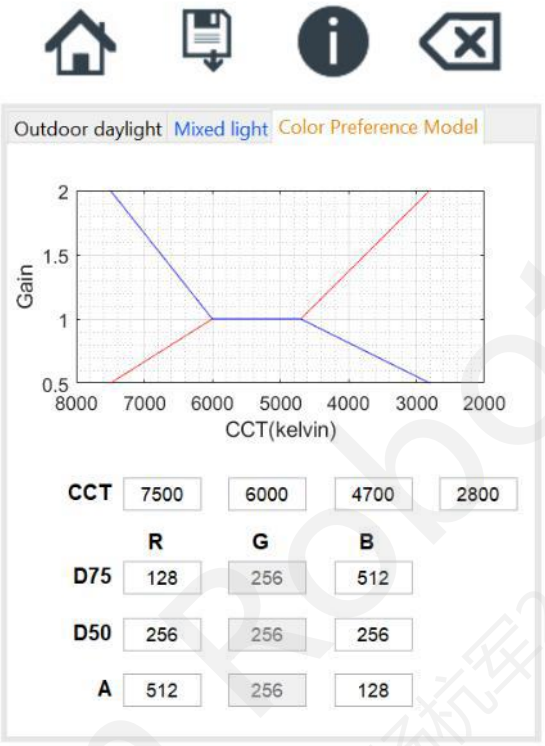


Figure 3.3-15 Color preference model tab

The AWB algorithm produces a gain value for each color channel that will make grey objects appear grey. However, you might want to manipulate these gains to prioritize beauty over accuracy. The color preference model applies an extra 8-bit precision gain to each color channel after AWB and divides the color temperature range using 4 CCT values.

The plot in Figure 3.3-15 shows the how the parameters are used.

- Images with a measured CCT higher than the largest value (7500 in Figure 3.3-15) will be modified using the gain values listed on the D75 row.
- Images with a measured CCT between the middle values (4700 to 6000 in Figure 3.3-15) will be modified using the gain values listed on the D50 row.
- Images with a measured CCT lower than the final value (2800 in Figure 3.3-15) will be modified using the gain values listed on the A row.

3.3.5.6 AWB verify

After AWB fine tuning，According to the current parameters, batch simulation output the results after AWB function.

3.3.5.6.1 Verify AWB window

When you open the AWB module, click the "Verify AWB" button. A window like Figure 3.3-16 will pop up.

Figure 3.3-16 Verify AWB window

3.3.5.6.2 Window features

Table 3.3-7 lists the function of each region.

Region	Sub-region name	Function
Images	-	Displays ColorChecker image (if loaded)
Options	Input	Load an image for verification
	Static White Gains	Edit gains to adjust the static white point
	Exposure Value Input	Input for exposure value (EV) or load it from txt file

	Grey Patch Test	Run the grey patch test
	CCM	Apply CCM model
	Save results and plots	Save results and plots
	Color Preference	Apply Color Preference on verify image
	Mixed Light	Apply Mixed Light compensation on verify image
	Apply digital gain	Brighten the image before analysis (for intentionally underexposed HDR images). Set the gain to 0 for automatic gain calculation. Note: the AWB stats won't be accurate if digital gain applied.
Results	-	Probability plot with overlaid AWB image statistics
	>> *button*	Undock current plots from window – Allows for rotation, zooming, printing or saving the figure

Table 3.3-7 List of all regions and their functions in the verify AWB mesh window

3.3.5.6.3 Verification procedure

1. After clicking the "Verify AWB" button in the AWB calibration module a window similar to Figure 3.3-16 should be visible.
2. Before the verification process can begin, the EV needs to be set for the chosen verification images. This is done in a variety of ways:
   a. Automatically (recommended) – Selecting the "Read EV from header .TXT" option will automatically read the EV of the input image.
   b. Manually – Uncheck the "Read EV from header .TXT" radio button to enable the "EV:" input box. This will allow EV to be manually entered.

   If importing multiple images for verification, the EV input method should be chosen first (using the radio button). If manually entering EV, no value will be required in the "EV:" input box. Instead, when using multiple images, an additional window will open to manage them (see Figure 3.3-17(b)). Input the values using this new window. EV will automatically be input for all images if using the "Read EV from header .TXT" functionality is used (Figure 3.3-17

(a)).

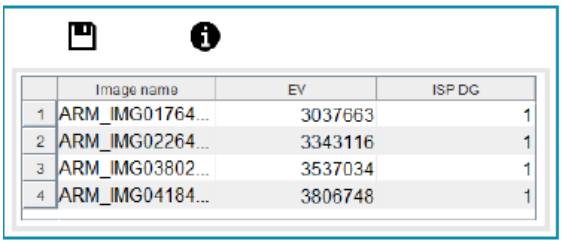


	Image name	EV	ISP DG
1	ARM_IMG01764...	3037663	1
2	ARM_IMG02264...	3343116	1
3	ARM_IMG03802...	3537034	1
4	ARM_IMG04184...	3806748	1

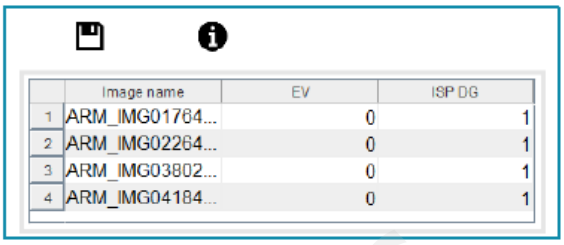


	Image name	EV	ISP DG
1	ARM_IMG01764...	0	1
2	ARM_IMG02264...	0	1
3	ARM_IMG03802...	0	1
4	ARM_IMG04184...	0	1

（a）- Multiple image input, automatic EV detection. （b）-Multiple image input, manual EV detection.

Figure 3.3-17 EV input options for multiple images

After saving the values using the save button, an EV_data.dat file will be output so this does not have to be repeated.

If EV values were entered into the EV to Lux table in the AWB calibration module (prior to AWB verification), an alternative is provided. A window similar to Figure 3.3-17 (a) will appear with EV already input. The user will also be prompted to confirm/modify the values as required.

Note: If EV values are not entered, there is a possibility of inaccurate color temperature estimation. Also, changing the EV will have no effect until the verification images are reloaded.

3. (Optional) – Selecting the "Run grey patch test" option can be used for testing the accuracy of AWB over a batch of sample images. This must be done before loading in the image, as the patch selection functionality is run during the import process.

   When this process is run, a container window similar to Figure 3.3-18 will open requesting an ROI to be placed over the grey patch. The cursor should then be used to define the top left, followed by the bottom right corner of patch #21.

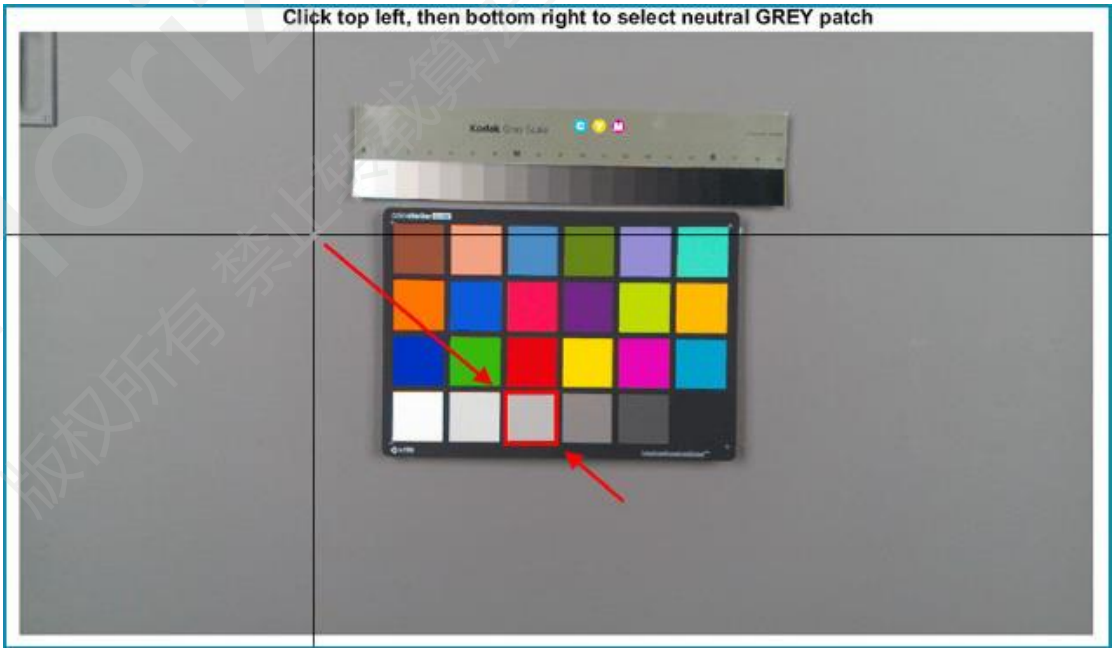


Figure 3.3-18 Grey patch selection process

The output from this process is are two files, "coords.dat" and "AWB_mesh_ISP.dat", found in the image folder. "coords.dat" stores the ROI coordinates for each of the images that have been processed. Meaning that the next time this is done for the same image, the ROI is automatically determined. The "AWB_mesh_ISP.dat" file provides the following AWB statistics:

- Image name
- Mean red value of the patch
- Mean green value of the patch
- Mean blue value of the patch
- R-G
- B-G
- R/G
- B/G
- Delta ex. deviation from natural grey

4. It's recommended to run the grey patch test every time the probability mesh is modified.

   Import the image(s) to be used for verification through selecting the "Load image(s)" button. This should originate with the ColorChecker images used for calibration, but further verification can be done on any captured image, including field test imagery.

5. Once the image(s) are imported, the window will look like Figure 3.3-19.

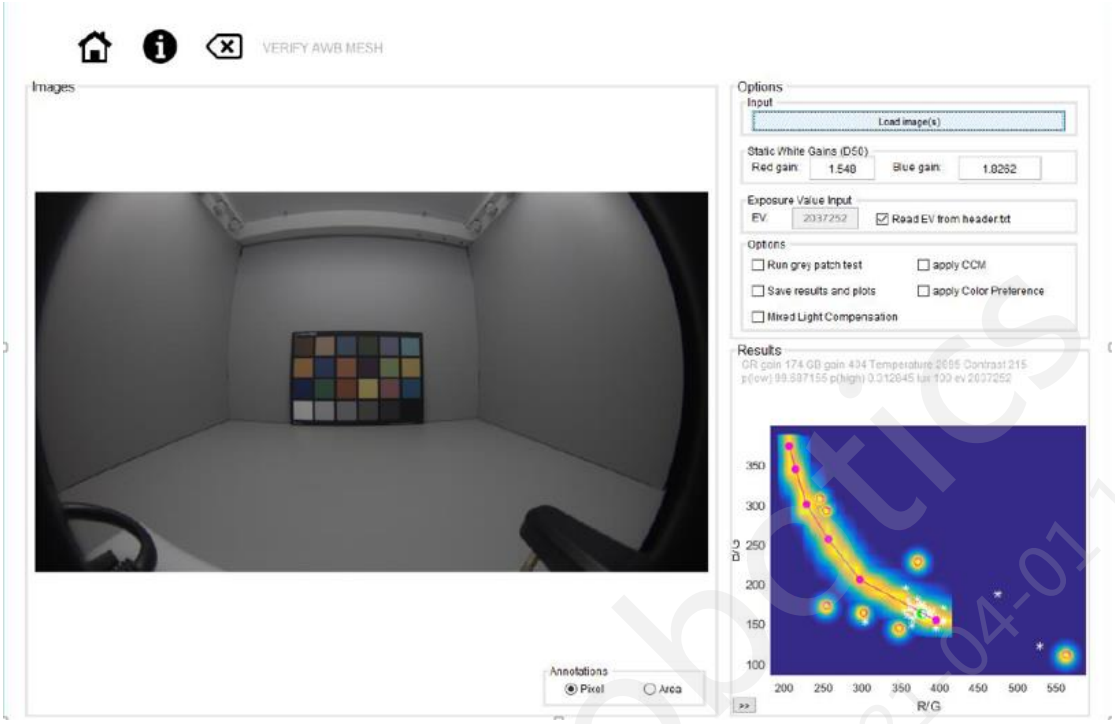


Figure 3.3-19 Verify AWB mesh window after image import.

Notice that the "Results" plot now shows several white, green and pink data points. Here, the white represents the gathered AWB statistics, green shows where the algorithm converged, while the magenta are the illuminant calibration points. It's expected that an image captured under a certain illuminant will converge around the calibration point of that illuminant. For example, an image taken under D50 conditions, should converge around the D50 calibration point.

Note: White and green data points will be visible in the "Results" plot if the multiple image import option was used. This is also the case if multiple images are imported one-by-one. However, the image shown in "Images" will be the last image loaded in.

In the "Options" region, the color temperature and red/blue gains are calculated and displayed. Here, the static white gains can be adjusted if using the verification option to test a different module, using the current ISP calibration.

When working with a particular point, the output results can be used in the table of listed white points. This will accurately set the coordinates or the estimated temperature of the point, to output a more accurate plot. This is useful if the point of conversion is not aligned with the currently available data.

6. Choosing the "Save results and plots" option will save the processed image(s) and the corresponding graphs and data points.
7. Green convergence points are the main indicator of AWB performance. If these data points are not achieving the expected targets, then AWB is failing and should be adjusted for these illuminants. This is done back in the main AWB window (click the AWB icon) before returning

here for verification of the new settings.

3.4 Color Correction Matrix（CCM）

3.4.1 Overview

The CCM module corrects colors and adjusts them based on customer preferences. It changes the chroma values from an image to match those of a standard color space. This is done by capturing the Colorchecker chart and using all patches and their known reference chroma values. Most often the standard colors do not provide optimal imag quality. Depending on the application/customer perferences, these values are changed to better suit the purpose of the product.

The tuning process allows for control of overall hue and saturation and is described bellow:

- During phase one - Tune CCM using .RAW images in Calibration Tool.
- During phase two - Tune CCM in lab conditions for most scenes using .RGB images in Calibration Tool.
- During phase three - Fine-tune CCM parameters to suit all scenes by modulating parameters according to gain

3.4.2 Tuning Theory

This module applies linear color correction to input r, g, b pixel values. The matrix coefficients (found in the hardware registers) are calculated by the firmware according to the color temperature recorded in the AWB module. During CCM tuning, the colors can be saturated/desaturated according to lighting conditions to achieve optimal image quality.

Key firmware parameters in API：

API parameters are in the "API" tab of Control Tool.

Parameter name	Source	Description
Manual_saturation	TSYSTEM	Enable/disable manual saturation
Saturation_target	TSYSTEM	Manual saturation control

Table 3.4-1 CCM API parameters

Key firmware parameters in static calibrations

Static calibration parameters are in the "static_calibrations.c" file, and in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
calibration_mt_absolute_ls_d50_ccm	Calibration values for CCM under high CCT light source

calibration_mt_absolute_ls_d40_ccm	Calibration values for CCM under middle CCT light source
calibration_mt_absolute_ls_a_ccm	Calibration values for CCM under low CCT light source

Table 3.4-2 CCM static calibration parameters

Key firmware parameters in dynamic calibrations

Dynamic calibration parameters are in the "dynamic_calibrations.c" file, and in the"Dynamic Calibrations"-tab of Control Tool.

Parameter name	Description
Saturation_strength	LUT of saturation strength with respect to gain
CCM_one_gain_threshold	Gain threshold for Unity CCM applied

Table 3.4-3 CCM dynamic calibration parameters

An identity matrix is applied to R, G, B pixels if gain exceeds the threshold defined by CCM_ one_gain_threshold. This is used to decrease noise visibility at the expense of color accuracy.

3.4.3 Tuning during phase one

Prerequisites

Prerequisite	Status / value
Black level	Tuned
Lens shading	Tuned
Initial gamma	Tuned
CNR	Disabled
Manual exposure	−0.25 < exposure error < 0.25 (f-stops). This can be verified through Imatest. Avoid clipping of patch 19 from the ColorChecker chart

Table 3.4-4 CCM phase one prerequisites

Tuning procedure

During this initial tuning stage CCM is not expected to be perfect. Using the system information already available it produces a rough estimate. The most important stages for CCM tuning come in phase two and three.

To tune CCM open Calibration Tool and click the CCM icon. CCM window function introduction:

Region	Sub-region name	Function

La*b* Color Error	-	Shows error vector plot and color error values
	>> button	Undock current plots from window – Allows for rotation, zooming, printing or saving the figure
	View	Change perspective of 3D plot
CCM Calibration Settings	Target Hue	Adjust the hue (radial angle from standard targets)
	Target Saturation	Adjust the saturation (radial distance from center)
	Gain	Set or view gain value. This corrects the exposure of the calibration image.
CCM Calibration Settings Cont.	Optimize	Automatically set gain value
	Color Error	Select the color metric used to calculate error
	Square Error	If enabled, values refer to the square of the error. These are the minimized values during CCM calculation
	ISP Gamma	Set to the gamma curve used previously
	Target Colorspace	Colorspace in which La*b* color error is evaluated
	Patch Weight Selection	Table of weight values applied to each ColorChecker patch (in order). These can be manually adjusted
	Reset Patch Weights	Resets patch weights to default values
Load Calibration Image	Color Temperature	Select the current color temperature to tune
	Load RAW Image	Load an input .RAW image for CCM calibration
	Load RGB Image	Load an input .RGB image for CCM calibration
	Load Lab Target	Load CCM target values from a .txt file
	Reset Lab Target	Reset the loaded target values to default
ColorChecker	-	Image of the calibrated ColorChecker
Results	CCM Chroma Correction	Displays amount of chroma correction applied
	CCM Saturation Correction	Displays amount of saturation correction applied

	CCM weight penalty	Displays amount of weight penalty applied
	[High/Medium/Low] Temperature CCM	Table showing calculated CCM
	Re-Calculate CCM	Recalculate CCM based on current settings

Table 3.4-5 CCM window function description

1. Calibration images：

   Capture 3 images of a ColorChecker chart using illuminants of three color temperatures, high, medium and low. High can be D65 or D50, medium can be D40, CWF or TL84 and low can be A or Halogen.

   When photographing the ColorChecker, make sure it is framed correctly with uniform illumination (see Figure 3.4-1). This can be checked with a colorimeter such as the Konica Minolta CI-200.

Figure 3.4-1 Input ColorChecker image

2. Calibration procedure：

   1) Use the “Color Temperature” radio buttons to select a color temperature (e.g. High).

   2) Load the captured ColorChecker calibration image for the chosen color temperature (Load RAW image or Load RGB image buttons).

   3) The Colorchecker patch selection window will open.Select the patch location and click save to return to the CCM window.

4) Now under "ISP Gamma", select the gamma used during gamma calibration. This could be a standard formula such as Rec. 709 or a custom LUT you specified in the Gamma & Tone module.

5) The plot will show the error in each patch in Lab color space, between ideal (circular data point) and corrected (square data point). Fine-tune this result using the following parameters:

   A. Patch Weight - Manually adjust the weighting of each patch using the "Patch Weight Selection" grid, with reference to the numbered data points in the plot. Grid coordinates correspond to the ColorChecker patches, for example, (1, 4) represents patch 19 (white). You may want to prioritise skin tone accuracy (patches 1 and 2) or primaries (13-15).

   B. Target Saturation - Adjust this value to alter saturation (as described in Table 2.2-1). This parameter will have the most impact on the output "Mean chroma" value produced in the "La*b* Color Error" plot.

   C. Target Hue - Adjust this value to alter hue (as described in Table 2.2-1). This should be carefully considered as altering hue could cause noticeable unwanted shifts in color if not required. Normally, this parameter only requires adjustment if most patches are suffering from a hue shift.

   D. Gain - Either select the "Optimize" option to automatically calculate the gain value (recommended), or manually set a value.

   E. Output CCM - All the above parameters are used to help calculate the CCM. In the "Results" region the user can find "CCM Chroma Correction" and "CCM Saturation Correction" values (with sliders). Note that these are automatically generated by setting during CCM calculation. Hence, they are rarely used for actual tuning, instead they act more as indicators of the overall processing.

6) Examining all the information provided in the "La*b* Color Error" region will help evaluate the results. Also try changing the "View" option or ">>" undock button to explore the plot interactively.

7) Repeat steps 1 to 6 for the other two color temperatures. Bear in mind that at low temperatures, "Mean Chroma (Saturation)", seen in the "La*b* Color Error", is usually lower than other temperatures, and the results tend to be less well behaved.

8) Use the save button to store your calibration and return to the home window.

3.4.4 Tuning during CCM phase two

Prerequisites

Prerequisite	Status / value

Phase one prerequisites	Complete
AWB	Tuned
gamma	Disabled
CCM	Disabled
AutoLevel	Disabled
AE	−0.25 < exposure error < 0.25 (f-stops). This can be verified through Imatest.

Table 3.4-6 CCM phase two prerequisites

Tuning process

Using Imatest's "Colorcheck" module, verify exposure error before proceeding to CCM tunning phase two. If exposure error falls out of the range given in Table 3.4-4, re-tune AE_target parameters to adjust exposure. Repeat this process until correct exposure is achieved.

Afterwards tuning is very similar to that of phase one.

3.4.5 Fine tuning demosaic phase three

Phase three is comprised of two main segments, updating calibration and saturation modulation. The first of these focuses on accounting for alterations made in previous modules. The other defines saturation with respect to gain, the idea being that it gives more control over noise at higher gain/lower lux levels.

Note: The trade-off between color accuracy and noise visibility is carefully considered for each gain. Most often, the aim is to provide the highest saturation without revealing unnecessary noise. Remember to tune with respect to either the customer specification or the intended application.

Updating calibration

With the numerous amounts of fine tuning taking place during phase three, it is often the case that modules such as gamma are altered, which CCM is dependent upon. This means retuning using the process described for CCM during phase two is required in order to update the CCM accordingly.

Saturation modulation

1. Set Saturation_strength=0,128 in dynamic_calibrations.c. This sets saturation magnitude at the midpoint (128 out of 256) for all gains equal to 1 (Above 128, increase saturation; below 128, decrease saturation).

2. Starting with the lowest gain values in lux levels of 1,000+, the Saturation_target will likely need to be increased. Normally, while there is minimal interference from noise, saturation is increased in an attempt to improve subjective quality of image features, such as blue sky and grass. A recommended target value is around 120% (where 100% is the saturation value of the authentic ColorChecker patches).

3. Verification of saturation and exposure error should now be conducted through the “Colorcheck” module previously used in Imatest. Mean chroma (saturation) value can be found in the "a*b* color error" window.

4. When these two values meet the criteria, set them in Saturation_strength for all corresponding gains before moving onto the next lighting condition.

5. Now tune the “middle” gain values at 100 lux, while repeating the verification process.Across this range of gains, the Saturation_target is likely to be around a value of 100%.

6. Once middle gains meet the criteria, move onto the highest gain levels using 10 lux and 1 lux conditions. As this is where noise should be the most visible, the Saturation_target is often lowered beneath 100% to around 90%. Keep an eye on chroma noise, attempting not to increase it, but also making sure that CCM passes the IQ requirements.

7. With all gains accounted for, capture a ColorChecker chart image at each gains, running them through Imatest for verification to ensure CCM matches the customer specification.

Additional tuning information

Shown below is an example of a fully calibrated saturation LUT and unity CCM gain setting:

```
Saturation_strength={
                        {0,136},
                        {3*256,128},
                        {4*256,115},
                        {6*256,105},
                        {7*256,80},
                        {8*256,128}
                    };
CCM_one_gain_threshold = {7*256};
```

In this example, till gain 7, saturation is modulated by gains. Once gain reaches 7, identity matrix will be applied to CCM. Remember to set gain 8 saturation target to 128 in modulation LUT, otherwise identity matrtic will be desaturated too.

3.5 Auto Focus（AF）and ZOOM

3.5.1 Overview

AF does not always require tuning (dependent upon the application of the imaging system). For example, security cameras are often manually focused, where there is no need for an AF function, unlike in a device such as a smartphone. In the situation that AF does not need tuning then focus should be manually adjusted for all images captured during and after tuning. For applications that require the AF module, tuning is done at the earliest convenience in phase two. This is not a necessity

as AF is independent of other modules, but will save time focusing while tuning other modules. AF during each of the phases consists of：

- *Phase 1* - AF is **not** tuned, but manually adjusted for all images captured. Note that for this purpose, the lens driver should be fully working and tested.
- *Phase 2* - AF tuned to suit all scenes at earliest convenience.
- *Phase 3* - Previously tuned AF used throughout and verification of accuracy, speed, repeatability, and stability.

3.5.2 Tuning theory

Alteration of AF is done through Control Tool and is generally a fairly simple process. The intention of AF tuning is to define the lens positions (LPs) for the system's far end, infinity, macro and near end focal points. For this, there are very few parameters that needs to be set.

Key firmware parameters in API

API parameters are found on the relevant source page listed in the "API" tab of Control Tool

Parameter name	Source	Description
AF_mode_id	TALGORITHMS	Select AF mode(auto/manual)
AF_manual_control_id	TALGORITHMS	Manually set focal point
AF_roi_id	TALGORITHMS	ROI for gathering focal data

Table 3.5-1 AF API Parameter

Key firmware parameters for dynamic calibrations

Dynamic calibration parameters are found both in the "dynamic_calibrations.c" file, and on the relevant page listed in the "Dynamic Calibrations" tab of Control Tool:

Parameter name	Description
AF_lms	LUT of AF lens positive values

Table 3.5-2 AF dynamic parameter

3.5.3 AF tuning during phase one

As previously mentioned, AF is technically not used during phase one. Instead using the options provided by AF_mode_id (as seen in Figure 3.5-1), select "AF_manual". From here the lens position can be manually adjusted through AE_manual_control_id using either the slider, or inputting a specific value. This process needs to be done for all images captured during phase one, regardless of module

Figure 3.5-1 AF_MODE_ID selection

3.5.4 AF tuning during phase two

Tuning procedure – Basic AF

AF Basic tuning should start from horizontal orientation of camera.

1. Illuminate a resolution chart (ISO12233 or slanted edge chart are suitable) at 1,000lux. The distance between the chart and the camera system should represent infinity (over 2 metres).

2. Now change the AF_mode_id to "AF_calibration", this will output the lens position andcalculated image sharpness in the command window (see Figure 3.5-2).

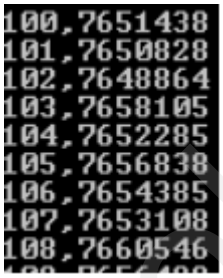


Figure 3.5-2 Lens position（left）and sharpness（right）

3. Copy these values into a spreadsheet such as Microsoft Excel. Then use them as coordinates to output the focus response graph for infinity (see Figure 3.5-3)

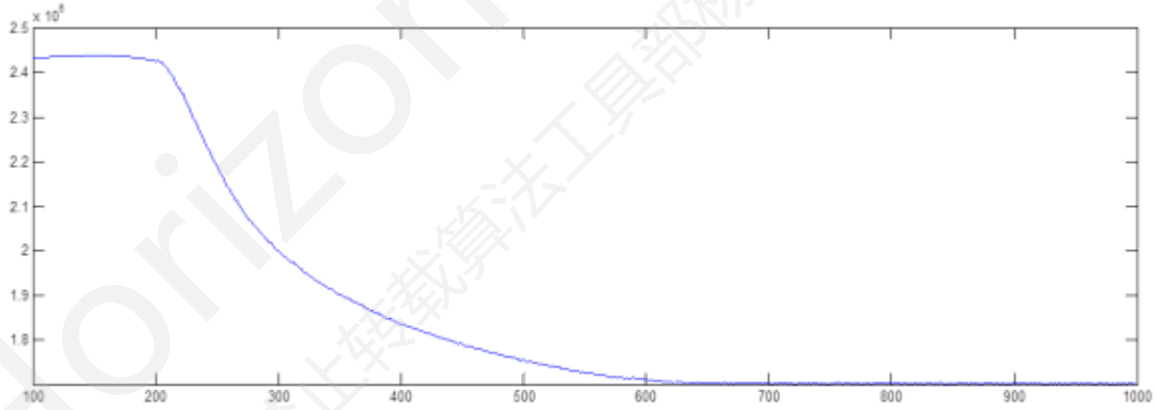


Figure 3.5-3 Focal response curve for infinity.

4. The sharpness before 190 LP is quite stable, but the AF algorithm ideally needs to see an obvious decrease when moving the lens. This means 190 should be used as the lens position for infinity.

5. Repeat the process a couple more times to check the accuracy of the LP value. A tolerance of around 1% difference is acceptable between runs. If results are in this range accurate value, then set this as the lens position for infinity in AF_lms.

6. To acquire a lens position for the far end focal point, reduce the value obtained for infinity by

15% (multiply the LP of infinity by 0.85). Enter this into the calibration LUT.

7. For the macro focal point, set a small resolution chart 10cm from the imaging system in similar 1000 lux conditions. Run the calibration as before to output the focal response curve.

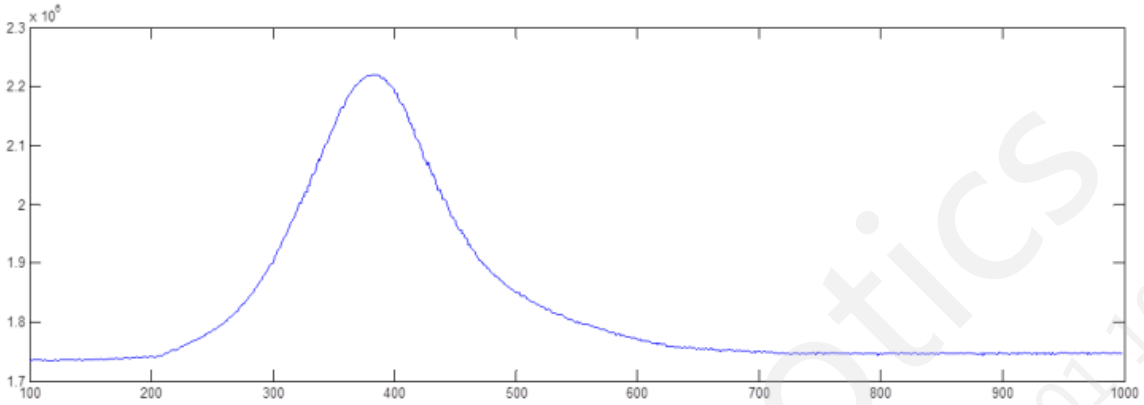


Figure 3.5-4 Focal response curve for macro.

8. Here, the LP is taken at the maximum sharpness value and input into AF_lms as the macro value. Around 384 in the example given. Although again, it's worth repeating a couple more times for accuracy.

9. Derive the near end focal point can by increasing the value achieved for macro by 10% (multiplying the LP of macro by 1.1). Then input this into the calibration LUT.

Repeat these operations for upward and downward orientations, then fill the AF calibration LUT in dynamic calibration c file as the format below.

Parameter name	Description
Down_FarEnd	Downward orientation FarEnd lens position
Horizontal_FarEnd	Horizontal Orientation FarEnd lens position
Up_FarEnd	Upward Orientation FarEnd lens position
Down_Infinity	Downward orientation Infinity lens position
Horizontal_Infinity	Horizontal orientation inifinity lens position
Up_Infinity	Upward orientation inifinity lens position
Down_Macro	Downward orientation macro lens position
Horizontal_Macro	Horizontal orientation macro lens position
Up_Macro	Upward orientation macro lens position
Down_NearEnd	Downward orientation NearEnd lens position
Horizontal_NearEnd	Horizontal orientation NearEnd lens position

Up_NearEnd	Upward orientation NearEnd lens position

Table 3.5-3 AF AF basic calibration parameters.

There are some other extend calibration parameters to control AF behavior listed below.

Parameter name	Description
Step_Number	The total steps from FarEnd to NearEnd
Skip_frames-in	Number of frames be skipped before AF start
Skip_frames_moves	Number of frames be skipped before each AF iteration
Dynamic_range_th	Focus Value Contrast threshold
Spot_tolerance	Focus value dynamic range minimum threshold of zones which used to filter out unqualified ROI in multi-spot
Exit_th	Focus Value downhill threshold
Caf_trigger_th	CAF scene change trigger threshold
Caf_stable_th	CAF scene change stable threshold

Table 3.5-4 AF extend calibration parameters.

In control tool, scene change factor value will be output in API, the Caf_trigger_th should be tuned to detect real changed scene which means the variance of scene change factor value of real scene change is bigger than Caf_trigger_th set in AF calibration lut. Furthermore, Caf_stable_th should also be tuned to detect stable scene which means the variance of scene change factor value is lower than Caf_stable_th set in AF calibration lut. CAF will be re-trigger once the is higher than Caf_trigger_th and then lower than Caf_stable_th.

3.5.5 AF tuning during phase three

With AF tuning completed in phase two, there is rarely any need to revisit the module. In general, AF can now be used to focus all images before capture. This is simply done by selecting either of the two following options in AF_mode_id:

- ✧ AF_auto_single- Each time the scene needs to be focused set the focus to AF_manual, then select the AF_auto_single option. The camera will now attempt to focus on the scene and will leave the VCM at the final lens position. To refocus, or focus on another scene, this process needs to be repeated.
- ✧ AF_auto_continuous- Selecting the AF_auto_continuous mode will tell the system toconstantly refresh the focus data. This will attempt to continuously look for a better focal point. While this does attempt to perpetually provide the best focus result, it should be noted that this will use more processing power than AF_auto_single.

3.5.6 AF +ZOOM tuning

Some lens includeing optical ZOOM function, that need consider ZOOM and auto focusing function to get good focusing position. ZOOM is amplify or shrink image depan independing of other modules. And ZOOM tuning also is independent, just need set parameters in ZOOM_MANUAL_CONTROL_ID in TALGORITHM whichh is used to set ZOOM ratio as below figure, 140 means 1.4x ratio.

Figure 3.5-5 ZOOM ratio

First step，Confirm ralation between ZOOM ratio and motor .Take examples as below table. (Realrelation should refer to device)

ZOOM_ratio	ZOOM_step
1	0
1.1	180
1.2	361
1.3	541
1.4	722
1.5	902
1.6	1082
1.7	1263
1.8	1443
1.9	1624
2.0	1804

Table 3.5-5 ZOOM ratio and step corresponding table

Fill ZOOM step in Dynamic area of control tool, result as below figure:

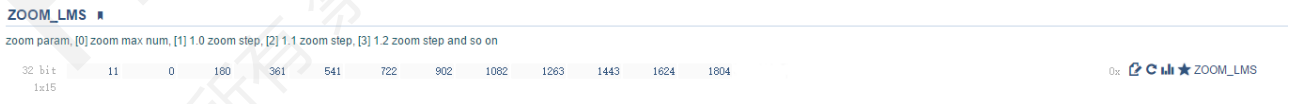


Figure 3.5-6 ZOOM_LMS step filling

First postion in ZOOM_LMS is total num of ZOOM ratio.

Second step, set ratio to 1.0 means ZOOM_MANUAL_CONTROL_ID set 10. Make auto focusing calibration after confirming ZOOM ratio is 1.0 to get AF calibraton parameter basing this ZOOM ratio. Modify other ZOOM ration and repeat above operation to get AF calibraton parameter basing

current ZOOM ratio. Repeat above operation throught all ZOOM ratio value to get all AF parameter corresponding ZOOM ratio. Fill all ZOOM ratio and AF parameter in ZOOM_AF_LMS LUT of Dyanmic.

That finished AF calibration basing on all ZOOM ratio. Filling result as below figure:

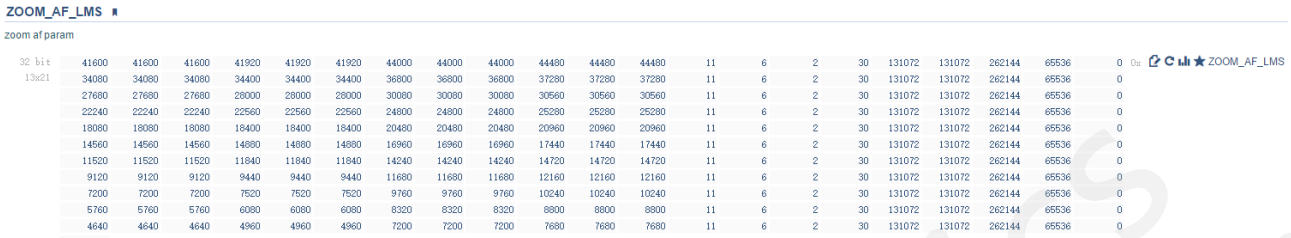


Figure 3.5-7 ZOOM_AF_LMS filling data

3.6 Black Level

3.6.1 Overview

The black level modules handles the offset which is applied by the image sensor. You should tune this module first because all other modules rely on accurate calibration of black level.

Tuning procedure is described below:

- During phase 1 - Tune Black level in lab conditions for a range of analog gain values using Calibration Tool.
- During phase 2 - No tuning required, unless you notice poor performance of previous tuning.
- During phase 3 - No tuning required, unless you notice poor performance of previous tuning.

3.6.2 Tuning theory

Image sensor black level varies according to temperature, analogue gain and to a lesser extent, integration time. Most sensor manufacturers do not provide information about temperature. So in ISP black level is only modulated by analog gain.

Key hardware registers

Hardware registers are in the “Hardware” tab of Control Tool.

Register name	Source	Description
Black_00	Offset & Sensor Offset Pre Shading	Red channel offset
Black_01	Offset & Sensor Offset Pre Shading	Green (red row) channel offset
Black_10	Offset & Sensor Offset Pre Shading	Green (blue row) channel offset

Black_11	Offset & Sensor Offset Pre Shading	Blue channel offset

Table 3.6-1 Black level hardware registers

Key firmware parameters for static calibrations

Static calibration parameters are in the "static_calibrations.c" file, and in the "Static Calibrations" tab of Control Tool

Parameter name	Description
Black_level_gr	Black level of Green (red row) channel
Black_level_gb	Black level of Green (blue row) channel
Black_level_b	Black level of blue channel
Black_level_r	Black level of red channel

Table 3.6-2 Black level static calibration parameters

3.6.3 Black level Phase One Tuning

3.6.3.1 Capturing calibration images

To calibrate black level, capture a set of raw black images at different analog gains, which will be loaded into the Calibration Tool for black level calibration:

1. Cover the sensor and lens assembly using black/opaque material. Reduce all ambient light as much as possible, to ensure no light reaches the sensor.
2. Launch the Control Tool and load the .xml and command.json files for your software version.
3. Select the "API" tab in the top left corner of the Control Tool. From the list provided, select the "TSYSTEM" page:

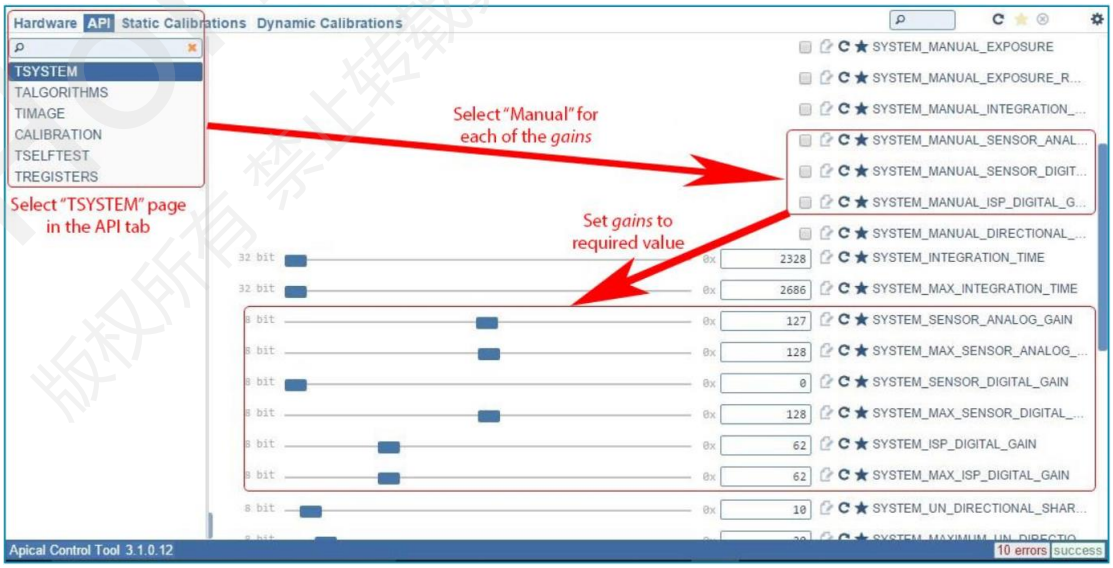


Figure 3.6-1 system parameters

4. Set all gain parameters to manual control using the tick-boxes highlighted in Figure 3.6-1.
5. Set all gain parameters to 0 and capture the first black image.
6. In steps of 32, increase the value of "System_sensor_analog_gain", and capture an image each time. Continue until you reach the maximum analog gain. Values of up to 255 are supported by the Control Tool but may not be supported by your sensor.

3.6.3.2 Black level main windows

When you open the black level module, the window shown in Figure 3.6-2 pop up.

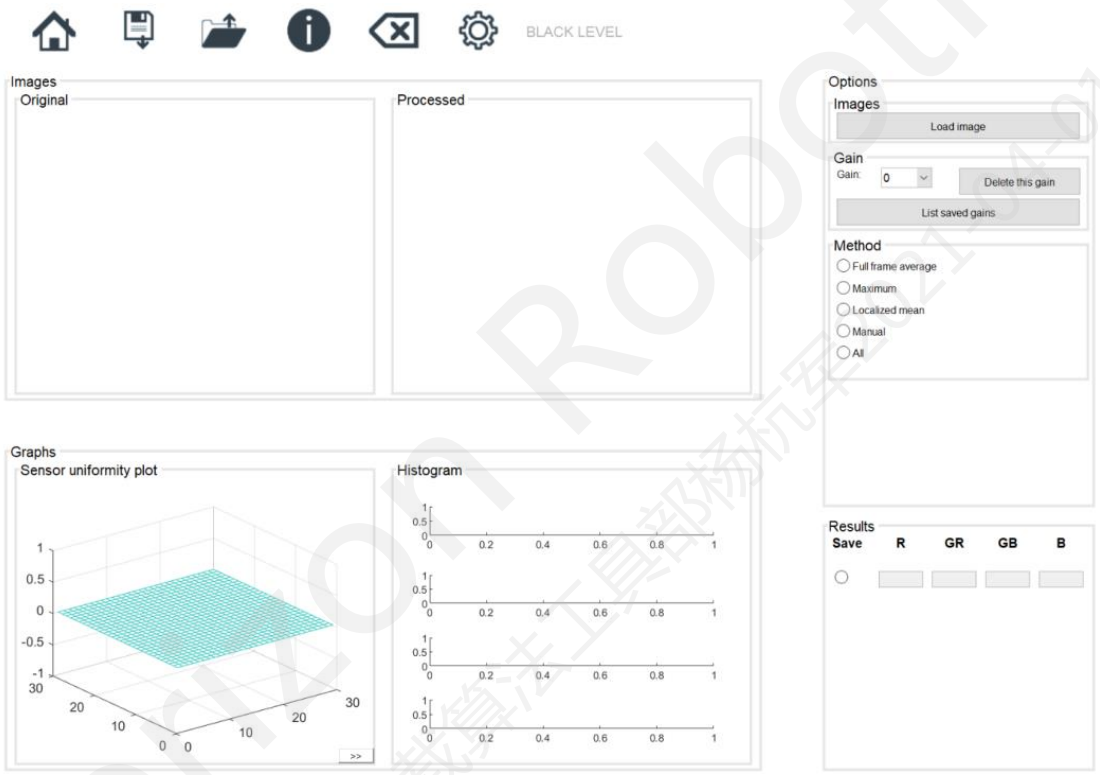


Figure 3.6-2 Black level window

Table 3.6-3 lists the function of each region.

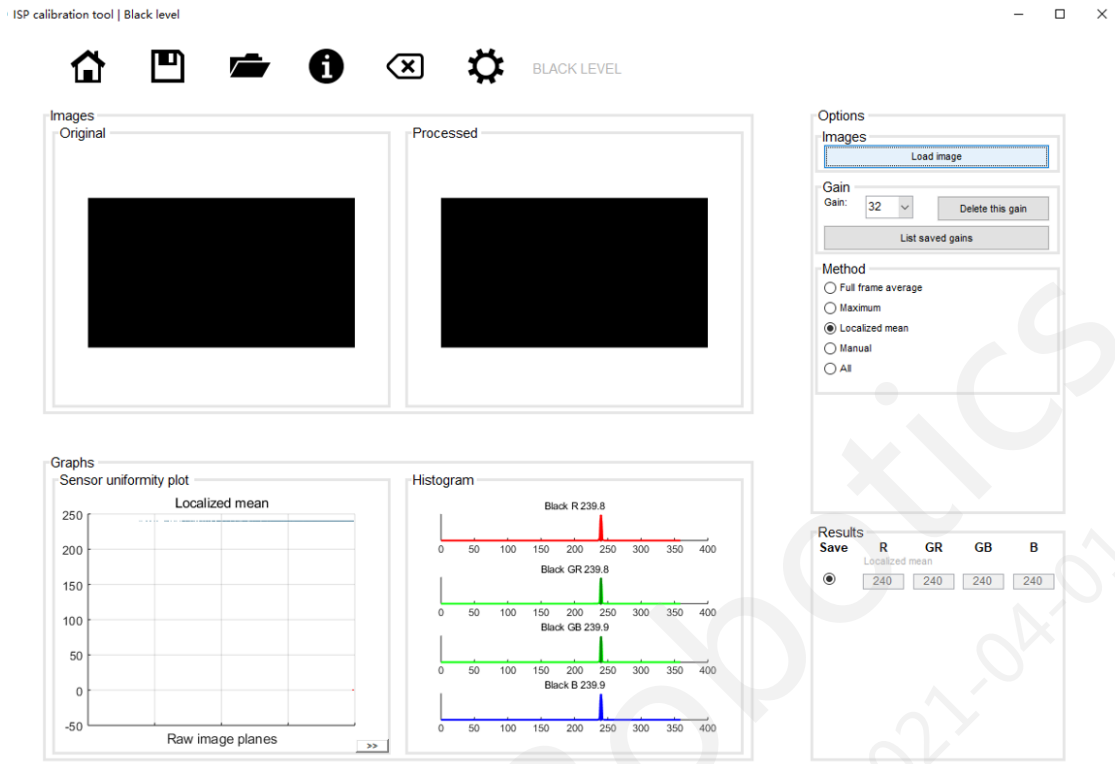
Region	Sub-region name	Function
Images	Original	Displays the captured image
	Processed	Displays the corrected image
Graphs	Original flat field	Displays 3D uniformity plot of the original image

	Corrected flat field	Displays 3D uniformity plot of the corrected image
	>>button	Undock current plots from window – Allows for rotation, zooming, printing or saving the figure
Options	Imge-Load image	Load an image for the current gain value
	Gain-Gain	Set the analog gain value used during image capture
	Gain-Delete this gain	Delete black level data for the current gain value
	Gain-List saved gain	Lists black level data for all gains
	Meghod	Choose black level calculation algorithm
Results	-	Shows calculated black level value for each channel

Table 3.6-3 List of all regions and their functions in the black level window

3.6.3.3 Start tuning

Set the gain level one the calibration interface, load the corresponding RAW.



3.6.3.4 Method

Full Frame Average – Average of each color channel over the whole image.

Maximum – The maximum of the channel-by-channel means of 5 regions (centre and corners).

Localized Mean – Channel-by-channel mean of 5 regions (centre and corners).

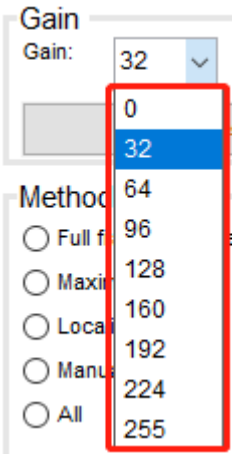
Manual – If black level values are known, the user can enter these manually. No .RAW images are needed for this method.

All – If "All" is selected, all 4 of the above methods are shown in the results window for comparison, and you can choose which one to use.

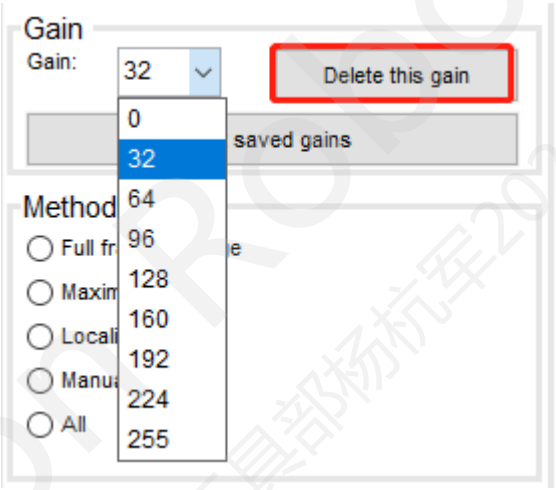
Note: We recommend Localized Mean, but If the variation across the sensor is large, Maximum is best. It will reduce color casts in darker regions in low light, at the expense of detail.

3.6.3.5 Gain

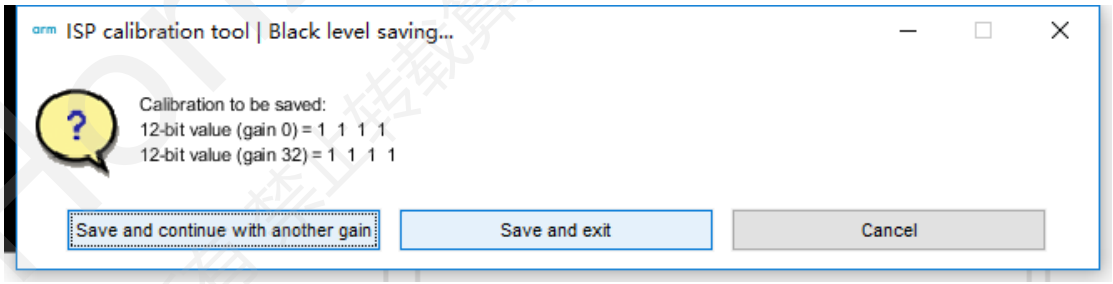
Calibration for each analog gain, 0 corresponds to 1xGain, 32 corresponds to 2xGain,and so on.



If you import the wrong RAW, you can delete the current data.



3.6.3.6 Save the black level tuning result



3.6.3.7 Additional tuning information

Providing black level is tuned correctly, then there should be no reason to revisit the module in the second and third phases of tuning. If a fault with black level is noticed at this stage, then the whole system will require verification/retuning after adjusting the black level.

In the whole pipeline, there are several blocks with registers related to black level, which need to be set properly.

Additional hardware registers related to black level list below:

Register name	Source	Description
Black level(l/m/s/vs)	Frame_stitching	12-bit value
Offset	Digital_gain	20bit value
Black_level_in	Sqrt	20bit value
Black_level_out	Sqrt	16bit value
Black_level_in	Square_be	16bit value
Black_level_out	Square_be	20bit value

Table 3.6-4 Black level related hardware registers

3.7 Green Equalization（GE）

3.7.1 Overview

This module corrects imbalances between pixels in the green/red (Gr) and green/blue (Gb) color planes by adjusting the planes to have equal sensitivity. Correcting for this reduces the appearance of checker patters after demosaic and can also improve false color correction in the demosaic algorithm. Figure 3.7-2 demonstrates how this pixel imbalance can be identified by plotting pixel response, with respect to color plane, against their position on the sensor.

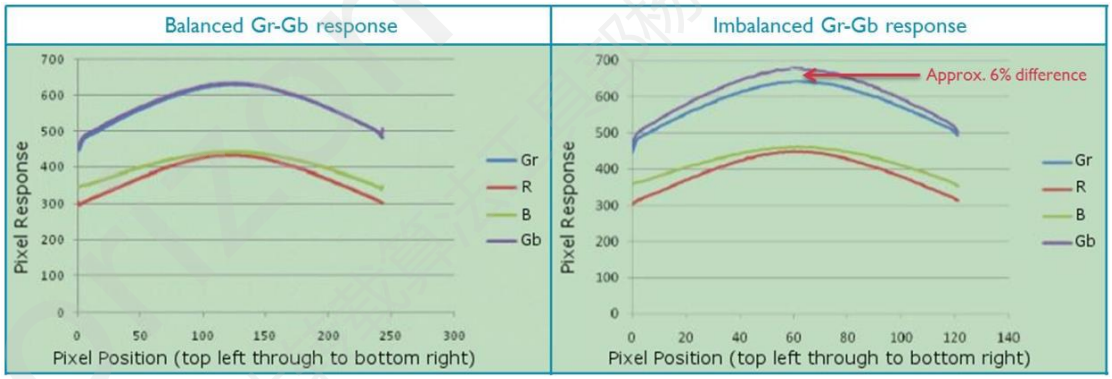


Figure 3.7-1 Pixel responses showing balanced Gr-Gb values and imbalanced Gr-Gb values

Most modern sensors have very little green imbalance, meaning default green equalization values are normally acceptable. However, tuning of GE is likely to be required for older/lower quality sensors. Either way GE should still be verified, whereby the process is outlined below:

- During phase 1 - No tuning of GE required.
- During phase 2 - GE verified before tuning “demosaic”.
- During phase 3 - No tuning of GE required.

3.7.2 Tuning Theory

Key hardware registers

Hardware registers are found on the relevant source page listed in the "Hardware" tab of Control Tool.

Register name	Source	Description
GE_enable	RAW Frontend	Enable/disable green equalization
GE_strength	RAW Frontend	Green equalization strength control
GE_threshold	RAW Frontend	Required Gr/Gb difference value for GE to take place
GE_slope	RAW Frontend	Gr/Gb blending for values above the threshold
GE_sens	RAW Frontend	Green equalization sensitivity control
Debug_select	RAW Frontend	Enable/disable GE blending mask

Table 3.7-1 GE hardware registers

Green pixels are compared with the average of the neighboring Gr and Gb values. If the difference is smaller than the GE_threshold, Gr and Gb pixels are replaced with a blended value based on the amount of difference and their raw pixel values. Increasing either GE_threshold or GE_slope will result in more aggressive correction. Figure 30 shows the basis of how these two parameters are implemented in the blending function with respect to the difference in Gr-Gb channels.

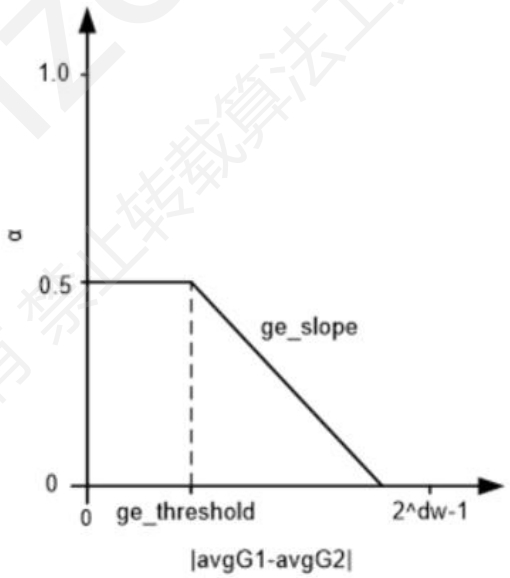


Figure 3.7-2 Illustration of GE parameters

As correction of the greens should not take place on edges as additional GE_sens parameter is

used to adjust sensitivity of the green equalization to detected edges. A higher GE_sens value will result in more aggressive green equalization on edges.

3.7.3 Tuning GE during phase two

As green equalization is not used in phase one, tuning starts just before the "demosaic" module in phase two. This process requires the prerequisites listed in Table 3.7-3 to be complete.

Prerequisite	Status / value
Black level	Tuned
Noise proflie	Derived

Table 3.7-2 GE prerequisites

Set the values shown in Table 3.7-3 below:

Parameter	Value
GE_enable	0
GE_strength	64
GE_threshold	170
GE_slope	285
GE_sens	128

Table 3.7-3 GE default values

If the green imbalance is a problem, the increase GE_strength parameter. Record the values set either in the .json file or in custom_initialization(void) function. There is no need to revisit GE during phase three unless it is seen as a potential source of error.

3.8 Dynamic Defective Pixel Correction（DPC）

3.8.1 Overview

There are two main methods of DPC, being static and dynamic, which can be used to identify and correct defective pixels in the sensor through interpolation from near neighbors. While both are available for use, static is currently used least as its reliability seems to fluctuate with analogue gain, whereas dynamic is much more flexible.

Defect pixels can be classed as one of the following:

- *Hot pixel* - Where the pixel is permanently bright.
- *Dead pixel* - Where the pixel is permanently dark.
- *Weak pixel* - Where the pixel does not provide a linear response.

While ideally DPC would account for all of these, it's not feasible. As such, dead pixels have

minimal impact on image quality while weak pixels are difficult to predict, meaning DPC focuses on accounting for the most noticeable, hot pixels. On top of this, there are further complications dependent upon the proximity of one hot pixel to another, whereby they can be classified into one of the following groupings:

- *Single hot pixel* - One hot pixel where all locally surrounding pixels are correct.
- *Double/copied hot pixel* - Two closely located hot pixels from the same color plane.
- *Triple hot pixel* - Three closely located hot pixels from the same color plane.
- *Cluster of hot pixels* - Four or more closely located hot pixels from the same color plane.

As for the tuning process itself, it is advised that if enabling static DPC, this should be checked before tuning dynamic DPC. In theory this should remove pixels without causing residual error in situations where dynamic correction would, before then allowing dynamic correction to fix those that static cannot. Tuning primarily takes place in the second phase where:

- *During phase one* - No tuning required.
- *During phase two* - Static DPC checked, if needed then dynamic DPC is tuned in lab conditions for all scenes.
- *During phase three* - No tuning required, unless previous tuning or other modules which dynamic DPC is dependent upon causes poor performance.

3.8.2 Tuning Theory

Dynamic DPC is capable of correcting up to two defect pixels from the same color plane. Bigger cluster of defect pixels, more than 2 per colour plane in adjacent bayer quadrants, will not be correctable or if corrected a residual of the defect pixels will be visible.

Analysis shows that starting with double hot pixels on a flat-field area, the residual error linearly increases with a growing magnitude of defect pixels. This error is caused from the averaging of neighbourhood pixels, the more defective pixels in this neighbourhood, the greater the influence they have on the derived output value. Due to this, if a system is seen to contain clusters of hot pixels, the sensor manufacturer should be contacted as this is not acceptable. The process described here focuses on the tuning of dynamic DPC only.

Key hardware registers

Hardware registers are found on the relevant source page listed in the “Hardware” tab of Control *Tool*.

Register name	Source	Description
DP_enable	RAW Frontend	Enable/disable dynamic DPC
DP_threshold	RAW Frontend	Controls strength of dynamic DPC
DP_slope	RAW Frontend	Alpha blend between original and corrected images
DPdev_threshold	RAW Frontend	False color strength on edges
Line_threshold	RAW Frontend	Controls directional nature of replacement algorithm
Sigma_in	RAW Frontend	Manually override noise estimation

DP_blend	RAW Frontend	Alpha blend between directional and non-directional replacement values
Show_dynamic_ defect_pixel	RAW Frontend	Display pixels removed by dynamic DPC

Table 3.8-1 Dynamic DPC hardware registers

Key firmware parameters in dynamic calibrations

Dynamic calibration parameters are found both in the "dynamic_calibrations.c" file, and on the relevant page listed in the "Dynamic Calibrations" tab of Control Tool.

Parameter name	Description
Dp_slope (LUT)	LUT of Dp_slope with respect to gain
Dp_threshold (LUT)	LUT of Dp_threshold with respect to gain

Table 3.8-2 Dynamic DPC dynamic calibration parameters

Implementation of DP_threshold and DP_slope in DPC algorithms can be seen in Figure 3.8-1. As temperature changes alongside aging sensors can introduce defect pixels, always enabling dynamic DPC is recommended.

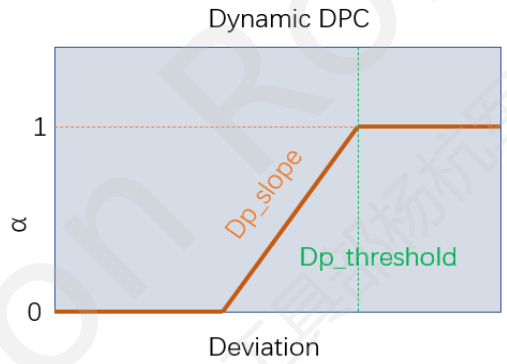


Figure 3.8-1 Illustration of DPC parameter implementation.

With reference to Figure 3.8-1: *pixel output = (1 - α) × pixel input + (α) × replacement pixel value*.

3.8.3 Tuning DPC during phase two

As dynamic DPC is not required in phase one, tuning starts during phase two. This can only be done after completing the prerequisites listed in Table 3.8-23.

Prerequisite	Status / value
Black level	Tuned
DP_threshold	4095 (maximum)
DP_slope	170
DPdev_threshold	2048
DP_blend	0
Line_threshold	0
Sigma_in	0

Analogue and digital gain	0 (minimum)

Table 3.8-3 DPC phase two prerequisites.

Tuning procedure

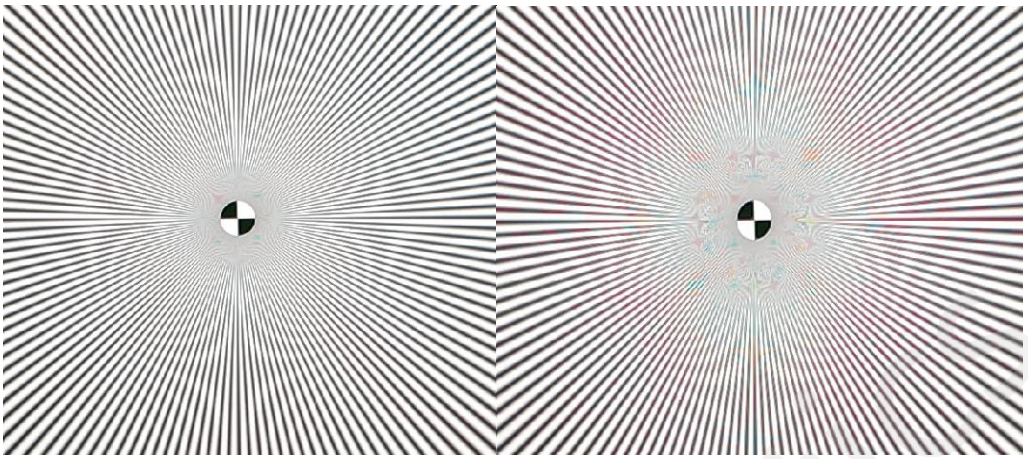
1）Point the imaging system at a studio scene, similar to that in Figure 3.8-2 and wait for AE to settle. Now enable Manual_exposure on the "TSYSTEM" page of Control Tool (see "API" tab).

Figure 3.8-2 Example of a studio scene.

2）Now enable dynamic DPC by setting DP_enable to 1. Given that DP_threshold is at the maximum value, all hot pixels should be visible. Decrease this value until hot pixels are removed, taking caution not to introduce artifacts such as aliasing or excessive blurring of high frequencies.

3）If DP_threshold either can't remove the defect pixels, or it has no effect on the image, then increase DP_slope. In increments of 10%, adjust DP_slope, setting DP_threshold to 4095 and repeating step 2 each time, until defect pixels are successfully removed.

Note: Keep in mind that increasing DP_slope too much will result in decreased resolution as shown in Figure 3.8-2. It may also be preferable to freeze firmware before adjusting these parameters. This can be done by enabling "Freeze_firmware" on the "TSYSTEM" page in the "API" tab of Control Tool. Disabling this after parameter adjustment may help subjectively evaluate the impact of parameter adjustment. Hence, enabling/disabling Freeze_firmware throughout tuning of DP_threshold and DP_slope may be useful.



a)Correctly tuned defect pixel chart

b)resolution loss due to increase in dp_slope and decrease in dp_threshold

Figure 3.8-3 DPC tuning examples with respect to DP settings.

3） With DPC at the current gain level meeting the specification, set DP_threshold and DP_slope values. This is done in the respective “DP_threshold (LUT)” and “DP_slope (LUT)” modulation tables provided in “dynamic_calibrations.c”.

3.8.4 Fine tuning DPC during phase three

Fine tuning of DPC parameters simply requires the process described for phase two tuning to be repeated for all remaining gain levels. As a result, all gain levels should have DP_threshold and DP_slope values represented in the respective “DP_threshold (LUT)” and “DP_slope (LUT)” modulation tables.

3.9 Lens Shading Correction（Mesh-based）

3.9.1 Overview

Vignetting (also known as shading), being the gradual intensity fall-off from the center towards the corner of the image, is corrected using the lens shading module. Alongside this, non-uniformity of colors (also known as chroma shading) can be corrected, which is extremely important when accurately reproducing colors in the final image.

Correcting shading artifacts can be a difficult task given that they are module dependent, this is due to the manufacturer’s error tolerance when mounting optics and IR filters.Additionally, different lighting conditions can also affect chroma shading. This happens because the IR-cut filter does not have the exact wavelength cut-off throughout the sensor, instead the spectrum cut-off occurs depending on the angle of incidence and is usually worse in small imaging devices/sensors with high pixel counts fitted with a wide angle lens (such as mobile devices).

An overview of module tuning is provided below:

- **During phase one** - Lens shading correction tuned in lab conditions for the majority of scenes through Calibration Tool.
- **During phase two** - No tuning required, unless previous tuning or other modules which shading correction is dependent upon causes poor performance.
- **During phase three** - No tuning required, unless previous tuning or other modules which shading correction is dependent upon causes poor performance.

3.9.2 Tuning Theory

The ISP shading module supports two different mechanisms to correct for the lens shading artifacts, being "mesh-based" and "radial-based". In this section, focus is on mesh-based shading which can correct spatial or asymmetric distortions because it can map the whole image. This method is the recommended to be used, however, it requires more memory to store all LUTs. Radial shading, assumes that the distortions are symmetric and does a correction from the center to the corner of the image. Radial shading can be used when memory is restricted and lens distortions are fairly symmetrical.

Key hardware registers

Hardware registers are found on the relevant source page in the "Hardware" tab of Control Tool.

Register name	Source	Description
Mesh_scale	Mesh Shading	Defines mesh coefficient
Mesh_strength	Mesh Shading	Controls strength of mesh shading

Table 3.9-1 Mesh-based lens shading Hardware registers

Key firmware parameters for static calibrations

Static calibration parameters are found both in the "static_calibrations.c" file, and on the relevant page listed in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
Shading_ls_d65_[R/G/B]	Shading_ls_d65_[R/G/B] D65 Shading 表
Shading_ls_tl84_[R/G/B]	Shading_ls_tl84_[R/G/B] TL84 Shading 表
Shading_ls_a_[R/G/B]	Shading_ls_a_[R/G/B] A Shading 表

Table 3.9-2 Mesh-based lens shading Static calibration parameters

Note: [R/G/B] indicates the presence of individual tabs/parameters for each red, green and blue color channel.

Key firmware parameters for dynamic calibration：

Dynamic calibration parameters are found both in the “dynamic_calibrations.c” file, and on the relevant page listed in the “Dynamic Calibrations” tab of Control Tool.

Parameter name	Description
Mesh_shading_strength	Modulation table of strength with respect to gain

Table 3.9-3 Mesh-based lens shading Dynamic calibration parameters

The ISP provides three pages, containing individual intensity correction maps. Each map provides a corrective gain for each pixel using bi-linear interpolation from a mesh of points. For each point in the mesh there is a single 8-bit coefficient representing the intensity gain value.

There are four possible types of coefficient, defined by the global Mesh_scale register. All of the coefficients are labeled by a certain mesh scale in the ISP, where each is double the gain of the previous mesh scale, up to 16x, as can be seen in table 40. Mesh scale is determined during calibration using Calibration Tool.

Mesh scale	Format of coefficient	Max gain
0	Unsigned Fraction 1.7	x 2
1	Unsigned Fraction 2.6	x 4
2	Unsigned Fraction 3.5	x 8
3	Unsigned Fraction 4.4	x 16

Table 3.9-4 Mesh scale format.

Note: With increasing gain values, the precision of the gain to be applied will decrease and there is a possibility that banding artifacts could be visible.

3.9.3 Tuning during phase one

Lens shading correction can only be tuned once the black level has been tuned, making this the only prerequisite.

3.9.3.1 Mesh shading window

When you open the mesh shading module, a window like Figure 3.9-1 will pop up.

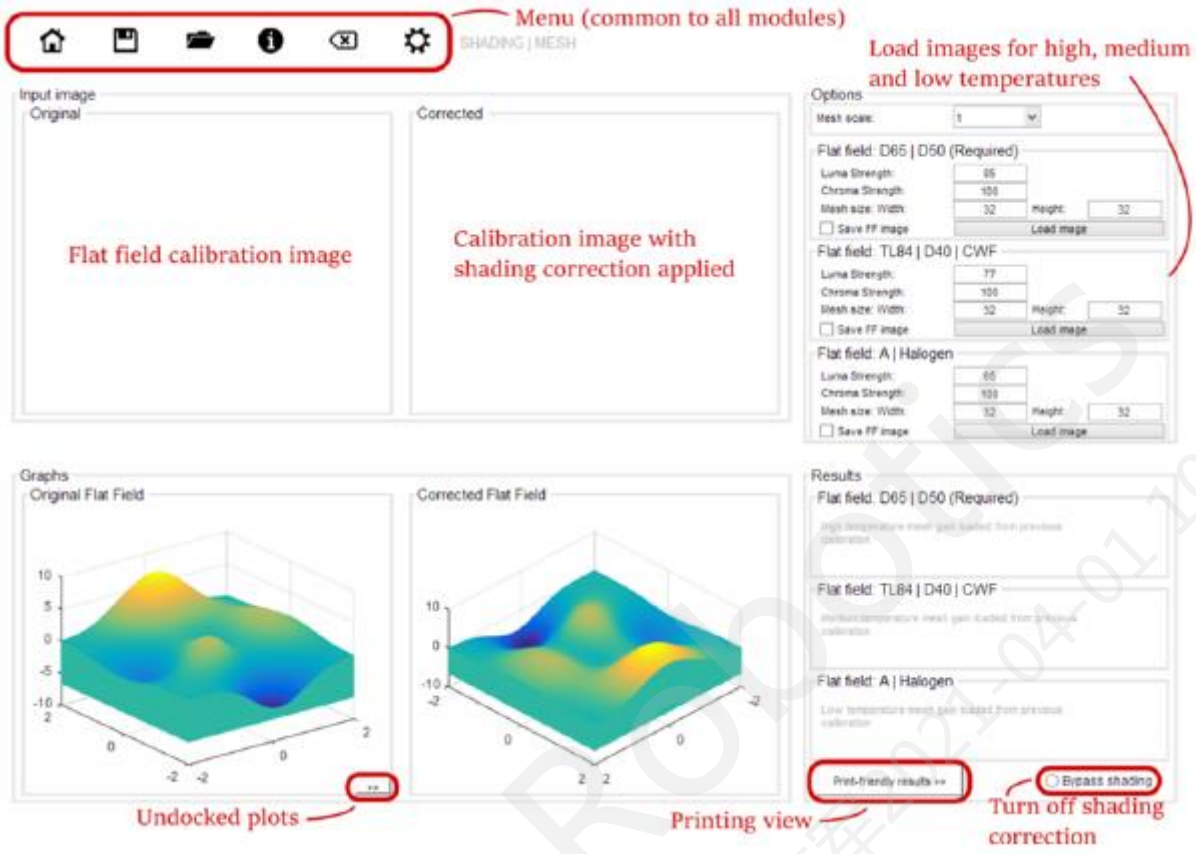


Figure 3.9-1 Mesh shading Window

3.9.3.2 Window features

Table 3.9-5 lists the function of each region.

Region	Sub-region name	Function
Input image	Original	Displays input flat field image
	Corrected	Displays image after mesh shading correction
Graphs	Original Flat Field	Downsampled 3D plot of each color channel before correction
	Corrected Flat Field	Downsampled 3D plot of each color channel after correction
	>> button – Undock plots	Undock current plots from window – Allows for rotation, zooming, printing or saving the figure
Options	Mesh scale	Mesh precision value.
	Mesh size	Set the vertical and horizontal grid size of

		the mesh
	Preview enhancement	Brighten preview images, leave it blank for automatic gain calculation
	Fisheye ROI	Tick to apply elliptical ROI and adjust on X and Y bar
	Luma strength	Set the strength of luma (brightness) shading correction for each illuminant temperature
	Chroma strength	Set the strength of chroma (color) shading correction for each illuminant temperature
	Save FF image	Save corrected flat field image to a file
	Load image	Load flat field image(s)
Results	Flat field: [illuminant names]	Describes the current calibration status for each illuminant temperature
	Print-friendly results	Opens a window with a summary of results for the current illuminant temperature
	Bypass shading	Do not apply mesh shading correction (uses flat values)

Table 3.9-5 List of all regions and their functions in the mesh shading window

3.9.3.3 Calibration images

To calibrate Mesh shading, capture a set of flat field images under illuminants of three different color temperatures.

1. Start by pointing the camera at a light box/panel capable of uniformly emitting multiple color temperatures. Alternatively, place a diffuser in front of the lens to produce uniform illumination. Make sure the lens/sensor are as parallel to the light panel/diffuser as possible.
2. Capture an image for each color temperature - high, medium and low. For example, D65/D40/D30 or D50/CWF/A. The images should be well exposed if possible, with no clipping.
3. If flicker (see Figure 3.9-2) is visible in any of the captured images, the shading calibration will not be correct. We recommend capturing 64 images so that the flicker effects will be averaged. This is usually only necessary for incandescent (A) lighting.

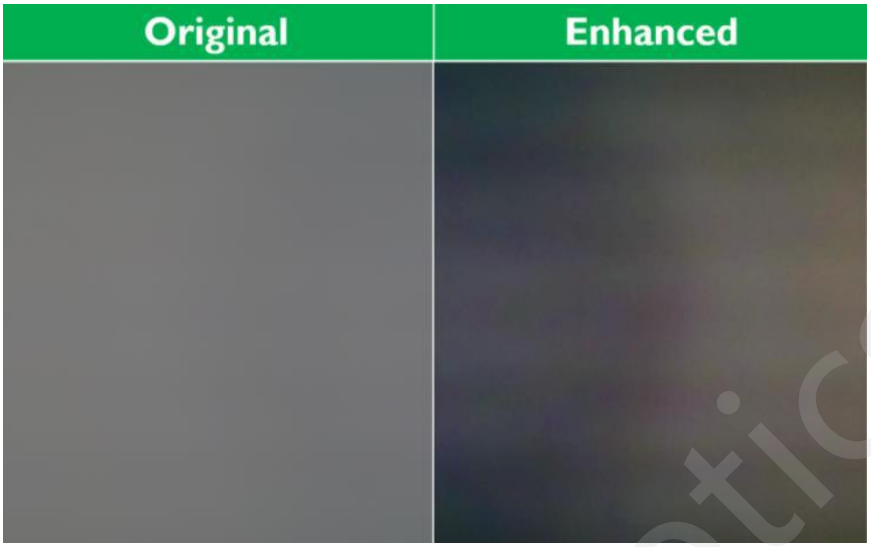


Figure 3.9-2 Example of flicker (contrasting "stripes") on flat field image. Where the left image is the image captured and the right image is the same image, enhanced to help see the banding artefacts.

3.9.3.4 Calibration procedure

1. Open the mesh shading calibration module from the home window of the Calibration Tool.
2. Set the "Mesh size" for each flat field color temperature. This is normally left at the default of 32 x 32.
3. Set "Luma strength" and "Chroma strength" to 100% for all three color temperatures. Luma strength is a multiplier for luma (brightness) shading correction, and Chroma strength is a multipler for color shading correction. Chroma should always be 100% corrected, to avoid color inconsistencies.
4. (Optional) Tick Fisheye ROI if fisheye lens applied, a green circle will display on Original window, adjust X and Y bar after "Load image" to find a correspond region then enter Luma, Chroma strength to see the preview on Corrected window. Notice, depends on Mesh size the edge of ROI might shows over bright blocks, please reduce ROI size as Figure 3.9-3 to eliminate this effect.

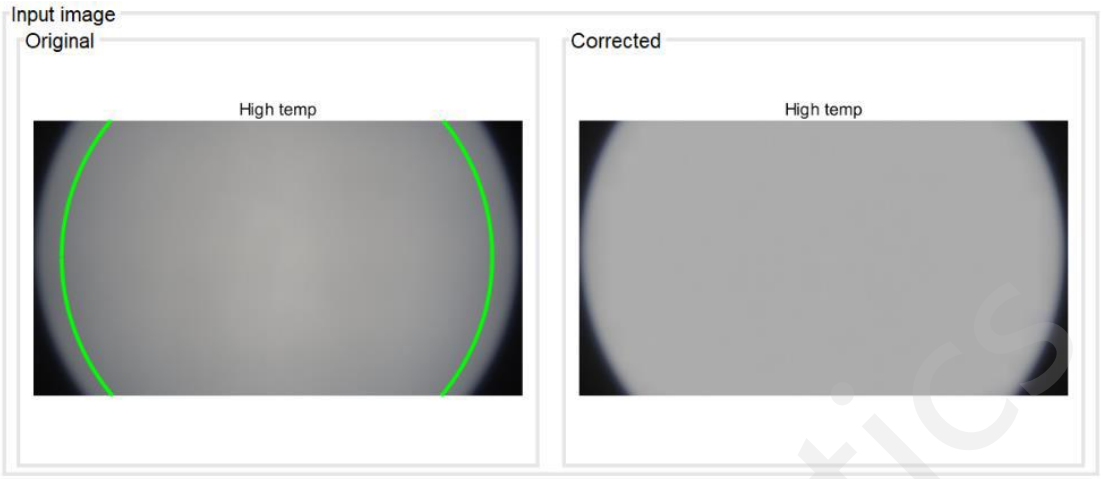


Figure 3.9-3 Example of Fisheye ROI.

5. Click the "Load image" button for high color temperature and select the flat field calibration image(s) for that color temperature.
6. If the highest gain is larger than the highest value allowed by the mesh, you will be prompted to increase mesh scale. You can choose to do this, or decrease luma strength, or accept that some parts of the image will not be completely corrected.

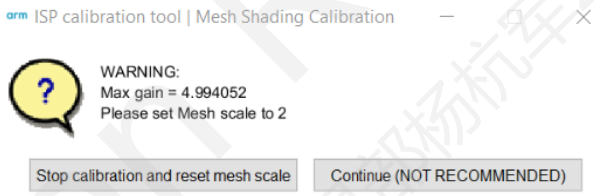


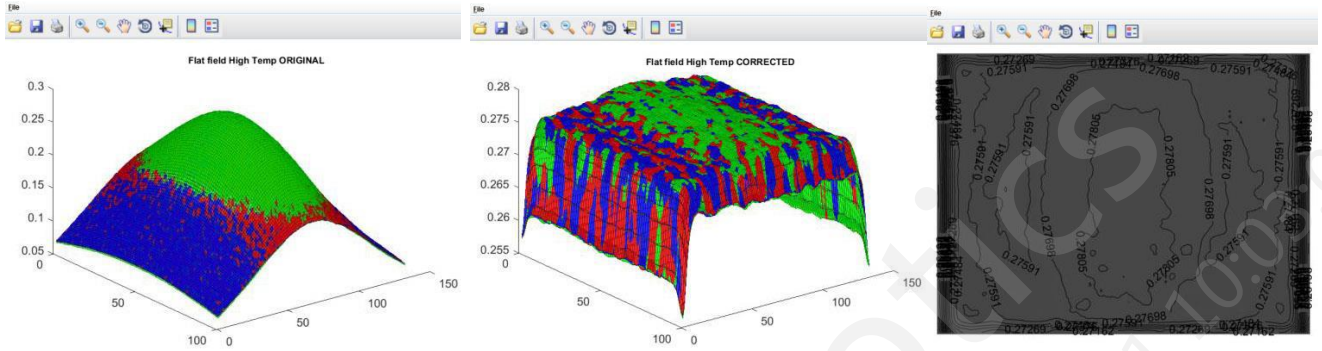
Figure 3.9-4 Max gain warning dialog box.

   a. Decreasing luma strength: This will apply less luminance correction, making corners slightly darker. It also reveals less unwanted noise in the corners.
   b. Increasing mesh scale: this increases the maximum gain that can be applied (described in the table below). Precision of the correction will be reduced, and high gains will result in more noise in the corners of the image.

Mesh scale	Format of coefficient	Max gain
0	Unsigned Fraction 1.7	X2
1	Unsigned Fraction 2.6	X4
2	Unsigned Fraction 3.5	X8
3	Unsigned Fraction 4.4	X16

Table 3.9-6 Mesh scale format

7. (Optional) Use the ">>" (undock) button to view/save plots. This will produce both the original and corrected flat field plots, along with a contour plot of channels (refer to Figure 3.9-5).



（a）Shows 3D plot of the original flat field image （b）Shows 3D plot of the corrected flat field image （c）Shows a contour plot of channels for the corrected image.

Figure 3.9-5 Undocked windows

8. (Optional) Click the "Print-friendly results >>" button to view a summary of all the calibration data and save as a PDF if you want.
9. Repeat steps 5-8 for the medium and low color temperatures, setting "Luma strength" as necessary each time.
10. Click the save icon to store the calibration and close the window.

3.9.4 Fine tuning mesh-based lens shading phase three

Having a tuned black level is still the only prerequisite required in order for modulation of parameters to take place. This is due to lens shading being so early in the pipeline. However, by phase three all other modules should have at least had initial tuning. As a result, field testing may make shading artifacts more noticeable under particular conditions. This means that better judgment of how much shading correction to apply can be made before revisiting parameters.

Tuning procedure

The shading correction strength of calibrated LUTs can be controlled through Mesh_shading_strength, allowing for modulation according to gain. To correctly set the shading strength for each gain value, tuning must be done under different lighting conditions (lux levels) to get a correct value for every gain stated.

During calibration, observe that depending on the system specs, the lower gains (with bright lighting) may not require a decrease in strength. As light decreases and gains increase, the image will become noisier, thus it may not be possible to correct 100% of the shading artifacts without revealing too much noise in the corners. For this reason, shading strength values can be adjusted so that at higher gains noise is not boosted/revealed, where in some cases, this value may even have to be set to'0' for the highest gains (with minimal lighting).

Additionally, sinter contains radial de-noising functionality which can be combined with shading

strength. Meaning, if sinter cannot remove all of the noise in the corners without introducing too much blur, then shading strength should be adjusted to stop the noise being revealed at these gain values. In these scenarios, the trade-off between noise and "Luma correction" should be subjectively evaluated, taking into account the needs of the customer/device application.

Shading strength should be tuned by subjectively analyzing the amount of visible noise in the image, before compensating for this by correcting the respective value in Mesh_shading_strength. This manual calibration should be done for every gain in order to get the correct tuning for a range of lighting conditions and gains applied by the system.

1. Point the imaging system at a well-lit studio scene, preferably with mid-gray tones in some corners, making sure that gain is at 0 (see Figure 3.9-6. Based on phase one tuning, this should be the best achievable shading result and as such, works well as a performance indicator against worse-lit scenes requiring higher gains. Observe the corners of the image in Figure 3.9-6, noting that there is minimal noise and artifacts, allowing shading strength to be set to the maximum value (4095).

Figure 3.9-6 ISP Gain 0, 1000 lux scene.

2. Reduce the lighting until the next gain level is reached, subjectively evaluate noise levels in the scene and set a value for the corresponding gain in Mesh_shading_strength. Repeat this process for all gains. At lower gains the value may not have to be changed at all since the lighting conditions will be good and should therefore provide low noise levels.

   Some useful examples of subjective evaluation when tuning, in addition to Figure 3.9-7, are provided below:

Figure 3.9-7 ISP Gain 3, 500 lux scene.

Figure 3.9-7 demonstrates a scene in fair lighting conditions (500 lux), giving a gain value of roughly 3. At this point, noise visibility has started to increase in the image corners, thus it is acceptable to reduce the shading strength value slightly for gain 3 of the modulation table.

As for Figure 3.9-8 and Figure 3.9-9 below, the scene depicted is in low light conditions at 20 lux, resulting in maximum gain. The first of these shows a large quantity of noise in the corners, a result of the initial calibration value (4095). The second demonstrates that by reducing shading strength (to 2000 in this example), image noise is reduced while the darkening of corners is not so obvious.

Figure 3.9-8 ISP max gain, 20 lux scene - prior to calibrations mesh strength = 4095.

Figure 3.9-9 Max gain scene - post calibration. Mesh strength = 2000.

Additional tuning information

Mesh_shading_strength comes in the form of a table displaying the gain value followed by the shading strength value in the form {'n'*256, 'm'}. The shading strength register is a 16-bit register, where values of 4095 equals 100% correction. For example, the shading strength LUT after tuning will look similar to:

Mesh shading_strength=

{{0*256,4095},{4*256,4095},{6*256,2000},{8*256,1000},{10*256,0}}

Note: "Gain Value" in the context of LUTs refers to log2gain on the TCALIBRATIONS page in the "API" tab.

3.10 Lens Shading Correction（radial-based）

3.10.1 Overview

"Radial mode" of the lens shading module with Calibration Tool is the alternative to using mesh-based shading. In this method lens shading is corrected through a radial model wherein the position of the shading function for each color channel can be independently set and shading correction coefficients are provided through independent N-element LUTs.

Before using radial-based lens shading, consider that while it is useful, particularly when device memory is restricted, it also assumes that lens shading is symmetrical. In the majority of scenarios, providing memory is sufficient, mesh-based shading correction is recommended.

An overview of module tuning is provided below:

- **During phase one** - Lens shading correction tuned in lab conditions for the majority of

scenes through Calibration Tool.

- **During phase two** - No tuning required, unless previous tuning or other modules which shading correction is dependent upon causes poor performance.
- **During phase three** - No tuning required, unless previous tuning or other modules which shading correction is dependent upon causes poor performance.

3.10.2 Tuning Theory

Radial shading algorithms are fundamentally driven by a LUT made up of a series of nodes. These nodes are mapped from the first, at the center of the image, to the last, in the corner of the image. This mapping is radially distributed around the center, meaning that while center-to-corner image orientations are covered by the full range of nodes, the center-to-edge orientations are never affected by values set in the last few nodes. This is depicted in Figure 3.10-1 with a default of 33 nodes.

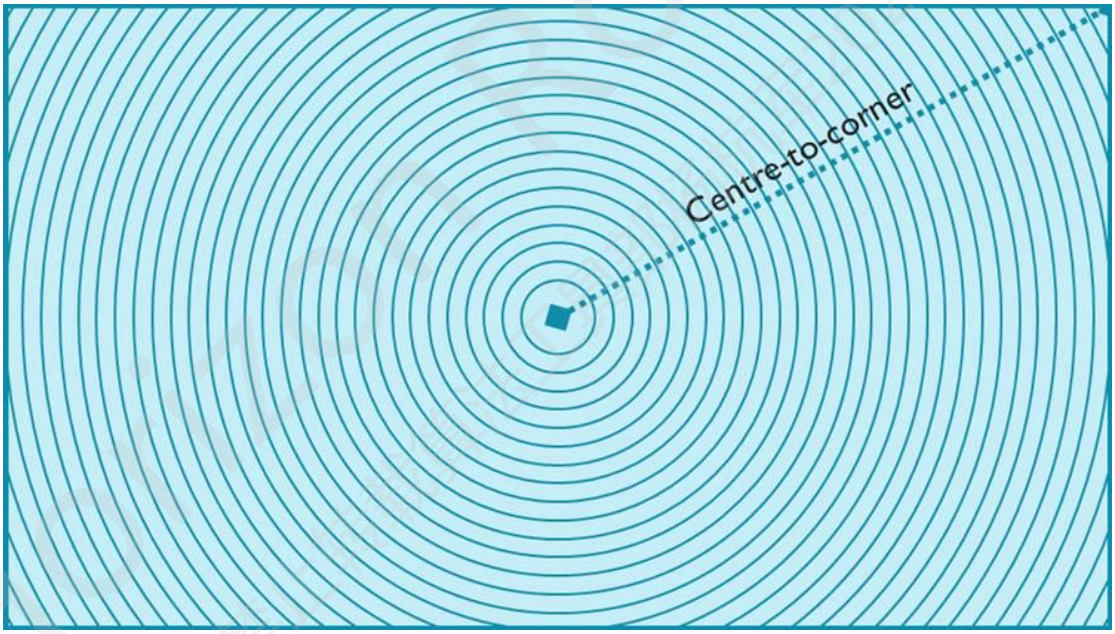


Figure 3.10-1 Radial orientation of nodes from image center.

Key hardware registers

Hardware registers are found on the relevant source page in the “Hardware” tab of Control Tool.

Register name	Source	Description
Center[R/G/B]_[x/y]	Radial Shading	Central image coordinate

Off_center_ mult[R/G/B]	Radial Shading	Scales radial table to fit center-to-corner image dimension

Table 3.10-1 Radial-based lens shading hardware registers.

Key firmware parameters for static calibrations

Static calibration parameters are found both in the "static_calibrations.c" file, and on the relevant page listed in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
Shading_Radial_[R/G/B]	LUT of radial shading correction values
Shading_radial_center_and_mult	LUT containing all central coordinates and scalars

Table 3.10-2 Radial-based lens shading static calibration parameters.

Note: [R/G/B] indicates the presence of individual tabs/parameters for each red, green and blue color channel.

3.10.3 Tuning during phase one

Lens shading correction can only be tuned once the black level has been tuned, making this the only prerequisite.

3.10.3.1 Radial shading window

When you open the radial shading module, a window like Figure 3.10-2 will pop up.

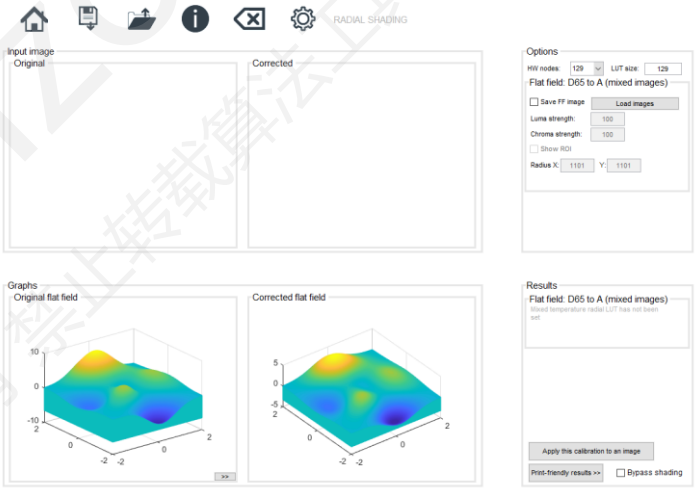


Figure 3.10-2 Radial shading window

3.10.3.2 Window features

Table 3.10-3 lists the function of each region.

Region	**Sub-region name**	**Function**
Input image	Original	Displays input flat field image
	Corrected	Displays image after mesh shading correction
Graphs	Original Flat Field	Downsampled 3D plot of each color channel before correction
	Corrected Flat Field	Downsampled 3D plot of each color channel after correction
	>> button – Undock plots	Undock current plots from window –Allows for rotation, zooming, printing or saving the figure
Options	HW Nodes	Size of the LUT as defined by hardware
	LUT size	Set the size of the shading correction LUT. We recommend you set this to the same value as HW Nodes
	Save FF image	Save corrected flat field image to a file
	Load image	Load flat field image(s)
	Luma strength	Set the strength of luma (brightness) shading correction
	Chroma strength	Set the strength of chroma (color) shading correction
	Show ROI	Display calculate region on input original image
	Radius X, Y	Set the radius of X and Y direction
Results	Flat field: D65 to A	Describes the current

		calibration status
	Apply this calibration to an image	Apply current calibration result to selected image
	Print-friendly results	Opens a window with a summary of results
	Bypass shading	Do not apply radial shading correction (uses flat values)

Table 3.10-3 List of all regions and their functions in the radial shading window

3.10.3.3 Prerequisites

- Black level calibrated.
- Gamma FE calibrated (Native WDR modes only)

3.10.3.4 Calibration images

To correctly calibrate Radial shading, capture a flat field image under 2-3 illuminants of different color temperatures.

1. Start by pointing the camera at a light box/panel capable of uniformly emitting multiple color temperatures simultaneously. Alternatively, place a diffuser in front of the lens to produce uniform illumination. Make sure the lens/sensor are as parallel to the light panel/diffuser as possible
2. Capture an image of the panel/diffuser.
3. (Optional) – If flicker (see Figure 3.10-3) is visible in the captured image, the shading calibration will not be correct. We recommend capturing 64 images so that the flicker effects will be averaged. This is usually only necessary for incandescent (A) lighting.

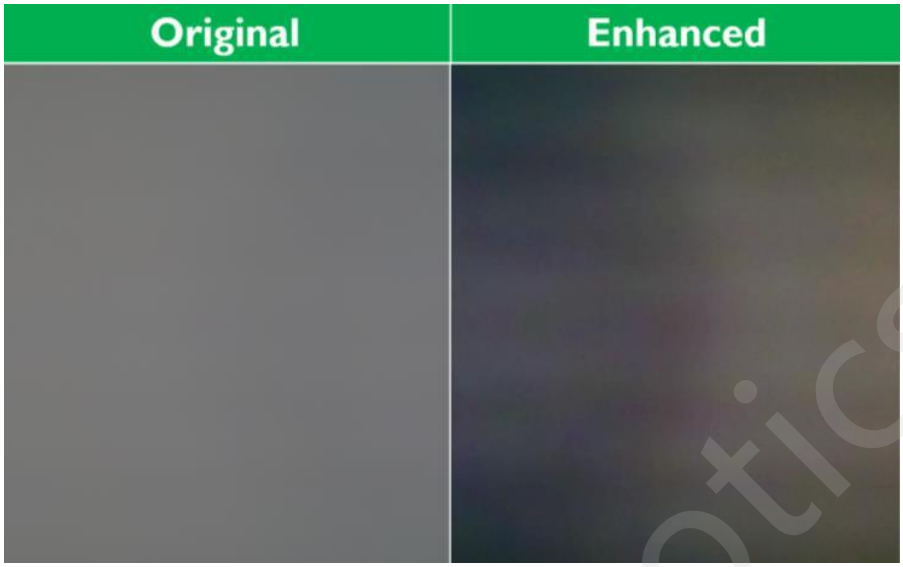
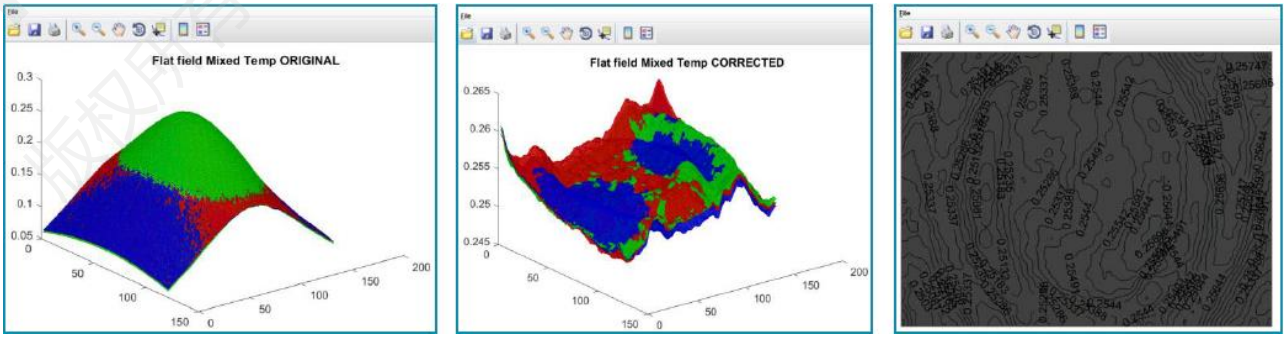


Figure 3.10-3 Example of banding (contrasting "stripes") on flat field image. Where the left image is the image captured and the right image is the same image, enhanced to help see the banding artefacts.

3.10.3.5 Calibration procedure

1. Open the radial shading calibration module from the home window of the Calibration Tool.
2. Set the values for "HW Nodes" and "LUT size". These are hardware architecture-dependent.
3. Initially set both "Luma strength" and "Chroma strength" to 100%. Luma strength is a multiplier for luma (brightness) shading correction, and Chroma strength is a multipler for color shading correction. Chroma should always be 100% corrected, to avoid color inconsistencies.
4. Click the "Load image" button and select the flat field calibration image(s).
5. Tick "Show ROI" and adjust Radius X and Y to cover full of flat field, nevertheless some of fisheye lens might need to accommodate the image circle to isolate compensation on dark corners.
6. (Optional) – Use the ">>" (undock) button to view/save plots. This will produce both the original and corrected flat field plots, along with a contour plot of channels (refer to Figure 3.10-4)



（a）Shows 3D plot of the original flat field image. （b）Shows 3D plot of the corrected flat field image. （c）Shows a contour plot of channels for the corrected image.

Figure 3.10-4 Undocked windows

7. (Optional) – Click the "Print-friendly results >>" button to view a summary of all the calibration data. This is often useful for including in reports and can be saved using the save button.
8. Click the save icon to store the calibration and close the window.

Note: As with mesh-based shading, "Chroma shading" should be 100% corrected for while "Luma shading" is normally just below this (around 85-95%).

Unlike mesh-based shading, there is no LUT to modulate results according to gain, meaning by the end of phase one, radial shading should be fully tuned with no need to revisit in phase three.

3.11 Sinter

3.11.1 Overview

The sinter hardware IP core is an advanced image noise reduction filter. It operatesin the spatial domain with RAW data and can effectively reduce the perceptual appearance of noise in an image, while maintaining texture and fine detail. The resultant processed image appears natural and avoids any visually disconcerting artifacts (for example, posterization, banding and so on) which are common among other noise reduction solutions. Register configuration also allows for fine grain control over the aggressiveness of noise reduction processing.

An overview of the tuning process can be seen below:：

- *During phase one* - No tuning required.
- *During phase two* - Sinter tuned in lab conditions for the majority of scenes.
- *During phase three* - Sinter parameters are fine tuned to suit all scenes. Primarilyinvolving modulation of parameters according to gain.

3.11.2 Tuning Theory

Sinter tuning primarily revolves around the alteration of hardware registers, the correct sinter noise profile must be provided to sinter in order to tune the block effectively.

Key hardware registers：

Hardware registers are found on the relevant *source* page listed in the "Hardware" tab of *Control Tool*.

Register Name	Sourcing	Description
Enable	Sinter	Enable/disable sinter functionality

Thresh_[1/4][v/h]	Sinter	Noise threshold for spatial frequencies
Strength_[1/4]	Sinter	Alpha blend between noisy and noise reduce image
View_filter	Sinter	Debug output - can view different scales (1/4)
Scale_mode	Sinter	Scale number
Filter_select	Sinter	Selects different filter modes (don't modify)
Int_select	Sinter	De-noising on either intensity or RAW image
Rm_enable	Sinter	Enable/disable radial de-noising
Rm_center_x	Sinter	Central x-axis coordinate
Rm_center_y	Sinter	Central y-axis coordinate
Rm_off_center_mult	Sinter	Scales radial LUT to fit center-to-corner dimension
Rm_shading_lut	Sinter	LUT of radial correction values
Int_config	Sinter	Alpha blend between RAW and intensity image
Nlm_en	Sinter	Enable/disable non-local mean filtering
Nonlinear_wkgen	Sinter	Enable/disable SAD filtering
Sad_filt_thresh	Sinter	Balanced contrast between edge detail and flat region smoothness

Table 3.11-1 Sinter Hardware Register

***Note**：[1/4] denotes that there are two registers present, where those affecting high spatial frequencies can be identified by "1" and those relating to low spatial frequencies are classified by "4". The same logic is applied to [v/h], used to denote the given vertical ("v") or horizontal ("h") orientation.*

Key firmware parameters for dynamic calibrations：

Dynamic calibration parameters are found both in the "dynamic_calibrations.c" file, and on the relevant page listed in the "Dynamic Calibrations" tab of *Control Tool*.

Register Name	Description
Sinter_strength	Modulation LUT - {gain, strength}
Sinter_strength1	High frequency modulation LUT - {gain, strength}
Sinter_thresh[1/4]	Modulation LUT - {gain, threshold}
Sinter_sad	Modulation LUT - {gain, } Sad_filt_thresh
Sinter_radial_lut	LUT defining sinter threshold offset with respect to radial distance from image center

Sinter_strength_MC_contrast	Modulation LUT -{contrast calculated in iridix, noise level 0 in sinter noise profile } this is only for fs-lin mode
Sinter_strength_intConfig	Modulation LUT -{gain, intConfig }

Table 3.11-2 Sinter Dynamic Calibration Parameter

3.11.3 Tuning Sinter during phase two

As sinter is not required during sensor characterization in phase one, tuning starts in phase two. This can only be done after completing the prerequisites listed in Table 3.13-3.

Prerequisiters	**Status / Value**
Black level	Tuned
DPC	Tuned
Green equalization	Tuned
Demosaic	Tuned
Temper	Tuned
Sinter noise profile	set

Table 3.11-3 Sinter prerequisites

As well as these requirements, a properly focused studio scene is also needed. This should contain a ColorChecker chart, lots of textured objects, colorful textures, flat areas and/or resolution chart ISO 12233.

Tuning procedure - Overview

An outline of the tuning process is depicted in Figure 3.11-1. Tuning of sinter requires a great deal of subjective and objective judgment, where for proper tuning, a large number of images varying in content, detail and lighting conditions are required for processing and analysis. The general rule of thumb is to apply minimum sinter threshold at lower gain values and increase sinter threshold at higher gains.

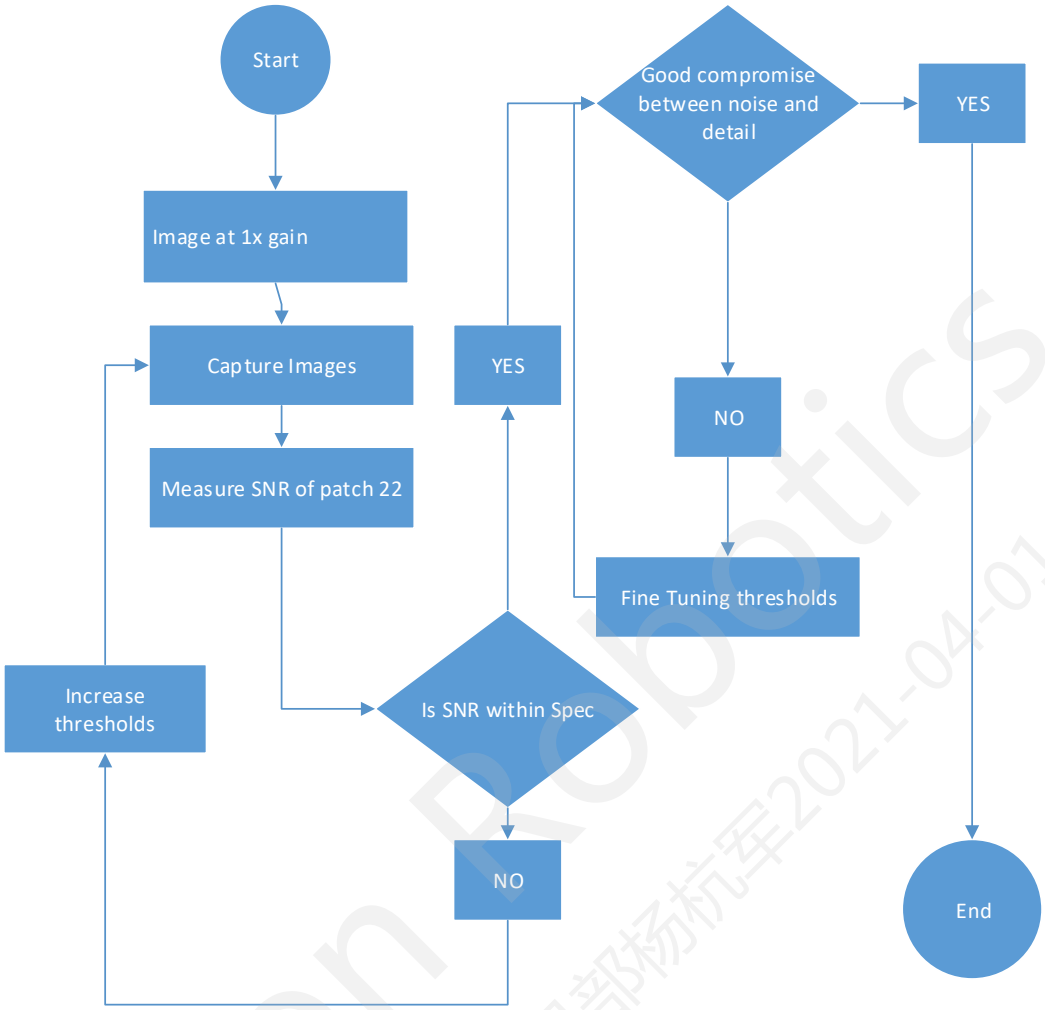


Figure 3.11-1 Flow diagram for tuning sinter.

Note: SNR measurements are relatively good as a starting point for noise evaluation. However, they do not reflect visual perception of noise vs detail and cannot be used for subjective analysis.

As tuning for sinter has a very subjective nature, it will be broken down and individually described for each group of registers.

1. Set the default sinter values listed in Table 3.11-4 below

Parameter	Value
Sinter_enable	0
Sad_filt_thresh	7
Nlm_en	1
Nonlinear_wkgen	1
Thresh_0h	0
Thresh_1h	0

Thresh_2h	0
Thresh_4h	0
Thresh_0v	0
Thresh_1v	0
Thresh_2v	0
Thresh_4v	0
Strength_0	255
Strength_1	255
Strength_2	255
Strength_4	255
View_filter	0
Scale_mode	3
Filter_select	0
Int_select	1
Int_config	7
Rm_enable	0
Rm_center_x	Set (system dependent)
Rm_center_y	Set (system dependent)
Rm_off_center_mult	Set (system dependent)
Rm_shading_lut	Set (system dependent)

Table 3.11-4 Sinter default register

2. Set Enable to *0* to disable sinter and capture an image at *Gain 1* of the studio scene containing a ColorChecker chart, before running the image through Imatest's ColorChecker module.

3. Use Imatest to analyze SNR from patch 22, also defined as patch 4 in the output SNR plot (see Figure 3.11-2). This will be the starting point before sinter is enabled.

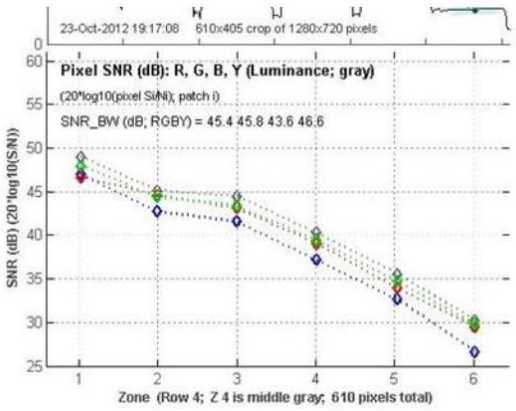


Figure 3.11-2 Imatest ColorChecker SNR output

4. Now set Enable to *1* and observe/inspect the image, observing any changes in image texture and SNR impact.

Tuning pocessing – Global noise Filtering

The global noise filtering will affect image in all frequencies. Linear mode: The strength of sinter global noise filtering is equal to global_offset plus noise_level_0 in sinter noise profile. FS-Lin mode: The strength of sinter global noise filtering depends on the exposure from which the pixel originates. The strength is equal to global_offset plus noise_level_N, where N is the number corresponding to the exposure. For example, if global_offset = 5 and noise_level_0 = 64, the total strength used for pixels from exposure 0 will be 69.

Tuning processing– Frequency noise filtering

Alongside the global noise control previously mentioned, it is also possible to fine tune the aggressiveness of the noise reduction according to each frequency band through Thresh_ 1h, Thresh_1v, Thresh_4h and Thresh_4v. Additionally, Strength_1 and Strength_4 control the (linear) blending between the original noisy input content and the filtered result. here, a strength value of *255* for a given scale indicates that only the filtered result is being used.
While "1" relates to high spatial frequencies and "4" denotes low spatial frequencies are being altered, these registers are closely related in that the tuning principle behind them is the same. Low frequency noise is seen as blotches throughout the image, while high frequency noise appears more speckled in the form of "salt and pepper" noise.

5. Identify whether either/both types of noise are present in the image. If a type of noise is not present then the relevant Thresh_[1/4]v and Thresh_[1/4]h should both be setto *0*. Where these forms of noise are visible, increase the respective Thresh_[1/4]v and Thresh_[1/4]h to improve SNR. Note that these values should be equal in value, and if increased too much, excessive blurring will occur.
In order to avoid too much blurring, which can create a synthetic looking image, it's possible to decrease the applicable Strength_[1/4]. As this lets noise through, it should only be done with the intention that noise will replicate detail/texture.

Tuning processing – Non-local mean filtering

Non-local mean (NLM) filtering is present in *scale 1*, affecting high frequencies. These registers can

be used to remove noise without blurring/losing edges in the image.

6. In order to control NLM, set both Nlm_en and Nonlinear_wkgen to *1*.

7. Now use Sad_filt_thresh to balance edge preservation with smoothness of flat areas. Continually analyze the image subjectively until the optimum trade-off is found.

Note: It is generally recommended that tuning of NLM filtering takes place alongside that of other scale 1 registers, being "Thresh_1[v/h]***" and "***Strength_1***".***

If a lens shading/vignetting compensation block is present in the ISP pipeline, it is likely to have exacerbated noise in the corners of the image frame. To help reduce this, sinter contains radial noise reduction registers which variably increase the strength of multi-scale filter thresholds, dependent on pixel location. More specifically, the pixel needs to fall in the programmed radial distance from the user defined image center. Figure 3.11-3 depicts the 33 nodes radial orientation with respect to the image frame which acts as the framework for Rm_shading_lut.

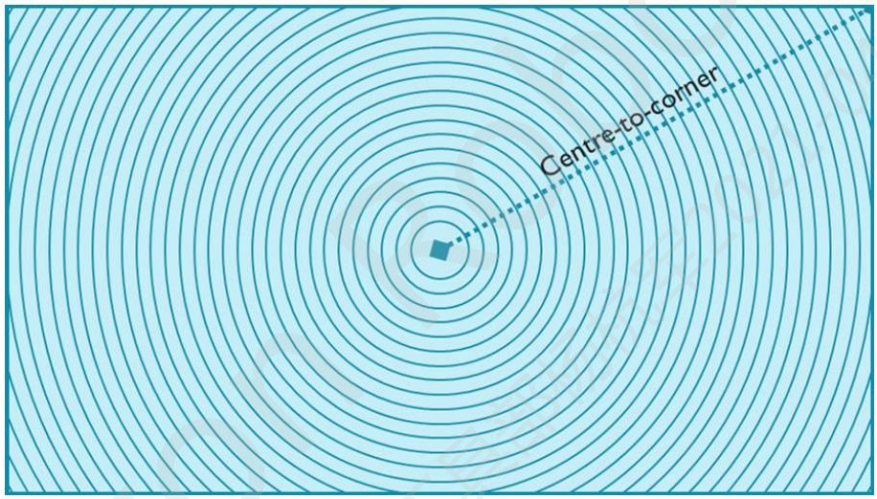


Figure 3.11-3 Radial orientation of nodes from image center.

8. If radial noise correction is not required, then set Rm_enable to *0* and continue with tuning the rest of sinter. Where radial correction is required, start by setting the center point, defined by Rm_center_x and Rm_center_y, to half the image resolution for each respective orientation. For example a 512x512 image would give central coordinates of Rm_center_x = 256 and Rm_center_y = 256.

9.Now set the Rm_off_center_mult where：

$$\text{rm\_off\_center\_mult} = 2^{31} \ / \ maximum\_radial\_distance^2$$

$$\text{maximum\_radial\_distance} = rm\_center\_x^2 + rm\_center\_y^2$$

Usin the 512 x 512 examples,the appropriate vaue for rm_off_center_mult is：

$$\text{rm\_off\_center\_mult} = 2^{31} \ / \ (256^2 + 256^2) = 16384$$

10. Assigning values to nodes in Sinter_radial_lut is the next step. These values will apply an offset to the thresholds set in all scales and main thresh. In order to set appropriate values, visually and objectively (SNR) assess the whole image. In the example demonstrated below, the extra threshold applied during radial filtering only affects the few nodes nearest to the corners, while the rest of the image is left untouched.

Sinter_radial_lut[]={0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,2,4,6,8,64}

Tuning processing – Intensity Filtering

In addition to the filters previously mentioned, Int_config allows for the removal of saturated high frequency noise ("sparkles"). It acts as an alpha blend coefficient between the *RAW* and *intensity* image, where a value of *0* only filters the intensity image and a value of *15* only filters the *RAW* image.
11. Int_config should be set to *7* to give an even blend, unless DPC cannot remove all hot pixels. In many cases, sparkles may be introduced by a combination of sinter's, demosaic's and sharpening's core tuning. Thus, before modifying Int_config make sure that the previously mentioned modules are not the cause.
Adjustment of strength of intensity filter with respect to total gain uses the Sinter_strength_intConfig LUT, see an example of this below:

Sinter_strength_intConfig={

1. {0*256,0},
2. {1*256,10},
3. {5*256,20},
4. {10*256,30}

};

Tuning processing – Verification

12. After visually adjusting all parameters explained above, capture another image of the studio scene for *all gains* and run the ColorChecker chart through the Imatest's ColorChecker module. If SNR values and details/textures are well preserved then sinter tuning can be considered as complete until phase three. If not, repeat the tuning and verification cycle until sinter meets the specification.

Additional tuning information

The following SNR values in Table 3.11-5 could be used as reference SNR targets. Figure 3.11-4 shows an example of sinter tuning where SNR values are *above 40* (*> 40*), observe how in the example image - the noise is low and the textures are preserved.

Figure 3.11-4 Sinter tuned at 1x Gain

D65 1x gainsetting	Settings（halogen）@max gain(system dependent)
Good: 40 - 50	Good: 25-35
Fair: 35 - 40	Fair: 15 - 25
Poor: < 35	Poor: < 15

Table 3.11-5 Target SNRvalues

Note: It should be pointed out that these targets mentioned in the table above are a rough guide and are very much sensor noise dependent.

3.11.4 Fine tuning demosaic phase three

Tuning of sinter continues in phase three where previously tuned registers are adjusted with respect to total system gain (log2gain). Other ISP modules will also have an impact on sinter tuning. Therefore it is recommended to tune DPC, sinter, temper, demosaic and sharpening together, gain by gain, to ensure that each gain holds correctly tuned values for each of the fine tuning parameters. Iridix is affected by contrast level, higher contrast level may lead higher strength in iridix strength and dark_enh strength, therefore, higher noise may have in shadow region. Sinter_strength_MC_contrast is modulated by contrast to reduce noise in this case. (this is only for fs-lin mode)

Latest imaging sensors can be almost noise free in good lighting/gain 1 conditions. If this is the case, reduce the lighting conditions to the point gains increase and noise is visible. For all gain where noise was not visible, threshold should be kept to a minimum.

Tuning processing – Global noise Filtering

Adjustment of global noise filtering with respect to total gain uses the Sinter_strength LUT, see an example of this below:

```
Calibration_sinter_strength={
                                {1*256,1},
                                {3*256,16},
                                {5*256,30},
                                {10*256,58}
                              };
```

In practice the effect of these parameters is to add a global offset to all scale filters as follows:

Total_thresh = noise_level_N + global_offset + rm_lut_thresh

For scale 1 only = Total_thresh + thresh_1

For scale 4 only = Total_thresh + thresh_4

Set sinter_strength values to meet the oriteria for all gains.

Adjustment of global noise filtering with respect to iridix contrast uses the *Sinter_strength_MC_contrast* LUT (this is only for fs-lin mode), see an example of this below:

```
Sinter_strength_MC_contrast={
```

{10,0},

{20,40},

{80,70},

{100,90}

};

Tuning processing – Frequency noise filtering

Modulation LUTs are also provided for high and low frequency filtering. Note that the modulation table automatically controls both horizontal and vertical thresh registers, anexample is provided below:

Calibration_sinter_thresh4_[h/v]={

{0,1},

{7*256,1},

{9*256,4},

{10*256,8},

{12*256,12}

};

Calibration_sinter_thresh1_[h/v] = {

{0,1},

{11*256,1},

{12*256,4}

};

Calibration_sinter_strength1 = {

{0,190},

{4*256,195},

{5*256,205},

{10*256,215}

};

Tuning processing – Non-local mean filtering

In dynamic_calibrations.c, a modulation table is provided to adjust Sad_filt_thresh according to total gain of the system (log2gain). It is recommended to tune this parameter alongside Sinter_thresh1 and Sinter_strength1. An example of the LUT is shown below:

Sinter_sad={

{0,3},

{4*256,7},

{9*256,15},

{11*256,28}

};

3.12 Temper

3.12.1 Overview

This module is a motion-adaptive temporal noise reduction filter. Noise is reduced through image averaging with respect to time. A brief overview of the tuning procedure is described below:

- During phase one - No tuning required.
- During phase two - Temper tuned in lab conditions for the majority of scenes.
- During phase three - Temper parameters are fine tuned to suit all scenes. Primarily involving modulation of parameters according to gain.

3.12.2 Tuning Theory

Image averaging is conducted recursively, taking the current frame and averaging it against the previous history of frames. This recursion level is set locally by the degree of local motion detected in the current frame.

Key hardware registers

Hardware registers are found on the relevant source page listed in the "Hardware" tab of Control Tool.

Register name	Source	Description
Global_Offset	Temper NP	Global strength of temper
Recursion_limit	Temper	Controls length of frame accumulation

Table 3.12-1 Temper hardware registers.

Key firmware parameters for dynamic calibrations

Dynamic calibration parameters are found both in the "dynamic_calibrations.c" file, and on the relevant page listed in the "Dynamic Calibrations" tab of Control Tool.

Parameter name	Description
Temper_strength	Modulation LUT - {gain, temper strength}

Table 3.12-2 Temper dynamic calibration parameters.

3.12.3 Tuning Temper during phase two

As temper is not required for sensor characterization, tuning starts in phase two. This can only be done after completing the prerequisites listed in.Table 3.12-3

Prerequisite	Status / value
Global_Offset	78
Recursion_limit	1

Table 3.12-3 Temper prerequisites.

To tune temper, set up a similar studio scene to that used for sinter, containing a moving object and the usual ColorChecker chart, textured objects, colorful textures, resolution charts and flat areas. Including a moving object allows for verification of excessive ghosting, as seen in figure 48 where a toy train is used. A metronome is another example of a good verification object.

Figure 3.12-1 Studio scene for tuning temper.

Tuning procedure

In order to set the strength of temper, in Temper noise profile Tab Global_Offset parameter need to be adjusted to the point where temporal noise is removed but no ghosting artifacts are visible. The correct trade-off between the two must be found.

1. Set the system so that total gain = 1 and adjust the Global_Offset to the point where ghosting artifacts are not visible and noise is reduced.
2. Set the temper strength value in the modulation LUT for the gain tuned.
3. If default performance, based on the configured temper noise profile, is not optimal, adjust the Recursion_limit to increase or decrease the strength of noise reduction. If performance is

still not optimal, adjust the Global_Offset parameter to further increase or decrease noise reduction strength.

The value given to Recursion_limit sets the average depth of recursion. In turn this sets the effective number of frames over which temporal averaging is performed, depending on the extent of motion in a given region. Increasing this parameter leads to smaller recursion depth, with smaller noise reduction effect and minimized motion artifacts. When Recursion_limit is set to 0, up to 16 frames can be averaged. When Recursion_limit is set to 0xf, frames are not averaged, this is the equivalent of disabling temper.

3.12.4 Fine tuning Temper during phase three

The strength of temper can be controlled through the Temper_strength modulation LUT. It is driven by total system gain (log2gain) and adjusts Global_Offset.

Rough guidelines for range of parameter values after tuning temper across all gains are presented below, note that these will be dependent on the sensor.

60 > Minimum temper strength < 90
90 > Maximum temper strength < 160

By adjusting the lighting on the studio scene, complete the appropriate Temper_strength modulation table for temper at each gain.

Note: It is recommended that tuning is conduced alongside that for sinter and demosaic modules.

3.13 Chromatic Aberration (CA) correction

3.13.1 Overview

Chromatic aberration (CA) generally describes the failure of a lens to focus all wavelengths of light to the same point.

Lateral CA occurs when light of different wavelengths is focused to different points on the image sensor. This results in a spatial offset between the pixel data of different color channels, typically most severe in the corners of the image. The CA correction module corrects this.

Note: Longitudinal CA is a related phenomenon in which light of some wavelengths is focused behind (or in front of) the sensor. Unlike lateral CA, this is not directional and is not corrected using this module.

The CA correction module corrects these spatial offsets by shifting the pixel values for each color channel horizontally and vertically in the Bayer domain, to align them with the green color channel.

The spatial offset varies across the image, so first the image is divided into a 64x64 grid of tiles. Within each tile, a horizontal and vertical shift are applied to the pixels of each color channel.

An overview of module tuning is provided below:

- During phase one - CA correction tuned in lab conditions for the majority of scenes through Calibration Tool.
- During phase two - No tuning required, unless previous tuning causes poor performance.
- During phase three - No tuning required, unless previous tuning causes poor performance.

Key hardware registers

Hardware registers are found on the relevant source page in the "Hardware" tab of Control Tool.

Register name	Source	Description
Mesh_scale	CA correction	Multiplier for ca correction_mem values

Table 3.13-1 CA correction hardware registers.

The default Mesh_scale value of 0 means that shift values in the correction mesh are twos complement signed integers with units of 1/16th pixel, as shown in Table 3.13-2. Do not change this value.

Mesh scale	Format of coefficient
0	Signed fixed point s3.4
1	Signed fixed point s4.3
2	Signed fixed point s5.2
3	Signed fixed point s6.1

Table 3.13-2 Mesh scale format.

Key firmware parameters for static calibrations

If supported by your firmware, static calibration parameters are found both in the "static_calibrations.c" file, and on the relevant page listed in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
ca_correction_mem	Mesh of horizontal and vertical correction values
ca_filter_mem	FIR filter coefficients used for shift

Table 3.13-3 Mesh-based lens shading static calibration parameters.

Key firmware parameters for dynamic calibrations

The CA correction module uses a static mesh, so no dynamic calibration is required.

3.13.2 CA tuning during phase one

CA correction has no prerequisites, and can be tuned as early or late during phase one as desired.

3.13.2.1 CA calibration window

When you open the CA module, a window like Figure 3.13-1 will pop up. This module handles all lateral chromatic aberration (LCA) calculations.

In this document, "CA" describes the phenomenon of lateral chromatic aberration, which is treated as a spatially-variant misalignment vector between CFA channels.

In this document, "LCA" describes the length of this vector in pixel-widths.

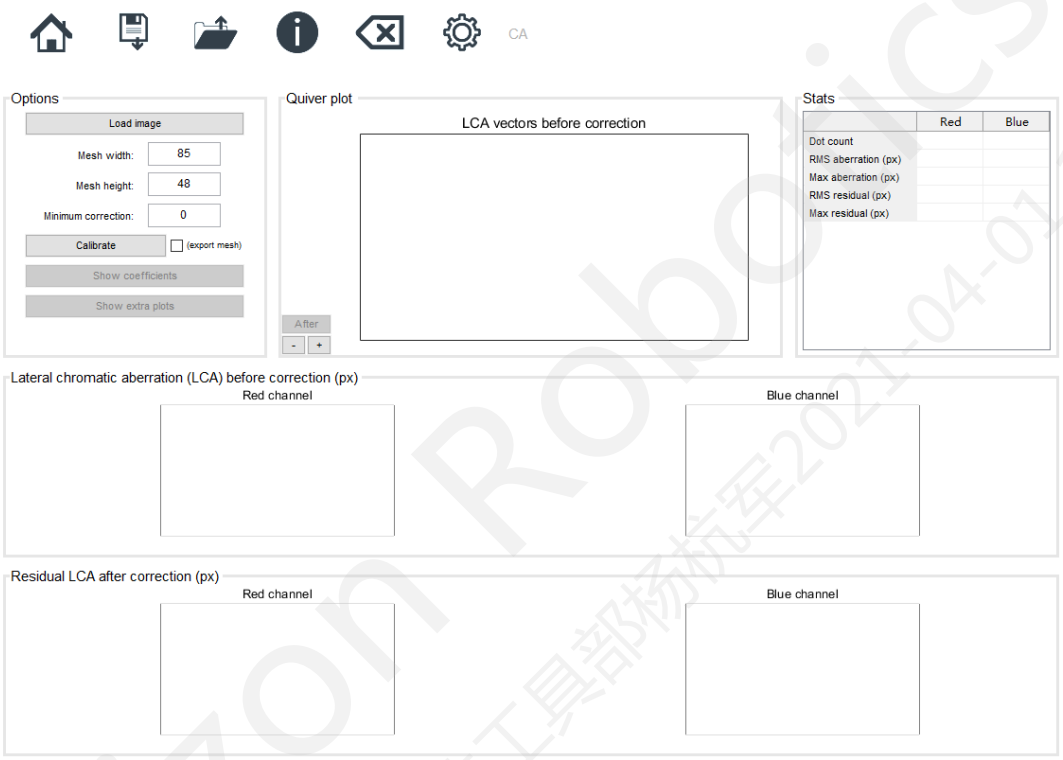


Figure 3.13-1 CA window

3.13.2.2 Window features

Table 3.13-4 lists the function of each region.

Region	Sub-region name	Function
Options	Load image	Load a dot pattern image (and launch the new image interface)
	Mesh width/height	Set the size of the correction mesh. This affects the CA model calculation
	Minimum correction	All values in the correction mesh that are smaller than this will be forced to zero.

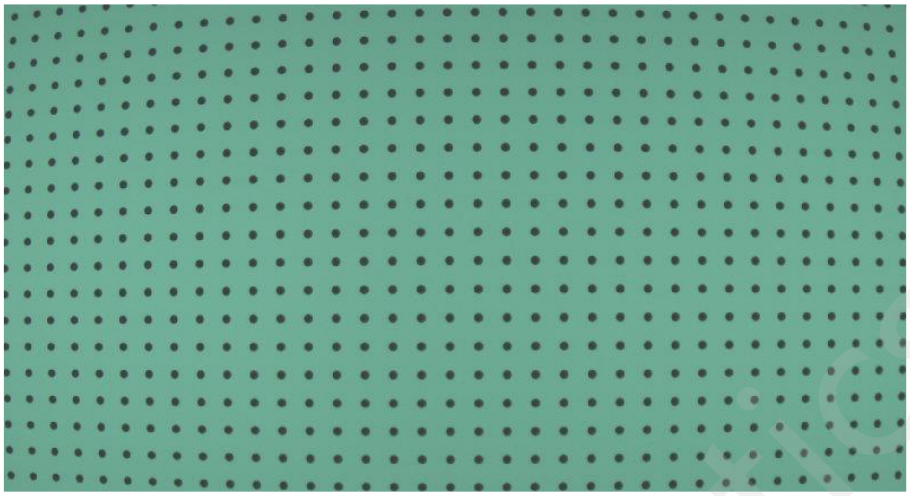
	Calibrate	Generate CA model and display contour plots and statistics
	Show coefficients/show quiver plot	Toggle between displaying the CA correction model values and the quiver plot
	Show extra plots	Pop out windows with CA model visualizations
Quiver plot	LCA vectors plot	Plot showing the aberrations calculated for every dot in the captured image(s)
	After/before	Toggle between showing CA vectors before correction and after correction
	-/+	Decrease/increase scale of vector arrows, to make the data easier to see
Stats	Table	Statistics calculated from each dot in the captured image(s)
LCA before correction (px)	Contour plots	Graphic showing the spatial variation in CA magnitude (LCA in pixels)
Residual after correction	Contour plots	Visualizes the meshes that have been interpolated from the correction vectors, showing magnitude only (not direction)

Table 3.13-4 List of all regions and their functions in the CA window

3.13.2.3 Calibration images

Before using the CA module, set the correct image format for your test images, using the Image format window (icon). You do not need to calibrate any other modules before calibrating CA.

To calibrate CA, capture at least 1 focused and properly exposed raw image of a dot pattern chart:

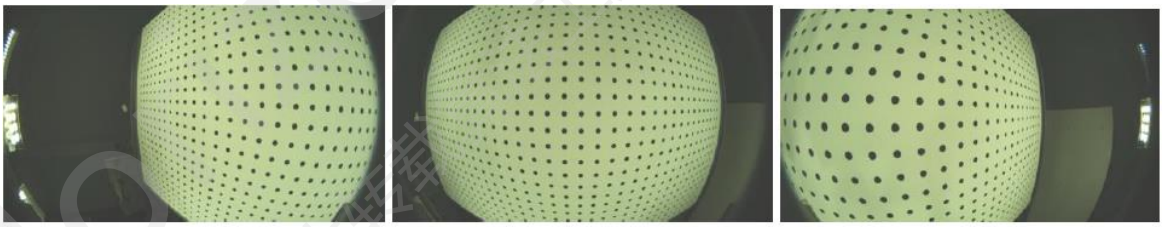


Information about dot pattern test charts can be found here:http://store.imatest.com/dot-pattern-test-chart.html

It is not essential to use the official chart, as the dots do not need to be precisely positioned or printed in very high resolution. You can create and print your own dot pattern (Imatest software has a utility for this). If you do this, make the spacing between the dots at least 2x the diameter of the dots.

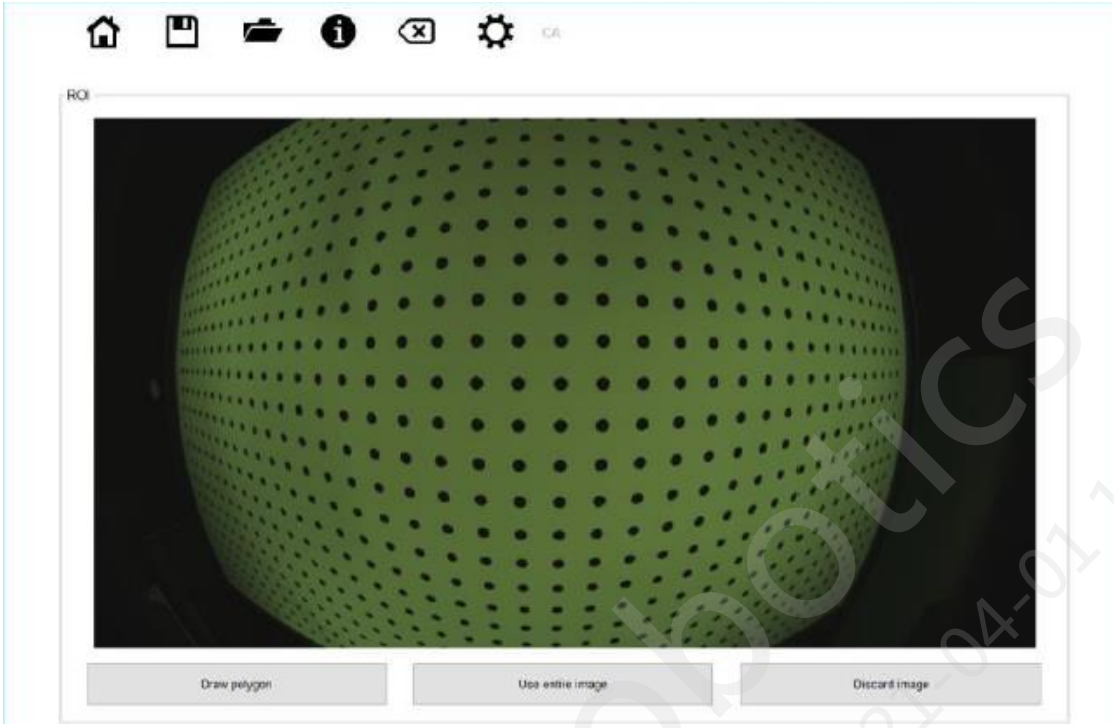
The CA module will calculate the lateral chromatic aberration at the location of each dot, so position the camera so that there are dots in every area of the field of view, especially in the extreme corners of the image.

If you are using a fisheye lens, it will not be possible to do this with one image, so capture multiple images, one with a lot of dots in the center of the image, and extra images with the chart in the extreme right and left of the image:



3.13.2.4 Calibration procedure

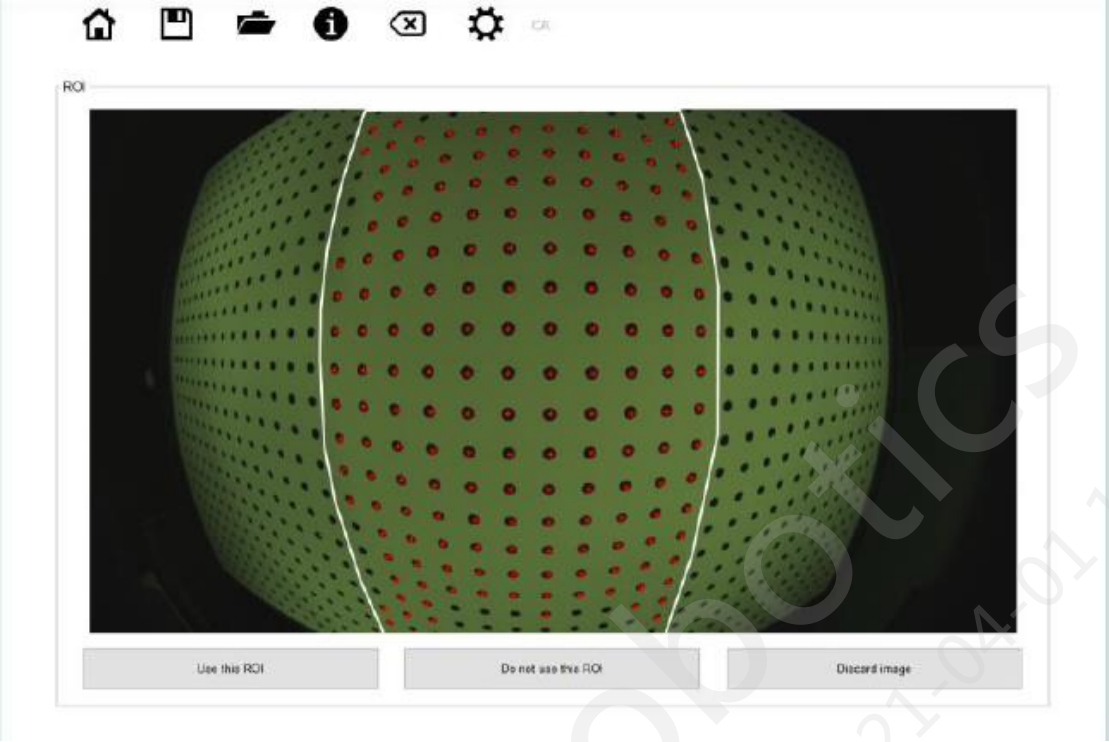
1. Click “Load image”, then use the file browser to select 1 image
2. The image will appear (see below). If the chart fills the image, click “Use entire image”. Otherwise, click “Draw polygon” to select the region covered by the chart.



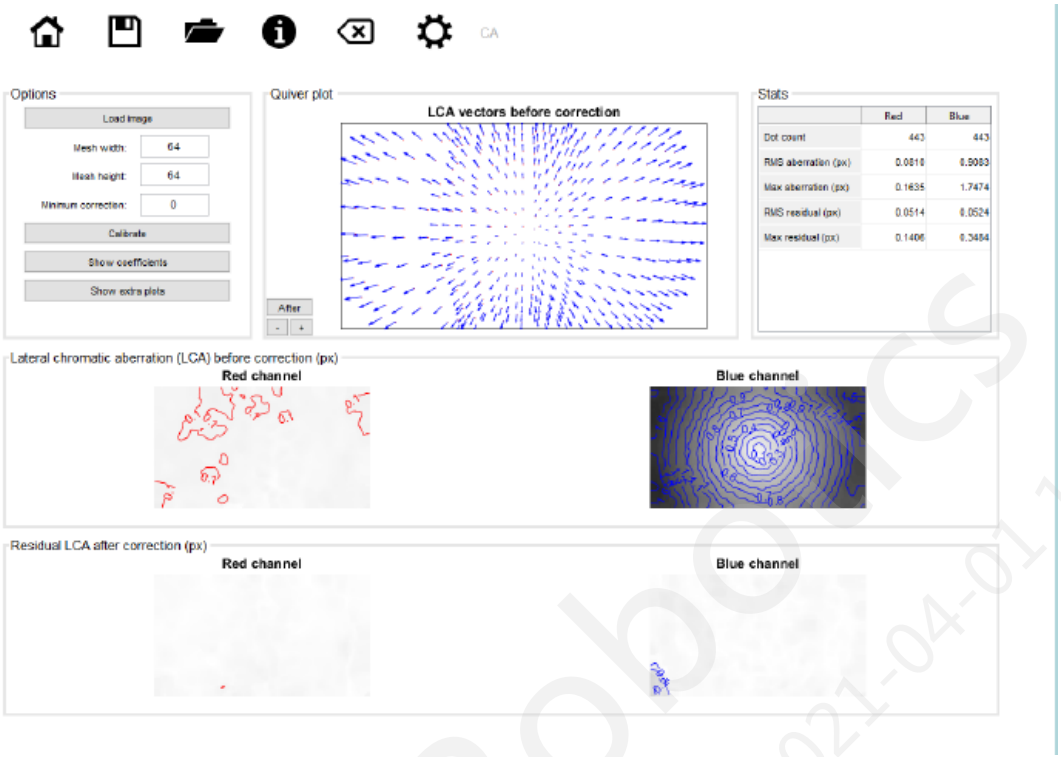
3. When drawing a polygon, click your mouse to place a corner of the polygon. When you are ready to complete the shape, right click anywhere on the image or hover the mouse over the first corner and the cursor will turn into a circle. Click to complete the shape.

   Note: If the image settings are not correct, or you chose the wrong image, click "Discard image" to return to the CA window

4. The utility will find the dots and display their locations on the image as red stars. Check the location of each red star (see below). If any of them are not on dots in the test chart, click "Do not use this ROI" and draw the polygon again, avoiding the region that had an incorrect red star.
5. Once all red stars are on dots, click "use this ROI". The aberrations at each dot will be calculated and plotted.



6. If you have more images, go back to step 1.
7. Once there is data across the full extent of the frame, click “Generate mesh”. This will model the horizontal (h) and vertical (v) components of aberration for each color channel separately using 3rd order polynomial surfaces. For a sensor with an RGGB color filter array, there are 4 models (rh, rv, bh and bv) with 10 coefficients each.
8. “Before correction” and “after correction” contour plots will appear for each color channel (not including green, which is used as the reference).



Note: The polynomial models are only valid for the mesh width and mesh height values shown in the top left. Initial width and height values are calculated by the tool using the aspect ratio of the image, and are limited by the size of the CA mesh (16384 8-bit values) and the minimum tile width (30 pixels).

9. Try changing mesh width and mesh height and check the RMS residual values in the stats table. The default width and height might not give the best results, but a value below 0.1 pixels implies that LCA will be imperceptible after correction.
10. Click the save button to record the results, then return to the home window.

    Note: Before correction, the image can have extremely low LCA in the centre of the frame. For this small region the LCA after correction might be higher than before correction. If this happens, use the "minimum correction" parameter to set a minimum LCA to correct in units of 1/16th pixel. A value of 1 is high enough for most cases.
11. Then,the parameters will be saved in "static_calibrations.json", It can be translated to C source and generated in dynamic .so lib。

3.14 Iridix

3.14.1 Overview

Iridix uses local tone mapping for dynamic range compression (DRC), attempting to bring back detail from areas of reduced visibility in an HDR scene without affecting the global image. Overall

this increases the range of tones available to local regions by increasing gain with respect to scene content (See Figure 3.14-1).

Figure 3.14-1 Iridix example (disabled/enabled).

Iridix tuning is introduced at slightly different stages in the tuning process depending on the ISP mode used. However, the tuning process for iridix stays the same for each ISP mode, requiring the majority of other modules to have previously been tuned. For linear mode the process is outlined below:

- **During phase one** - No tuning required
- **During phase two** - Iridix tuned in lab conditions for the majority of scenes. Tuning is also verified at the end of phase two.
- **During phase three** - Iridix parameters are fine tuned to suit all scenes. Primarily involving modulation of parameters according to gain.

Note: Where iridix aims to increase the perception of content in a scene, color and sharpness are preserved while contrast is enhanced.

3.14.2 Tuning Theory

Using space-variant/pixel-by-pixel algorithms, iridix dynamically adapts, producing a different response curve for each input pixel. This means all lighting conditions can be accounted for by setting core parameters.

Key hardware registers

Hardware registers are found on the relevant source page listed in the "Hardware" tab of Control Tool.

Register name	Source	Description
Iridix_on	Iridix	Enable/disable iridix.
Iridix_LUT_Asymmetry	Iridix	Iridix response curve
Black_level	Iridix	iridix processing will be active only on intensity levels above the black level value set
White_level	Iridix	iridix processing will be active only on intensity levels below the while level value set
Svariance	Iridix	Spatial adaptivity control.
Strength_inroi	Iridix	Strength of iridix application
Strength_outroi	Iridix	Strength of iridix application
Roi_[hor/ver]_[start/end]	Iridix	Defines ROI for Strength
Dark_enh	Iridix	Limits gain to dark regions
Bright_pr	Iridix	Limits gain to prevent clipping in highlights
Contrast	Iridix	enhances local contrast

Table 3.14-1 Iridix hardware registers.

Note: [hor/ver] acts as a place holder, indicating there are actually two registers, one for each respective "horizontal" and "vertical" plane. The same principle is applied to [start/end], where a register is present for each.

The fundamental register is Iridix_LUT_Asymmetry, providing the initial response curve. It can be considered that while the asymmetry curve is what primarily drives iridix, this curve is shaped and altered by dark enhancement, contrast and bright preservation. There are two hardware registers providing the main control over iridix, being Strength (Strength_inroi and Strength_outroi) and Dark_enh (dark enhancement). These two are driven by the ISP driver.

Key firmware parameters for static calibrations

Static calibration parameters are found both in the "static_calibrations.c" file, and on the relevant page listed in the "Static Calibrations" tab of Control Tool.

Parameter name	Description
Iridix_asymmetry	LUT for an iridix parameter in the source code

Table 3.14-2 Iridix static calibration parameters

Key firmware parameters for dynamic calibrations

Dynamic calibration parameters are found both in the "dynamic_calibrations.c" file, and on the relevant source page listed in the "Dynamic Calibrations" tab of Control Tool.

The parameters listed below are used to drive the previously mentioned **Strength, Dark_enh**.

Parameter name	Source	Description
Time_filter	Iridix_avg_coef	Iridix time filter
Iridix_strength_maximum	Iridix_strength_maximum	Maximum iridix strength
Iridix_min_max_str	Iridix_min_max_str	Minimum iridix strength
Iridix_ev_lim_full_str	Iridix_ev_lim_full_str	Above this EV value, iridix strength is set to max.
Iridix_ev_lim_no_str [0]	Iridix_ev_lim_no_str	Below this EV value, iridix strength is set to Iridix_min_max_str.
Iridix_ev_lim_no_str [1]	Iridix_ev_lim_no_str	Below this EV value, iridix dark enh is set to min Min_dk.
Dark_prc	Iridix8_str_dk_enh_ctrl	Defines the dark region in percentage of pixels
Bright_prc	Iridix8_str_dk_enh_ctrl	Defines the bright region in percentage of pixels
Min_dk	Iridix8_str_dk_enh_ctrl	Minimum applicable Dark_enh
Max_dk	Iridix8_str_dk_enh_ctrl	Maximum applicable Dark_enh
PD_cut_min	Iridix8_str_dk_enh_ctrl	Minimum intensity cut for dark regions in which dk_enh will be applied"
PD_cut_max	Iridix8_str_dk_enh_ctrl	Maximum intensity cut for dark regions in which dk_enh will be applied
Dark_contrast_min	Iridix8_str_dk_enh_ctrl	Minimum contrast driving Dark_enh

Dark_contrast_max	Iridix8_str_dk_enh_ctrl	Maximum contrast driving Dark_enh
Iridix_gain_max	Iridix8_str_dk_enh_ctrl	Maximum iridix gain
Min_str	Iridix8_str_dk_enh_ctrl	Minimum iridix strength in percentage 50 = 1x gain,100 = 2x gain
Max_str	Iridix8_str_dk_enh_ctrl	Maximum iridix strength in percentage 50 = 1x gain,100 = 2x gain
Dark_prc_gain_target	Iridix8_str_dk_enh_ctrl	target in histogram (percentage) for dark_prc after iridix is applied
Contrast_min	Iridix8_str_dk_enh_ctrl	clip factor of strength for LDR scenes
Contrast_max	Iridix8_str_dk_enh_ctrl	clip factor of strength for HDR scenes
max_iridix_gain	Iridix8_str_dk_enh_ctrl	Maximum iridix gain allowed
Print_debug	Iridix8_str_dk_enh_ctrl	Displays all parameter information

Table 3.14-3 Iridix dynamic parameters.

Note: Iridix8_str_dk_enh_ctrl, Max_str_dk_ev and Min_str_dk_ev are shorthand parameter names, used in this document for ease of reference.

Figure 3.14-2 and Figure 3.14-3 provide a rough overview of how some of these parameters can be identified with respect to a typical LDR histogram before the application of iridix. The first parameters to take note of are Dark_prc and Bright_prc. These are defined in terms of percentages to allow parameters to adapt to different scenes. As shown in Figure 3.14-2, a cumulative histogram is used to identify the pixels assigned to these regions.

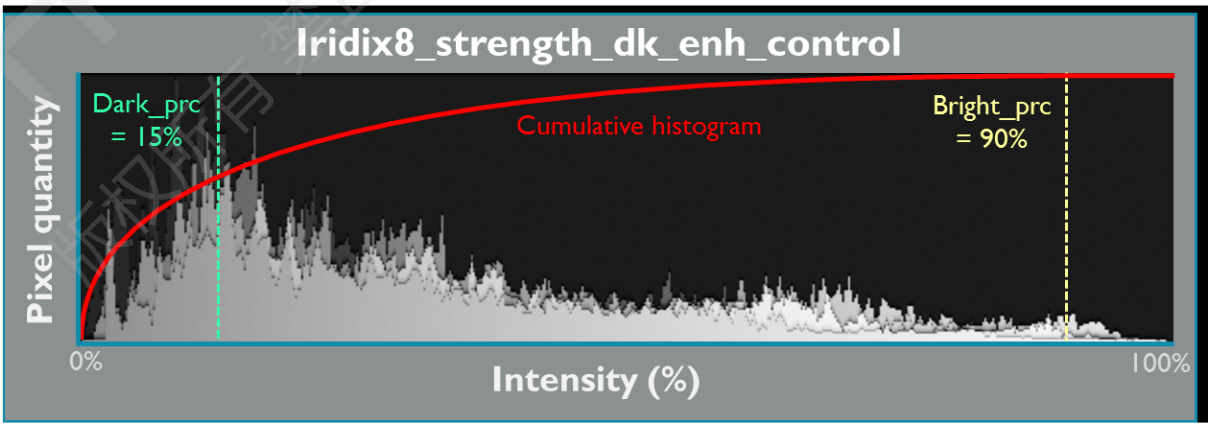


Figure 3.14-2 Cumulative histogram showing Dark_prc and Bright_prc.

With the regions dark and bright regions identified, pD_cut and pH_cut are used to define the respective Dark_prc and Bright_prc in terms of pixel value for the current scene. These two values can then be used to calculate Contrast through the equation shown in Figure 3.14-3.

Note: Print_debug can be used to output the values of pD_cut and pH_cut if required by setting the value to 1. As most interactions with these values are done automatically in firmware, this is not usually necessary.

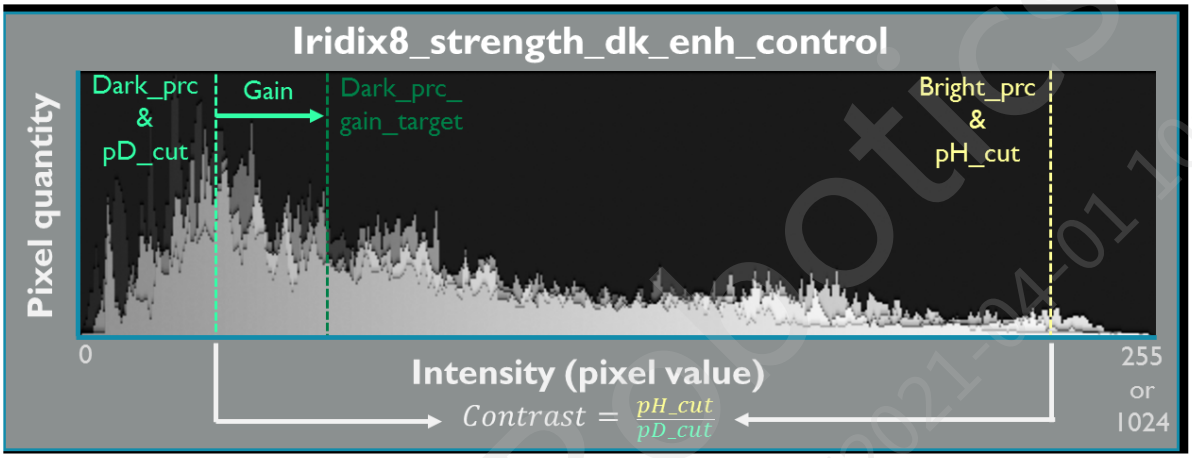


Figure 3.14-3 Typical relation between firmware parameters and histogram.

Additionally, Dark_prc_gain_target is the intended destination for Dark_prc. This value is used alongside pD_cut and Iridix_gain_max to calculate overall iridix Strength where:

$$\text{Iridix gain} = \frac{\texttt{Dark\_prc\_gain\_target}}{\texttt{PD\_cut}}$$

and

$$\texttt{Strength} = \frac{\text{Iridix gain}}{\texttt{Iridix\_gain\_max}}$$

The dynamic range compression performed by iridix will be between the Black_level and White_level set. That is, iridix will not apply any gain to values below or above these two limiters.

For tuning and demonstration purposes of iridix, a Roi_[hor/ver]_[start/end] can be configured by setting the pixel coordinates of the ROI, and setting the corresponding strength for each register.

As for Time_filter, this is usually set to the duration of AE convergence time.

All other parameters listed in table 61 will be described with respect to tuning in the relevant tuning section below.

3.14.3 Iridix Tuning during phase one

Iridix requires no form of tuning or alteration in phase one. However, the iridix module in Calibration Tool needs to be opened and saved (without making any adjustments). This can be done while using Calibration Tool when tuning other modules such as black level and AWB.

3.14.3.1 Iridix window

When you open the iridix® module, a window like Figure 3.14-4 will pop up.

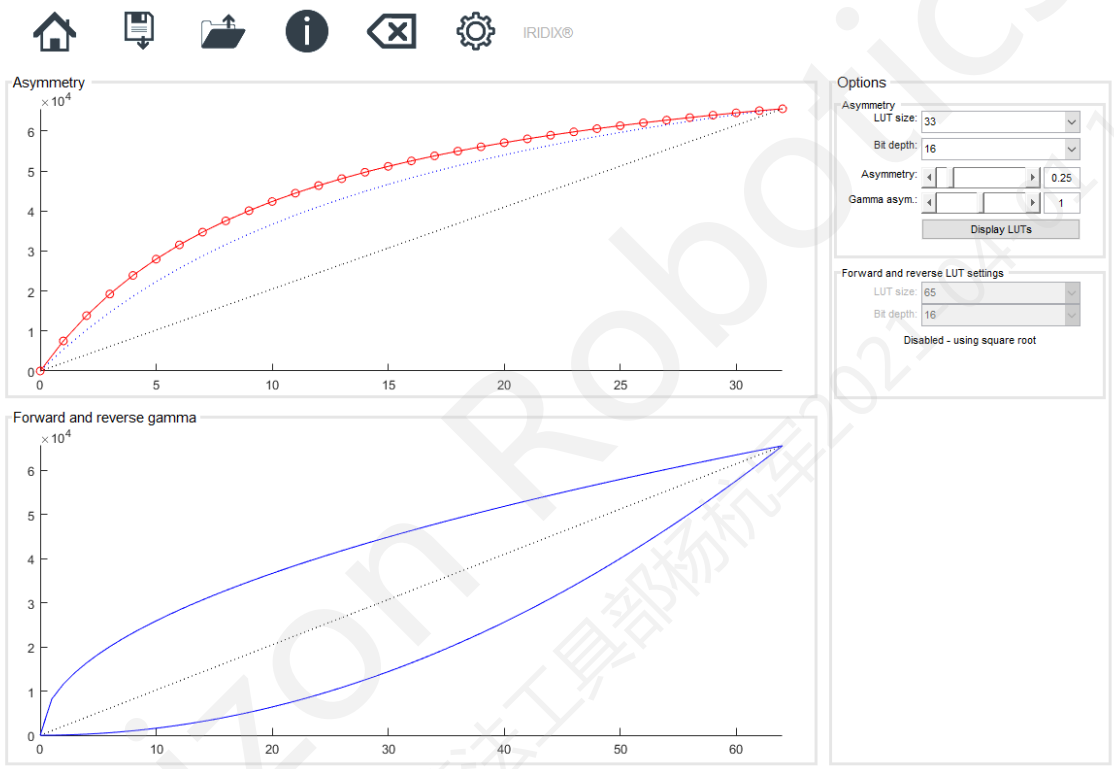


Figure 3.14-4 iridix® window

3.14.3.2 Window features

Table 3.14-4 lists the function of each region.

Region	Sub-region name	Function
Asymmetry	-	Plot of the Asymmetry curve – this is the reference global tone curve which iridix® uses to produce its local mapping
Forward and Reverse	-	Plot of the forward and reverse curves (shape is imported from *gamma & tone* module)

Options	Asymmetry – LUT Size	Set the Asymmetry LUT node count
	Asymmetry – Bit Depth	Set the Asymmetry curve bit depth
	Asymmetry – Asymmetry	Select asymmetry value
	Asymmetry – Gamma Asym	Select a gamma asymmetry value
	REV-FWD LUTs – LUT Size	Set the REV-FWD LUTs node count
	REV-FWD LUTs – Bit Depth	Set the REV-FWD LUTs bit depth

Table 3.14-4 List of all regions and their functions in the iridix® window

Note: Using rev-fwd LUTs is not recommended and may be disabled in your Calibration Tool release. See the ISP tuning guide for more information.

3.14.3.3 Calibration process

1. Open the iridix® calibration module from the home window of the Calibration Tool.
2. Set the "Asymmetry" curve settings in the "Options" menu:
   a. LUT Size – Set the LUT size for the Asymmetry curve, choose between 33 or 65 nodes.
   b. Bit Depth – Set the bit depth value for the Asymmetry curve, choose between 12 and 20 bits.
   c. Adjust the Asymmetry and Gamma Asym values to shape the curve. Start by using the default values, and recalibrate if needed during iridix® tuning. The Assymetry LUT is the global tone curve that iridix® uses as a reference. Making the curve steeper in the dark (close to 0) region will boost contrast in that region.
3. Set the "REV-FWD" curve settings in the "Options" menu:
   a. LUT Size – Set the LUT size for the REV-FWD curve to match your ISP hardware
   b. Bit Depth – Set the bit depth value of the REV-FWD to match your ISP hardware
4. Click the save button to finalize your calibration and close the module.

3.14.4 Iridix fine tuning phase two

As iridix is not used during phase one, tuning starts in phase two after completing the prerequisites listed in Table 3.14-5. Ideally all of the modules in a pipeline will have been previously tuned, in practice however this isn't always the achievable. As a result, iridix is initially tuned in phase two, after noise reduction but should be verified at the end of phase two, before continuing with phase three.

Prerequisite	**Status / value**
Black level	Tuned

Lens shading	Tuned
AWB	Tuned
CCM revisited	Tuned
Sinter	Tuned
Temper	Tuned
Set Iridix_on	1
Gamma	Tuned with respect to AE_target
AE_target	Set with respect to gamma

Table 3.14-5 Iridix phase two prerequisites.

Alteration of firmware parameters requires some subjective interpretation, where greater accuracy can be achieved during phase three of tuning by evaluating a large selection of captured images.

3.14.4.1 Tuning procedure - Asymmetry LUT

With iridix being the most difficult module when it comes to predicting its behavior it could be assumed that this also makes it challenging to tune. However the reality is that using dynamic calibrations provides a fairly simple means of tuning, while accounting for a wide range of scenes. Initial tuning generates the asymmetry curve before adaptation and manipulation through the parameters previously mentioned.

Note: Dynamic range of a digital camera can be described as the ratio of maximum light intensity measurable (at pixel saturation), to the minimum light intensity (above read-out noise). This is a very important definition to remember while tuning iridix because the gain applied by iridix is proportional to the noise boost visibility of the image. Thus the maximum gain that should be applied by iridix is determined by the dynamic range of the camera system.

1. Generation of the asymmetry LUT (Iridix_LUT_Asymmetry) can be done in two ways, the simplest being to use Calibration Tool where a GUI similar to Figure 3.14-5 is presented, allowing easy modification of the generated curve.

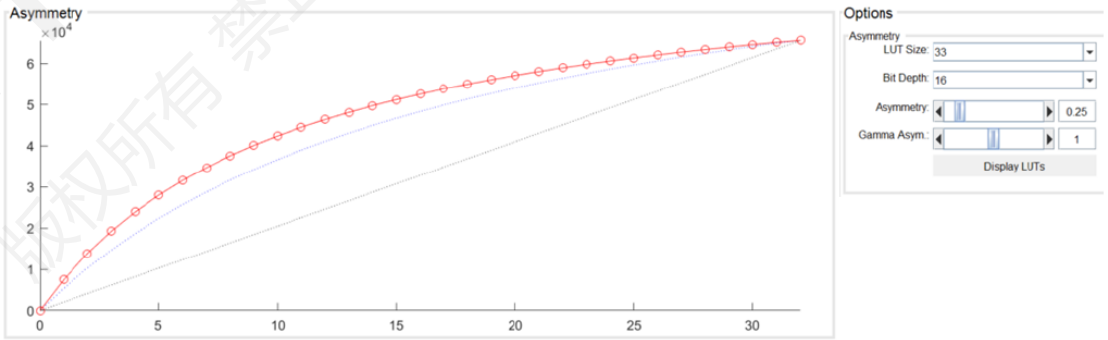


Figure 3.14-5 Iridix GUI.

Note: For more information on generating the asymmetry curve through Calibration Tool, see the documentation provided in the iridix module of the tool.

The other option is to follow the pseudo code described in Figure 3.14-6.

```
function lut = AsymmetryTable(Asymmetry, SecondPole)
x  = (Asymmetry+1)/257 * 2 - 1;
ai = floor(0.5 + 255 * (1- 1/(1000*x*x*x) + x - ((x >= 0).*2)));
ii = (0:32)';
if ai >= 0
   x = ii / 32;
else
   x = (32 - ii) / 32;
end
as = abs(ai ./ 255);
dp = SecondPole ./ 255;
y = round((dp+(1-dp).*((abs(1-dp-x)./dp).^3)) .* (x.*(as+1)./(as+x) ) .* 65535);
y = max(0, min(y, 65535));
if ai < 0
   y = 65535 - y;
end
lut = round(y);
```

Figure 3.14-6 Asymmetry LUT pseudo code.

2. Tune the asymmetry and second pole values according to the desired DRC effect.

   The initial values for asymmetry and second pole are 70 and 250 respectively. Some examples of this are provided in Figure 3.14-7.

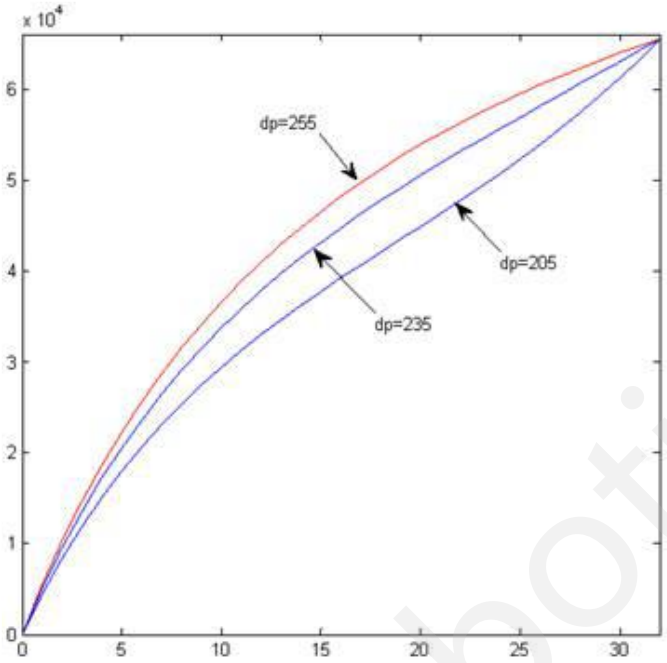


Figure 3.14-7 Asymmetry curve and how (dp) SecondPole can be used.

3.14.4.2 Tuning procedure – Svariance

Svariance specifies the half-width of the Gaussian function as depicted in Figure 3.14-8.

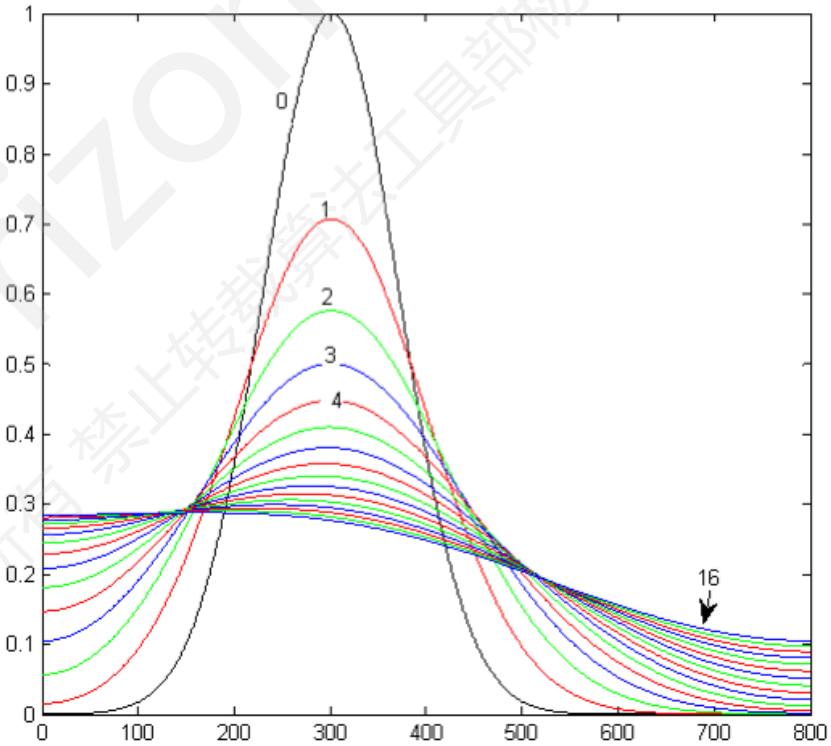


Figure 3.14-8 Variance kernel in one dimension for different variance values for a pixel with horizontal

coordinate 300 and image width 800.

When controlling spatial sensitivity using Svariance, a value of 0 will extend the contrast in the smallest possible regions. However, doing this will lead to over-enhancement and possibly "haloing" artifacts. Higher values of Svariance will restrict the adaptability of the transforms generated by iridix so that the overall compression is closet to the asymmetry curve shape.

Based on image quality testing, the most natural looking images occurs under the full range of conditions when Svariance is set between 4 and 8.

3. Set Svariance (between 4 and 8) according to the desired DRC effect for the system being tuned.

3.14.4.3 Tuning procedure - Dark enhancement

Dark enhancement is intended to adaptively apply gain to dark regions of an image in order to boost the local contrast. The amount of gain applied is dependent on the value of Dark_enh derived from firmware.

While a value can be manually assigned in hardware, iridix 8 uses dynamic parameters to automatically adjust the Dark_enh value depending on scene content. Initially, to allow for the variation of values, Dark_enh is considered in terms of Min_dk and Max_dk. This effectively provides a range of Dark_enh values to work with.

4. Initially set Min_dk to 1,100 and Max_dk to 2,900. Note that these values are in U24.8 format.

   From here there are three main groups of parameters which interact with Min_dk and Max_dk to determine the overall Dark_enh value. These revolve around the intensity of dark pixels, scene contrast and exposure. The tuning procedure listed here for dark enhancement will document all three in the order of interaction with Dark_enh.

5. Start by setting Dark_prc, a recommended starting value is 15% for Dark_prc. It is also advised that if further adjustment is required, this value should remain in the 10-20% range.

   This will allow the generation of pD_cut, defining the intensity of dark pixels in the current scene. pD_cut_min and pD_cut_max can then be set to control Min_dk and Max_dk based on the input pD_cut value as seen in Figure 3.14-9.

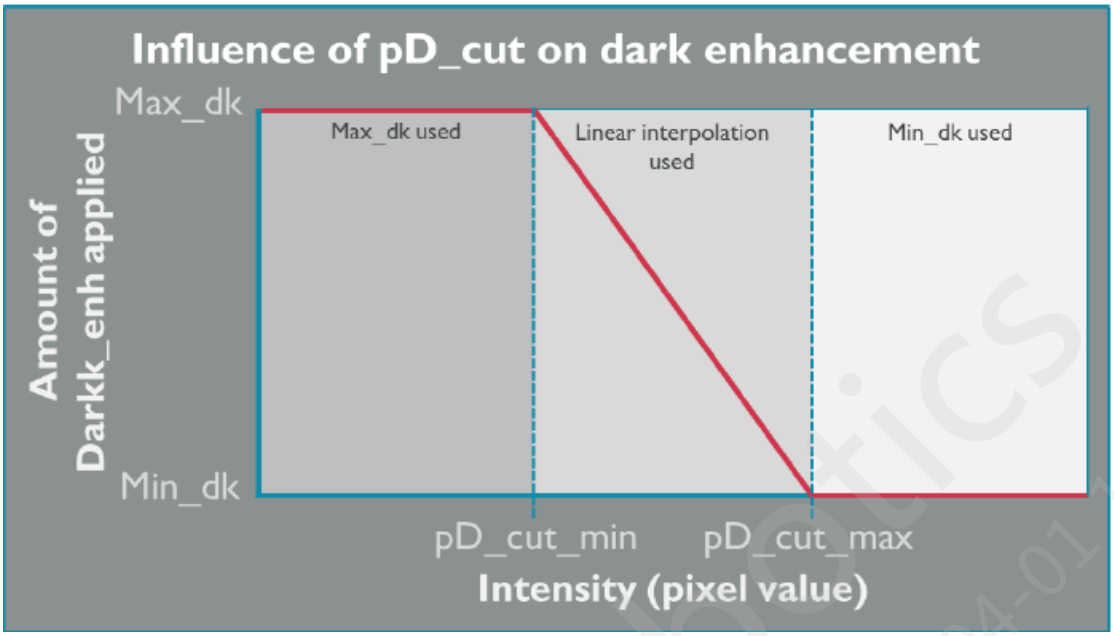


Figure 3.14-9 Relationship between Dark_enh and pD_cut

6. Set pD_cut_min and pD_cut_max to 8 and 30 respectively. If these require some adjustment after testing, increase or decrease the values in a narrow range (+/- 10 pixels). Here a degree of subjective evaluation and a trial and error approach will be required.

   The next group of parameters to interact with Min_dk and Max_dk are based on Contrast of a scene. As previously mentioned along with Figure 3.14-3, Contrast is derived from pD_cut and pH_cut.

7. As Dark_prc is already set, generating pD_cut, set Bright_prc in order to generate pH_cut. A recommended starting value for Bright_prc is 99%, but if further adjustment is required then a value in the range of 90-99% is considered acceptable.

   As scene contrast can now be determined, Dark_contrast_min and Dark_contrast_max can be used to adjust Dark_enh. As seen in Figure 3.14-10, the output dark enhancement value will depend on where the output scene Contrast falls with respect to Dark_contrast_min and Dark_contrast_max over the full contrast range.

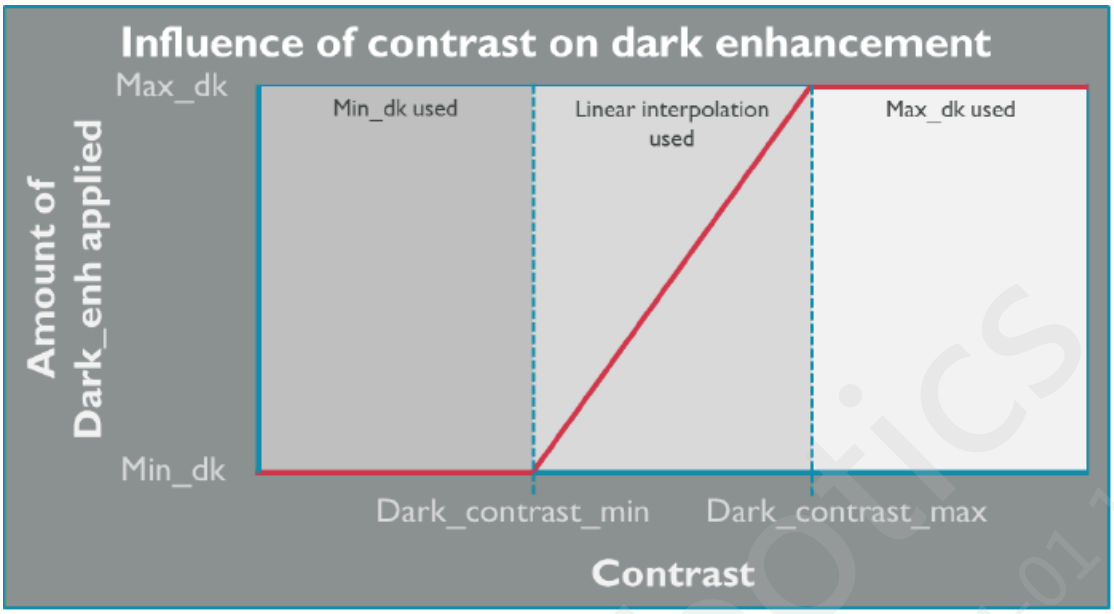


Figure 3.14-10 Relationship between Dark_enh and Contrast.

8. Set Dark_contrast_min and Dark_contrast_max values, note that these are in bit-shift (left) format. Recommended starting values are 16«8 and 60«8 respectively.

   The final group of parameters which are responsible for automatic calculation of Dark_enh are related to exposure value, these are tuned during phase three.

3.14.4.4 Tuning procedure – Strength

Strength is considered as the main register (aside from Asymmetry) influencing the amount of iridix adjustment to apply to a scene. The amount of compression to be applied to the image is controlled through the Strength_inroi and Strength_outroi registers, where higher values will lead to higher differential gains between shadows and highlights.

When Strength is set to 0, all curves are linear for any image, providing equal input/output values. As Strength is increased, the variation between curves in shadows, mid-tones and highlights also increases, thus providing larger dynamic range compression.

Similarly to Dark_enh, iridix 8 provides an automatic means of altering the Strength value based on scene content, using dynamic parameters. To allow for the variation in value, Strength is considered in terms of Min_str and Max_str, this effectively provides a range of Strength values

9. Set Min_str and Max_str. In the majority of scenarios Min_str and Max_str can remain at 0 and 100 respectively. Doing this provides iridix with the functionality to range from disabled (as at 0 asymmetry becomes linear so input/output remains equal) through to the full application of gain (at 100).

   As well as the equation mentioned in (3.14.2), Strength is also affected by Contrast (tuned in phase two) and exposure value (tuned in phase three). These processes mimic those used for dynamic calculation of Dark_enh, where the calculated Contrast value for a scene is evaluated against Contrast_min and Contrast_max. See Figure 3.14-11 below:

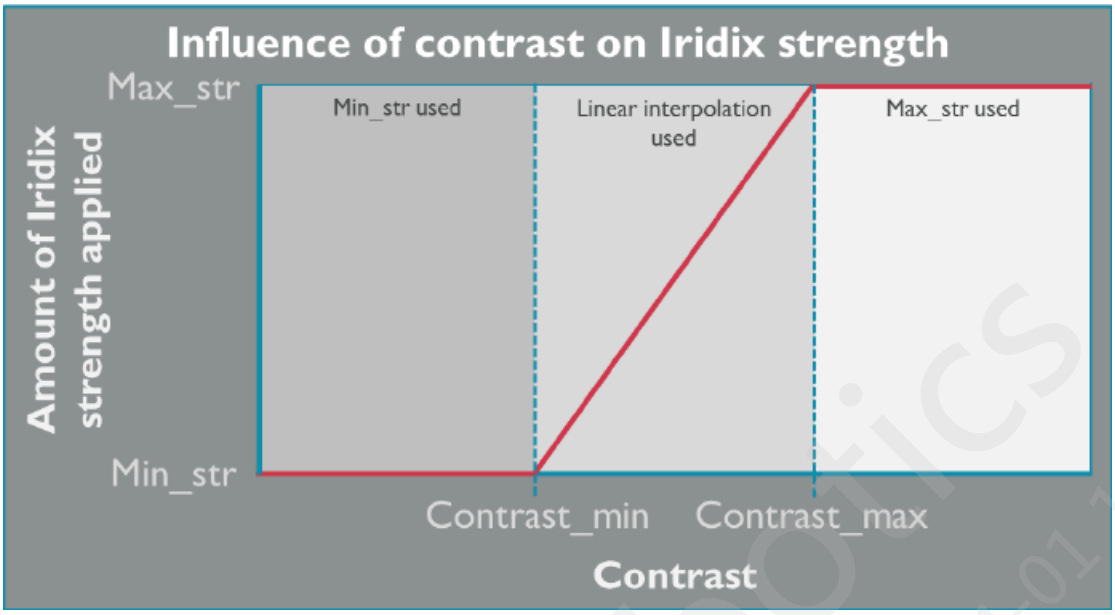


Figure 3.14-11 Relationship between Strength and Contrast.

10. Once again bit-shift is used to define values for Contrast_min and Contrast_max. Respectively set these to 20«8 and 50«8.

3.14.4.5 Tuning procedure - Bright preservation

In a situation where iridix would normally cause highlight clipping, bright preservation will limit iridix's ability to increase gain. A value of 255 allows for application of full gain on highlights - as this is decreased, the amount of gain applied is reduced. Meaning, as the value is reduced, iridix will reduce tonal adjustments on bright pixels.

11. Bright preservation values can only be set in hardware using the Bright_pr register. Start with this set at 255 and reduce the value as required during testing to preserve highlights.

3.14.5 Iridix fine tuning phase three

After completing the prerequisites listed in Table 3.14-6, final adjustments to iridix can be made. Once again, it is worth revisiting iridix at the end of phase three tuning to ensure correct performance.

Prerequisite	**Status / value**
Phase two prerequisites	Complete
Sinter	Tuned for all gains
Temper	Tuned for all gains
Demosaic	Tuned for all gains

Table 3.14-6 Iridix phase three prerequisites

3.14.5.1 Tuning procedure – Strength

By default iridix will estimate the maximum amount of dynamic range compression to be applied to a given image, even though this may exceed the maximum dynamic range of the system. In these cases, noise will be revealed and/or the overall intensity of the image will be overexposed.

Accounting for this, Strength of iridix effect should be calculated as follows:

$$\text{Iridx strength} = \max\left(0, \min\left(\frac{\text{Gain} - 1}{\text{Gain}_{\max} - 1}, f(\text{EV})\right)\right) * 255$$

Where gain is the desired gain to be applied to the image calculated in FW. This gain is controlled by the maximum gain parameter (Max_iridix_gain) and f(EV) which is the function that determines the maximum iridix strength in relation to noise. f(EV) function is characterized as follows (Figure 3.14-12):

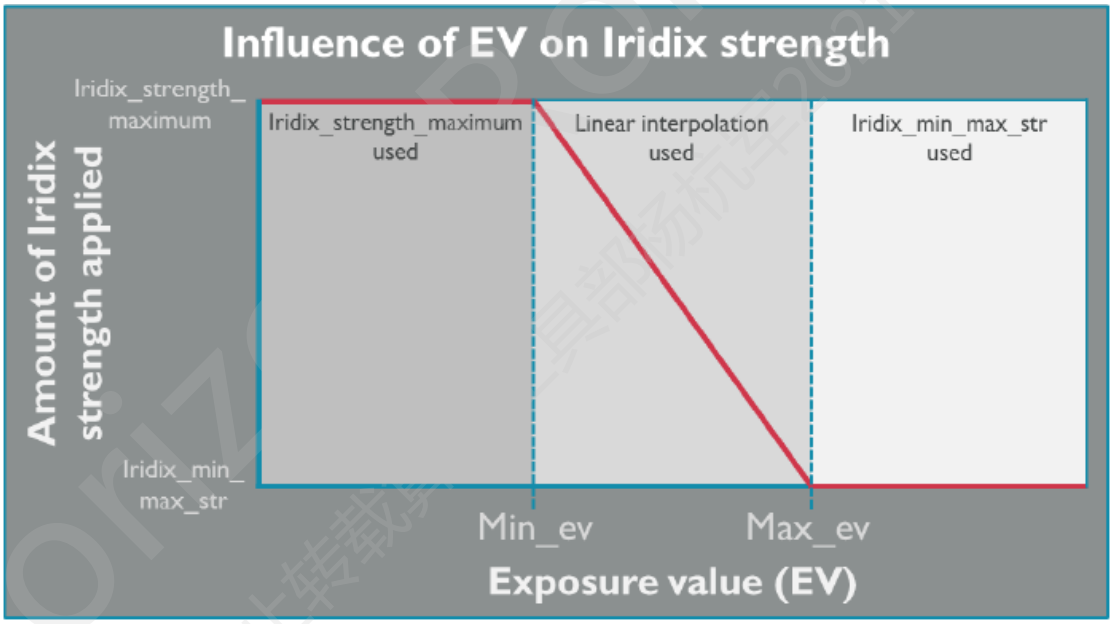


Figure 3.14-12 Iridix strength in relation to exposure value.

1. Set Iridix_ev_lim_full_str and Iridix_ev_lim_no_str, typical values for these are 1,000,000 and 3,330,000 respectively - defined in terms of EV_log2. If adjustments are required to suits specific scenes during fine tuning, these values should be adjusted based on the derived EV_log2 value in the .txt file of the under-performing scene.
2. Also set Min_str_dk_ev and Max_str_dk_ev to 0 and 255, defining the full possible Strength range. The advantage here is that altering the minimum and maximum Strength value is independent of the Min_str and Max_str parameters used for automatic Contrast control.

3.14.5.2 Tuning procedure - Dark enhancement

As previously mentioned, Dark_enh is also modulated with respect to EV. Figure 3.14-13 demonstrates how the two are related in terms of Min_dk, Max_dk, Iridix_ev_lim_no_str and Iridix_ev_lim_full_str.

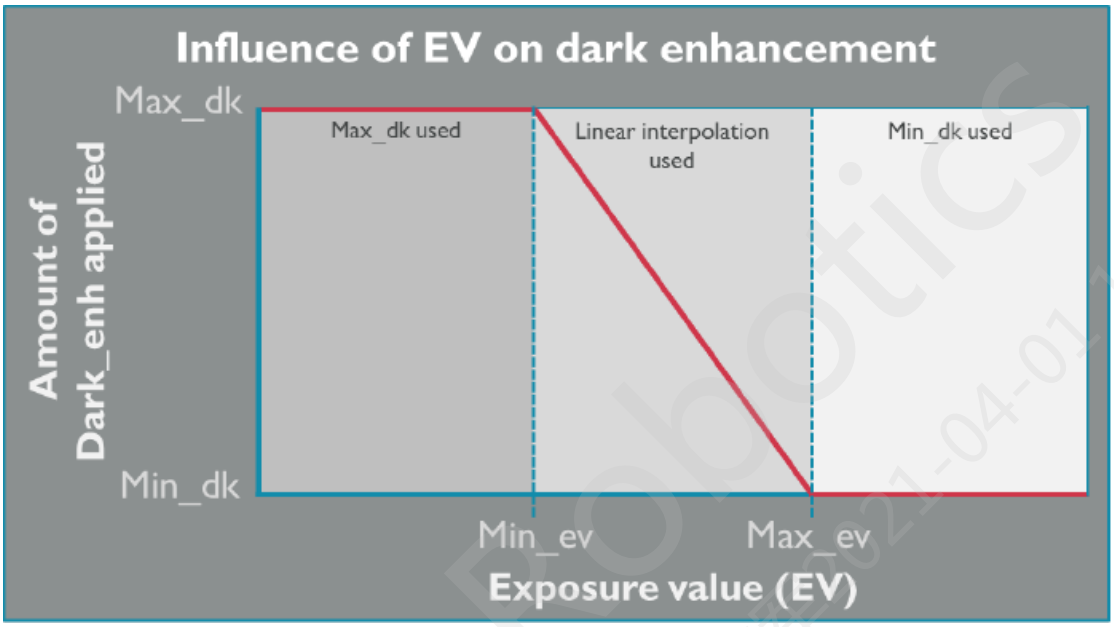


Figure 3.14-13 Dark enhancement strength in relation to exposure value.

For scenes where EV is calculated to be below Iridix_ev_lim_no_str, the value assigned to Max_dk is used. Where the derived EV is above Iridix_ev_lim_full_str, Dark_enh will be limited to Min_dk.

By this stage, all of the parameters required for EV to control Dark_enh should have previously been tuned, thus no further tuning is required.

3.15 Demosaic (RGGB)

3.15.1 Overview

Demosaic is the fundamental source of image color, providing a means of obtaining RGB values for each individual pixel. Here, interpolation of color channels is done with respect to direction to reduce the artifacts introduced, along with the application of sharpening with respect to spatial frequency.

- *During phase 1* - No tuning required.
- *During phase 2* - Demosaic blend, gradient detection and false color parameters tuned in lab conditions for the majority of scenes.
- *During phase 3* - Demosaic parameters are fine tuned to suit all scenes. Primarily involving tuning sharpening and modulating parameters according to gain.

3.15.2 Tuning Theory

RAW pixel values are interpreted with respect to the Bayer array in order to achieve output values in the RGB color space. This uses isotropic and anisotropic non-linear color interpolation algorithms to interpolate the detection of gradients. Edge-adaptive sharpening is incorporated with this, minimizing amplification of high frequency noise and false color suppression.

The main objective of demosaic tuning is to maximize both isotropic and anisotropic gradient detection. However, in order to do this, tuning is applied separately for each gradient detection, where there are three main groupings of parameters responsible for specific directions. Blend_VH is used to denote the application of vertical and horizontal interpolation, blend_AA is used to denote 45◦ and 135◦ interpolation and blend_VA refers to the combination of these four directions. blend_UU on the other hand is treated in a similar fashion, but is more concerned with the blend between isotropic and anisotropic interpolation. All four of these are defined by individual sets of hardware registers.

Key hardware registers

Hardware registers are found on the relevant *source* page listed in the "Hardware" tab of *Control Tool*.

Register name	Source	Description
[VH/AA/VA/UU]_slope	Demosaic	Edge detection slope control
[VH/AA/VA/UU]_thresh	Demosaic	Edge detection threshold control
[VH/AA/VA/UU]_offset	Demosaic	Edge detection offset
Dmsc_config	Demosaic	Enable/disable a range of blending masks
FC_slope	Demosaic	False color slope control
FC_alias_slope	Demosaic	Aliasing slope control
FC_alias_thresh	Demosaic	Aliasing threshold control
Sharp_alt_ld	Demosaic	High frequency strength parameter
Sharp_alt_ldu	Demosaic	Medium frequency strength parameter
Sharp_alt_lu	Demosaic	Low frequency strength parameter
Sad_amp	Demosaic	Blend between Sharp_alt_ld and Sharp_alt_ldu
Sat_slope	Demosaic	Saturation blending slope control
Sat_thresh	Demosaic	Saturation blending threshold control
Sat_offset	Demosaic	Saturation blending offset control
Lum_thresh	Demosaic	Luminance threshold for directional sharpening
AC_slope	Demosaic	AC slope control
AC_thresh	Demosaic	AC threshold control
AC_offset	Demosaic	AC offset control
Sharpen_alg_select	Demosaic	Enable/disable new sharpening logic
Min_d_strength	Demosaic	Minimum threshold for directional sharpening
Min_ud_strength	Demosaic	Minimum threshold for un-directional sharpening
Weight_lut	Demosaic	Noise profile LUT
NP_offset	Demosaic	Noise profile offset
grey_det_thresh	Demosaic	gray detection threshold control

grey_det_slope	Demosaic	gray detection slope control
lg_det_thresh	Demosaic	low green detection threshold control
lg_det_slope	Demosaic	low green detection slope control
luma_thresh_ low/high_d/ud	Demosaic	directional/undirectional sharpening luma thresh control
luma_slope_low/high_ d/ud	Demosaic	directional/undirectional sharpening luma slope control
luma_offset_ low/high_d/ud	Demosaic	directional/undirectional sharpening luma offset control

Table 3.15-1 Demosaic hardware registers.

Key firmware parameters for dynamic calibrations

Dynamic calibration parameters are found both in the "dynamic_calibrations.c" file, and on the relevant page listed in the "Dynamic Calibrations" tab of *Control Tool*.

Parameter name	Description
Sharp_alt_d	High frequency strength modulation
Sharp_alt_du	Medium frequency strength modulation
Sharp_alt_ud	Low frequency strength modulation
Demosaic_np_offset	Noise profile offset

Table 3.15-2 Demosaic dynamic calibration parameters.

Note: Unlike most dynamic parameters, Sharp_alt_du is only accessible through the dynamic calibration file. As such, is not found in Control Tool.

Furthering this, individual blending masks for each of the directional/blend components are provided by Dmsc_config, where:

• Dmsc_config = 4 - Shows the blending mask of UU at output. In the illustration presented, all detected edges are shown as white, while all flat areas appear black.

• Dmsc_config = 17 - Shows the blending mask of AA at output. In the illustration presented, 135◦ edges are shown as white, 45◦ edges appear black, while flat areas are designated as mid-gray.
• Dmsc_config = 18 - Shows the blending mask of VA at output. In the illustration presented, 45◦ and 135◦ edges are shown as white, vertical and horizontal edges appear black, while flat areas are designated as mid-gray.
• Dmsc_config = 19 - Shows the blending mask of VH at output. In the illustration presented, vertical edges are shown as white, while horizontal edges appear as black.
• Dmsc_config = 27 - Shows the blending mask of high frequency components (gray/white) against low frequency components (black) for blending through Sad_amp.

• Dmsc_config = 31 - Shows the mask of gray detection.
• Dmsc_config = 32 - Shows the mask of low green detection.

It can be very easy to unwillingly introduce artifacts into an image during demosaic, as any form of interpolation tends to leave some form of residue. Using the relevant blending mask while tuning

should help to avoid this problem, but care should still be taken. Figure 3.15-1 portrays the process applied by demosaic registers from input to output. VH and AA algorithms are employed to detect edges. These are combined under VA algorithms for analysis of all detections. Interpolation is done with respect to these edges and false color correction is applied. This is taken, along with the interpolation of low frequencies, as an input for sharpening. High, medium and low frequencies are independently considered before blending high and medium frequencies. The output blend is then balanced against low frequency and edge detection results to optimize the final output RGB image.

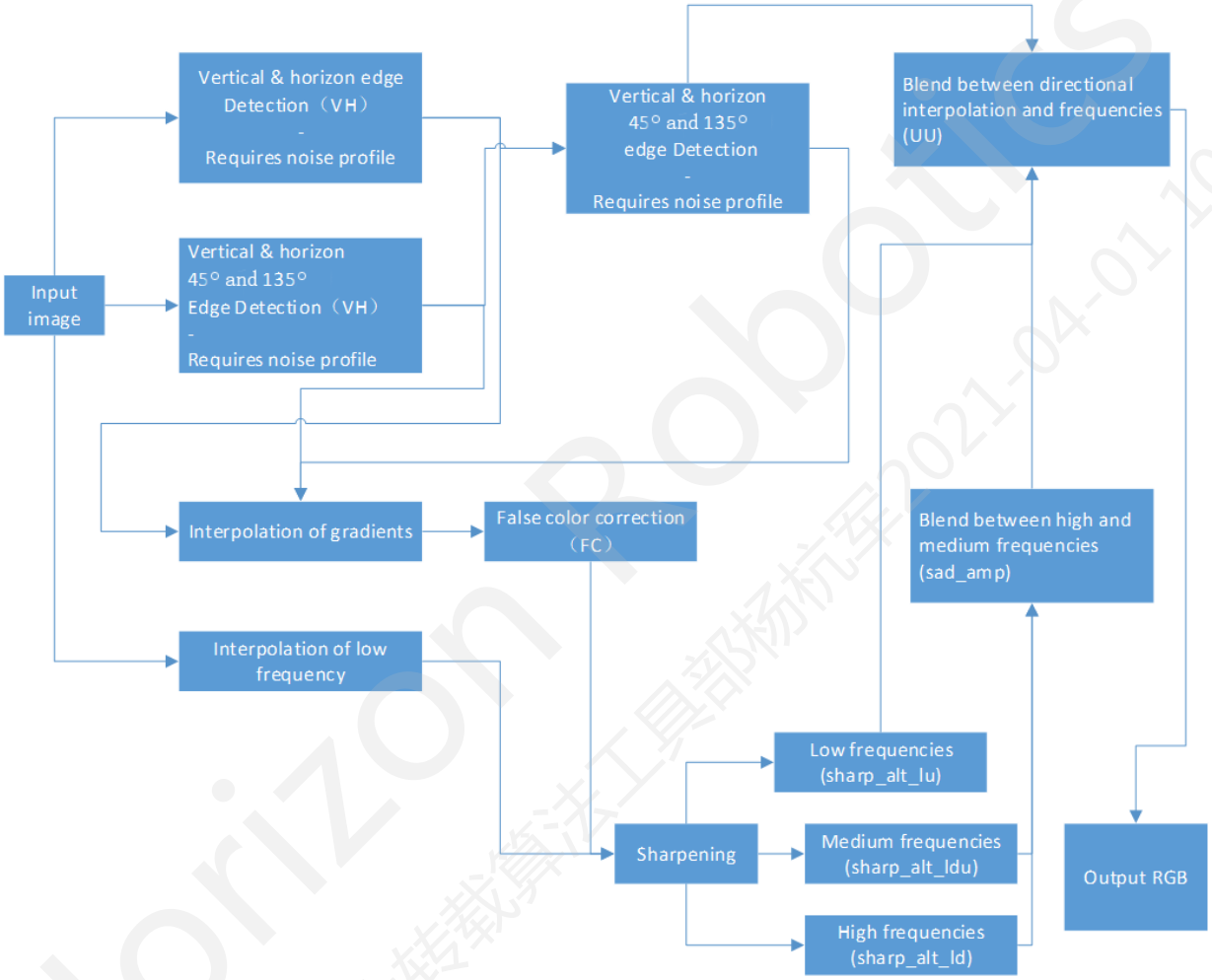


Figure 3.15-1 Relationship of key demosaic hardware registers

3.15.3 Tuning during Demosaic phase two

As initial tuning uses *.RAW* images, the demosaic module is left untouched until phase two. Tuning can only start after completing the prerequisites listed in Table 3.15-3 below:

Prerequisite	**Status / value**
Black level	Tuned
Noise profile	Derived
Green equalization	Tuned

Defective pixel correction	Tuned
Sinter	Tuned and enabled
Temper	Tuned and enabled
RGB sharpening	Disabled (if applicable)

Table 3.15-3 Demosaic prerequisites.

Set the default demosaic values listed in Table 3.15-4 below:

Parameter	Value
Sat_slope	128
Sat_thresh	369
Sat_offset	0
Lum_thresh	150
Dmsc_config	0
AC_thresh	435
AC_slope	207
AC_offset	0
FC_slope	0
FC_alias_slope	255
FC_alias_thresh	0
Sharpen_alg_select	1
Sharp_alt_ld	16
Sharp_alt_ldu	16
Sharp_alt_lu	16
Sad_amp	16
Min_d_strength	7987
Min_ud_strength	7987
Luma_thresh_low_d	100
Luma_slope_low_d	16384
Luma_offset_low_d	0
Luma_thresh_high_d	4000
Luma_slope_high_d	16384
Luma_offset_high_d	0

Table 3.15-4 Default demosaic values.

In dynamic_calibrations.c set:

- const modulation_entry_t Demosaic_np_offset_table[]=0,0;.
- const modulation_entry_t Sharp_alt_d_table[]=0,5;.
- const modulation_entry_t Sharp_alt_du_table[]=0,0;.
- const modulation_entry_t Sharp_alt_ud_table[]=0,0;.
- Ensure NP LUT for demosaic is set.

Tuning procedure - Gradient detection

Figure 3.15-2 shows the tuning process for gradient detection, described in more detail

throughout steps 1-8.

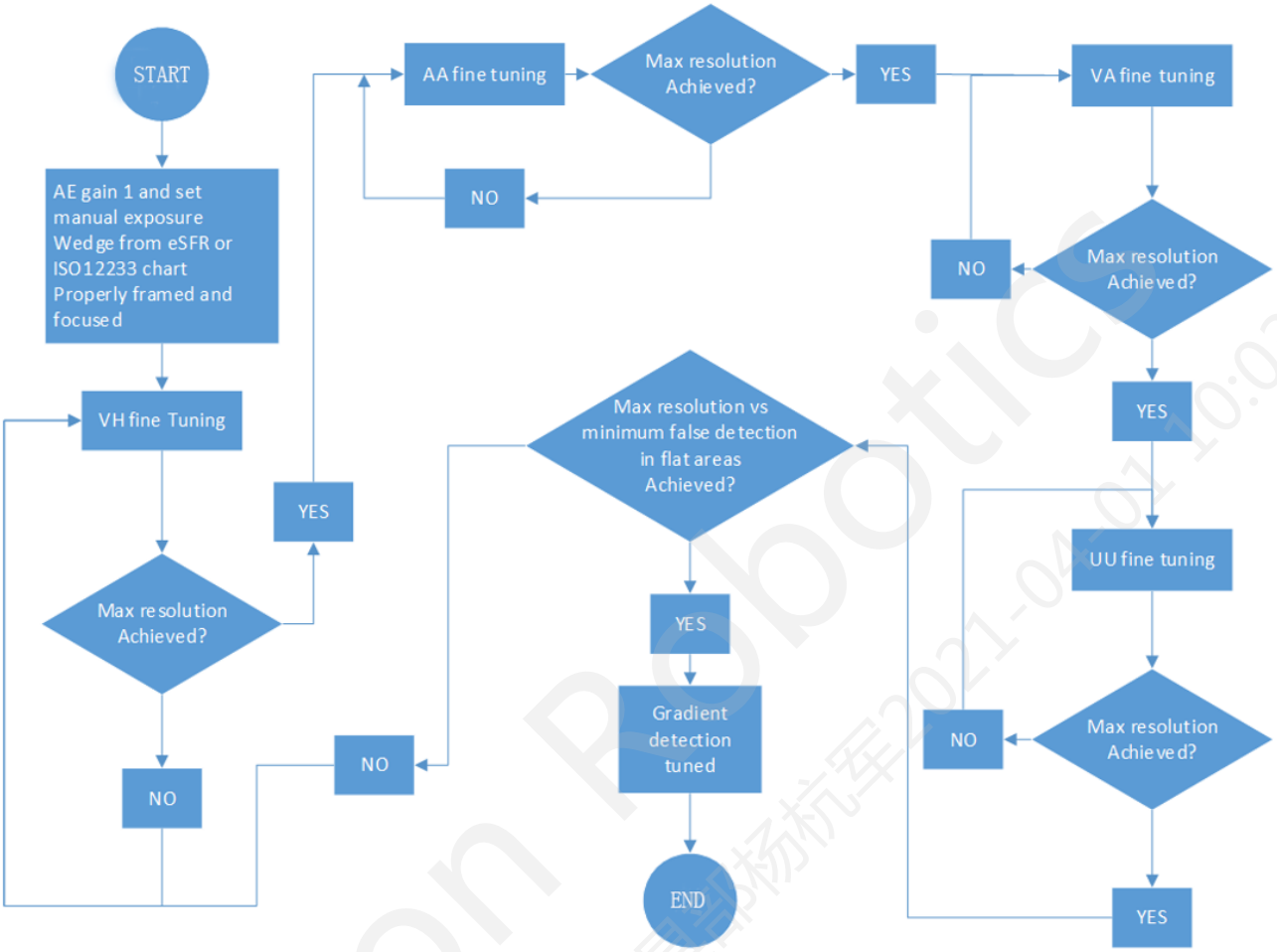


Figure 3.15-2 Gradient detection process.

1. Start by setting up a resolution chart, such as the ISO12233 depicted in Figure 3.15-3 Correct resolution chart framing for: (a) wide angle lens barrel distortion, (b) no lens distortion., making sure to keep the framing markers even and in frame.

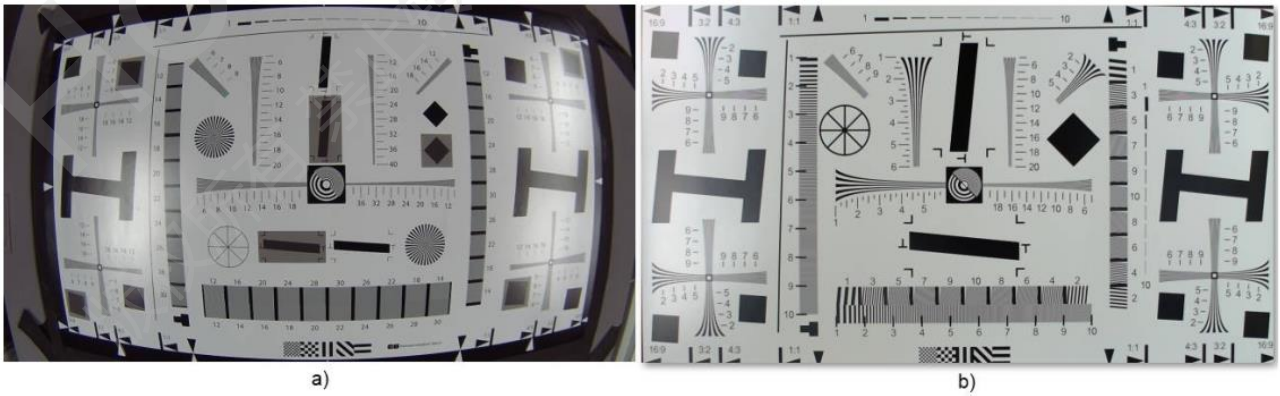


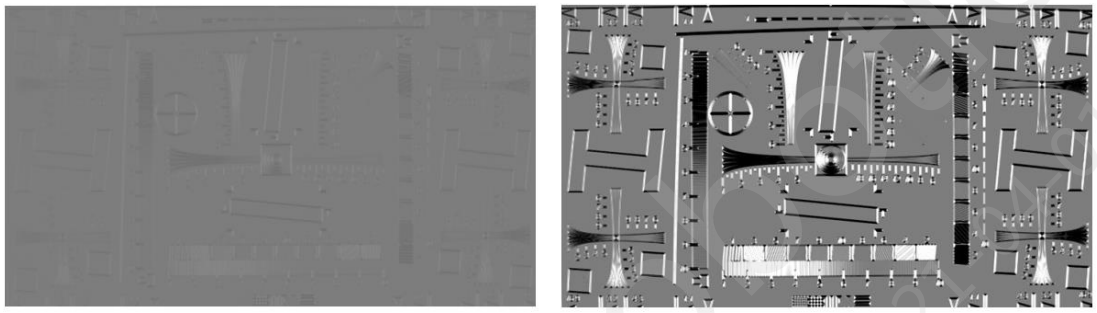
Figure 3.15-3 Correct resolution chart framing for: (a) wide angle lens barrel distortion, (b) no

lens distortion.

2. To start tuning the *blend_VH* using the mask (see Figure 3.15-4), set the values in Table 3.15-5 below:

Parameter	Value
VH_slope	128
VH_thresh	0
VH_offset	2048
Dmsc_config	19

Table 3.15-5 Default *blend_VH* values.



(a) Initial VH mask. **(b) Tuned VH mask**

Figure 3.15-4 VH blend mask, before and after tuning.

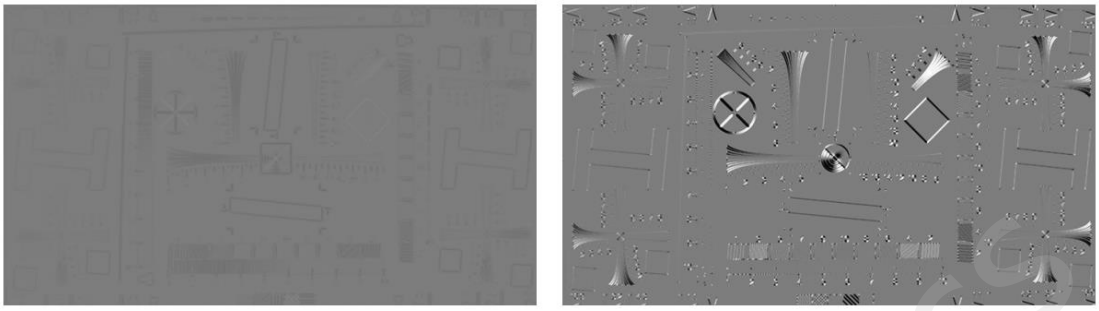
3. Focusing particularly on the vertical and horizontal lines in the chart, increase VH_slope to either just before, or equal to the Nyquist frequency. A value in the region of *170-190* is often acceptable.

4. Now adjust VH_threshold to a point where false detection of angular lines such as 45◦ does not occur, removing them from view. Leave VH_offset at it's default value. It is likely that some noise will have been detected, so there will be a trade-off between
resolution, detection of angular lines and noise when tuning this register. If 45◦ or 135◦ lines are visible, there is a high probability that aliasing artifacts will appear in the image. Refer back to Figure 3.15-4b for an idea of what a resolution chart looks like with *blend_VH* tuned.

5. Once *blend_VH* is optimal, repetition of steps 3-4 should be followed for *blend_AA*, this time aiming to achieve the blending masks shown in Figure 3.15-155 using the starting values in Table 3.15-6 below:

Parameter	Value
AA_slope	128
AA_thresh	0
AA_offset	2048
Dmsc_config	17

Table 3.15-6 Default *blend_AA* values.



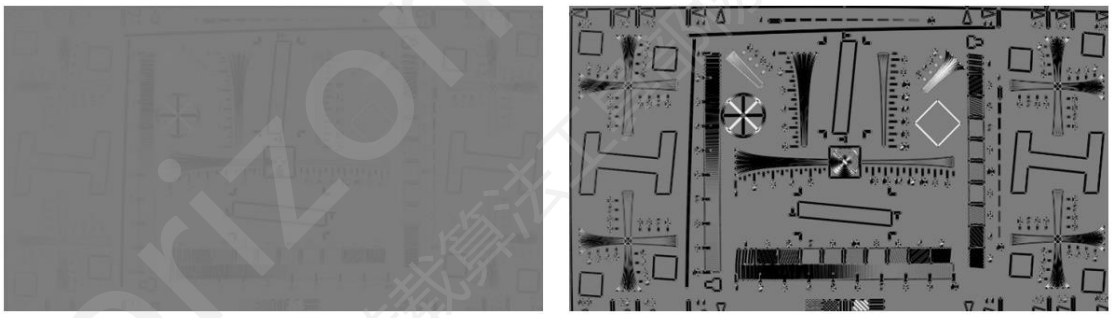
(a) Initial AA mask. (b) Tuned AA mask.

Figure 3.15-5 AA blend mask, before and after tuning.

6. Once *blend_AA* is optimal, repetition of steps 2-4 should be followed for *blend_VA*, this time aiming to achieve the blending masks shown in Figure 3.15-6 using the starting values in Table 3.15-7 below:

Parameter	Value
VA_slope	128
VA_thresh	0
VA_offset	2048
Dmsc_config	18

Table 3.15-7 Default *blend_VA* values.



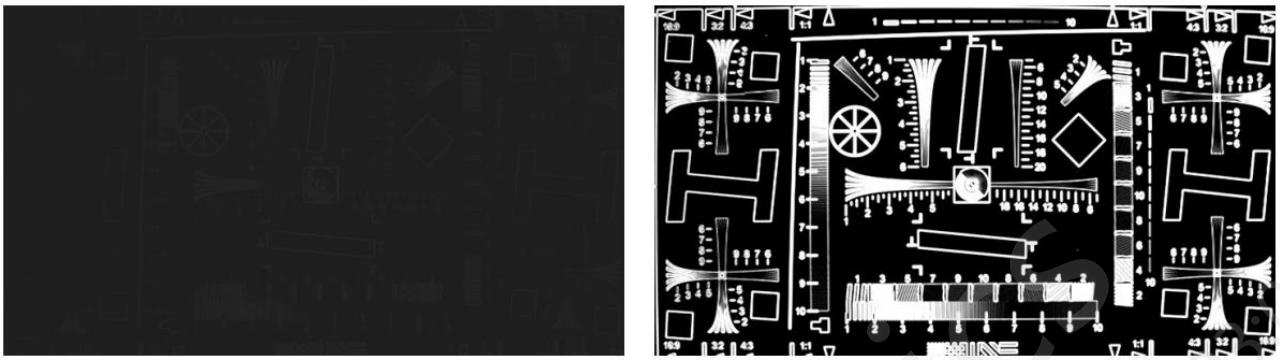
(a) Initial VA mask. (b) Tuned VA mask.

Figure 3.15-6 VA blend mask, before and after tuning.

7. Once *blend_VA* is optimal, repetition of steps 2-4 should be followed for *blend_UU*, this time aiming to achieve the blending masks shown in Figure 3.15-7 using the starting values in Table 3.15-8 below:

Parameter	Value
UU_slope	128
UU_thresh	0
UU_offset	2048
Dmsc_config	4

Table 3.15-8 Default blend_AA values.



(a) Initial UU mask. (b) Tuned UU mask.

Figure 3.15-7 UU blend mask, before and after tuning.

8. To view the final image and analyze the applied calibrations, set:

• Dmsc_config = 0. When tuning meets the criteria, record all settings and set them in the custom_initialization function.

Tuning procedure - False colour

Hardware parameters controlling false color correction are the next set to be tuned. False color correction is responsible for removing the Moir épattern caused by frequencies reater than the Nyquist frequency of the imaging system.

9. To start tuning false color, which should initially look something similar to Figure 3.15-8, set the values in Table 3.15-9 below:

Parameter	Value
FC_alias_slope	255
FC_alias_thresh	0
FC_slope	0
Dmsc_config	0

Table 3.15-9 Default blend_FC values.

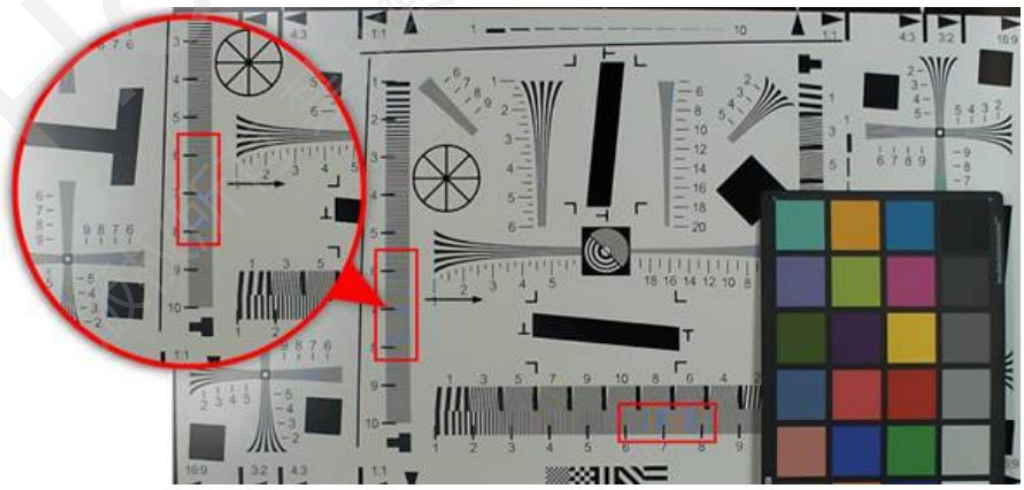


Figure 3.15-8 Example of an ISO12233 with initial false color settings. Note the visible Moiré pattern.

10. Increase FC_slope to the point where false color is reduced/removed and artifacts start to appear in the ColorChecker chart patches, seen in Figure 3.15-9. Evidence of color fading is also apparent in this figure, when this occurs stop increasing FC_slope. Typical values are between *110* and *160*.

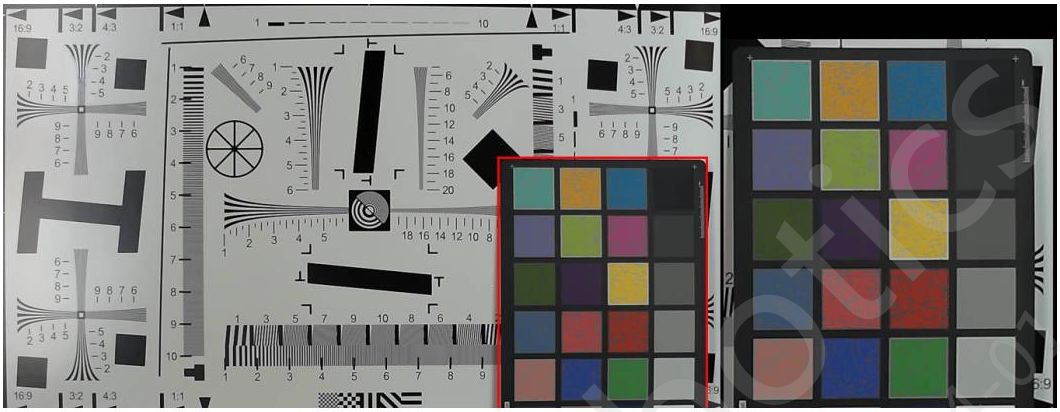


Figure 3.15-9 Example of reduced false color, ColorChecker artifacts and color fade.

11. Now decrease FC_alias_slope to the point where aliasing is removed and previously visible ColorChecker artifacts are no more, as demonstrated in Figure 3.15-10. Typical values are between 60 and 100.

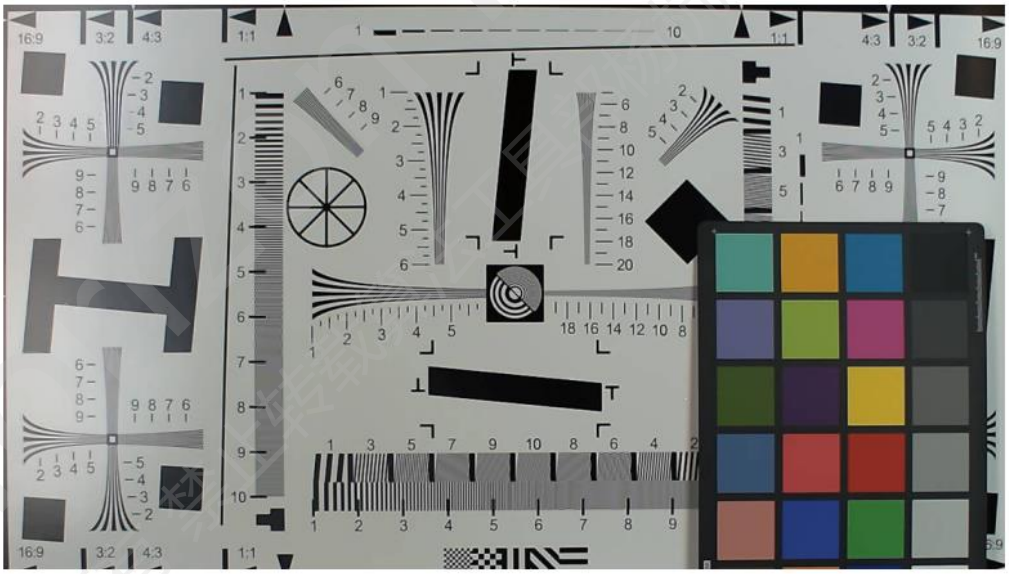


Figure 3.15-10 Example of tuned false color correction.

Tuning procedure - Gray detection

Hardware parameters controlling gray color detection. Grey detection is part of the function to enhance very high frequency detail interpolation in demosaic.

12. This step to achieve proper setting for gray detection, using the starting values in Table 3.15-10 below:

Parameter	Value
grey_det_thresh	0
grey_det_slope	65535
Dmsc_config	31

Table 3.15-10 initial gray detection values.

Increase the grey_det_thresh value till the patch 22(18% gray patch) is totally white which means it is detected as gray. Decrease the grey_det_slope till the patch 22 is like the figure gray detection tuned mask, and all the gray patches brighter than patch 22 are black.

(a) gray detection init mask. (b) gray detection tuned mask.

Figure 3.15-11 Gray detection tuning.

Tuning procedure - Low green detection

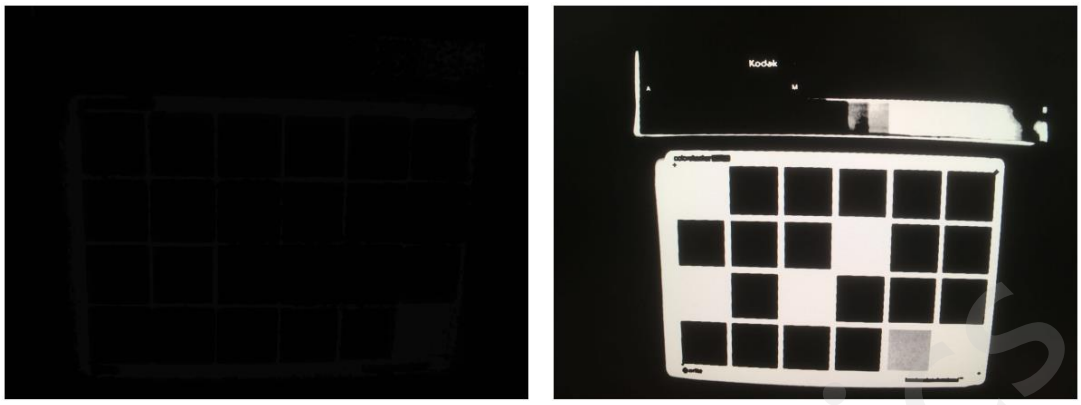
Hardware parameters controlling low green detection. Low green detection is part of the function to reproduce fine detail in low green edge without artefact.

13. This step to achieve proper setting for low green detection, using the starting values in Table 3.15-11 below:

Parameter	Value
lg_det_thresh	8
lg_det_slope	65535
Dmsc_config	32

Table 3.15-11 initial low green detection values.

Increase the lg_det_thresh till the patch 13(blue) and patch 15(red) are totally white which means they are detected as low green. Decrease the lg_det_slope till the patch 23 is like the figure gray detection tuned mask, and all the gray patches brighter than patch 23 are black.



(a) low green detection init mask. (b) low green detection tuned mask.

Figure 3.15-12 low green detection tuning.

Tuning process - Sharpening

The sharpening block found in the demosaic module is edge dependent, essentially leaving the flat areas untouched while only sharpening edges. The aim here is to ensure the right trade-off between image sharpness and artifacts. Demosaic sharpening is often applied to high contrast data in order to increase perceived image sharpness.

Note: Although this may improve measured objective performance, it could result in the degradation of subjective image quality. Often this is in the form of artifacts caused by oversharpening such as "haloing" and noise.

MTF measurements are conducted in Imatest during the tuning process to certify that, while subjective image quality may be acceptable, resolution specifications are still being met. Decision logic for frequency components are controlled through two parameters, Sad_amp and *blend_UU*:

• Sad_amp - While responsible for blending of medium and high frequency components, setting a value of *0* for Sad_amp will bias the decision towards medium frequency. A value of *16* leaves the balance between the two untouched, but beyond this value a bias is applied to high frequency components.

• *Blend_UU* - Blend high/medium with low frequencies. Setting Dmsc_config=27 displays the frequency component classification mask to help visualize the frequency biasing process.

Note: If you need to set different sharpening strengths for texture and edges, use Sharp_alt_ld**,** Sharp_alt_ldu ***and*** Sharp_alt_lu ***to alter respective high, medium and low frequencies.*** Sharp_alt_ldu ***can help avoiding worming artifacts while improving texture.***

An overview of methodology for sharpness tuning is provided in Figure 3.15-13 and described further throughout steps 12-15. This process requires a great deal of objective and subjective assessment during performance testing. Tuning should be performed using a large number of images from various lighting conditions and scenes, starting at the minimum gain before repeating for each available

system gain.

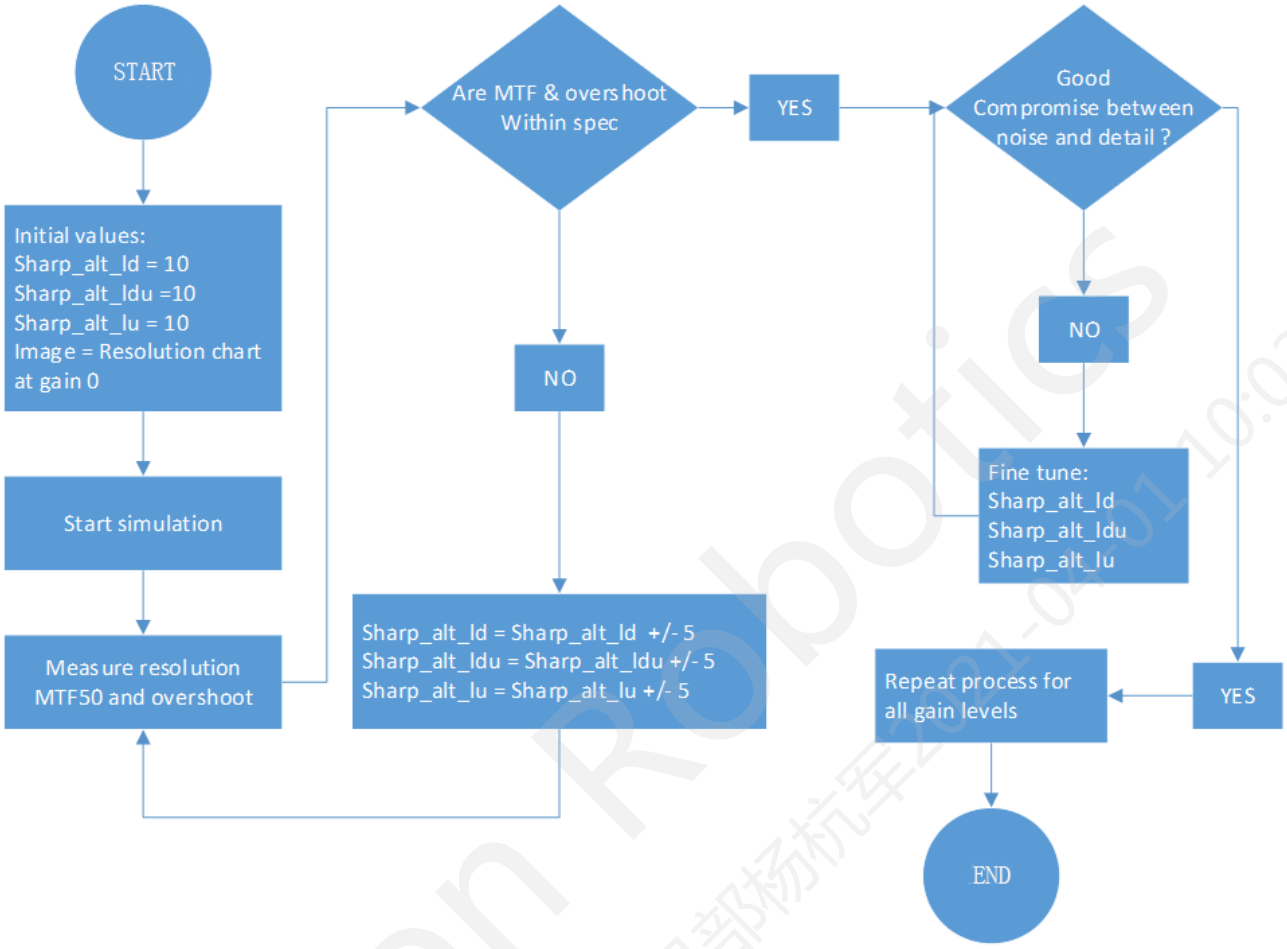


Figure 3.15-13 Sharpness tuning process.

Sharp_alt_ld, Sharp_alt_ldu and Sharp_alt_lu parameters can be modified in different ways:

- Demosaic page: freeze FW before modifying sharpening parameters on the Demosaic *page*.
- Modulation tables: modify the tables as needed. However, FW will need to be recombined every time new values need to be tested.

14. Set the default calibration values seen in Table 3.15-12, found on the Demosaic *page*.

Parameter	Value
Sharp_alt_ld	10
Sharp_alt_ldu	10
Sharp_alt_lu	5

Table 3.15-12 Default Sharp_alt_[...] values.

15. Correctly set up and frame a resolution chart as done previously (see Figure 3.15-3), adjusting the lighting so that total gain (Gain_log2) is 1. Then capture an image.

16. Open Imatest and run the image through the SFR module. Analyze the resulting “Edge profile”

and “SFR (MTF)” plots for MTF50 and overshoot values.
17. If unacceptable, adjust Sharp_alt_ld, Sharp_alt_ldu and Sharp_alt_lu by +/-5, analyze the result and repeat until requirements are fulfilled.

The luma_thresh_low/high_d/ud, luma_slope_low/high_d/ud, and luma_offset_low/high_d/ud are parameters for sharpening control according to intensity. This function is used to avoid artefact caused by over sharpening in dark and high light region.

Additional tuning information

In the case that the final resolution target (number of TV lines) has not been achieved or artifacts are clearly visible in the image, tuning needs to be optimized. Recommendations on how this could be achieved are listed below:

• Checker/maze patterns could be seen if demosaic slopes are set with very high values, if this is the case, decrease the slopes or adjust the thresholds.

• If UU_slope is set too high, edges and flat areas may become noisy, introducing orming artifacts. If this is the case, retune UU parameters.

• Check dynamic pixel correction (DPC) is not introducing artifacts by disabling the module in *Control Tool*. If artifacts disappear then DPC will require tweaking, adjusting DP_threshold and DP_slope, evaluating any impact on other modules once complete.

• Check green equalization (GE) is not introducing artifacts by disabling the module in *Control Tool*. If artifacts disappear then GE_threshold and GE_strength should be adjusted by repeating the GE calibration processes. If artifacts only occur on edges, GE_sens can be adjusted, where a higher values results in more aggressive green equalization on edges.

• If vertical/horizontal resolution takes higher priority than angular directions, decrease AA_slope, increase VH_slope and adjust VA_slope to bias VH resolution.

• Check the image isn’t too blurry as a result of strong noise reduction engines, especially in good lighting conditions.

• Know the limitation of the camera system, as the sensor and optics used has an impact on the overall achievable resolution.

3.15.4 Fine tuning demosaic phase three

Modulation of demosaic parameters is done with respect to a range of gain levels.

Prerequisites

Prerequisite	Status / value
Phase 2 tuning	Completed
Sinter	Tuned
Temper	Tuned
RGB sharpening	Disabled

Table 3.15-13 Sharpening prerequisites.

Tuning procedure - Sharpness

Repeat steps 1-4 previously described for tuning sharpening during phase two for all other gains, filling out the modulation tables provided in dynamic_calibrations.c.

Tuning procedure - NP_offset

Sensor noise is proportional to the gain level set by AE, hence as the gain increases, so does the sensor noise, therefore demosaic sensitivity and sharpening strength must be modulated with respect to gain. NP_offset parameters provide a means of doing this, where at lower gains NP_offset requires no modification. Thus, before being able to tune this parameter, make sure the system is set to a high gain with visible noise, note that noise reduction engines should be tuned at this gain. These are the conditions under which the procedure is demonstrated in the following example.

1. Set the following parameters in the “Demosaic” page:
   - NP_offset = 0.
   - Dmsc_config = 0.
2. Observe the following in Figure 3.15-14:
   - False edge detection due to noise.
   - Checker pattern in horizontal lines at resolution 500.
   - Maze pattern at resolution 700.

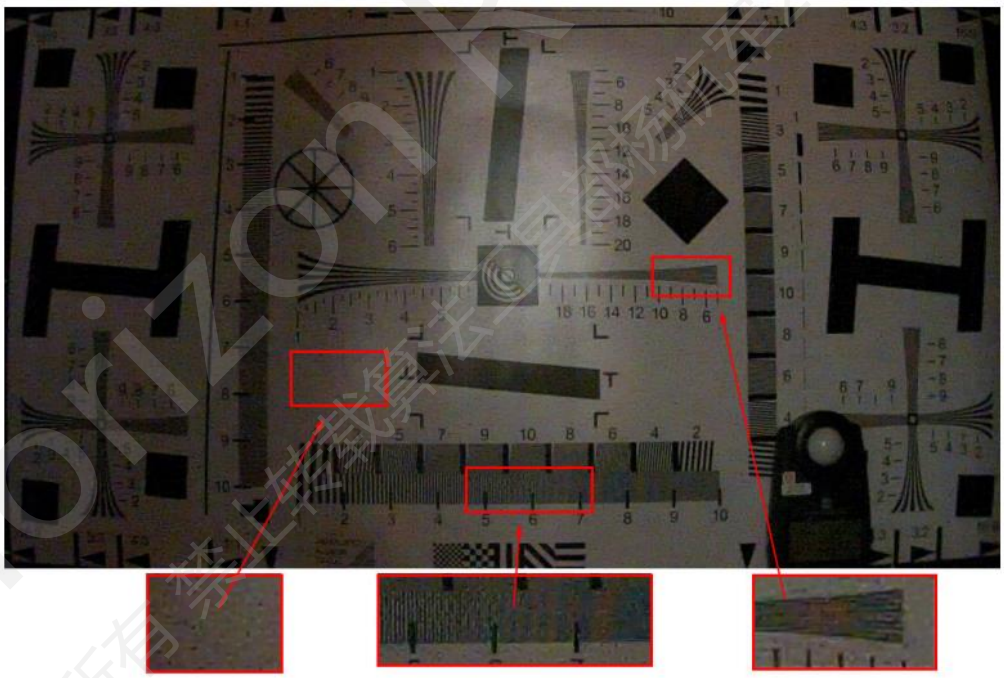


Figure 3.15-14 Image at maximum gain and NP_offset = 0.

3. Now increase the offset to a point where the false edges in the flat areas are reduced without degrading resolution, as seen in Figure 3.15-15. Also ensure that most of the checker pattern has been reduced, making it less visible. Note that false color in the high frequency regions tends to increase.

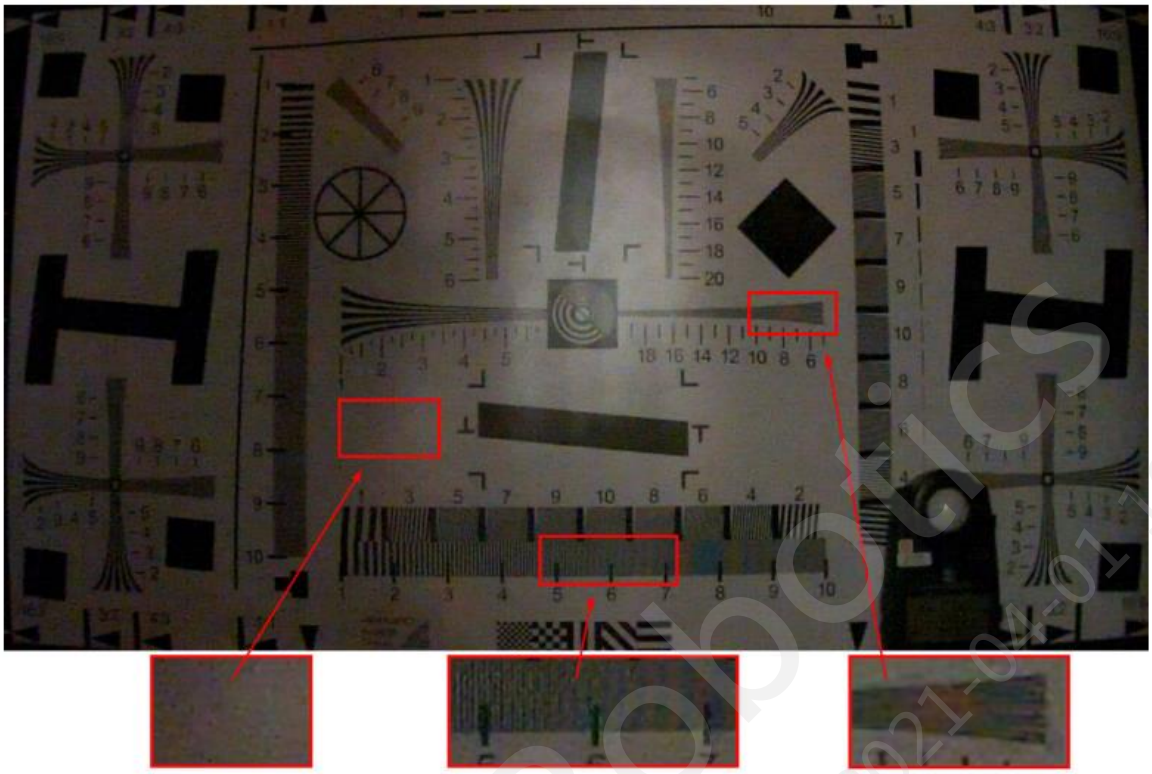


Figure 3.15-15 Image at maximum gain and NP_offset = 50.

4. Repeat steps 1-3 for all available gain levels.

Additional tuning information

• A tabulated example of NP_offset modulation is as follows:

Gain	0	1	2	3	4	5	6	7
NP_offset	1	1	1	1	8	16	24	32

Table 3.15-14 NP_offset modulation example.

Once this LUT is generated, it will need to be set up in FW as _Calibration_demosaic_np_offset_ for the calibrations to take effect. An example of this LUT is shown below:

_Calibration_demosaic_np_offset_={{0,1},{10*256,1}{11*256,12}{12*256,40}};

• A example of sharpening control according to intensity.

The luma_thresh_low/high_d/ud, luma_slope_low/high_d/ud, and luma_offset_low/high_d/ud are parameters for sharpening control according to intensity. This function is used to avoid artefact caused by over sharpening in dark and high light region. The example is as below.

Parameter	**Value**
Luma_thresh_low_d	100
Luma_slope_low_d	16384
Luma_offset_low_d	0
Luma_thresh_high_d	4000
Luma_slope_high_d	16384

Luma_offset_high_d	0

Table 3.15-15 Table 79: example values.

The knee points are Luma_thresh_low_d and Luma_thresh_high_d, the slope and offset will control the gradient of the ramp. In this trapezoidal curve, sharpening strength in hight light and dark region is decreased.

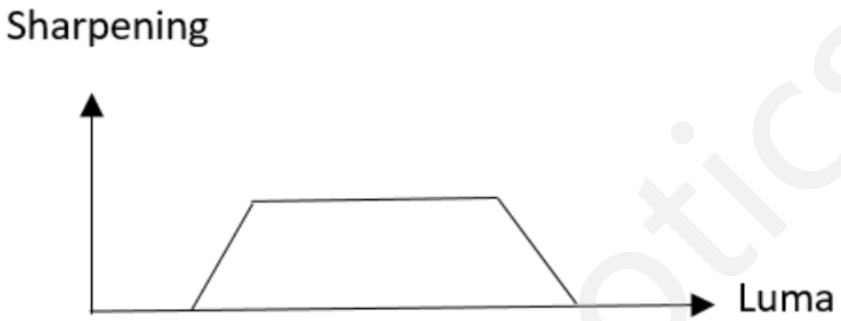


Figure 3.15-16 sharpening control according to intensity

3.16 Purple Fringing Correction（PFC）

3.16.1 Overview

Purple fringing is a form of chromatic aberration which causes an undesirable purple/blue hue around high contrast edges. Tuning takes place during phases two and three as described below:

- During phase one - No tuning required.
- During phase two - PF correction tuned in lab conditions for the majority of scenes.
- During phase three - No tuning required, unless previous tuning or other modules which PF correction is dependent upon causes poor performance.

3.16.2 Tuning Theory

The PF correction module removes purple/blue bordering by isolating the affected areas using masks for hue, saturation, luma, edge detection and radial position (see Table 3.16-1. The fringing is then desaturated.

Key hardware registers

Hardware registers are found on the relevant source page listed in the "Hardware" tab of Control Tool.

Register name	Source	Description
Enable	PF Correction	Enable/disable PF correction
Debug_sel	PF Correction	Enable mask (debug) outputs

Use_color_corrected_rgb	PF Correction	Generate masks using internal CCM
CCM_coeff_[rr...bb]	PF Correction	CCM (automatically generated)
Off_center_mult	PF Correction	Radial distance multiplier
Shading_lut	PF Correction	Radial mask LUT
Purple_strength	PF Correction	Final mask multiplier
Saturation_strength	PF Correction	Fringing saturation after correction
Hue_strength	PF Correction	Hue mask multiplier
Sat_strength	PF Correction	Saturation mask multiplier
Luma_strength	PF Correction	Luma mask multiplier
Sad_slope	PF Correction	Slope of lower SAD limit
Hue_[l/h]_slope	PF Correction	Slope of [lower/upper] hue limit
Sat_[l/h]_slope	PF Correction	Slope of [lower/upper] saturation limit
Luma[1/2]_[l/h]_slope	PF Correction	Slope of [lower/upper] luma limit
Hsl_slope	PF Correction	Slope of combined mask lower limit
Sad_thresh	PF Correction	Value of lower SAD limit
Hue_[l/h]_thresh	PF Correction	Value of [lower/upper] hue limit
Sat_[l/h]_thresh	PF Correction	Value of [lower/upper] saturation limit
Luma[1/2]_[l/h]_thresh	PF Correction	Value of [lower/upper] luma limit
Hsl_thresh	PF Correction	Value of lower combined mask limit
Sad_offset	PF Correction	Mask value below SAD lower limit
Hue_[l/h]_offset	PF Correction	Mask value [below/above] hue [lower/upper] limit
Sat_[l/h]_offset	PF Correction	Mask value [below/above] saturation [lower/upper] limit
Luma[1/2]_[l/h]_offset	PF Correction	Mask value [below/above] luma [lower/upper] limit
Hsl_offset	PF Correction	Final mask value below combined mask lower limit

Table 3.16-1 PF correction hardware registers

Note: [l/h] indicates that there are two parameters present, one for each respective "low" and "high" limiter. The same approach is taken towards [1/2] where "1" and "2" respectively represent "dark" and "light" luminance.

Masks

PF correction is tuned by isolating affected areas using several masks. Masks are manipulated with linear threshold functions, characterized by Slope, Thresh and Offset parameters. Table 3.16-2 lists each mask and the Debug_sel value required to access them.

Debug_sel **value**	**Mask functionality**
1	Shows the radial mask
2	Shows the SAD (edge detection) mask
3	Shows the hue mask
4	Shows the saturation mask
5	Shows the luma mask
6	Shows final PF mask (combination of masks 1-5)

Table 3.16-2 PF correction Debug_sel masks

If tuning all 6 masks, the goal is for the fringing areas of the mask to be as light as possible, and the non-fringing areas to be as dark as possible.

3.16.3 Tuning during phase two

PF correction is first tuned during the early stages of phase two as it is not required in phase one. Start by making sure the prerequisites listed in Table 3.16-3 are complete.

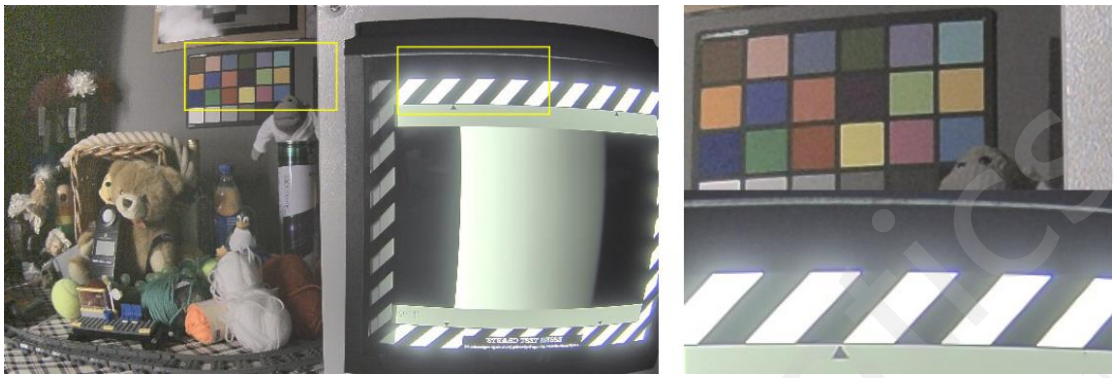
Module	**Status / value**
Black level	Tuned
Chromatic aberration	Tuned (if present in ISP)
Iridix	Tuned
Temper	Tuned
DPC	Tuned
Green equalization	Tuned
Demosaic	Tuned
CCM	Tuned
AWB	Tuned

Table 3.16-3 PF correction phase two prerequisites

3.16.3.1 Tuning procedure – Setup

Purple fringing is most easily produced in daytime outdoor scenes featuring objects silhouetted against the sky. For tuning however, it is best to mimic these conditions in a controlled lab

environment if possible. In the example shown in Figure 3.16-1a, a light source with a striped filter has been placed in view to create high contrast edges. Most of the tuning procedure will refer to the two ROIs highlighted in yellow (also see Figure 3.16-1b, the idea being to remove purple fringes without affecting the ColorChecker patches.



(a) Purple fringe tuning scene. (b) Purple fringing region of interest.

Figure 3.16-1 PF ROI in tuning scene

1. Focus on a test scene containing high contrast edges and a ColorChecker chart.

3.16.3.2 Tuning procedure – Hue mask

2. The hue mask is the first mask requiring tuning, start by setting the values listed below:

Register	Value
Debug_sel	3
Use_color_corrected_rgb	1
Hue_low_slope	284
Hue_low_offset	0
Hue_low_thresh	1690
Hue_high_slope	1422
Hue_high_offset	0
Hue_high_thresh	2150
Hue_strength	768

Table 3.16-4 Initial register values for hue mask

These registers define the linear threshold function, which is visualized in Figure 3.16-2. With this function shape, the mask will isolate blues.

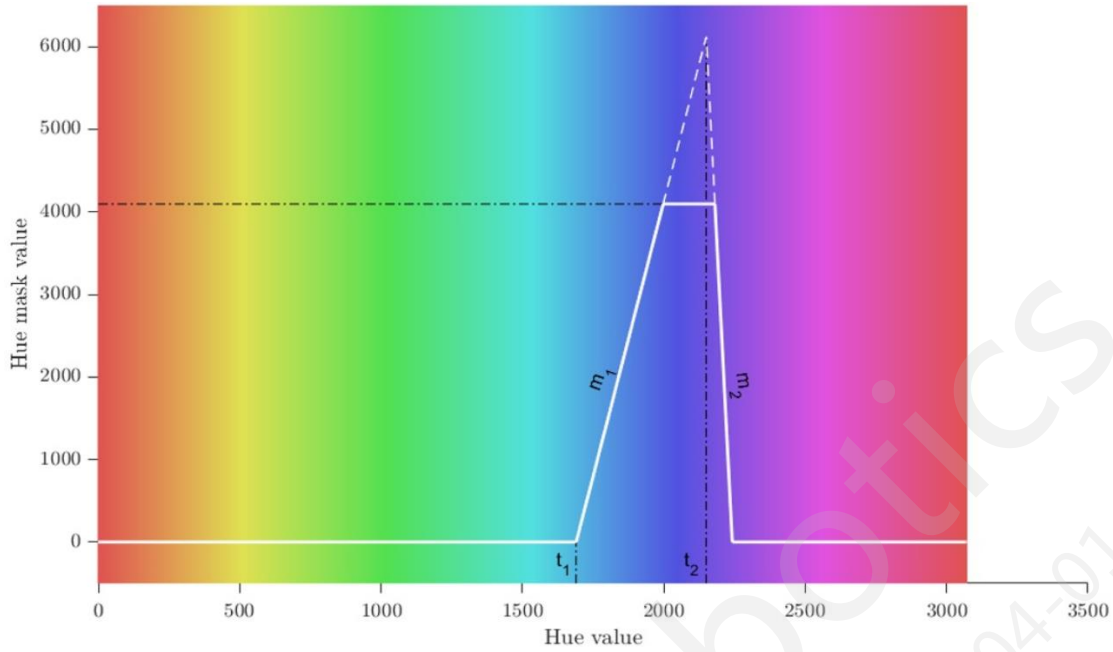


Figure 3.16-2 Hue parameters and curve

$$\textit{Where: } t_1 = \text{Hue\_low\_thresh}$$
$$t_2 = \text{Hue\_high\_thresh}$$
$$m_1 = \text{Hue\_low\_slope} \times \text{Hue\_strength} \div 2^{14}$$
$$m_2 = \text{-Hue\_high\_slope} \times \text{Hue\_strength} \div 2^{14}$$

3. Modify the hue selection using the thresh and slope registers until all fringing regions are included in the mask, with the others minimized. Referring to Figure 3.16-2, thresh registers will adjust the width of the envelope, while slope registers will alter the gradient of the transition. Figure 3.16-3 shows a tuned hue mask.

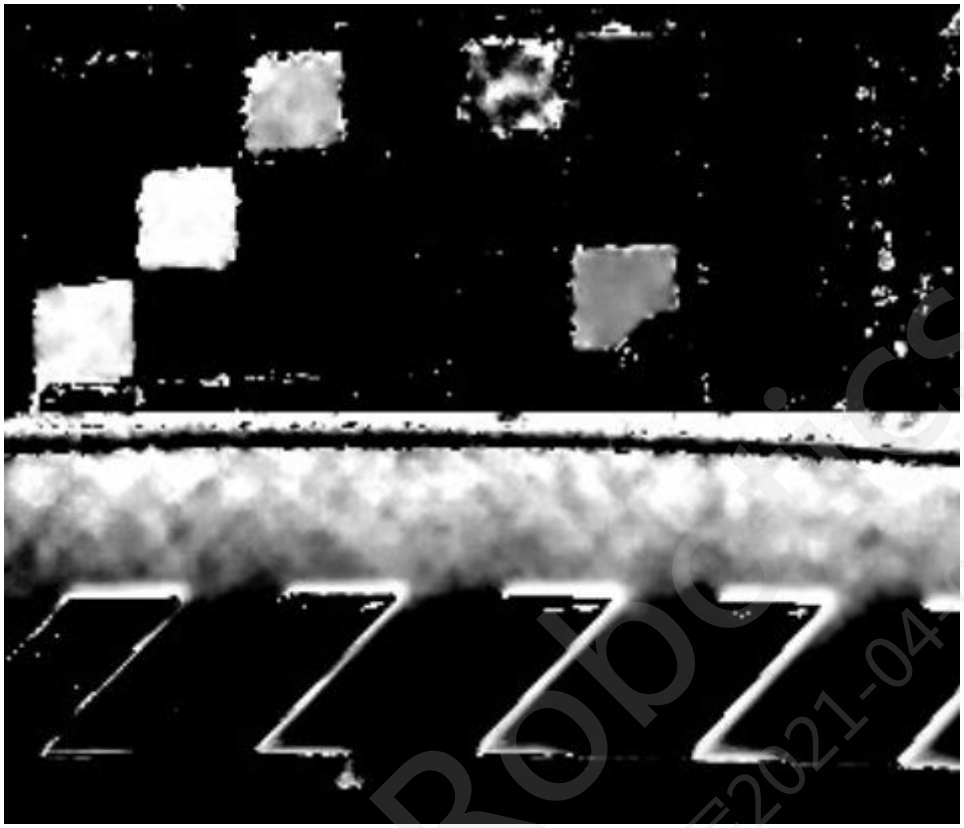


Figure 3.16-3 Tuned hue mask

4. The Use_color_corrected_rgb register provides further customization - if set to 1 (recommended), the image used for mask generation will be color-corrected using internal CCM settings. If set to 0, the masks will be generated from uncorrected data. In either case, this only affects the mask generation - the image which the mask is applied to is always uncorrected. Set preferred Use_color_corrected_rgb value

3.16.3.3 Tuning procedure – Saturation mask

5. The second mask requiring tuning is saturation. Set the values listed below:

Register	Value
Debug_sel	4
Sat_low_slope	123
Sat_low_offset	0
Sat_low_thresh	164
Sat_high_slope	0
Sat_high_offset	4095
Sat_high_thresh	0

Sat_strength	512

Table 3.16-5 Initial register values for saturation mask

These registers define the linear threshold function, which is visualized in Figure 3.16-4. With this function shape, the mask will isolate pixels of medium to high saturation.

6. Referring to Figure 3.16-4, adjust the shape of the filter using Sat_low_thresh and Sat_low_slope until all fringing regions are included in the mask, with other regions minimized. The initial "Sat_high" values will have no effect, but can be adjusted to reject high saturation pixels if desired. Figure 3.16-5 shows a tuned saturation mask.

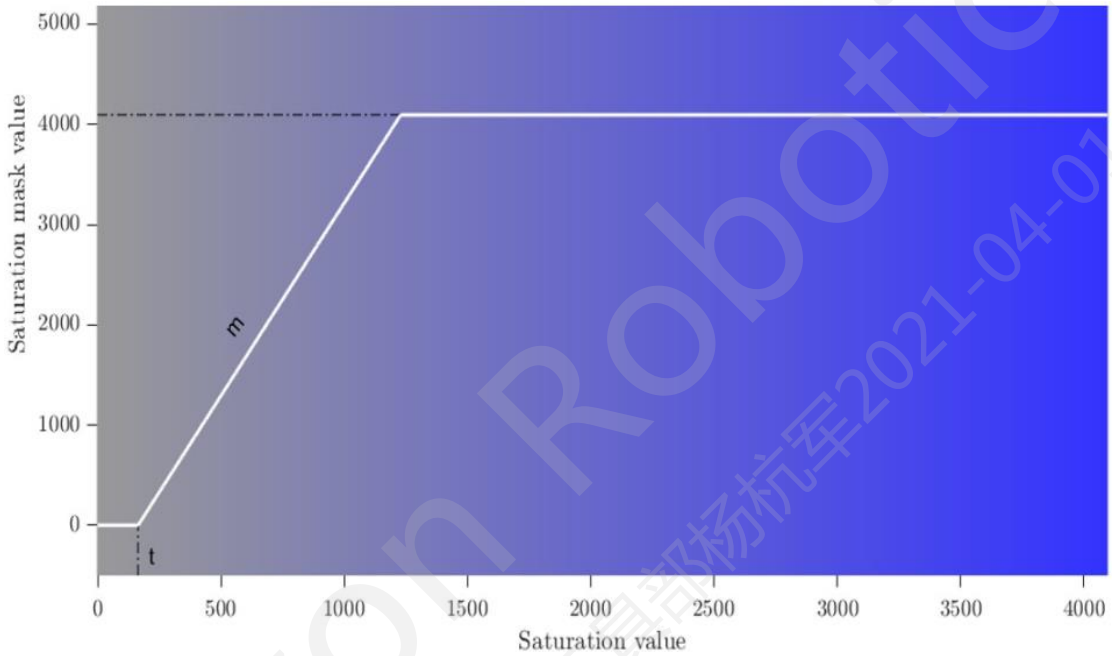


Figure 3.16-4 Saturation parameters and curve

Where: t_1 **= Sat_low_thresh**

m_1 **= Sat_low_slope** $\times$ **Sat_strength** $\div 2_{14}$

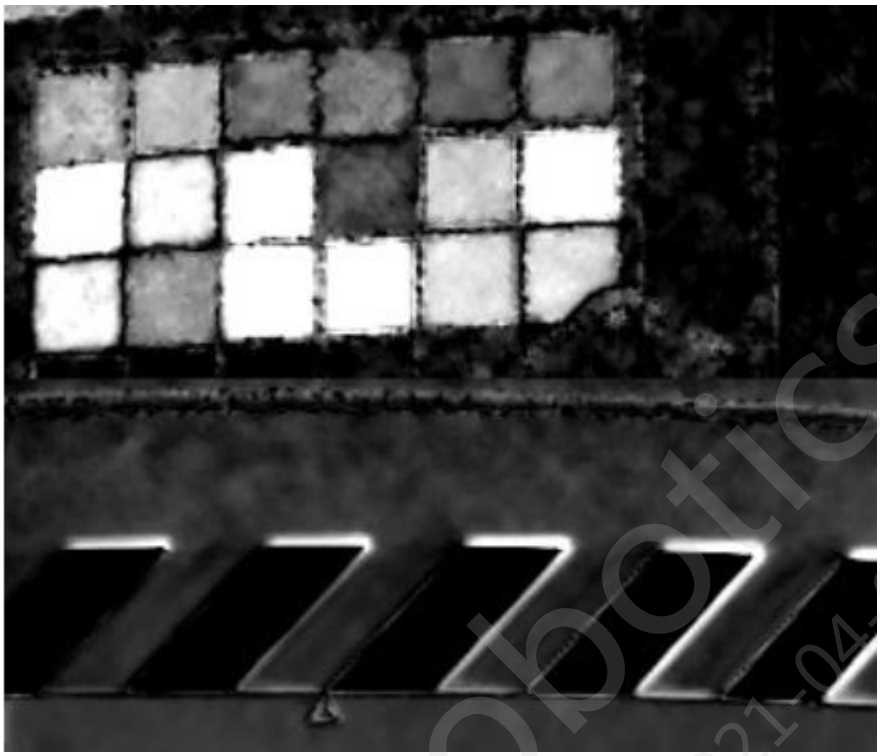


Figure 3.16-5 Tuned saturation mask

3.16.3.4 Tuning procedure – Luma mask

7. The luma mask is the next mask to be tuned - start by setting the values listed below:

Register	Value
Debug_sel	5
Luma1_low_slope	110
Luma1_low_offset	0
Luma1_low_thresh	205
Luma1_high_slope	70
Luma1_high_offset	0
Luma1_high_thresh	1500
Luma2_low_slope	400
Luma2_low_offset	0
Luma2_low_thresh	3450
Luma2_high_slope	1500
Luma2_high_offset	0
Luma2_high_thresh	3900

Luma_strength	1024

Table 3.16-6 Initial register values for luma mask

These registers define the linear threshold function, depicted in Figure 3.16-6. As fringing occurs around edges, the function allows for two peaks - one for each dark and light side of an edge. "Luma1" registers isolate tones associated with fringing on the dark side of each edge. "Luma2" does the same for the light side of each edge.

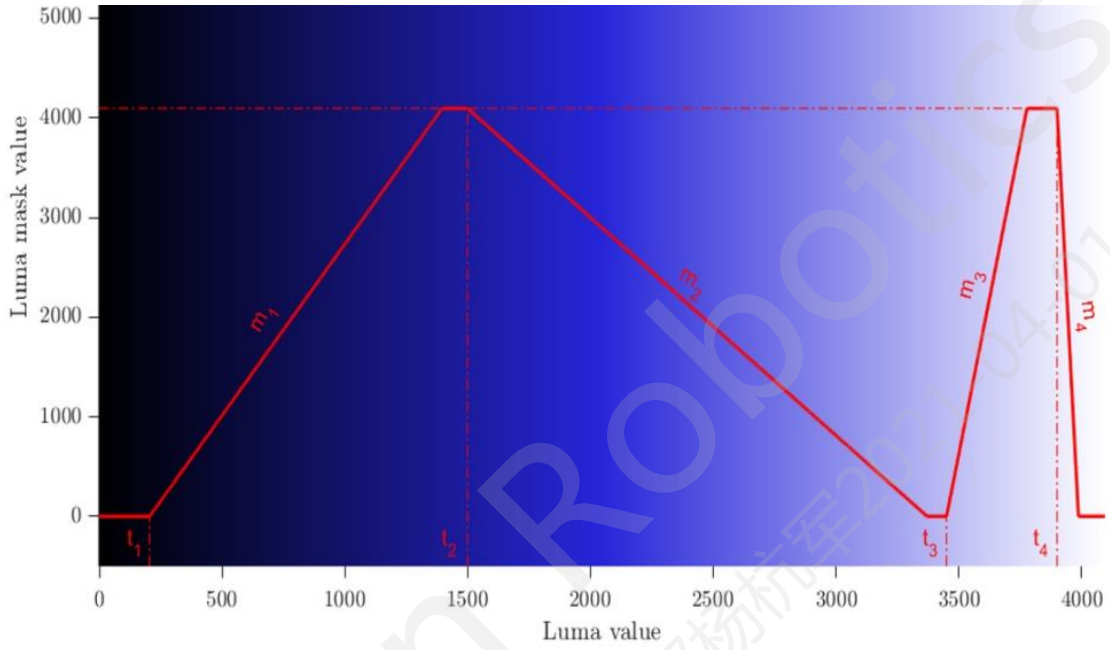


Figure 3.16-6 Luma parameters and curve

Where: t_1 = Luma1_low_thresh
t_2 = Luma1_high_thresh
t_3 = Luma2_low_thresh
t_4 = Luma2_high_thresh
m_1 = Luma1_low_slope × Luma_strength ÷ 2_{15}
m_2 = -Luma1_high_slope × Luma_strength ÷ 2_{15}
m_3 = Luma2_low_slope × Luma_strength ÷ 2_{15}
m_4 = -Luma2_high_slope × Luma_strength ÷ 2_{15}

8. Referring to Figure 3.16-6, adjust the shape of the filter using "thresh" and "slope" registers until all fringing regions are included in the mask, with other regions minimized. Figure 3.16-7 shows a tuned luma mask.

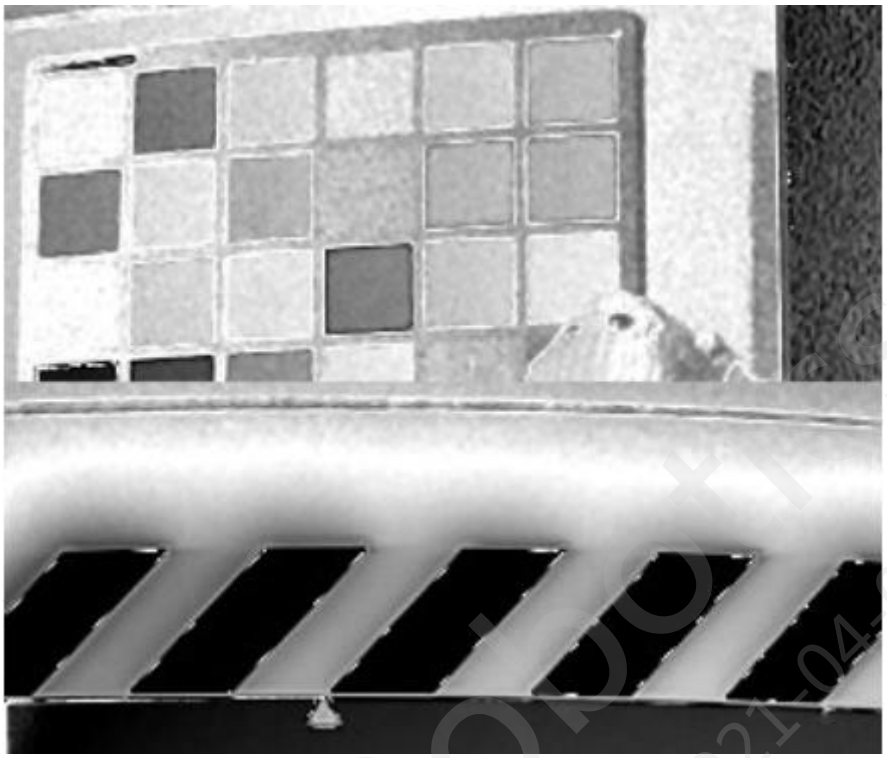


Figure 3.16-7 Tuned luma mask

3.16.3.5 Tuning procedure - SAD mask

Tuning procedure - SAD mask

9. The SAD mask is the next mask to be tuned - start by setting the values listed below:

Register	Value
Debug_sel	2
Sad_slope	1024
Sad_offset	0
Sad_thresh	1200

Figure 3.16-8 Initial register values for SAD mask

These registers define the linear threshold function, depicted in Figure 3.16-9. The SAD input is produced by edge detection. The purpose of this threshold function is to differentiate between edges which produce purple fringing and those which do not.

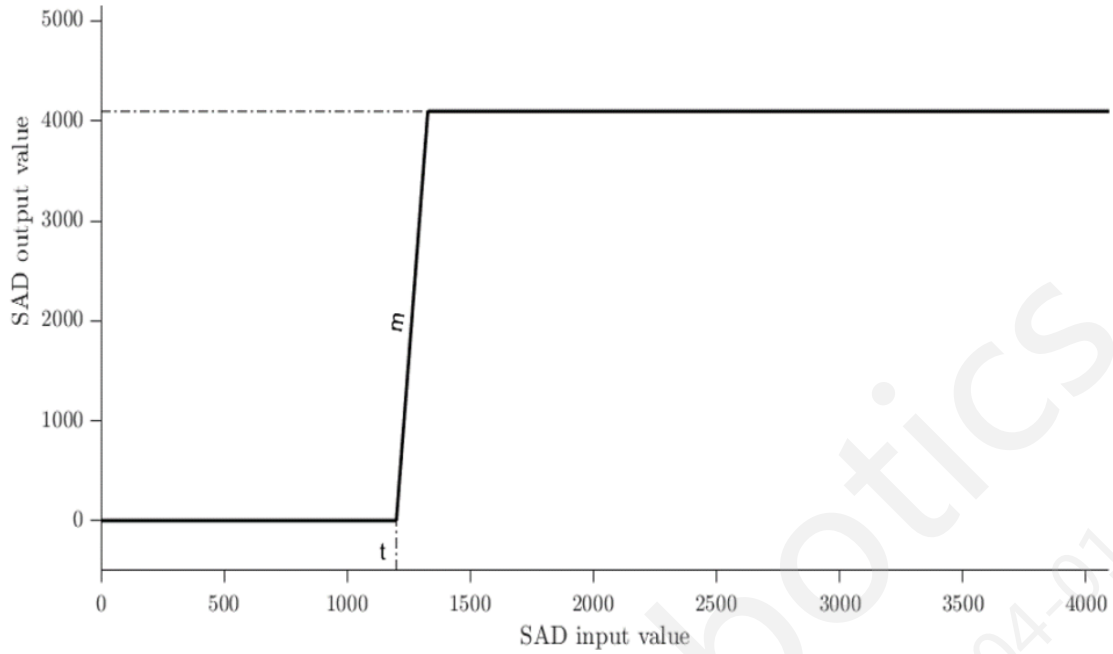


Figure 3.16-9 SAD parameters and curve

$$Where: t = \text{Sad\_thresh}$$
$$m = \text{Sad\_slope} \div 2^5$$

10. Referring to Figure 3.16-9, adjust Sad_thresh so that all purple fringing areas are white, but other edges (such as the ColorChcker) are as thin as possible. Figure 3.16-10 shows a tuned SAD mask.

Figure 3.16-10 Tuned SAD mask.

3.16.3.6 Tuning procedure - Radial mask

11. The radial mask is next to be tuned - start by setting the values listed in Table 3.16-7.

Register	Value
Debug_sel	1
Off_center_mult	3983
Shading_lut	[0, 0, 0, 0, 23, 46, 70, 93, 116, 139, 162, 185, 209, 232, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255, 255]

Table 3.16-7 Initial register values for radial mask

These registers define the linear threshold function, the profile of which is depicted in Figure 3.16-11. Using a radial mask allows for more aggressive correction without affecting central regions. This works as purple fringing is usually more noticeable in edges that are further from the image center.

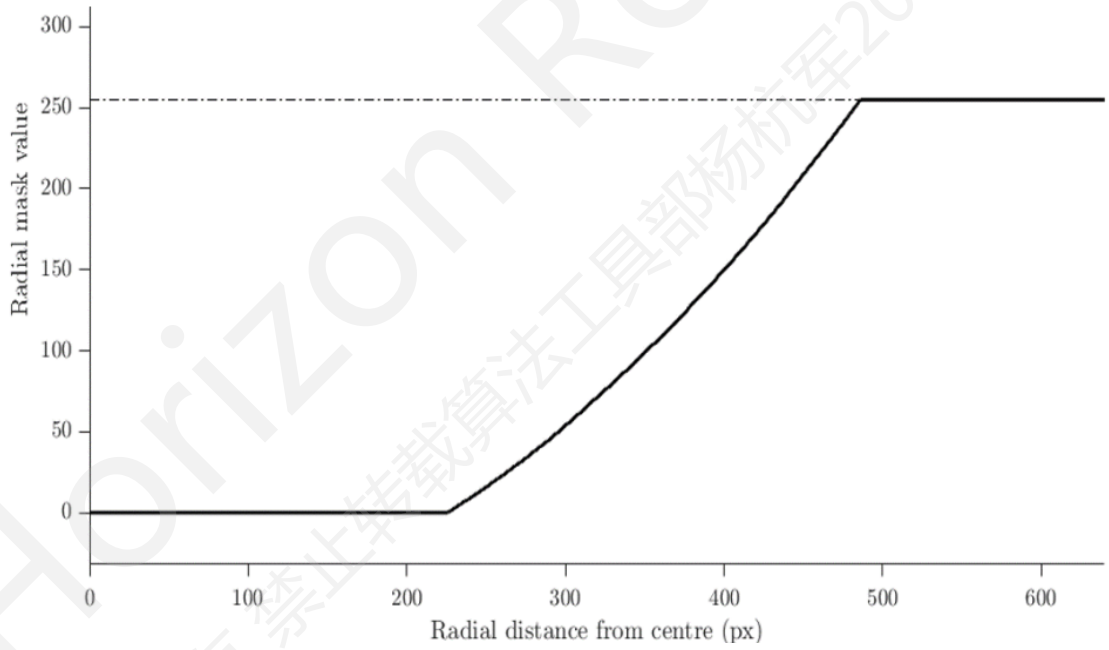


Figure 3.16-11 Radial parameters and curve.

12. Adjust Off_center_mult to spatially scale the radial profile. Adjust Shading_lut values for greater control over the radial mask profile. Figure 3.16-12 shows a tuned radial mask.

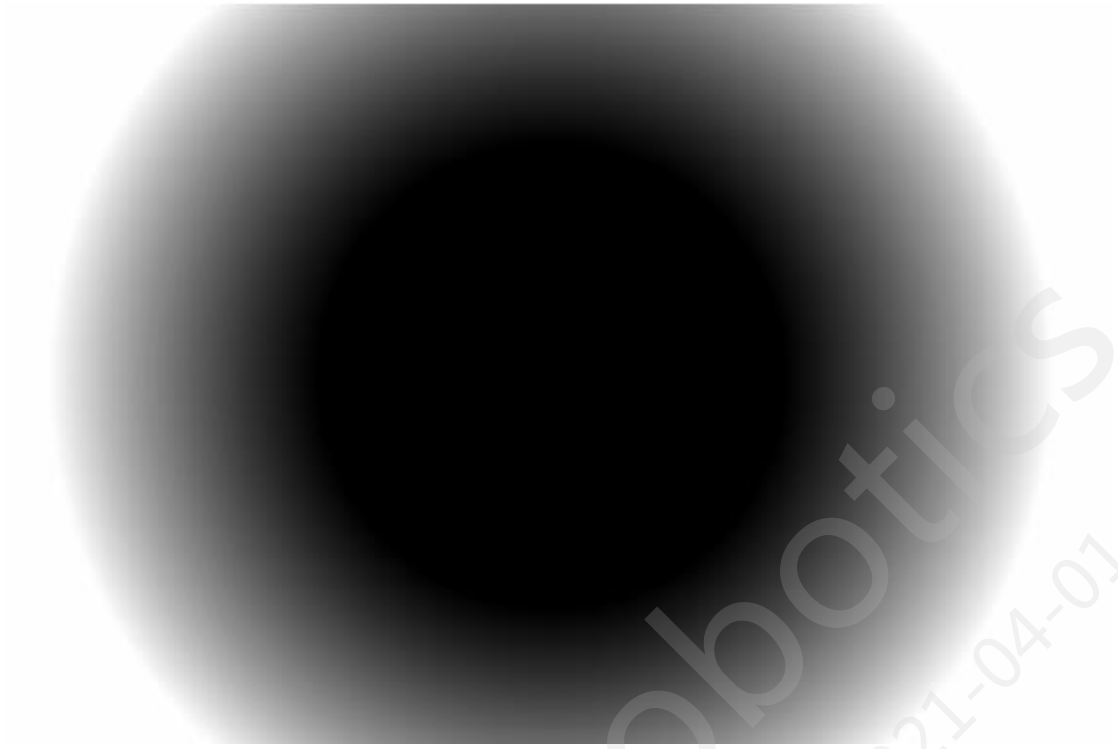


Figure 3.16-12 Tuned radial mask

3.16.3.7 Tuning procedure - Final (combined) mask

13. The combined mask is the last to be tuned - set the values listed below:

Register	Value
Debug_sel	6
Hsl_slope	36
Hsl_offset	0
Hsl_thresh	0
Purple_strength	768

Table 3.16-8 Initial register values for final mask

These registers define the linear threshold function, depicted in Figure 3.16-13. The input to this function is a combination of all the previously described masks, so its purpose is to provide some final control over the fringing detection.

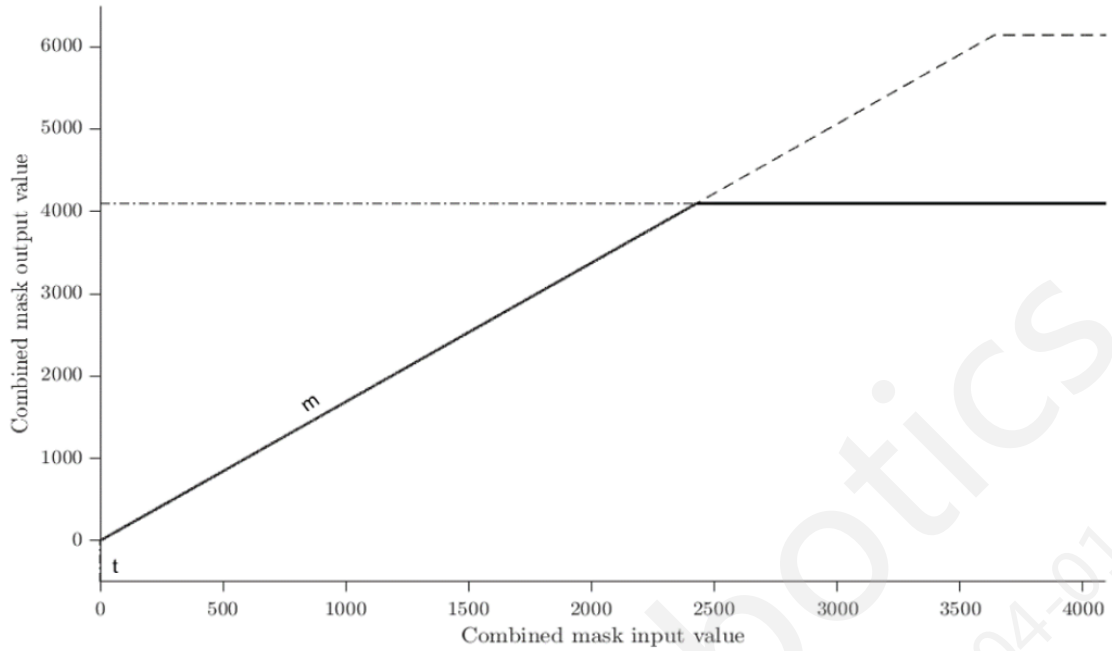


Figure 3.16-13 Final mask parameters and curve.

$$Where: t = hsl\_thresh$$
$$m = hsl\_slope \times Purple\_strength \div 2^{14}$$

14. Referring to Figure 3.16-13, adjust Hsl_thresh to make fringed areas as light as possible, with all other areas as dark as possible. Figure 3.16-14 shows a tuned final mask.

Figure 3.16-14 Tuned final mask

3.16.3.8 Tuning procedure – Desaturation

15. The last stage when tuning PF correction focuses on desaturation - start by setting the values listed below:

Register	Value
Debug_sel	0
Saturation_strength	0

Table 3.16-9 Initial register values for desaturation

16. Desaturation is applied to the highlighted areas, removing the fringing. Set Saturation_strength between 0 (complete desaturation of fringes) and 255 (no effect on fringes).Figure 3.16-15 shows the impact of the tuned purple fringing correction module.

(a) Before. (b) After.

Figure 3.16-15 Regions of interest before and after purple fringing correction

3.17 Noise Profile (NP)

3.17.1 Overview

3.17.1.1 Noise profile window

When you open the noise profile module, a window like Figure 3.17-1 will pop up.

Figure 3.17-1 Noise profile window

The purpose of the noise profile module is to find the relationship between pixel intensity and temporal noise in the raw images of the sensor. This relationship is encoded in 3 look-up tables (LUTs), designed for different blocks of the ISP:

- CALIBRATION_NOISE_PROFILE – This is used by DPC, temper™ and sinter® blocks, and is affected by the frontend companding function of the ISP. If square root companding is used, ignore this LUT. It is a logarithmic representation of noise. This LUT is plotted in the noise profile window (NP LUT button).
- CALIBRATION_DEMOSAIC – This is used by the demosaic block and is not sensor dependent.
- CALIBRATION_WDR_NP_LUT – This is used by the WDR stitching block in FS-Lin mode and is not affected by the companding function of the ISP. It is a linear representation of noise.

3.17.1.2 GUI features

Figure 3.17-1 Noise profile windowlists the function of each region.

Region	Sub-region name	Function
Images	Dark	Displays dark image
	Light 1	Displays the first light image
	Light 2	Displays the second light image

Graphs	View	Choose plot type
	Show saturation effects	Choose whether to display saturation effects in the plot trendlines (Linear and FS-Lin mode only)
	Show square root effects	Choose whether to display square root effects on the plot (Native-WDR and Native-WDR-Lin mode only)
	Print-friendly results	Opens a pop-up with results and plots
Options	Images – load dark image	Load your dark image
	Images – load light images	Load your two light images
	Settings – LUT size	Set the LUT node count of your hardware
	Settings – Trend data max	Set maximum pixel value for trendline estimation
	Settings – Companding function	Frontend function applied to image data in ISP
Results	-	Table of sensor characteristics calculated from your images

Table 3.17-1 List of all regions and their functions in the noise profile window

3.17.2 User guide

3.17.2.1 Prerequisites

- Black level correctly calibrated.
- Gamma FE correctly calibrated (Native-WDR modes only).
- Noise profile image set captured (1 dark, 2 light).

3.17.2.2 Capturing a noise profile image set

To tune noise profile you need 3 images – two light (illuminated scene) frames and one dark (no light reaching the sensor) frame, all captured using the same settings.

1. **Set up the scene**

Fix the camera to a stable tripod, facing a lab scene, with a calibrated monitor facing away from the scene for you to use to assess the exposure. The images must have the widest possible range of pixel intensities, and the flattest possible histogram.

Figure 3.17-2 Noise profile calibration scene

The scene in Figure 3.17-2 is a good example. Notice that the darkest objects in the scene (black ball of wool) are much darker than ColorChecker patch 24. Remove everything from the scene that is not static (plants, clocks for example). Only illuminate the scene with non-flickering artificial lighting.

Note: The noise profile tuning will only be accurate if you follow every guideline above.

2. **Apply camera settings**
   Set up the system as follows, using Control Tool:
   - Linear ISP mode
   - Iridix off
   - Manual integration time (set to maximum value)
   - Manual analog gain (set to 0)
   - Manual sensor digital gain (set to 0)
   - Manual ISP digital gain (set to 0)

   Note: The documentation for the "black level" module has more information about setting manual values.

3. **Fine-tune the exposure**
   Adjust the exposure until patch 19 of the ColorChecker is clipped white in your calibrated monitor, and patch 20 is almost clipped. This makes sure the data range is large enough for calibration. To decrease exposure, reduce the brightness of the illumination. If you can't do that, decrease integration time.

   To increase exposure, increase the brightness of the illumination. If you can't do that, increase analog gain.

4. **Capture two identical raw light images**
   Make sure the camera and scene do not move between captures – do not touch the camera if you can avoid it, and avoid floor vibrations.

5. **Cover the lens, capture raw dark image**
   Make the room as dark as possible and cover the lens entirely with opaque material. Make sure no light can reach the sensor. Use the same approach as you used to capture black level images, but keep the camera settings the same as for the two light images.

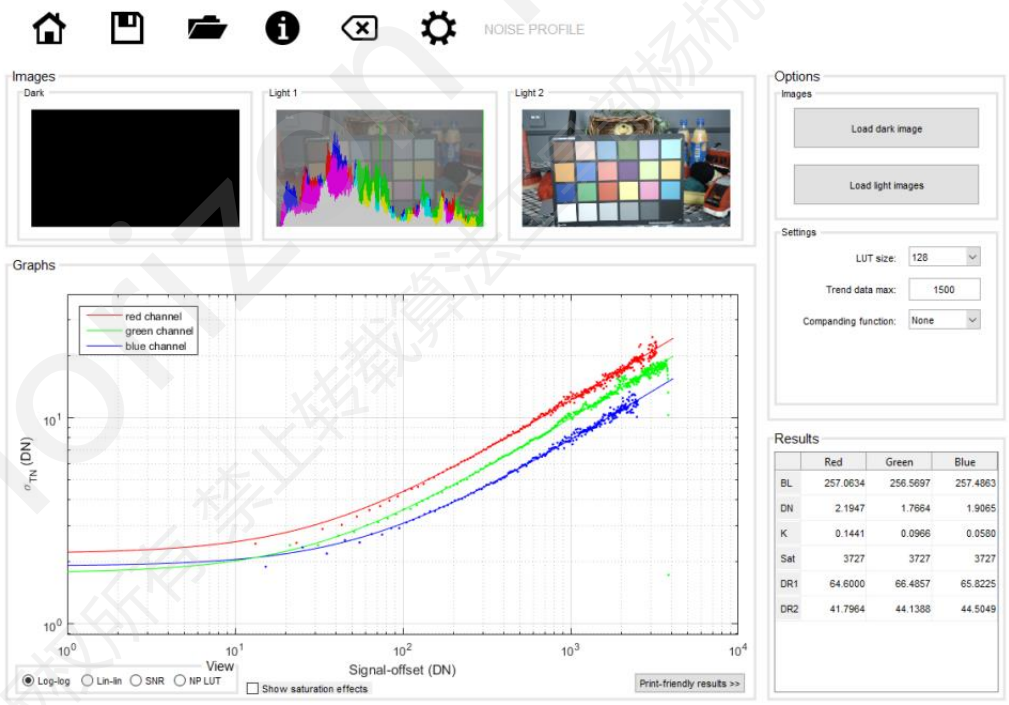
3.17.2.3 Calibration procedure

1. **Load dark and light images in either order using the buttons**

   Thumbnails appear – hover over them to see a histogram. Check that the histogram contains data for the full intensity range in the green channel. The noise profile calculation will automatically start.

2. **Check that the calculated green channel trendline follows the scatter plot data**

   The example below shows a good calibration. Click the Log-log button to check that the green trendline is accurate for very dark pixels too. If it is not, reduce the value of Trend data max until it is. Trend data max limits the regression data set by removing pixels whose noise distribution is affected by saturation.



3. **Look at the sensor characteristics in the Results panel**

   This is generally useful data, but it does not affect the noise profile LUTs. The unit “DN” stands for “digital numbers”. This is the value of the least significant bit of the pixel data.

Parameter	Acronym	Units	Meaning
Black Level	BL	DN	Sensor offset
Dark Noise	DN	DN	Sensor noise standard deviation under no illumination
K	K	DN/e	Sensor conversion gain. This is the ratio of digital signal to photodiode well population
Saturation	Sat	DN	Effective maximum output of the sensor, after subtracting black level. For every pixel brighter than this, the noise behaviour is dominated by the effects of clipping
Dynamic Range 1	DR1	dB	Ratio between the saturation value and the signal due to dark noise
Dynamic Range 2	DR2	dB	Ratio between the saturation value and the lowest signal with SNR>20dB

4. Set LUT size and Companding function to match your hardware

There are 3 companding function options:

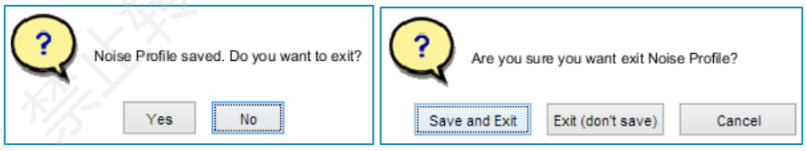
• None – when any frontend companding functions are bypassed. CALIBRATION_NOISE_PROFILE matches the green trendline, normalized by the dark noise level.

•Square root – when the frontend companding function is a square root. CALIBRATION_NOISE_PROFILE becomes flat. The LUT has logarithmic scaling, so after normalization it is just zeros.

• Gamma DL – when the frontend companding function is Gamma DL. Set alpha to match the value used in the companding hardware block. CALIBRATION_NOISE_PROFILE changes according to the shape of the function.

5. Save parameters and return to the home window

Save your data using one of the dialog boxes below before leaving the noise profile module or your data will be lost. In the home window, click the same icon to export your data to a .mat file.



(a) – Save dialog window. (b) – Home dialog window.

Figure 3.17-3 Noise profile saving options

3.18 Color Noise Reduction (CNR)

3.18.1 Overview

Color noise, also referred to as Chroma noise, appears as blotches of incorrect color. This module corrects color for each pixel in YUV space by alpha-blending its chromaticity with an intelligently estimated local mean. A brief outline of the process is described below:

- *During phase one* - No tuning required
- *During phase two* - CNR tuned in lab conditions for the majority of scenes.
- *During phase three* - CNR parameters are fine tuned to suit all scenes. Primarily involving modulation of parameters according to gain.

3.18.2 Tuning Theory

CNR is tuned by manipulating a series of masks using linear threshold functions, while a range of global hardware registers remain fixed. These are both described below:

Key hardware registers – Global

Hardware registers are found on the relevant *source* page listed in the "Hardware" tab of *Control Tool*.

Register name	Source	Description
Enable	CNR	Enable/disable CNR module
[U/V]_center	CNR	Center point of [U/V] plane
Line_length	CNR	Length of recursive (vertical) filter
Effective_kernel	CNR	Length of Gaussian (horizontal) filter
Global_offset	CNR	Maximum value of uv_seg[1/2] mask
Global_slope	CNR	Minimum value of uv_seg[1/2] mask
Debug_reg	CNR	Selects visible blending mask

Table 3.18-1 CNR global hardware registers.

Debug_reg is used to visualize the each of the masks used for tuning. An overview of values can be seen in Figure 3.18-2.

Debug_reg **value**	**Mask name**	**Property represented by mask**
0	no mask	
2	uv_seg1	Saturation of pixel
3	uv_seg2	Saturation of pixel
4	umean1	Local mean chromaticity
5	vmean1	Local mean chromaticity
6	umean2	Local mean chromaticity

7	vmean2	Local mean chromaticity
8	uv_var1	Difference between pixel chromaticity and local mean
9	uv_var2	Difference between pixel chromaticity and local mean
10	uv_var1_avr	Local mean uv_var1 value
11	uv_var2_avr	Local mean uv_var2 value
12	uv_delta1	Alpha value for blending of pixel chromaticity
13	uv_delta2	Alpha value for blending of pixel chromaticity

Table 3.18-2 Viewing masks with Debug_reg.

Masks are characterized by slope, threshold and offset parameters. CNR appears to contain a lot of parameters, but tuning is actually quite straightforward, involving only 3 of the offset parameters and 3 of the slope parameters, which are then copied across to multiple filters. For all CNR masks, threshold values are always 0 and require no tuning.

Key hardware registers - Masks

Register name	Source	Description
UV_seg[1/2]_offset	CNR	Saturation lower limit
UV_seg[1/2]_slope	CNR	Sensitivity of uv_seg[1/2] mask
[U/V]mean[1/2]_offset	CNR	Edge detection lower limit
[U/V]mean[1/2]_slope	CNR	Sensitivity of edge detection
UV_var[1/2]_offset	CNR	Color difference lower limit
UV_var[1/2]_slope	CNR	Sensitivity of color difference mask
UV_var[1/2]_scale	CNR	Slope multiplier for color difference mask
UV_delta[1/2]_offset	CNR	Color noise lower limit
UV_delta[1/2]_slope	CNR	Strength of color noise reduction
Delta_factor	CNR	Color bleeding correction

Table 3.18-3 CNR hardware registers for masks.

Note: [U/V] notation indicates there are actually two parameters, one for each of the respective "U" and "V" planes. The same principle is applied to [1/2], where a parameter is present for each enumeration.

3.18.3 Tuning CNR during phase two

CNR is first tuned during the early stages of phase two as it is not required in phase one. Start by making sure the prerequisites listed in Figure 3.18-4 are complete.

Prerequisite	Status / value
Black level	Tuned
CCM	Tuned
DPC	Tuned
Sinter	Tuned
Temper	Tuned
Demosaic	Tuned

Green equalization	Tuned
Set Enable	0 (disables CNR)

Table 3.18-4 CNR phase two prerequisites.

Tuning procedure

1. Properly focus a test scene including a ColorChecker chart and a range of textures. Now illuminate this scene using low D65 lighting (for example 10 lux) so that color noise becomes easy to see, as demonstrated in Figure 3.18-1. Notice that it is particularly visible in the mid-tones and textured chart on the right.

Figure 3.18-1 CNR tuning scene.

2. Tuning starts with the UV_seg mask by setting the global and UV_seg parameters listed in Table 3.18-5 CNR initial UV_seg register values.Values in brackets are for a *12-bit* module.

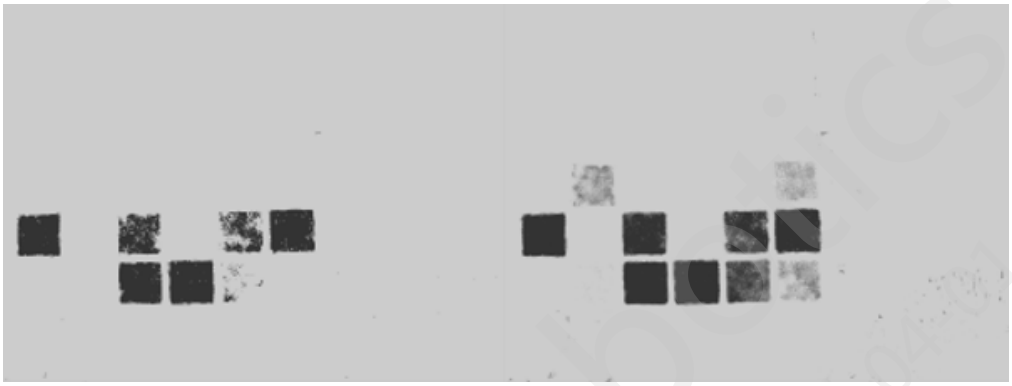
Register	**Value**
Enable	1
Mode	0
Debug_reg	2
Line_length	940
U_center	512 (2048)
V_center	512 (2048)
Global_offset	818 (3276)
Global_slope	205 (819)
UV_seg1_slope	0
UV_seg1_threshold	0
UV_seg1_offset	150 (600)

Table 3.18-5 CNR initial UV_seg register values.

3.The UV_seg masks differentiate saturated color from unsaturated. The initial UV_seg1_slope value of *0* produces the sharpest possible transition between saturated and unsaturated areas. High saturation pixels will be dark, while low saturation pixels will be light. Adjust UV_seg1_offset so

that some of the patches of the ColorChecker chart are dark, but any color noise in neutral areas is not visible. An example of this is shown in Figure 3.18-2*a*.

4. Now increase UV_seg1_slope. This will produce a more gradual transition between saturated and unsaturated pixels. Try bringing the value to *65535*, then decrease it until the color noise in neutral areas is not visible in the mask. When this meets criteria (see Figure 3.18-2b), UV_seg can be considered tuned.



(a) Tuned UV_seg1_offset. (b)Tuned UV_seg1_slope

Figure 3.18-2 Tuning UV_seg.

5. Moving onto the UVmean masks, set the values seen in Table 3.18-6.

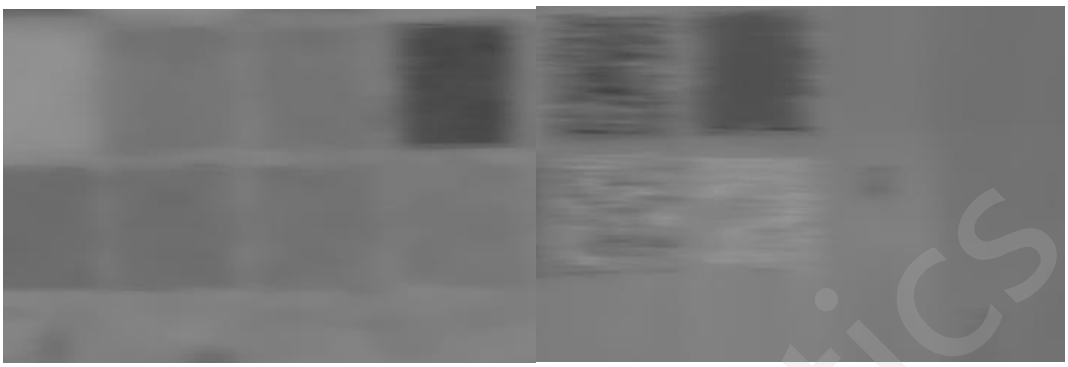
Register	Value
Debug_reg	4
Umean1_slope	57000
Vmean1_slope	57000
Umean2_slope	57000
Vmean2_slope	57000
Umean1_threshold	0
Vmean1_threshold	0
Umean2_threshold	0
Vmean2_threshold	0
Umean1_offset	60 (240)
Vmean1_offset	60 (240)
Umean2_offset	60 (240)
Vmean2_offset	60 (240)

Table 3.18-6 CNR initial UVmean register values.

In the tuned masks, all 4 tuned offset values will match each other, and the same can be said for the 4 slope parameters, so there are actually only two parameters to tune.

6. Correctly tuned UVmean masks show no blotches due to color noise, while retaining the edges of ColorChecker patches. Adjust Umean1_offset, noting that decreasing this too far will allow blotches to show in the mask (see Figure 3.18-3a), while increasing it will reduce the edge-preservation. The filter behavior produces horizontal lines in the ColorChecker patches, and this is expected. However,

the mottled effect shown in Figure 3.18-3b means that Umean1_offset is too high. Once a value has been chosen for Umean1_offset, try adjusting Umean1_slope to improve the mask further.

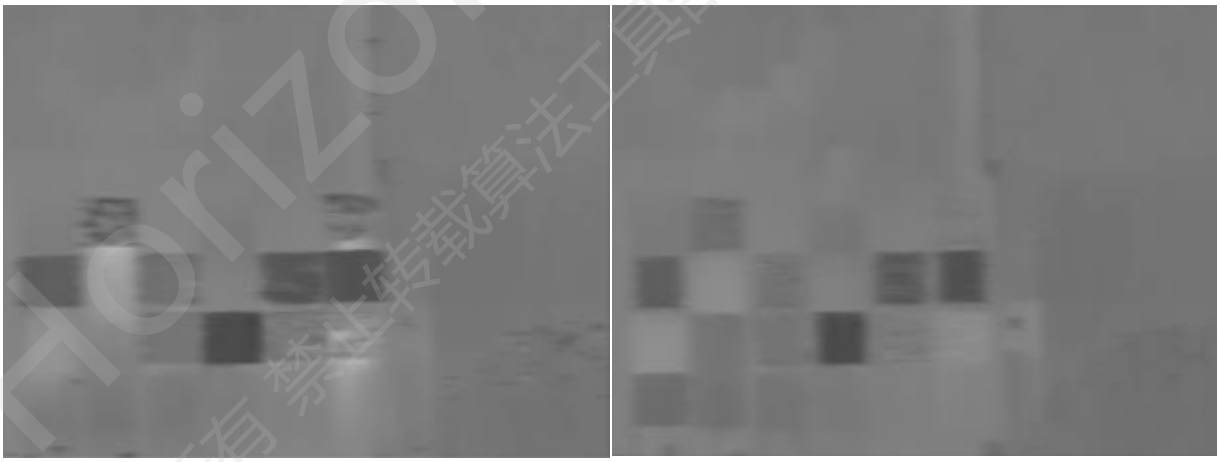


(a) Umean1_offset too low. (b) Umean1_offset too high.

Figure 3.18-3 Tuning Umean1_offset.

7. After producing an initial tuning for the Umean1 mask, check that the Umean2 mask meets the same criteria. First set Umean2_offset to the same value as Umean1_offset, and set Umean2_slope to the same value as Umean1_slope. Then set Debug_reg to *6*. It is common for the Umean2 mask to produce the vertical smearing behavior demonstrated in Figure 3.18-4a (seen as wide white streaks beneath patches). If this is visible, Umean2_offset is too high.
8. Once Umean1 and Umean2 masks meet the criteria with the same offset and slope value, copy those values to Vmean1_offset, Vmean1_slope, Vmean2_offset and Vmean2_slope.

Tuning of [U/V]mean[1/2] masks should now be complete and look similar to Figure 3.18-4b.



(a)Umean2_offset too high. (b)Umean1_offset tuned

Figure 3.18-4 uning [U/V]mean[1/2]_offset.

9. With [U/V]mean[1/2] masks within specification, move onto UV_var masks. These require no tuning as such, but the values shown in Table 3.18-7 need to be set.

Register	Value

UV_var1_scale	16
UV_var2_scale	16
UV_var1_slope	65280
UV_var2_slope	65280
UV_var1_threshold	0
UV_var2_threshold	0
UV_var1_offset	1023 (4095)
UV_var2_offset	1023 (4095)

Table 3.18-7 CNR initial UV_var register values.

The UV_var1 mask can be viewed by setting Debug_reg to 8, which should look similar to figure

Figure 3.18-5 UV_var1 tuned.

10. Now UV_var meets the criteria, move onto UV_delta masks. Start by setting the values seen in Table 3.18-8.

Register	Value
Debug_reg	12
UV_delta1_offset	0
UV_delta2_offset	0
UV_delta1_slope	65535
UV_delta2_slope	65535
UV_delta1_threshold	0
UV_delta2_threshold	0
Delta_factor	307 (1229)

Table 3.18-8 CNR initial UV_delta register values.

These masks dictate which pixels will be corrected, and by how much. In the tuned masks, both offset values will match each other, while both slope values will be matching, resulting in only two parameters requiring tuning.

11. Simply increase UV_delta1_offset until only the very edges of some of the patches are shown in the ColorChecker, as seen in Figure 3.18-6.

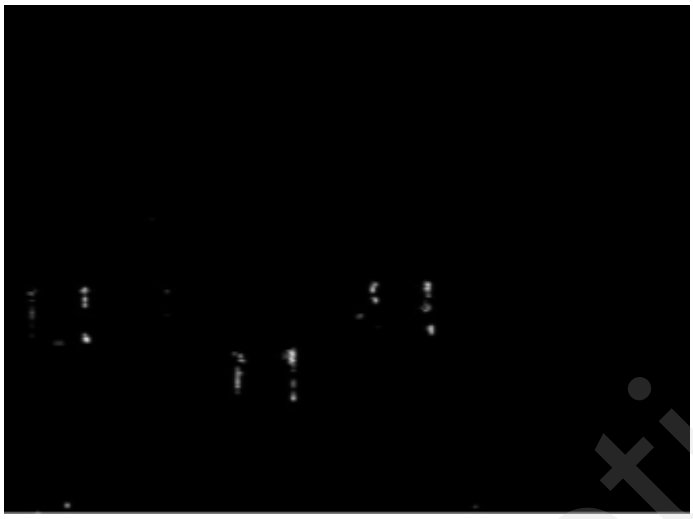


Figure 3.18-6 Figure 98: UV_delta1_offset tuned

Now set Debug_reg to *13* and tune UV_delta2_offset in the same way. Both offsets much match, so if (for example) UV_delta1_offset is *120* and UV_delta2_offset is *130*, set both to *125*. 12. With UV_delta1_offset and UV_delta2_offset tuned, set Debug_reg to *0* in order to view the output image itself. There may be some residual smearing effect, where areas take on the color of a neighboring region or become desaturated. This is called *color bleeding* and can be addressed by increasing Delta_factor.

13. Now set UV_delta1_slope and UV_delta2_slope to *0* so there is no noise reduction, then increase it to the point that colour noise begins to disappear. If slope values are too high, false colors will start to appear around the edges of some ColorChecker patches. These parameters (UV_delta1_slope and UV_delta2_slope) are used to control the strength of the color noise reduction according to the total gain of the system (log2gain). This is done through Calibration_cnr_uv_delta12_slope found in dynamic_calibrations.c as demonstrated here: Calibration_cnr_uv_delta12_slope[][2] ={{0,800},{1*256,800},{2*256,800},{3*256,1300},{4*256,1300},{5*256,2133}};In each value pair, the first entry is the gain, and the second value is both UV_delta1_slope and UV_delta2_slope. Enter the tuned slope value at the current gain. Phase two of SNR tuning is now complete, so wait until phase three to fine tune CNR.

3.18.4 Fine tuning CNR phase three

Repeat step 13 from phase two tuning, adjusting the lighting conditions to suit different gain levels. Once the full range of gains is accounted for in Calibration_cnr_uv_delta12_slope CNR tuning can be classed as complete.

3.19 RGB sharpening

3.19.1 Overview

Commonly known as back end sharpening, this module is designed to work with the demosaic

sharpening block. Demosaic sharpening should be used to achieve the required resolution, but aggressive sharpening can lead to artifacts and unnatural looking textures. The aim of this module is to perceptually reduce artifacts while providing a sharp image. A brief outline of the tuning process is described below:

- During phase one - No tuning required
- During phase two - RGB sharpening tuned in lab conditions for the majority of scenes.
- During phase three - No tuning required, unless previous tuning or other modules which RGB sharpening is dependent upon causes poor performance

3.19.2 Tuning Theory

In the RGB sharpening core, it is possible to control the following:

- Undershoot
- Overshoot
- Sharpening application within specific intensity levels

Key hardware registers

Hardware registers are found on the relevant source page listed in the "Hardware" tab of Control Tool

Register name	Source	Description
Enable	Sharpen	Enable/disable RGB sharpening
Strength	Sharpen	Sharpening strength control
Control_r	Sharpen	luma transform red coefficient
Control_b	Sharpen	luma transform blue coefficient
Alpha_undershoot	Sharpen	Alpha blend of under/overshoot,0 = only undershoot, 255 = only overshoot
Luma_threshold_low	Sharpen	Luma threshold where below this value, no sharpening will be applied
Luma_offset_low	Sharpen	luma offset (min value) after linear thresholding
Luma_slope_low	Sharpen	luma linear threshold slope
Luma_thresh_high	Sharpen	Luma threshold where above this value, no sharpening will be applied
Luma_offset_high	Sharpen	luma offset (min value) after linear thresholding
Luma_slope_high	Sharpen	luma linear threshold slope

Clip_str_max	Sharpen	clips sharpening mask max value, this will control overshoot. Note that this value will be taken as a negative value in HW
Clip_str_min	Sharpen	clips sharpening mask min value,this will control undershoot

Table 3.19-1 RGB sharpening hardware registers

3.19.3 Tuning during phase two

RGB sharpening is first tuned during the early stages of phase two as it is not required in phase one. Start by making sure the prerequisites listed in Table 3.19-2 are completed

Prerequisite	**Status / value**
Phase 1 modules	Tuned
DPC	tune
Sinter	Tuned
Temper	Tuned
Demosaic	Tuned
Set Enable	1

Table 3.19-2 RGB sharpening phase two prerequisites

RGB sharpening phase two prerequisites：

Register name	**Default value**	**Format**
enable	1	U1.0
strength	16	U5.4
control_r	76	U1.8
control_b	30	U1.8
alpha_undershoot	10	U1.7
luma_thresh_low	300	U0.10
luma_slope_low	1000	U8.8
luma_offset_low	0	U0.8
luma_thresh_high	1000	U0.10
luma_slope_high	1700	U8.8
luma_offset_high	0	U0.8

clip_str_max	280	U10.4
clip_str_min	280	U10.4

Table 3.19-3 RGB sharpening default values

Tuning procedure - Sharpening

Adjust Strength of sharpening filter to make image sharpness reach certain objective target, for example, MTF50, under and overshoot. This sharpening operation in the fine tuning phase can be done subjectively as well to make the image look natural and sharp. The sharpening strength is modulated by gain in dynamic calibration c file in format: [gain, strength], example lut is as below.

```
static uint16_t _calibration_sharpen_fr[][2] = {
{0 * 256, 42},
{1 * 256, 27},
};
```

Tuning procedure - Under/overshoot

Two filters are used in the sharpening logic to boost either undershoot or overshoot on edges. The output of these two filters is blended before applying the final sharpening. The Alpha_undershoot register is the coefficient between these two filters. A value of 0 will totally bias the logic towards an undershoot response, where a value of 255 favors totally towards overshoot.

Clip_str_min and Clip_str_max registers can also be used to control under and overshoot, but these act as a clipping threshold as shown in Figure 3.19-1

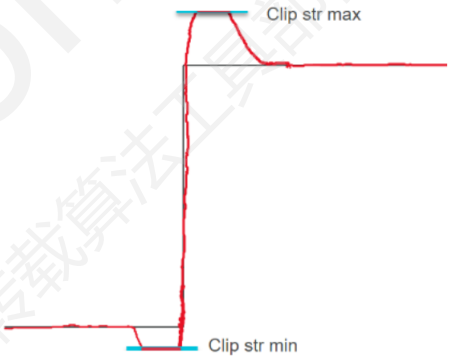


Figure 3.19-1 Min max clipping controls.

Tuning procedure - Luma threshold

The RGB sharpening block provides logic to threshold the sharpening effect according to intensity. Figure 3.19-2 illustrates the relationship of the luma parameters. It is highly recommended to always use the luma threshold logic as it will ensure that sharpening does not clip the pixels in highlights and noise in dark regions is not revealed.

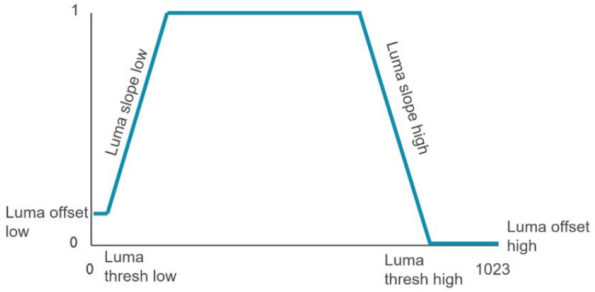


Figure 3.19-2 Luma RGB sharpening thresholding

4 Using Wide Dynamic Range（WDR）modes

4.1 Decompander

4.1.1 Overview

For some WDR sensors or sensor modes, the output image is compressed by linear compression or other compression methods, for example, from 20-bit to 12-bit by piece wise linear compression. The decompander block can be used to decompand the input image to linear. This operation is implemented by decompander LUT0, LUT1 in the figure blow.

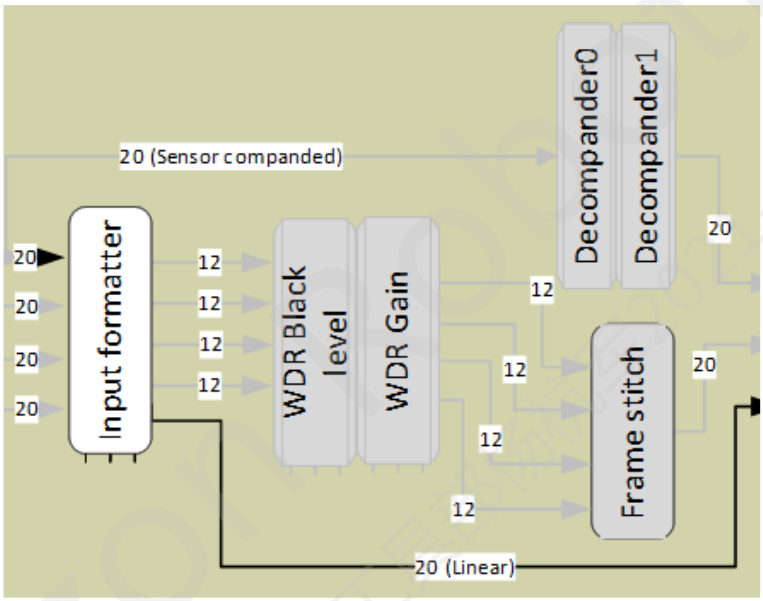


Figure 4.1-1 A diagram to show the decompander block.

4.1.2 Functional description

The decompander LUT consists of a 33-node LUT, cascaded with a 257-node LUT. This allows user to improve the precision in critical parts of the curve. For instructions to generate the decompander LUTs, refer to the documentation for the “gamma FE” module of ISP calibration tool.

4.1.3 Tuning in phase one

4.1.3.1 Gamma FE window

The purpose of the Gamma FE module is to calibrate the hardware block of the same name (Gamma FE). This block converts the pixel values from the sensor into the data format that is expected by the ISP. The module is available in 3 ISP modes, and the user interface changes, depending on the mode:

Mode	Input format	Output (target) format
FS-HDR	12-bit linear	12-bit square root curve
Native-WDR	12-bit companded (e.g. piecewise linear)	12-bit square root curve
Native-WDR-Lin	12-bit companded (e.g. piecewise linear)	24-bit linear

Pixel values are converted from the input format to the output format using one 4097-node LUT, or a 33-node LUT (lut0) followed by a 257-node LUT (lut1).

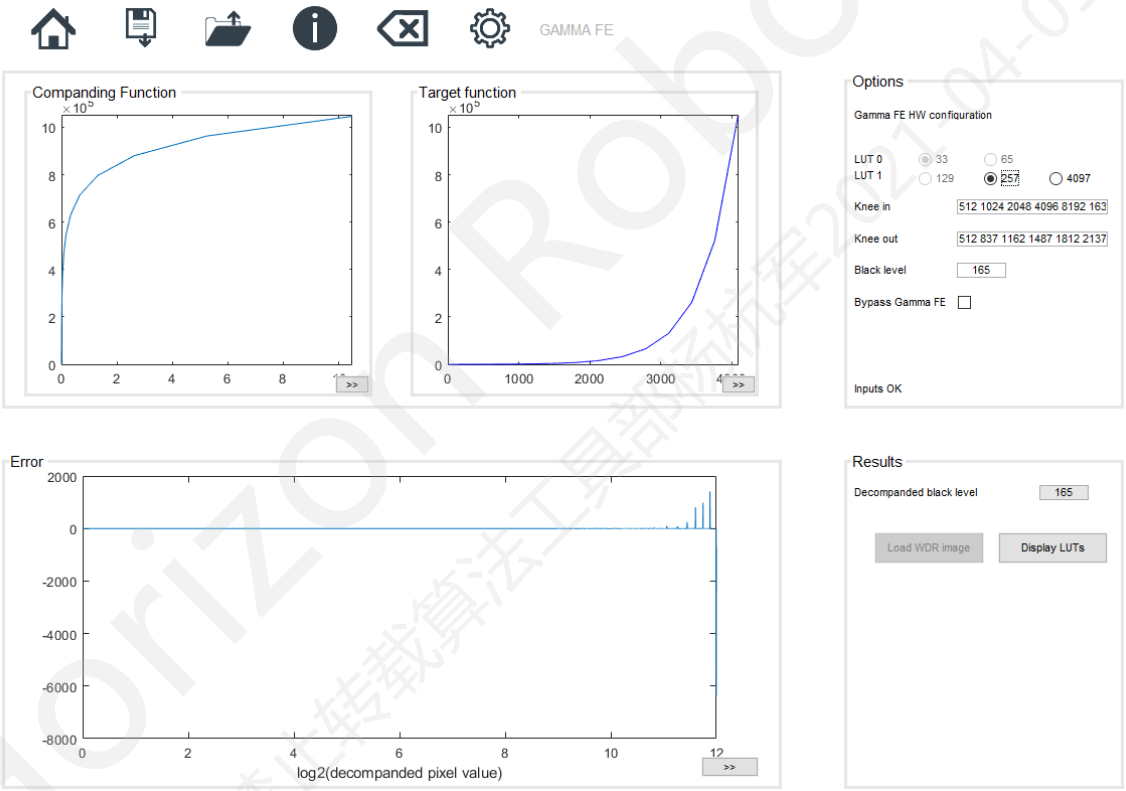


Figure 4.1-2 Gamma FE window for Native-WDR-Lin mode

4.1.3.2 Window features

Table 4.1-1 lists the function of each region.

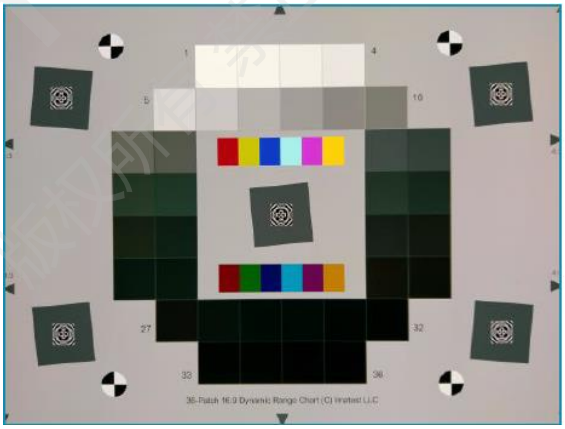
Region	Sub-region name	Function
Companding	-	Plot of sensor companding function

Function		based on knee point data
	>> *button*	Undock current plots from window for more freedom
Target Function	-	Plot of decompanding function
	>> *button*	Undock current plots from window for more freedom
Error	-	Plot of precision error, compared with a perfect 12 to 20 bit decompander
Options	LUT 0	The number of hardware nodes in LUT 0
	LUT 1	The number of hardware nodes in LUT 1
	Knee in	20-bit input knee values
	Knee out	12-bit output knee values
	Black level	Calibrated offset for 1x gain from black level module
Results	Pedestal	Displays the adjusted black level after decompanding
	Load WDR image	Load a WDR image for verification (save calibration to enable this feature)
	Display LUTs	Pop up table showing LUT data

Table 4.1-1 List of all regions and their functions in the gamma FE window

4.1.3.3 Calibration procedure

No images are required for calibration of the gamma FE module. However, it's recommended to have some WDR images on hand to verify the calibration. These can be lab of field based images, but a mixture of both is preferable (see Figure 4.2-3). To view images, calibrated the black level module first.



a) – ITDR 36 patch test chart (b) – Outdoor HDR scene

Figure 4.1-3 Examples of HDR imagery – (a) lab-based (b) field-based

Note: HDR imagery does not require the specific images shown in Figure 4.2-3, this simply shows two HDR options. More generally, HDR images should include areas of extreme darks and highlights. Hence, an alternative HDR lab image could be of a studio scene which includes an illuminant, such as a small lamp.

1. Open the gamma FE calibration module from the home window.
2. Use the "Lut 0" and "Lut 1" radio buttons to match the number of nodes in each LUT to the hardware configuration. Refer to ISP hardware documentation to find the correct number of nodes.
3. (Native-WDR and Native-WDR-Lin ISP mode only) – Set the knee points of the sensor as space-separated decimal values.

   Knee point values can normally be found in the sensor datasheet.
4. (Native-WDR and FS-HDR ISP mode only) Use the plots in the black level module to set "Min Signal Level" to match the arrow shown (see Figure 4.1-4):

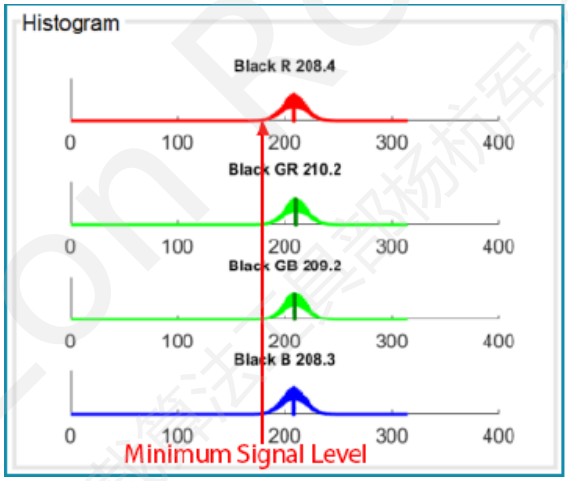


Figure 4.1-4 Black level calibration data.

5. LUT1 will be updated every time you change settings, but to minimize the error (Native-WDR and FS-HDR ISP modes only), click "Calculate LUTs". The LUTs will be optimised – it may take a few minutes.
6. (Optional) – Alternatively, LUTs can be manually set by selecting the "Display LUTs" option. The window shown in Figure 4.2-5 will pop up, and you can manually adjust the LUT values.

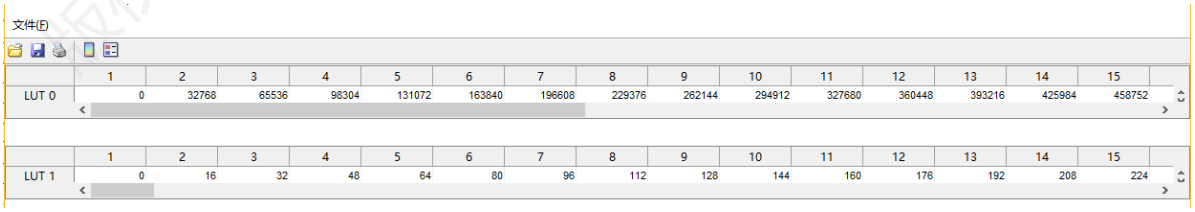


Figure 4.1-5 Finual LUT adjustment window.

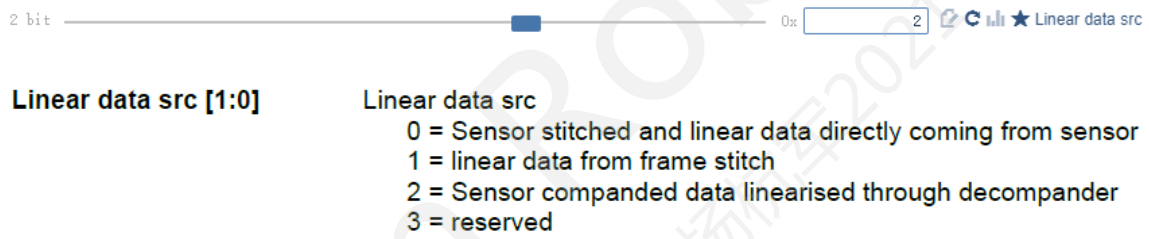
7. Save the parameter values using the save button. Return to the home window.
8. Using the “Load WDR image” will help to visually verify the chosen *gamma FE* settings.

Note: In FS-HDR and Native-WDR modes, the “Target Function” is a square root function based on the black level and minimum signal. This curve is used as the “ideal”, and the FE LUTs are calibrated to match it.

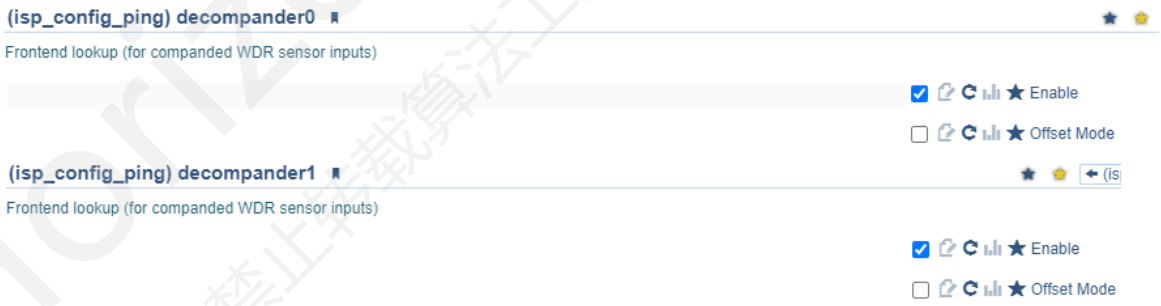
4.1.4 Tuning in phase two

In this mode the input data must be sent on channel-0 and must be MSB aligned. The crossbar must be in the default configuration so that the input data is directly mapped to its same output channel. The input formatter is bypassed. The decompander-0 (33 nodes) and decompander-1 (257 nodes) are programmed to create an equivalent log curve. The linear data src register is programmed appropriately to select the decompanded data.

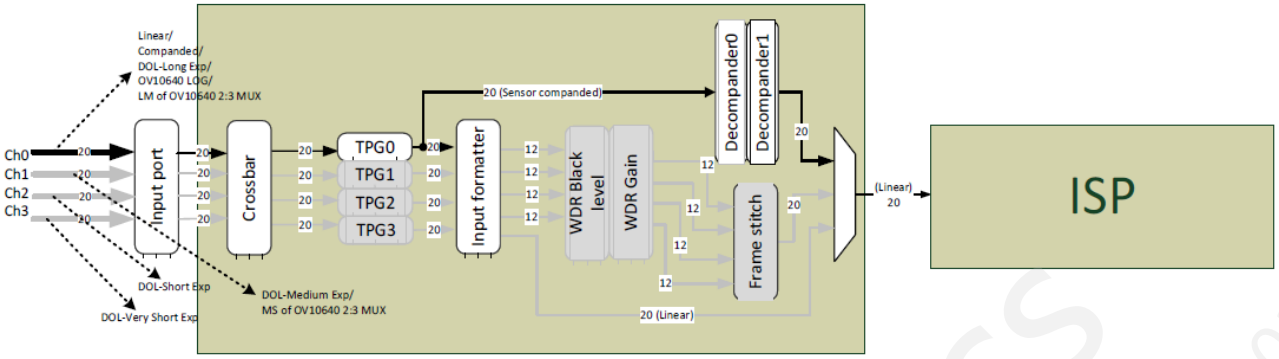
- Sensor mode select



- ISP module enable/disable in control tool：



In the following figure the white blocks indicate the active blocks and dark black lines indicate the actual signal flow. The grey blocks and signal lines are inactive.



4.2 Frame Stitching

4.2.1 Overview

WDR exposure stitching is used to combine two, three, or four *temporally-proximal* frames of different exposure length. This produces a single frame of greater precision than the either of the original frames individually.

Fundamentally, the idea is that the short exposure can be used to avoid clipping, while the long exposure can be used to avoid excessive noise in darker regions, and thus increase the dynamic range of the camera system.

4.2.2 Tuning theory

Key firmware parameters for dynamic calibrations

Dynamic calibrations are found both in the “dynamic_calibrations.c” file, and on the relevant page listed in the “Dynamic Calibrations” tab of Control Tool.

Parameter name	Description
calibration_stitching_ [lm/ms]_np	coefficient multiply to noise level, which will control start point of the ramp for motion detection. modulation lut - [gain,] calibration_stitching_[lm/ms]_np
calibration_stitching_ [lm/ms]_mov_mult	control the slope gradient of motion detection ramp. modulation lut - [gain,calibration_stitching_] [lm/ms]_mov_mult
calibration_stitching_lm_ med_noise_intensity_thresh	threshold to judge if pixel from medium exposure is dark. If intensity of pixel from medium exposure is lowerthan this threshold,in exposure mask, this pixel will be mark as dark medium, the de-noising strength will be controlled by noise level 3 in sinter noise profile.

	modulation lut - [gain,calibration_stitching_lm_med_]noise_intensity_thresh
calibration_fs_mc_off	if current gain is higher than this gain threshold, motion compensation (mc) off mode will be enabled.

Table 4.2-1 Frame stitching dynamic parameters

The modulation example is as below :

```
static uint16_t _calibration_stitching_lm_np[][2]= {
{0, 16},
{1 * 256, 16},
{2 * 256, 16},
{3 * 256, 55}
};
static uint16_t _calibration_stitching_lm_mov_mult[][2] = {
{0,0} // 0 will disable motion
{0 * 256, 200}, //
{1 * 256, 400}, //
{1664, 300}, // 6.5
{7 * 256, 250}, //
{8 * 256, 135}, //
};
static uint16_t _calibration_stitching_lm_med_noise_intensity_thresh[][2] = {
{0, 32},
{6 * 256, 32},
{8 * 256, 4095},
};
static uint16_t _calibration_stitching_ms_np[][2] = {
{0, 3680},
{1 * 256, 3680},
{2 * 256, 2680},
};
static uint16_t _calibration_stitching_ms_mov_mult[][2] = {
{0, 128},
```

{1 * 256, 128},

{2 * 256, 128},

};

static uint16_t _calibration_fs_mc_off[] = {

8 * 256, // gain_log2 threshold. if gain is higher than the current gain_log2. mc off mode will be enabled. }

[lm/ms] indicates that there are actually two parameters present. One for "long-medium" exposure (lm) and one for "medium-short" exposure (ms).

[lm/ms]_thresh_high should always be greater than the respective [lm/ms]_thresh_low. It should also be less than 1 − Black_level_long for lm, and 1 − Black_level_medium for ms.

For αmov, setting a value of 0 will fully disable motion compensation. The default value is 128, where increasing this will bias compensation towards the medium/short exposure.

Functional description

A demonstration of two exposures (one long, one short) can be seen below in Figure 4.2-1. It is assumed that the sensor has a linear response and thus the ratio "R", between integration times is the same as the ratio between the gradients of the responses.

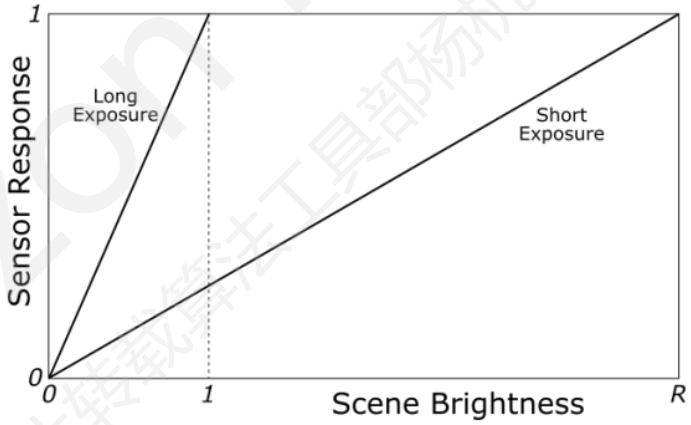


Figure 4.2-1 A graph to show the intensity response of two exposures.

To create a simple stitch of these two exposures, first divide the long exposure by the exposure ratio "R" to align the two exposures. Each pixel value is defined by comparing the value of the longer exposure against a threshold "t", this identifies which of the two exposures should be assigned.

$$t < \frac{1}{R}$$

For values below the threshold the (scaled) long exposure is selected and for values above, the short exposure is selected. Thus the image is comprised from the longer exposure in dark regions,

while the brighter regions are acquired from the short exposure. In practice, this introduces artifacts caused by transitioning suddenly between pixels from different exposures. To avoid this, two thresholds, "t1" and "t2" are selected and the output result is an alpha-blend between the two exposures as seen in Figure 4.2-2.

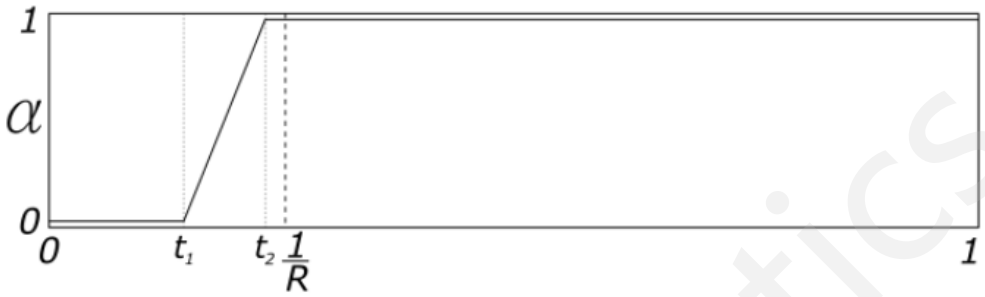


Figure 4.2-2 A plot to show the alpha-blend between the two exposures

Two major issues remain. Firstly, the RAW data is Bayer data, and so a pixel's intensity depends on local color. The higher the saturation of a region, the greater the disparity between pixel values form different color channels. Selecting pixels based purely on their values will result in the pixel in saturated regions being selected from different exposures depending on their channel. For example, green from the short exposure and red and blue from the long exposure. This will lead to color artifacts in the de-mosaicked image.

To avoid color artifacts, pixel selection should be based on the local intensity surroundingthe pixel.

The second major issue is motion, as when a scene contains motion or the camera is moving, there will be disparity between consecutive exposures. In this case, pixels at the same sensor location may have very different values between the two exposures after scaling has been applied. Applying the simple stitching method described above will result in image artifacts such as discontinuities and objects being repeated.

Below is an example of the artifacts that occur when there is motion between stitched exposures. The stitched result was produced using the simple method described above.

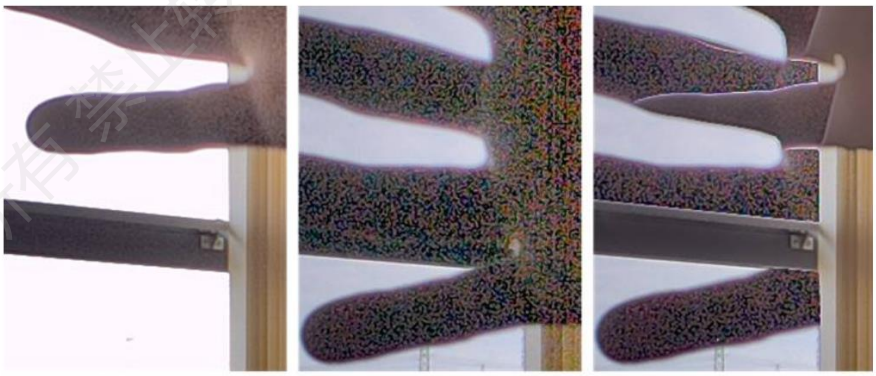


Figure 4.2-3 Example of possible artifacts

4.2.3 Detection motion

Motion can be detected by examining the differences between pixel values once exposures have been scaled according to exposure ratio. If the difference is higher than estimated noise level, which is n (xl - xs) in the figure of motion detection and calculated based on noise profile, this pixel will be regard having certain level motion. Increasing calibration_ stitching_[lm/ms]_np will increase n(xl - xs) start point of the ramp, which will make motion detection less sensitive. Increasing calibration_stitching_[lm/ms]_mov_mult will increase the slope of the ramp, which will make the motion detection more sensitive. If calibration_stitching_[lm/ms]_mov_mult is set to 0, then motion detection is disabled.

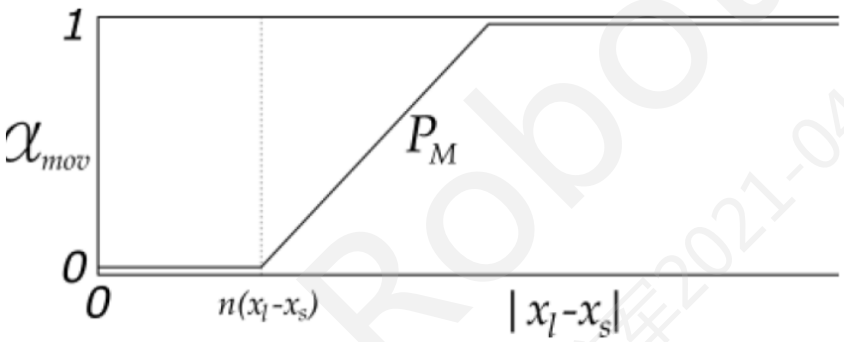


Figure 4.2-4 motion detection

4.2.4 Dealing with motion

Having detected motion, there are two ways in which to attempt modification of the alpha map used to blend the two exposures:

- Always use the short exposure in regions of motion.
- Always use the long exposure in regions of motion.

The advantage of falling back to the short exposure in regions of motion is that is is less likely to have motion blur, the disadvantage is that the short exposure will have much more noise. In darker areas, the short exposure may be so noisy that there is no real detail present and so this approach will lead to ugly, noisy black regions.

Alternatively, the advantage of falling back to the long exposure is that it is much less noisy. However clipped pixels must be dealt with in two ways. Firstly, when detecting motion the fact that they are clipped must be taken into account by modifying the motion ramp equation as previously described. Failure to do so will result in sufficiently bright regions always being confused by motion.

Secondly, if clipped pixels are directly used, they will result in the final image having incorrect colors - usually a shift towards magenta. To understand why this is, imagine a very bright region in which all three color channels are clipped in the long exposure. Later in the pipeline, white

balance correction will be applied by a multiplication (greater than 1) of red and blue channels. The result gives a magenta color where white/light gray should be present.

This can be accounted for by preempting the white balance correction, and increasing the the green channel accordingly. Though the magenta shift can be corrected for, it is not always possible to recover the true color of the region. Hence, the information which has been lost due to clipping may not be recovered.

Whichever exposure is used to replace motion has a penalty for false positives in motion detection. The first will introduce extra noise by falling back to the short exposure. The second will remove detail and potentially introduce false colors by using clipped pixels form the long exposure.

When the current gain (log2gain) is higher than gain value in calibration_fs_mc_off lut, MCoff_NC_Enable register will be enabled, which means mc off mode is enabled, then motion compensation mode is off. This mode should be use when it is impossible to distinguish noise and motion, usually at very low light levels. The difference between mc on and off is that in mc off, motion regions will be an average of all exposures, where as in mc on mode, the motion regions will be compensated by taking the shorter exposure.

4.2.5 Tuning Frame Stitching Block

To tune the frame stitching block, a scene like Figure 4.2-5 with a motion/rotation object inside should be set-up. Using long medium short 3:1 exposures stitching as example: start from calibration_stitching_lm_mov_mult set to 128, calibration_stitching_lm_np set to 1, if the motion artefact is obvious, increase calibration_stitching_lm_mov_mult to the point motion is detected. If set too high, noise could be considered as motion and thus the image SNR will drop. To compensate for the noise, adjust the calibration_stitching_lm_np until a trade off point between noise and stitching artefacts in motion areas is reached. Repeat this process for all gain of the system. As the noise increases, decrease calibration_stitching_lm_mov_mult to keep the motion detection similar to previous gain, and increase calibration_stitching_lm_np to reduce the probability of the motion detection logic being fool by noise. Follow the same procedure for medium short exposures stitching case.

Figure 4.2-5 A figure to show the frame stitching block tuning

5 Lens Distortion Correction (LDC)

5.1 J3 LDC introduction

LDC (Lens distortion correction) In J3, in order to reduce the DRAM bandwidth requirements, a special module LDC is used to correct the commonly used small angle (less than 10%) barrel distortion. These modules use J3's IRAM (IternalRAM) for caching, the cache that can be allocated for distortion depends on the remaining cache size of IRAM. A simple visualization tool is also provided to assist users in evaluating cache requirements and previewing correction results.

5.2 Tuning Overview

5.2.1 Supported modes

	Operation mode	Bandwidth requirements	Calibration mode
LDC	Online	None	Only barrel distortion，limited by the size of the iram that can be allocated and the formula, and fov will be lost

5.2.2 Tuning Tools

LDC visual calibration tool (LDC GUI Tool)

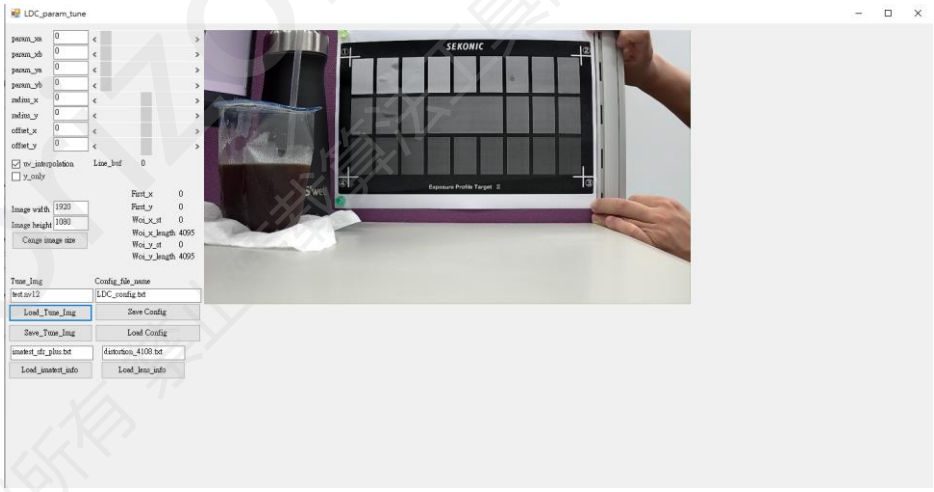


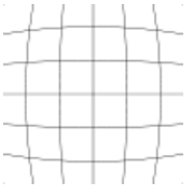
Figure 5.2-1 LDC Calibration Tool

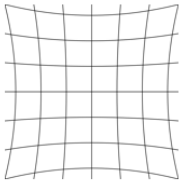
5.3 LDC tuning process

5.3.1 Overview

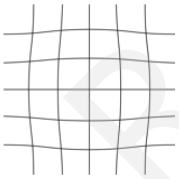
Lens distortion can be divided into following types

- Barrel Distortion and Pincushion Distortion
  - Increase with the distance from the center R

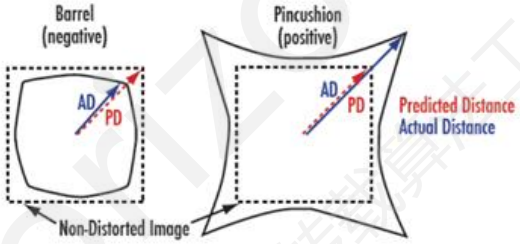


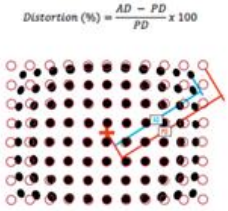


- Mustache Distortion



1. The deformation of barrel distortion and pincushion distortion can be measured by the following formula





2. Information about Lens distortion can also be found on the following website
   https://www.imatest.com/docs/distortion-methods-and-modules/
3. Limitation:
   1) Only supports Barrel correction (barrel correction)
   2) If a black border appears in the preview, the hardware does not support this operation
   3) The maximum correction capability is limited by the allocable size of IRAM
      i. Maximum IRAM 1280KB (If preview image is exceeded, the color will be abnormal, shown below)

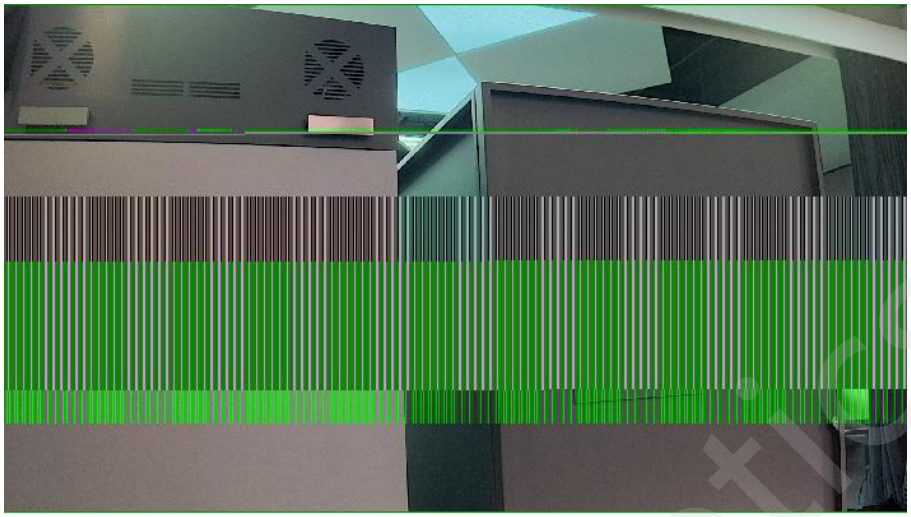


Figure 5.3-1 Abnormal picture caused by excessive buffer line

ii. The number of Line buffer must be a multiple of 2

iii. The pictures stored in buffer and used for correction are in nv12 format

5.3.2 Key parameter

5.3.2.1 Param position

Key parameters can be modified in the screenshot below (for example, ar0231), which takes effect at setup. The specific reference is as follows:

```
root@x3dvbx3-hynix1G-2666:/etc/vio_tool# ls
ar0144_raw_12bit_1280x720_offline_Pipeline.json   dump_raw.json                          os8a10_raw10bit_2k_dol2.json
ar0144_raw_12bit_1280x720_online_Pipeline.json    dump_yuv.json                          os8a10_raw10bit_2k_dol2_offline.json
ar0231_raw_12bit_1920x1080_offline_Pipeline.json  hb_x3player.json                       os8a10_raw10bit_4k_dol2.json
ar0231_raw_12bit_1920x1080_online_Pipeline.json   imx327_raw_12bit_1952x1097_dol2.json    os8a10_raw10bit_4k_dol2_offline.json
ar0233_raw_12bit_1920x1080_offline_Pipeline.json  imx327_raw_12bit_1952x1097_dol2_offline.json  os8a10_raw10bit_4k_linear.json
ar0233_raw_12bit_1920x1080_online_Pipeline.json   imx327_raw_12bit_1952x1097_offline.json  os8a10_raw10bit_4k_linear_offline.json
dump_config.json                                  imx327_raw_12bit_1952x1097_online.json
root@x3dvbx3-hynix1G-2666:/etc/vio_tool#
```

Addition, control tool page parameter interface as follows

Note: If you want to manually modify the parameters on the control tool, you need to fill in the parameters you want to modify, click 'updata' to update and take effect together.

5.3.2.2 Param description

It can be modified under the LDC_PARAM under the API of the control tool or in the configuration file.

Parameter name	Description
LINE_BUF	Number of Line buffers used for calculation, min value is 0
X_PARAMTER_A	X axis direction coefficient A; [15] sign bit, [0-14] unsigned data
X_PARAMTER_B	X axis direction coefficient B; [15] sign bit, [0-14] unsigned data
Y_PARAMTER_A	Y axis direction coefficient A; [15] sign bit, [0-14] unsigned data
Y_PARAMTER_B	Y-axis direction coefficient B; [15] sign bit, [0-14] unsigned data
RADIUS_XOFFSET	Radial X axis deflection; signed number (complement of 2), [-128-127]
RADIUS_YOFFSET	Radial Y axis deflection; signed number (complement of 2), [-128-127]
CENTER_XOFFSET	Center coordinate X axis displacement; signed number (complement of 2), [-128-127]
CENTER_YOFFSET	Center coordinate Y axis displacement; signed number (complement of 2), [-128-127]

5.3.3 Tuning process

1. Manual mode: Use the calibration tool (offline GUI Tool), manually adjust to achieve satisfactory results, and then throw the resulting image to imatest to test the degree of distortion (need to be converted into a format recognized by imtest such as png or jpg), view the correction the result of.
2. Correction data 1: Using the distortion module to dump SFRPlus chart, and get the parameter settings for calculation by imatest. The settings can be written to GUI_tool to verify the accuracy of the parameters and the line buffer.
3. Correction data 2: Obtain the lens distortion parameters from the lens factory, import them into GUI_tool according to the format, and then get the parameter settings and line buffer.

5.3.3.1 GUI tool user guide

1. User Interface – mainly function

1> Left pane:

- Dump and save picture
- Dump and save param
- Set the param
- Reference data

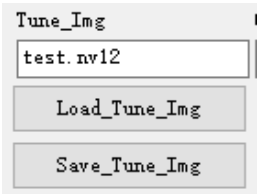
2> Right pane:

Preview block

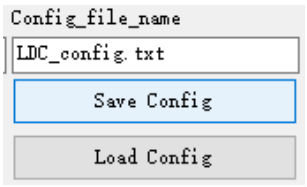
2. Left pane:

1> Image size setting

2> Dump or save picture：



3> Set or save param：

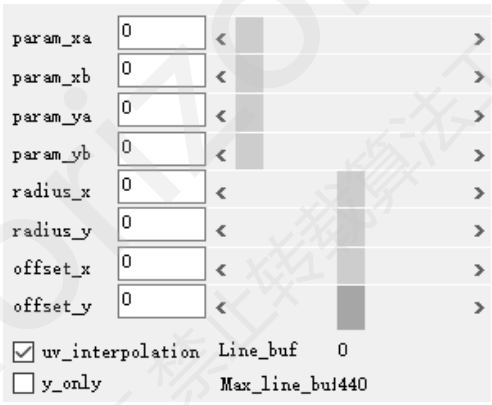


Format of setting param：

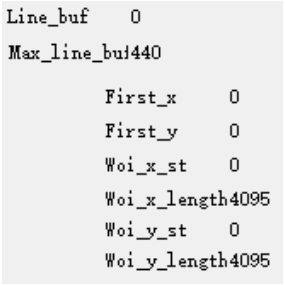
```
Param_xa=256
Param_xb=256
Param_ya=64
Param_yb=0
Radius_x=0
Radius_y=0
Offset_x=0
Offset_y=0
Uv_interpolation=1
Y_only=0
```

4> Param setting block：

You can see the result immediately when you modify the param

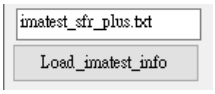


5> Reference data：



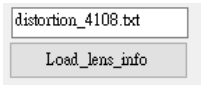
6> Extract SFRPlus test results

Refer to the screenshoot for details in GUI tool



7> Extract lens distortion data

Refer to the screenshoot for details in GUI_tool



3. Right pane:

Preview interface of calibration result, preview interface will change in real time when param modified.

4. Position of test result and data

Placed in the same directory as GUI_tool

5.3.3.2 Imatest SFRPlus chart guide

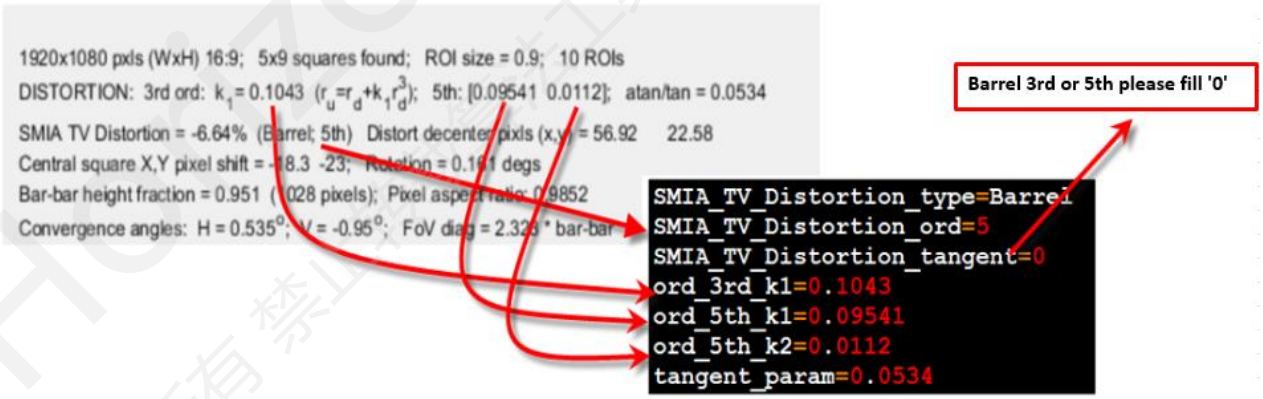
1. Take a PNG or jpg picture of the normally recognizable SFRPlus chart.
2. Use IMATEST SFRPlus AUTO test

(Instructions :https://www.imatest.com/docs/sfrplus_instructions/)

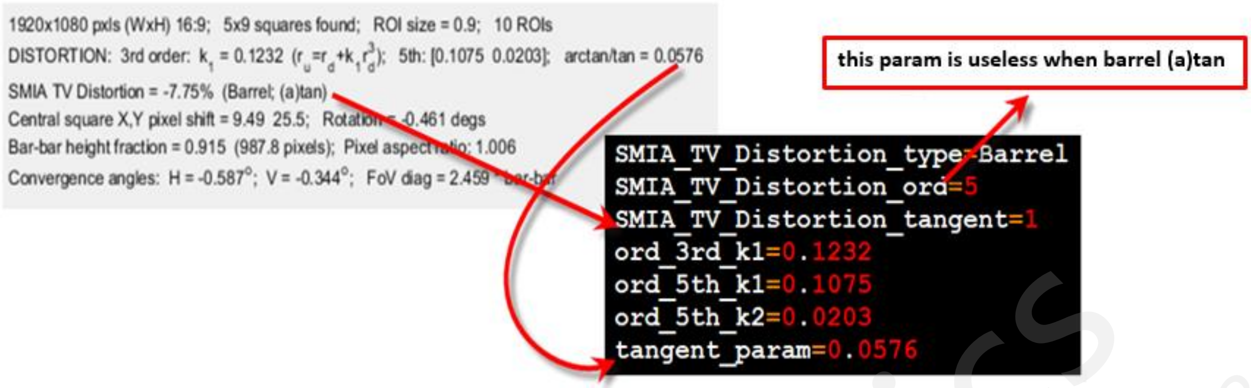
3. Dump requirements
   i. Pixel shift, rotation, adjust to 0
   ii. Distort decenter pixel, adjust to 0
   iii. Charts should fill the picture, but do not lose the edges

4. Fill in the .txt according to the parameters tested by imatest

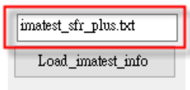
   Example 1 (polynomial solution):



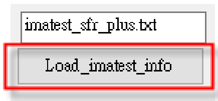
   Example 2 (tangent function solution):



5. Modify the name of the .txt file to be extracted



6. Press "Load_imatest_info" to generate calibration data and results



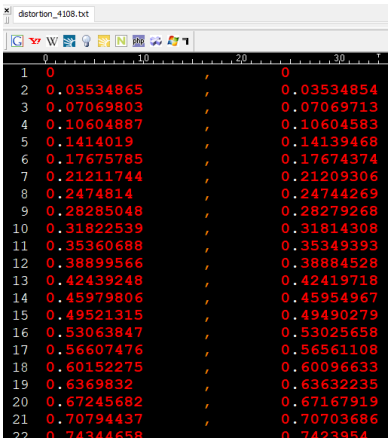
7. If the correction result of GUI_tool is satisfactory, apply the correction parameters to the control tool online mode.

5.3.3.3 Use Lens information guide

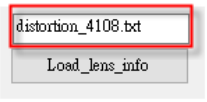
8. Obtain the following data from Lens factory

4083		4108	
		過鏡頭變形後	變形前
Real Height (mm)	Ref. Height (mm)	Real Height (mm)	Ref. Height (mm)
0.00000000	0.00000000	0.00000000	0.00000000
0.03535940	0.03535991	0.03534854	0.03534865
0.07071807	0.07072221	0.07069713	0.07069803
0.10607531	0.10608928	0.10604583	0.10604887
0.14143038	0.14146351	0.14139468	0.14140190
0.17678257	0.17684728	0.17674374	0.17675785
0.21213114	0.21224299	0.21209306	0.21211744
0.24747538	0.24765304	0.24744269	0.24748140
0.28281454	0.28307981	0.28279268	0.28285048
0.31814789	0.31852571	0.31814308	0.31822539
0.35347469	0.35399316	0.35349393	0.35360688
0.38879419	0.38948456	0.38884528	0.38899566
0.42410565	0.42500234	0.42419718	0.42439248
0.45940830	0.46054892	0.45954967	0.45979806

9. Fill the above data into the .txt file in the following format



10. Modify the name of .txt to be imported



11. Press "Load_lens_info" to generate calibration data and results

12. If the calibration result of GUI_tool is satisfactory, apply the calibration parameters to the online mode of the control tool.

5.3.3.4 Calibration result

One certain 1080P sensor, distortion before correction is -7.61%, distortion after correction is -0.2%;

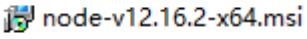
6 Geometric Distortion Correction (GDC)

6.1 J3 GDC introduction

This guide explains the basic functionality of the GDC Tool. The main purpose of this tool is to provide a user-friendly interface for configuring GDC (geometric distortion correction) transformations. This tool allows you to apply the configured transformations to a source image, returning and displaying the result. The tool also provides means for saving transformation parameters (as JSON file) and corresponding binary file containing the transformation sequence.

6.2 GDC Tool Installition

1. Install “node js”

   1) Find 

from the software package, double-click the file and start the installation.

   2) Click “next step”, the default installation path is C:\Program Files\nodejs.

2. Install component environment

   Open a command window and go to the directory: C:\Program Files\nodejs

   Execute the following command:

   npm install express -g

   npm install jade

3. Enter the directory: gdc-tool-gui-0.23.63-windows from command window, then run GDC program.

4. Enter the following server address: http://localhost:3000/ in your chome brewser to open the GDC user interface.

6.3 GDC user interface

The GDC tool’s user interface – the workspace – is divided into the following two principalareas:

- The left pane
- The right pane

Figure 6.3-1 GDC user interface

6.3.1 Left pane

The left pane contains all the controls for changing the image parameters at both project and region level. The left pane is divided into 5 sections: Input/Output/Settin，Regions，Process，Layout，Tranformation.

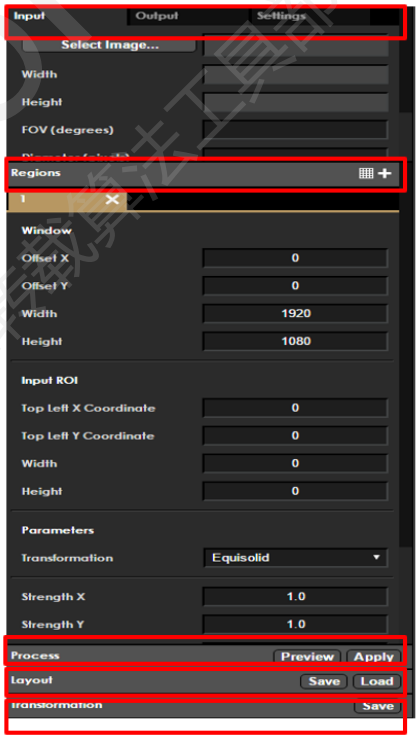


Figure 6.3-2 left pane

6.3.1.1 Input/Output/Setting

This contains controls for changing project level input and output settings.

6.3.1.2 Regions

This contains the controls for adding, removing and editing regions.

6.3.1.3 Process

This provides buttons for previewing and running the transformation described by the current project layout.

6.3.1.4 Layout

This provides a button for downloading the current layout, and for loading a layout file.

6.3.1.5 Transformation

This provides buttons for downloading the two binary files that can be generated from the project's layout.

6.3.2 Right pane

The right pane is divided into the following two sections:

- The input view (top)
- The output view (bottom)

6.3.2.1 The input View

The input view displays the current input image and overlays any preview grids that have been generated. It contains a button (lop left) for saving the current view as an image (including all grids).

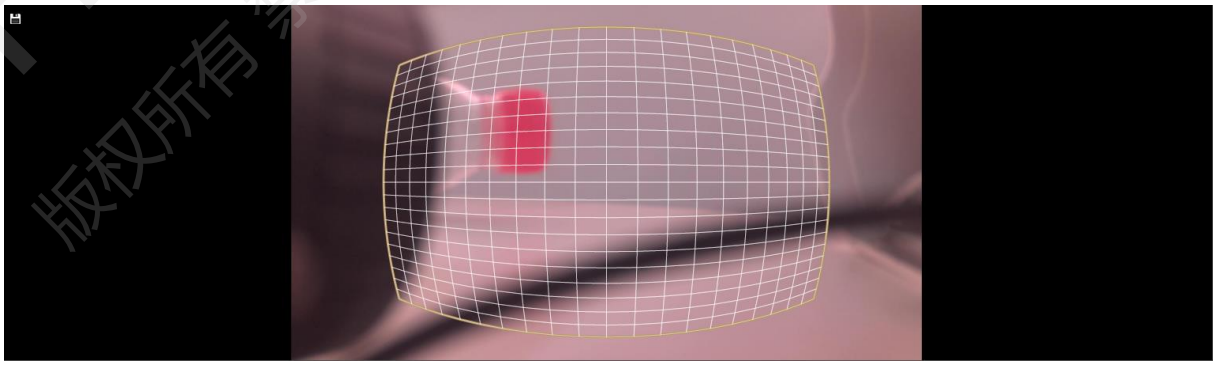


Figure 6.3-3 the input View

6.3.2.2 The output View

The output view displays the output windows for the regions that have been created by the user.

If the transformations have been applied then the region windows will be overlaid onto the output transformation image. This view also has a button for downloading the image but this image will only include the actual output image generated by GDC, not the region rectangles.

Figure 6.3-4 the ouput View

6.4 Input and Output setting

The input / output section of the sidebar contains controls for setting the input image and changing additional settings concerning the input and output image.

6.4.1 Input settings

To access the source image settings, click the 'Input' tab at the top of the sidebar. This will bring up the Input settings in the sidebar. To add a new source image, click 'Select Image' in the sidebar and select the file you wish to work with (currently only JPEG images are supported).

Once the file is uploaded you can change the settings of the source image using the following input boxes beneath the 'Upload Source' button:

- **FOV（Degree）**

The field of view defines the visible angle of the input image, affecting the curvature of the source grid. A bigger FOV causes a bigger perspective deformation.

- **Diameter**

Defines diameter in pixels of the input circular area containing the actual fisheye photo on the rectangular input image. For some cameras the diameter of this circular image area can be bigger or less than dimensions of the rectangular canvas (may be clipped sometimes).

- **Offset X（pixels）**

The input area offset in pixels from the top left corner along the X axis.

- **Offset Y（pixels）**

The input area offset in pixels from the top left corner along the Y axis.

6.4.2 Output setting

At the top of the sidebar click the 'Output' tab. In the section that opens inside the sidebar you will be able to change resolution of the output image. Normally you can leave this unchanged.

Figure 6.4-1 output Setting

6.4.3 Setting

At the top of the sidebar click the 'setting' tab. In the section that opens inside the sidebar you will be able to change format of the output image. Currently only supports YUV420 Semiplanar format, please select this option. The ECC selects ECC is disabled.

6.5 Regions

Each transformation that you set up has its own region within the output image, where the result of that transformation is projected. To start setting up a new transformation, the first step is to create its output region.

To create a new region, click the '+' button on the 'Regions' header in the sidebar. This add a new region to the output and resize all existing regions to the best fit for the resolution.

To remove a region, click the cross on the right of the region tab header in the sidebar.

To reset all region windows to default (best fit), click the window icon next to the '+' button on the regions section header.

Please note that when you start a new project then all regions will be removed. Also, changing output resolution and adding/removing regions will result in all remaining regions being reset to their default sizes.

The size of a region window can be changed either by dragging the region handles in the output view or by changing the parameters listed under 'Window' in the region's tab. If the user changes a parameter to a value over the maximum or under the minimum constraint, then the value will be capped at the minimum or maximum respectively.

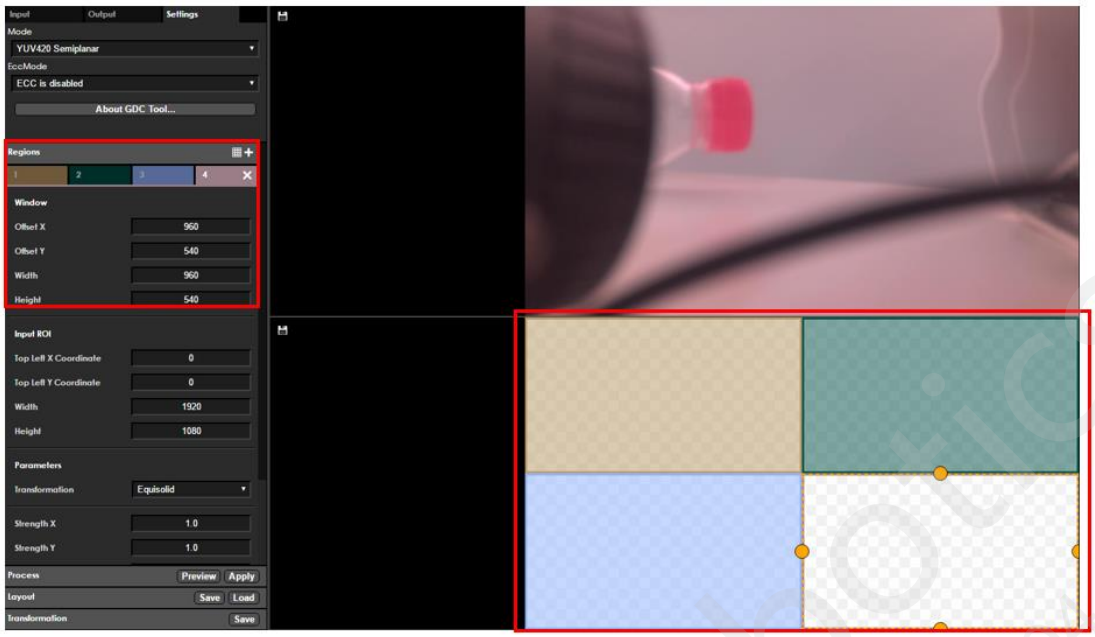


Figure 6.5-1 the output region

6.6 Process

The GDC tool supports four types of transformation:

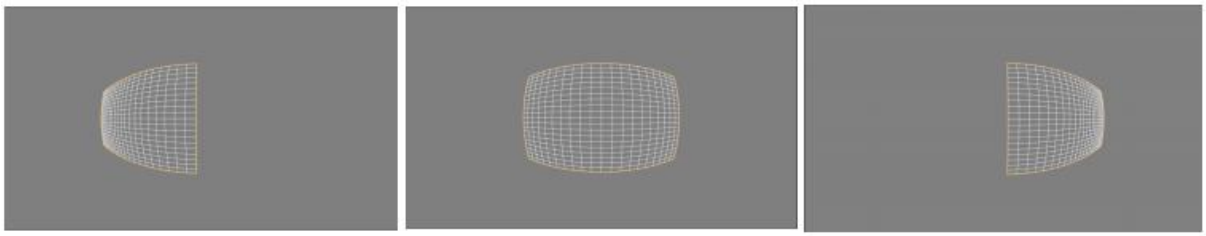
- Equisolid projected to plane
- Equisolid projected to cylinder
- Equidistant projected to arbitrary plane
- Custom
- Affine

Use the dropdown list with the label 'transformation' in the region's tab to change the type of transformation being applied to this region. Changing this drop down will automatically change the available parameters to the correct ones for that transformation.

All transformation types have the following three common parameters (examples shown on the Equidistant transformation):

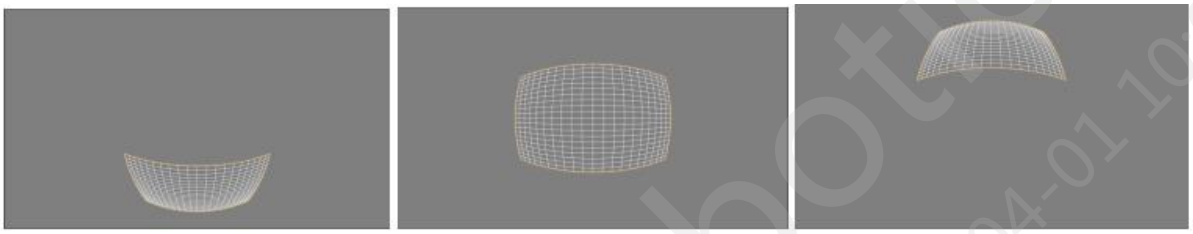
- **Pan**

Offsets the transformation grid by the given number of pixels in a horizontal direction along the transformation surface.



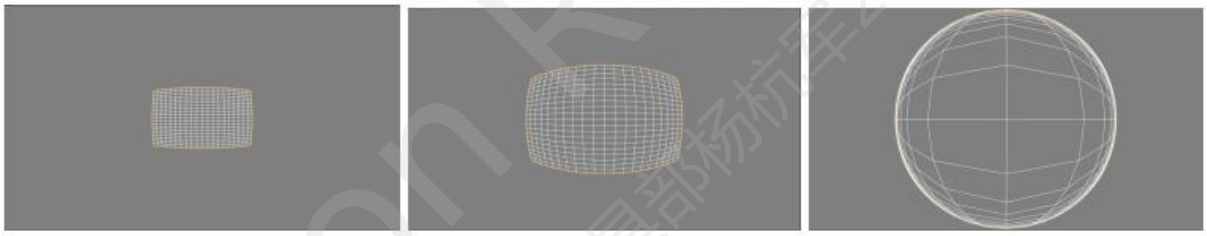
- **Tilt**

Offsets the transformation grid by the given number of pixels in a vertical direction along the transformation surface.



- **Zoom**

Scales the transformation output by the factor provided.



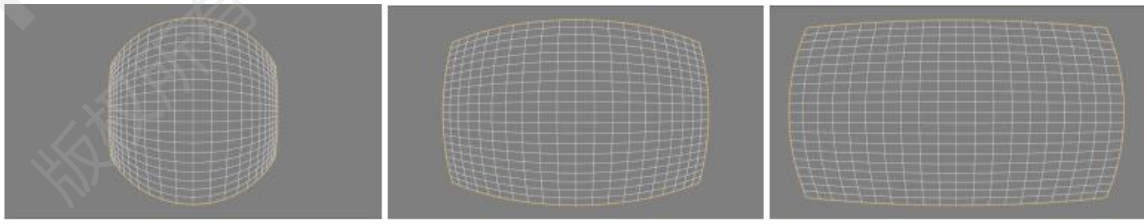
6.6.1 Equisolid projected to plane

This transformation provides Equisolid (panoramic) correction and displays the result as a projection on a plane.

This transformation has three more specific parameters:

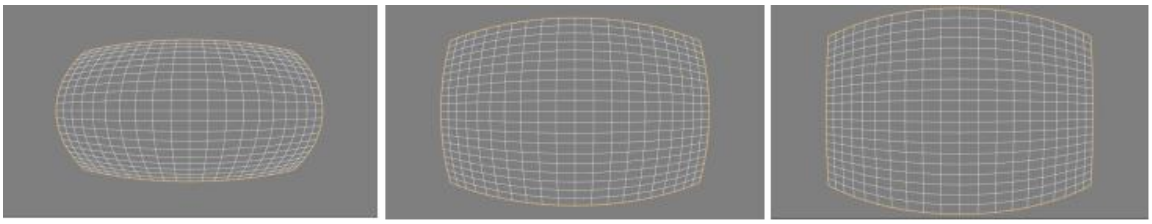
- **Strength X**

The strength of the transformation along the X axis (dimensionless non-negative parameter).



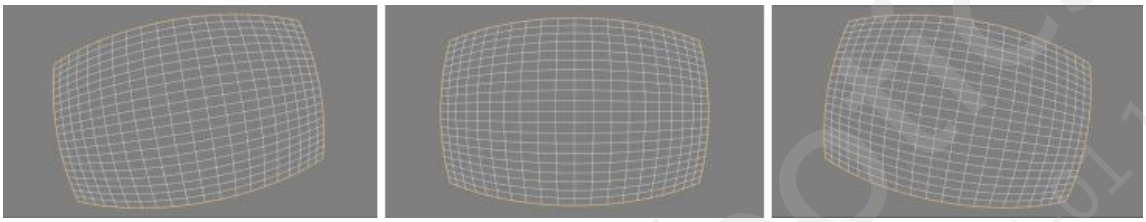
- **Strength Y**

The strength of the transformation along the Y axis (dimensionless non-negative parameter).



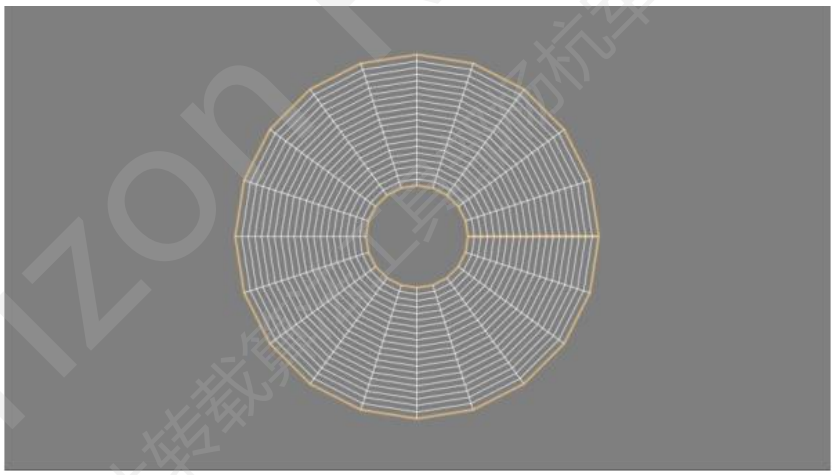
- **Rotation（degrees）**

The angle that the region will be rotated on its axis (from -180 to 180).



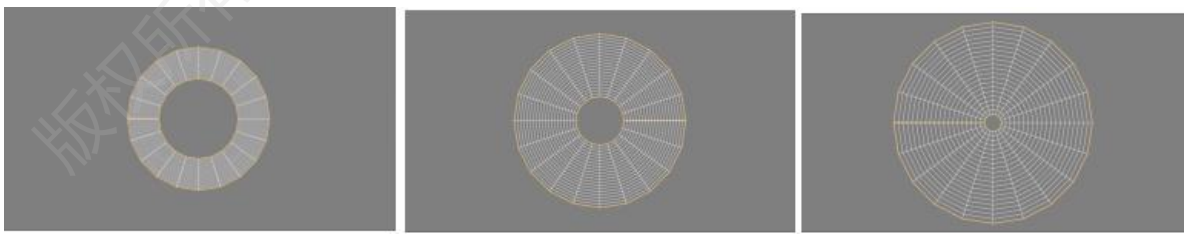
6.6.2 Equisolid projected to cylinder

This transformation provides Equisolid correction for a full fisheye frame projecting the resulting image to a cylindric panorama.



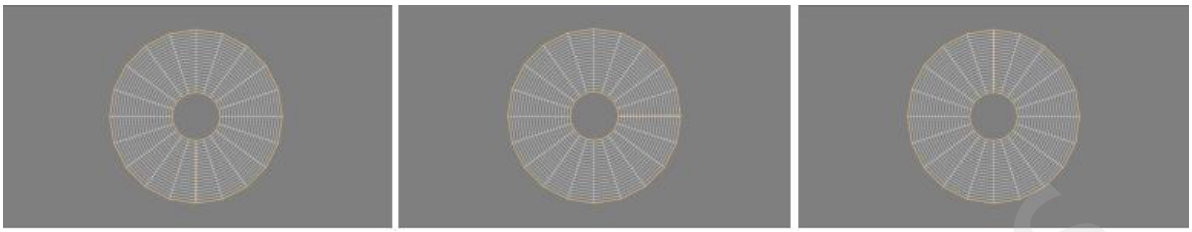
- **Strength**

The strength of the transformation (dimensionless non-negative parameter).



- **Rotation（degrees）**

The angle that the region will be rotated on its axis which in the case of stereographic transformations is the centre of the region (from -180 to 180).



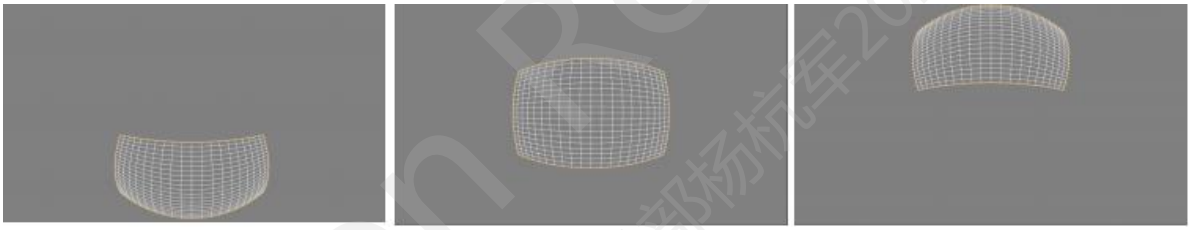
6.6.3 Equidistant projected to arbitrary plane

The Equidistant transformation contains many parameters which allows it to provide a whole range of different target planes for projection. This gives you more freedom to select the desired region of a fisheye frame that you want to transform.

Universal transformations have the following specific parameters:

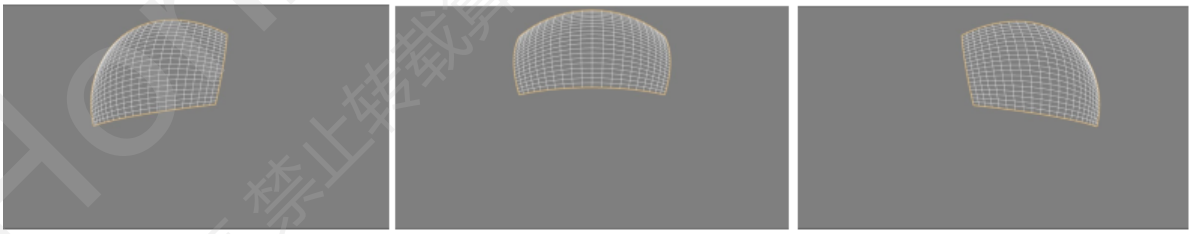
- **Elevation**

This defines the tilt of the projection axis in degrees



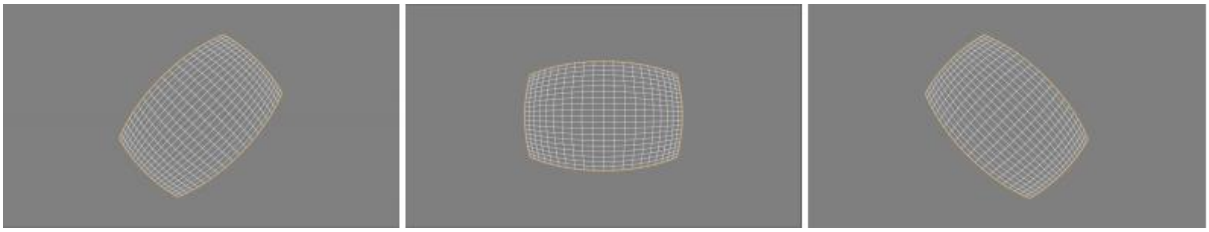
- **Azimuth（Shown here with elevation at 40 degrees）**

This defines the rotation of the projection axis in degrees. As this parameter rotates the axis and not the grid, if elevation is 0 then Azimuth will have no visible effect.



- **Rotation**

The angle of main projection axis rotation around itself in degrees. Positive angles correspond to anticlockwise rotation. This parameter is useful if the image needs to be rotated to become vertical. This parameter is particularly important for dome cameras and must be equal to Azimuth to achieve vertical output images.



- **FOV width**

Describes the size of the output field of view in the vertical dimension in degrees. The range of valid values is from 0 to 180.

- **FOV height**

Describes the size of the output field of view in the vertical dimension in degrees. The range of valid values is from 0 to 180.

Normal eye-look value is about 90degrees. For transformation where cylindricity (see below) is equal to '0' the values of FOV width and height 180 will lead to infinite stretching of the image

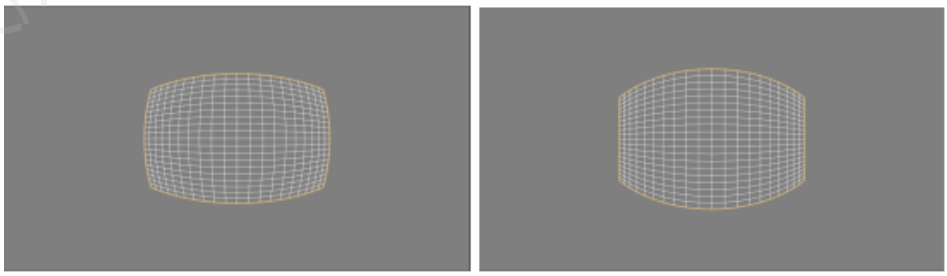
Note that when the 'Keep Ratio' parameter is switched on the FOV height parameter is ignored and its value is calculated automatically to keep same stretching strength in both horizontal and vertical directions.

- **Cylindricity X**

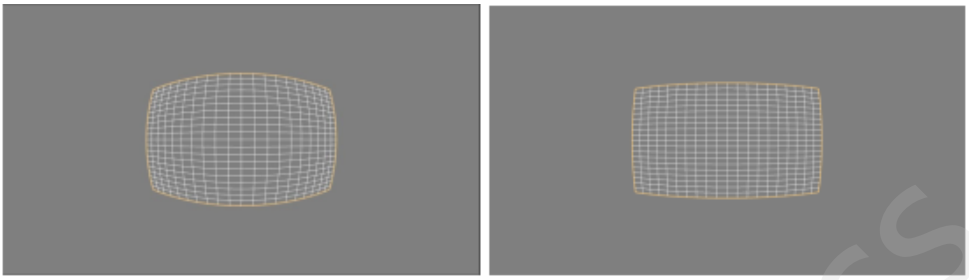
Describes how spherical the target projection will be along the X axis. This value is from 0 to 1, with 1 being spherical. If this value is set to 1 and the Cylindricity Y value is set to 0, the projection will form a cylinder along the X axis.

- **Cylindricity Y**

Describes how spherical the target projection will be along the Y axis. This value is from 0 to 1, with 1 being spherical. If this value is set to 1 and the Cylindricity X value is set to 0, the projection will form a cylinder along the Y axis.

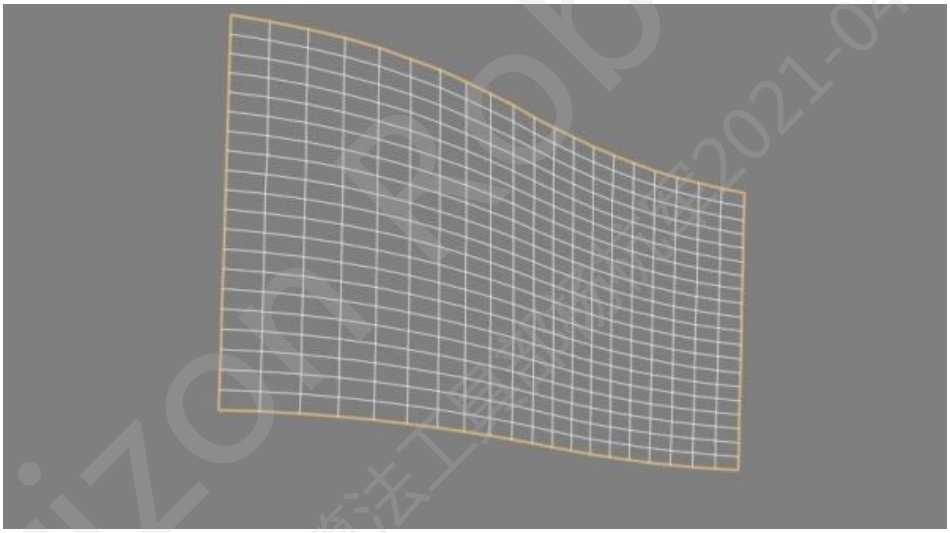


If both X and Y Cylindricity values are set to 1 then projection will be spherical. If both of them are 0, then the transform will be rectangular.



6.6.4 Custom

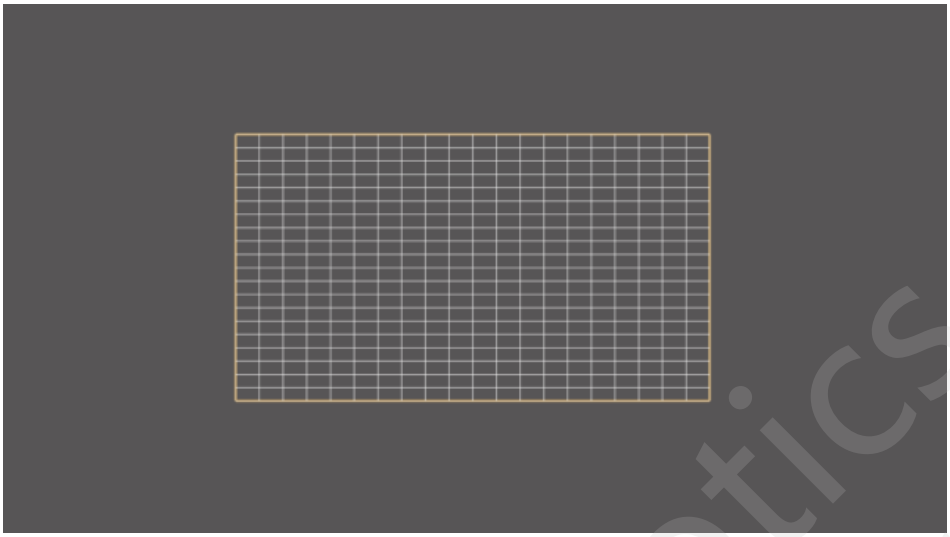
These are used to create transformations that cannot be described by any provided transformation. To correct arbitrary distortions, a special calibration file must be provided to the GDC tool).



To generate this calibration file, we must capture an image with predefined pattern under the same conditions as the real one. Example of such image is a grid of black dots on a white sheet. The distance between the points in horizontal and vertical direction on the original image should be the same. This tool cannot generate such calibration files, it can only use those files to configure custom transformations.

6.6.5 Affine

The Affine transformation allows to perform image rotation in addition to pan, tilt and zoom without any geometric distortion correction.

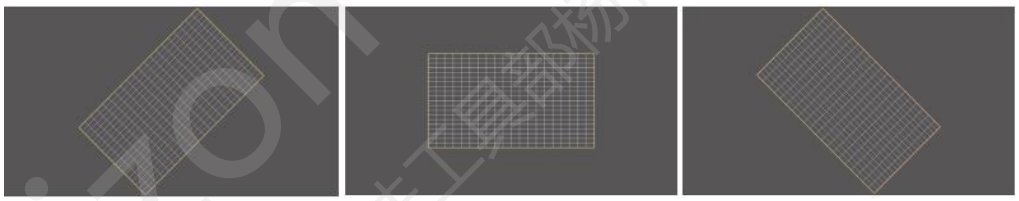


Affine transformations have the following specific parameter:

- **Rotation**

The angle of main projection axis rotation around itself in degrees. Positive angles correspond to anticlockwise rotation. This parameter is useful:

- ✓ If an image needs to be rotated to become vertical.
- ✓ For dome cameras. And it must be equal to the Azimuth to achieve vertical output images.



6.7 Saveing and loading layout

The layout is a set of regions along with associated transformations and their settings. This information (without the source image) can be saved to a JSON file by clicking 'Save Layout' in the 'Layout' section of sidebar.

These JSON files can be reloaded to the workspace by clicking 'Load Layout' in the 'Layout' section of the sidebar. When 'Load Layout' is clicked, a dialog window will open to allow the user to select a layout file. If the layout file's input resolution does not match the current input resolution then the input data from the layout file will not be loaded.