

# Qualcomm Spectra™ 2xx Deep Dive

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

80-P9301-60 Rev. H

**Confidential and Proprietary – Qualcomm Technologies, Inc.**

**NO PUBLIC DISCLOSURE PERMITTED:** Please report postings of this document on public servers or websites to: [DocCtrlAgent@qualcomm.com](mailto:DocCtrlAgent@qualcomm.com).

**Restricted Distribution:** Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

**Confidential and Proprietary – Qualcomm Technologies, Inc.**

**NO PUBLIC DISCLOSURE PERMITTED:** Please report postings of this document on public servers or websites to: [DocCtrlAgent@qualcomm.com](mailto:DocCtrlAgent@qualcomm.com).

**Restricted Distribution:** Not to be distributed to anyone who is not an employee of either Qualcomm Technologies, Inc. or its affiliated companies without the express approval of Qualcomm Configuration Management.

Not to be used, copied, reproduced, or modified in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm Technologies, Inc.

All Qualcomm products mentioned herein are products of Qualcomm Technologies, Inc. and/or its subsidiaries.

Chromatix is a trademark of Qualcomm Technologies, Inc. registered in the United States and other countries. Hexagon, Qualcomm, and Qualcomm Spectra are trademarks of Qualcomm Incorporated, registered in the United States and other countries. Other product and brand names may be trademarks or registered trademarks of their respective owners.

This technical data may be subject to U.S. and international export, re-export, or transfer ("export") laws. Diversion contrary to U.S. and international law is strictly prohibited.

Qualcomm Technologies, Inc.  
5775 Morehouse Drive  
San Diego, CA 92121  
U.S.A.

© 2017-2018 Qualcomm Technologies, Inc. and/or its subsidiaries. All rights reserved.

# Revision History

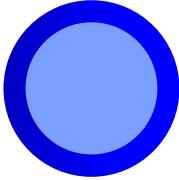
Revision	Date	Description
A	May 2017	Initial release
B	May 2017	Added information on Adaptive bayer filter (ABF) and 2D LUT
C	May 2017	Minor edit to the content slide
D	May 2017	Numerous changes were made to this document; it should be read in its entirety
E	August 2017	Added section on upscaler tuning guidelines
F	August 2017	Added section on Global tone mapping (GTM)
G	October 2017	Updated HNR and ANR sections
H	April 2018	Updated ANR and TF sections and added MFNR section

# Contents

---

- Introduction to Qualcomm Spectra 2xx Components
- Qualcomm Spectra 2xx Pipeline
- Bad Pixel Correction/Bad Cluster Correction/PD Pixel Correction (BPC/BCC/PDPC)
- Adaptive Bayer Filter (ABF)
- Green Imbalance Correction (GIC)
- Global Tone Mapping (GTM)
- Hybrid Noise Reduction (HNR)
- Image Correction and Adjustment (ICA)
- Advanced Noise Reduction (ANR)
- Temporal Filter (TF)
- Multi-Frame Noise Reduction (MFNR)
- Local Tone Mapping (LTM)
- 2D LUT
- Chroma Suppression (CS)
- Adaptive Spatial Filter (ASF)
- Upscaler
- Grain Adder (GRA)
- Questions?

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com



# Introduction to Qualcomm Spectra 2xx Components

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Objective

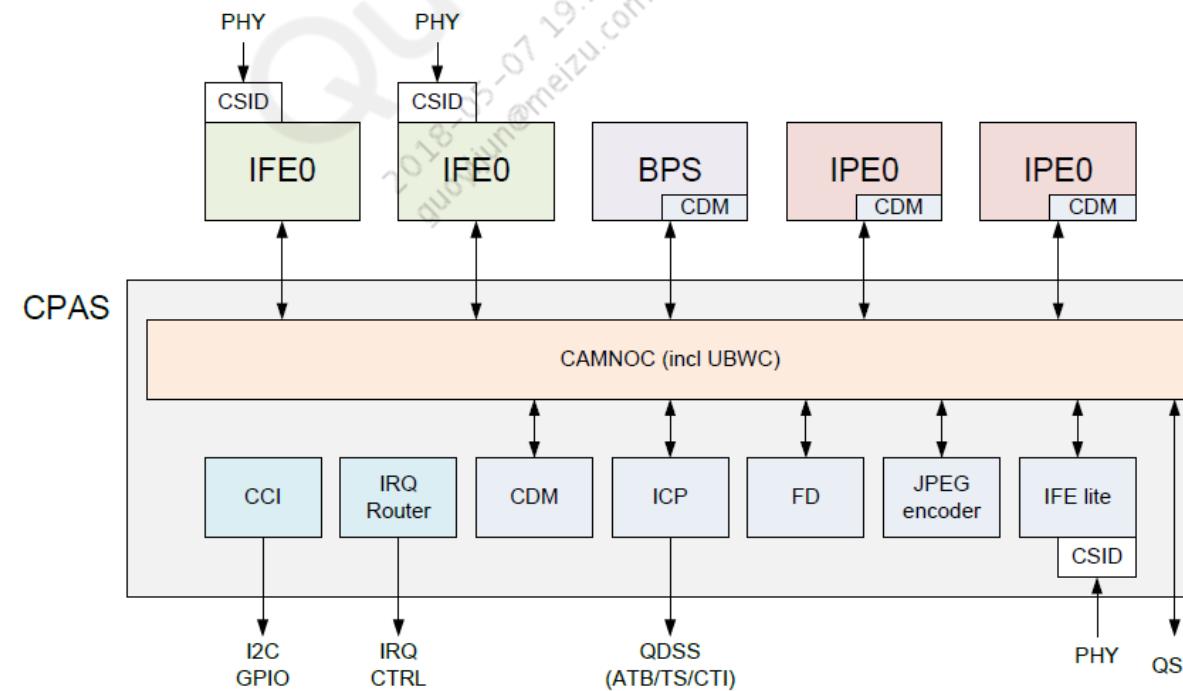
---

- At the end of this presentation, you will have an understanding of the following Qualcomm Spectra 2xx modules, which are the main hardware components for image quality:
  - Image front end (IFE)
  - Bayer processing segment (BPS)
  - Image processing engine (IPE)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Qualcomm Spectra 2xx Component – Introduction

- This document introduces modules in IFE, BPS and IPE, which are the main hardware components for image quality.
- Qualcomm Spectra 2xx camera subsystem consists of Image Front End (IFE), Bayer processing segment (BPS), Image processing engine (IPE) and Camera peripheral and support (CPAS). The following diagram gives a high level understanding of the subsystem:



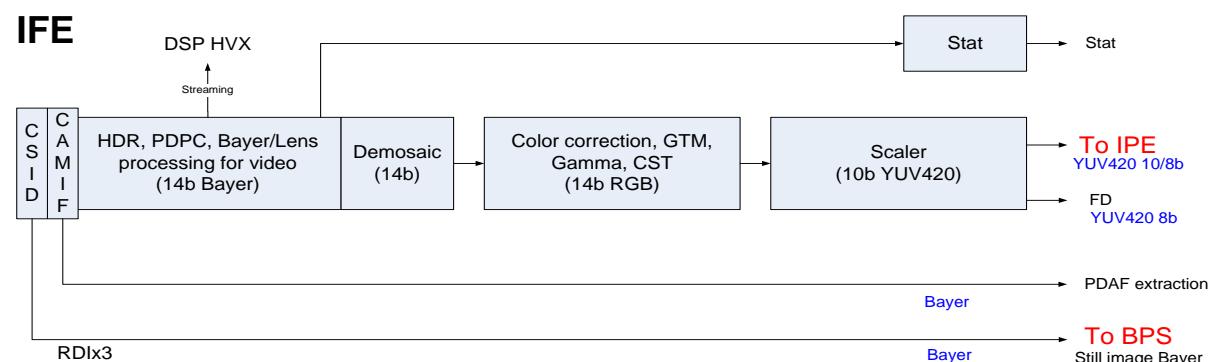
# Qualcomm Spectra 2xx Component – Introduction (cont.)

- **Image front-end engine (IFE) x 2**

- Bayer process for preview/video only
- Stats for 3A
- 4 tape-in/out points for Qualcomm Hexagon™ Vector eXtensions(HVX) streaming
- Compared to previous chipsets, major luma/color processing modules (that is CAC, LTM, CV, SCE, CS) are moved to IPE
- Multipass outputs (1:1, 1:4, 1:16) feeding to IPE
- IFE provides 3 paths for ideal raw dump
  1. From CAMIF (Bayer format)
  2. From Lens shading correction (LSC) output (Bayer format)
  3. From Global tone mapping (GTM) output (RGB14-bit format)

- **IFE-lite x 1**

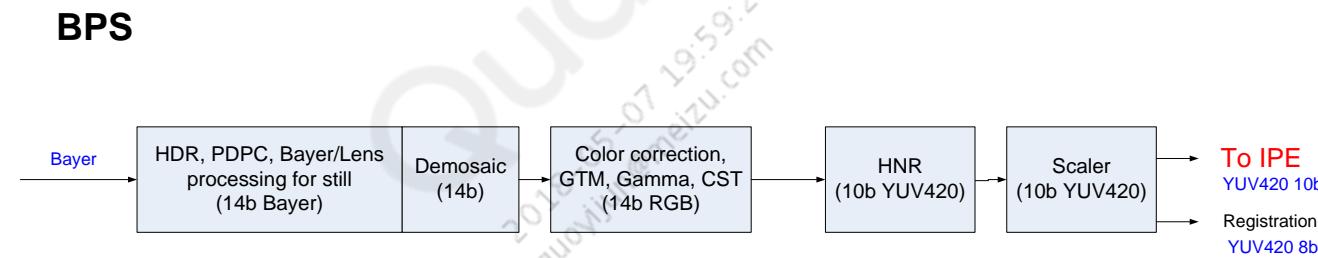
- 4 RDI streams (no processing)



# Qualcomm Spectra 2xx Component – Introduction (cont.)

- **BPS × 1**

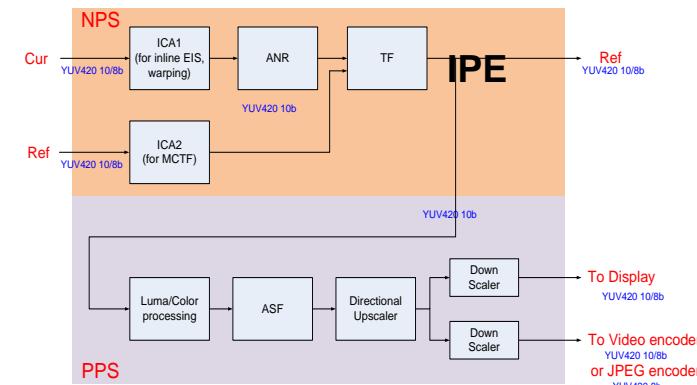
- Bayer process and noise reduction for snapshot only
- Simple stats (BG and HDR Bhist) for special offline processing
- Multipass outputs (1:1, 1:4, 1:16, 1:64) feeding to IPE



# Qualcomm Spectra 2xx Component – Introduction (cont.)

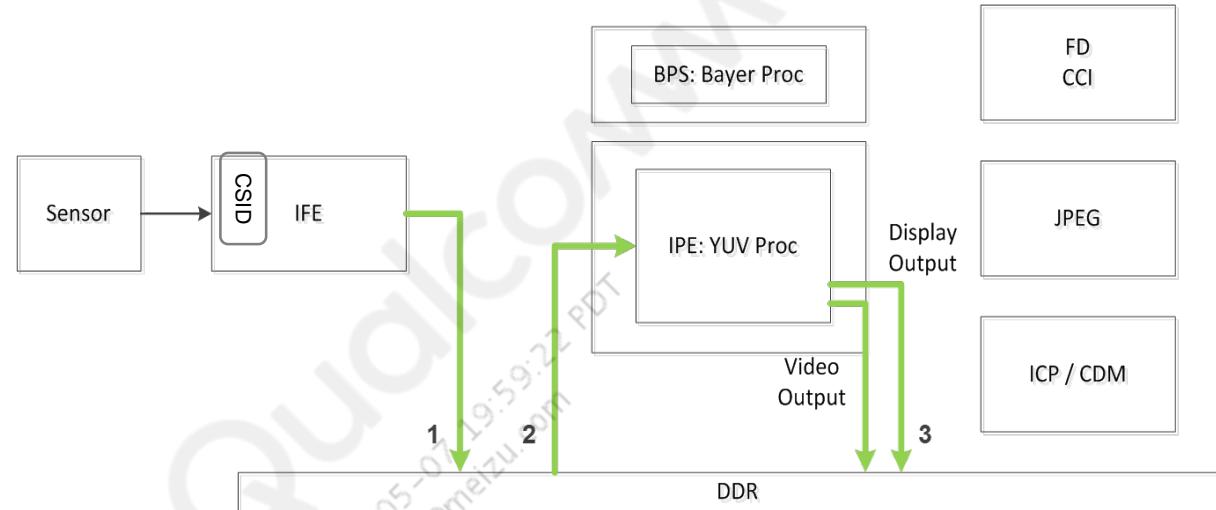
- **IPE × 2**

- Each IPE consists of 2 parts. Noise processing segment (NPS) and Post processing segment (PPS)
- NPS
  - ICA1 for inline EIS and image warping
  - ICA2 and Temporal filter (TF) for Motion compensation temporal filtering (MCTF)
  - Advanced noise reduction (ANR) for spatial noise reduction
- PPS
  - CAC, LTM, 2DLUT, CV, CS, SCE modules for luma/color processing
  - Moving above modules after noise reduction is one of the main changes in Qualcomm Spectra 2xx
  - Adaptive spatial filter (ASF) for detail enhancement
  - Scaler (No rotator)
  - Single input multiple outputs (SIMO)

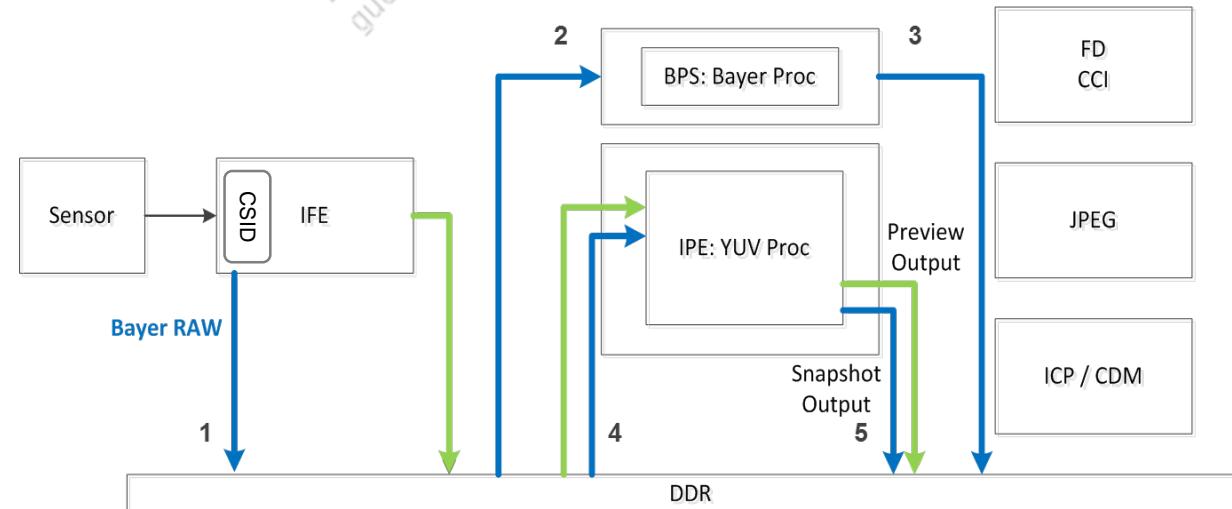


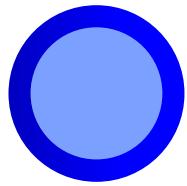
# Qualcomm Spectra 2xx Data Flow

- Preview and Video Data Flow



- Snapshot Data Flow



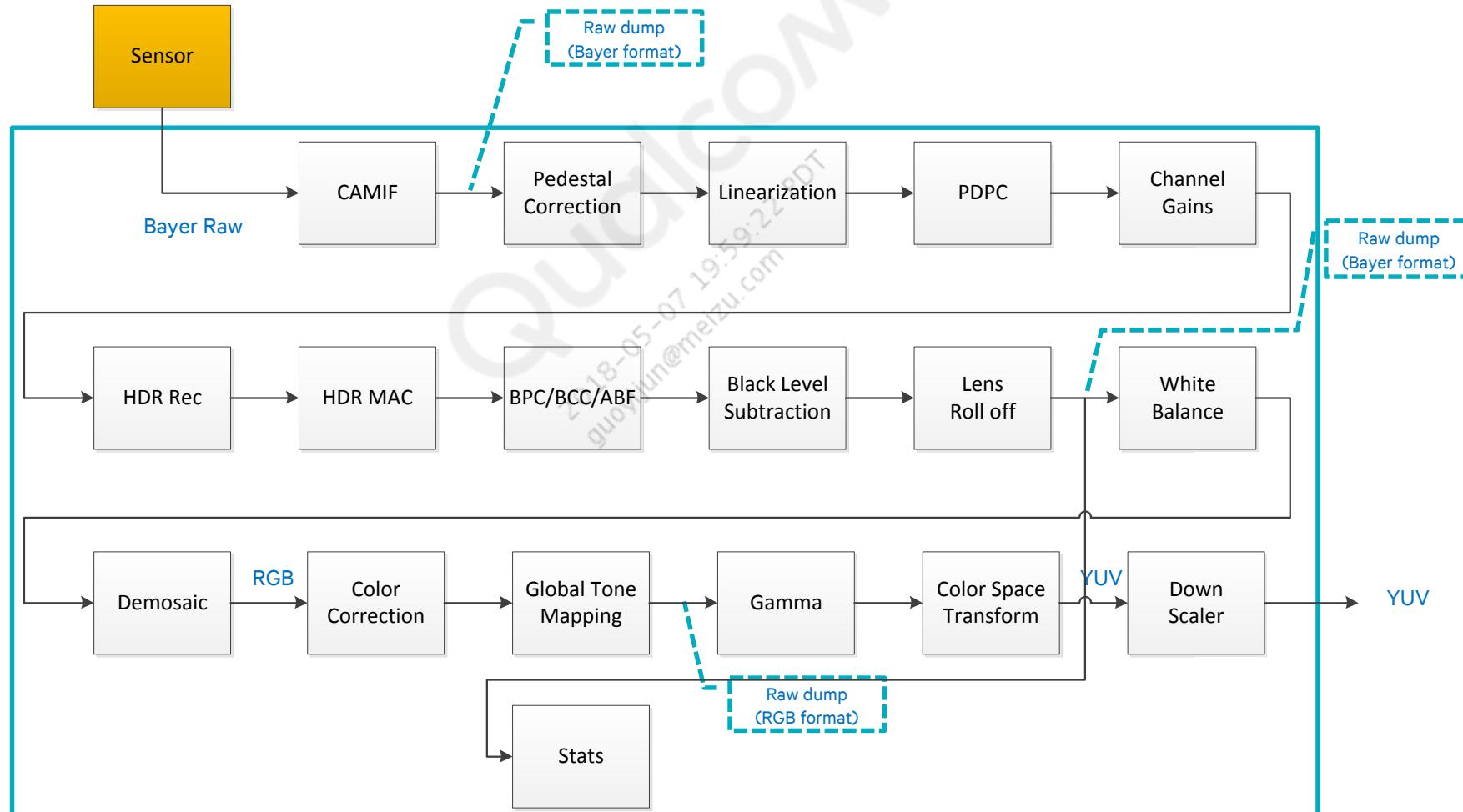


# Qualcomm Spectra 2xx Pipeline

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

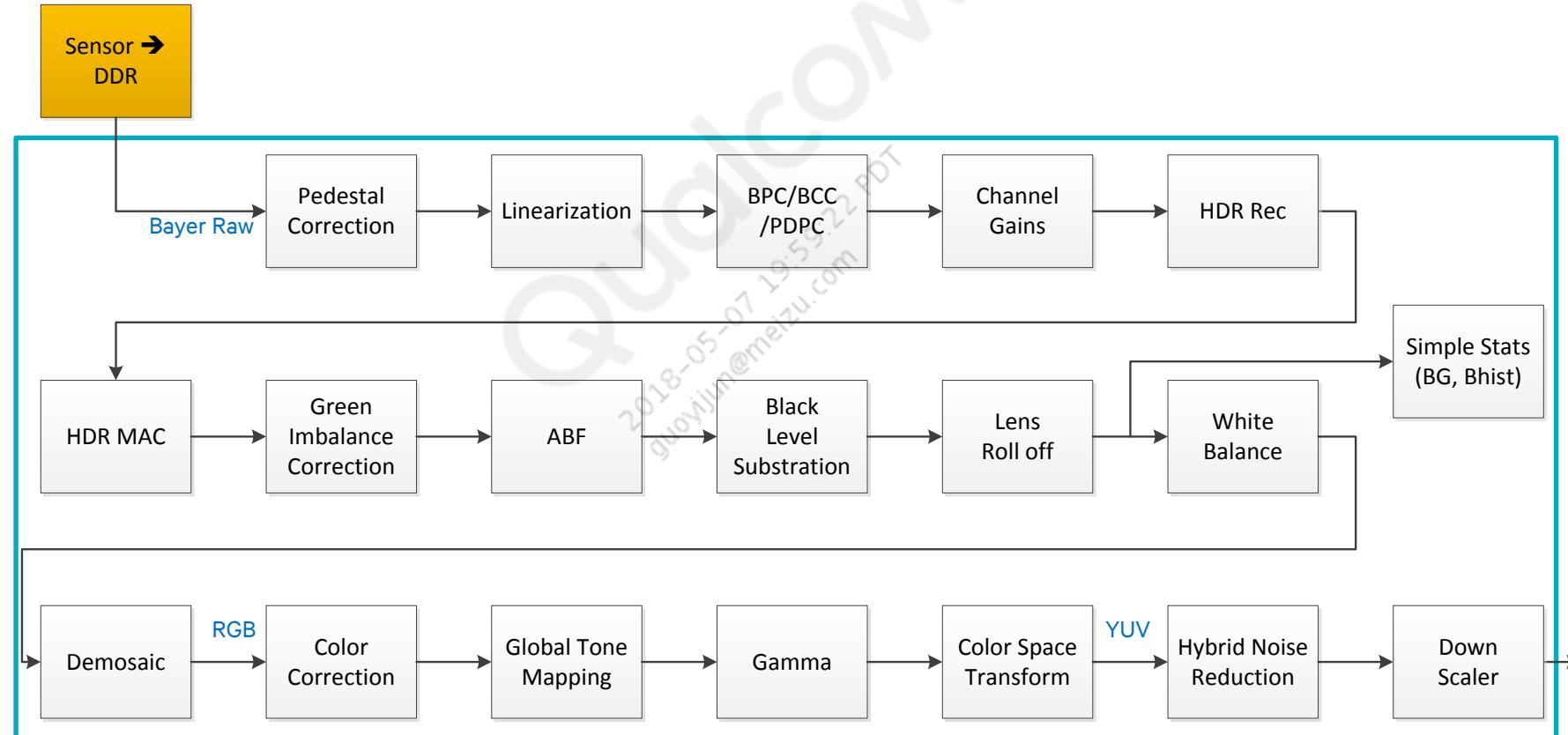
# Qualcomm Spectra 2xx Pipeline (Simplified IQ Version)

IFE



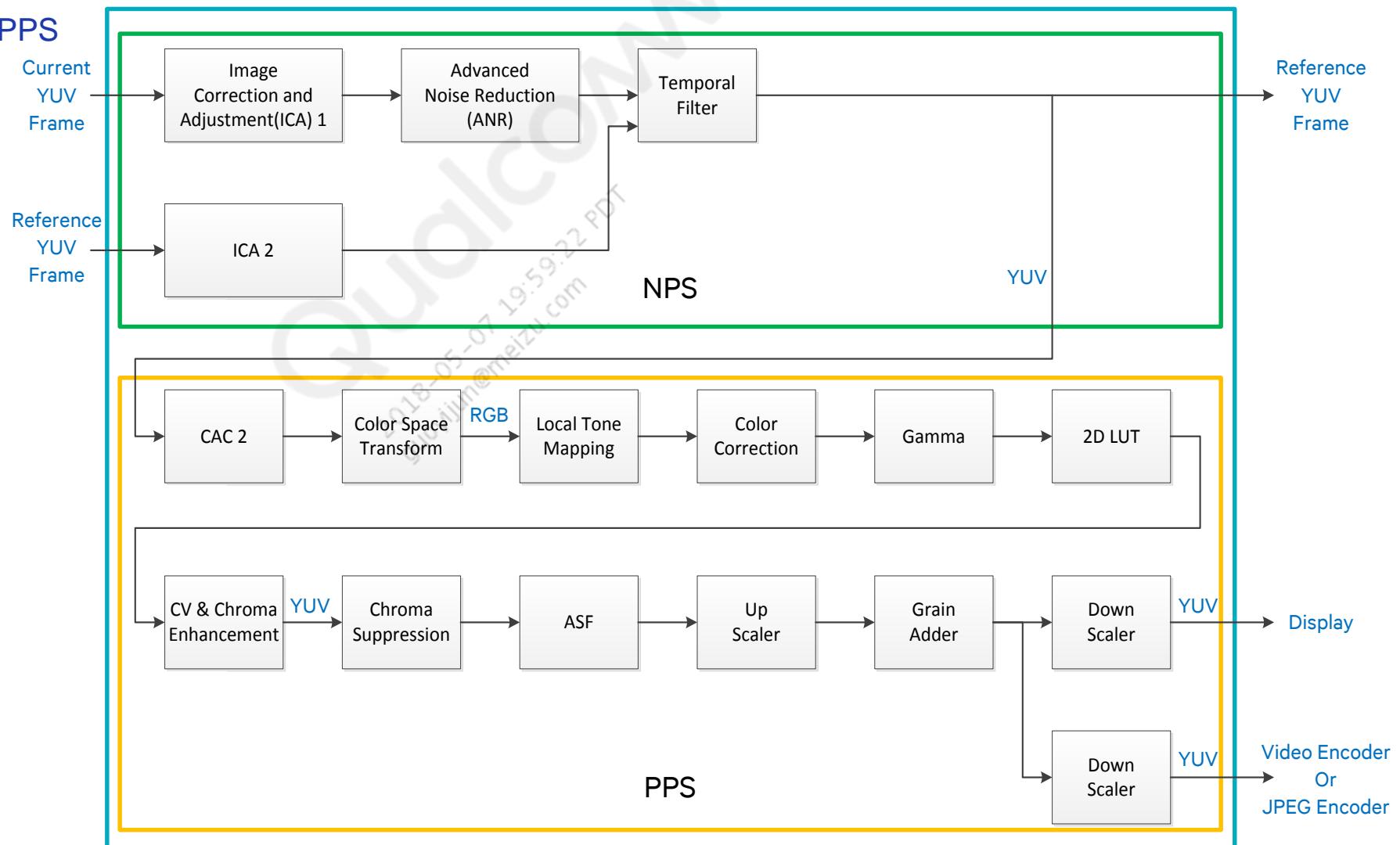
# Qualcomm Spectra 2xx Pipeline (Simplified IQ Version) (cont.)

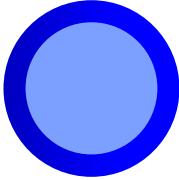
BPS



# Qualcomm Spectra 2xx Pipeline (Simplified IQ Version) (cont.)

IPE: IPE consists of NPS and PPS





# Bad Pixel Correction/Bad Cluster Correction/PD Pixel Correction (BPC/BCC/PDPC)

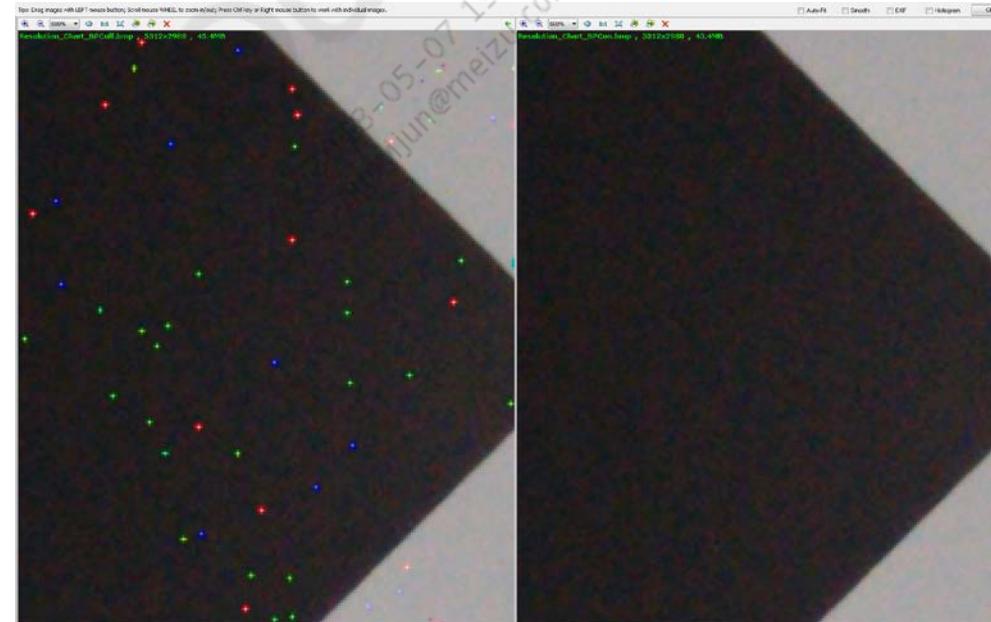
Qualcomm  
2018-05-01 09:59:22 PDT  
guovjun@meizu.com

# BPC/BCC/PDPC – Introduction

## Problem statement

- Bad pixel correction (BPC)/Bad cluster correction (BCC)
  - A defective pixel is defined as a pixel whose response is noticeably different from the response of the other pixels in the array under dark or uniformly illuminated conditions
  - BPC and BCC modules are designed to remove the defective pixels
  - Supports single and couplet corrections but is not supported for big cluster (for more than 3)

Examples :

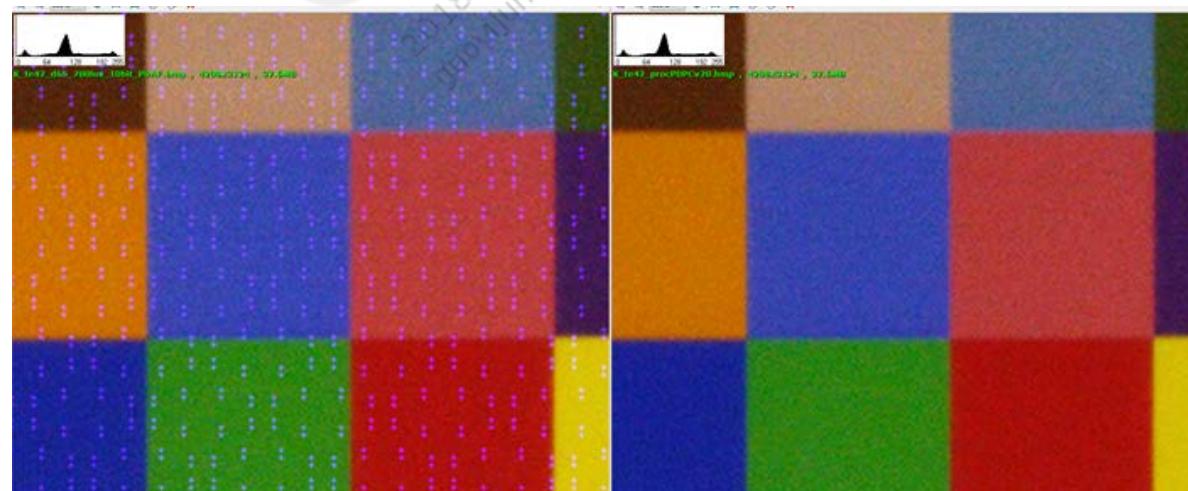


Left: Before BPC/BCC

Right: After BPC/BCC

# BPC/BCC/PDPC – Introduction (cont.)

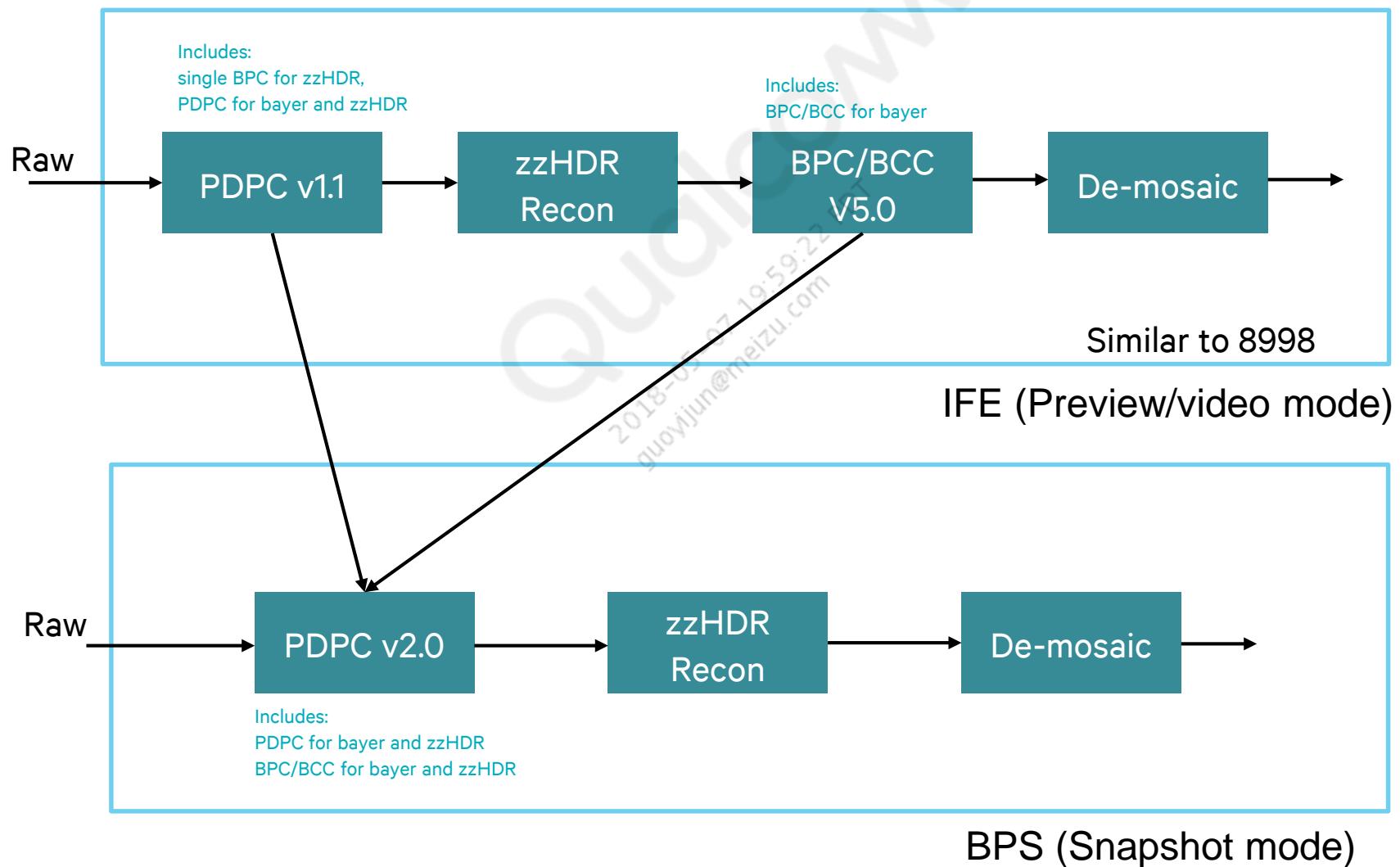
- PD pixel correction (PDPC)
  - Recently, a novel technique called Phase Pixel was introduced into CMOS Sensor to speed up autofocus. The basic principle of ‘Phase Pixel’ is that a single image is split into two, left and right hand sided halves.
  - PDPC module is designed to correct defective pixels on phase pixel location
  - PD pixels locations are known and the PD pixels can only be on red or blue location but not on a green surface
  - PD pixels are corrected using neighboring pixels
  - Support Bayer pattern and zzHDR (PD pixel on long exposure only)



Left: PDPC Off (PD pixel on blue channel)    Right: After PDPC On

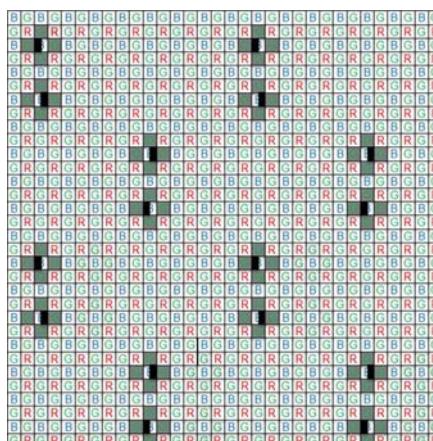
# Step 1: Overview of PDPC/BPC Workflow

Block diagram

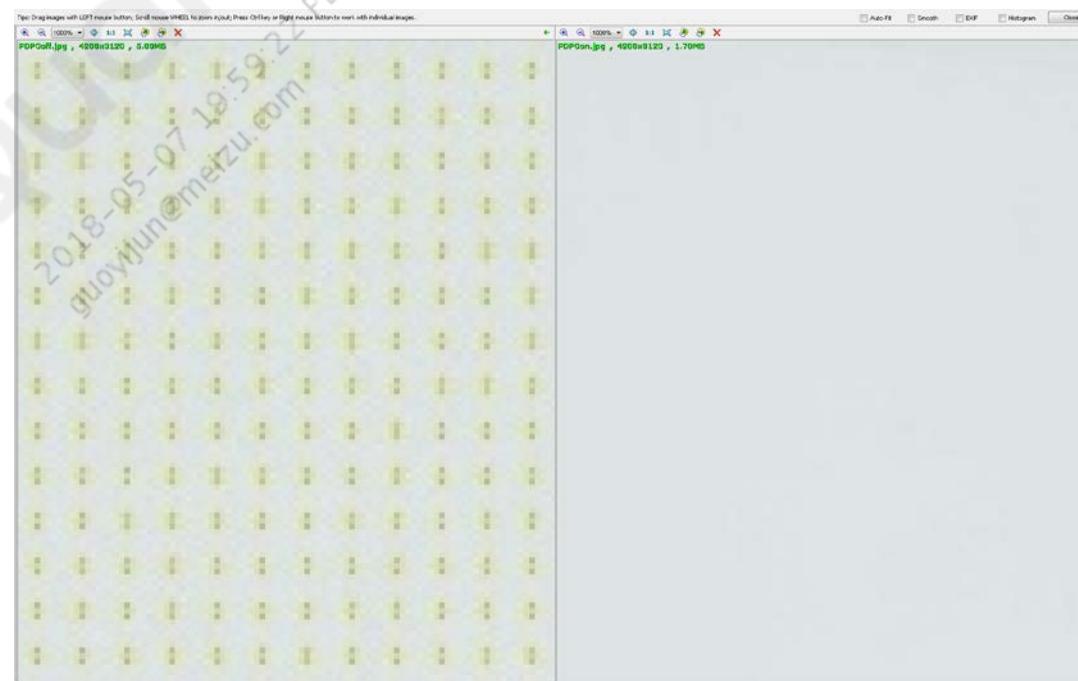


# Step 1: Overview of the PD Pixel Correction

- Design wise PD pixel correction part are the same for PDPC v1.1 and PDPC v2.0 module (same as PDPC v1.0 in MSM8998)
- Based on the PDAF pixel location, if current pixel is PDAF pixel, its value is replaced with the corrected value using neighboring pixels



Example of PD pixel locations



PDPC Off

PDPC On

# Step 1: Overview of BPC/BCC

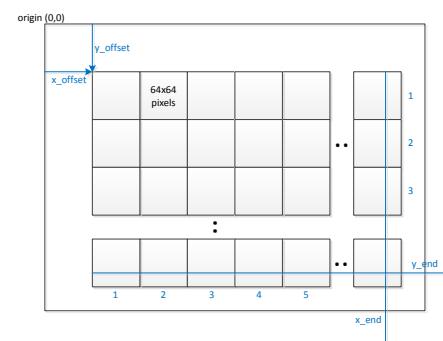
---

- In IFE, with the combination of BPC in PDPC v1.1 and BPC/BCC v5.0 we can run BPC/BCC for zzHDR and bayer
  - BPC/BCC v5.0 is based on the bayer pattern, similar to the BPC/BCC in MSM8996 and MSM8998 bayer mode BPC/BCC
  - Inside PDPC v1.1, a single bad pixel correction is designed for the zzHDR mode
- In BPS, the BPC/BCC part is used in PDPC v2.0 to deal with zzHDR pattern and bayer pattern, the design is similar to MSM8996 and MSM8998, with some improved feature and extra consideration of the zzHDR pattern and PD pixels
- Same channel and cross channel are used for the BPC/BCC to avoid false colors and artifacts
- A pixel is considered a “bad pixel” based on its value:
  - Value is significantly higher than its neighbors>hot pixel
  - Value is significantly lower than its neighbors>cold pixel
- If pixel is identified as bad pixel, its value will be replaced with the corrected value using its neighboring pixels

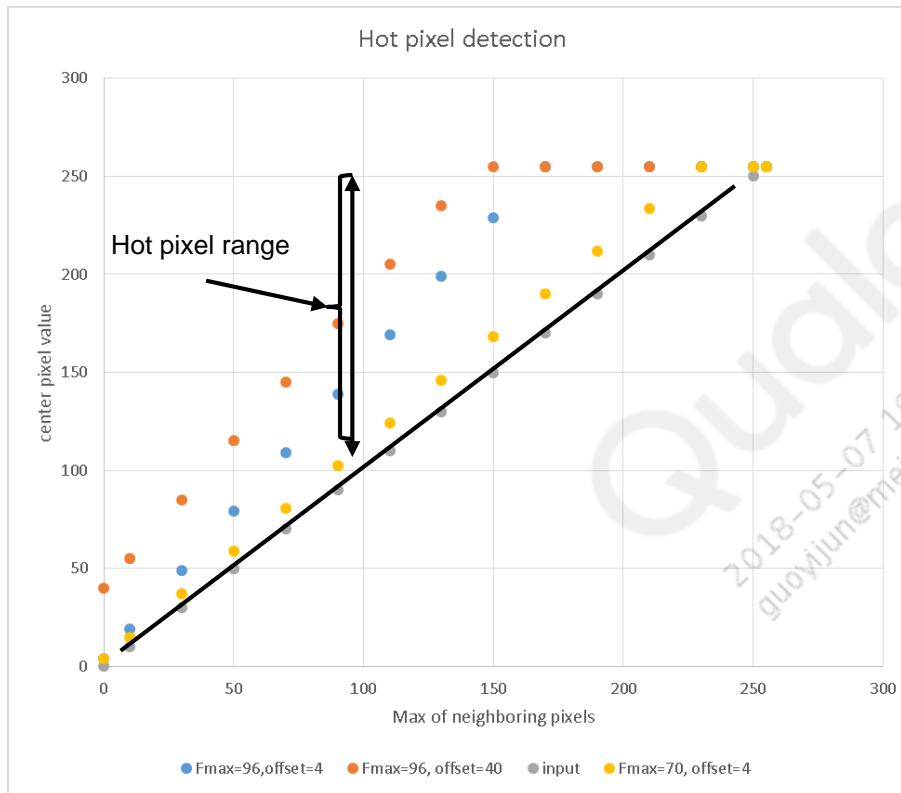
## Step 2.1: PD Pixel Correction

- PD pixel correction only requires the PDAF pixel locations, and does not need any tuning parameters
- PDAF pixel pattern is assumed to be periodic across 64x64 pixel blocks
- If the PD pixel repeat block is smaller than 64x64, copy the block to populate it to full 64x64 block
- A 32x64 PDAF\_PD\_Mask is generated based on PD pixel locations by skipping green pixel locations

Pdaf_global_offset_x	14	PDAF 64x64 pattern global offset x- Pixel offset (0 means first pixel from left)
Pdaf_global_offset_y	14	PDAF 64x64 pattern global offset y- line offset(0 means first line from top)
Pdaf_global_offset_y	14	Horizontal PDAF pixel end location +1 (0 means first pixel from left)
Pdaf_y_end	14	Vertical PDAF pixel end location +1 (0 means first line from top)
PDAF_PD_Mask[32*64]	2048*1	PD location 1 bit mask LUT- 0: not PD pixel; 1: PD pixel



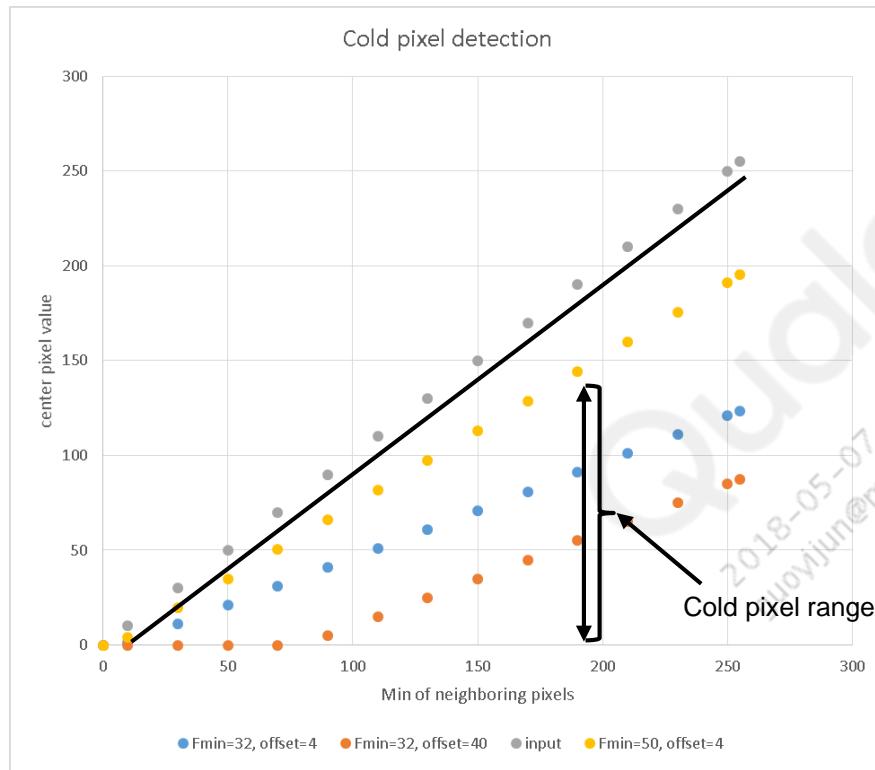
## Step 2.2: BPC/BCC Basic Tuning Steps (Hot Pixel)



- Parameters to tune: Fmax and BPC offset, BCC offset
  - 1. Set the parameters to default based on the different lighting conditions
  - 2. Check whether any bad pixels are missed, for highlight region, lowering Fmax has more impact than lowering offset, for lowlight region, lowering offset has more impact than reducing Fmax
  - 3. Use the adjusting Fmax, offset settings and compare to the original images without BPC on, check whether there is any resolution loss or false color
  - 4. Fine tune the Fmax and offset till the resolution loss and false colors are acceptable

Larger Fmax > less hot pixel detected  
Larger offset > less hot pixel detected  
BCC offset is the offset used for detecting clusters

## Step 2.2: BPC/BCC Tuning Steps (Cold Pixel Part) Basic Tuning



- Parameters to tune: Fmin and BPC offset, BCC offset
  - Set the parameters to default based on the different lighting conditions
  - Check whether any bad pixels have been missed, for the highlighted region. Increasing Fmin has more impact than lowering offset. For lowlight region, lowering offset has more impact than adjusting Fmin.
  - Use the adjusting Fmin offset settings and compare the result with the original images without BPC on. Check whether there is any resolution loss or false color.
  - Fine tune the Fmin and offset till the resolution loss and false colors are acceptable

Smaller Fmin>less cold pixel detected  
Larger offset>less cold pixel detected  
BCC offset is the offset used for detecting clusters

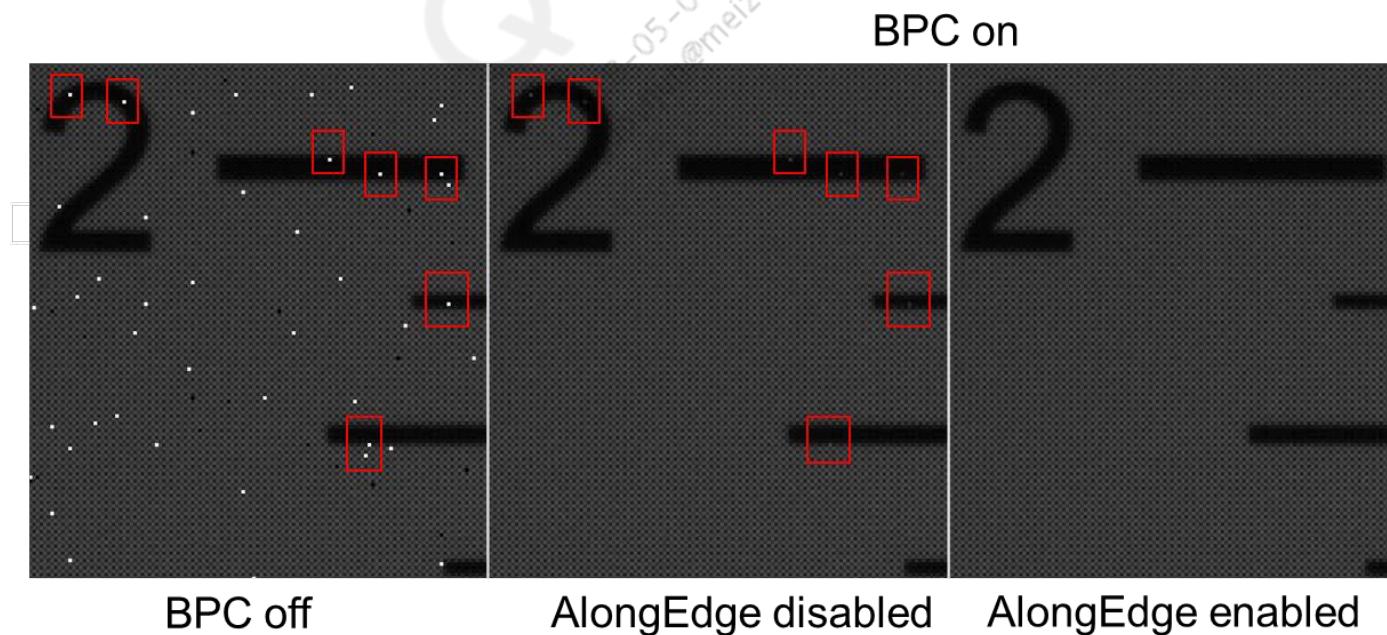
**Note:** Offset values are shared with hot and cold pixel detections, when adjusting offset, need to double check both hot pixel and cold pixels.

## Step 2.3: BPC/BCC Advanced Tuning Steps

- **UsingCrossChannel:** Enable/disable the BPC correction from cross-channel.

Note: Detection in cross channel is always on. Default value should be enabled to avoid artifacts.

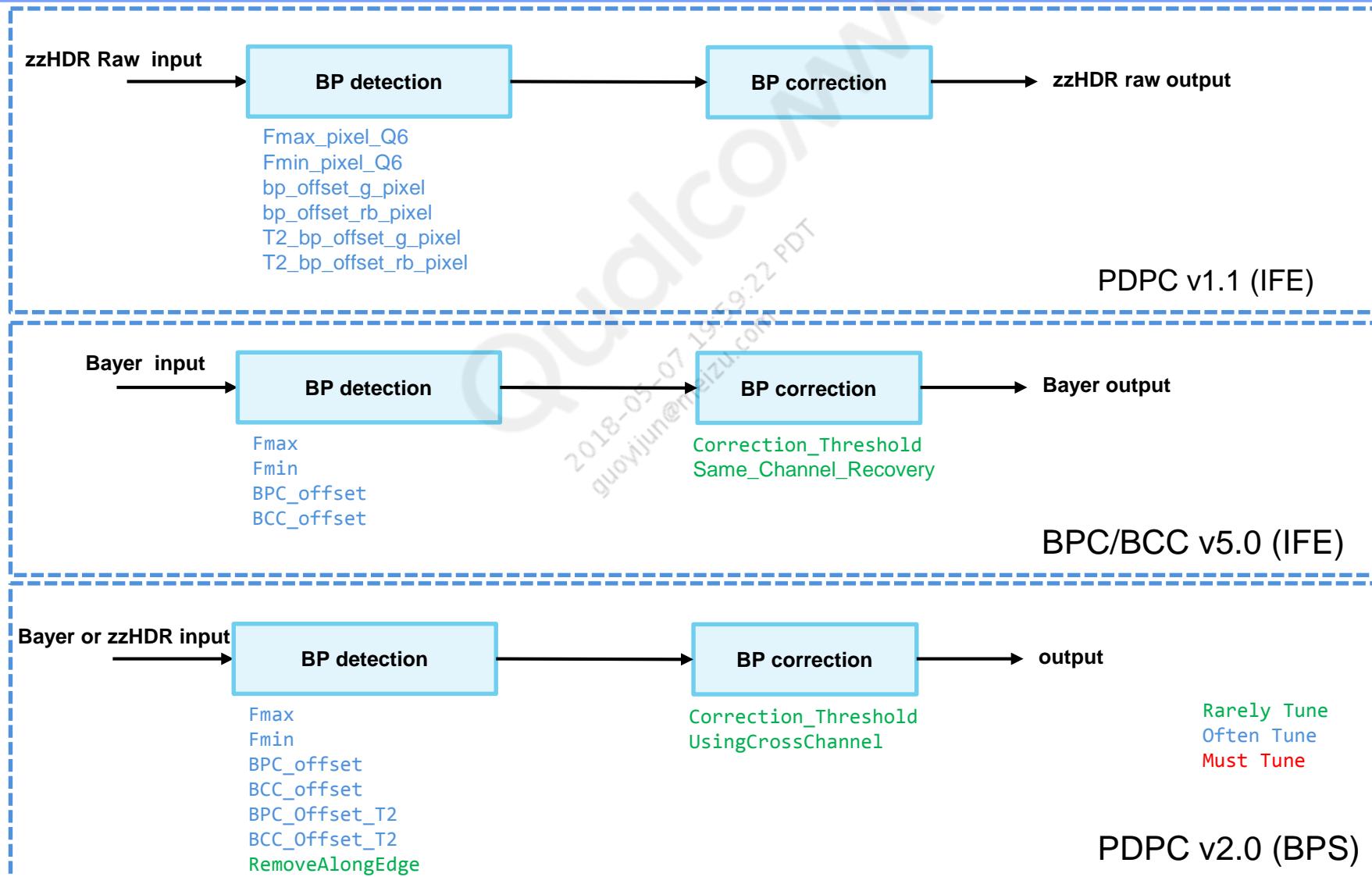
- **RemoveAlongEdge:** This feature can help for some bad pixels along the edges. But not all the bad pixels along the edge areas, it also has potential of introducing artifacts, we recommend to turn off this feature.



# BPC/BCC Supporting Types

Category	Type	Example	
Flat Region	Single		
	Same Channel Couplet		
	Cross Channel Couplet		
	>=3 Cluster		
Edge Region	Single along the edge		
	SC Couplet along the edge		
	CC Couplet along the edge		
	Bad Pixel is on the edge or inside the 'transitional edge'		
	Bad Pixel is between two edges		

# Effects of Parameters: Must Tune and Often Tune



# Enable/Disable Module in Different Mode for Bad Pixel

- Video mode: Use PDPC v1.1 and BPC/BCC v5.0, one for zzHDR, one for regular bayer
- Snapshot mode: Use PDPC v2.0 which is a merged BPC/BCC for bayer and zzHDR
- PD pixel correction enable/disable: pdaf\_pdpc\_en in PDPC v1.1 or PDPC v2.0

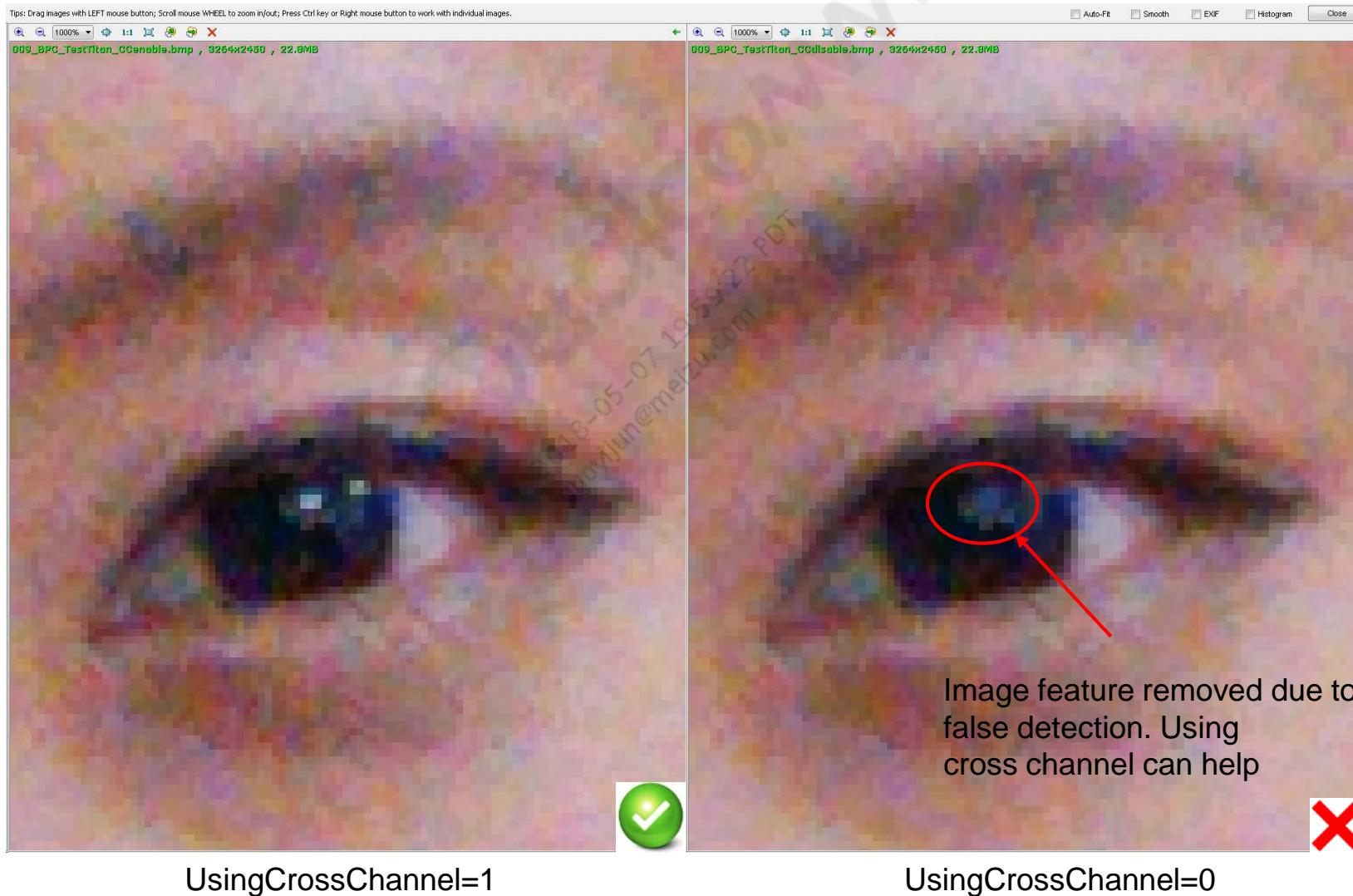
	Format	PDPC v1.1	BPC/BCC V5.0	PDPC v2.0
Video	Non-HDR	Off	On	
	zzHDR	On	Off	
Snapshot	Non-HDR			On
	zzHDR			On

# Impact of Fmax/Fmin and Offset

For PDPC v1.1, BPC/BCC v5.0 and PDPC v2.0 behaviors are the same.



# UsingCrossChannel (PDPC v2.0)



# RemoveAlongEdge (PDPC v2.0)



## Step 3: How to Coordinate with Other Modules

---

- PDPC/BPC is the first module that does not rely on other modules, while having the impact on all the following modules if not tuned properly
- For other modules tuning, they can turn off the BPC to avoid the negative impact from the un-tuned BPC settings
- For PD sensors, PDPC should always be turned on
- For PDPC/BPC tuning, only enable the basic modules that are needed to have final YUV images
  - Non-HDR: Enable black-level cancelation, AWB, CCM, tone mapping and so on. All the de-noise module and edge enhancement modules need to be turned off, including image quality enhancement module, such as CAC
  - zzHDR: Besides the modules listed in non-HDR mode, also enable the zzHDR recon and tone mapping part

# Test Plan for PD Pixel Correction

To confirm if this module works normally:

1. Take a raw image from a PDAF sensor

If we don't have a PDAF sensor, a simulation on the non-PD sensor can also work

2. Ensure the PD configuration part is correct
3. Check if the PD pixels are being corrected. While the other non-PD pixels are un-touched.
4. Check if there is any artifacts or false color
5. Ensure the PD locations are configured properly, since there are no tuning parameters available for the PD pixel correction
6. Support PD sensor types:
  - a) Non-HDR: PD pixel on red or blue
  - b) zzHDR: PD pixel on red or blue, only on long exposure



PDPC off  
pdaf\_pdpc\_en=0

PDPC on  
pdaf\_pdpc\_en=1

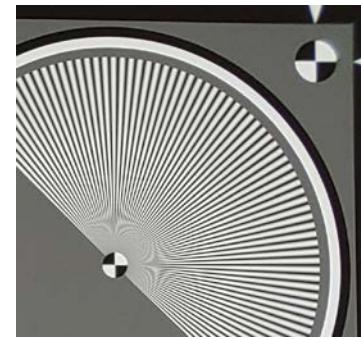
# Test Plan for PD Pixel Correction (cont.)

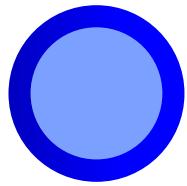
To confirm if this module works normally:

1. Take raw image from non-HDR and zzHDR sensor
2. Turn bad pixel detection off and on to check before and after
3. Check if there is any bad pixels left
4. Check if there is any significant details loss, if so, reduce the BPC/BCC strength
5. Check if there is any false color or artifacts that don't exist in BPC off image
6. Check if there is any feature or small detail loss
7. Adjust BPC settings till issues are solved or trade-off are acceptable

Test images

- Test charts and natural images
- All light conditions
  - Outdoor/indoor/lowlight/very lowlight from 1500lux to 4lux
  - Number of bad pixel is sensitive to lighting conditions, usually low light conditions has more bad pixels
- High frequency components: Trees, small texts, high frequency details



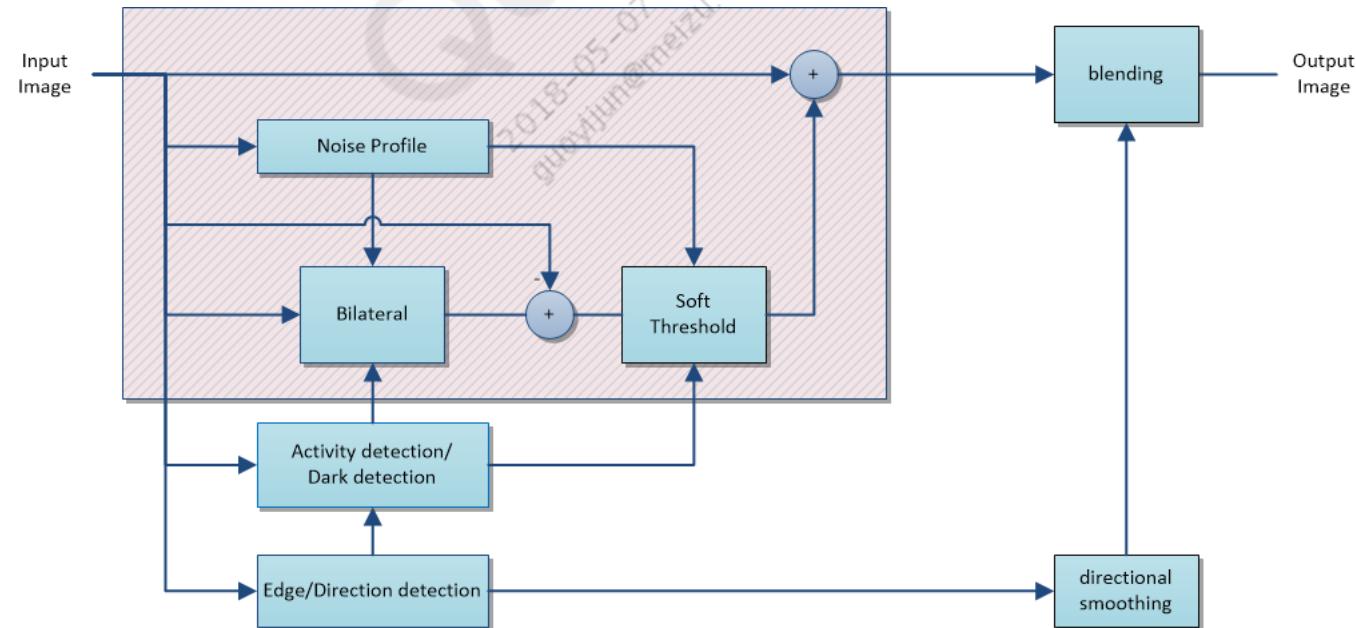


# Adaptive Bayer Filter (ABF)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Overview

- Denoising block in bayer domain
- An updated version of ABF3.4 in IFE and BPS
  - More flexible filtering controls
  - Directional smoothing
  - Dark desaturation and smoothing
  - Denoising strength adjustment based on activity



# Tuning Procedure

---

1. Follow radial and noise profile calibration procedures
  - a) Noise profile: **noise\_std\_lut**
  - b) Radial-based adjustment: **radial\_noise\_prsv\_adj** and **radial\_edge\_softness\_adj**
2. Enable desired features in the enable and reserved section
3. Tune important parameters first
  - a) Noise preservation: **noise\_prsv\_base** and **noise\_prsv\_anchor**
  - b) Denoise strength: **denoise\_strength**
  - c) Edge softness: **edge\_softness**
  - d) Radial adjustment for noise preservation: **radial\_noise\_prsv\_adj**
  - e) Radial adjustment for edge-softness: **radial\_edge\_softness\_adj**
4. Advanced tune different features
  - a) Min-Max filter (single BPC) for bad pixel correction or Peak noise reduction (PNR)
  - b) Directional smoothing
  - c) Activity-based adjustment for filtering
  - d) Dark region smoothing and desaturation
  - e) Other control knobs

# Step 1: Noise Profile and Radial Control Calibrations

---

- Noise profile: **noise\_std\_lut**
  - Defines noise level (standard deviation) vs. intensity level
  - Length: 65
  - Min: 0; Max: 512
  - Effect: Higher value stronger noise reduction
  - Calibration: Tool uses MCC chart to calibrate
- Radial adjustment for noise preservation: **radial\_noise\_prsv\_adj**
  - Defines noise preservation adjustment factor vs. radial distance percentage
  - Length: 5
  - Min: 0.0; Max: 1.0
  - Effect: Higher value, more noise preservation
  - Calibration: Tool uses gray chart to calibrate
- Radial adjustment for edge softness: **radial\_edge\_softness\_adj**
  - Defines edge-softness adjustment factor vs. radial distance percentage
  - Length: 5
  - Min: 0; Max: 15.99
  - Effect: Higher value, stronger noise reduction (blurrier result)
  - Calibration: Tool uses gray chart to calibrate

## Step 2: Parameters for Enable Controls

---

- Parameters in enable section:
  - **bilateral\_en**: noise reduction enable
  - **minmax\_en**: min-max filter enable
  - **dirsmth\_en**: directional smoothing enable
- Parameters in reserve section:
  - **cross\_plane\_en**: enable Gb and Gr cross-channel filtering
  - **dark\_desat\_en**: enable dark region desaturation
  - **dark\_smooth\_en**: enable dark region smoothing
  - **act\_adj\_en**: enable activity-based filtering adjustment

# Step 3.1: Noise Preservation Control

---

- **noise\_prsv\_base**

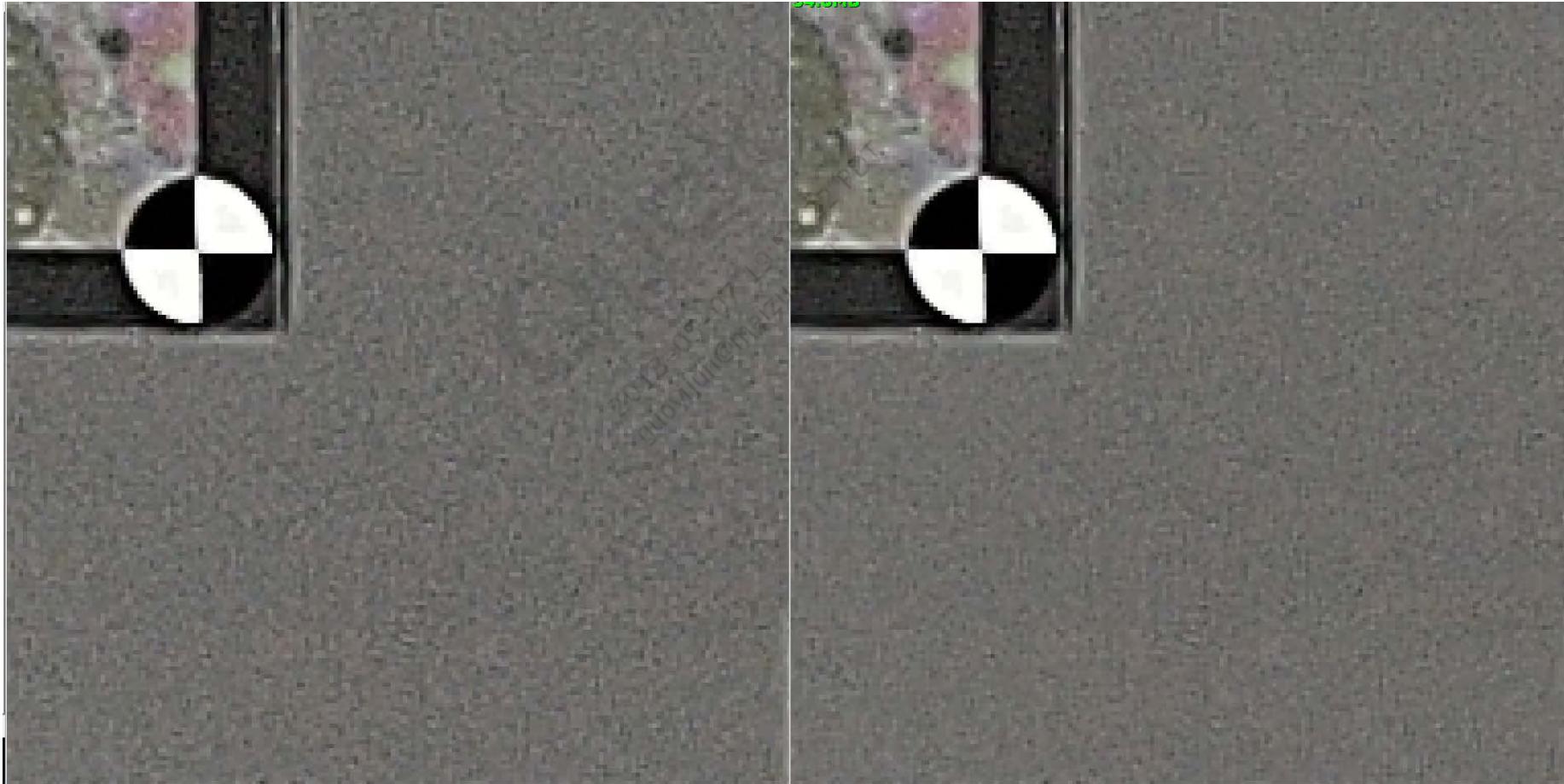
- Defines the percentile of preserved noise (similar parameter is called “filter\_strength” in the previous chipsets)
- Length: 10 (entry 1-5 for R/B channel, entry 6-10 for Gb/Gr channel)
- Default: All 0.2 array
- Min: 0; Max: 1.0
- Effect: Lower value stronger noise reduction

- **noise\_prsv\_anchor**

- Anchor table for **noise\_prsv\_base**
- Length: 5
- Default: [0.0, 0.625, 0.125, 0.800, 1.0] (The first and last entries are fixed)
- Min: 0; Max:1.0

## Step 3.1: Noise Preservation Control (cont.)

### Example



## Step 3.2: Extra Filtering Strength

---

- **denoise\_strength**
  - Adjust strength according for different channels
  - Length: 4
  - Default: [1.0, 1.0, 1.0, 1.0] ([R, Gr, Gb, B])
  - Min: 0; Max: 1.0
  - Effect: Larger value, stronger noise reduction
  - Suggestion: Tuning is only needed when noise level is different among the channels

## Step 3.3: Edge-softness

---

- **edge\_softness**
  - Adjust bilateral filter strength for different channels
  - Length: 4
  - Default: [3.0, 3.0, 3.0, 3.0] ([R, Gr, Gb, B])
  - Min: 0; Max: 15.99
  - Effect: Larger value, stronger noise reduction

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

## Step 3.4: Radial Adjustment

- **radial\_noise\_prsv\_adj**

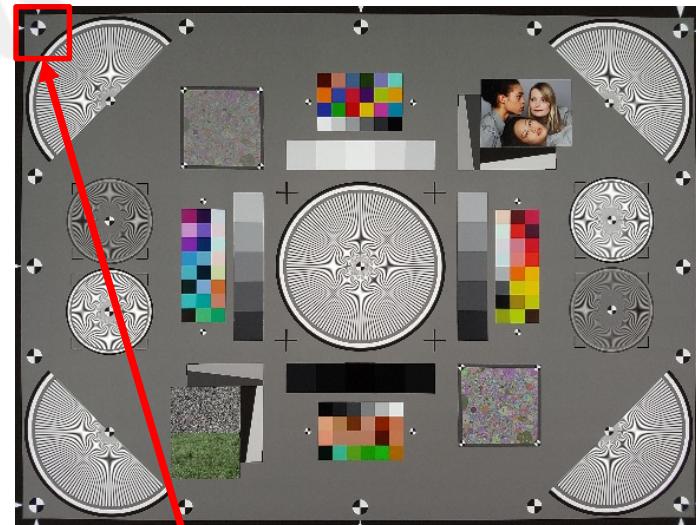
- Defines noise preservation adjustment factor vs. radial distance percentage
    - Length: 5
    - Min: 0.0; Max: 1.0
    - Effect: Smaller value, stronger noise reduction

Examples:

- radial anchor: [0.0 0.4 0.6 0.8 1.0]
  - setting #1: [1.0000 1.0000 1.0000 1.0000 1.0000]
  - setting #2: [1.0000 0.8750 0.7500 0.6250 0.5000]

- **radial\_edge\_softness\_adj**

- Defines edge-softness adjustment factor vs. radial distance percentage
    - Length: 5
    - Min: 1.0; Max: 15.99
    - Effect: Larger value, stronger noise reduction

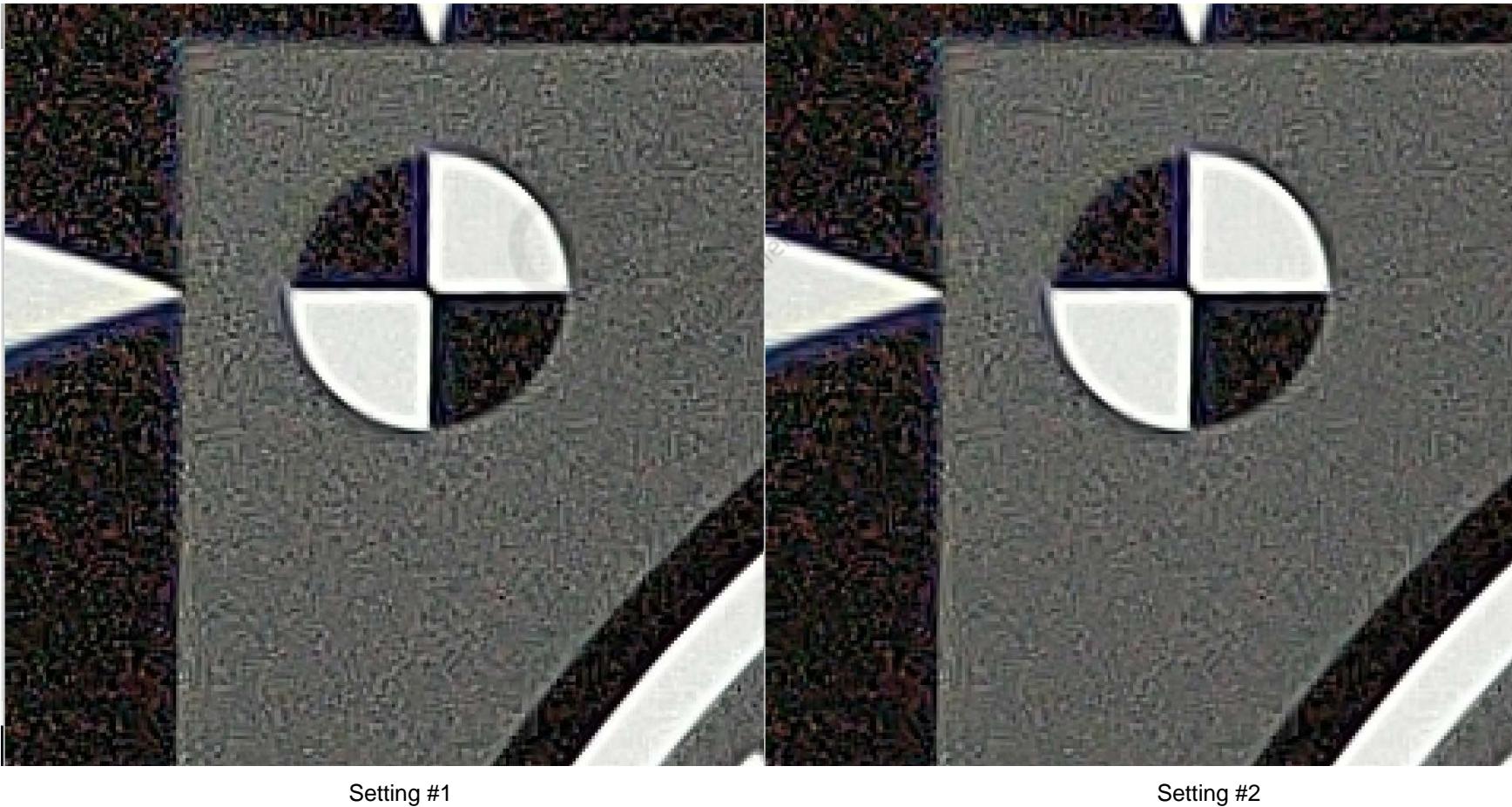


## Step 3.4: Radial Adjustment (cont.)

Example of radial adjustment for noise preservation

setting #1: [1.0000 1.0000 1.0000 1.0000 1.0000]

setting #2: [1.0000 0.8750 0.7500 0.6250 0.5000]



# How to Coordinate with Other Modules

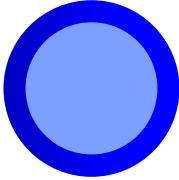
---

- ABF is not the main block for denoising in Qualcomm Spectra 2xx
  - ANR + TF for video denoising
  - HNR + ANR for snapshot denoising
  - HNR + ANR + TF for multi-frame denoising
- ABF strength (Bilateral filtering strength) should be set to mild in order to preserve more detail/texture
- Kernel size of ABF is similar to the kernel size of GIC, HNR and first pass of ANR
  - For snapshot, balance is among GIC, HNR and ABF4.0
  - For video, balance is among GIC, ABF3.4 and first pass of ANR
- ABF can be used to balance noise level for different channels
  - **filter\_strength**
  - **noise\_prsv\_base**

# How to Coordinate with Other Modules (cont.)

---

- For special RCCB or RGBW sensor, ABF4.0 could help reduce more noise in the early part of the pipeline
  - **blk\_opt**: using Gr and Gb channel in block matching
  - Dark smoothing and dark desaturation
- Directional smoothing should be turned off when one works on an initial spatial tuning
  - Rely on edge alignment in ASF3.0
- Activity based strength adjustment should be turned off during initial spatial tuning
  - Rely on HNR1.0
- Min-Max filter should be off in normal tuning. We should rely on:
  - BPC/BCC/PDPC2.0
  - PNR in GIC



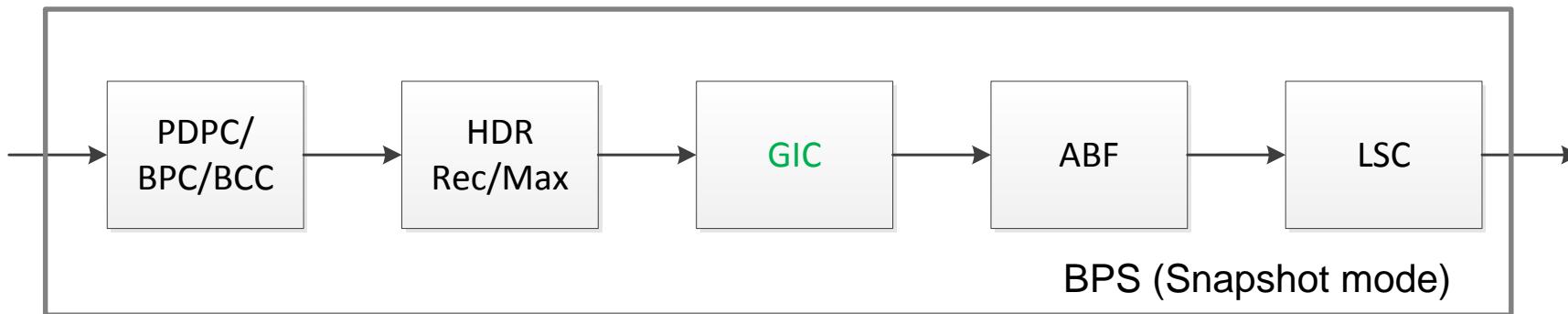
# Green Imbalance Correction (GIC)

2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Green Imbalance Correction (GIC) – Introduction

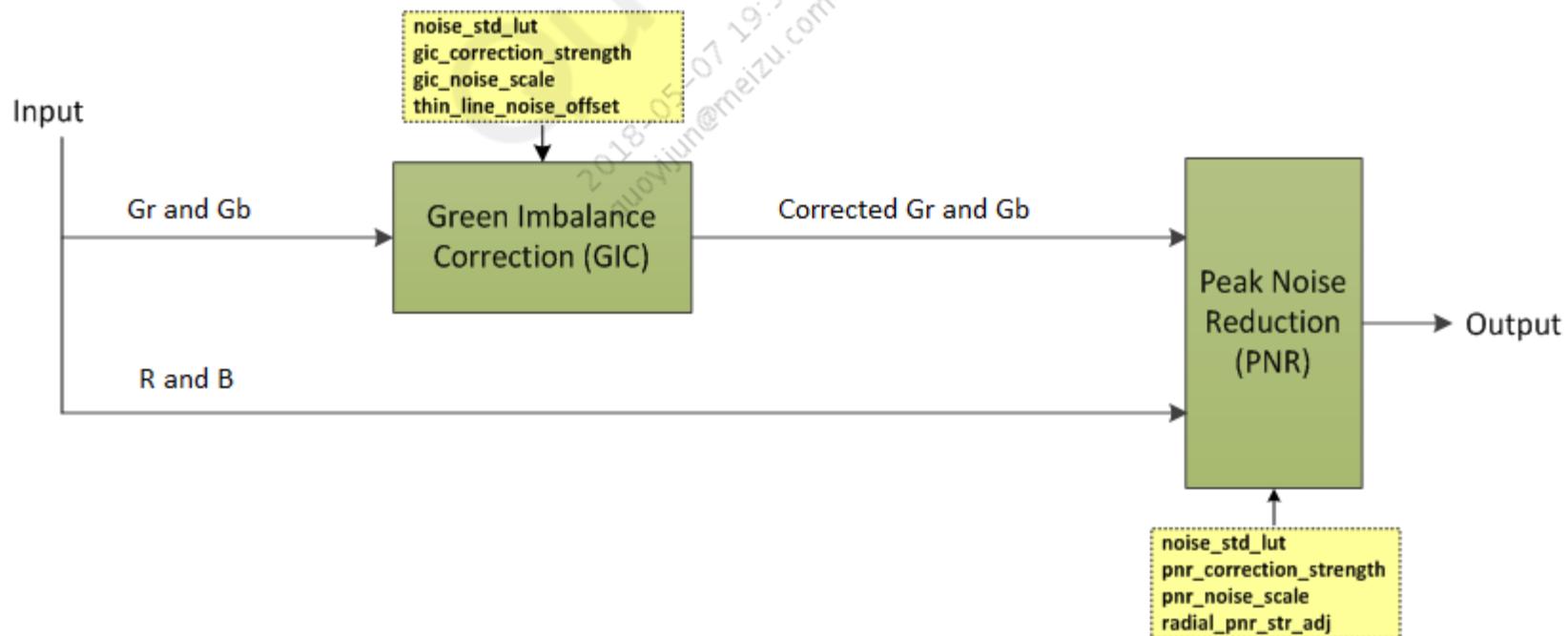
## Problem statement

- Some Bayer sensors have local imbalance between Gr and Gb channels and the imbalance becomes more noticeable as the sensor resolution increases recently
- The imbalance cannot be corrected by operations like channel gain and lens roll-off correction. ABF can correct some amount of green imbalance artifacts, but if ABF is set too strong, it may cause significant blurring
- GIC isolates Gr/Gb imbalance from signals and efficiently reduce the local imbalance
- On Qualcomm Spectra 2xx BPS, GIC is applied in Bayer domain and located before ABF and Lens shading correction (LSC) modules



# Step 1: Overview

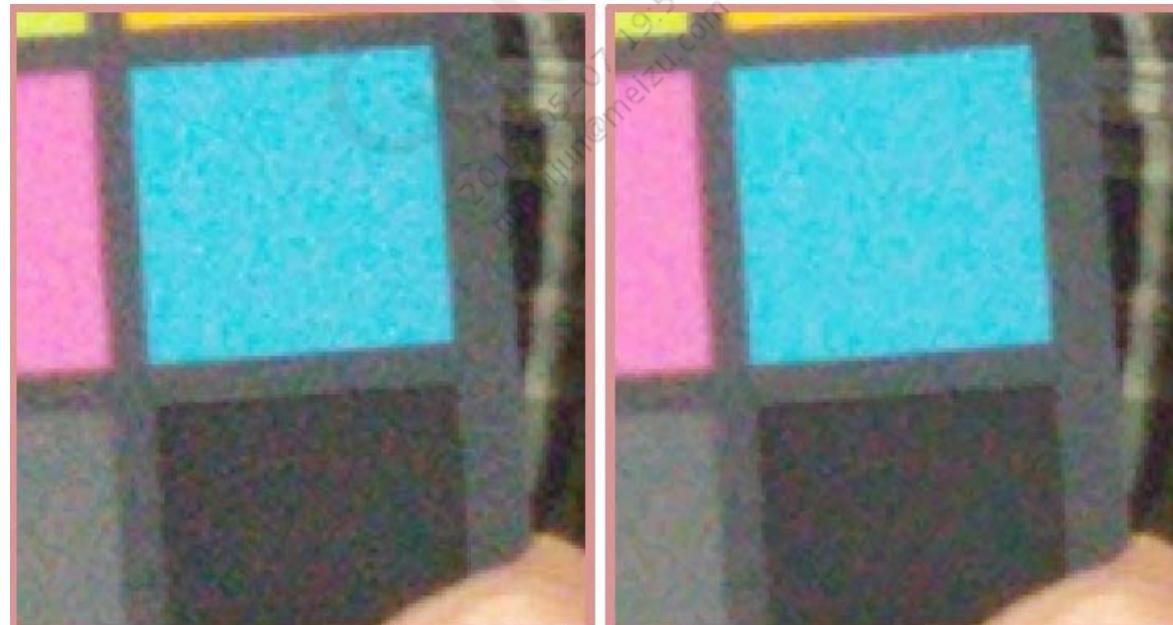
- Qualcomm Spectra 2xx GIC is an update module combined with green imbalance correction and PNR. It is located in BPS, which is used for snapshot processing only
- Imbalance artifacts of Gb and Gr channels are corrected first
- Corrected Gb and Gr channels work with R and B channels for PNR



## Step 2: Parameters to Enable Controls

- Parameters in enable section:
  - `gic_global_enable`: Enable/disable entire GIC/PNR module
- Parameters for sub-module enable:
  - `enable_gic`: Enable/disable GIC sub-module
  - `enable_pnr`: Enable/disable PNR sub-module

Examples:



Left: GIC and PNR Off, Right: GIC and PNR On

## Step 3.1: GIC Correction Strength

---

- **gic\_correction\_strength**
  - Green imbalance correction strength
  - Length: 1
  - Default: 0.6
  - Min: 0; Max:1
  - Effect: Lower value results in stronger correction

Examples:



gic\_correction\_strength = 1.0

gic\_correction\_strength = 0.6

## Step 3.2: GIC Noise Scale

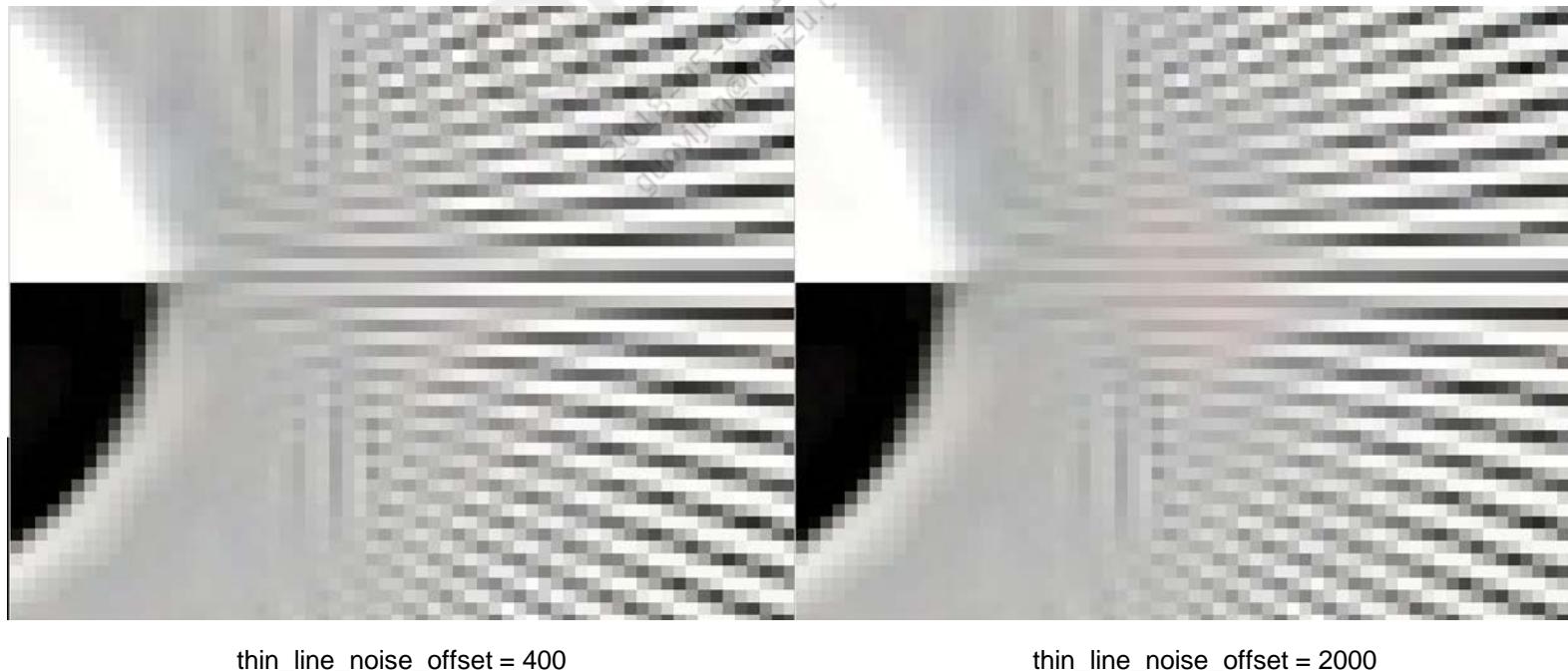
---

- [gic\\_noise\\_scale](#)
  - Noise scale factor of green imbalance correction which is used for Gb/Gr channel difference clamping
  - Length: 1
  - Default: 1.0
  - Min: 0; Max: 15.98
  - Effect: Higher value results in weaker correction

## Step 3.2: GIC Thin Line Noise Offset

- **thin\_line\_noise\_offset**
  - Noise offset of edge and thin line detection
  - Length: 1; Default: 400
  - Min: 0; Max: 16383
  - Effect: Higher value results in weaker thin line detection (thin lines might be impacted by GIC)

Examples:



# Step 4.1: PNR Correction Strength

---

- `pnr_correction_strength`
  - Peak noise reduction strength
  - Length: 1
  - Default: 0.6
  - Min: 0; Max:1
  - Effect: Lower value results in stronger reduction

Examples:



`pnr_correction_strength = 1.0`

`pnr_correction_strength = 0.6`

## Step 4.2: PNR Noise Scale

---

- **pnr\_noise\_scale**
  - Noise scale factor of PNR for each channel which is used to separate flat region and texture region
  - Length: 4
  - Default: [1.0, 1.0, 1.0, 1.0]
  - Min: 0; Max: 15.98
  - Effect: Higher value, more area could be detected as flat region
  - Suggestion: setting **pnr\_noise\_scale** could impact texture region

# Advanced Step 5.1: Noise Profile for GIC and PNR

---

- **noise\_std\_lut**
  - Noise profile (standard deviations) vs. local intensity
  - Length: 65
  - Default: calibrated and sharing with ABF4.0
  - Min: 0; Max: 512.0
  - Effect: Higher value results in weaker green imbalance correction and stronger PNR

## Advanced Step 5.2: Radial PNR Strength Adjustment

---

- **radial\_pnr\_str\_adj**
  - Adjust PNR strength based on radial distance percentage
  - Length: 7
  - Default: 1.0
  - Min: 0; Max: 15.98
  - Effect: Smaller value results in stronger PNR

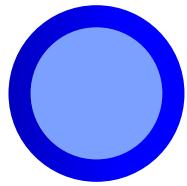
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Tuning Procedure

---

1. Follow radial and noise profile calibration procedures
2. Enable desired features in enable flag section
3. Tuning parameters to reduce green imbalance artifacts
  - a) Tuning correction strength: [gic\\_correction\\_strength](#)
  - b) Tuning noise scale: [gic\\_noise\\_scale](#)
  - c) Adjust thin line processing: [thin\\_line\\_noise\\_offset](#)
4. Tuning parameters for PNR
  - a) Tuning correction strength: [pnr\\_correction\\_strength](#)
  - b) Tuning noise scale: [pnr\\_noise\\_scale](#)
5. Advanced tuning
  - a) Fine tune noise profile: [noise\\_std\\_lut](#)
  - b) PNR radial-base strength adjustment: [radial\\_pnr\\_str\\_adj](#)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

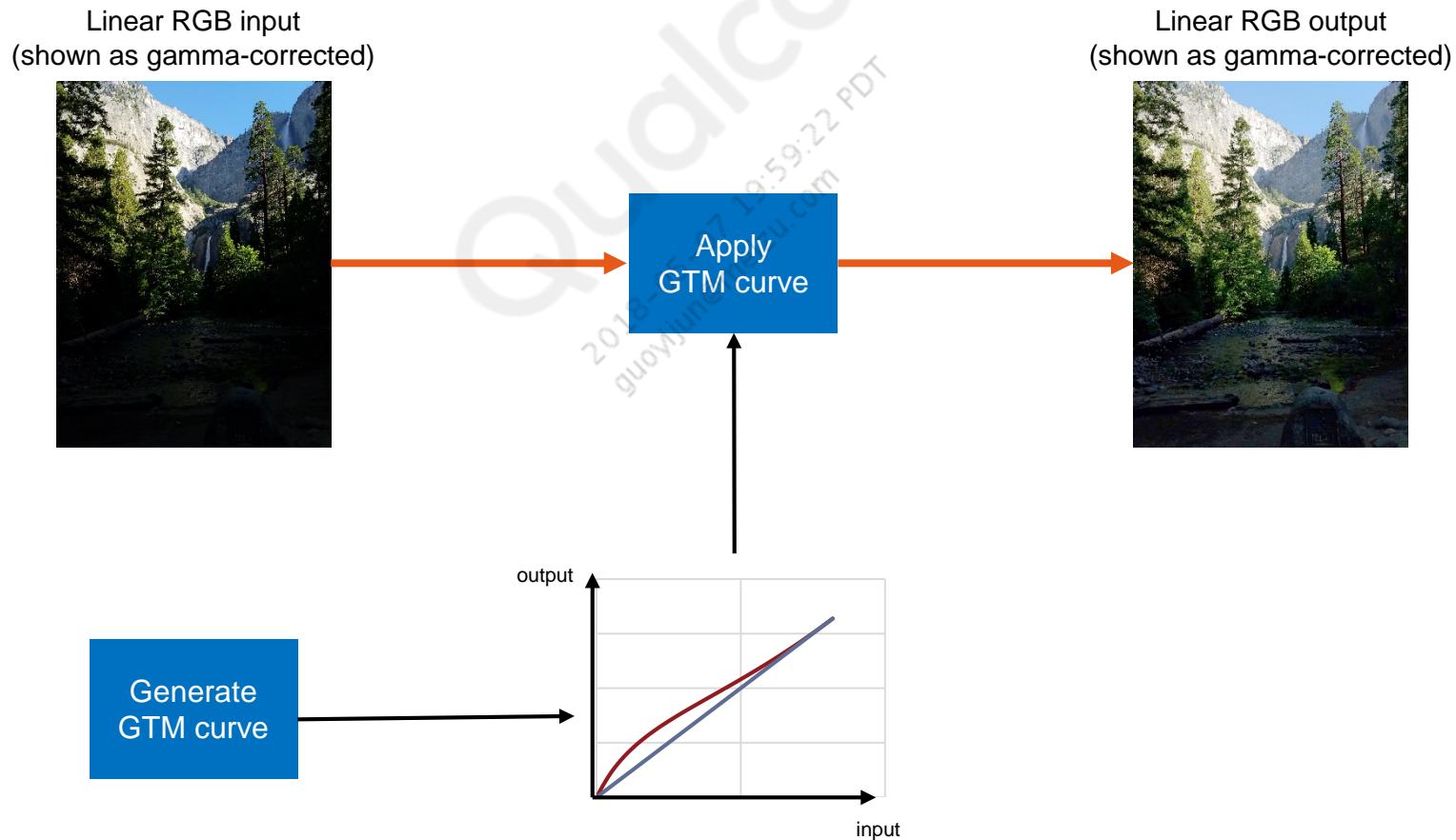


# Global Tone Mapping (GTM)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Introduction

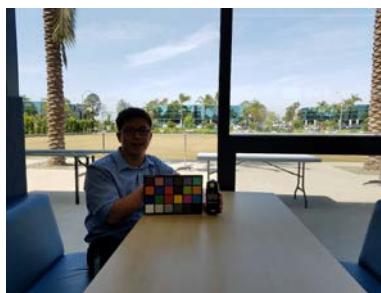
- GTM applies a global tone curve to each pixel to enhance visibility. It is used in HDR or automatic dynamic range compression (ADRC) application to compensate the enhanced image dynamic range.



# HDR: GTM/LTM Tuning Image Capture

- Capture HDR raw images, covering 6 exposure lux index zones and exposure ratios from 1x to 16x
- For each scene, capture raw images of HDR mode and non-HDR mode
- HDR scenes image content suggestions:
  1. Window scene: Half bright outdoor, half dark indoor, containing a person holding MCC chart by the window, bright outdoor scene through the window.
  2. Shade scene: Outdoor scene with half bright sun and half dark shade.
  3. Outdoor texture: A person standing under direct sunlight, holding a texture test chart. There are dark shade on the background.
  4. Backlit scene: A person holding MCC chart under shade with bright background, for example, sky with clouds.

Lux Index	Exp Ratio
100 - 200	1 – 2
200 – 250	2 – 4
250 - 300	4 – 6
300 – 380	6 – 8
380 – 450	8 – 12
450 - 600	12 - 16



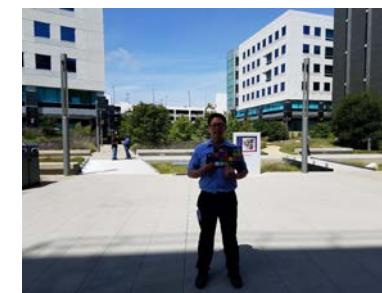
1. Window scene



2. Shade scene



3. Outdoor texture



4. Backlit scene

# HDR: GTM/LTM Tuning Image Capture (cont.)

5. Garage scene: Inside the parking building with bright outside background
6. Building scene: Buildings with dark shadows
7. Night scene: Building with lights at night
8. Indoor scene: Indoor decoration with small outdoor areas
9. Nature scene: Natural objects including trees, grass, water and buildings, containing highlights and shadows
10. Resolution chart in lab scene. The resolution chart are placed half in the bright side and half in the dark side.
11. Night scene with people's face, shop windows/lights. It would be best if the person can swing the hands or so on to generate some motion.



5. Garage scene



6. Building scene



7. Night scene



8. Indoor scene



9. Nature scene

# GTM Tuning Introduction

---

- GTM tuning is used to generate GTM curve
- GTM curve generation automatically considers LSB or MSB mode
- There are two modes to generate GTM curve: Auto mode and Manual mode, which is controlled by *manual\_curve\_enable*. Default value is 0 for auto mode.
- GTM tuning table is 2D, triggered by lux\_index/gain and exp\_ratio, covering low light, normal light, high light regions and 6 exposure ratios
- GTM curves can be tuned for each of lux zone and exposure ratio based on OEMs requirement

# GTM Tuning Procedure

---

- Set all parameters to their default values
- Decide the mode (Manual or Auto)
- For each lux region and exposure ratio, tune GTM parameters so that the middle tone is similar to non-HDR mode
- Auto mode tuning: *a\_middletone, middletone\_w*

GTM curve is automatically generated based on *a\_middletone* and *middletone\_w*

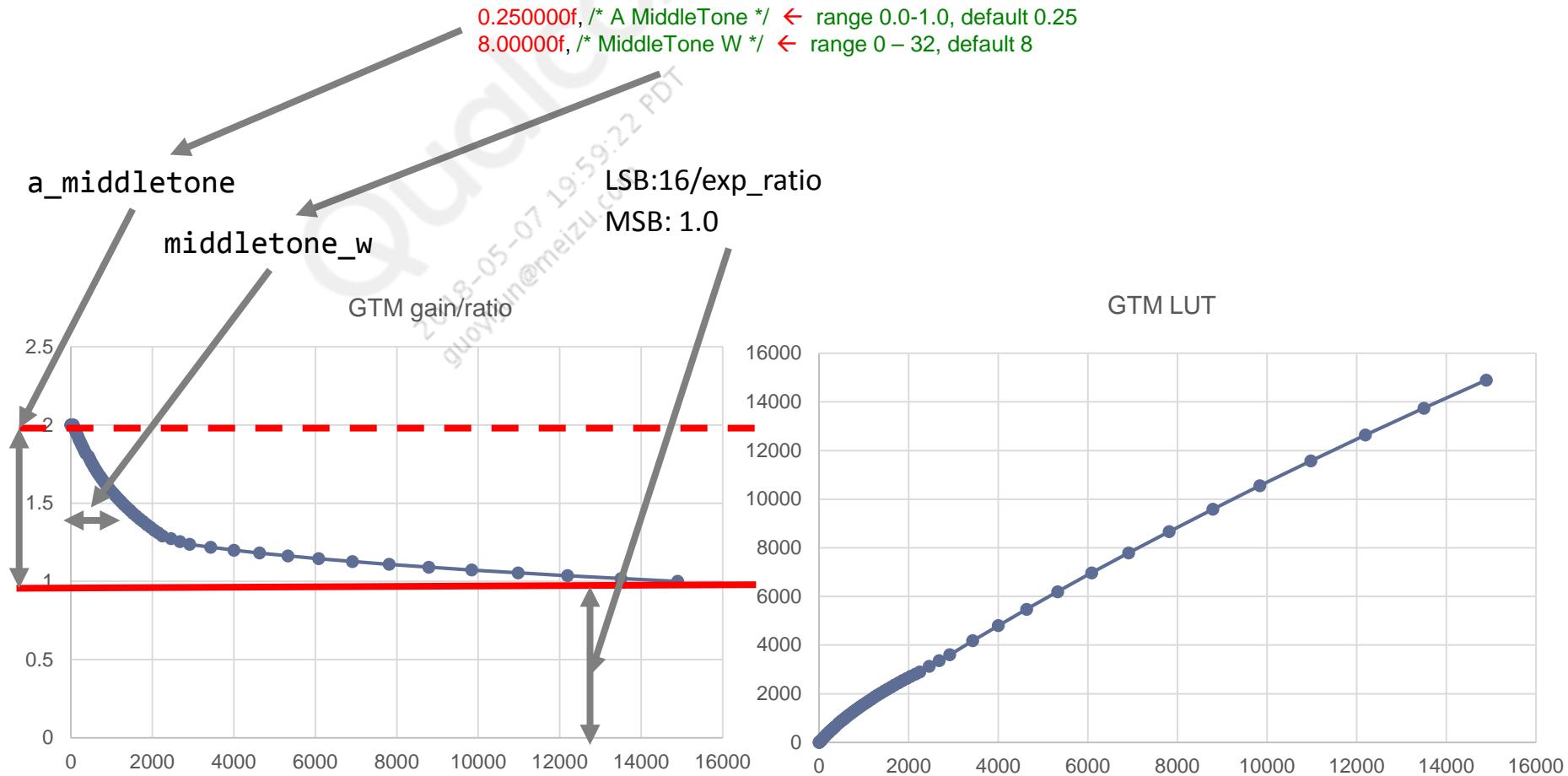
- Manual mode tuning: *a\_middletone, yratio\_base\_manual* [65]

GTM curve is based on *yratio\_base\_manual*, and adjusted by *a\_middletone*

# GTM Tuning Procedure (cont.)

- Auto mode tuning

Increasing *a\_middletone* and *middletone\_w* will increase image brightness.



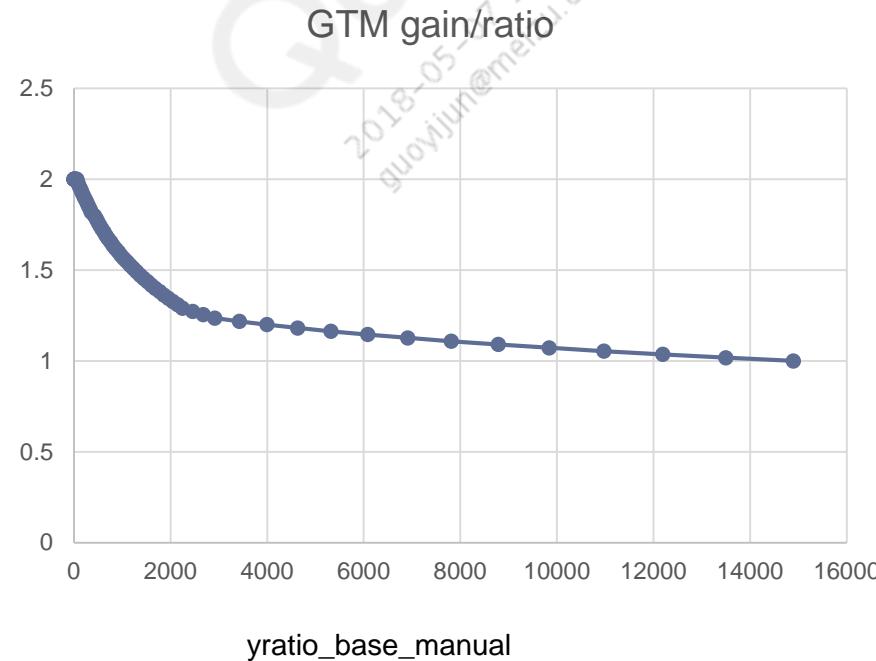
# GTM Tuning Procedure (cont.)

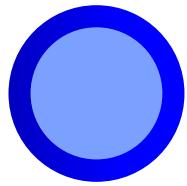
- Manual mode tuning

Manual curve *yratio\_base\_manual* is defined for each lux zone and exposure ratio.

Increasing *a\_middletone* will increase image brightness.

$$\text{Yratio\_base} = \text{a\_middletone} * \text{yratio\_base\_manual}$$





# Hybrid Noise Reduction (HNR)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Hybrid Noise Reduction (HNR) – Introduction

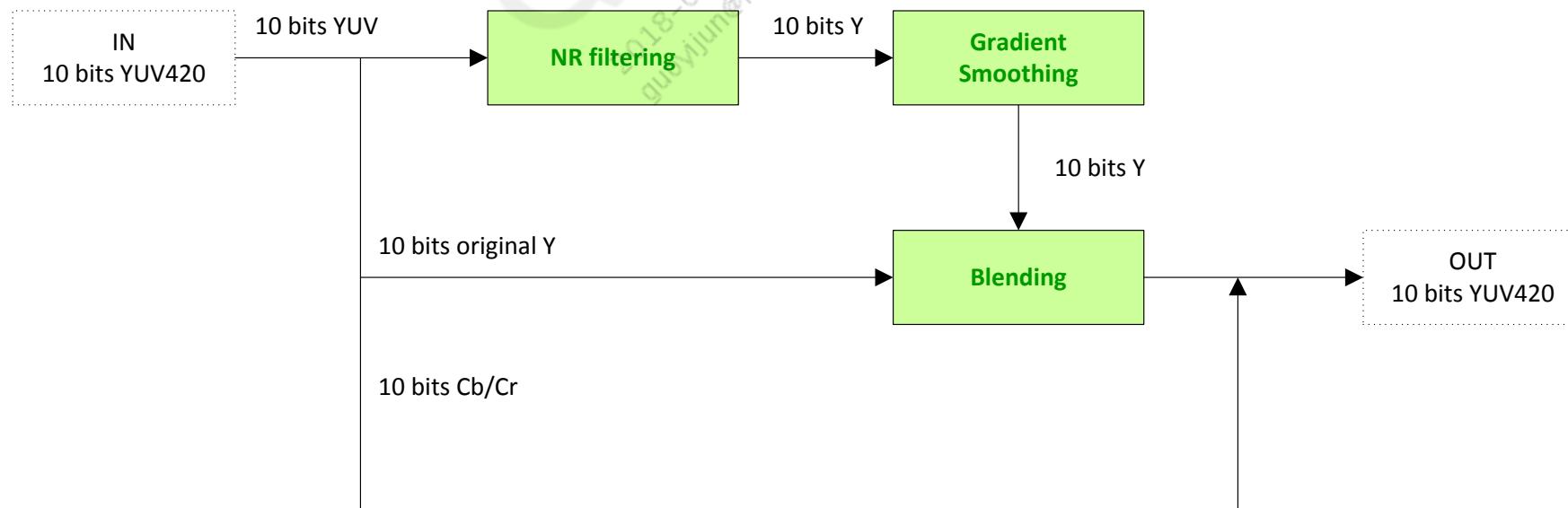
## Problem statement

- HNR module is a blending architecture to reduce luminance noise but to keep texture details
- HNR is located at the end of BPS, which means it's available for snapshot only
- The module consists of DCT-based frequency domain noise reduction, gradient smoothing and spatial domain blending
- Unlike conventional spatial domain noise reduction, DCT-based frequency domain takes advantages to distinguish weak texture and noise. Using this, HNR reduces noise while preserving edges and texture



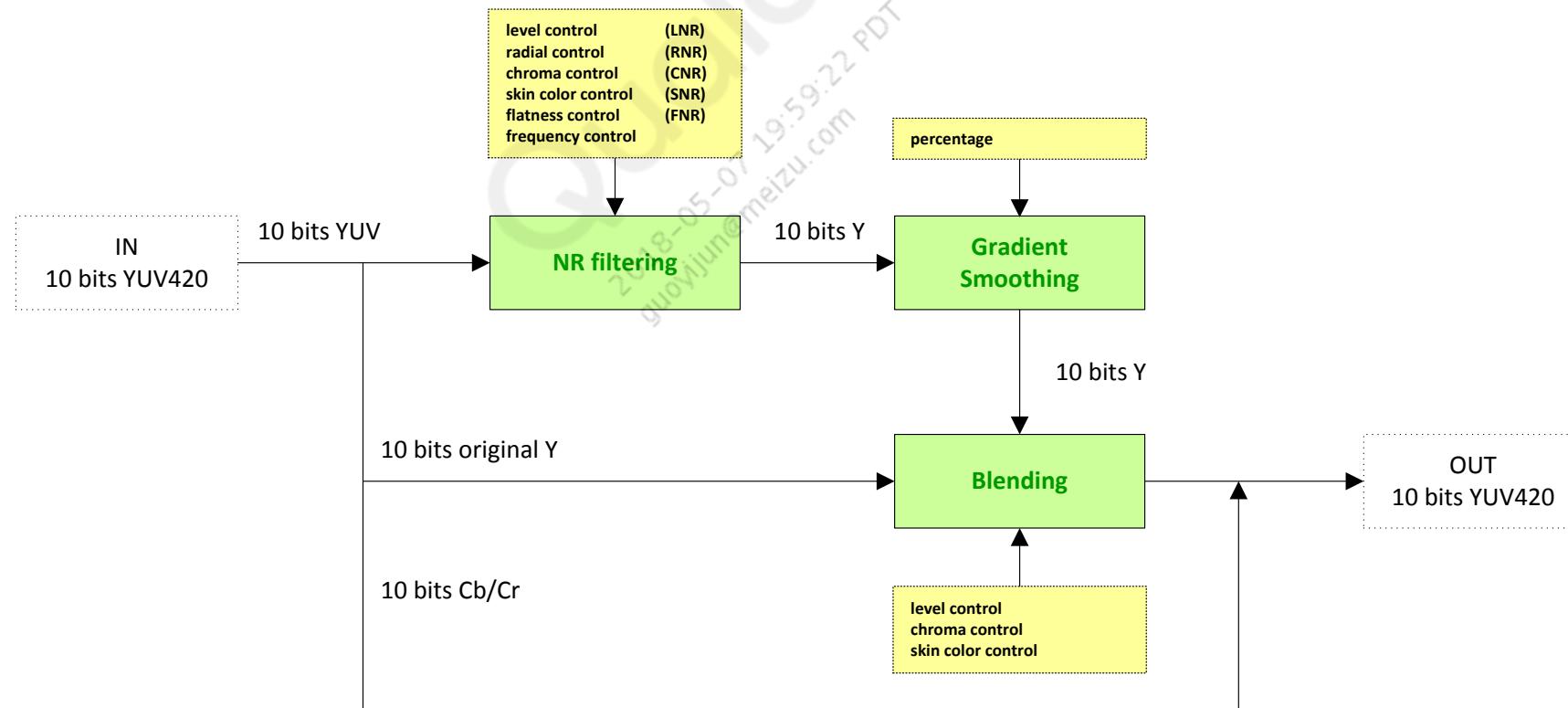
# Step 1: Overview

- HNR in BPS
  - Spatial and frequency-domain noise reduction
  - Blending with input image with weighting factors
  - Gradient smoothing for jaggy edge treatment
  - Multiple controls in both noise reduction and blending stages
- Luma noise reduction only
- 10-bit input and 10-bit output



# Step 1: Overview (cont.)

- Noise reduction controls: Level, radial, chroma, skin, flatness and, frequency
- Blending controls: Level, chroma, skin
- Gradient smoothing: Percentage



## Step 2: Parameters to Enable Controls

---

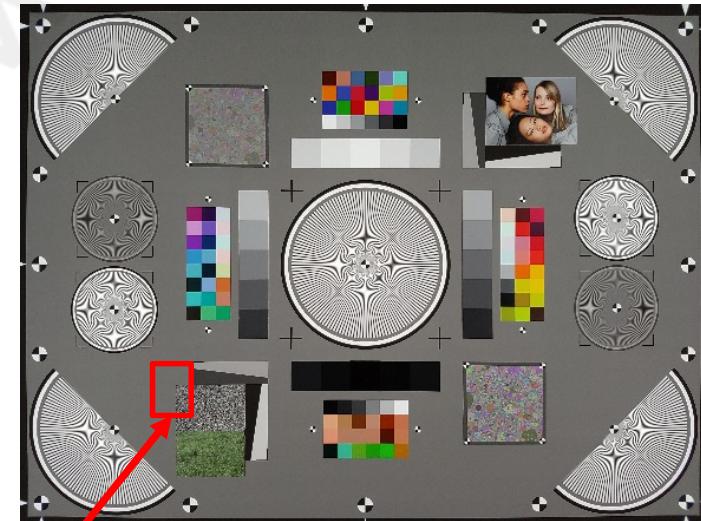
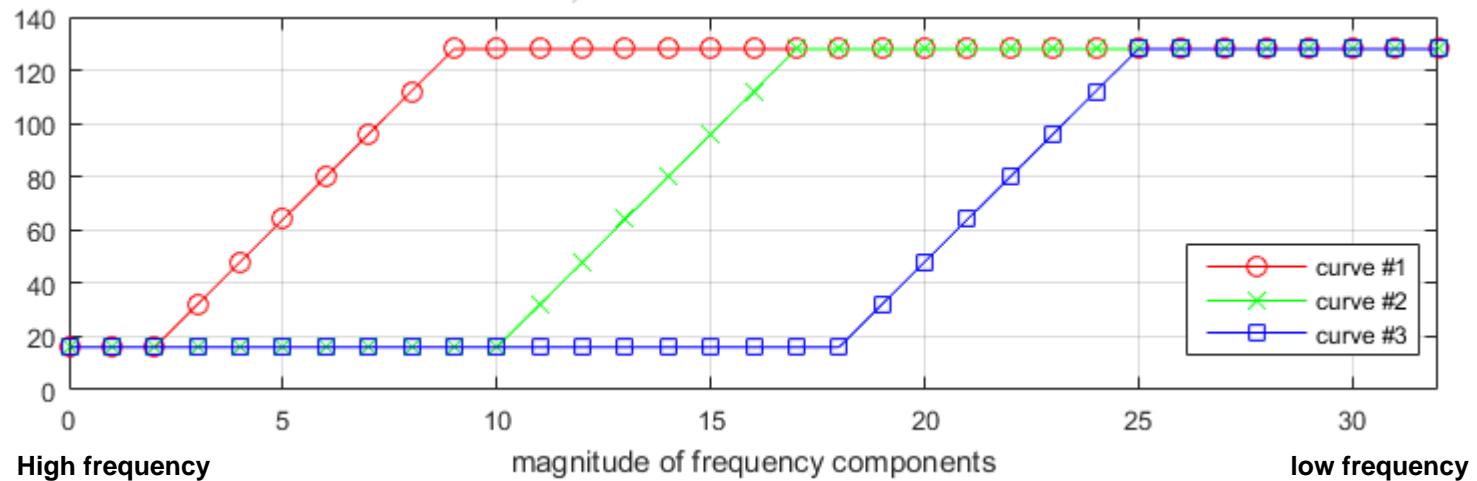
- Parameters in the Enable section:
  - `hn_r_nr_enable`: noise reduction enable
  - `hn_r_blend_enable`: noise blending enable
- Parameters in the Reserve section:
  - `lnr_en`: level-based noise reduction enable
  - `rnr_en`: radial-based noise reduction enable
  - `cnr_enable`: chroma-based noise reduction enable
  - `snr_enable`: skin-color based noise reduction enable
  - `fd_snr_enable`: face detection assisted SNR enable
  - `fnr_enable`: flatness control enable
  - `lpf3_en`: gradient smoothing enable
  - `blend_cnr_en`: chroma-based blending enable
  - `blend_snr_en`: skin-color based blending enable

2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Step 3.1: Noise Reduction Strength

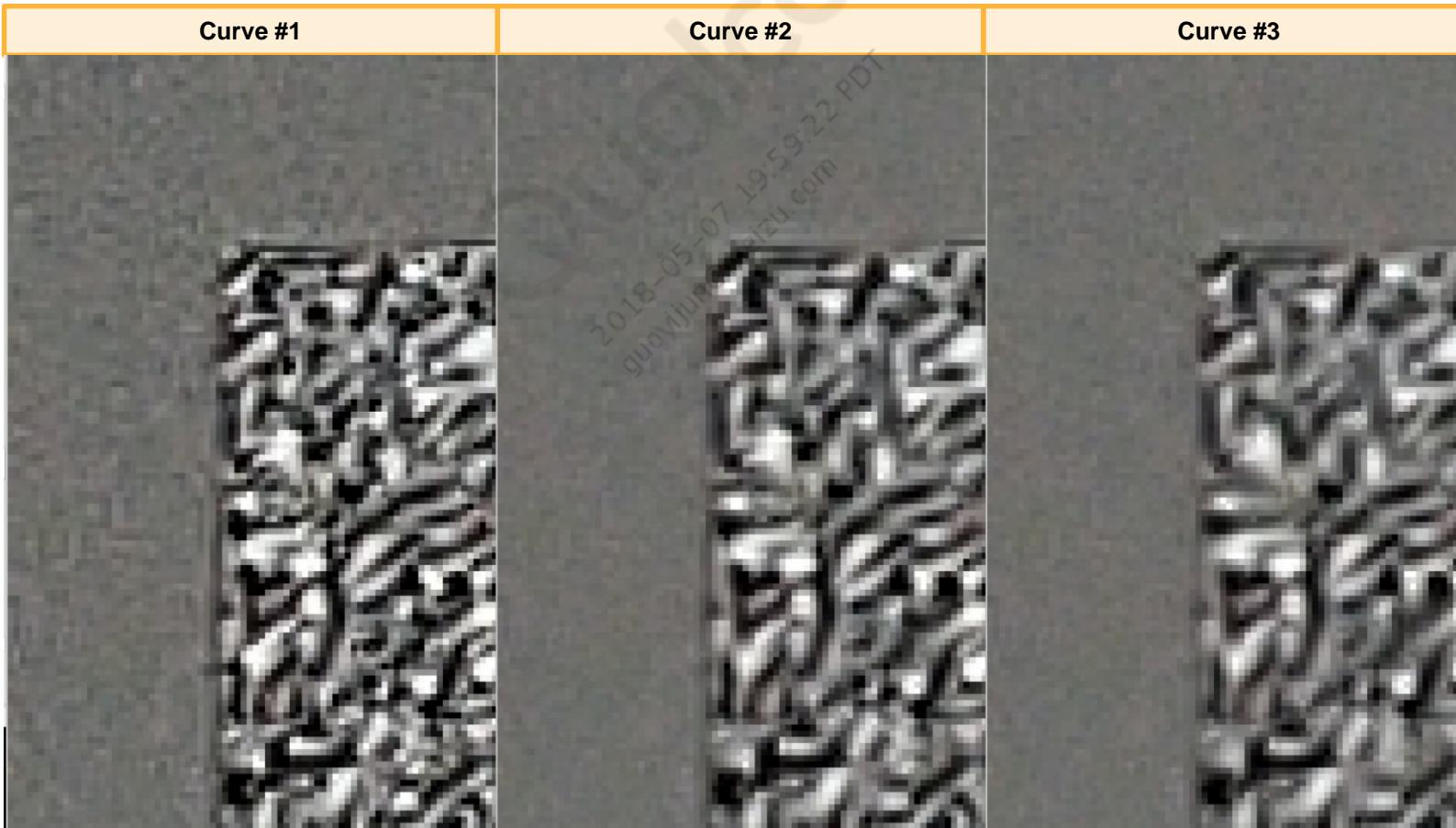
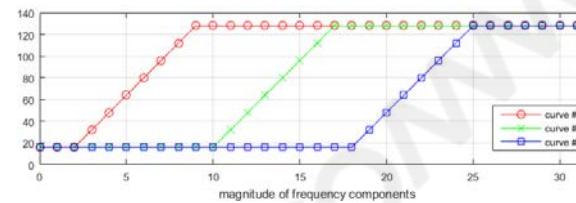
- filtering\_nr\_gain\_arr
  - Description: Overall strength of HNR
  - Length: 33
  - Default: All 128 array
  - Min: 0; Max: 255
  - Effect: Lower value stronger noise reduction
  - Calibration: Tool uses MCC chart to calibrate
  - Suggestion: Piece-wise smooth curve

Examples:



## Step 3.1: Noise Reduction Strength (cont.)

- filtering\_nr\_gain\_arr

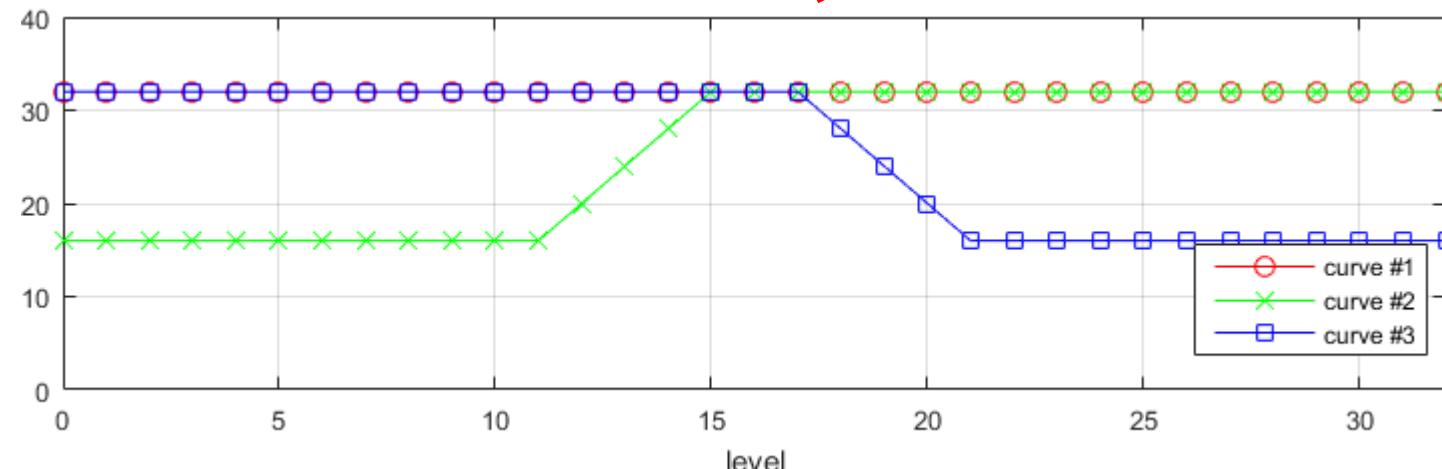
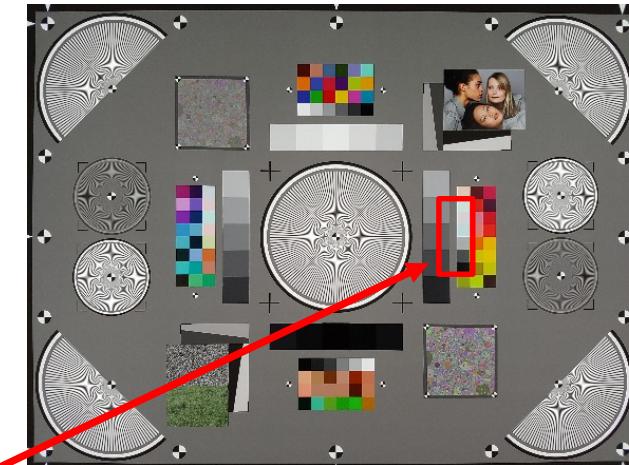


## Step 3.2: Level-Based Noise Reduction Adjustment

- **Inr\_gain\_arr**

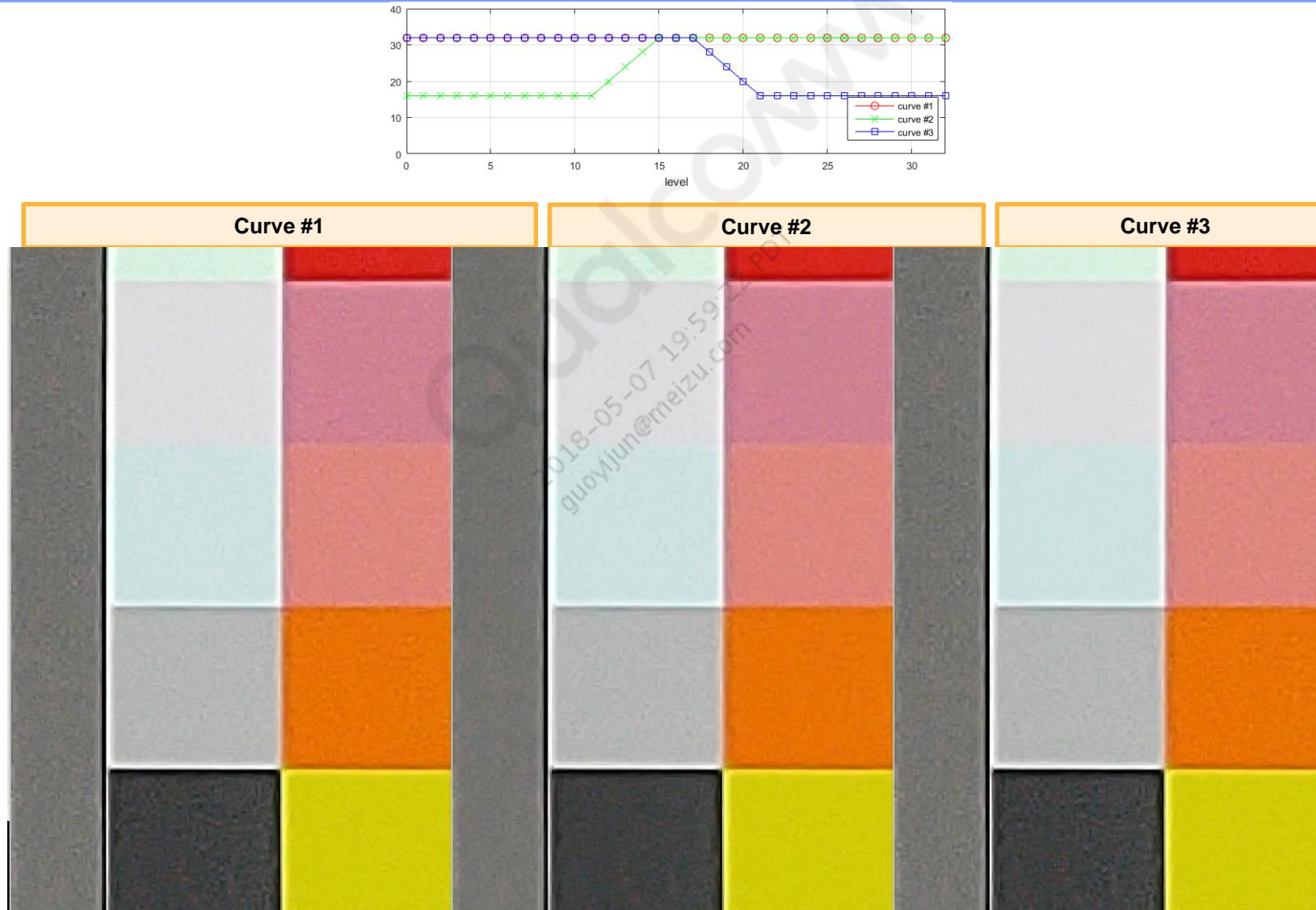
- Description: Adjust strength according to intensity level
- Length: 33
- Default: All 32 array
- Min: 0; Max: 63
- Effect: Lower value stronger noise reduction
- Calibration: None
- Suggestion: Piece-wise smooth curve
- Examples:

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com



## Step 3.2: Level-Based Noise Reduction Adjustment (cont.)

- lnr\_gain\_arr



## Step 3.3: Radial-Based Noise Reduction Adjustment

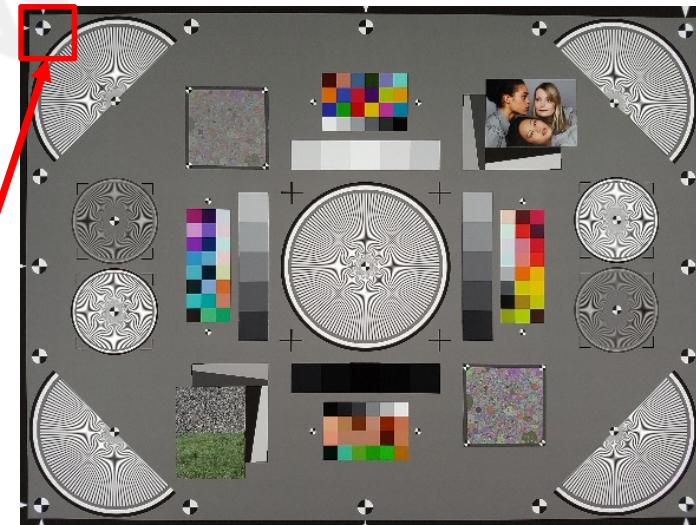
- **radial\_noise\_prsv\_adj**

- Description: Adjust strength based on radial distance
- Length: 7
- Default: All 1.0 array
- Min: 0; Max:1.0
- Effect: Lower value stronger noise reduction
- Calibration: Tool uses flat-field image
- Suggestion: Monotonically decreasing

Examples:

- radial anchor: [0.0 0.2 0.3 0.4 0.6 0.8 1.0]
- setting #1: [1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000]
- setting #2: [1.0000 1.0000 0.9375 0.8750 0.7500 0.6250 0.5000]

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

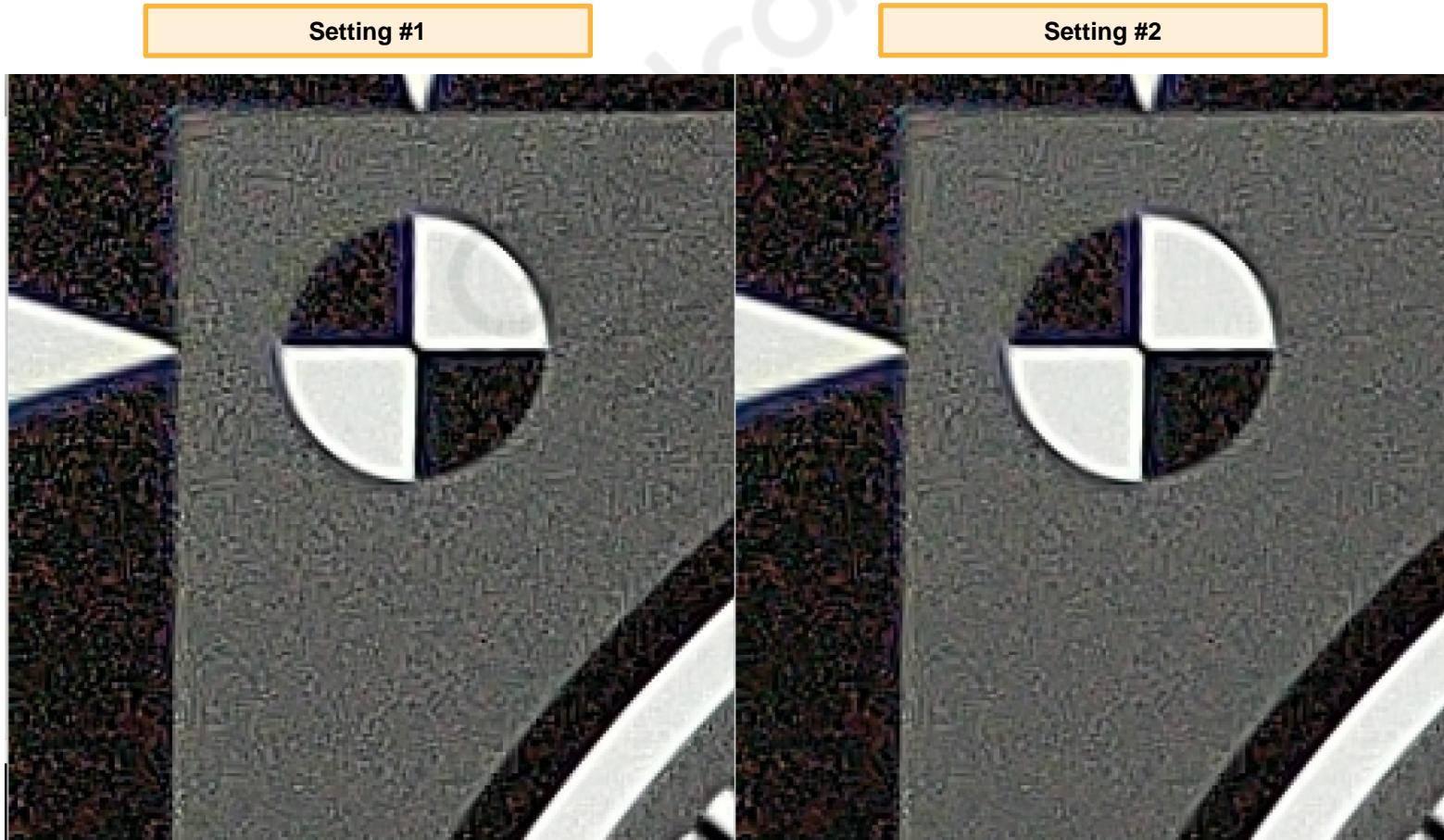


## Step 3.3: Radial-Based Noise Reduction Adjustment (cont.)

- `radial_noise_prsv_adj`

Setting #1: [1.0000 1.0000 1.0000 1.0000 1.0000 1.0000 1.0000]

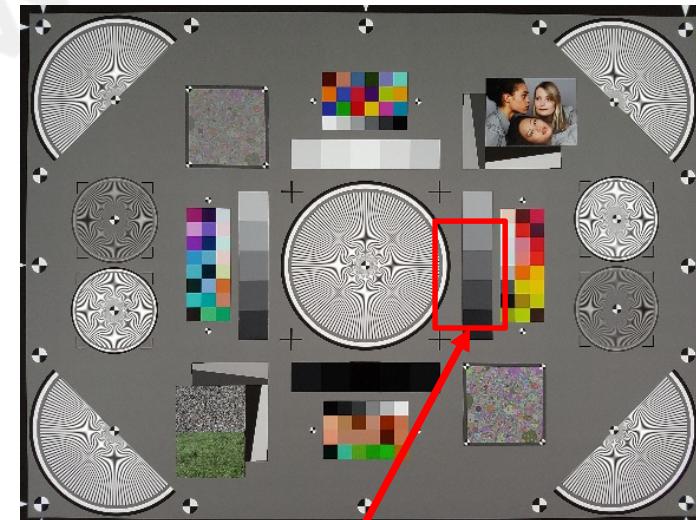
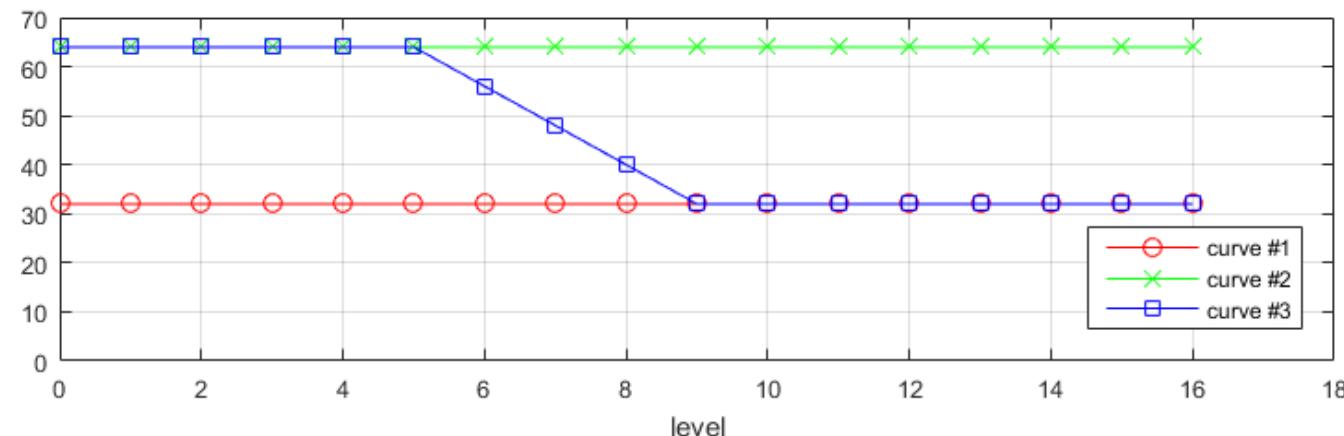
Setting #2: [1.0000 1.0000 0.9375 0.8750 0.7500 0.6250 0.5000]



## Step 3.4: Level-Based Noise Blending Adjustment

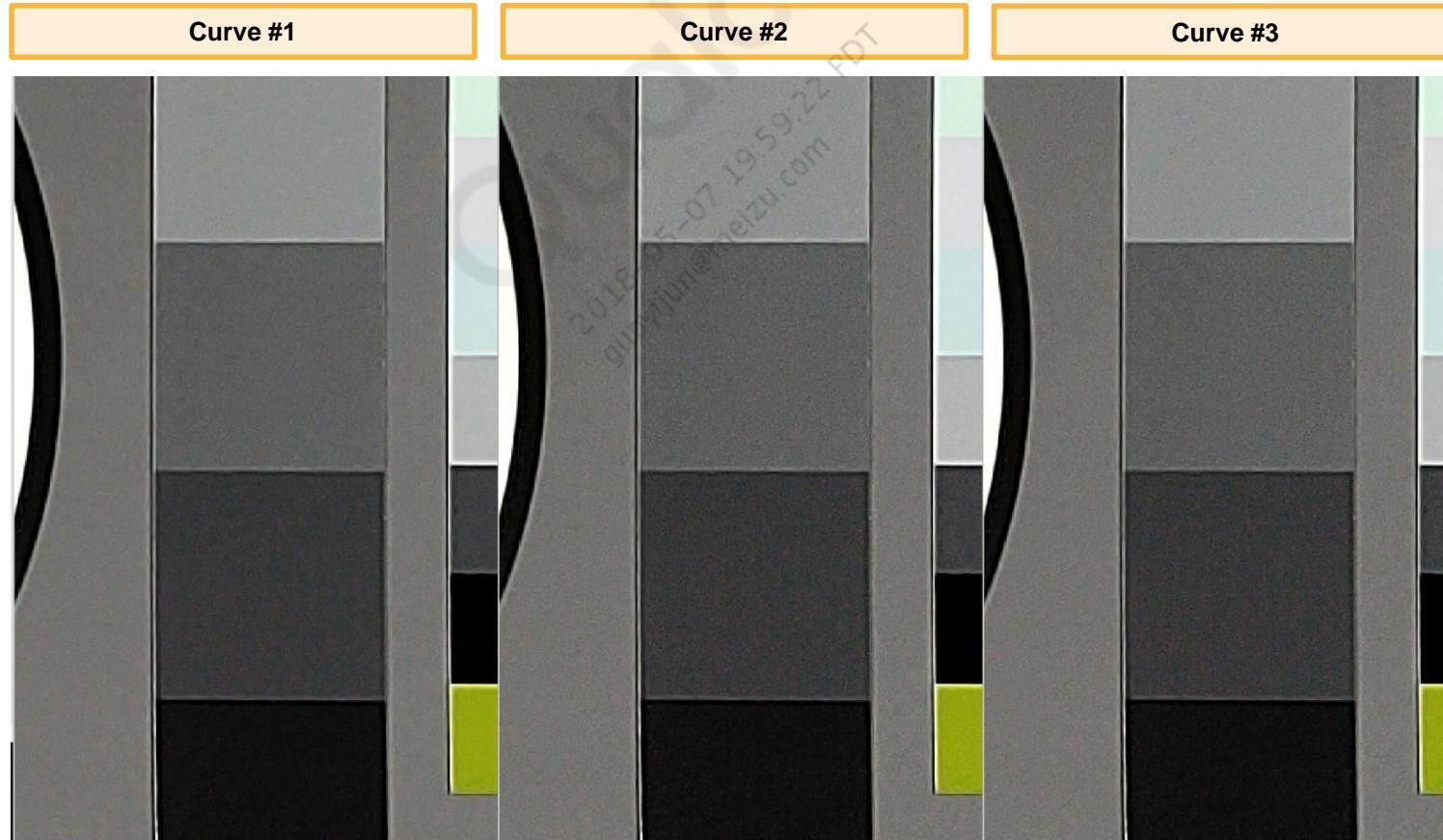
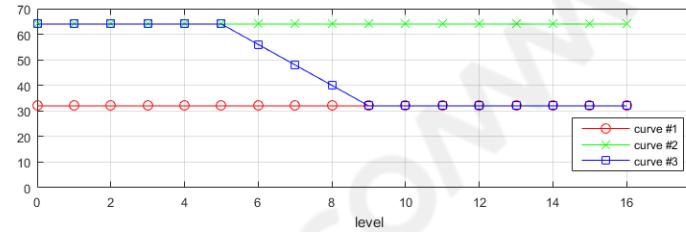
- blend\_lnr\_gain\_arr
  - Description: Blending weight of input luma channel
  - Length: 17
  - Default: All 32 array
  - Min: 0; Max: 255
  - Effect: Higher value noisier output
  - Calibration: None
  - Suggestion: "U" shaped curve to preserve more details on both dark and highlight regions.

Examples:



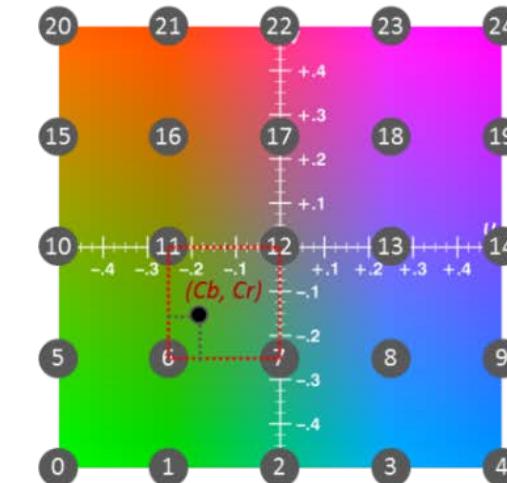
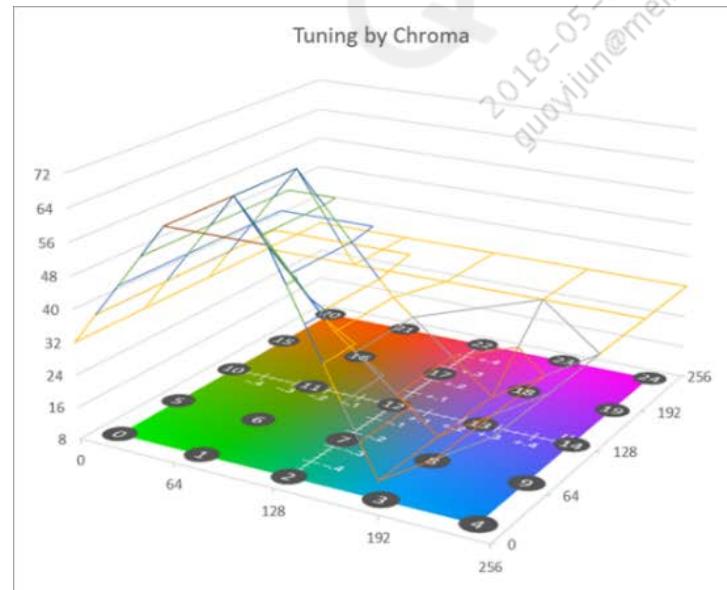
## Step 3.4: Level-Based Noise Blending Adjustment (cont.)

- blend\_lnr\_gain\_arr



# Step 4.1: Chorma-Based Noise Reduction Strength Adjustment

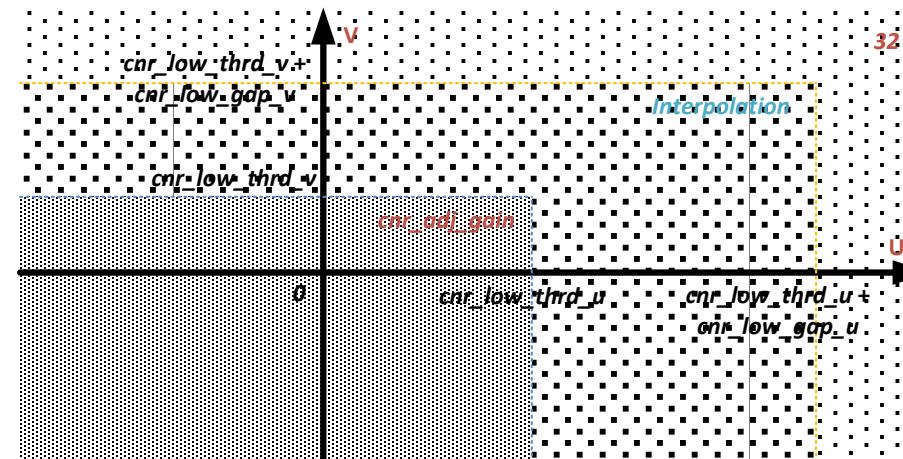
- cnr\_gain\_arr:
  - Description: Partition UV plane to 4x4 mesh with 25 points
  - Length: 25
  - Default: All 32 array (disabled)
  - Min: 0; Max: 63
  - Effect: Lower value stronger noise reduction
  - Calibration: None



## Step 4.1: Chorma-Based Noise Reduction Strength Adjustment (cont.)

- Parameter list
  - Define chroma region for adjustment with: `cnr_low_thrd_u` and `cnr_low_thrd_v`
  - Define interpolated region with: `cnr_low_gap_u` and `cnr_low_gap_v`
  - Adjust noise reduction strength with: `cnr_adj_gain`
- Blending weight of input luma channel

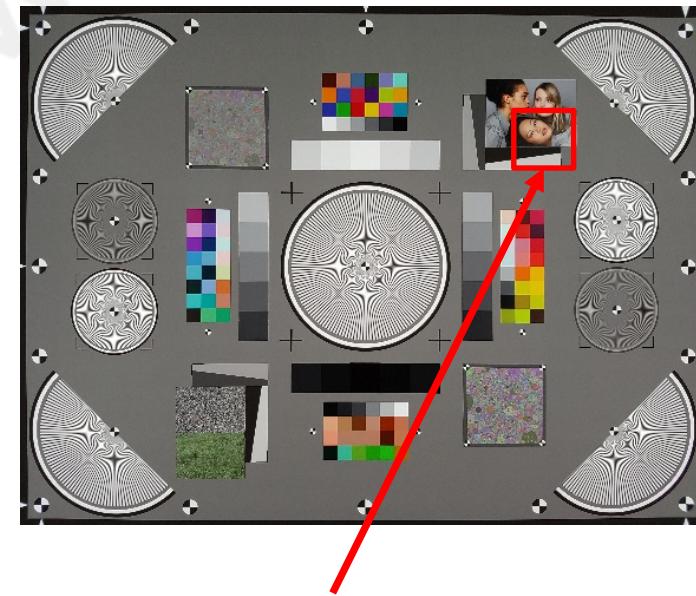
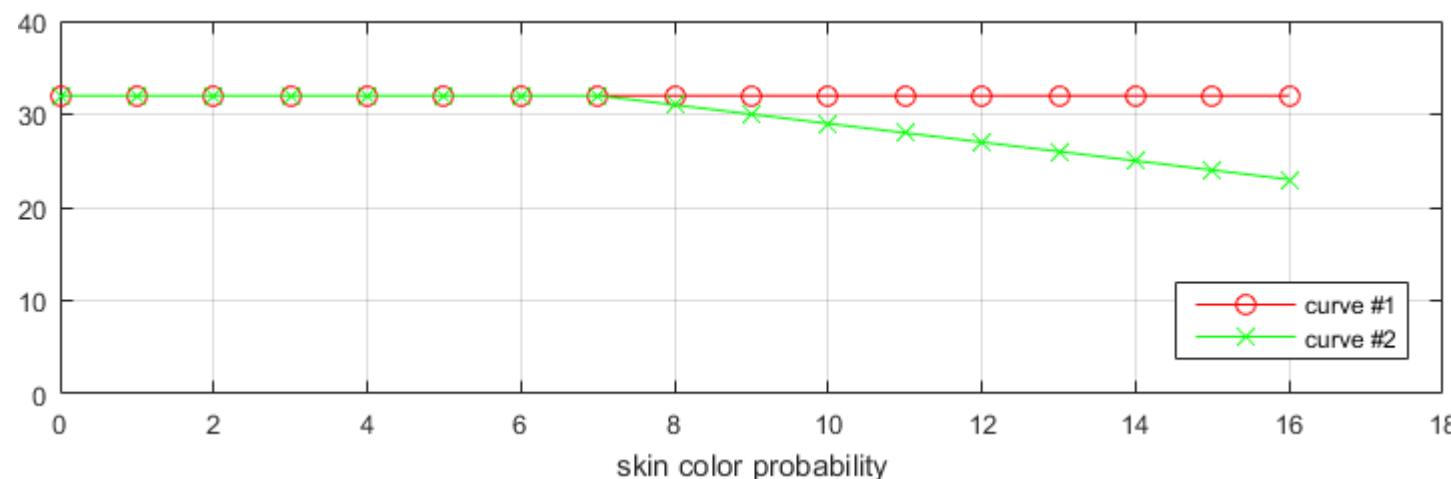
name	length	default	min	max	higher value	lower value
<code>cnr_low_thrd_u(v)</code>	1	2	-128	127	Impact more region	Impact less region
<code>cnr_low_gap_u(v)</code>	1	1	0	7	Larger interpolated region	Smaller interpolated region
<code>cnr_adj_gain</code>	1	32	0	63	Noisier output	Smoother output



## Step 4.2: Skin-Color Based Noise Reduction Strength Adjustment

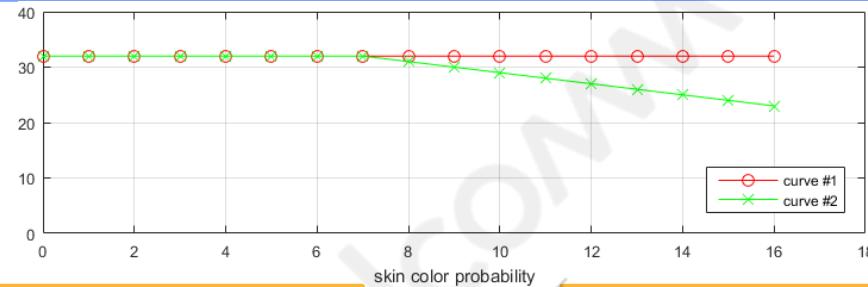
- **snr\_gain\_arr**
  - Description: Adjust noise reduction strength based on skin color
  - Length: 17
  - Default: All 32 array (disabled)
  - Min: 0; Max: 32
  - Effect: Lower value cleaner flat regions
  - Calibration: None
  - Suggestion: Using mild adjustment

Examples:



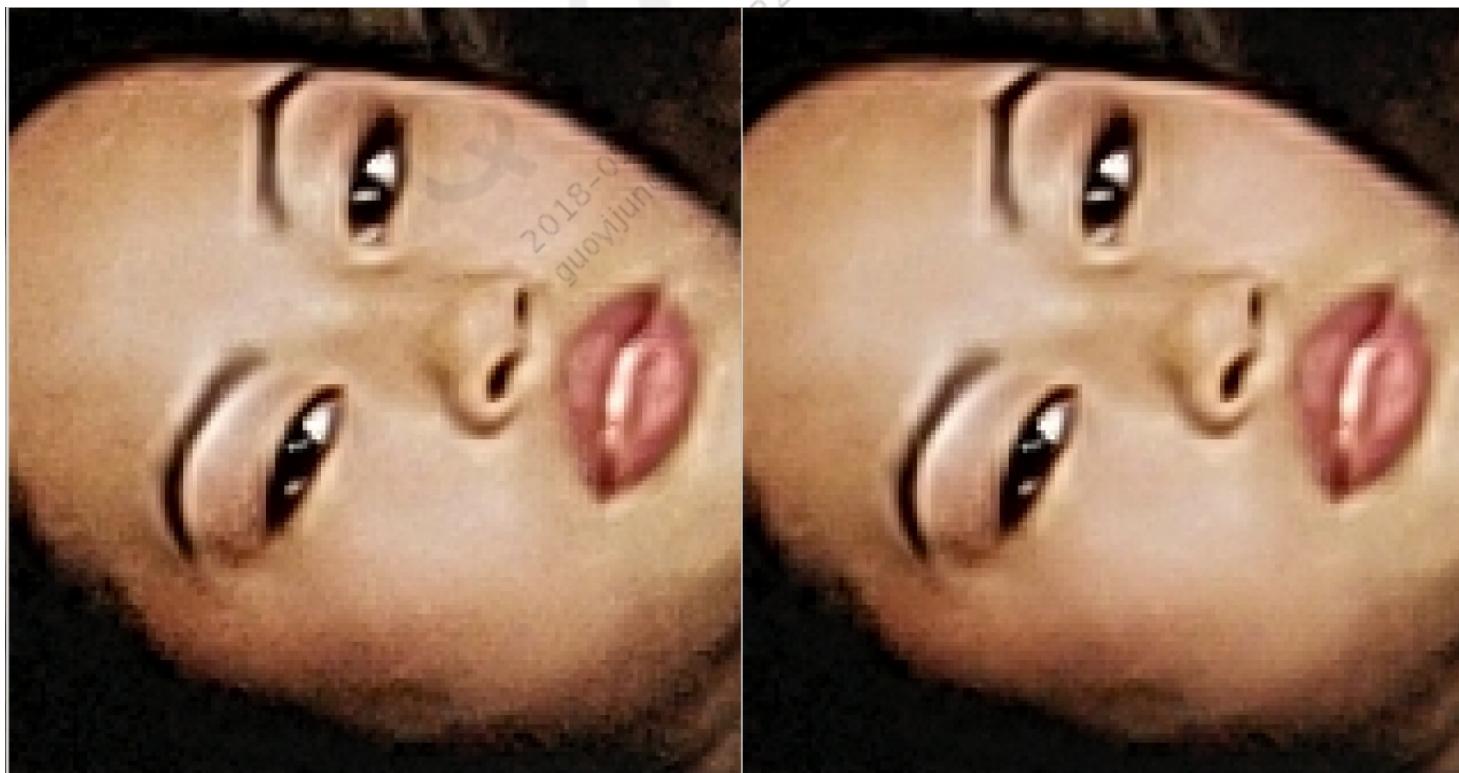
## Step 4.2: Adjust Noise Reduction Strength in Flat Regions

- `snr_gain_arr`



curve #1

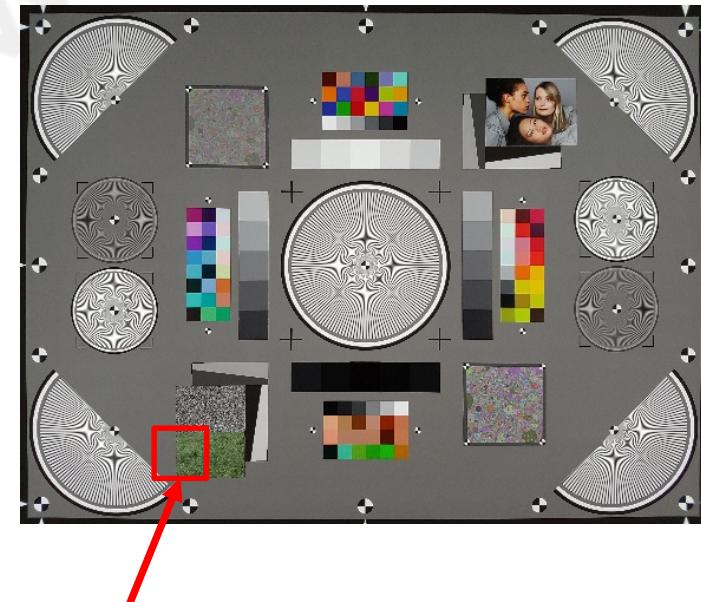
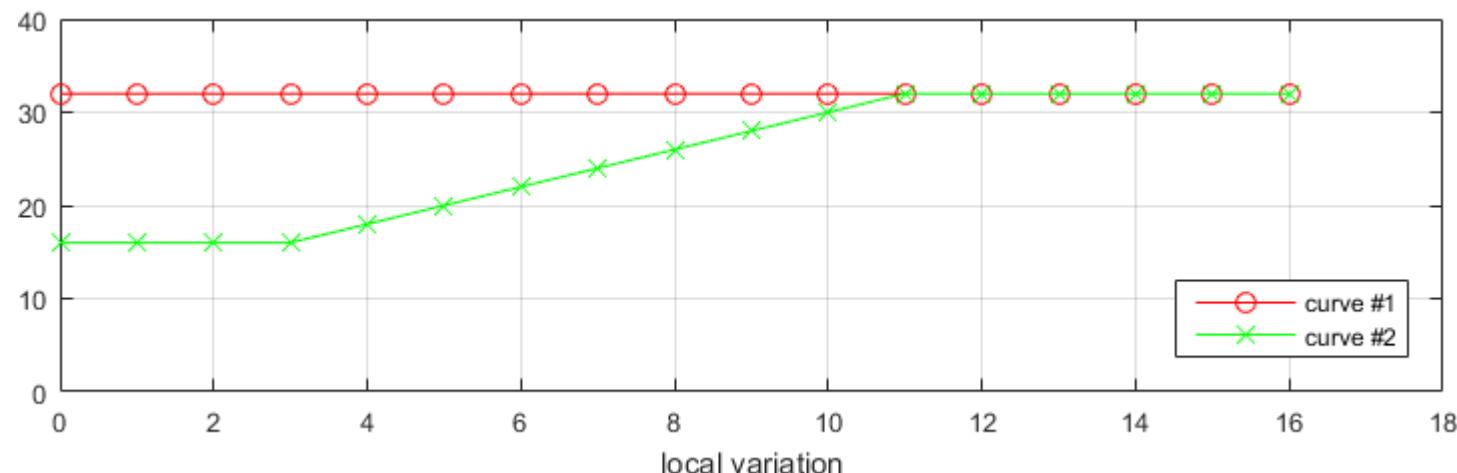
curve #2



## Step 4.2: Adjust Noise Reduction Strength in Flat Regions (cont.)

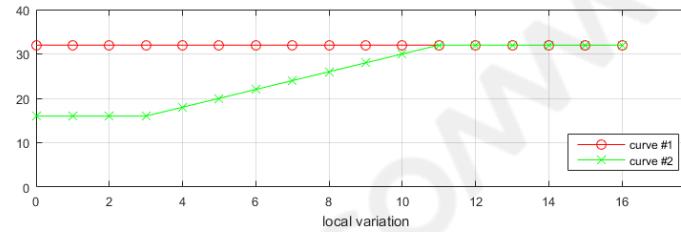
- **fnr\_gain\_arr**
  - Adjust noise reduction strength based on local variations.
  - Length: 17
  - Default: All 32 array (disabled)
  - Min: 0; Max: 63
  - Effect: Lower value cleaner flat regions
  - Calibration: None
  - Suggestion: Using mild adjustment

Examples:



## Step 4.2: Adjust Noise Reduction Strength in Flat Regions (cont.)

- fnr\_gain\_arr

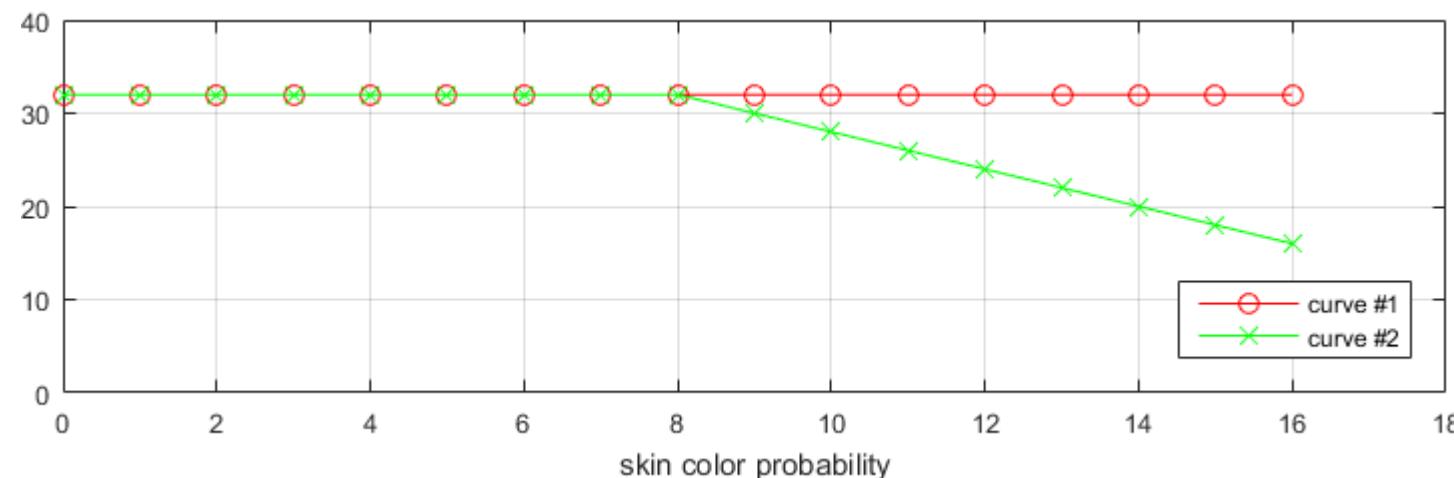


## Step 4.3: Skin-Color Based Blending Adjustment

- blend\_snr\_gain\_arr

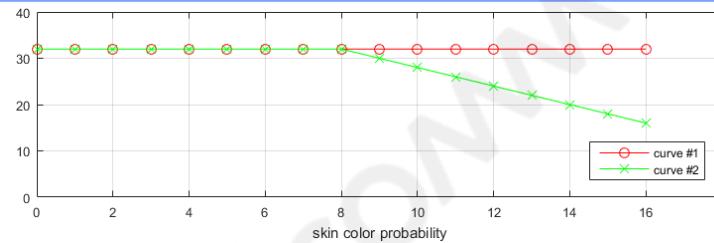
- Description: Blending weight of input luma channel
- Length: 17
- Default: All 32 array (disabled)
- Min: 0; Max: 32
- Effect: Lower value smoother skin region
- Calibration: None

Examples:



## Step 4.3: Skin-Color Based Blending Adjustment (cont.)

- blend\_snr\_gain\_arr



curve #1

curve #2

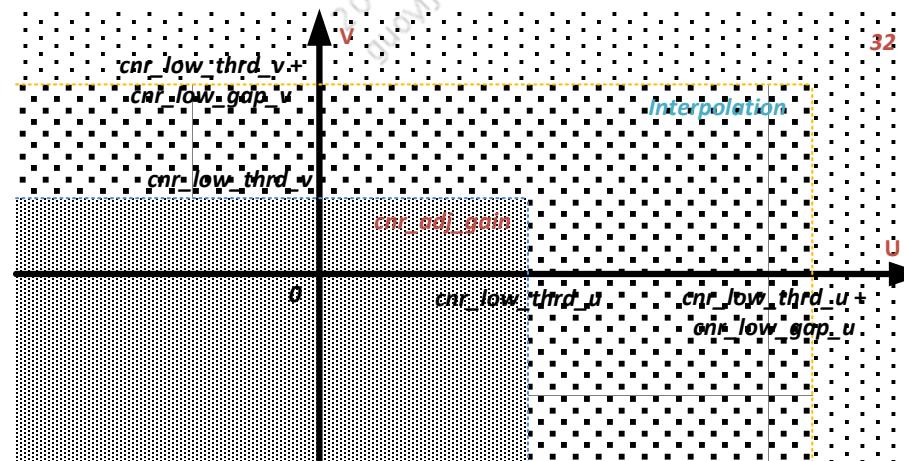


## Step 4.4: Chroma-Based Blending Adjustment

- blend\_cnr\_adj\_gain

- Blending weight of input luma channel based on defined regions
- Sharing the region definition with noise reduction stage
- Length: 1
- Default: 32
- Min: 0; Max: 63
- Effect: Lower value smoother output in defined chroma regions

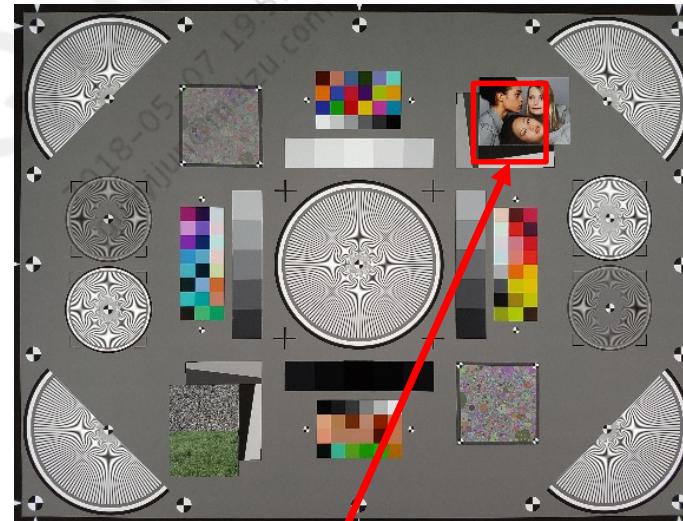
Examples:



## Step 4.5: Reducing Jaggy Artifacts Along Edges

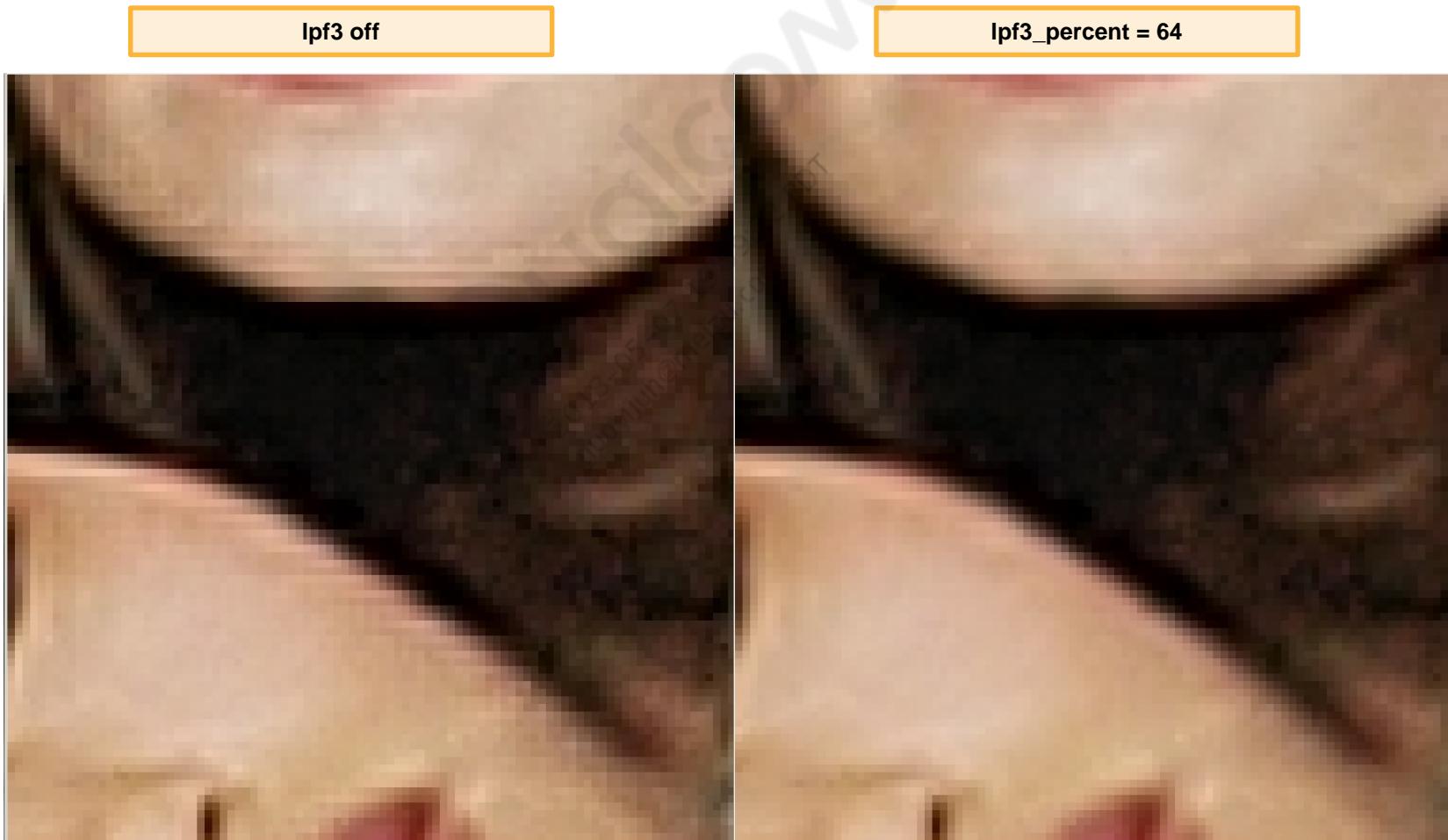
- **lpf3\_percent**
  - Controls threshold for smoothing
  - Length: 1
  - Default: 16
  - Min: 0; Max: 255
  - Effect: Higher value smoother edges
  - Calibration: None

Examples:



## Step 4.5: Reducing Jaggy Artifacts Along Edges (cont.)

- `lpf3_percent`



# Step 5.1: Adjust Skin Color Definition

---

- Parameter list for adjusting skin color
  - skin\_hue\_min
  - skin\_hue\_max
  - skin\_y\_min
  - skin\_y\_max
  - skin\_saturation\_min\_y\_max
  - skin\_saturation\_max\_y\_max
  - skin\_saturation\_min\_y\_min
  - skin\_saturation\_max\_y\_min
  - skin\_boundary\_probability
  - skin\_percent
  - skin\_non\_skin\_to\_skin\_q\_ratio

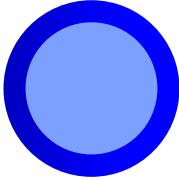
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

## Step 5.2: Facial Areas Adjustment

---

- **face\_boundary**
  - Lower bond factor for facial areas detected by face detection
  - Length: 1
  - Default: 1.2
  - Min: 0; Max: 8
  - Effect: Lower value smaller facial areas
- **face\_transition**
  - Lower bond factor for facial areas detected by face detection
  - Length: 1
  - Default: 2.0
  - Min: 0; Max: 8
  - Effect: Larger value larger transition areas
- **face\_boundary ≤ face\_transition**

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com



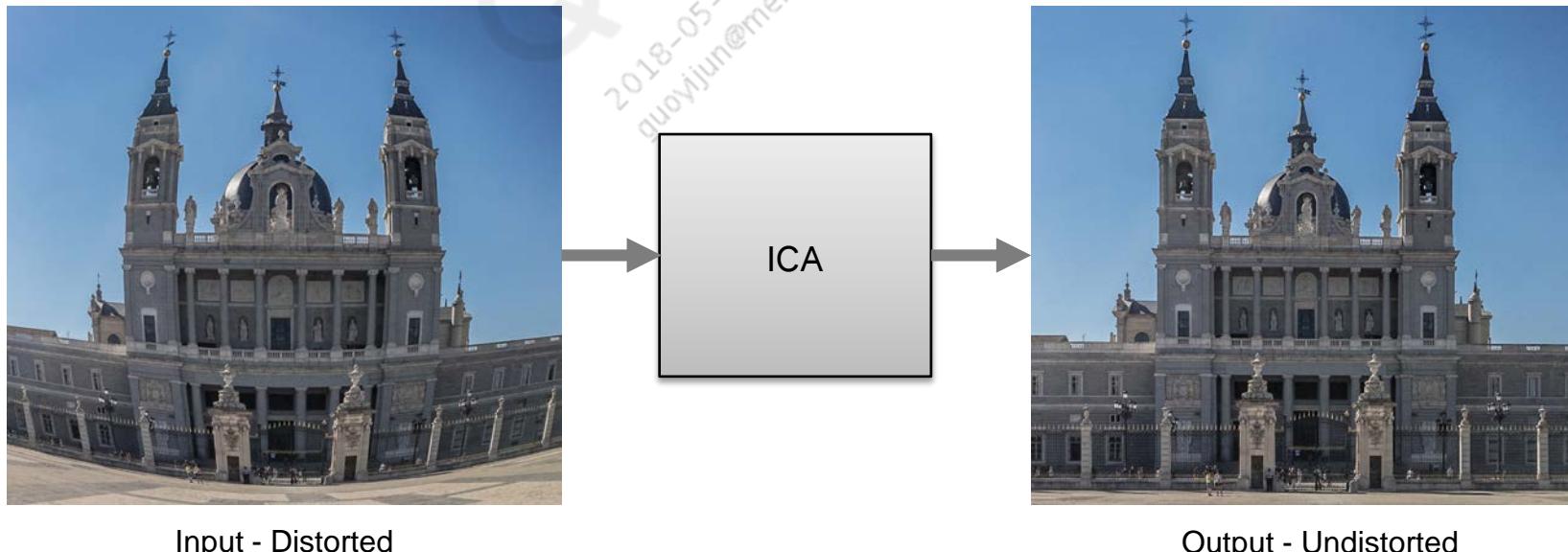
# Image Correction and Adjustment (ICA)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Image Correction and Adjustment (ICA) – Introduction

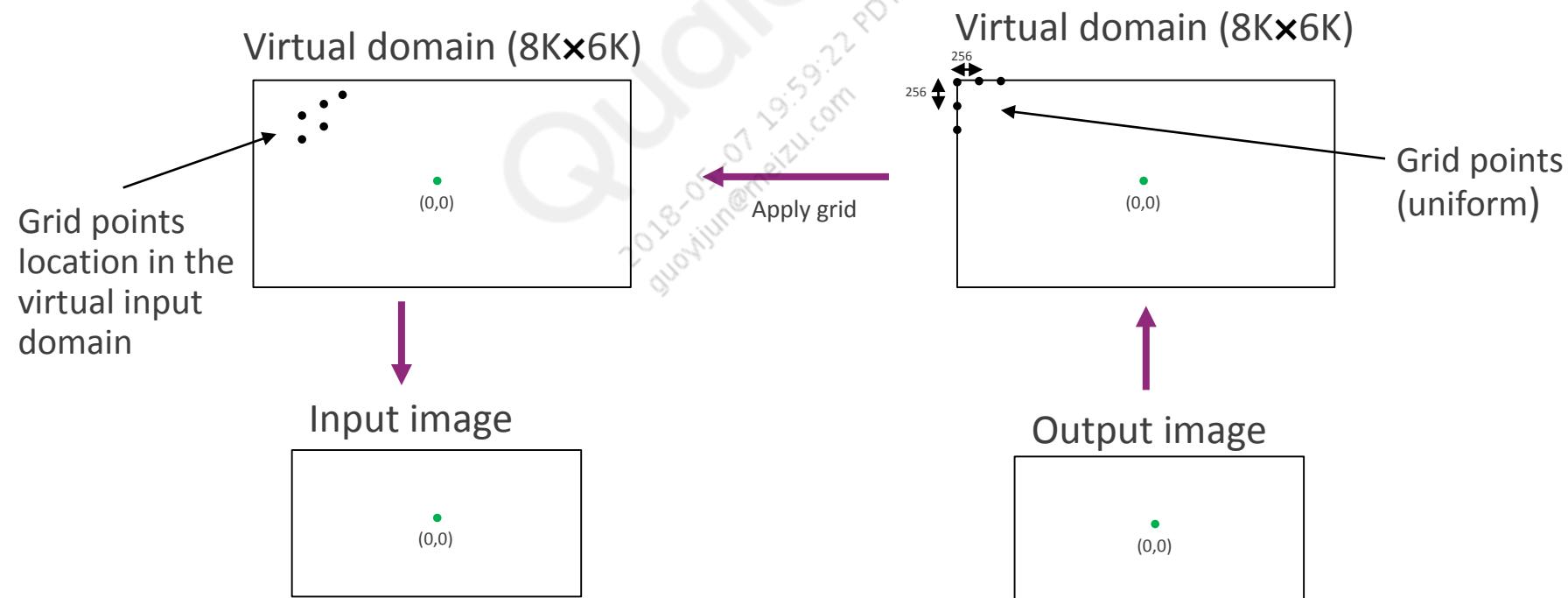
## Problem statement

- Image Correction and Adjustment (ICA) module is a hardware unit which is mainly used for correction of geometric distortions caused by camera lens and movement
- The ICA HW unit should act as a low-power real-time warping engine inline to IPE. Typical use-cases are Lens distortion correction (LDC) and Electronic image stabilization (EIS)
- It should also support image alignment in order to allow Motion compensated temporal filter (MCTF)



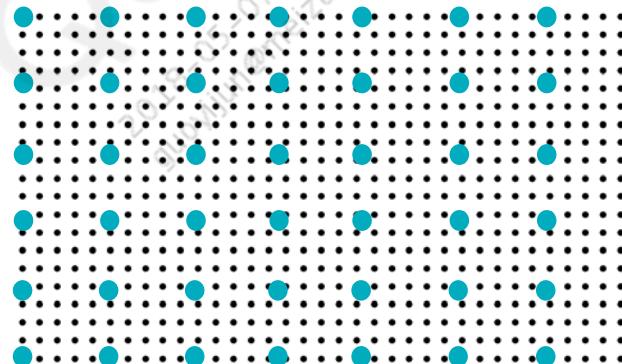
# Step 1: Grid Transform

- Grid samples do not depend on the resolution
  - Received and applied in virtual domain ( $8K \times 6K$ )
  - Origins (0,0) are at the center



## Step 1: Grid Transform (cont.)

- Grid transform can be used to apply a general transformation
- Essentially a warp map and each sample contains explicit mapping of output to input coordinate, both are in the virtual domain
  - The virtual domain is defined by 17 bits for X and 17 bits for Y
- It consists of a sparse uniform grid (33 by 25 samples) over the output image:
  - Intermediate coordinates calculated by the hardware using bi-cubic interpolation:



- Enable via CTC\_TRANSFORM\_GRID\_ENABLE parameter
  - (CTC : Coordinate transform calculator)

# Output Pixel Interpolation

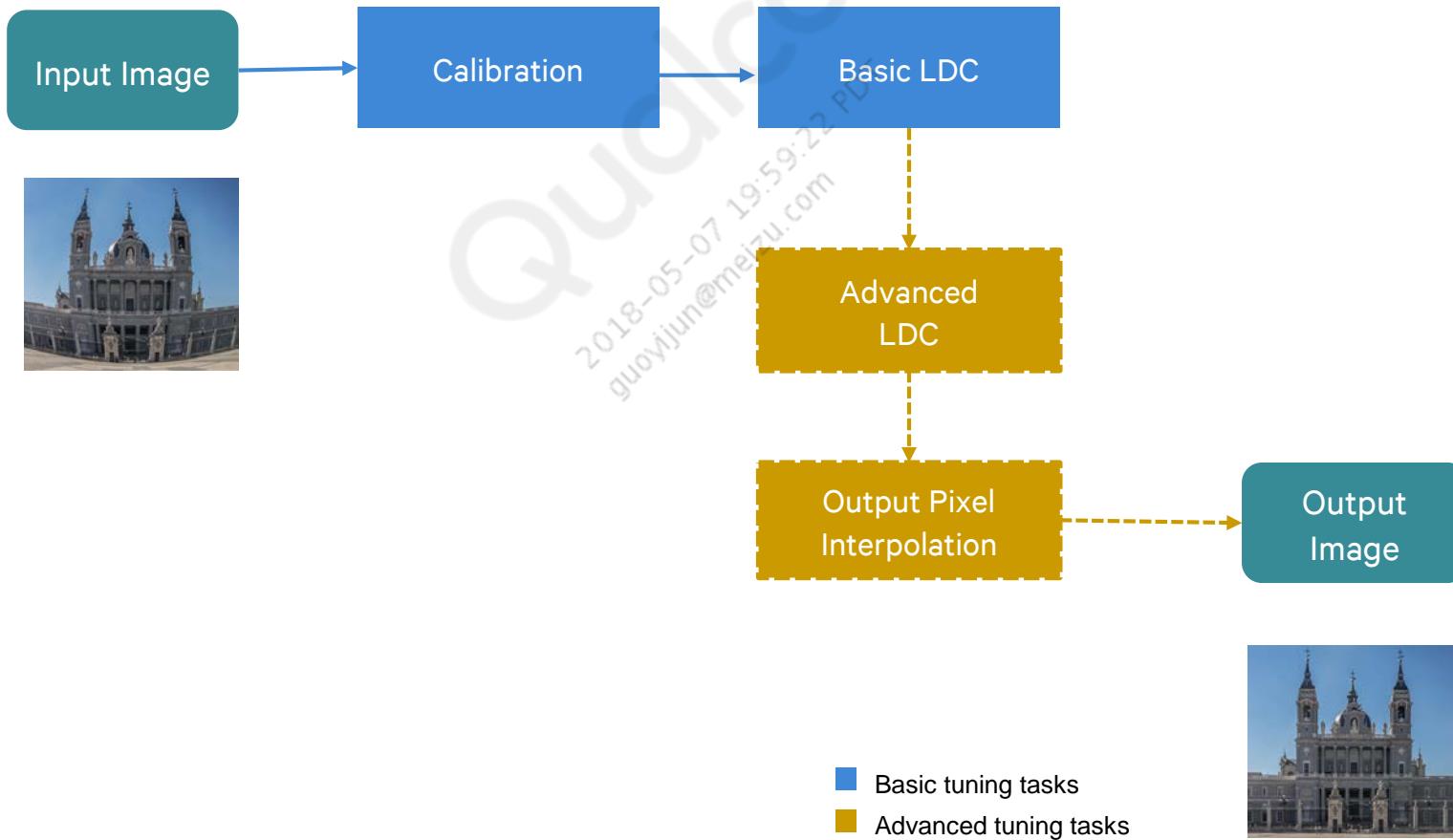
---

- Output pixels are generated via interpolation
  - Luma – Two interpolation options
    - Configurable 4x4 symmetric kernel
      - Defined by setting Y\_INTERPOLATION\_TYPE parameter to 0
      - Interpolation kernel is defined by OPG LUT (symmetric with 32 phases)
    - Native bi-linear interpolation
      - Defined by setting Y\_INTERPOLATION\_TYPE parameter to 1
  - Chroma
    - Bi-linear interpolation

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

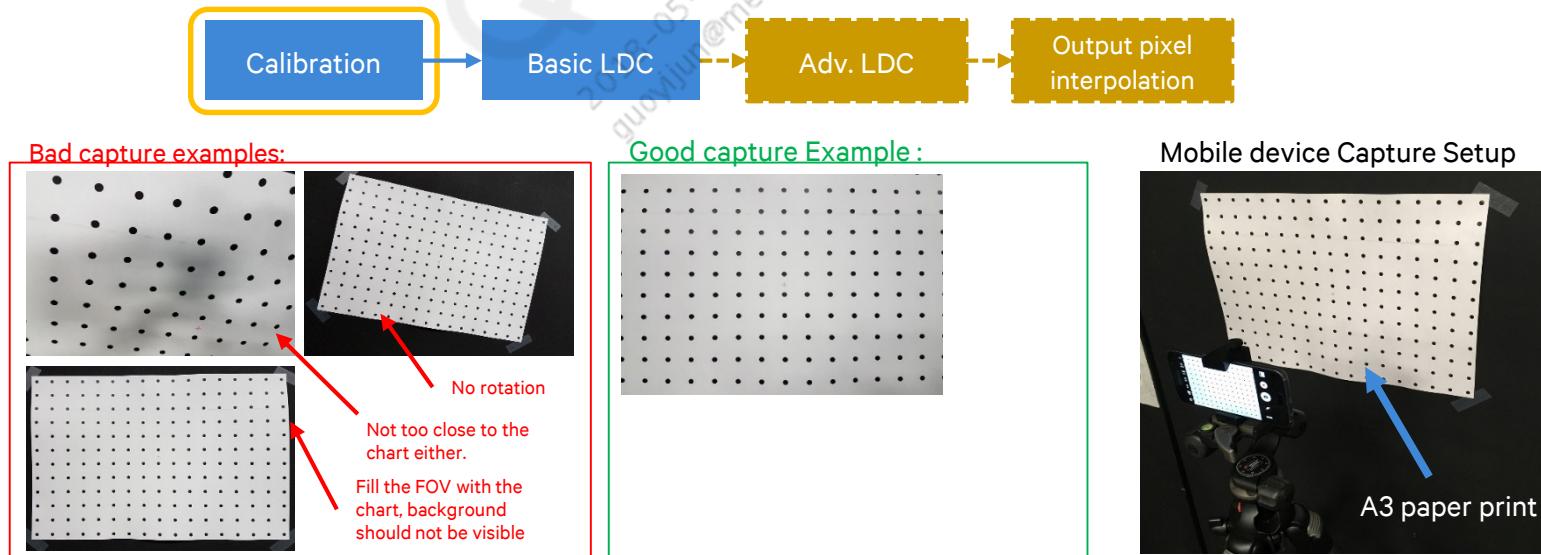
## Step 2: LDC Tuning Flow Diagram

- The following diagram describes the LDC tuning flow:



## Step 3.1.1: Calibration

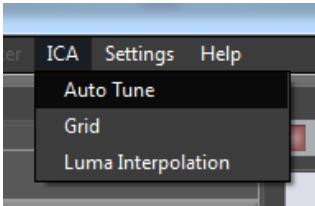
- For LDC calibration, a dot chart is required
  - Capture requirement
    - Light can be D65, TL84, D50 and should be around 400 lux
    - Using a tripod, the camera position should be parallel and aligned (not rotated) with the chart
    - Set camera to use minimum gain (to reduce noise) and set exposure so that there will be no flicker from light source
    - Chart should cover the entire FOV or little more: Image should include only the chart
    - Make sure that the captured image has no motion blur and AF focused



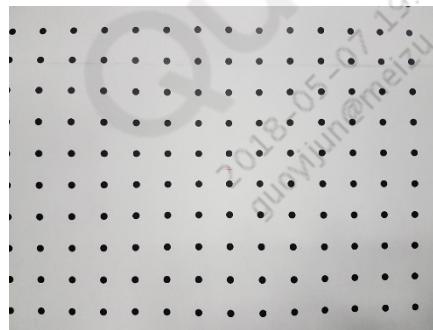
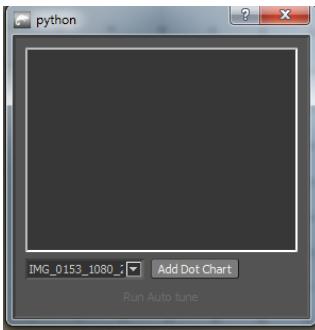
\* LDC tuning should be done per every camera, i.e. different tuning for front /rare, or between tele and wide...

## Step 3.1.2: Calibration

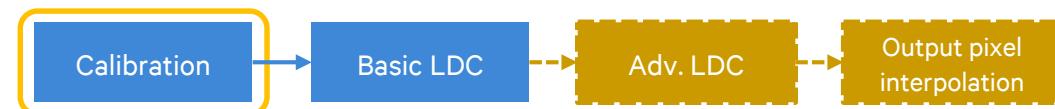
- Use Auto Tune option from the ICA menu



- Select the dot chart image captured

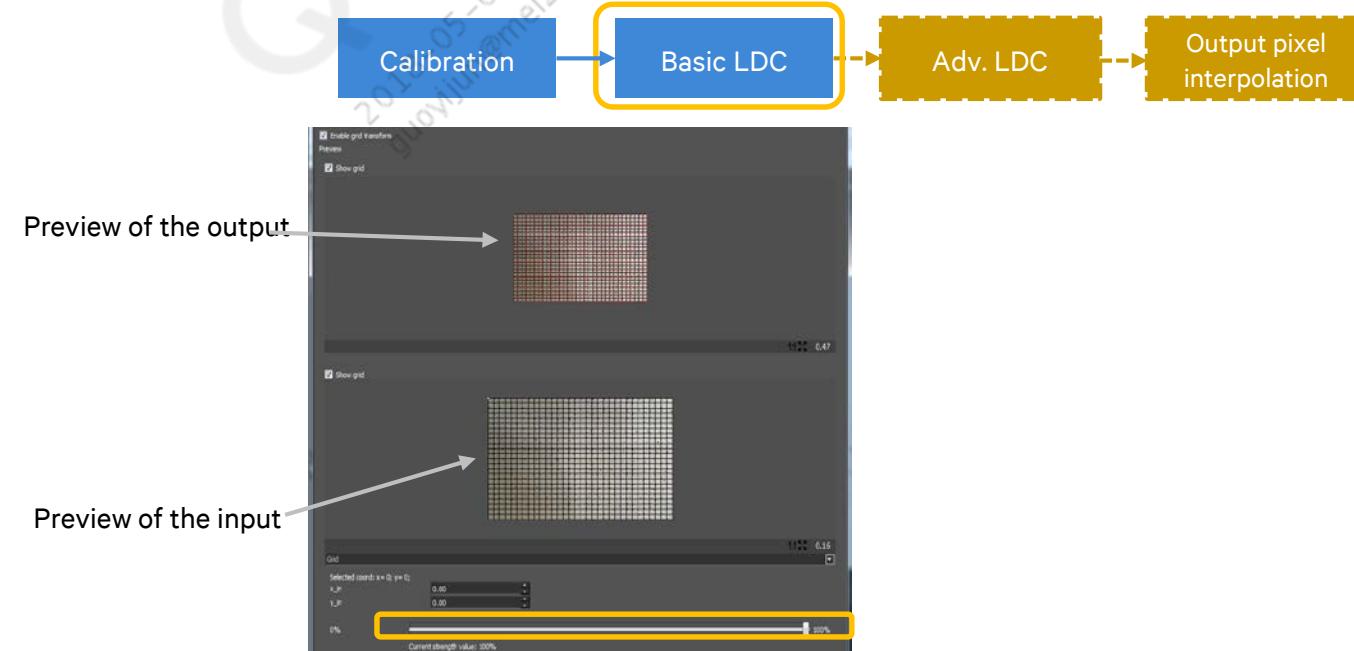


- Run Auto Tune to calibrate automatically



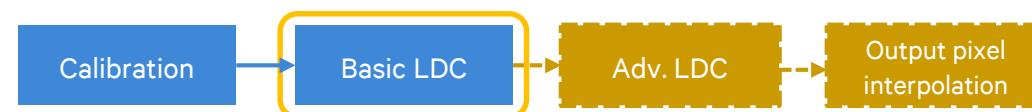
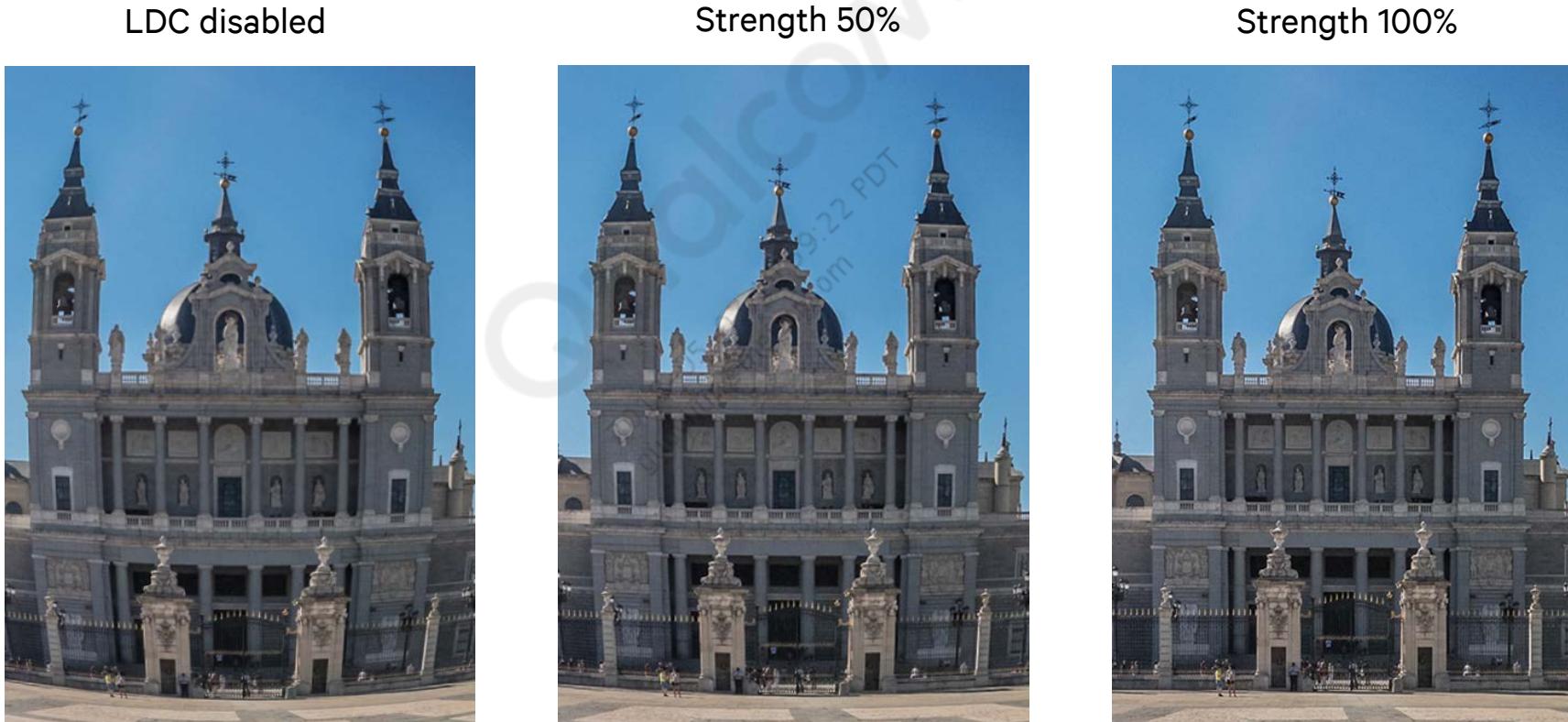
## Step 3.2.1: Basic LDC

- Correction Strength
  - Background
    - ICA LDC calibration corrects the image to full undistorted reference with 100% strength
    - A tuning engineer may choose to keep some of the distortion and lower the **Correction Strength**
  - Tuning Flow
    - Select Grid from the ICA menu
    - Use the **Correction Strength** slider (yellow box below) to adjust the strength of the effect
    - Access the impact from the preview of the output, then process and examine impact on an image



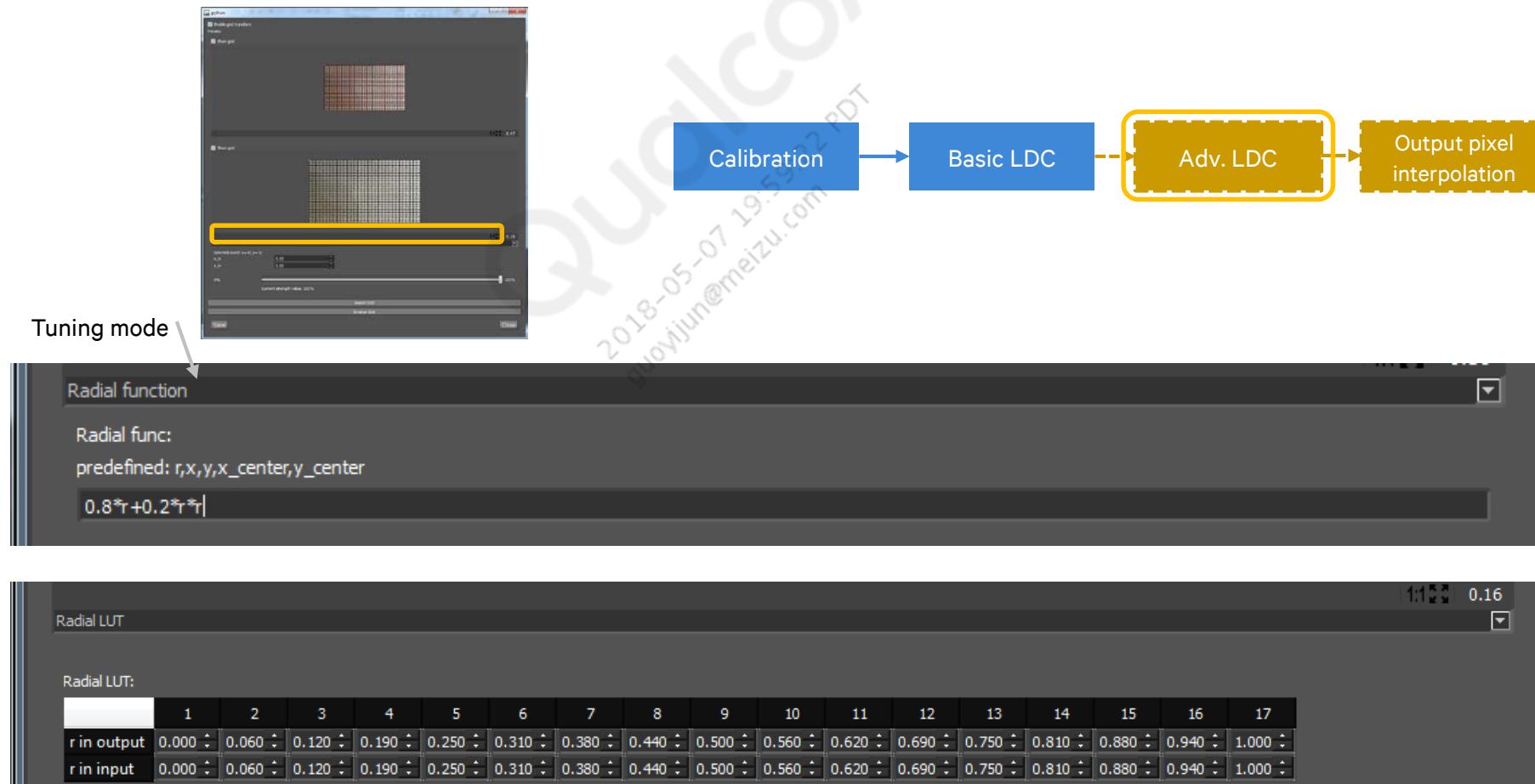
## Step 3.2.2: Basic LDC

- The effect of **Correction Strength**



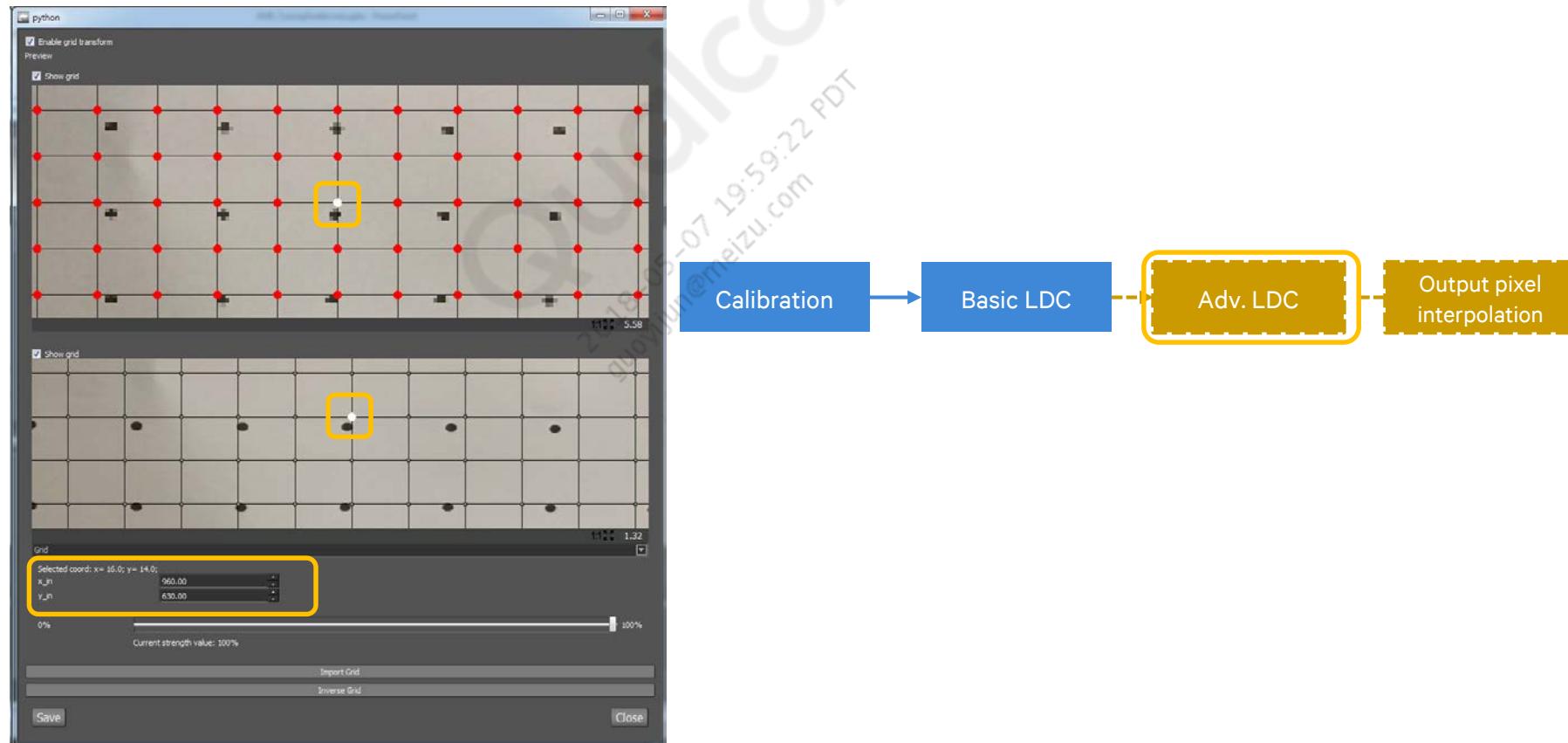
## Step 3.3.1: Advanced LDC

If a radial correction function or LUT are available, they can be used as well by changing the tuning mode



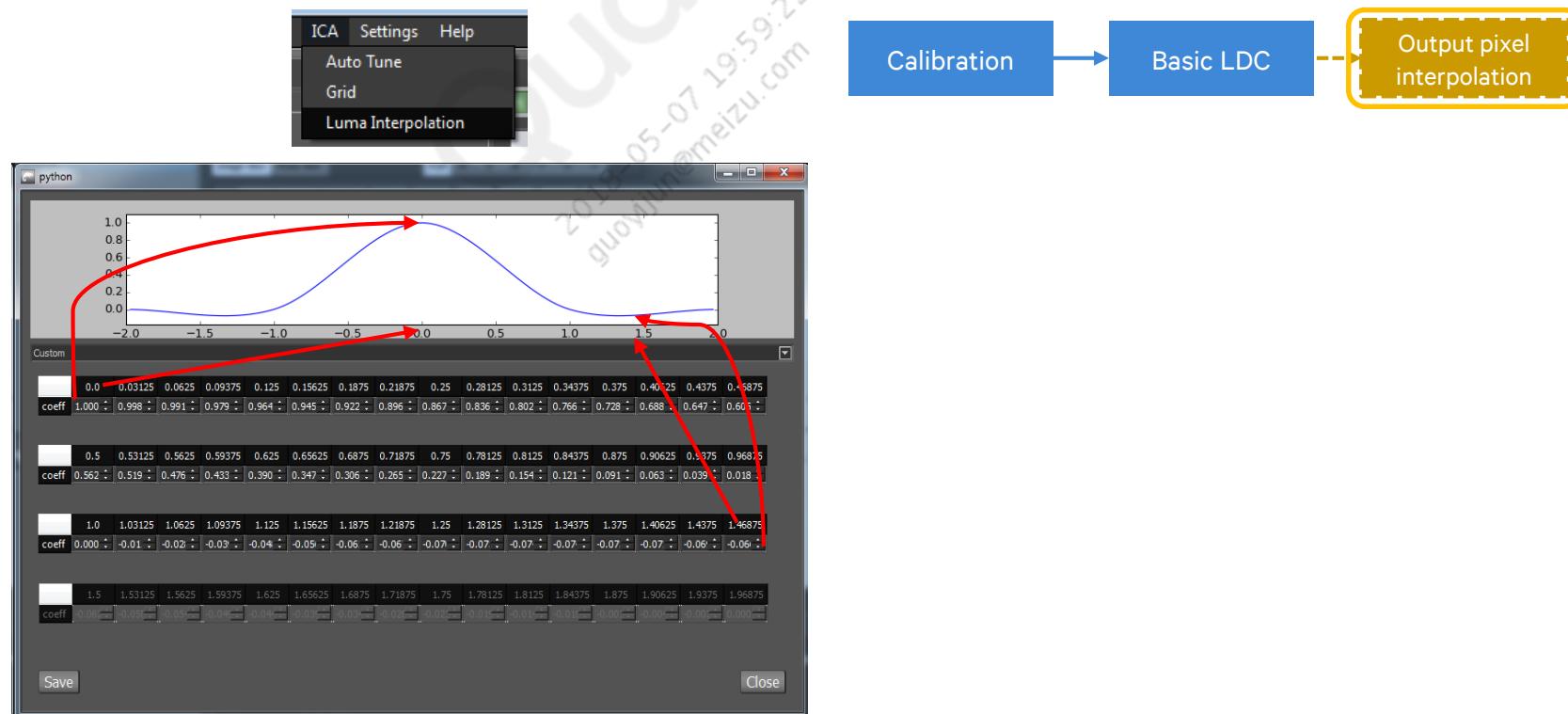
## Step 3.3.2: Advanced LDC

- Each grid sample also can be adjusted individually using the Grid Tuning mode by selecting a grid point and changing its mapped coordinates



## Step 3.4.1: Output Pixel Interpolation

- Output pixel interpolation can be used to compensate some resolution degrade after warping engine processing
  - Recommend not to enable at the beginning of projects since it affects overall sharpness and luma noise level
- The pixel interpolation kernel can be customized by selecting **Luma Interpolation** from ICA menu
- The kernel can be defined directly in Custom mode (bi-cubic or bilinear)

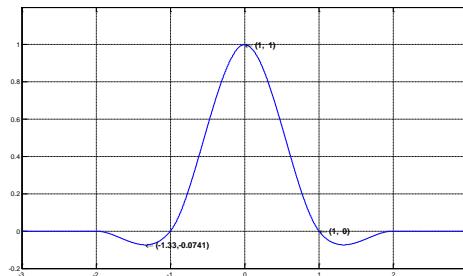


## Step 3.4.1: Output Pixel Interpolation (cont.)

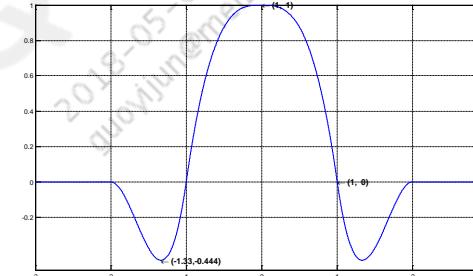
- $4 \times 4$  Kernel example
  - Consider the following formula for kernel selection with two parameters B and C

$$k(x) = \frac{1}{6} \begin{cases} (12 - 9B - 6C)|x|^3 + (-18 + 12B + 6C)|x|^2 + (6 - 2B) & |x| < 1 \\ (-B - 6C)|x|^3 + (6B + 30C)|x|^2 + (-12B - 48C)|x| + (8B + 24C) & 1 \leq |x| < 2 \\ 0 & otherwise \end{cases}$$

- $B = 0, C = 0.5$  (Bi-cubic)



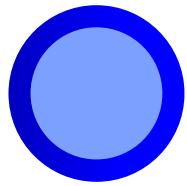
—  $B = 0, C = 3$



## Step 4: How to Coordinate with Other Modules

---

- ICA LDC should be tuned first in IPE
- Sharpening should be done at a later stage. It is not recommended to use ICA Luma Interpolation function for sharpening, instead it is suggested to use ASF
  - Snapshot case, use of the Luma Interpolation function would break current luma noise level which is already tuned
  - Preview and video case, use of the Luma Interpolation function would affect inputs of multi frame temporal filtering block

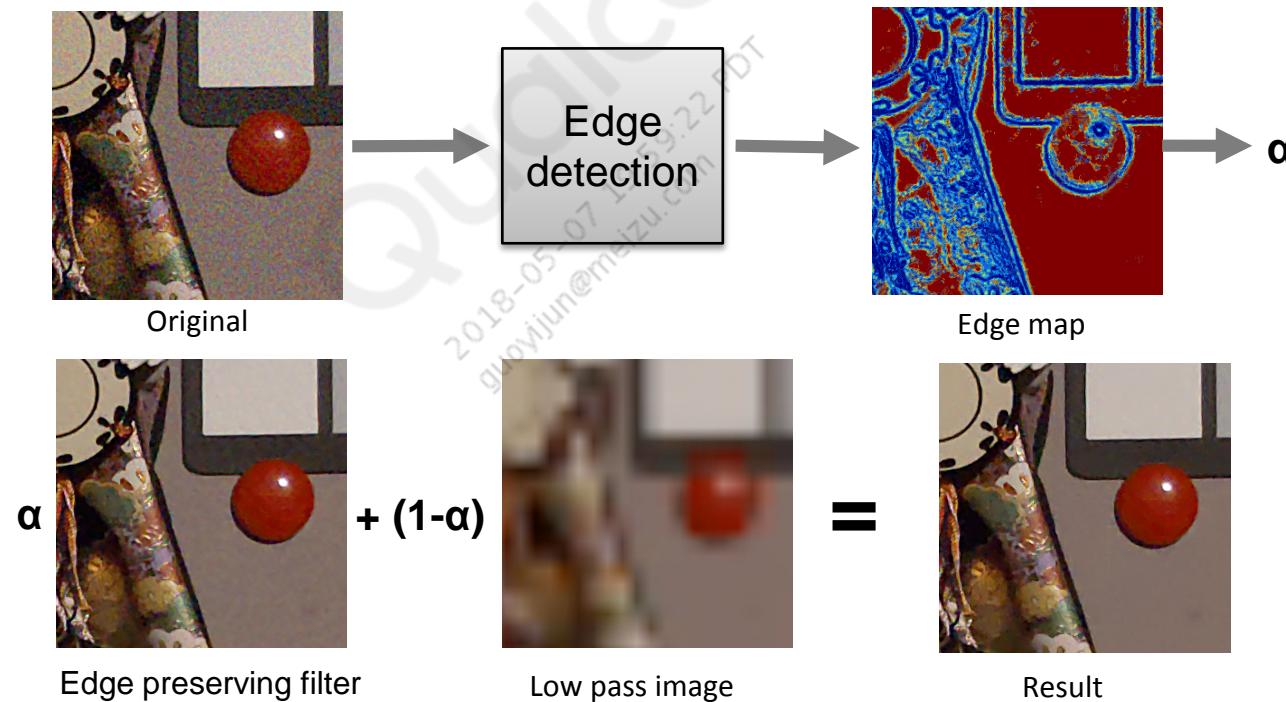


# Advanced Noise Reduction (ANR)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Advanced Noise Reduction – Background

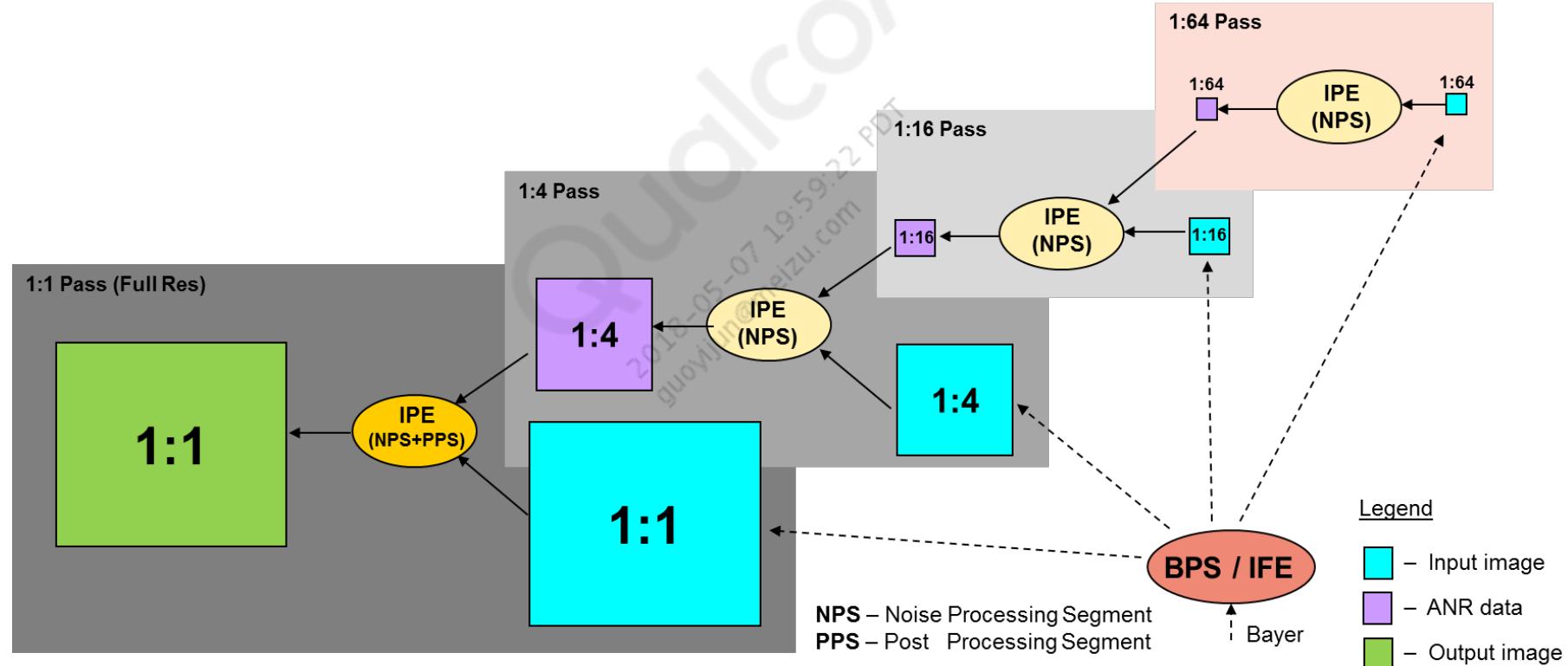
- Advanced noise filtering
- Multi pass spatial noise filtering
- Located in NPS of IPE, which means it is available for both video and snapshot



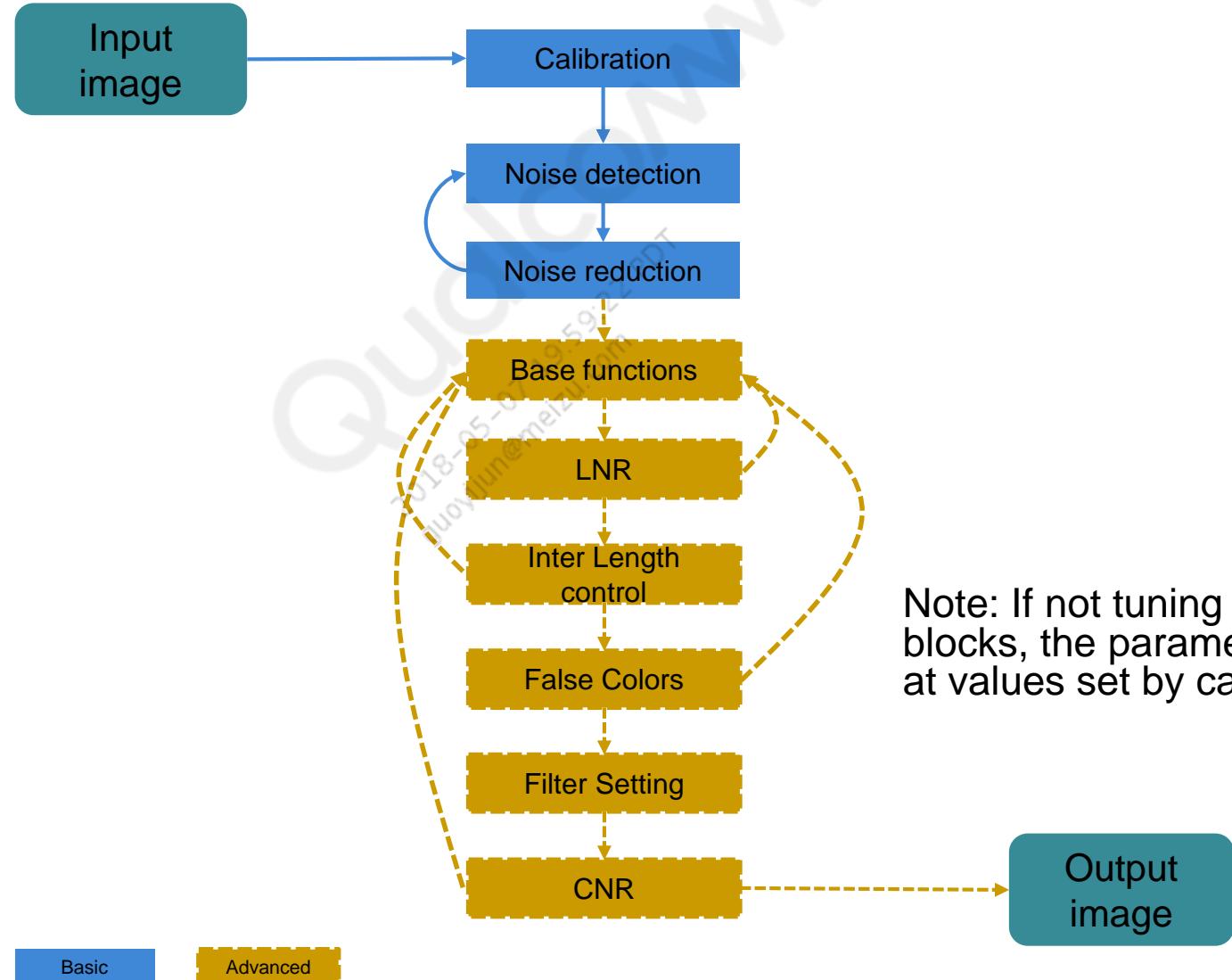
- $\alpha$  is determined for each pixel according to the edge map

# Advanced Noise Reduction – Overview

- General processing flow



# Tuning Flow Diagram



Note: If not tuning advanced blocks, the parameters remain at values set by calibration

# Calibration – Steps

1. Select 1 flat field image and at least 1 MCC.

Load MCC Images

Load Flat Image

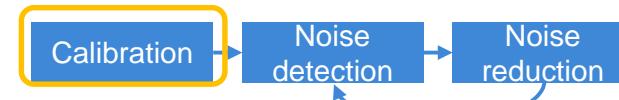
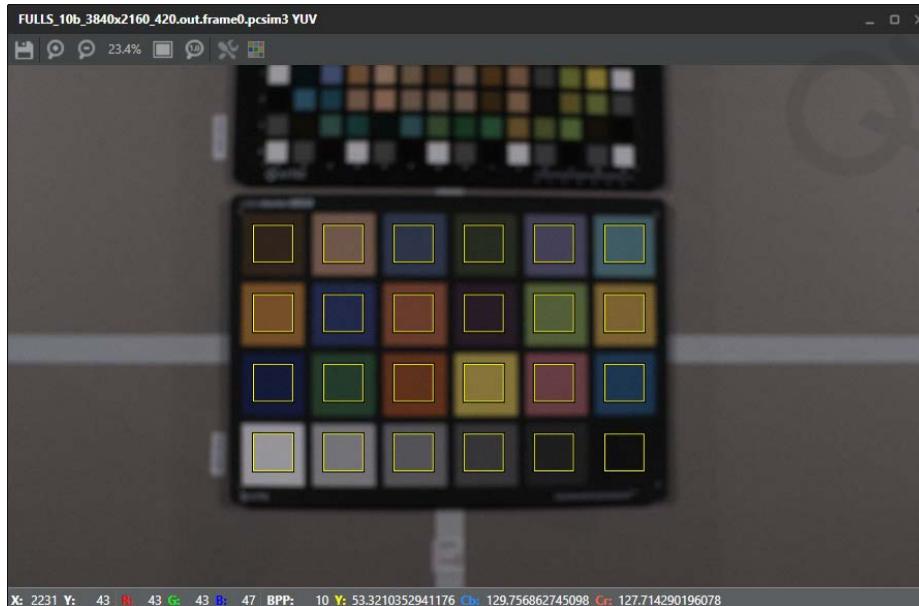
Make sure that the MCC patches are properly selected and covers ~80% of each patch and is center aligned. If not, correct it by dragging them to fit.

2. Click **Calibrate** and **Calibrate LNR**.

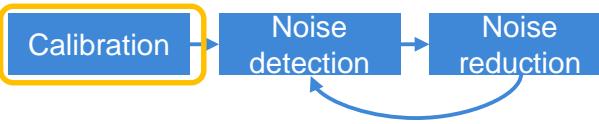
Calibrate

Calibrate LNR

3. Examine the result on a natural scene image.



# Calibration Troubleshooting



- Example of a good calibration



- Example of a bad calibration

- In most of the cases, bad MCC(s) input will result in over filtration

Damaged MCC

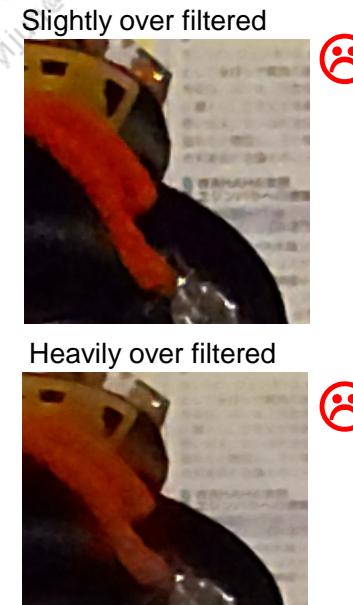


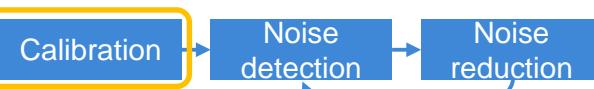
Too small patches

Or



Bad patches selection





## Calibration Troubleshooting (cont.)

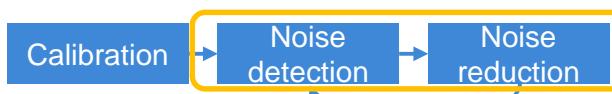
- Inaccurate flat field image may result radial noise patches and heavily effect calibration noise detection, especially in low frequencies

Calibration result using accurate flat field image



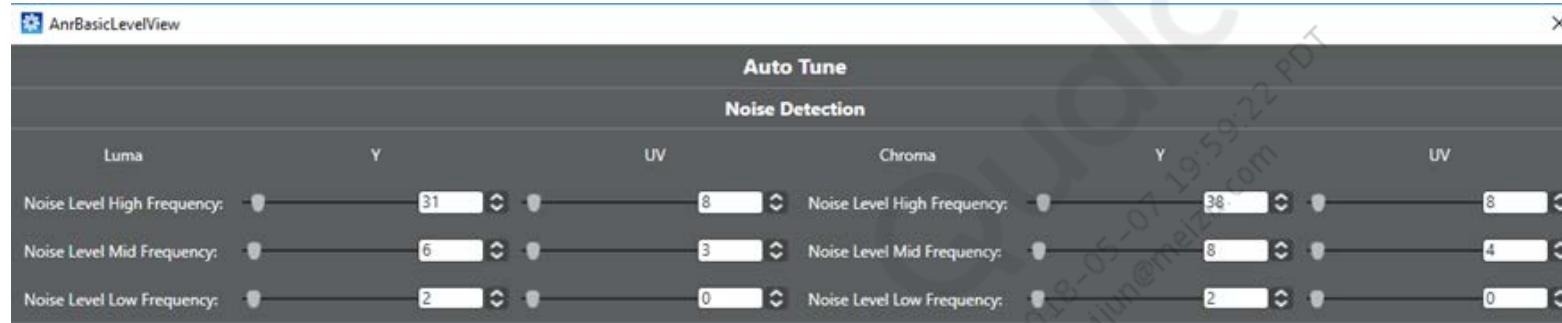
Calibration result using defected flat field image



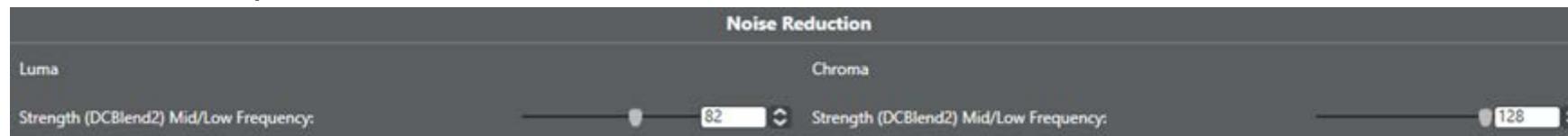


# Tuning Steps – Basic Level

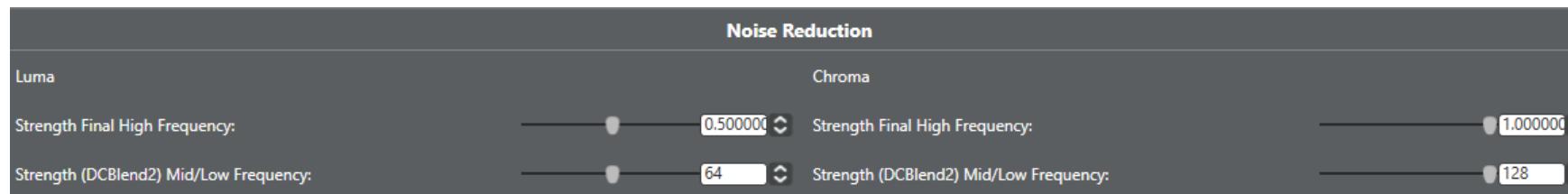
- Tune on a *natural* scene image
- Applicable only after calibration
- After each change, click **Save** and then click **Process**

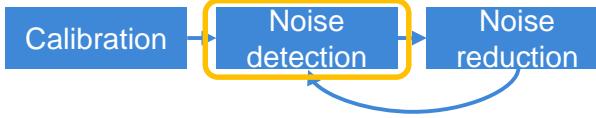


For video and preview modes



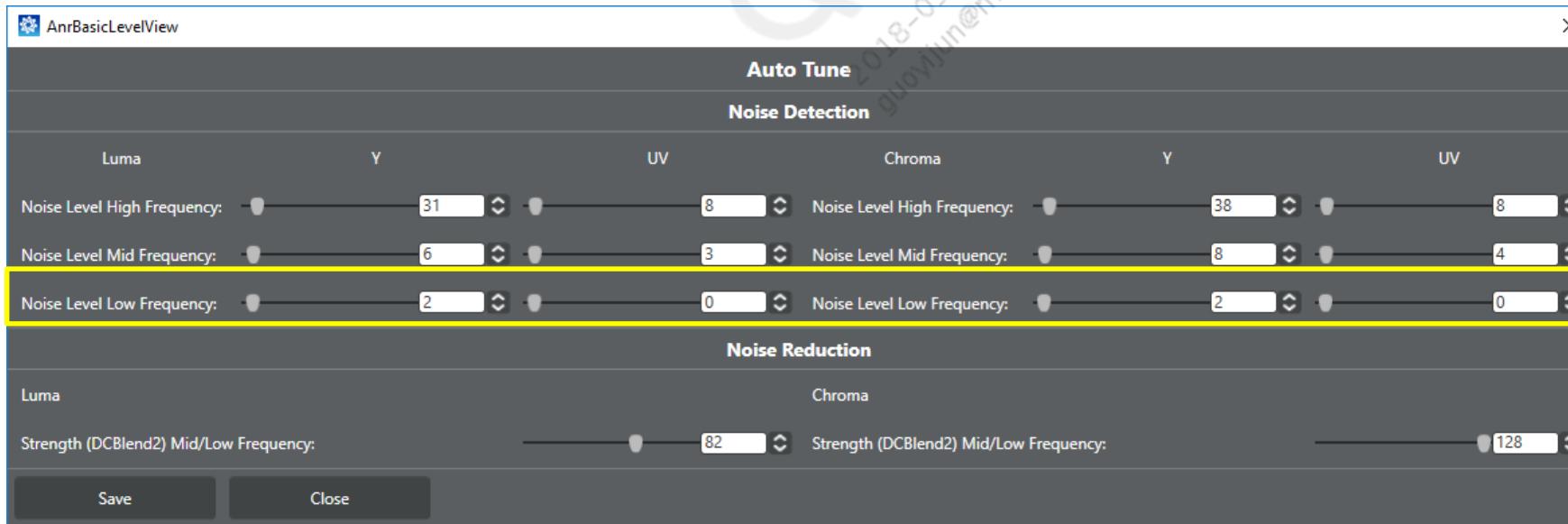
For snapshot mode

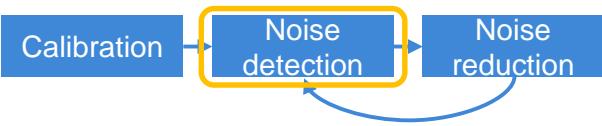




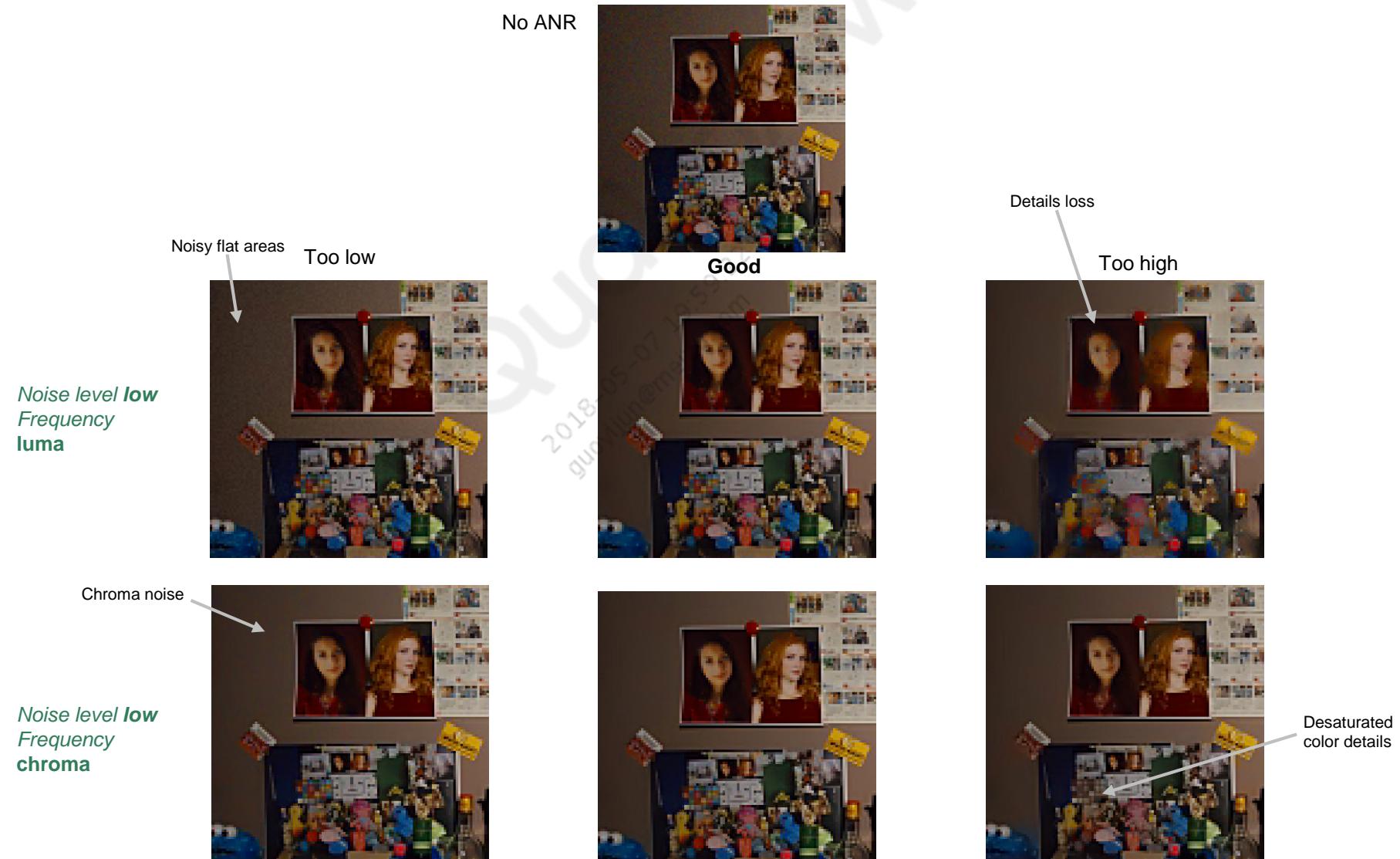
# Tuning Steps – Basic Level (cont.)

- Noise detection
  - Separates details from noise at each frequency range
    - Higher value (more details are interpreted as noise) = *eaten* edges, more filtering
    - Lower value (more noise is interpreted as details) = Noisier images, noise patches
  - Flow – 1<sup>st</sup> step
    - Select DC16 image in the viewer
    - Adjust if required, **Noise Level Low Frequency** (Luma and Chroma) to achieve a **noise-free** edge preserved DC16 image





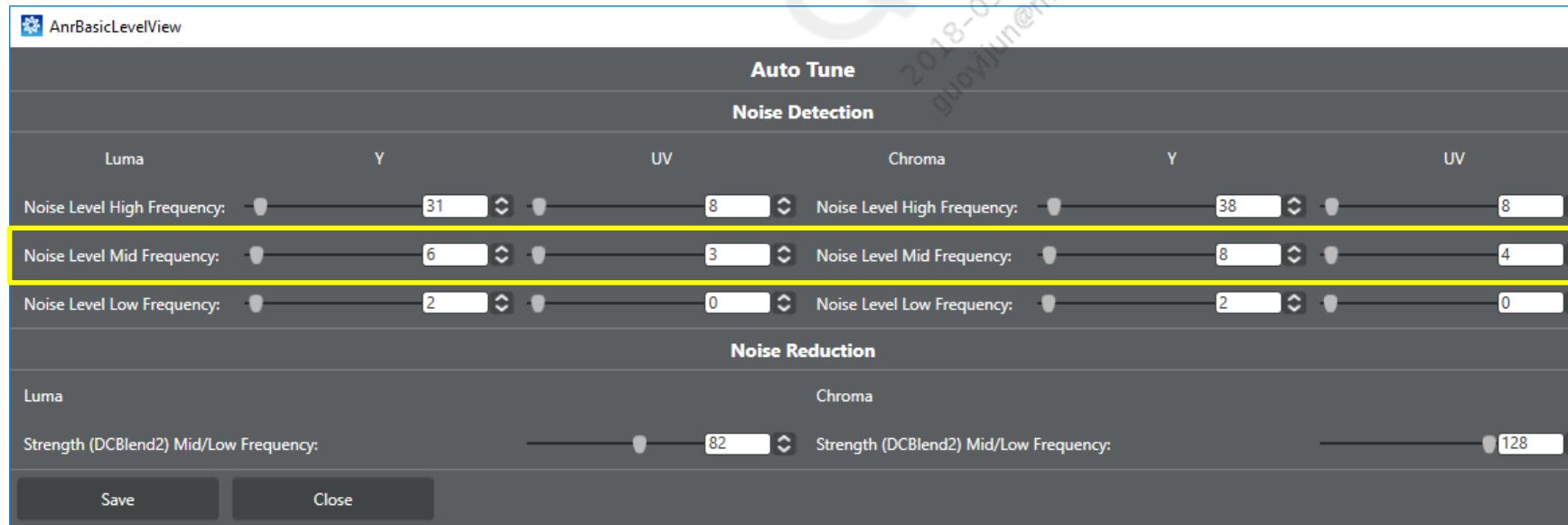
# Tuning Steps – Basic Level (cont.)



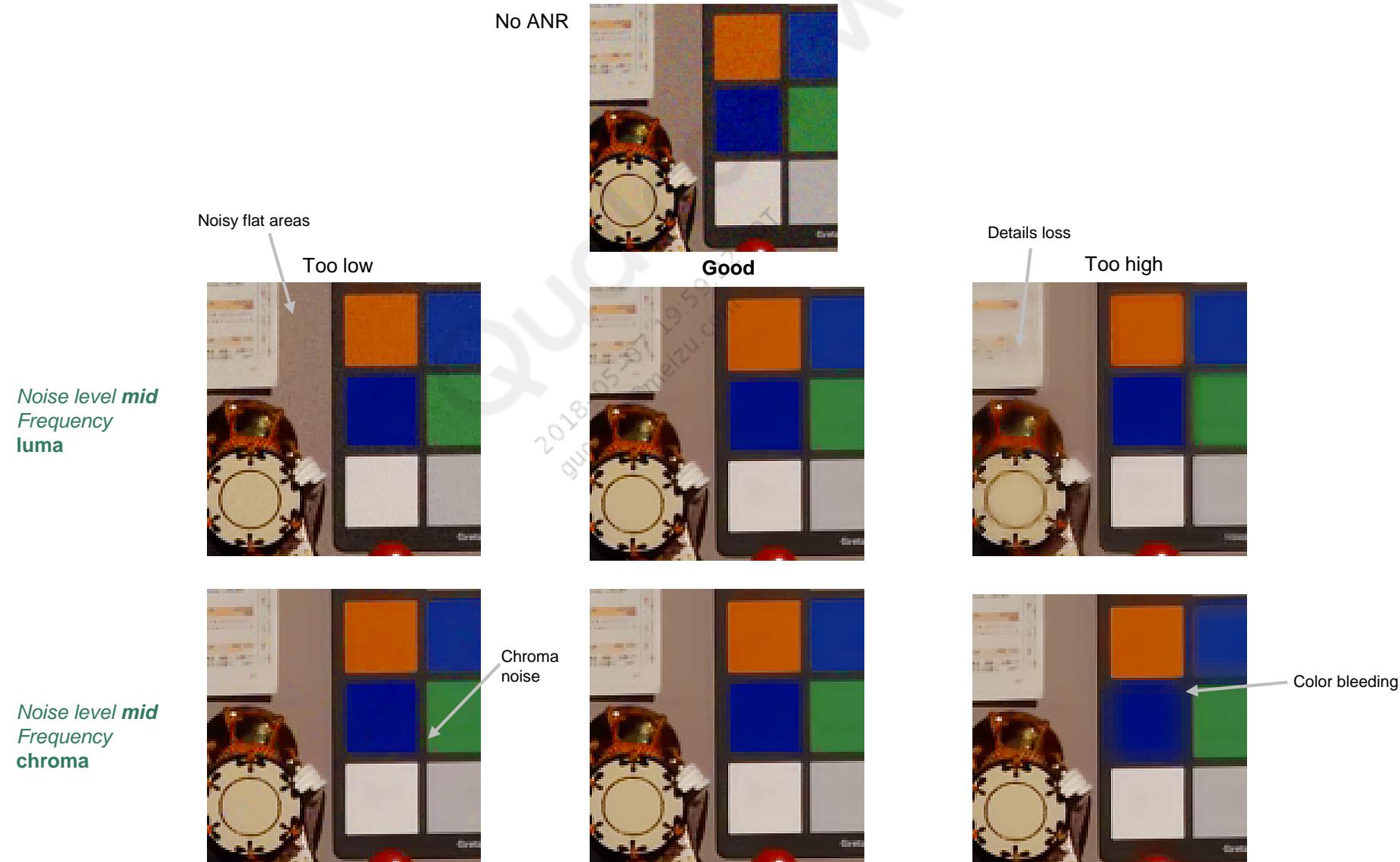
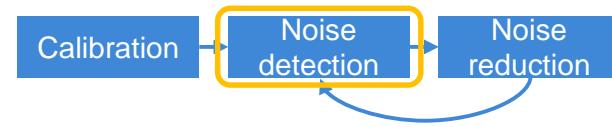
# Tuning Steps – Basic Level (cont.)

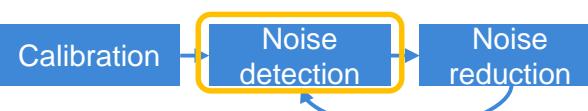
- Noise detection

- Separates details from noise at each frequency range
    - Higher value (more details are interpreted as noise) = *eaten* edges, more filtering
    - Lower value (more noise is interpreted as details) = Noisier images, noise patches
  - Flow – 2<sup>nd</sup> step
    - Select DC4 image in the viewer
    - Adjust if required, **Noise Level Mid Frequency** (Luma and Chroma) to achieve a **noise-free** edge preserved DC4 image



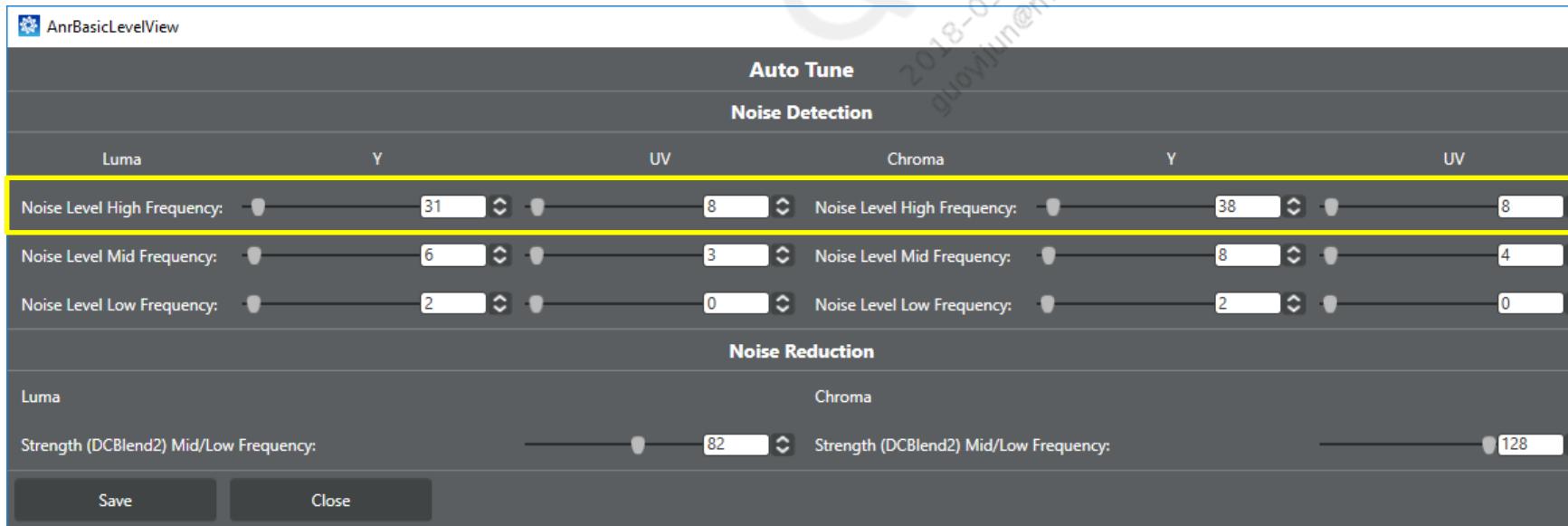
# Tuning Steps – Basic Level (cont.)



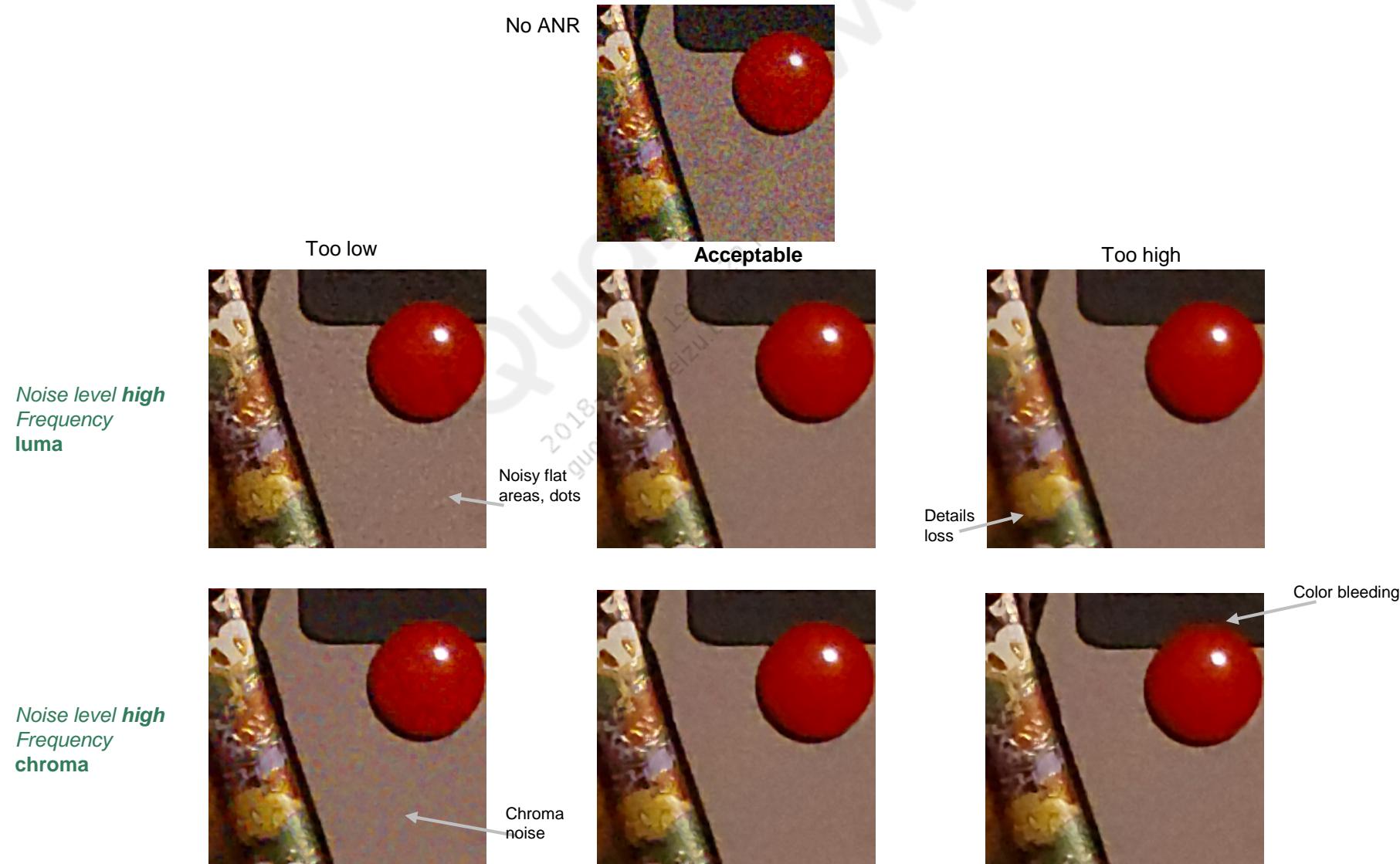
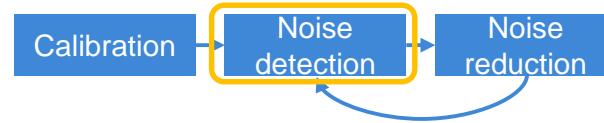


# Tuning Steps – Basic Level (cont.)

- Noise detection
  - Separates details from noise at each frequency range
    - Higher value (more details are interpreted as noise) = eaten edges, more filtering
    - Lower value (more noise is interpreted as details) = Noisier images, noise patches
  - Flow – 3<sup>rd</sup> step
    - Select **Full** image in the viewer
    - Adjust, if required, **Noise Level High Frequency** (Luma and Chroma) to achieve a good compromise between high frequency noise patches to fine texture details retention (in image part close to image center)



# Tuning Steps – Basic Level (cont.)





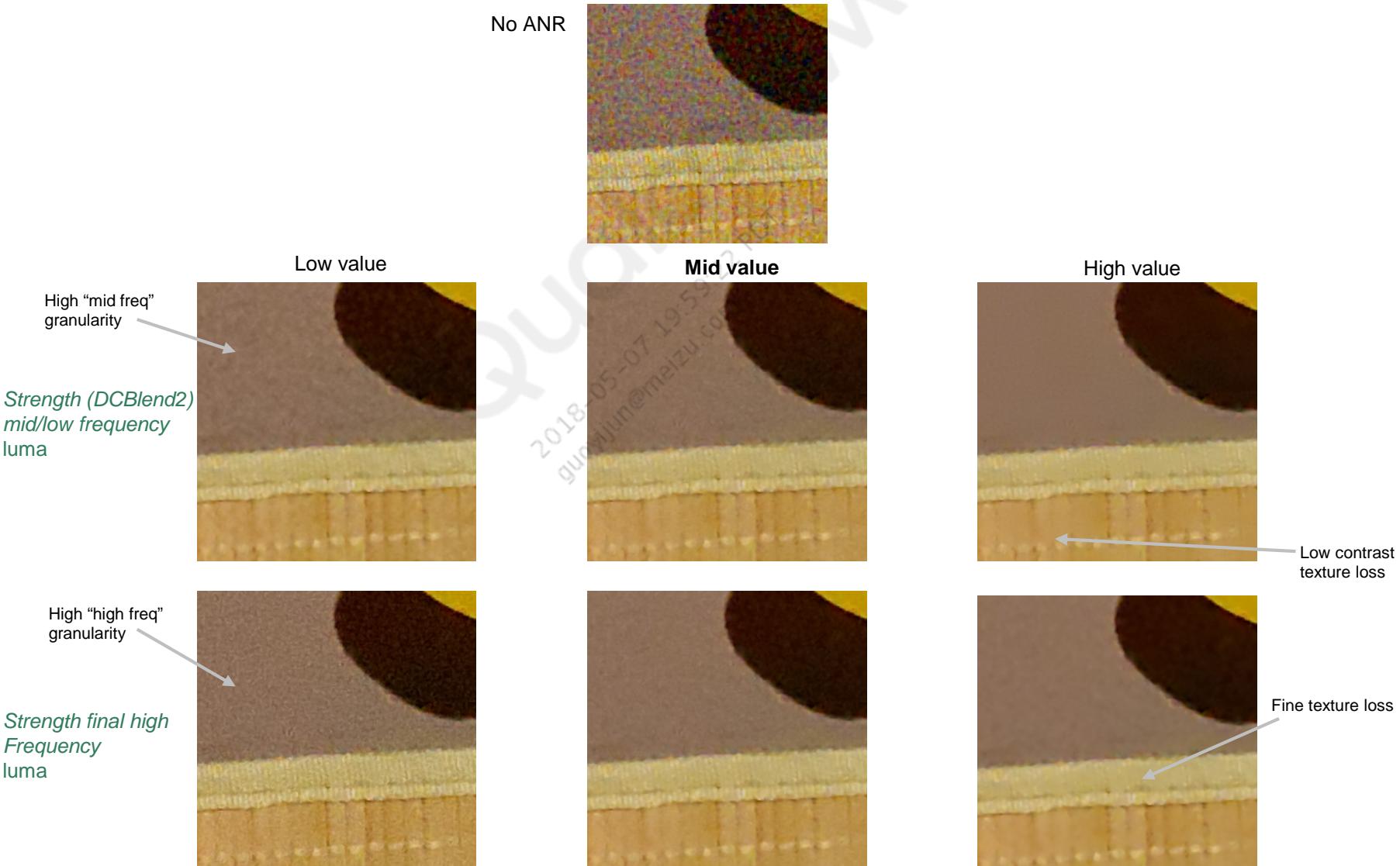
# Tuning Steps – Basic Level (cont.)

- Noise reduction amount
  - Defines the amount of blend between original image to filtered
    - Higher value (blend more) = softer image
    - Lower value (blend less) = more granular and noisy image
  - Flow
    - It is suggested to keep **Chroma** section in **noise reduction amount** sliders at max. value
    - Adjust **Strength (DCBlend2)** mid/low frequency slider to achieve
      - Clean image on flat areas
      - Retained low contrast textures in flat areas
      - Desired mid frequency granularity
    - Adjust **Strength final high frequency** slider to achieve\*
      - Clean image but leaving desired high frequency granularity
      - Texture retention and continuity
      - Reduce harsh transition between flat and not flat areas
    - The tuning engineer may go back to **Noise detection** section according to tuning flavor
      - For example, maybe after lowering **Strength final high frequency**, increase **Noise level high frequency**

\* Available only for stills



# Tuning Steps – Basic Level (cont.)



# Tuning Steps – Base Functions



## Advanced

- The central module of the ANR
- Defines the filtering thresholds according to derivatives
- *Noise detection* in *Basic* section has a direct impact on *Base functions*
- The tuning engineer should tune base functions
  - In all image passes: *Full*, *DC4*, *DC16*, (*DC64*)
  - For *Luma Filter* and for *Chroma Filter*
  - For *Ythsh*, *Utrsh*, *Vtrsh*
  - Total of
    - $4 \times 2 \times 3 = 24$  base functions for still
    - $3 \times 2 \times 3 = 18$  base functions for video
  - Start tuning from the lowest pass to highest
    - Examine the impact on the corresponding pass image
    - Then examine on the higher pass images
    - When trying to resolve an issue, find the lowest pass where the issue exists, place a *POI* (see next slide) and try to resolve
  - *DC64* mostly does not require tuning

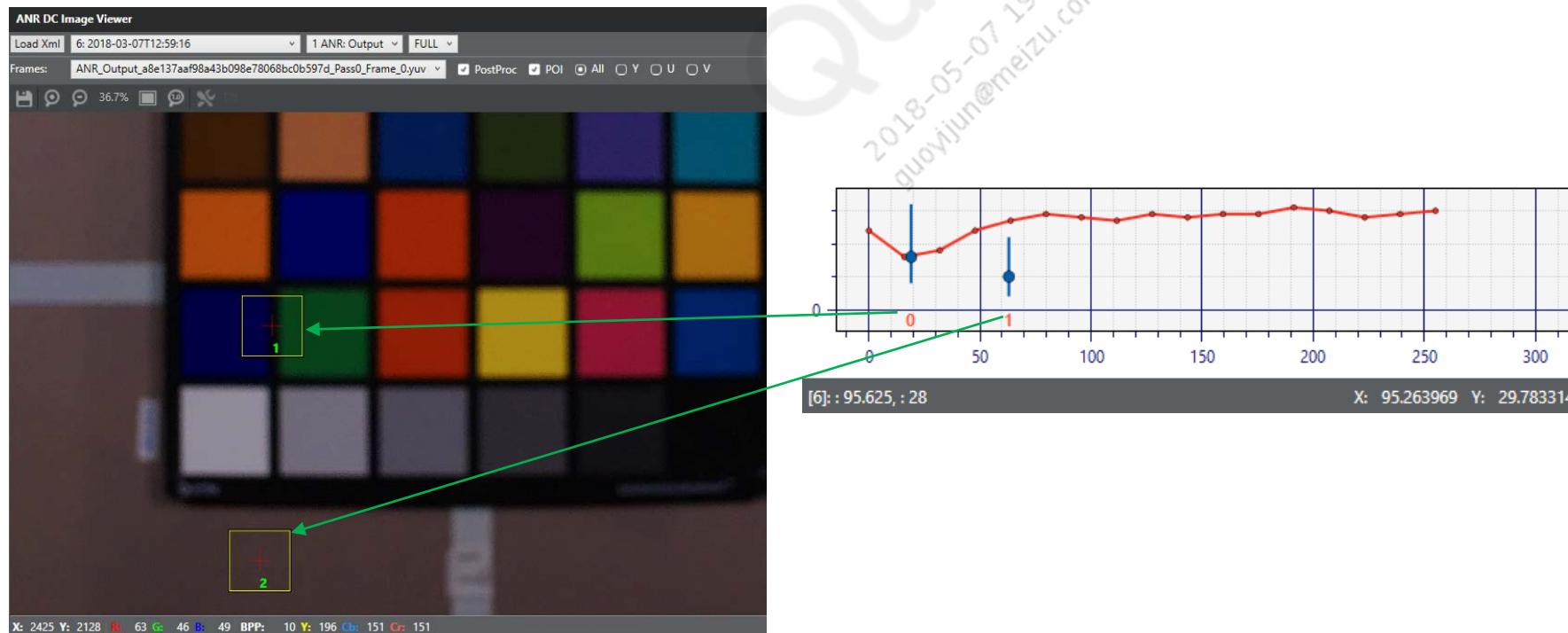


# Tuning Steps – Base Functions (cont.)



## Advanced

- Adding a Point-of-interest (POI)
  - Adding a POI is vital in tuning procedure. It allows the tuning engineer to understand filter decisions and treat issues accurately
  - The POIs are shown in *base functions* window in the form of derivative distribution (discussed in the next slide)
  - Below is an example of an image that passed a calibration and several POIs were placed on it



# Tuning Steps – Base Functions (cont.)



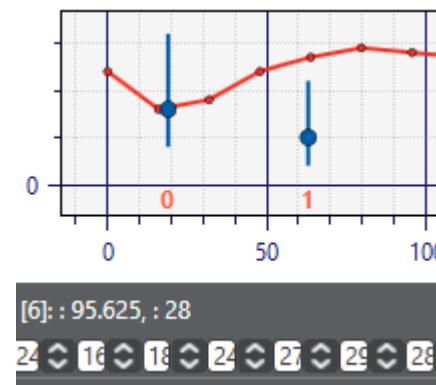
## Advanced

- Threshold function



**Percentage**  
50% of the pixels in the kernel have derivative higher than the blue point and the other 50% are below the blue point

**POI derivatives distribution**  
The lowest part of the "I" represents the lowest derivative in the filter around the POI.  
The top part of the "I" represents the highest derivative in the filter around the POI.



- Example

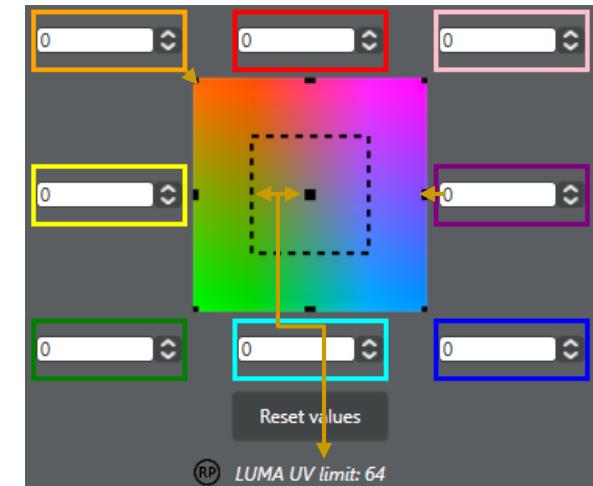
- In this example, the *sample POI* pixel will be filtered with all the pixels in the kernel
- The *pink* pixel will be filtered with 50% of the pixels in the kernel

# Tuning Steps – Base Functions (cont.)



## Advanced

- Color based threshold modifier
  - Offsets the *Threshold function* according to the color. The “I”s in the *Threshold function* will be adjusted accordingly
  - Controls
    - Each spinner box represents a black square
    - *UV limit*: The distance of the black squares from the center. Larger value to get less impact on low color saturation. Smaller value to get more impact on low color saturation. **Is fixed for all the tunings.**  
Range [32:127]

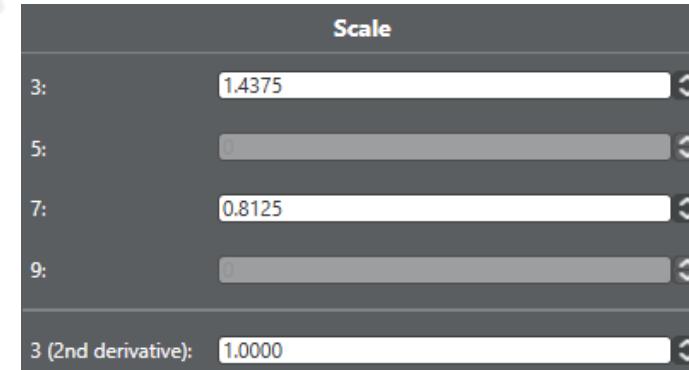


# Tuning Steps – Base Functions



## Advanced

- Scale and Offset
  - Base functions may be modified for different derivative lengths. The Threshold function, in the example mentioned in the previous slides, modifies derivatives of 5 pixel length.
  - The tuning engineer may want to have different *Threshold functions* for shorter or longer derivatives.
  - Controls
    - *Scale*: Scale the function that was set in *Threshold function* by *Scale* for this derivative length.
    - *Offset*: Offset the function that was set in *Threshold function* by *Offset* for this derivative length (find this in the Parameter Editor).
    - *3(2nd derivative)*: Second order derivative scale.
  - Recommendations
    - Keep *Offset* sliders on 0.
    - In some cases, the tuning engineer may prefer to have a higher *scale* for shorter (3x3) derivatives and a lower *scale* for long derivatives (7x7 or 9x9).
    - The tuning engineer may find *3(2nd derivative)* useful for treating contours and local peak detection.



# Tuning Steps – LNR



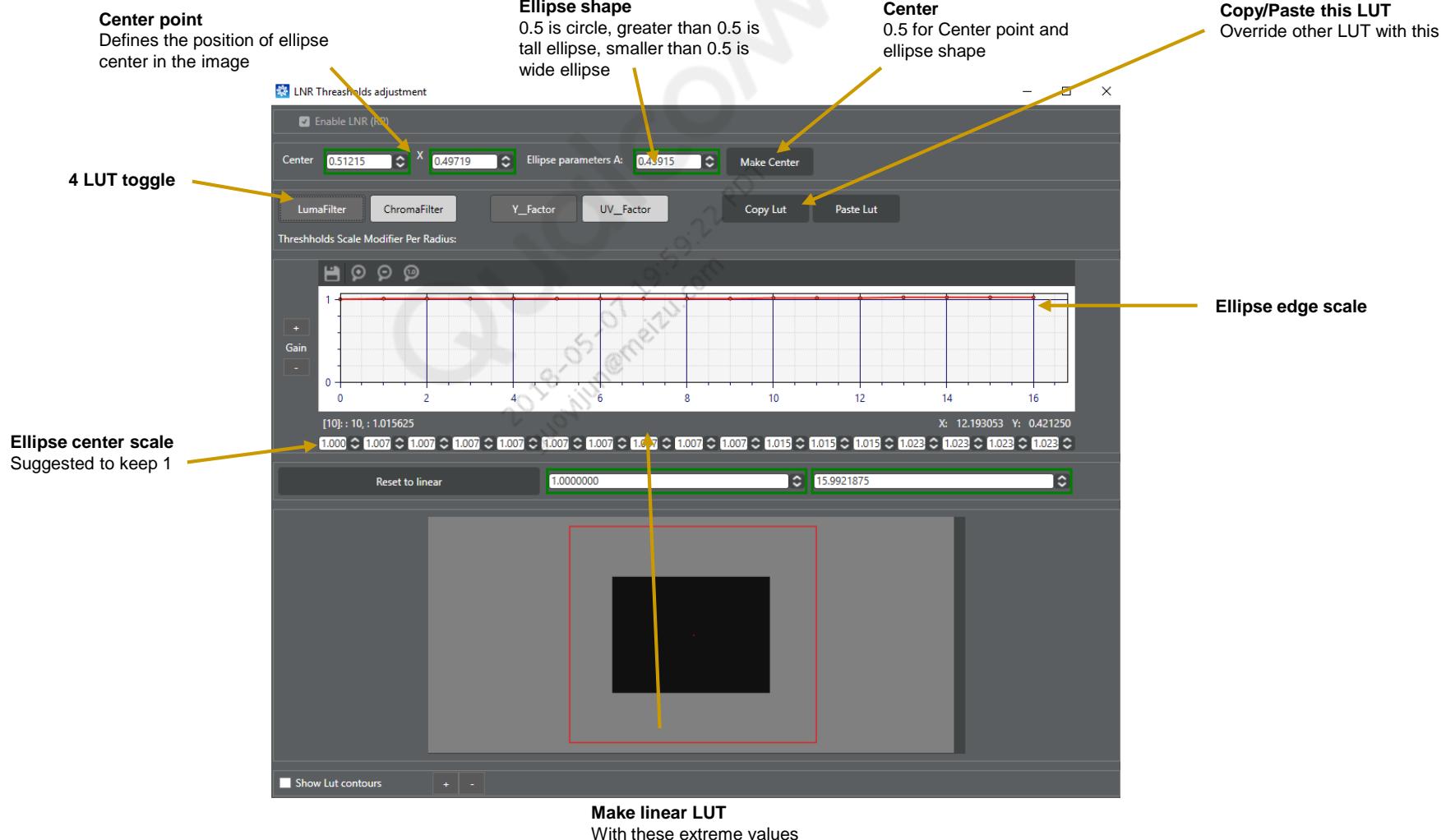
## Advanced

- Lens (dependent) noise reduction (LNR)
- Background
  - Lens shading correction (LSC) may apply gain radially on the image, causing different noise according to the LSC map
  - LNR scales the *Base Functions* threshold according to the distance from the center
- Tuning flow recommendation
  - *Calibration* with a good input generally produces accurate LUTs but the tuning engineer may want to achieve more aggressive or subtle radial filter behavior
  - Tune the *Center* point and *Ellipse shape* if needed
  - Evaluate noise at the *Center* of the image. If the image is over-filtered or too noisy, go back to *Base Functions* to achieve a more preferable result
  - There are four LUTs to adjust; it is recommended to tune one LUT and apply to all
    - Once a decent general LUT is found, adjust separately LUTs if needed, e.g., more radial filtration only on luma
  - The tuning engineer may use POIs at image corners to understand the amount of scale that should be applied to the *Base Function* thresholds
    - If different radial noise behavior is found between DC passes, a *LUT influence* of LNR LUT can be applied for lower passes in the *Advanced* tab

# Tuning Steps – LNR



## Advanced

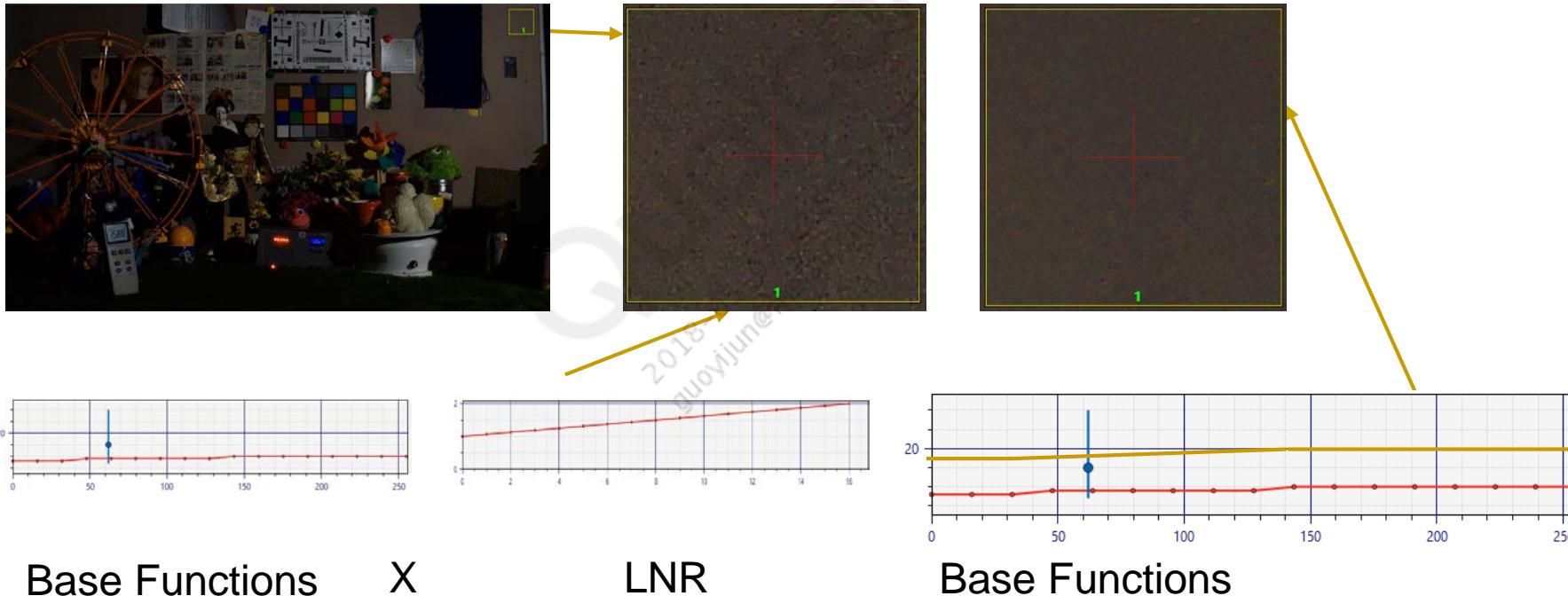


# Tuning Steps – LNR



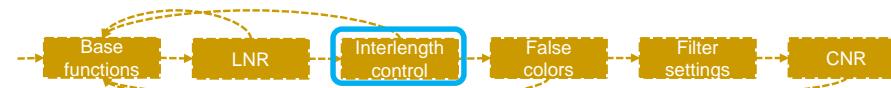
## Advanced

- Example



- In this example, a POI pixel is not filtered enough due to strong LSC in the corner (which amplified the noise)
  - Most of the POI derivative distribution is above the threshold, according to the *Base Functions* dialog
- LNR scales *Base Functions* according to this LUT by almost 2 in the POI location
  - The new threshold is now higher, resulting in more filtering

# Tuning Steps – Interlength Control



## Advanced

- Background
  - Sometimes flat image areas may contain noise leftovers due to high derivatives that do not pass the thresholds.
  - *Interlength control* offers a threshold ratio between flat to non flat areas. It uses a lower pass as a flat detector (as the lower pass is more tolerant to noise). For flat areas, the *Base Functions* of the current pass are scaled accordingly.
  - For example, assume a pixel is on a flat area in *Full* pass. If this pixel's derivatives are mostly higher than the *Base Functions* thresholds, it will not be filtered and noise will occur.  
However, *Interlength control* will check the filtered *DC4* pass – if it is determined as flat, it will raise the thresholds of the *Base Functions* by a certain scale, and thus filter more.
  - Using *Interlength control* allows the tuning engineer to lower *Base Function* thresholds and preserve fine details and still keep flat areas clean.
- Tuning flow recommendation
  - Find a good compromise between *Interlength control* and *base functions*.
  - The risks of *Interlength control* are generally.
    - Damaged low contrast details on flat areas.
      - The pass below is overfiltered – fix in *Base Functions*.
      - *Interlength control* detection should be higher and/or strength lower.
    - Noisy edges
      - *Interlength control* strength is too high.

# Tuning Steps – Interlength Control



## Advanced

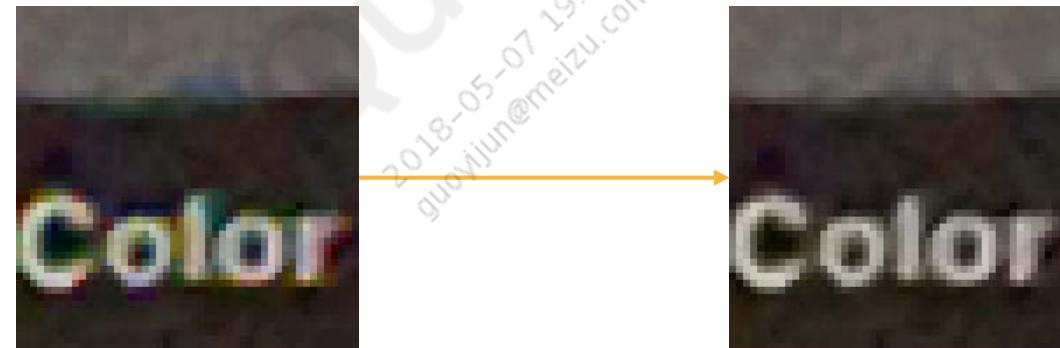
- Strength
  - *thrmodification\_scale*
    - If the underneath pass detected the area as flat, scale the *Base Functions* thresholds by this value
- Flat detection
  - *centerbinarization\_minflatval*
    - For a pixel in an underneath pass, if “flat length” is above this value mark as flat; this is done for each kernel direction
- Flat detection – by neighbors
  - *neighboursbinarization\_minflatval*, *neighboursagreement\_sensitivity*
    - If inside the kernel of the underneath pass there are more than *neighboursagreement\_sensitivity* pixels with “flat length” above *neighboursbinarization\_minflatval*, mark as flat
    - This value should be lower than *centerbinarization\_minflatval*, otherwise it is meaningless

# Advanced Tuning Steps – False Colors



- Background

- Chroma noise is harder to clean on edges due to high luma derivatives
- False colors have two mechanisms cascaded to correct this issue (effects only chroma channels)
  - **Grey Edge Treatment:** Detects strong edge which should be with low chromaticity (gray edge) and adjust chroma filter **Base Functions** to filter it
  - **Chroma Edge Treatment:** Performs median filter on gray edges and on areas without details that have color noise

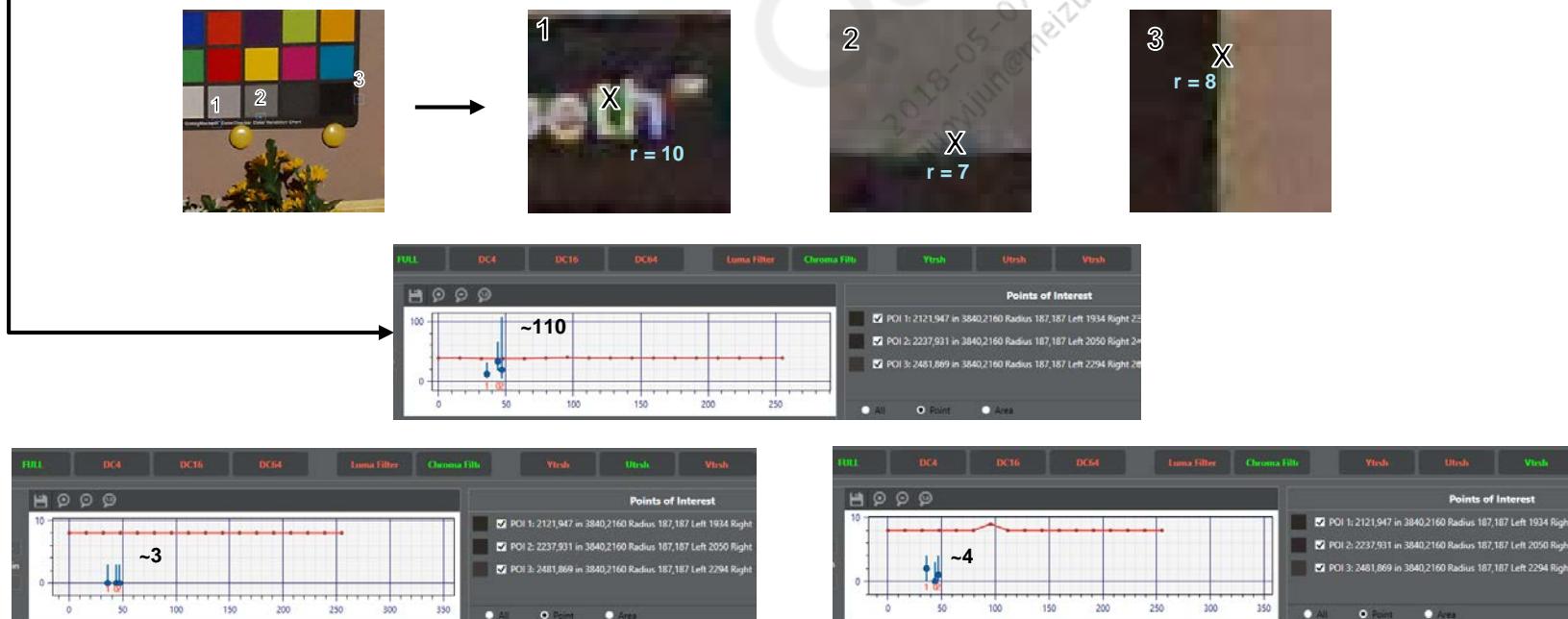


- Tuning flow recommendation
  - Disable **Chroma edge treatment** for all passes
  - Tune **Grey edge treatment** for **DC4** then for **FULL**, bypass for other passes
  - Tune **Chroma edge treatment**. Start tuning with **FULL**. Only if needed, carefully tune **DC4**. Bypass for other passes.

# Advanced Tuning Steps – False Colors (cont.)



- Gray edge treatment – Tuning
  - In a *natural scene* image find several regions where there is an obvious strong gray edge with chroma noise along the edge and place POIs on them
    - It is suggested to have a small color checker inside the *natural scene* for debug
  - For each of these points
    - Write down the high value of derivative distribution for **Y**, **U**, and **V** displayed in Base Functions of **Chroma Filter** for **DC4** and **Full**
    - Measure and write down the chroma radius (intensity) of the noise in **DC4** and **Full**

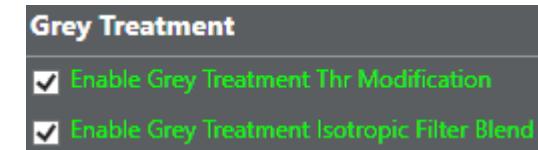


# Advanced Tuning Steps – False Colors (cont.)



- Tune for DC4 and Full
  - Chromaticity Threshold
    - Start: Fill the low observation of chroma radius (example: 7)
    - End: Fill the high observation of chroma radius (example: 10)
  - Enable `enable_grey_treatment_thr_modification`
  - Threshold modification target
    - Y: Fill **Y** high value of derivative distribution (example: 110)
    - U: Fill **U** high value of derivative distribution (example: 3)
    - V: Fill **V** high value of derivative distribution (example: 4)
  - For Full only: `enable_grey_treatment_isotropic_filter_blend`

## Reserved Parameters



## False Colors

Start	End	
Detection - Chromaticity Threshold:	2	3
Detection - Y Max Derivative Threshold:	255	255

(RP) Grey Treatment Thr Modification is Enabled

Y	U	V	
Threshold modification target:	0	0	0

(RP) Grey Treatment Isotropic Filter Blend is Enabled

## Advanced Tuning Steps – False Colors (cont.)



- Examine the impact on *natural scene* and on some other representative images if applicable. Start with **DC4** then go to **Full**
  - Detect less: Decrease **Chromaticity Threshold** or increase **Y max derivative threshold**
  - Detect more: Increase **Chromaticity Threshold** or decrease **Y max derivative threshold**
  - Try to find a good compromise between preserved color edges (for example, MCC patch corners in *natural scene*) to clean gray edges. Prefer the conservative approach (preserve color edges and details) because additional reduction might be possible using **Chroma Edge Treatment**

Qualcomm  
2018-05-07 19:59:22 ART  
guovjun@meizu.com

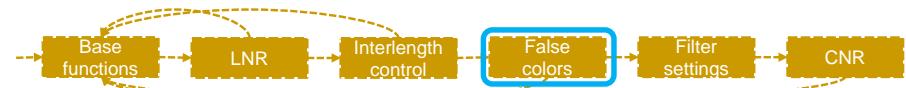
# Advanced Tuning Steps – False Colors (cont.)



- Gray edge treatment – Parameters

- Chromaticity threshold
  - A pixel with chromaticity higher than **End** won't be a candidate for gray edge; lower than **Start** will be a full confident candidate and interpolation between **start** and **End**
- Y max derivative threshold
  - A pixel with Y derivative lower than **Start** won't be a candidate for gray edge; higher than **End** will be a full candidate and interpolation between **start** and **End**
- A pixel will be detected as gray edge only if it was marked as a candidate in **Chromaticity threshold** and **Y max derivative threshold**
- **Enable\_grey\_treatment\_thr\_modification**
  - Enables increasing **Chroma Filter** thresholds in **base functions**
  - **Y, U, V**: will be increased to these values correspondingly (according to confidence)
- **Enable\_grey\_treatment\_isotropic\_filter**
  - Blends pixel's chroma with chroma isotropic filtered pixel according to confidence of gray edge detection
- **Enable\_grey\_treatment\_dcblend2\_chroma\_modification**
  - Blends pixel's chroma with chroma filtered lowpass image pixel according to confidence of gray edge detection

# Advanced Tuning Steps – False Colors (cont.)



- Chroma edge treatment – Parameters

- Chromaticity threshold and Y max derivative threshold

- Gray edge detection – similar to Grey Treatment section above. Pixels detected as gray edge will be filtered with a median filter. This detection overrides Detail Corner Sensitivity Y/UV and Detail Chromaticity Minority Th

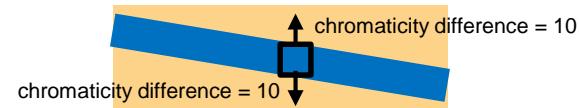
- Detail corner sensitivity Y/UV

- Higher value (more local details identified) = less filtering
    - Lower value (less local details identified) = more filtering

- Detail chromaticity minority th

- If a pixel is located on a strip between two other colors, performing median may smear the pixel
    - If the chromaticity difference in both directions is higher or equal than this value - median won't be applied

Example



If *Detail Chromaticity Minority th* == 8 then this pixel won't pass median filter

# Advanced Tuning Steps – False Colors (cont.)



- Neighbors isotropic detail sensitivity
  - Local details identification according to neighbors detail detection
  - Higher value (fewer neighbors required to be counted as details) = less filtering
  - Lower value (more neighbors required to be counted as details) = more filtering
- Neighbors directional detail sensitivity
  - Similar to Neighbors isotropic detail sensitivity but median still may be performed on directions that not detected as detail
  - It is suggested to set this value lower than Neighbors isotropic detail sensitivity
- Bilateral decision minimal size
  - If *flat length* on any direction is higher or equal to this value, a bilateral filter will be performed instead of median. In this case bilateral filter is generally preferable to median since it is using averaging

# Tuning Steps – Filter Settings



## Advanced

- *Full luma filter kernel*
  - Allows the tuning engineer to define the filter kernels for *flat* and *edge* areas.
    - This setting is for the kernel of the luma filter in *Full* pass only.
    - It is recommended to set the *Edge* kernel “flatter” than *Flat* kernel. This gives fine granularity on flat areas while better noise filtering on edges.
- *Flat\_kernel\_blend\_weight*
  - Defines the amount of blend of the *Flat* kernel on flat area.
    - This setting is for the kernel of the luma filter in *Full* pass only.
    - Max. value means that if the area was detected as fully flat, use the *flat* kernel.
    - Min. value means the only *edge* kernel will be used.
    - It is recommended to keep this parameter at the max. value.
- *Luma \ Chroma DCBlend2 Strength*
  - Defines the blend amount of lower passes, i.e., noise reduction in low frequency.
    - *Strength (DCBlend2) Mid/Low Frequency* in the *Basic Level* directly maps to this parameter.

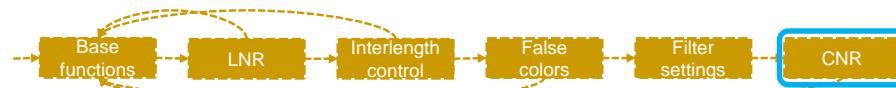
# Tuning Steps – Filter Settings



## Advanced

- Filter Final Strength – background
  - Sometimes noise is detected correctly, but a more subtle reduction is preferred to preserve details. Blending with the unfiltered image may reveal many unwanted peaks.
  - *Filter Final Strength* enables a smart blend of filtered and unfiltered image. The smart blend is designed not to blend back peak noise. If so, the unfiltered pixel may be blended up to  $1 - \text{parameter value}$ .
  - *Luma Strength HF* is directly mapped to *Strength final high frequency* in *Basic Level*. It is recommended to adjust only this parameter.

# Tuning Steps – CNR



## Advanced

- Background
  - In some areas with defined colors it is more flattering to have more filtering.
    - e.g., faces and sky
    - CNR raises the base functions for up to 5 areas in YUV space.
    - For skin color, CNR uses face detection indications to avoid overfiltering areas with similar colors to skin (i.e., wood).
- Tuning flow recommendations
  - Start tuning CNR from the lowest desired pass.
    - For example, if enabling CNR for *Full* and *DC4*, start from *DC4* then continue with *Full*. If enabling CNR for *Full*, *DC4*, and *DC16*, start from *DC16* then *DC4* and finally *Full*.
    - Examine color identification using the traces window.
    - Check the impact on additional images to assess the impact and understand the tradeoffs.

# Tuning Steps – CNR



## Advanced

- Tuning process
  - For each color measure several points in the traces window:
    - UV angle
    - UV radius
    - Y value
  - Enable colors using *Colors count*
  - Fill for each color
    - *Angle range* – extreme values of UV angle measurements for the particular color
    - *Y Range* – extreme values of Y value measurements for the particular color
    - *Saturation Range* – extreme values of UV radius measurements for the particular color
  - Using *Base Functions*, find the required gain to get the function above derivative distribution and enter it in *GainFactor Y/ UV*
  - If using face detection, consider checking *Face detection dependency*

# How to Coordinate with other Modules

---

- Mention when to tune the module
  - For example, Temporal filter (TF) should be tuned after ANR
  - Also, units which should be turned on/off during tuning, for example, in case of TF tuning mention the units for CCM on, ASF and so on
- Other module influence
  - Mention other modules which when changed require some tuning or full re-calibration (depending on the change and their impact)

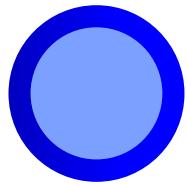
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Tuning Strategy for MCTF and MFNR

---

- ANR tuning for MCTF requires stronger settings than what ANR tuning for MFNR requires.
  - ANR in preview or video mode is the only block that reduces the noise of moving objects

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

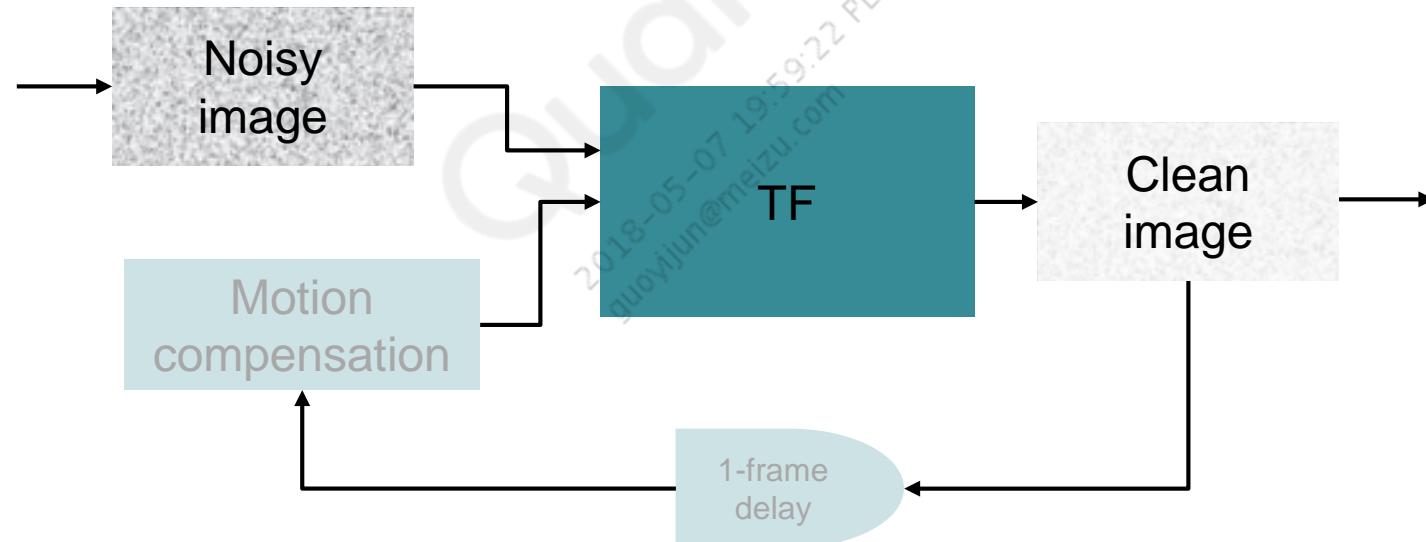


## Temporal Filter (TF)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

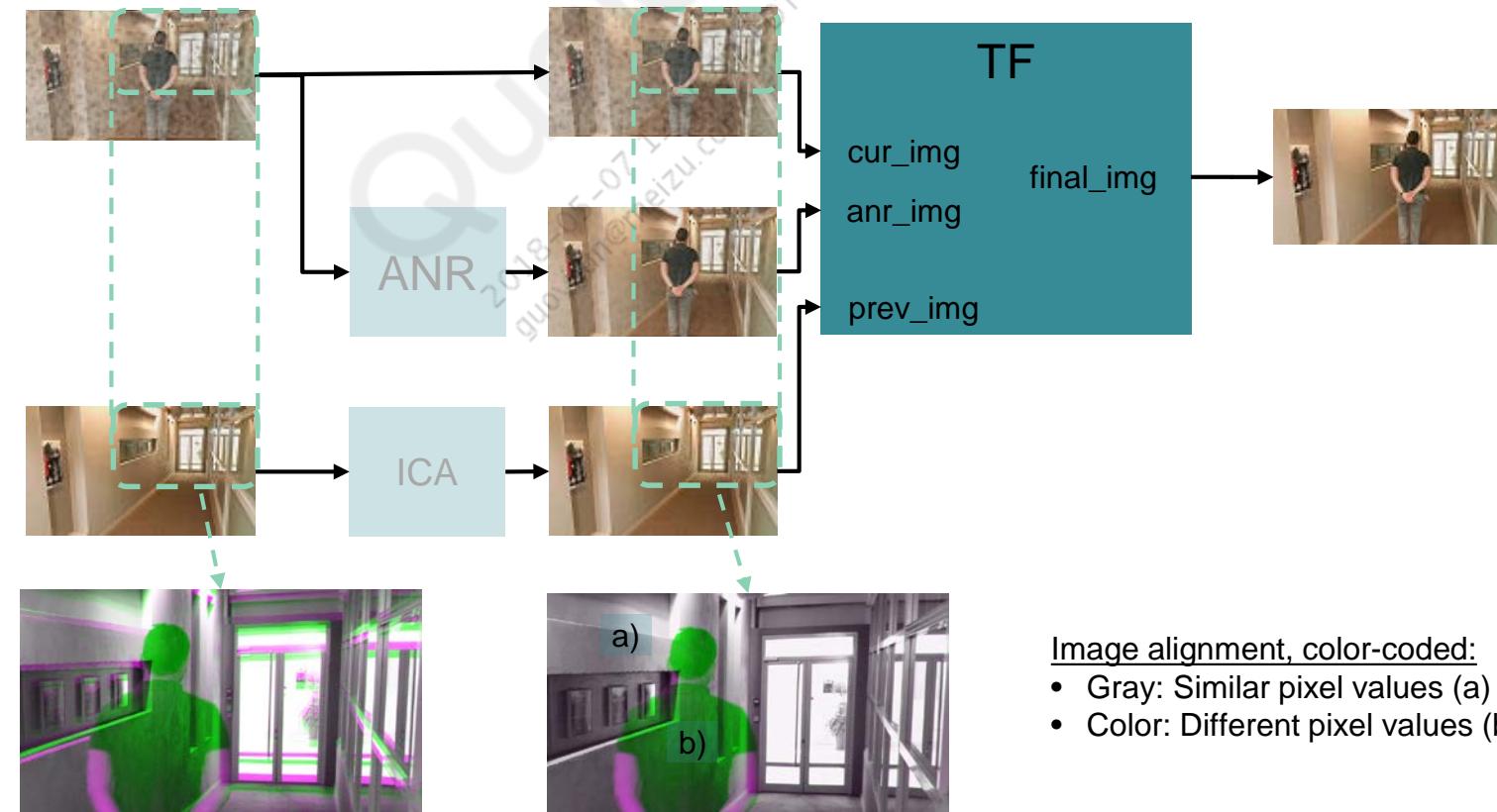
# Temporal Filter – Background

- Temporal noise filtering
  - Works on each video frame
  - Blends current frame with previous frame



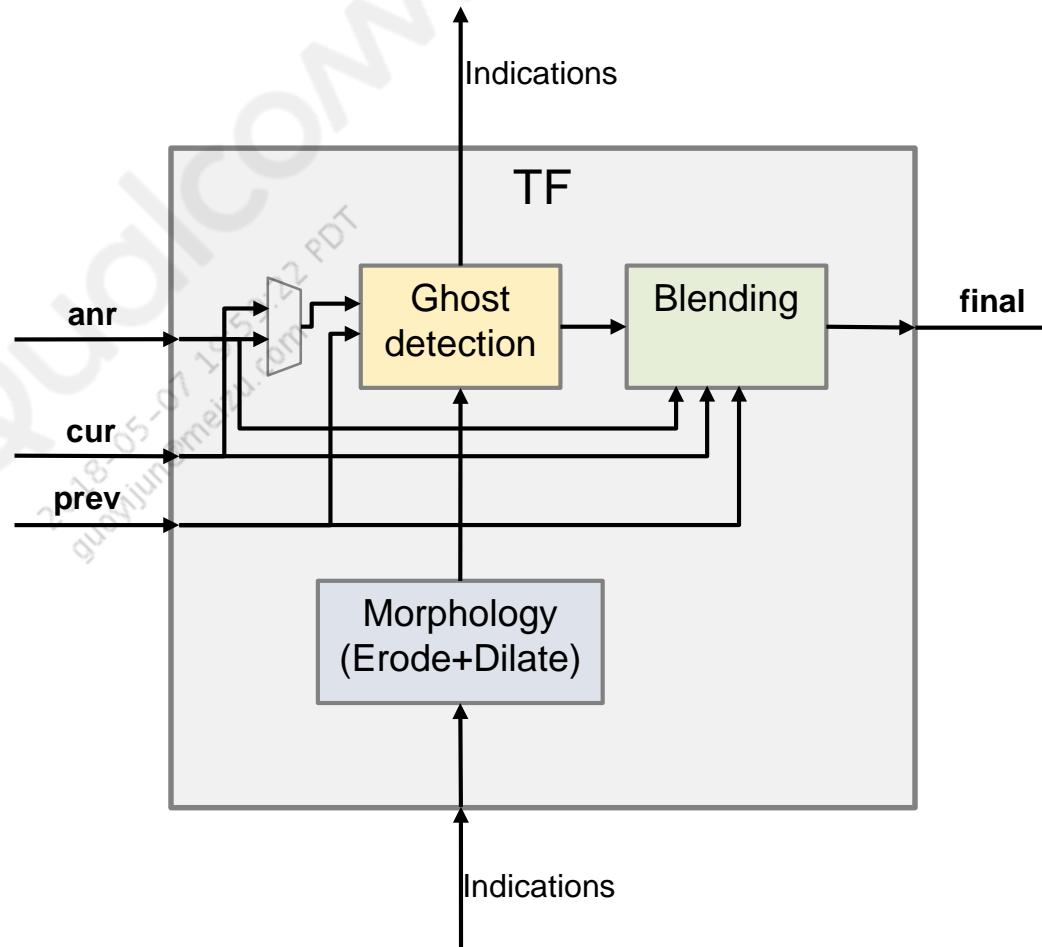
# Temporal Filter – Background (cont.)

- Blending between input images
  - a) Normal operation: blend cur\_img with prev\_img  
(previous frame is warped to align it with current frame)
  - b) Local motion: blend cur\_img with anr\_img



# Overview

- TF block diagram
  - Ghost detection
  - Blending
- Inputs
  - **cur** = Current image
  - **anr** = ANR output
  - **prev** = Previous image  
(after temporal filtering)
- Output
  - **final** = TF output



# Overview (cont.)

---

- TF parts
  - Ghost detection
    - Calculates temporal Filtering strength (FS)
    - If current and previous images have different content>FS=0
      - Otherwise, objects in video would appear twice (*ghosts*)
    - If current and previous images have same content>FS=63 (maximum)
      - High filtering strength ensures maximal noise reduction
  - Blending
    - **final =  $\alpha_1 * prev + (1 - \alpha_1) * cur\_spatial$** 
      - $\alpha_1$  (temporal blending factor) is a tunable function of FS
      - When FS = 0,  $\alpha_1 = 0$
      - When FS = 63,  $\alpha_1$  is maximal
      - $\alpha_{1,max}$  is big>cleaner output, but slow convergence
      - $\alpha_{1,max}$  is small>faster convergence, but noise reduction is weaker
    - **cur\_spatial =  $\alpha_2 * anr + (1 - \alpha_2) * cur$** 
      - $\alpha_2$  (spatial blending factor) is a tunable function of FS
      - When FS = 0,  $\alpha_2$  is maximal
      - When FS = 63,  $\alpha_2$  is minimal
      - $\alpha_{2,min}$  is big>less noise in output, but details may be degraded
      - $\alpha_{2,min}$  is small>more details are preserved in output, but less noise reduction

Qualcomm  
2018-05-07 19:59:28 DT  
guovjun@meizu.com

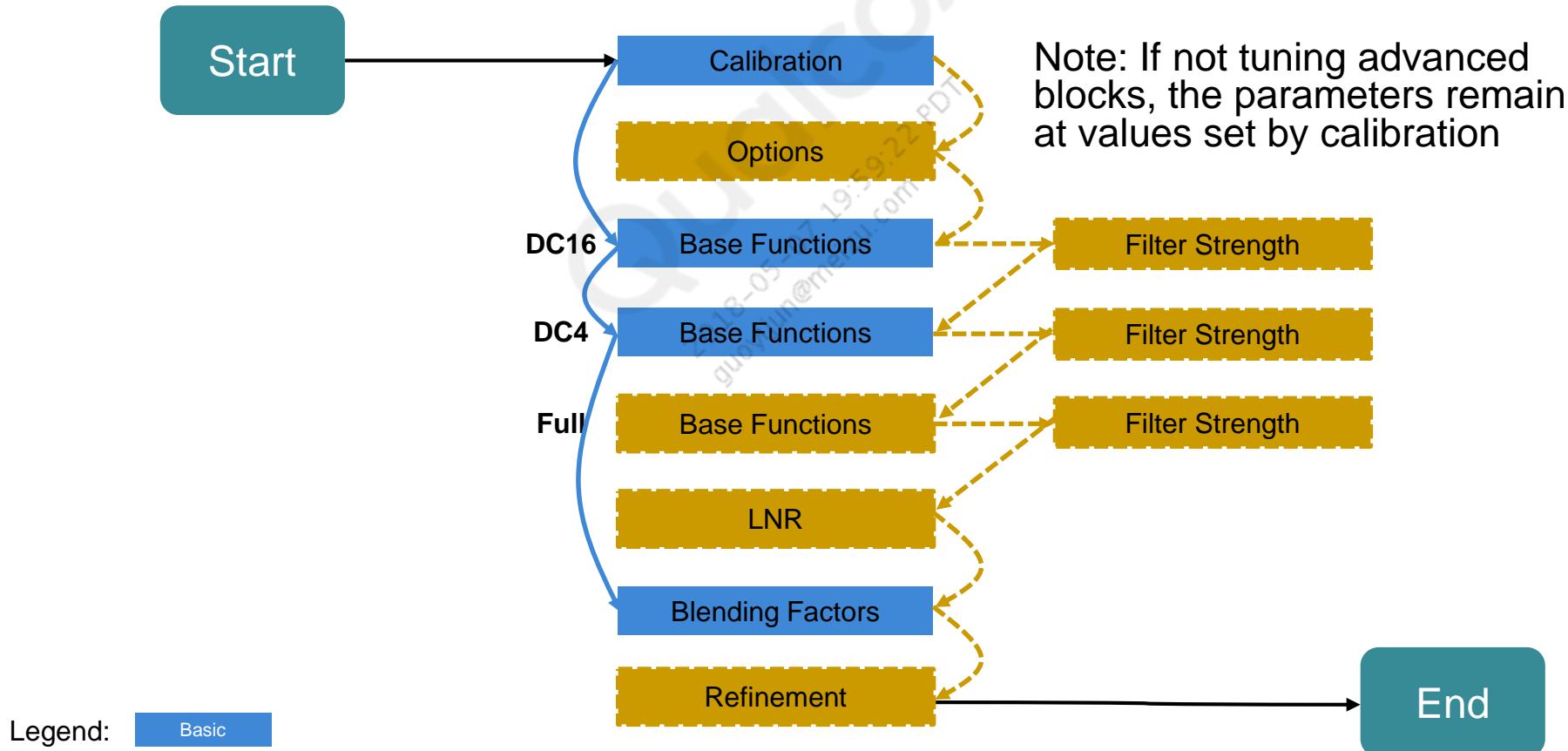
## Overview (cont.)

---

- Multi-scale operation
  - 3 scales: 1:16, 1:4, 1:1
  - Indications are passed from coarser scales to finer scales
  - 1:16 and 1:4 scales: Only ghost detection
  - 1:1 scale: Normally, only blending
    - Turn on 1:1 ghost detection for higher-quality processing, higher power consumption

# Tuning Flow Diagram

Where applicable, use tuning tabs in the order: DC16>DC4>Full



# Calibration – Steps

1. Select 1 flat field image and at least 1 MCC.

Load MCC Images

Load Flat Image

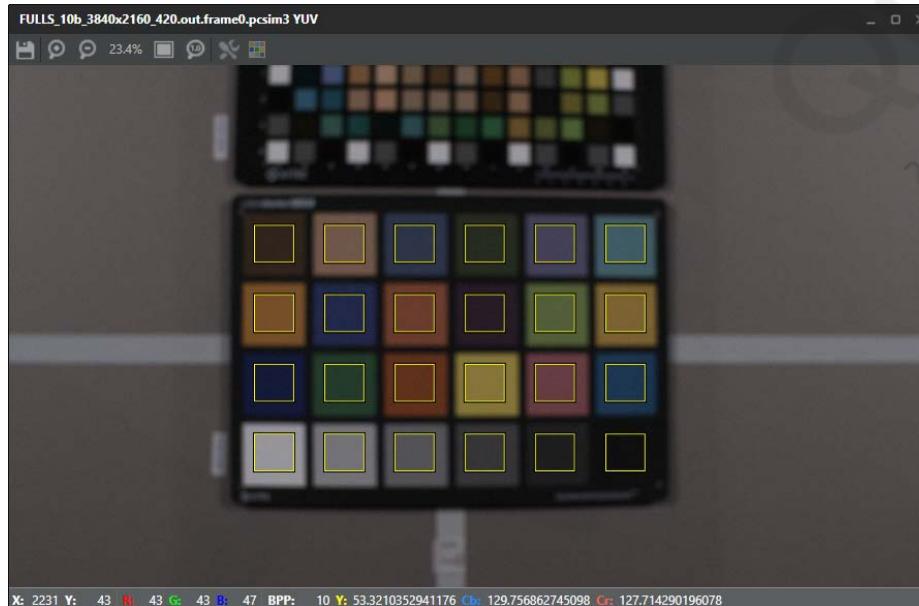
Make sure that the MCC patches are properly selected and covers ~80% of each patch and is center aligned. If not, correct it by dragging them to fit.

2. Click **Calibrate** and **Calibrate LNR**.

Calibrate

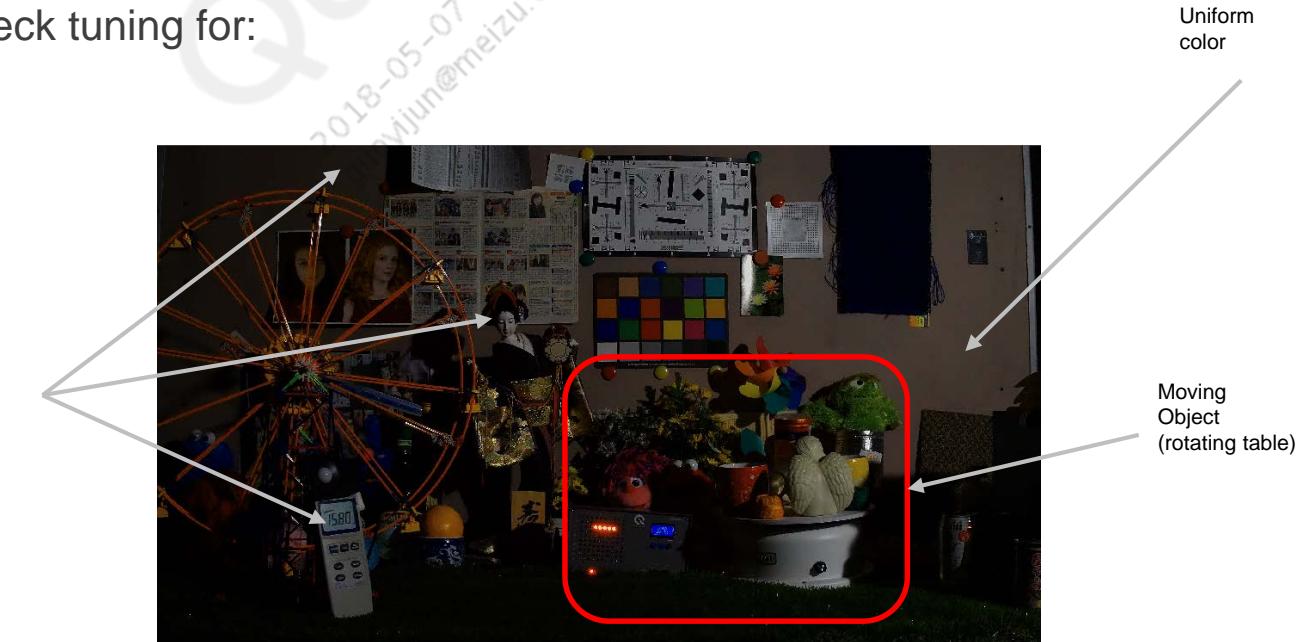
Calibrate LNR

3. Examine the result on a motionless natural scene video.



# Basic Tuning Steps

- Tune on video with a moving object
  - Camera should be fixed (use tripod)
  - The moving object (for example, face) should be ~1/3 the size of the frame
  - No changes in lighting
- Applicable only after calibration
- Look at video frame 5 or later (ignore first 5 frames)
- Use Points of interest (POI) to check tuning for:
  - Areas of uniform color
  - Stationary objects
  - Moving objects



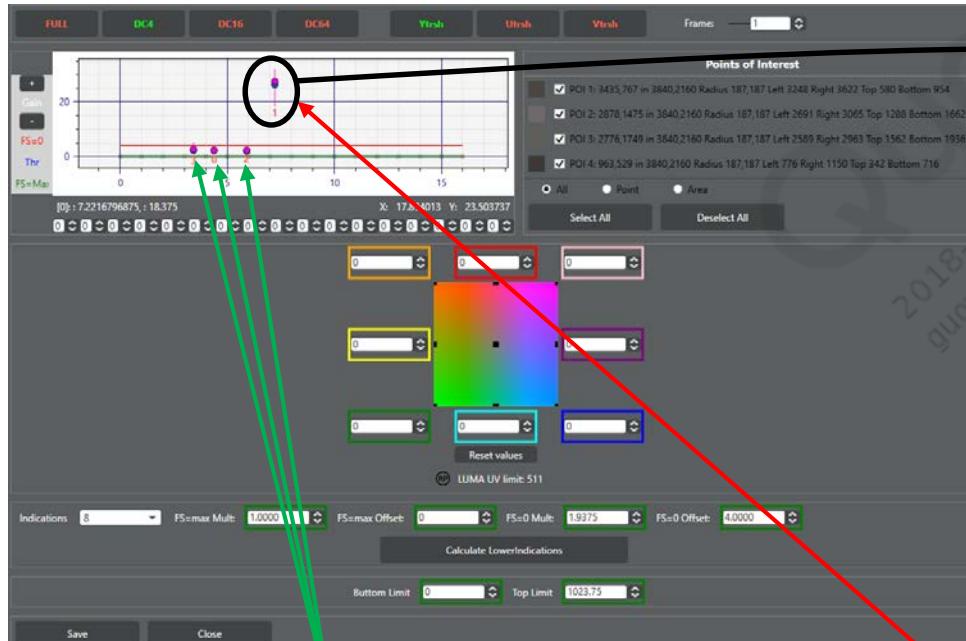
# Basic Tuning Steps (cont.)

- Using POI
  - Place POIs for stationary objects and moving objects
  - Each POI has a resizable square region around it
  - After placing POIs, process the image



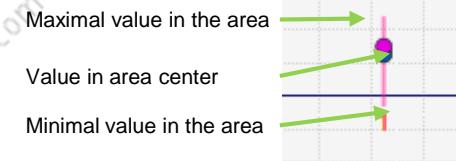
# Tuning Steps – Base Functions

- Base functions separate noise from details
  - Use - Gain + controls to move the red and green lines together
  - POIs for moving objects – values of statistics should be above the red line
  - POIs for stationary objects – values of statistics should be below the green line



POI for a moving object

POIs for stationary objects

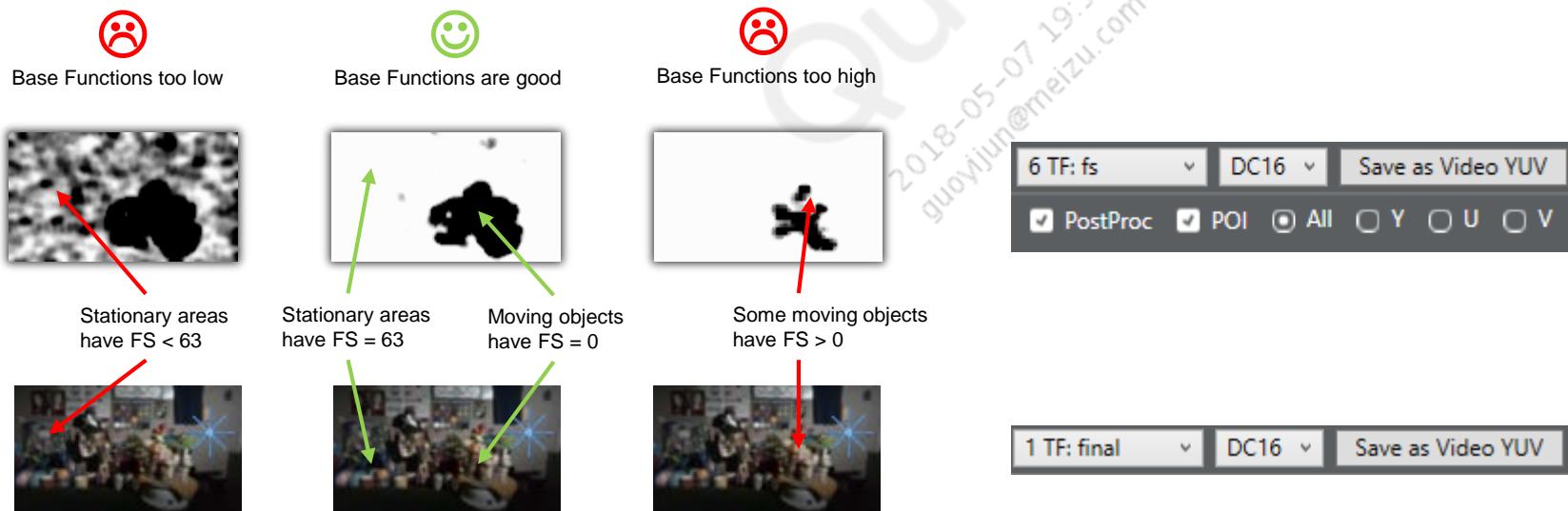


- If it is not possible, compromise and check quality

# Tuning Steps – Base Functions (cont.)

Calibration → **Base Functions** → Blending Factors

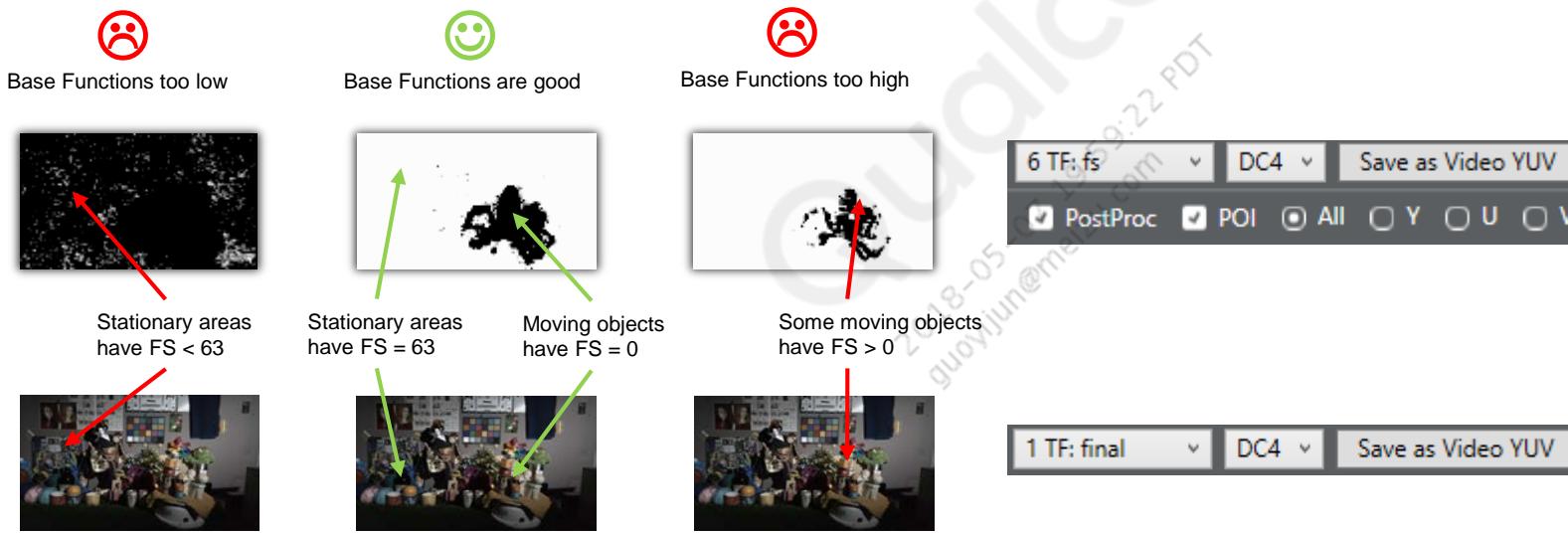
- Start tuning with DC16
- To assess quality of tuning, look at **Filtering strength (FS)**
  - Moving objects should have **FS = 0** (black)
  - Stationary objects should have **FS = 63** (white)



# Tuning Steps – Base Functions (cont.)



- After DC16 is good, tune DC4
  - Moving objects should have FS = 0 (black)
  - Stationary objects should have FS = 63 (white)



- Do not tune base functions for full pass

# Tuning Steps – Base Functions (cont.)

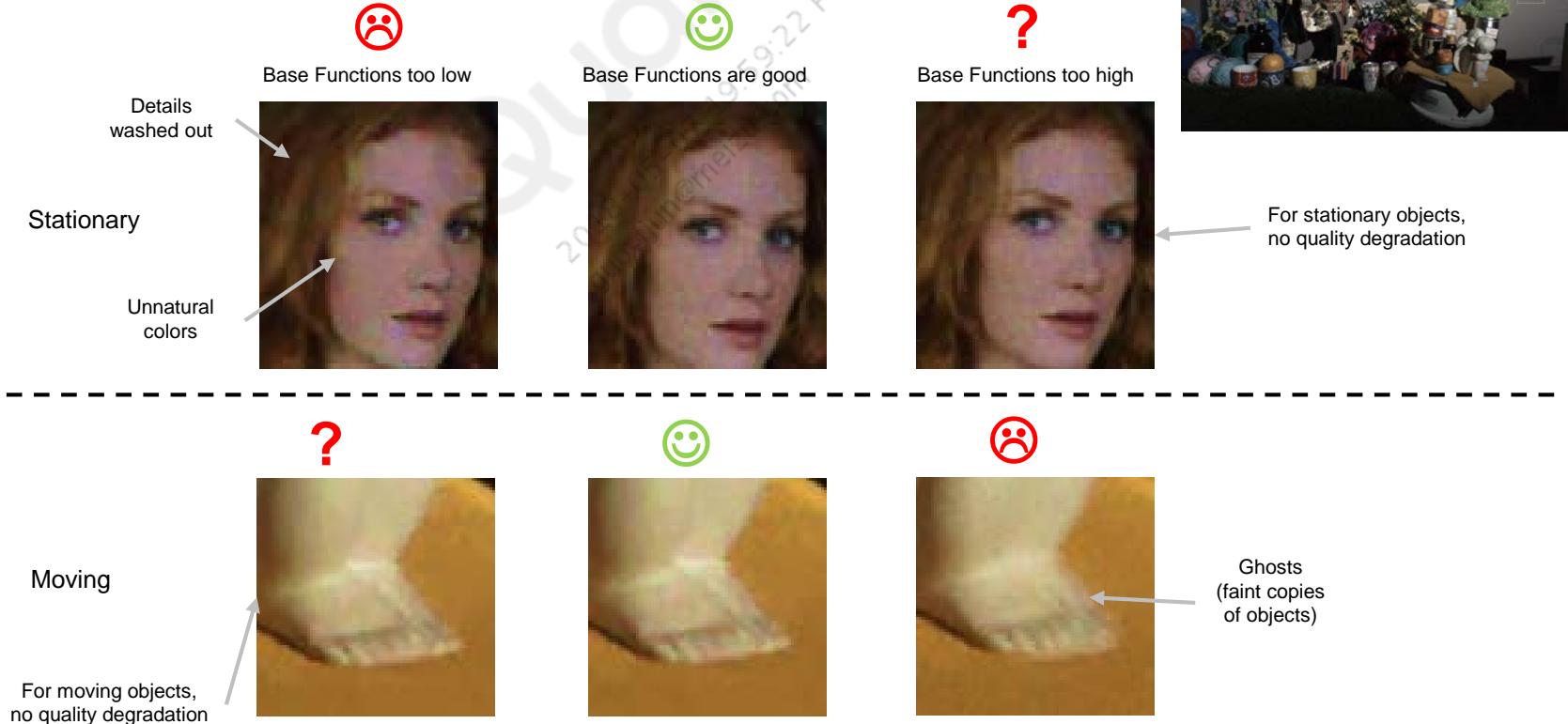
- If unsure whether FS is good, look at final image

— In the final image, problems are harder to spot

— Choose FULL Pass

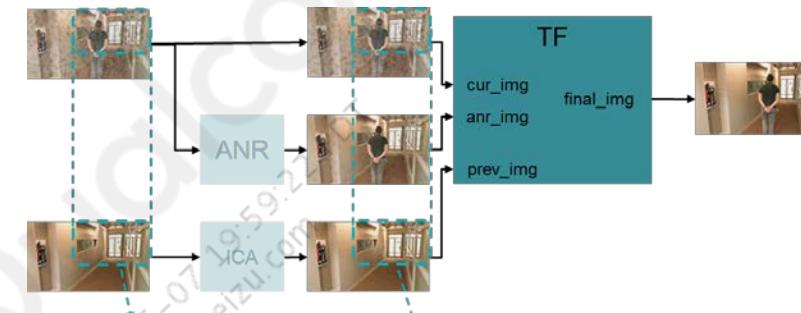
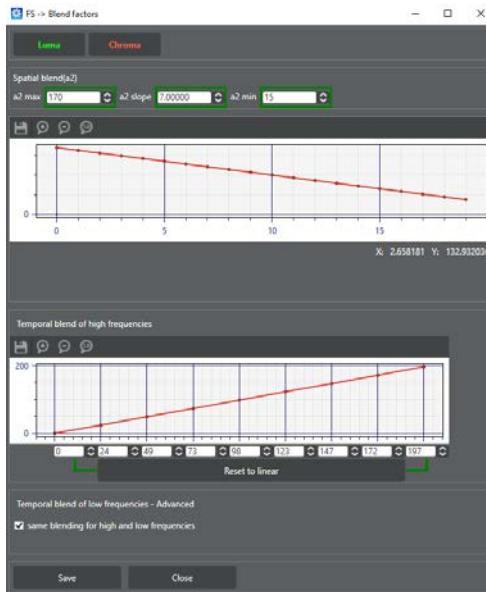
— Choose video frame 5 or later (ignore the first 5 frames)

1 TF: final    FULL    Save as Video YUV



# Tuning Steps – Blending Factors

- Blending factors control the strength of TF



- Spatial blend provides synergy between ANR and TF
  - a2 max:** Should be almost 100%
    - Lower values restore fine details on moving objects
  - a2 min:** Amount of spatial denoising to add to temporal denoising
    - For weak noise, use 0%
    - For very noisy videos, up to 70% may be necessary
    - Higher for Chroma than for Luma

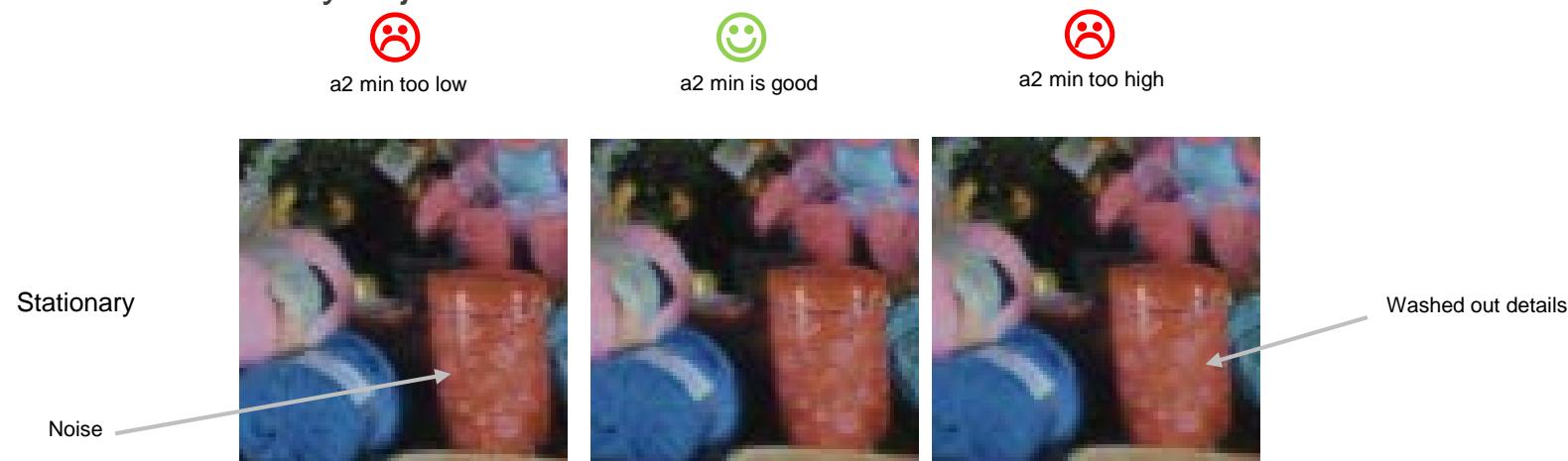
# Tuning Steps – Blending Factors (cont.)

Calibration → Base Functions → **Blending Factors**

- Spatial blend tuning
  - Choose FULL Pass
  - Choose video frame 5 or later (ignore first 5 frames)
- $a2$  max tuning – affects moving objects



- $a2$  min tuning – affects stationary objects



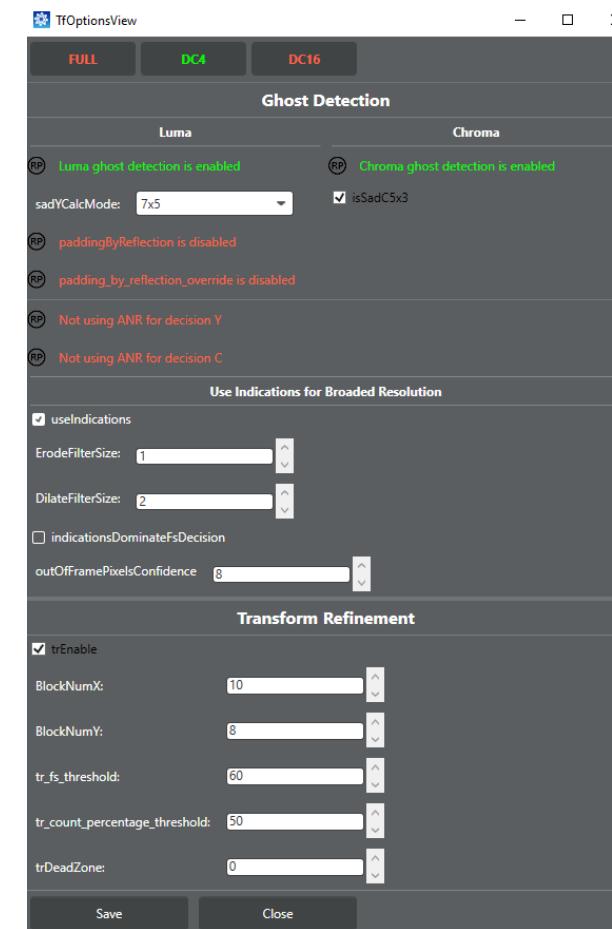
# Tuning Steps – Options



- Options

- DisableChromaGhostDetection
  - Set if not enough data for chroma tuning (e.g., no color chart)
- ErodeFilterSize, DilateFilterSize
  - 0 = disabled; 1 = 5x5; 2 = 3x3
  - Erode filter larger than Dilate filter → less ghost artifacts
  - Erode filter equal to Dilate filter → more noise reduction on moving objects
  - Erode filter smaller than Dilate filter – forbidden
- BlockNumX, BlockNumY, trDeadZone →

See Warping Transform Refinement in later slides

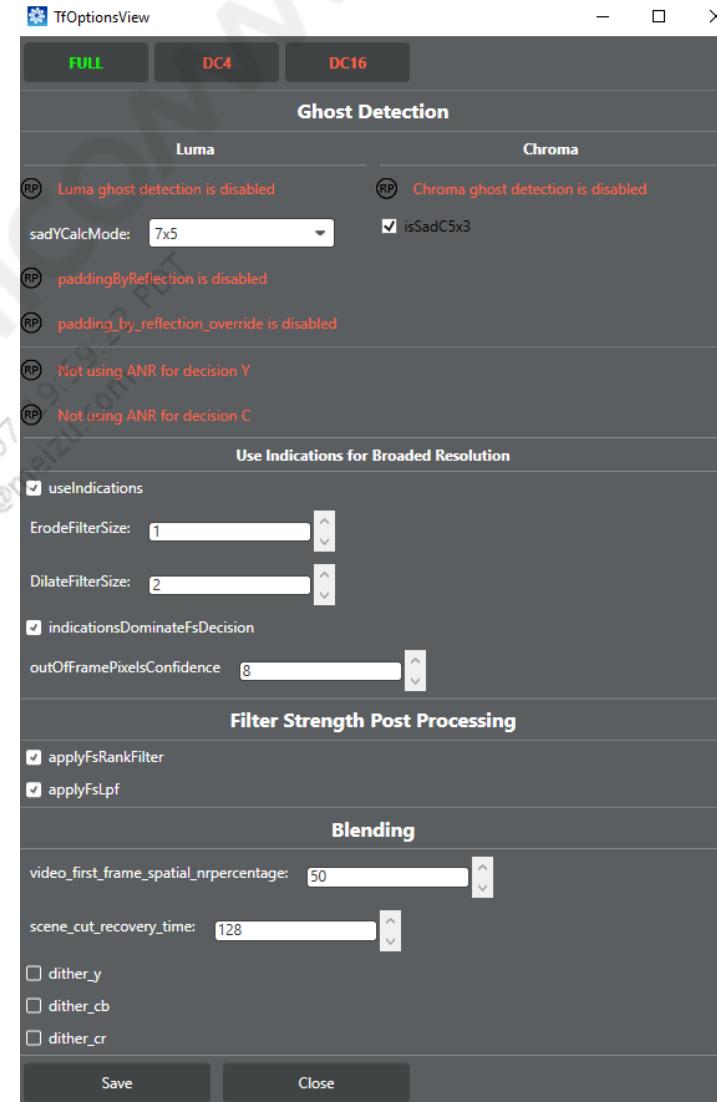


# Tuning Steps – Options



- Options (High-Quality mode)
  - Causes greater power consumption
  - Requires **Basic Functions** tuning

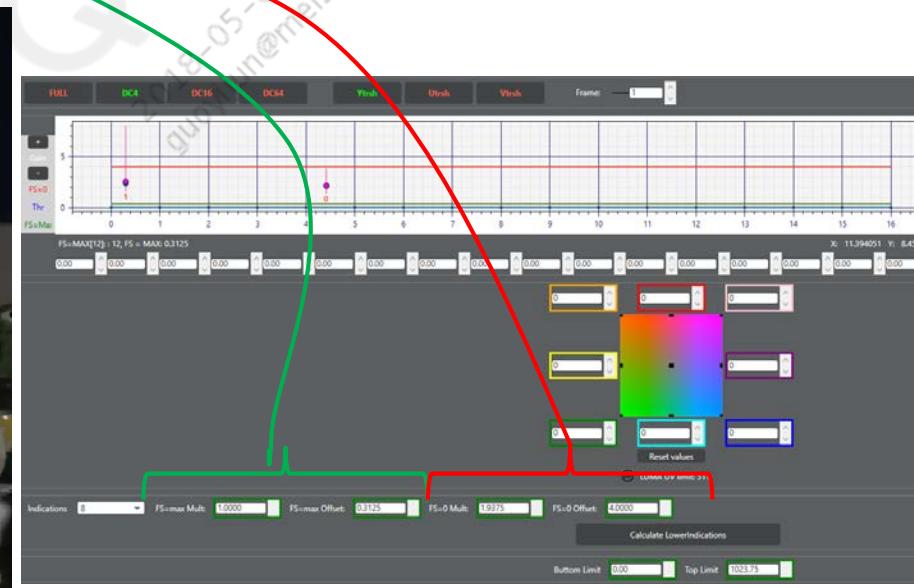
Ghost detection settings



# Tuning Steps – Filter Strength



- Improving **Base Functions** tuning
  - Higher than red line shows a moving object
  - Lower than green line shows a stationary object
  - Between the lines means it is uncertain
- Filter Strength – tune distance between the lines
  - FS = max controls affect the green line
  - FS = 0 controls affect the red line



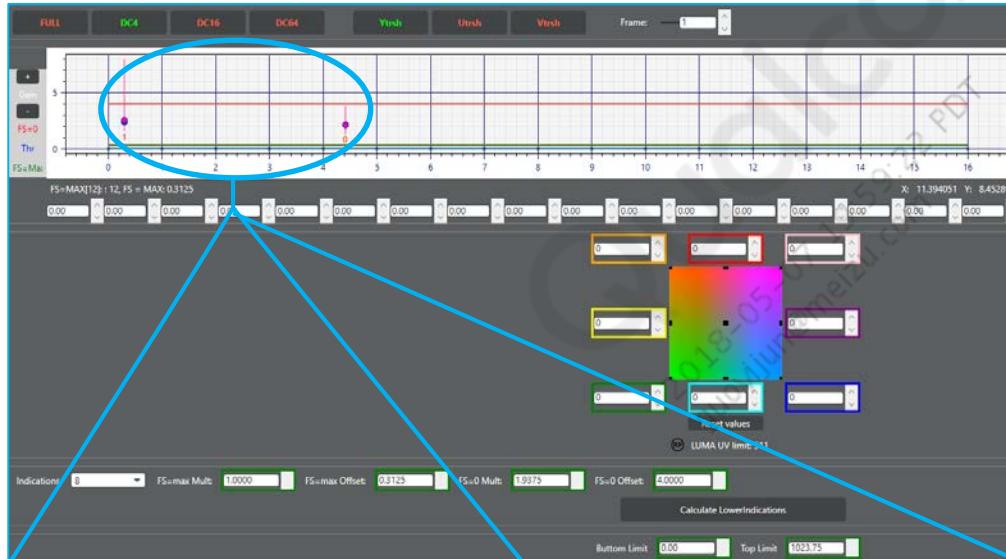
# Tuning Steps – Filter Strength



- After tuning for indications 8
  - Calculate for lower indications

Indications

Calculate LowerIndications



Red → green transition is OK 😊



Red → green transition is too sharp 😕



Red → green transition is too slow 😕

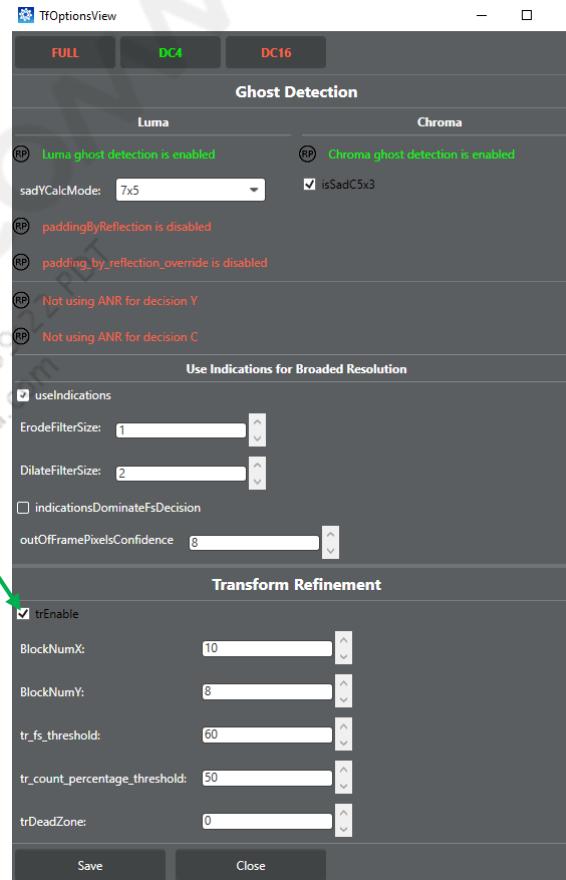
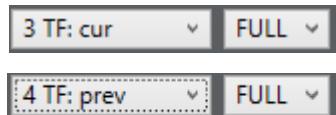


# Tuning Steps – Refinement



- Options – Warping Transform Refinement

- Controlled by trEnable in DC16 and DC4 tabs
- To assess the quality of image alignment:
  - Compare the current image with the previous image
  - (Use the **FULL** tab)



- BlockNumX, BlockNumY

- Image with trEnable=1 should be better than trEnable=0
- If same or worse, decrease BlockNumX and BlockNumY
- If BlockNumX= BlockNumY=1 and still worse:
  - Set trEnable = 0

# How to Coordinate with Other Modules

---

- Mention when to tune the module
  - TF should be tuned after ANR
  - During TF tuning, units should be turned on:
    - CAC
    - CCM
    - ASF
    - GLUT
    - 2D LUT
    - Chroma suppression
    - Skin color enhancement
    - GRA
    - M/N DS
- Other module influence
  - ANR – If tuning changes, re-tune TF blending factors

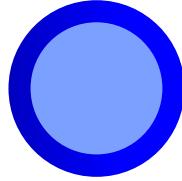
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Tuning Strategy for MCTF and MFNR

---

- TF tuning for MCTF requires conservative ways to avoid ghost artifacts, as ghost artifacts are seen more easily in video and preview modes than snapshot mode.
  - Thresholds in Base Functions for MCTF should be set relatively lower than the MFNR thresholds.

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com



# Multi-Frame Noise Reduction (MFNR)

2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Multi-Frame Noise Reduction

---

## MFNR Tuning

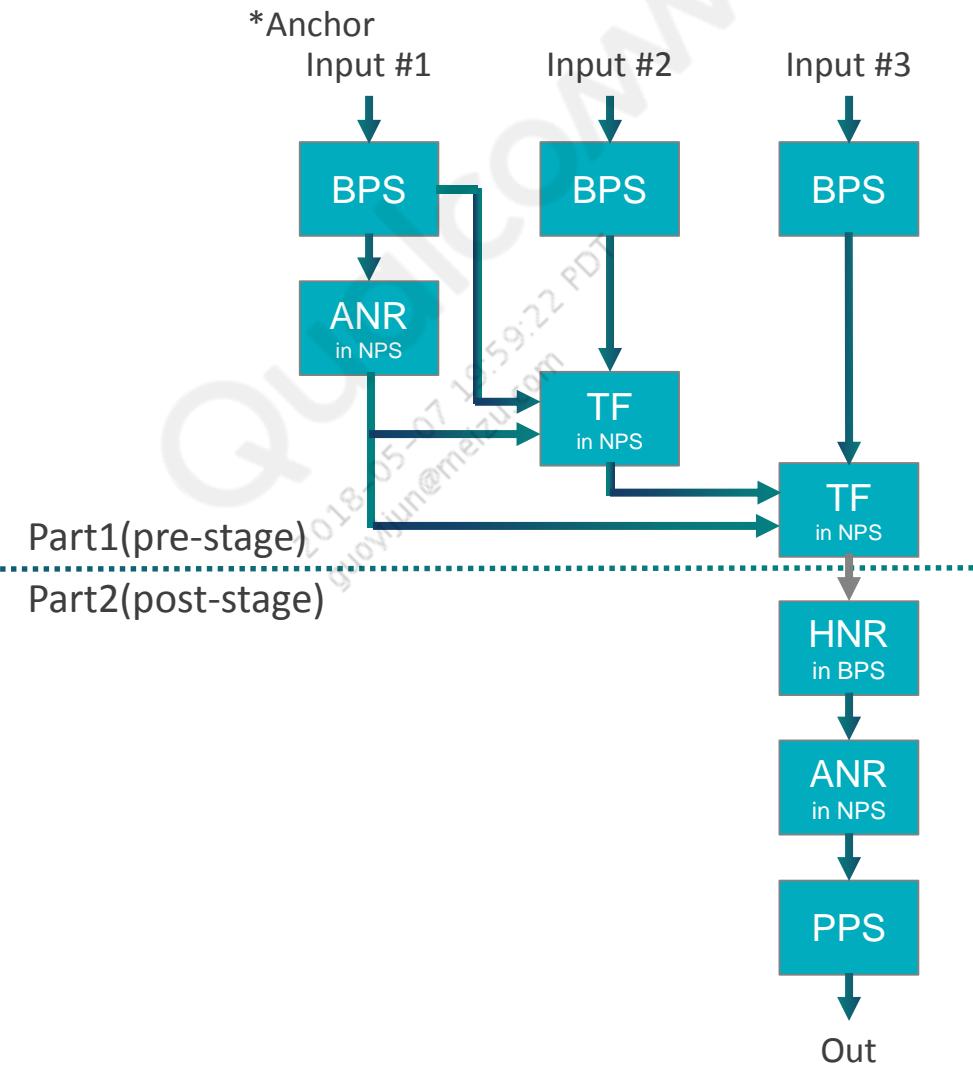
- MFNR flow
  - Overall tuning steps
- Pre-stage tuning
  - BPS and ANR tuning
  - TF tuning
- Post-stage tuning
  - HNR and ANR tuning

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# MFNR Flow

## Tuning Steps

1. BPS without HNR
2. ANR (pre-filtering)
3. TF (blending)
  - A. Base functions
  - B. Blending factors
4. HNR and ANR (post-filtering)
5. PPS



# Pre-stage Tuning

---

## BPS and ANR Tuning

- ANR filtered image is mainly used in moving objects
  - Tune ABF, GIC, and PNR in BPS to reduce strong noise
  - Noise in moving objects should be removed in BPS and ANR because TF cannot remove that kind of noise

## TF Tuning

- Check the regions that including moving or stationary objects
- Check the filter strength (FS) map to see if it is detected as a moving object
  - Change thresholds in Base Functions by using point of interests (POIs) on moving and stationary objects
  - Moving objects should have FS= 0 (black)
  - Stationary objects should have FS= 63 (white)
- Check the FS graph shape
- Blend the input image (tune  $\alpha_2$ )
  - Balance the noise levels of stationary and moving objects to appear similar

# Pre-stage Tuning

## Base Functions

- The area where the hand is moving can be detected as a moving object ( $FS=0$ ) by decreasing the thresholds of the base functions.
- The hand will be preserved because an anchor image is used only and multi-frames are not aggregated.



*High thresholds of base functions*



*Low thresholds of base functions*

# Pre-stage Tuning

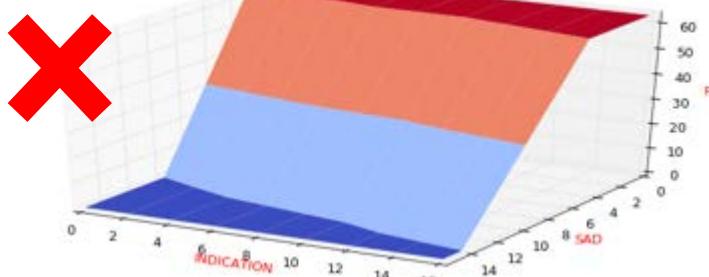
## Filter Strength (FS)

- The FS map could be incorrectly estimated when the FS is shown as below (left figure). It cannot properly use the information from a lower pass.
- Perform calibration again or tune Base Functions again.
  - Do not substantially change the following Base Function values: FS=max Mult, FS=max Offset, FS=0 Mult, and FS=0 offset

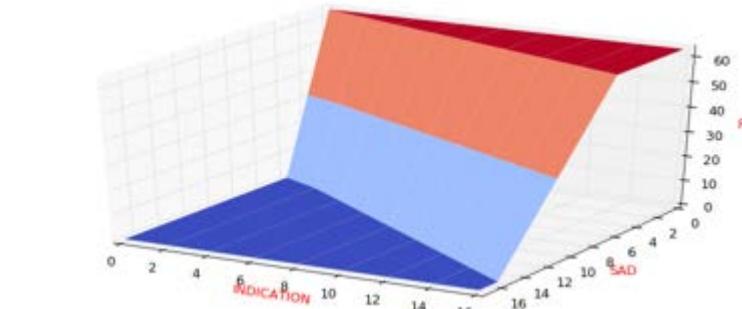
TfFilterStrengthView														
FULL	DC4	DC16	Luma	Chroma	Ind 0	Ind 2	Ind 4	Ind 6	Ind 8	Ind 10	Ind 12	Ind 14	Ind 16	OOF
FS = max mult	0.37	0.5	0.56	0.63	0.63	0.63	0.69	0.75	0.81	0.06	0.06	0.06	0.06	0.06
FS = max offset	1	0.25	0	0	0	0	0	0	0	0	0	0	0	0
FS = 0 mult	1	1.13	1.25	1.31	1.38	1.44	1.56	1.63	1.68	0.25	0.25	0.25	0.25	0.25
FS = 0 offset	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	1.00

TfFilterStrengthView														
FULL	DC4	DC16	Luma	Chroma	Ind 0	Ind 2	Ind 4	Ind 6	Ind 8	Ind 10	Ind 12	Ind 14	Ind 16	OOF
FS = max mult	0	1.13	0.25	0.38	0.5	0.63	0.75	0.88	1	0.06	0.06	0.06	0.06	0.06
FS = max offset	0	0	0	0	0	0	0	0	0	0	0	0	0	0
FS = 0 mult	0	0.19	0.44	0.69	0.94	1.19	1.44	1.69	1.94	0.25	0.25	0.25	0.25	0.25
FS = 0 offset	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	1.00

e.g., when threshold = 6



Bad shape of FS

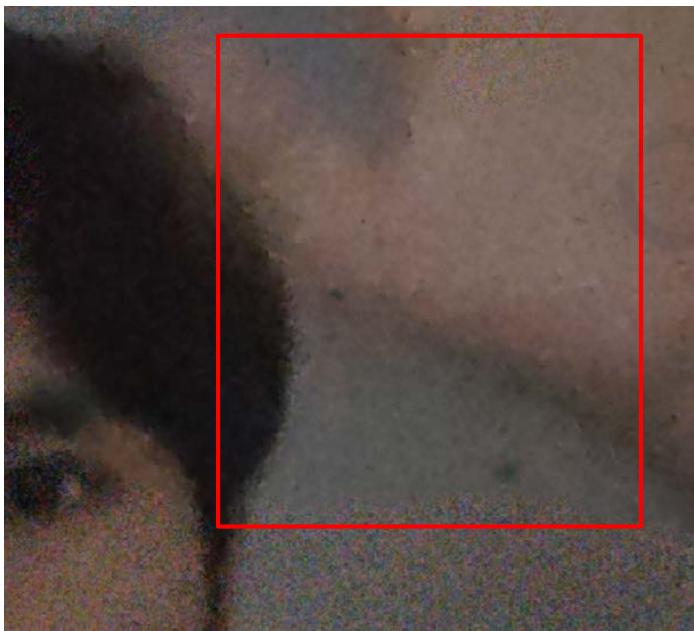


Good shape of FS

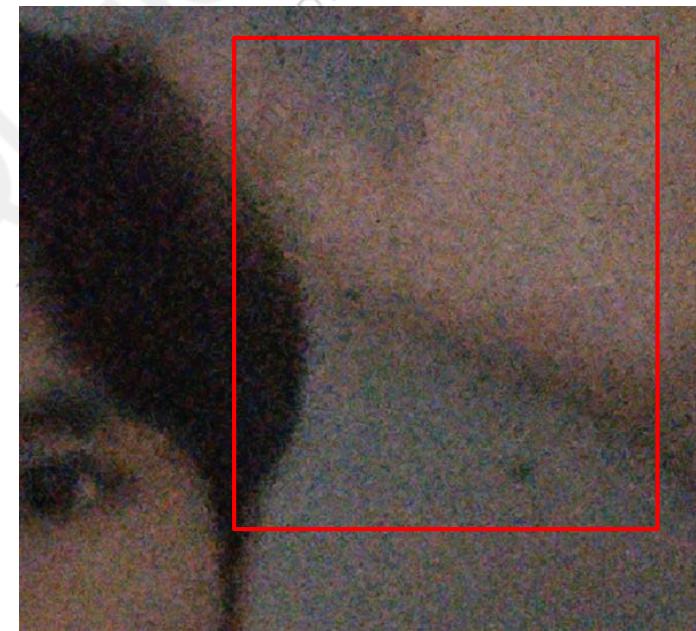
# Pre-stage Tuning

## Blending Factors

- Decrease the `constant_blending_factor` parameter to revive details and to match noise balance between stationary and moving objects.
- Follow the rule where  $\text{constant\_blending\_factor} + \text{a2\_max} = 255$



*constant\_blending\_factor = 255*



*constant\_blending\_factor = 150*

# Post-stage Tuning

---

## HNR and ANR Tuning

Tune HNR and ANR several times.

1. First, tune HNR (use calibrated ANR parameters).
  2. Tune ANR.
  3. Complete fine tuning for HNR and ANR to not lose high-frequency details.
- 
- Main parameters
    - HNR
      - filtering\_nr\_gain\_arr, lnr\_gain\_arr, blend\_lnr\_gain\_arr, snr\_gain\_arr
    - ANR
      - Base functions for luma and chroma (DC16 → DC4 → Full)
      - False colors

# Post-stage Tuning

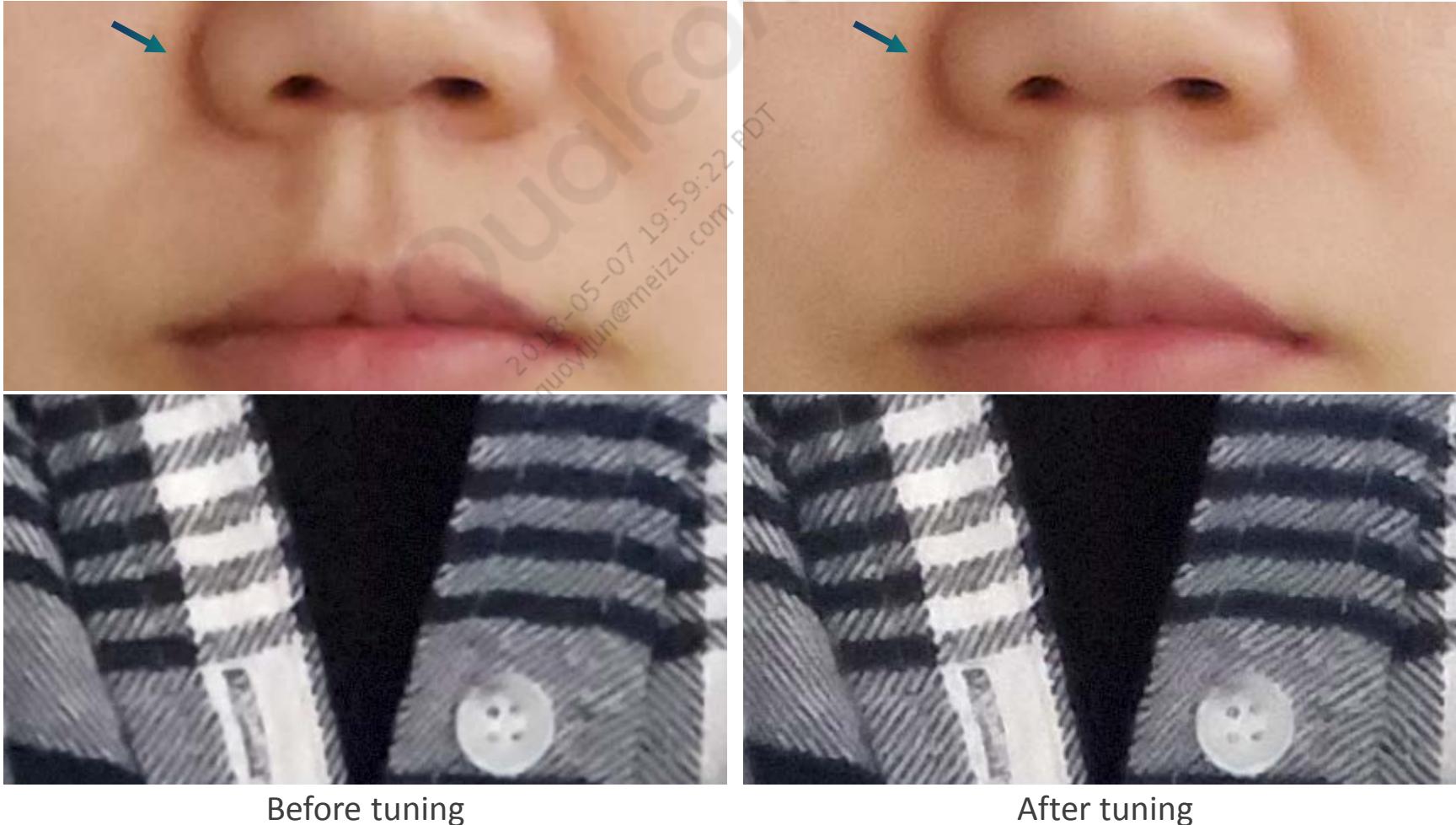
---

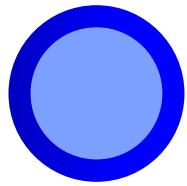
- Tuning factors
  - Reduce low-frequency noise
  - Keep details
  - Remove noise along edges
- HNR
  - Increase blend\_lnr\_gain\_arr\_tab (ex, 80 from 32)
  - Tune fliltering\_nr\_gain\_arr
- ANR
  - Increase anr\_final\_blender\_luma\_min\_strength\_low\_freq (e.g., 256 from 200)
  - Decrease anr\_final\_blender\_luma\_min\_strength\_high\_freq (e.g., 180 from 256)
  - Decrease y\_threshold\_per\_y of ANR full Y
  - Increase y\_threshold\_close3\_mod\_scale of ANR DC4 Y (e.g., 60 from 20)
  - Decrease dcblend2\_luma\_strength\_function\_tab (e.g., 0 41 82 82 82 or 0 50 100 100 100 from 0 20 40 60 82)

2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Post-stage Tuning

- Smooth transition around shadow
- Keep details



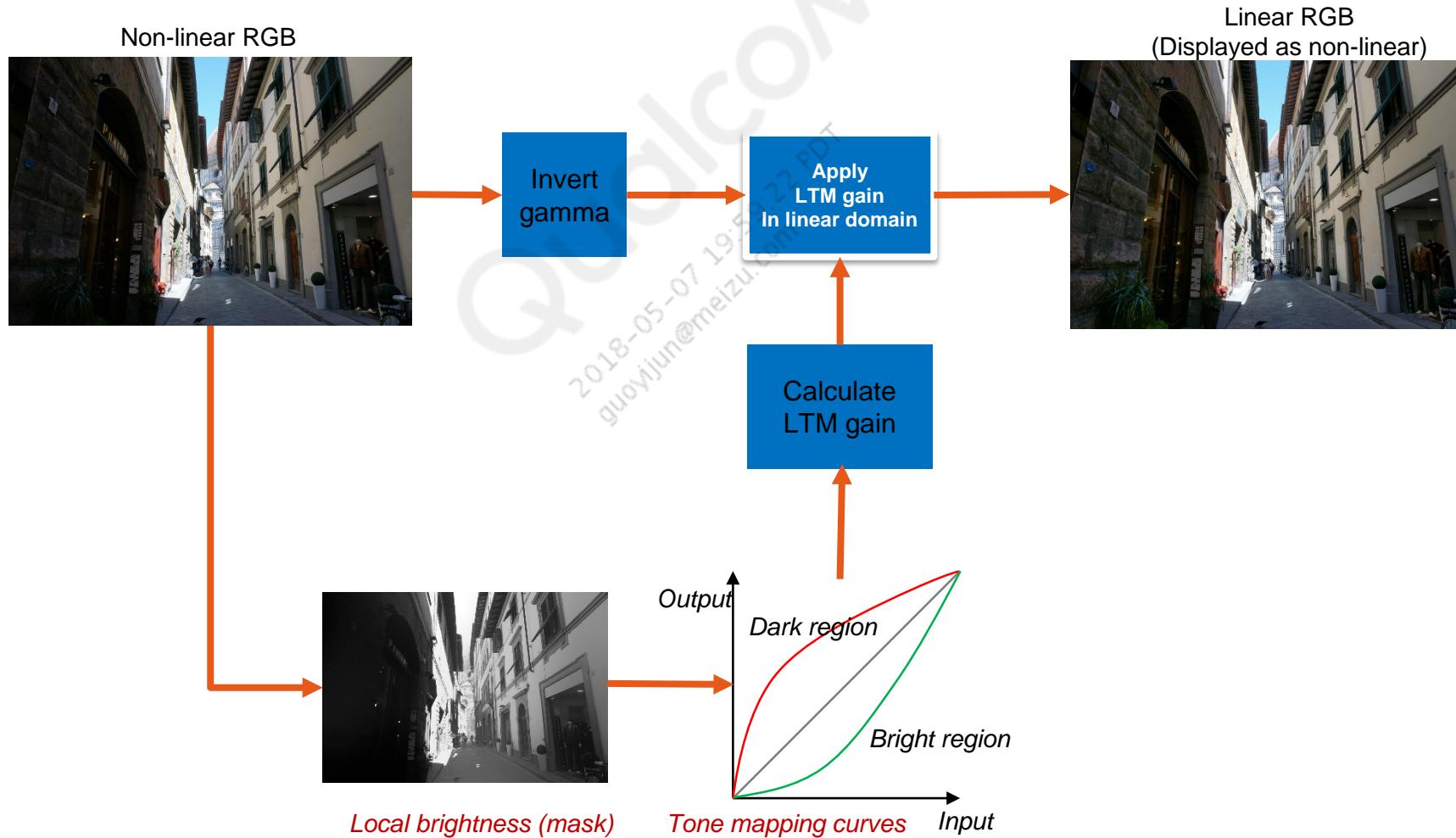


# Local Tone Mapping (LTM)

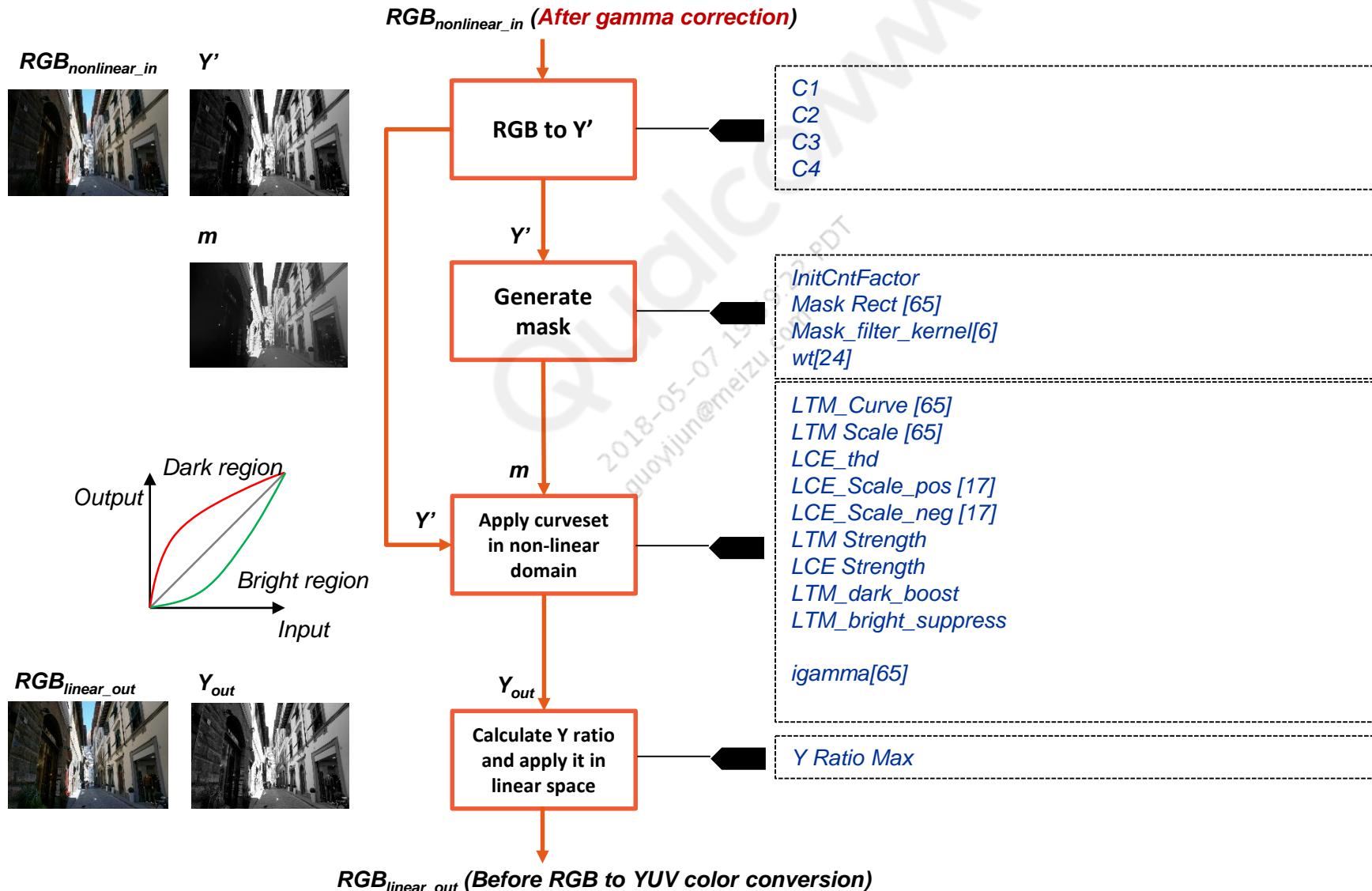
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# What is LTM

- LTM applies tone mapping gain to each pixel depending on the local brightness to enhance visibility.



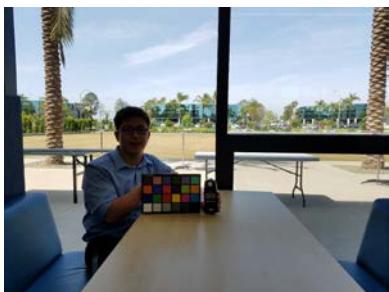
# LTM Processing Overview



# HDR: GTM/LTM Tuning Image Capture

- Capture HDR raw images, covering 6 exposure lux index zones and exposure ratios from 1x to 16x
- For each scene, capture raw of HDR mode and non-HDR mode
- HDR scenes image content suggestions:
  1. Window scene: Half bright outdoor, half dark indoor, containing a person holding MCC chart by the window, bright outdoor scene through the window.
  2. Shade scene: Outdoor scene with half bright sun and half dark shade.
  3. Outdoor texture: A person standing under direct sunlight, holding a texture test chart. There are dark shade on the background.
  4. Backlit scene: A person holding MCC chart under shade with bright background, for example sky with clouds.

Lux index	Exp ratio
100 - 200	1 – 2
200 – 250	2 – 4
250 - 300	4 – 6
300 – 380	6 – 8
380 – 450	8 – 12
450 - 600	12 - 16



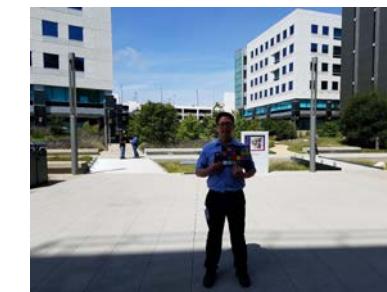
1. Window scene



2. Shade scene



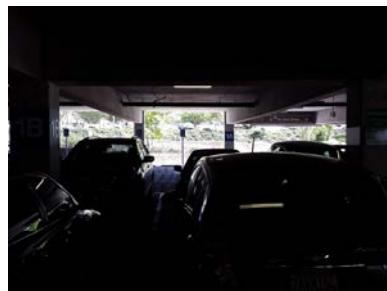
3. Outdoor texture



4. Backlit scene

# HDR: GTM/LTM Tuning Image Capture

5. Garage scene: Inside the parking building with bright outside background.
6. Building scene: Buildings with dark shadows.
7. Night scene: Building with lights at night.
8. Indoor scene: Indoor decoration with small outdoor areas
9. Nature scene: Natural objects including trees, grass, water and buildings, containing highlights and shadows.
10. Resolution chart in lab scene. The resolution chart are placed half in the bright side and half in the dark side.
11. Night scene with people face, shop windows/lights. It would be best if the person can swing the hands or so on to generate some motion.



5. Garage scene



6. Building scene



7. Night scene



8. Indoor scene



9. Nature scene

# Overview of LTM Tuning

---

- Set all parameters to default values
- For each region, tune *LTM\_dark\_boost*, *LTM\_bright\_suppress*, *LTM\_strength*, and *LCE\_strength*
  - If users need more dark boost, increase *LTM\_dark\_boost*
  - If users need more highlight suppression, increase *LTM\_bright\_suppress*
  - If users need stronger LTM effect, increase *LTM\_strength*
  - If users need stronger LCE effect, increase *LCE\_strength*
  - Users should check if the noise level is acceptable as increasing dark boost

Parameter name	Range	Default value	Larger value means
LTM_dark_boost	[0.0, 4.0]	1.0	More dark boost
LTM_bright_suppress	[0.0, 4.0]	1.0	More highlight suppression
LTM_strength	[0.0, 4.0]	1.0-1.5 for bright 1.0 for normal <0.5 for lowlight>	Stronger LTM effect
LCE_strength	[0.0, 4.0]	1.0	Stronger local contrast

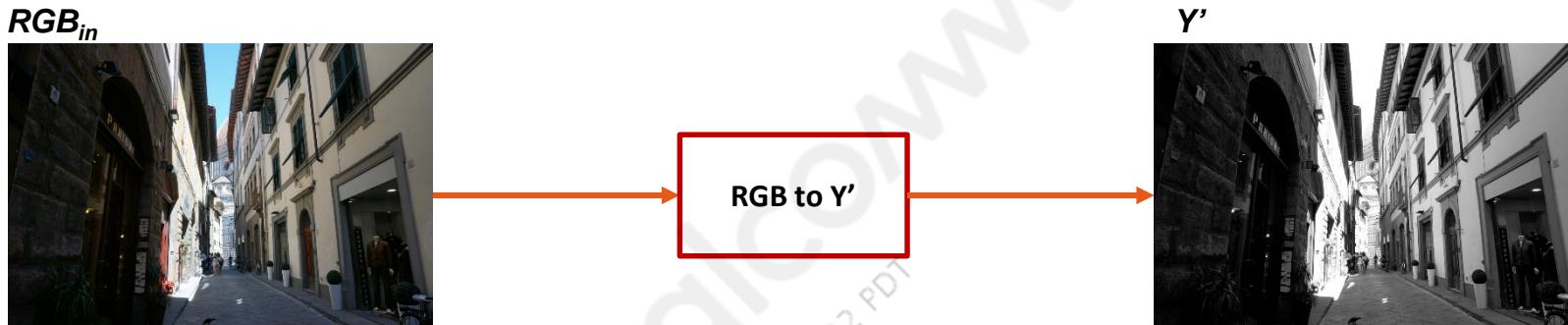
## Overview of LTM Tuning (cont.)

---

- If users need to fine tune specific range of tone, tune *LTM\_curve* and *LTM\_scale*
- If users need to fine tune local contrast, tune *LCE\_scale\_pos* and *LCE\_scale\_neg* respectively for positive and negative contrast
- If there is exposure oscillation issue, tune *Local Tone Strength* and *Local Tone Contrast*

Qualcomm  
2018-05-07 19:59:22  
guojun@meizu.com

# LTM Mask Creating: RGB to Y'



$$Y' = C1 \times R_{in} + C2 \times G_{in} + C3 \times B_{in} + C4 \times \max(R_{in}, G_{in}, B_{in})$$

- C1,C2,C3,C4
  - Description: weights of R/G/B/MAX(R,G,B) contribute to Y
  - Bit depth: Float
  - Length: 1
  - Default : 0.125,0.25,0.125,0.5
  - Min and Max: 0, 1.0f
  - Conversion: Q6

#### Note:

- ( $C1 + C2 + C3 + C4$ ) must be between 1.0 and 1.1 to keep  $Y'$  similar to Y
- It is suggested to use following parameters:  $C1=C3=0.125, C2=0.25, C3=0.5$
- Decrease  $C1, C2, C3$  and increase  $C4$ >increase  $Y'$ >increase suppression in color saturated regions
- The effect of these parameters is more significant in darker regions.

# LTM Mask Creating: Effect of C4

**Figure 1**

$(C_1, C_2, C_3, C_4) = (0.299, 0.587, 0.114, 0)$

**Figure 2**

$(C_1, C_2, C_3, C_4) = (0.125, 0.250, 0.125, 0.5)$



More suppression compared with Figure 1

**Note:**

- Decrease  $C_1, C_2, C_3$  and increase  $C_4$  if the difference between maximum and minimum of  $R_{in}, G_{in}, B_{in}$  is big enough, and then increase  $m$  (mask)
- According to *LTM\_Curve* and *LTM\_Scale* settings, increase  $m$  more suppression

# LTM Mask Creating: Generate Mask

## Collect data

$Y'$



1. Collect regional statistics inside hardware
2. Processed local statistics can be used to generate local density representation
3. Use  $2^{InitCntFactor}$  as initial pixel count for each region
4. Create mask value on fly for each pixel



## Generate mask



Apply mapping curve **Mask Rect** on each grid



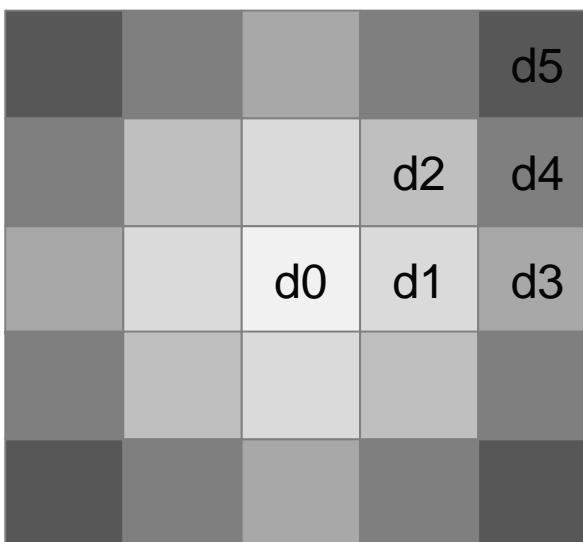
### Note:

- Decrease left side of **Mask Rect** > increase dark boost
- Increase right side of **Mask Rect** > increase highlight suppression
- In suggested use case, Mask Rect curve does not change from frame to frame

# LTM Mask Creating : Mask Filter Kernel

- n mean  $2^{n-1}$ , when n=0, it means 0
- 5 core coefficients that will be replicated to fill  $5 \times 5$  matrix  
Min and Max: d0: 1-6; d1: 0-5; d2: 0-4; d3: 0-3; d4: 0-2; d5: 0-1  
Default value: 5,3,3,2,2,1
- Reducing the center coefficient helps to smooth mask and increase micro-halo along edges. It reduces potential contrast reverse issue and improve boundary of boosted thin area.

Mask Filter Kernel



Example:

$$\begin{aligned} \text{sum} &= 2^{(d0-1)} * 1 + 2^{(d1-1)} * 4 + 2^{(d2-1)} * 4 \\ &\quad + 2^{(d3-1)} * 4 + 2^{(d4-1)} * 8 + 2^{(d5-1)} * 4 \\ &= 2^{(5-1)} * 1 + 2^{(3-1)} * 4 + 2^{(3-1)} * 4 \\ &\quad + 2^{(2-1)} * 4 + 2^{(2-1)} * 8 + 2^{(1-1)} * 4 \\ &= 76 \end{aligned}$$

$$\text{Mask\_filter\_scale} = (2^{16}) / \text{sum} = 862$$

$$\text{mask\_filter\_shift} = 16$$

# Use Mask Filter to Improve Boundary of Boosted Thin Area

Before improvement

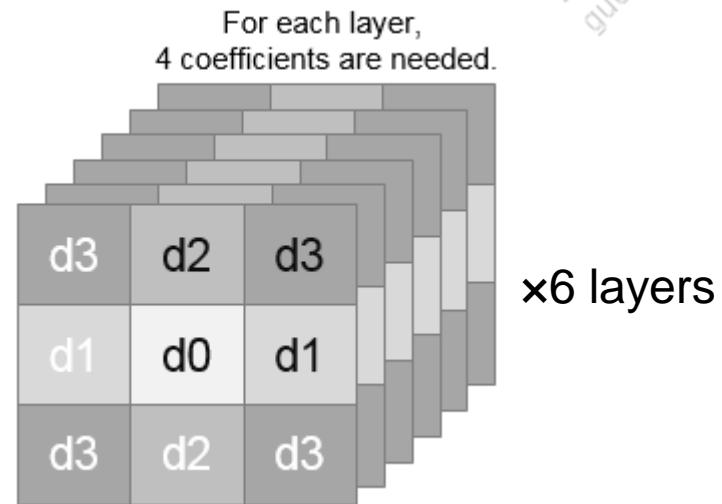


After improvement



# LTM Mask Creating: Wt [24]

- This is part of advanced user mode. Also, with mask filtering feature, it is less needed. It is recommended to use default setting.
- 3D kernel has  $3 \times 3 \times 11(\text{layer}) = 99$  coefficients
- Considering symmetric across layers and at each layer, 4 coefficients are needed on each layer and totally 6 layers are needed. So, totally 24 coefficients need to be defined. Data range 0 -255.
- Increasing neighboring coefficients ( $d_1, d_2, d_3$ ) increases smoothing of center region's statistics with neighboring regions.
- Increasing outer layer coefficients reduces the barrier to generate mask across different intensities. On the contrary, it may introduce halo.



```
Word16u weight[24] = {  
    255, 179, 179, 125,  
    128, 90, 90, 63,  
    64, 45, 45, 31,  
    16, 11, 11, 8,  
    2, 2, 2, 1,  
    0, 0, 0, 0  
}; // 8u,  
24=2x2x6
```

# LTM Mask Creating: Other Parameters and LUTs

---

- bin\_init\_cnt
  - Description: stat bin initialization value, n means  $2^n$
  - Length: 1
  - Default: –
  - Min and Max: 0 to 8
- mask\_rect\_curve[65]
  - Description: mask rectification curve
  - Bit depth: 12u
  - Length: 65

Note: Recommend not to adjust

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

## Curve Set: LTM\_curve/LTM\_scale/LCE\_scale\_pos/LCE\_scale\_neg

---

ltm\_curve[65]

- Description: Local tone map curve
- Bit depth: 12u
- Length: 65
- Default: –
- Min and Max: 0 to 4095

ltm\_scale[65]

- Description: Local tone map scale curve
- Bit depth: 12s
- Length: 65
- Default: –

Min and Max: -2048 to 2047

lce\_scale\_pos/neg[17]

- Description: Local contrast enhancement curve for positive/negative portion
- Bit depth: 12s (use positive side)
- Length: 2×17
- Default: –
- Min and Max: 0 to 2047

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# LCE Curves (65 LUT)

---

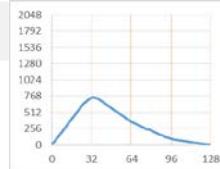
lce\_scale\_pos/neg[17]

- Description: Local contrast enhancement curve for positive/negative portion
- Bit depth: 12s
- Length: 2x17
- Default: –
- Min and max: 0 to 2047

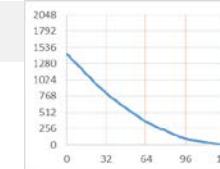
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Effect of LCE Scale

0 – 804 – 356 – 88 – 0



1432 – 804 – 356 – 88 – 0



## Note:

- LTM could control tone mapping and local tone enhancement strength separately
- In indoor/dark scenes, lower **LCE Scale** value at the beginning of LUT could reduce local tone enhancement effect in dark regions, and therefore keep luma noise low.

# LA Curve

---

la\_curve[ 65 ]

- Description: Luma adaptation curve
- Bit depth: 12u
- Length: 65
- Default: –
- Min and max: 0 to 4095

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

## LA Curve (cont.)

- Spectra LTM has a LA curve that can be applied to non-linear Y'. It is a global tone operation.
- Currently, as Spectra ISP pipeline has GTM module, it is not needed in normal use case (including ADRC and zzHDR).
- For low light scenes, LA curve can be activated depending on the histogram analysis. It can serve as a supplemental option to AEC.

Original



Only LTM, LA\_curve is off



LTM on, LA\_curve is on



# Software Aided Parameters

---

- LTM\_strength
  - Description: Overall scaling factor that will be applied to ltm\_scale curve
  - Bit depth: Float
  - Length: 1
  - Default: –
  - Min and max: 0.0 to 4.0
- LCE\_strength
  - Description: Overall scaling factor that will be applied to lce\_scale curve
  - Bit depth: Float
  - Length: 1
  - Default: –
  - Min and max: 0.0 to 4.0

**Note:**

- $LTM\_Scale = LTM\_Scale \times LTM\_Strength / 256$
- $LCE\_Scale = LCE\_Scale \times LCE\_Strength / 256$

# Effect of LTM\_Strength

---

LTM\_Strength – 128 (0.5x)



LTM\_Strength – 256 (1x)



LTM\_Strength – 512 (2x)



# Effect of LCE\_Strength



# Software Aided Parameters

---

- *LTM\_dark\_boost*
  - Update lower portion of *LTM\_scale* to control the dark boost of LTM
  - Data range – [0.0, 4.0]
  - Default value – 1.0
  - To tune:
    - Increase to boost dark region.
- *LTM\_bright\_suppress*
  - Update higher portion of *LTM\_scale* to control the highlight suppression of LTM
  - Data range – [0.0, 4.0]
  - Default value – 1.0
  - To tune:
    - Increase to suppress highlight region.

# Apply Curves – Software Aided Parameter

1. Dark boost (*LTM\_dark\_boost*) and highlight suppression

*LTM\_bright\_suppress* on *LTM\_Scale*

[Region A]

$$LTM\_Scale = LTM\_Scale \times LTM\_dark\_boost$$

[Region B]

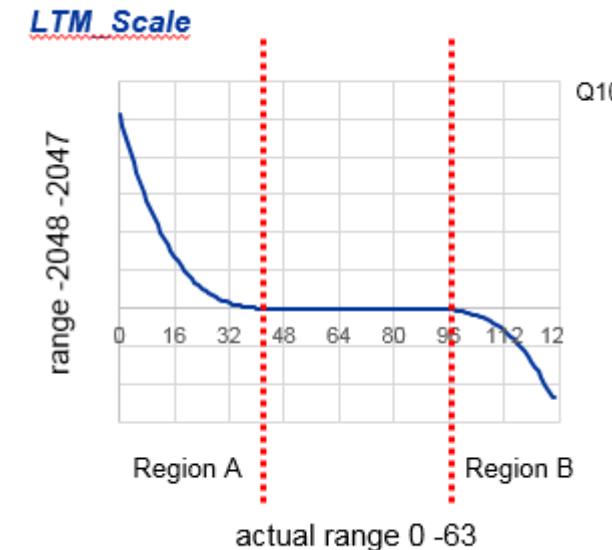
$$LTM\_Scale = LTM\_Scale \times LTM\_bright\_suppress$$

2. *LTM strength* on *LTM\_Scale* and *LCE\_strength* on

*LCE\_scale*

$$LTM\_Scale = LTM\_Scale \times LTM\_Strength / 256$$

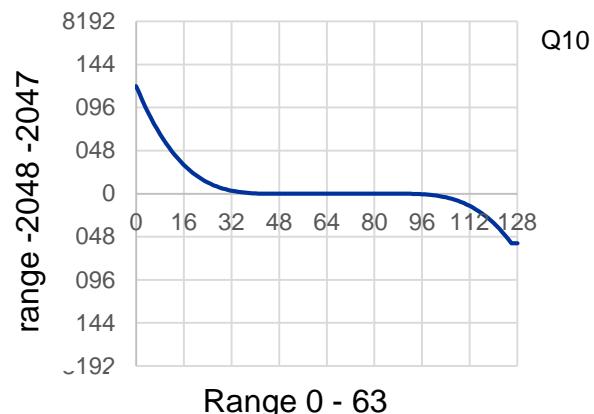
$$LCE\_Scale = LCE\_Scale \times LCE\_Strength / 256$$



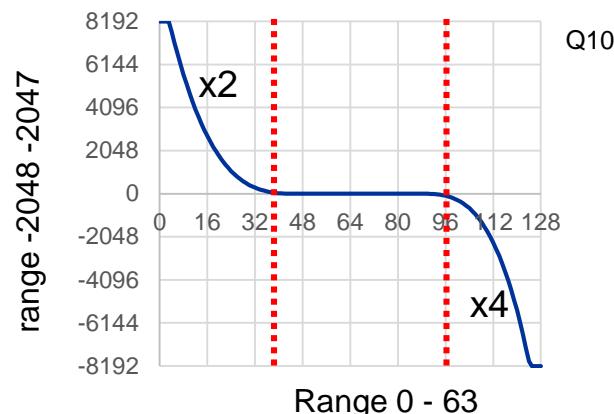
# Effect of LTM\_dark\_boost and LTM\_bright\_suppress



LTM\_dark\_boost: 1.0 LTM\_bright\_suppress: 1.0



LTM\_dark\_boost: 2.0 LTM\_bright\_suppress: 4.0

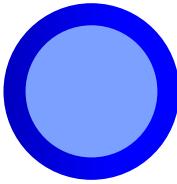


# Y Ratio Maximum

---

- Y ratio maximum
  - Control the maximum gain applied on a pixel
  - Data range – [0, 16.0]
  - Default value – 16.0
  - To tune:
    - It is recommended to retain the default value

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com



# 2D LUT

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# 2D LUT – Background

---

- Goals of color tuning:
  - Render colors accurately, pass color fidelity tests
  - Render visually pleasant colors (for example, blue sky, green grass)
- Common color tuning modules:
  - Color correction (CC): For overall color optimization
  - Color conversion (CV): For saturation adjustment
  - Skin color enhancement (SCE): For fine tuning of skin tone (or another color)
  - Memory color enhancement (MCE): For increasing saturation of red, green or blue colors
- Limitations of common modules:
  - Tuning one color affects other colors
  - Enhance chroma without considering luma values (may produce obtrusive enhancement)
  - Some modules (SCE, MCE) are not flexible, hard to tune

# 2D LUT – Overview

- Use 2D LUT module to fine tune a specific color without affecting other colors
- 2D LUT specifics:
  - Located in PPS, at the end of RGB domain
  - Pipeline order: CC>gamma>2D LUT>CV>Chroma suppression>SCE
  - Module in/out are RGB values, but use HSL (hue, saturation, lightness) color space inside module
  - Enhance specific colors without affecting others
  - Can adjust hue and saturation based on L (lightness) value
  - Can limit the Y (luma) value changes due to enhancement
- Advantages
  - Local color adjustment on the hue-saturation plane
  - Very easy and intuitive color adjustment



# Tuning Procedure Overview

---

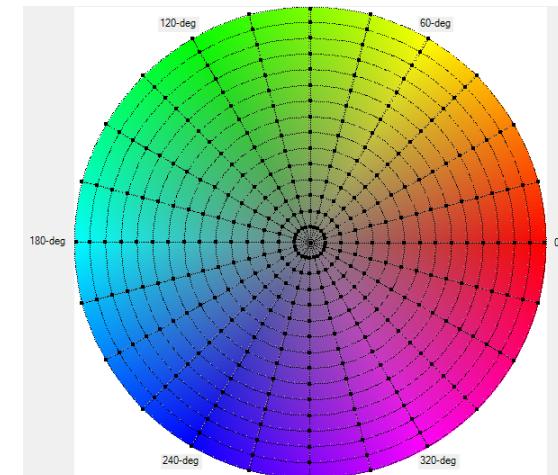
1. Tune CC and gamma before tuning 2D LUT
2. Set 1D knee points in `lut_1d_h[25]`, `lut_1d_s[16]`, based on the colors that need adjustment
3. For each of the colors that need adjustment
  - a) Get the HSL values of current and target colors
  - b) Set 2D **delta** hue and saturation LUTs `lut_2d_h[24][16]`, `lut_2d_s[24][16]` accordingly
  - c) Make sure delta LUTs are smooth and within range
4. Set L boundary points (`l_boundary_start_a`, `l_boundary_start_b`, `l_boundary_end_a`, `l_boundary_end_b`), if needed, so that the color adjustment is only for the defined brightness range
5. Set `Y_blend_factor` based on the preference of luma preservation vs color enhancement

# Step 1. Set 1D Knee Points

Based on the colors that need adjustment, set 1D knee points in `lut_1d_h[25]`, `lut_1d_s[16]` (reserved parameters, rarely tune)

## Related parameters

- `lut_1d_h[25]`, `lut_1d_s[16]` define the sampling in the hue/saturation plane, that is, which hue/saturation to adjust (see the hue/saturation color wheel below)
- `lut_1d_h[25]` represents hue angle
  - Default value: [0,15,30,45,60,75,90,105,120,135,150,165,180,195,210,225,240,255, 270,285, 300,315,330,345,360], that is, evenly spaced from 0-360 degree
  - Must have  $\text{Lut\_1d\_h}[0] = 0$  degree,  $\text{Lut\_1d\_h}[24] = 360$  degree
  - Must have: LUT are always programmed in ascending order, and no two entries can be equal
- `lut_1d_s[16]` represent saturation
  - Default value: [0.0625,0.125,0.1875,0.25,0.3125,0.375, 0.4375,0.5,0.5625,0.625,0.6875,0.75,0.8125,0.875, 0.9375,1.0], evenly spaced
  - Must have: LUT are always programmed in ascending order, and no two entries can be equal
  - Must have  $\text{Lut\_1d\_s}[15] = 1.0$



# Step 1. Set 1D Knee Points (cont.)

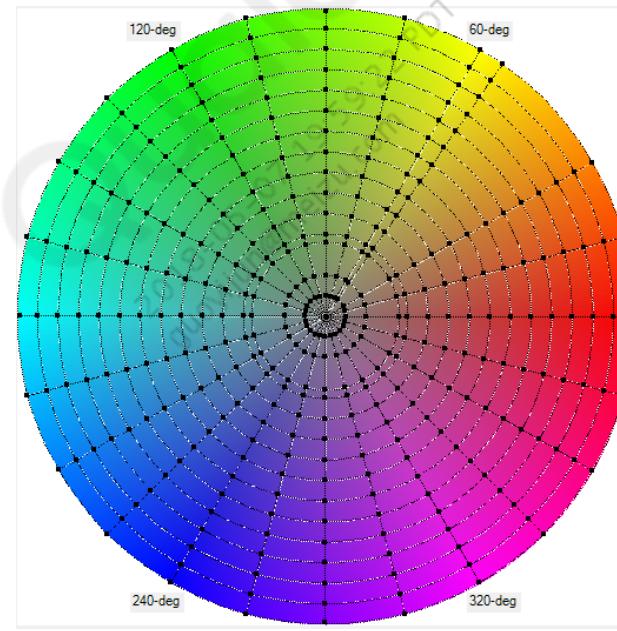
How to tune

- Users can edit lut\_1d\_h[25], lut\_1d\_s[16] in tool GUI (see the below left image)
- The updated 1D LUTs are reflected on the sampling points of color wheel

The screenshot shows the LUT Maker software interface. It has tabs at the top: Main, 1D LUT, Saturation LUT, Hue LUT, and Simulation. The 1D LUT tab is selected. There are two tables: 'Hue LUT' and 'Saturation LUT'. The 'Hue LUT' table has columns 'Index' and 'Hue (degree)'. The 'Saturation LUT' table has columns 'Index' and 'Saturation'. In both tables, the 3rd row is highlighted with a blue background.

Index	Hue (degree)
0	0
1	15
2	30
3	55
4	60
5	75
6	90
7	105
8	120
9	135
10	150
11	165
12	180
13	195
14	210
15	225
16	240
17	255
18	270
19	285
20	300
21	315
22	330
23	345

Index	Saturation
0	0.000
1	0.067
2	0.133
3	0.240
4	0.267
5	0.333
6	0.400
7	0.467
8	0.533
9	0.600
10	0.667
11	0.733
12	0.800
13	0.867
14	0.933
15	1.000

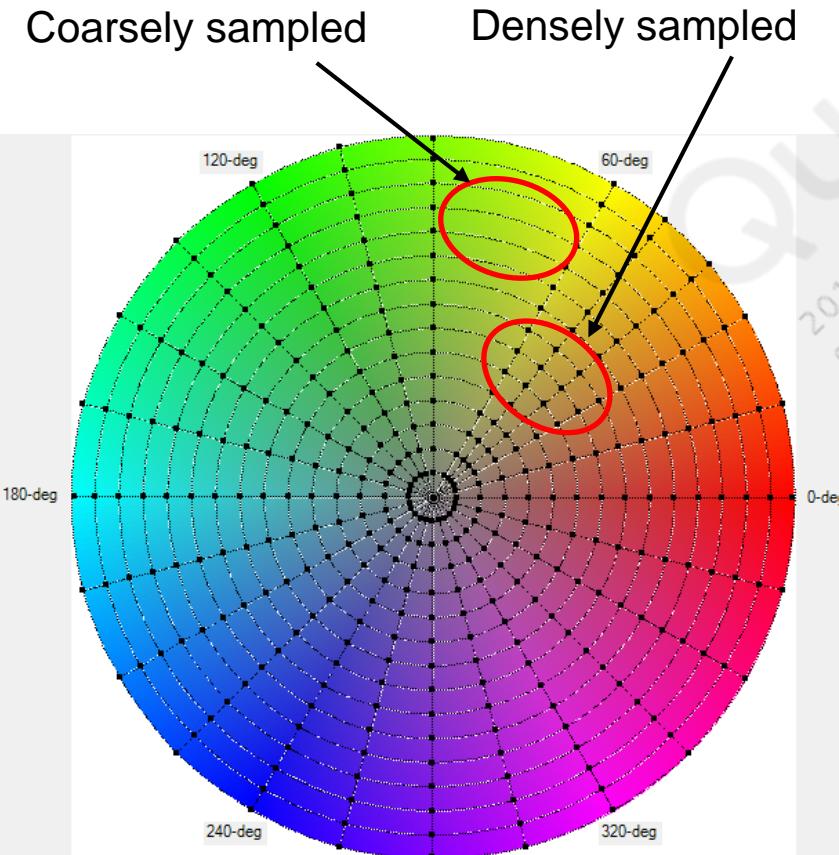


**Note:** Users can assign more sampling points for colors that need adjustment, like these 1D knee points don't have to be evenly spaced. 1D LUTs should be set before tuning 2D LUTs. Once tuned, 1D LUTs should not be changed when tuning 2D LUTs.

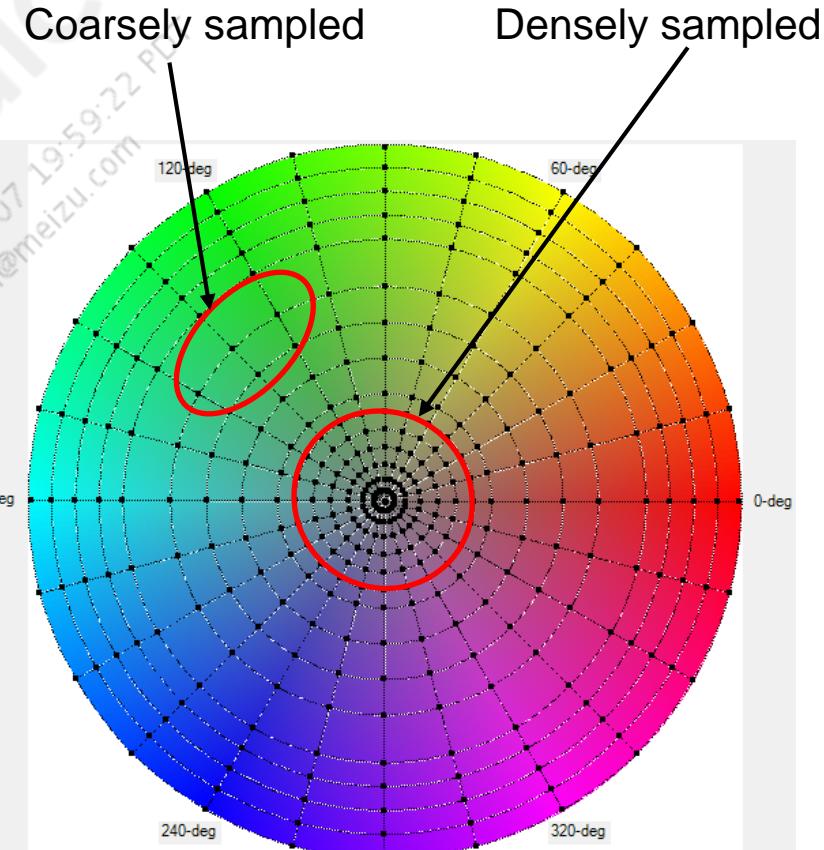
## Step 1. Set 1D Knee Points (cont.)

- Below are examples of non-uniformly sampled 1D knee points
- Users can assign more knee points at colors that need adjustment

Non-uniform 1D hue LUT



Non-uniform 1D saturation LUT



## Step 2. Set 2D Hue and Saturation LUTs

Set 2D LUTs in `lut_2d_h[24][16]`, `lut_2d_s[24][16]` – core parameters (that is, multiple regions, often tune)

Related parameters

- `lut_2d_h[24][16]`, `lut_2d_s[24][16]` represent `deltaHue` `deltaSaturation` at knee points, that is, how much hue/saturation to adjust
- Default value: All 0 entries for both LUTs

To tune:

- For each of the colors that need adjustment, users need to evaluate the difference of original color and target color, and set 2D LUTs accordingly
- There are two methods to set 2D LUTs. Users should choose from one of them. More details are given in the following slides

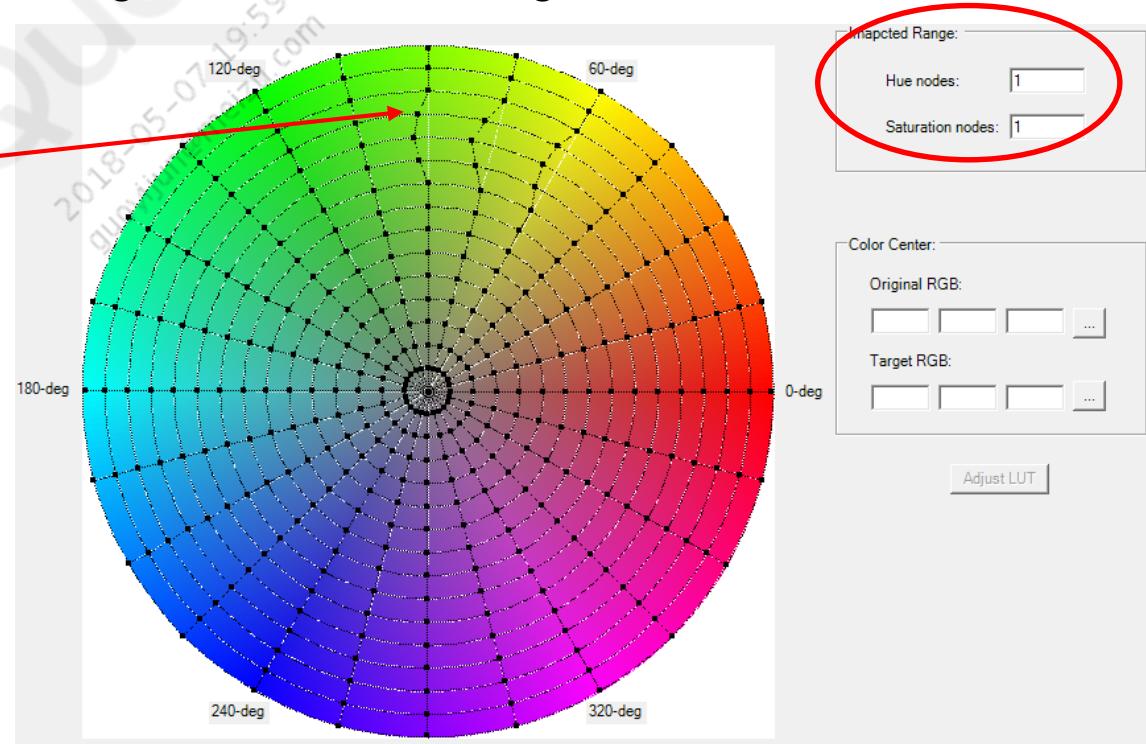
**Note:** To preserve gray area, hue and saturation should not change when saturation=0, that is, first column in `lut_2d_h[24][16]`, `lut_2d_s[24][16]` should be all 0

## Step 3. Method A: Dragging Points to Set 2D LUTs

### Steps for Method A

- At the color wheel of the tool GUI, users can drag grid points to the desired color
- Hue/saturation 2D LUTs will be updated accordingly
- Users can set Impacted Range, to control how many neighboring nodes are impacted with the adjustment. Larger adjustment requires larger impacted range, so that the changes are smooth

White grid lines denote original colors.  
Black grid lines denote target colors.



## Step 3. Method A: Dragging Points to Set 2D LUTs (cont.)

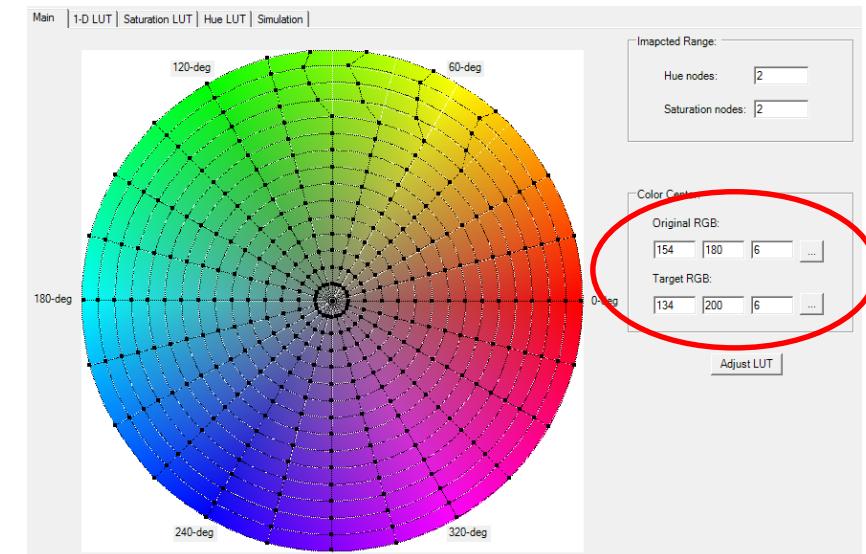
- Below is the target hue LUT generated from the color wheel from the previous slide. **Target hue** at angle 60, 75, 90 degree are updated
- By default, tool shows target hue/saturation in 2D LUTs. The Qualcomm Chromatix™ parameters `lut_2d_h[24][16]`, `lut_2d_s[24][16]` should be delta hue/saturation

Hue	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30
45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45
60	60	60	60	60	60	60	60	60	60	60	62	63	62	60	60	60
75	75	75	75	75	75	75	75	75	75	75	78	80	78	75	75	75
▶	90	90	90	90	90	90	90	90	90	90	92	93	92	90	90	90
105	105	105	105	105	105	105	105	105	105	105	105	105	105	105	105	105
120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120
135	135	135	135	135	135	135	135	135	135	135	135	135	135	135	135	135
150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150
165	165	165	165	165	165	165	165	165	165	165	165	165	165	165	165	165
180	180	180	180	180	180	180	180	180	180	180	180	180	180	180	180	180
195	195	195	195	195	195	195	195	195	195	195	195	195	195	195	195	195
210	210	210	210	210	210	210	210	210	210	210	210	210	210	210	210	210
225	225	225	225	225	225	225	225	225	225	225	225	225	225	225	225	225
240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240
255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255
270	270	270	270	270	270	270	270	270	270	270	270	270	270	270	270	270
285	285	285	285	285	285	285	285	285	285	285	285	285	285	285	285	285
300	300	300	300	300	300	300	300	300	300	300	300	300	300	300	300	300
315	315	315	315	315	315	315	315	315	315	315	315	315	315	315	315	315
330	330	330	330	330	330	330	330	330	330	330	330	330	330	330	330	330
345	345	345	345	345	345	345	345	345	345	345	345	345	345	345	345	345

## Step 3. Method B: Use Original/Target RGB to Set 2D LUTs

### Steps for Method B

- Users can input original and target RGB in text box. Alternatively, users can pick colors from sample images to get RGB values.
- Click Adjust LUT to adjust the grid points. Hue/saturation 2D LUTs will be updated accordingly
- Users can set Impacted Range, to control the number of neighbor nodes that will have an impact with the adjustment. Larger adjustment requires larger impacted range, so that the changes are smooth



## Step 3. Method B: Use Original/Target RGB to Set 2D LUTs (cont.)

- Below is the target hue LUT generated from the color wheel from the previous slide. **Target hue** at the angles of 45, 60, 75, and 90 degree are updated
- By default, tool shows target hue/saturation in 2D LUTs. The Chromatix parameters `lut_2d_h[24][16]`, `lut_2d_s[24][16]` should be delta hue/saturation

Hue	S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	S10	S11	S12	S13	S14	S15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15	15
30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30	30
45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	48	50
60	60	60	60	60	60	60	60	60	60	60	60	60	60	65	67	68
75	75	75	75	75	75	75	75	75	75	75	75	75	75	81	83	86
90	90	90	90	90	90	90	90	90	90	90	90	90	90	94	96	97
105	105	105	105	105	105	105	105	105	105	105	105	105	105	105	105	105
120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120	120
135	135	135	135	135	135	135	135	135	135	135	135	135	135	135	135	135
150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150	150
165	165	165	165	165	165	165	165	165	165	165	165	165	165	165	165	165
180	180	180	180	180	180	180	180	180	180	180	180	180	180	180	180	180
195	195	195	195	195	195	195	195	195	195	195	195	195	195	195	195	195
210	210	210	210	210	210	210	210	210	210	210	210	210	210	210	210	210
225	225	225	225	225	225	225	225	225	225	225	225	225	225	225	225	225
240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240	240
255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255	255
270	270	270	270	270	270	270	270	270	270	270	270	270	270	270	270	270
285	285	285	285	285	285	285	285	285	285	285	285	285	285	285	285	285
300	300	300	300	300	300	300	300	300	300	300	300	300	300	300	300	300
315	315	315	315	315	315	315	315	315	315	315	315	315	315	315	315	315
330	330	330	330	330	330	330	330	330	330	330	330	330	330	330	330	330
345	345	345	345	345	345	345	345	345	345	345	345	345	345	345	345	345

## Method B: Examples

---

Effects of the 2D LUTs from the previous slide. The grass is more greener with the adjustment.

Original color

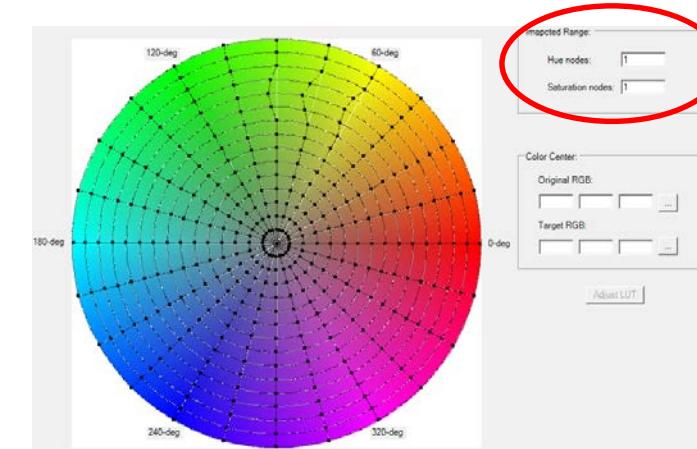


After 2D LUT enhancement



## Step 3.1 Check 2D LUTs

- 2D LUT module is used for color enhancement. It is NOT intended for a dramatic color change
- In general, we recommend delta hue in lut\_2d\_h[24][16] to be in the range of  $\pm 30$  degree, and delta saturation in lut\_2d\_s[24][16] to be in the range of  $\pm 0.20$
- Users should make sure the target nodes in the color wheel are smooth
  - Users can set Impacted Range, to control the number of neighbor nodes that will have an impact with the adjustment.  
Larger adjustment requires larger impacted range, so that the changes are smooth
- For fine tuning, users can directly modify 2D LUT data in the tool, and the nodes in the color wheel will update accordingly



# Examples of 2D LUT Color Enhancement

Blue sky is enhanced with +10 degree of hue, and +15% saturation

Original color



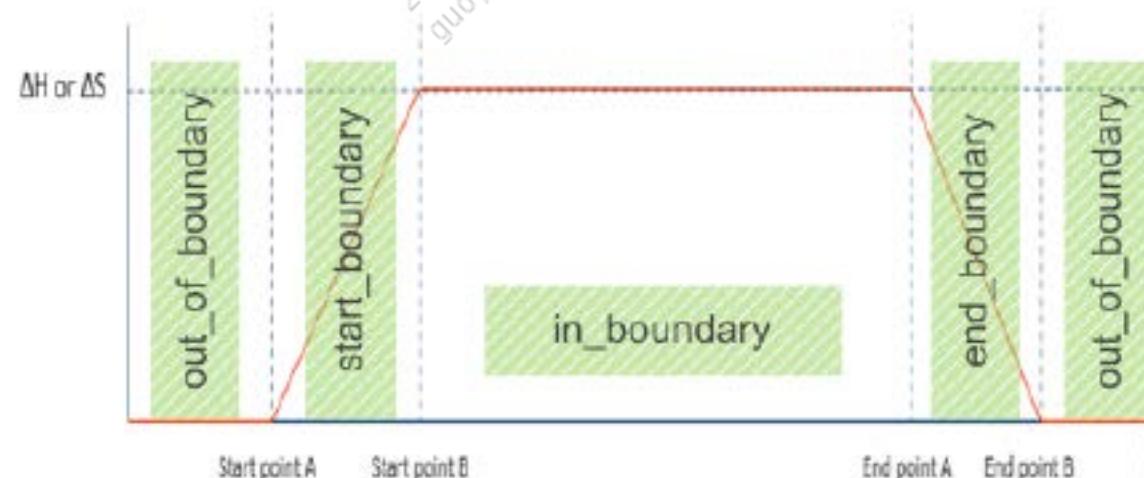
Enhanced color



## Step 4. Set L Boundary Points

If needed, set L boundary points (`l_boundary_start_a`, `l_boundary_start_b`, `l_boundary_end_a`, `l_boundary_end_b`), so that the color adjustment is only for the defined brightness range

- If in boundary, use deltaH/S from previous step (that is, from 2D LUT)
- If out of boundary, deltaH/S = 0
- If in transition area, interpolate
- All parameters are in the range [0.0, 1.0], that is percentage of Lightness range
- By default, `l_boundary_start_a` = `l_boundary_start_b` = 0, `l_boundary_end_a` = `l_boundary_end_b` = 1.0, that is, all lightness values use the full hue/saturation adjustment as defined in 2d LUTs



## Set L Boundary Points – Example

- Setting 1: Adjust only the blue color for higher L values. Blue sky is enhanced, and the ground color is kept the same
- Setting 2: Adjust blue color for all L values (that is using default L boundary points). Blue sky is enhanced, and ground color is changed

Original color



Setting 1



Setting 2



## Step 5. Set Y\_blend\_factor

Set **Y\_blend\_factor** based on the preference of luma preservation vs color enhancement

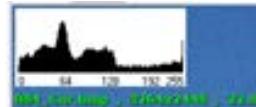
- In some cases, users may prefer to keep the luma value when enhancing color. Users can tune Y\_blend\_factor to achieve the desired results
- Y\_blend\_factor range [0, 1.0], default value 0
- When **Y\_blend\_factor** = 0, color is best expressed (recommended)
- When **Y\_blend\_factor** = 1, Y value is best preserved

## Step 5. Set Y\_blend\_factor – Example

Set `Y_blend_factor` based on the preference of luma preservation vs color enhancement

- Setting 1:  $Y_{blend\_factor} = 1.0$ .  $Y$  value is kept the same at 115
  - Setting 2:  $Y_{blend\_factor} = 0$ .  $Y$  value is changed from 115 to 119

## Original color



Y = 115

## Setting 1



$Y = 115$   
Yoffset weight = 1.0

## Setting 2

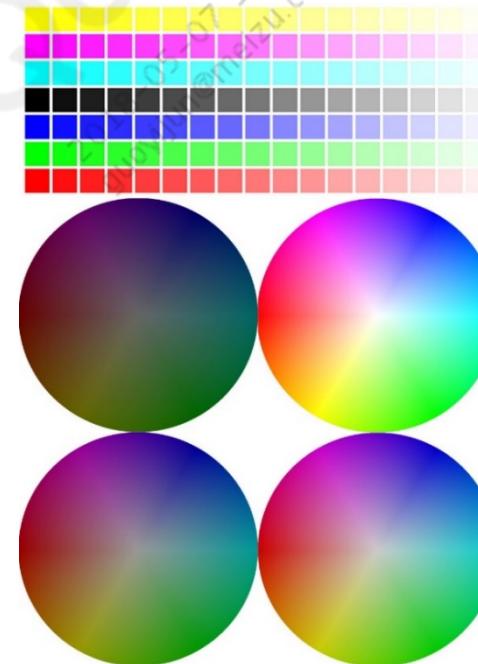


Y = 119  
Yoffset weight = 0.0

# How to Evaluate 2D LUT Tuning

---

- 2D LUT module is often used to match the colors of a reference phone
  - Typical colors that need enhancement are: Blue sky, green grass, skin colors
  - Users can use outdoor portrait images to evaluate those colors
- In addition to matching color preferences, users need to check:
  - Chroma noise is not enhanced with 2D LUT tuning, especially for low light
  - Use synthetic images like below to rule out side effects due to color tuning



# Reserved Parameters

Chromatix parameters	Type/ range in header	Default	Tuning level	Tuning notes
lut_1d_h[25]	float, [0, 360]	0,15,30,...,345, 360, evenly spaced	Rarely tune	-- Must have lut_1d_h[0] = 0, lut_1d_h[24] = 360 -- LUT are always programmed in ascending order, and no two entries can be equal.
lut_1d_s[16]	float, [0, 1.0]	0.0625,0.125, ...,0.9375,1.0	Rarely tune	-- Must have Lut_1d_s[15] = 1.0 -- LUT are always programmed in ascending order, and no two entries can be equal.
k_b_integer, k_r_integer	float, [0, 0.5]	k_b= 0.114, k_r= 0.299	Never tune	-- These parameters define RGB to Y conversion -- Users should not tune these
h_shift	uint, [0, 5]	1	Rarely tune	-- Default value means delta hue range is ±30 degree. -- Do not tune manually. This value is set dynamically by Chromatix tool. See next slide for details
s_shift	uint, [0, 3]	0	Rarely tune	-- Default value means delta saturation range is ±0.25 -- Do not tune manually. This value is set dynamically by Chromatix tool. See next slide for details

## Reserved Parameters (cont.)

---

- h\_shift: adjust delta hue range in lut\_2d\_h
  - When h\_shift=0, delta hue range is  $\pm 15$  degree
  - When h\_shift=1, delta hue range is  $\pm 30$  degree
  - When h\_shift=2, delta hue range is  $\pm 60$  degree
  - When h\_shift=3, delta hue range is  $\pm 120$  degree
  - When h\_shift=4, delta hue range is  $\pm 240$  degree
  - When h\_shift=5, delta hue range is  $\pm 360$  degree
- s\_shift: adjust delta saturation range in lut\_2d\_s
  - When s\_shift=0, delta saturation range is  $\pm 0.25$
  - When s\_shift=1, delta saturation range is  $\pm 0.5$
  - When s\_shift=2, delta saturation range is  $\pm 1.0$
- Chromatix tool will get  $\max(\text{abs}(\text{lut}_2d\_h))$  and  $\max(\text{abs}(\text{lut}_2d\_s))$ , and set h\_shift and s\_shift accordingly
  - Should calculate max value of all data regions, since h\_shift s\_shift are global parameters controlling all regions

# Core Parameters

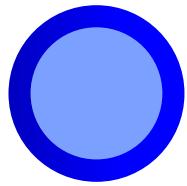
Chromatix parameters	Type/range in header	Default	Tuning level	Tuning notes
lut_2d_h[24][16]	float, [-360, 360]	0	Often tune	To preserve gray area , hue and saturation should not change when saturation=0, i.e. first column in lut_2d_h[24][16], lut_2d_s[24][16] should be all 0 .
lut_2d_s[24][16]	float, [-1.0, 1.0]	0	Often tune	To preserve gray area , hue and saturation should not change when saturation=0, i.e. first column in lut_2d_h[24][16], lut_2d_s[24][16] should be all 0
l_boundary_start_a, l_boundary_start_b, l_boundary_end_a, l_boundary_end_b	float, [0, 1.0]	start_a = start_b = 0, end_a = end_b = 1.0	Rarely tune	Can adjust hue and saturation based on L (lightness) value
y_blend_factor_integer	float, [0, 1.0]	0	Rarely tune	Can limit the Y (luma) value changes due to enhancement

# How to Coordinate with Other Modules

---

- Prior to color tuning
  - AEC (luma target) to be close to a target phone
  - AWB to be reasonably good
  - Recommend disabling chroma suppression
- Tune gamma and CCM
- Tune 2D LUT
- With 2D LUT, the following modules may not be needed for color tuning
  - Color conversion (CV): Use default
  - Memory color enhancement (MCE): Disable
  - Skin color enhancement (SCE): Disable

2018-05-21 19:59:22 PDT  
guovjun@meizu.com



# Chroma Suppression (CS)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Chroma Suppression – Background

- Chroma suppression
  - To suppress Chroma for certain Luma range
  - Color tint in highlight is removed using CS

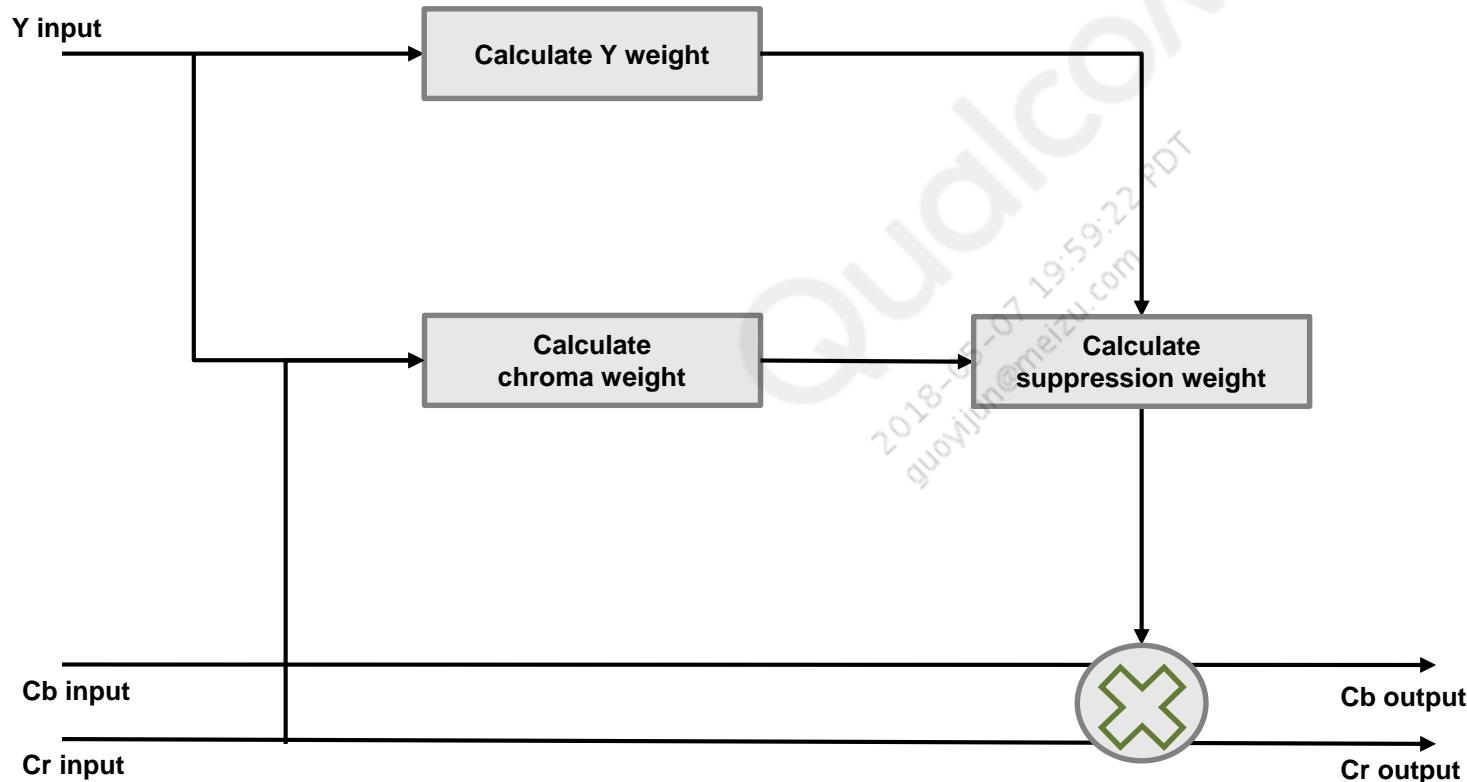


- Color artifacts in lowlight are suppressed.



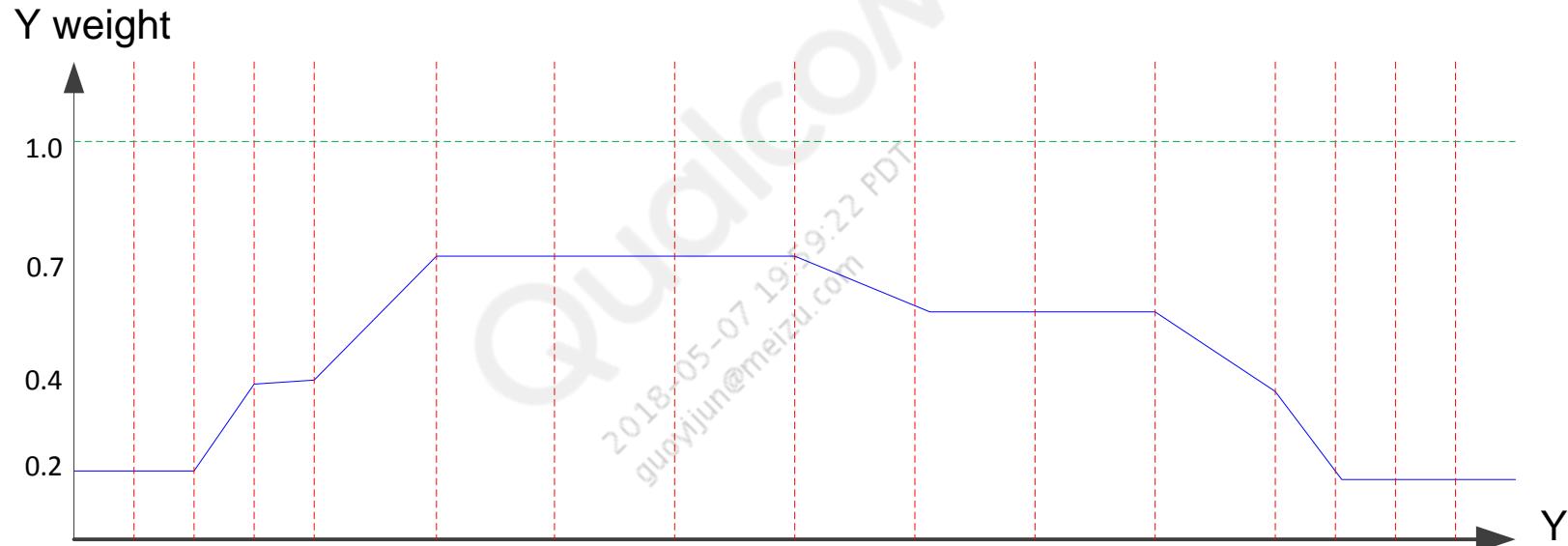
# Overview

- Block diagram



## Overview (cont.)

- Calculate Y weight

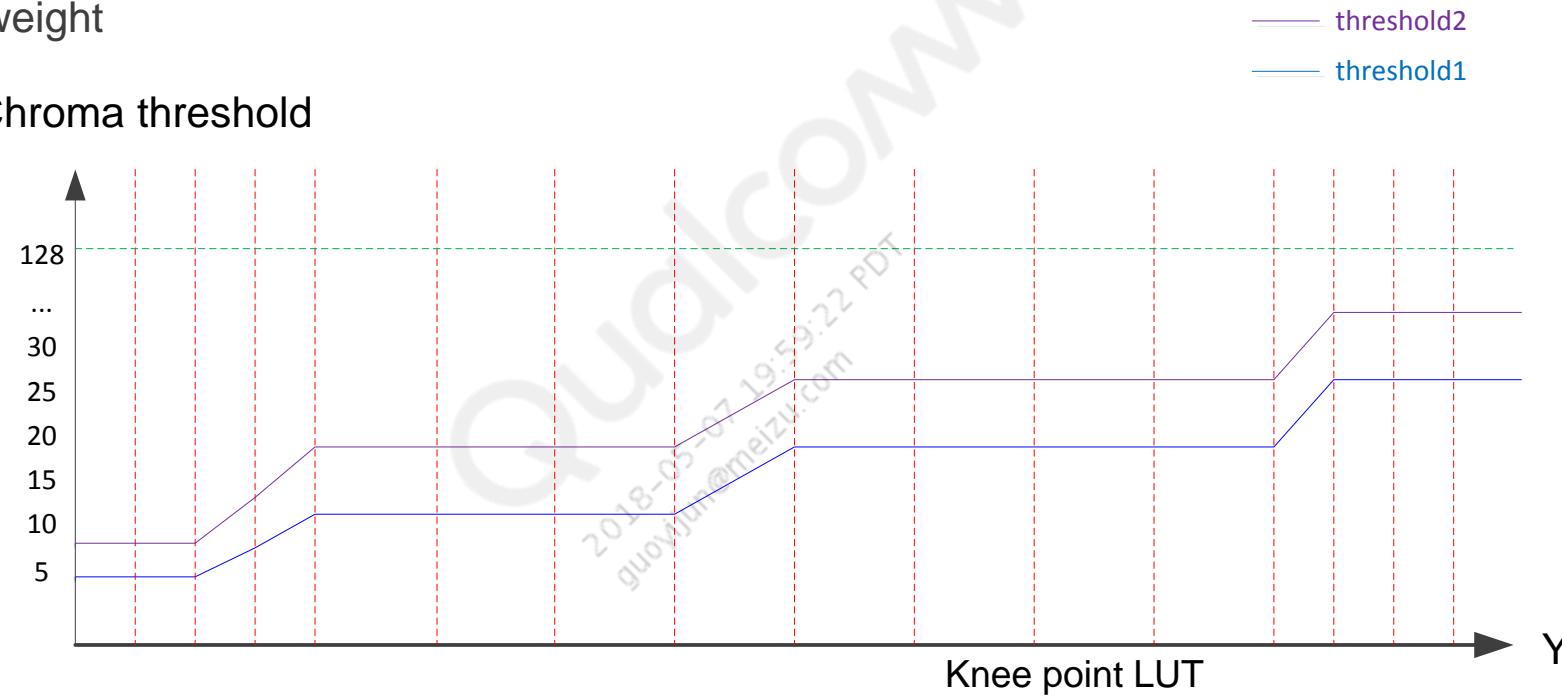


- Knee point LUT [16]
  - 1D 16 flexible LUT to control non uniformly spaced Y
- Y weight LUT [16]

## Overview (cont.)

- Calculate chroma weight

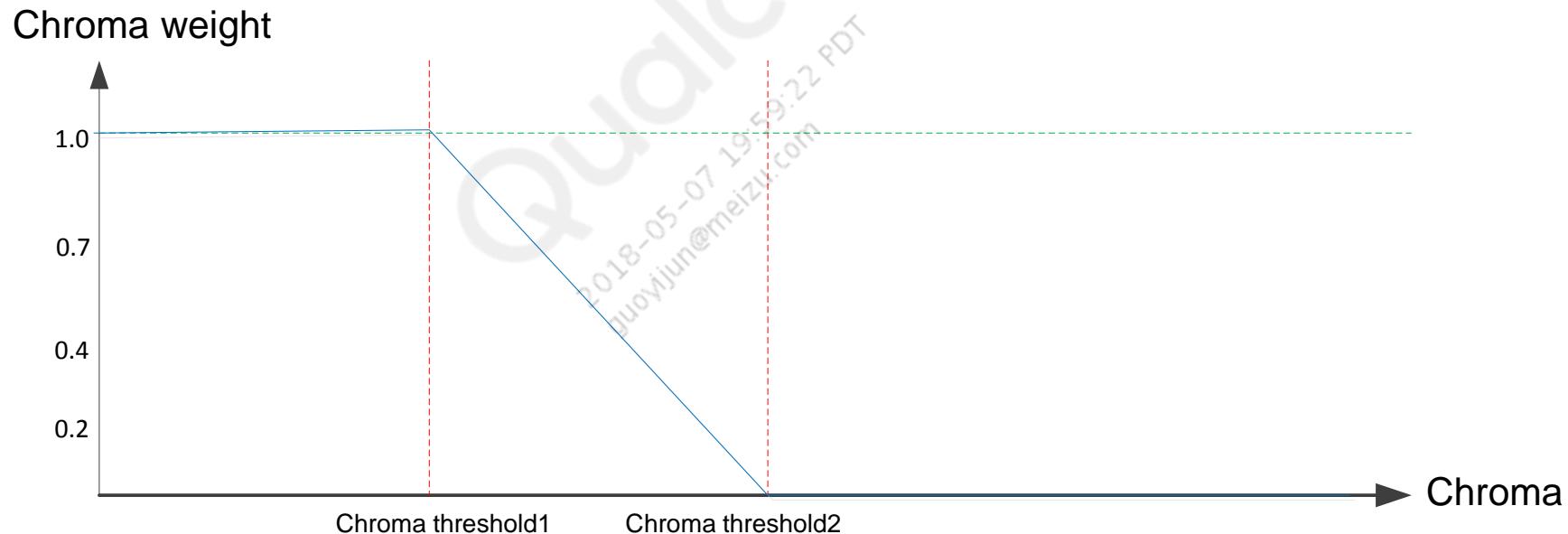
Chroma threshold



- Chroma threshold1 LUT [16]
- Chroma threshold2 LUT [16]

## Overview (cont.)

- Calculate chroma weight

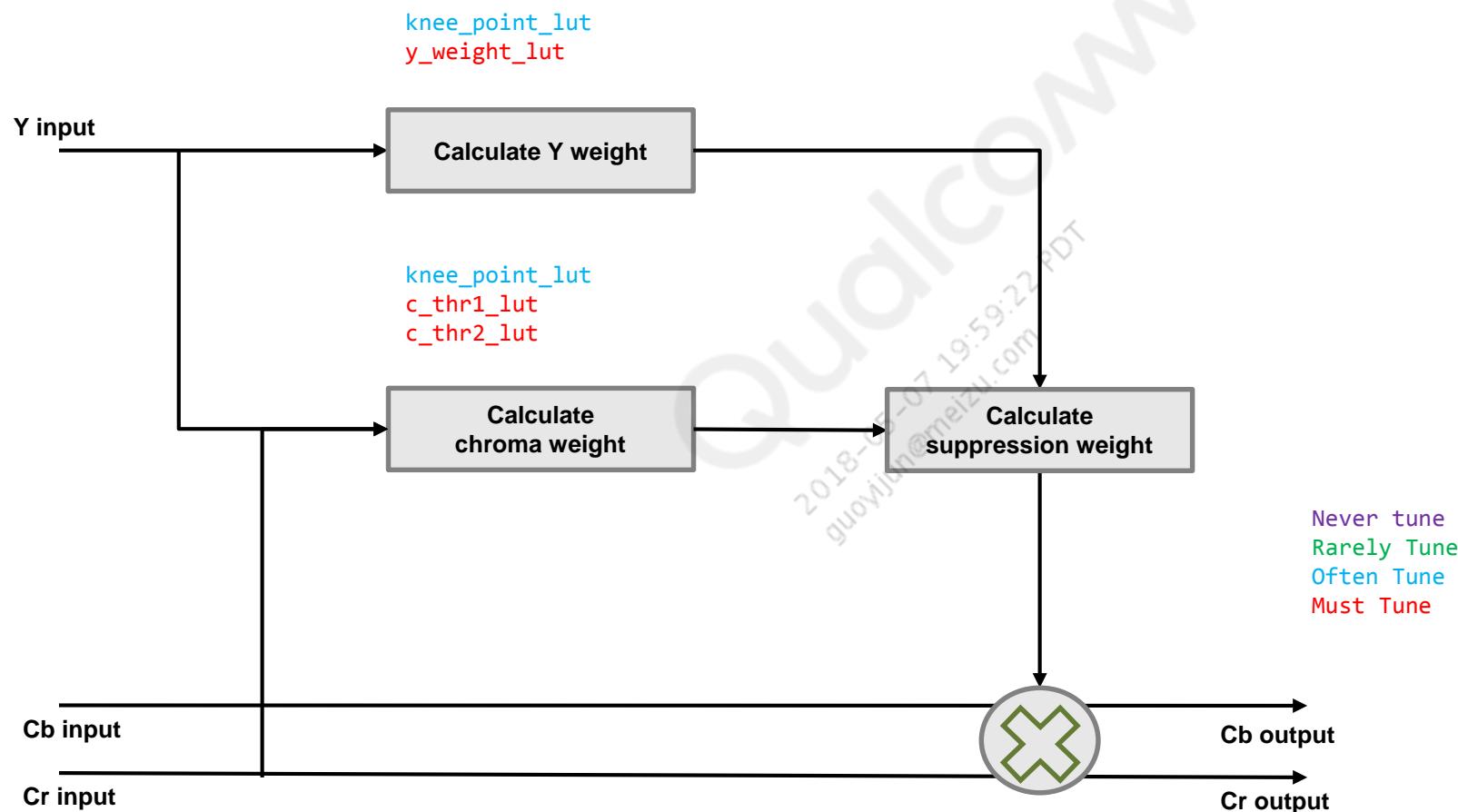


# Tuning Procedure

---

1. Set every parameter as default
2. Tune the **Must tune** and **Often tune** parameters
  1. Set knee\_point\_lut to do flexible chroma suppression based on intensity
  2. Set c\_thr1\_lut/c\_thr2\_lut to decide suppression amount based on chroma
  3. Set y\_weight\_lut to suppress less based on intensity

# Effects of Parameters



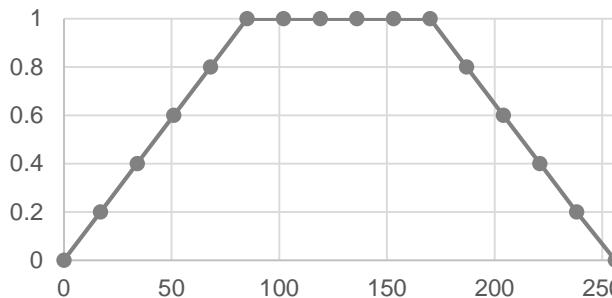
# Step 1: Knee Point LUT

- [knee\\_point\\_lut](#)
  - Define the luma knee points the users want to control

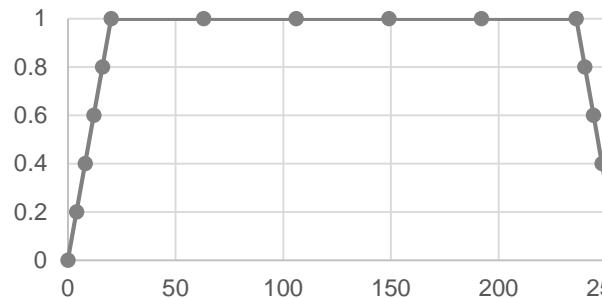
Length	Default	Min	Max	Effect
16	Evenly spaced	0	256	Define 16 knee points for both Y and chroma weight

- It applies to all 3 LUTs index (y\_weight\_lut/ c\_thr1\_lut/ c\_thr2\_lut)
- Default (last index 256 is hidden)
  - Evenly spaced
    - 0 17 34 51 68 85 102 119 136 153 170 187 204 221 238 256
  - Unevenly spaced
    - Recommend to use finer knee points for dark and highlight area to have flexible control
    - 0 4 8 12 16 20 63 106 149 192 236 240 244 248 252 256

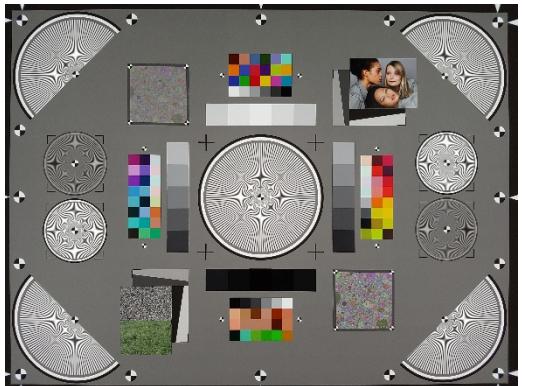
Evenly spaced knee point



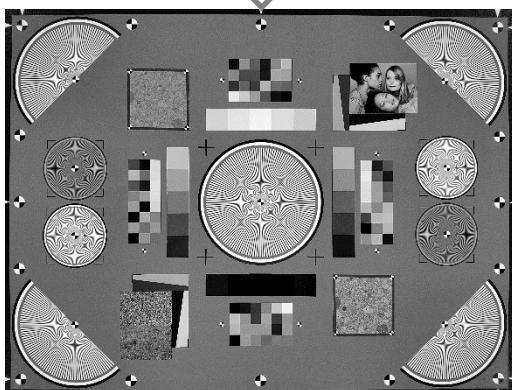
Unevenly spaced knee point



## Step 1.1: c\_thr1\_lut/c\_thr2\_lut



Strong suppression



- **c\_thr1\_lut [knee\_point]**

- Define chroma threshold1. Chroma value smaller than this threshold will be suppressed with y\_weight.

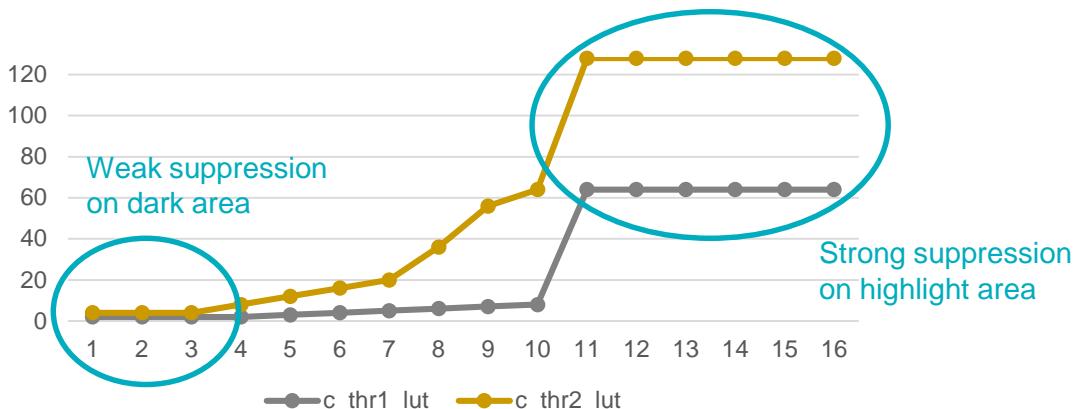
Length	Default	Min	Max	Higher value	Lower value
16	All 0	0	128	Stronger suppression	Weaker suppression

- **c\_thr2\_lut[knee\_point]**

- Define chroma threshold2. Chroma value bigger than this threshold will not be suppressed.

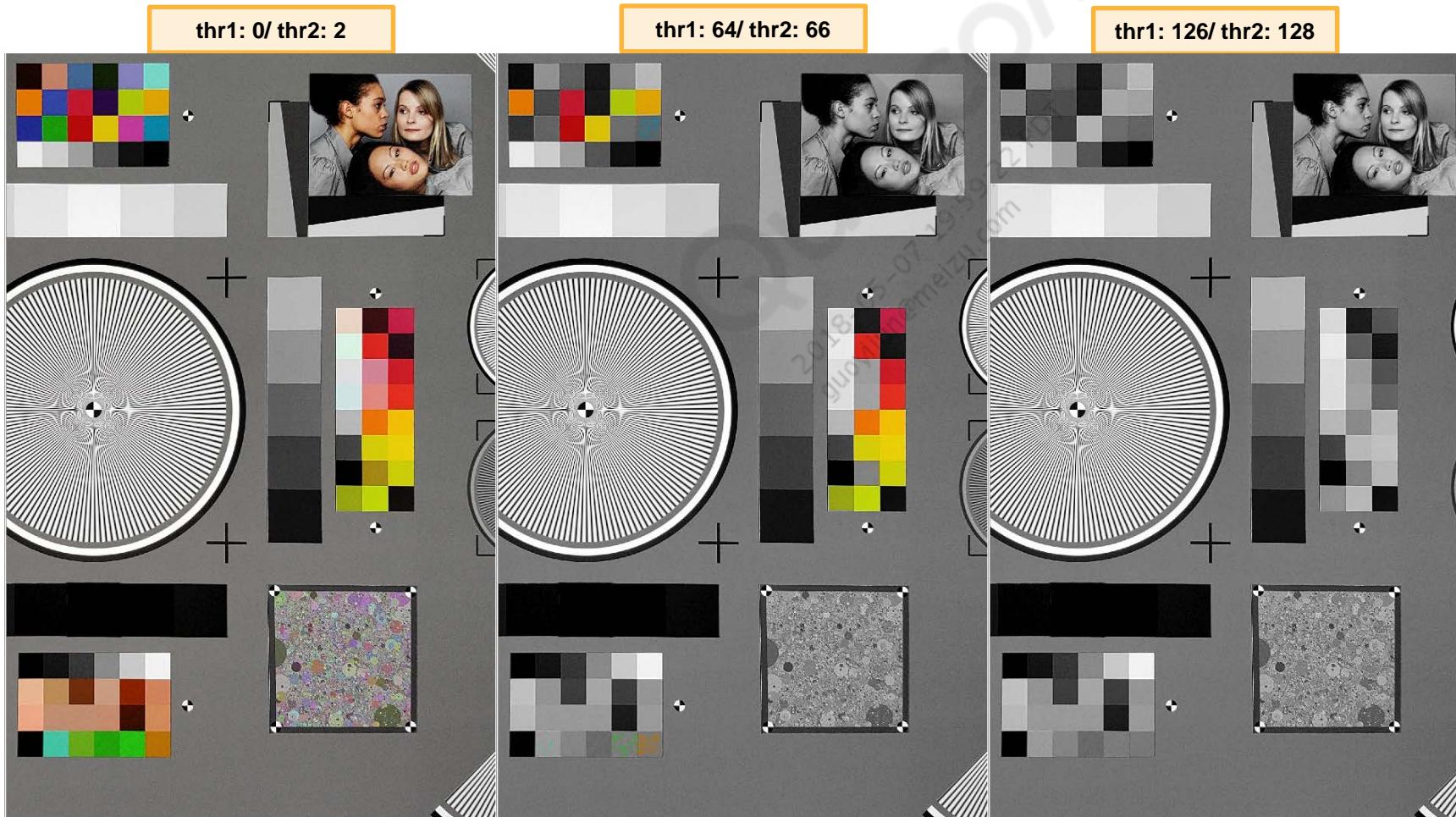
Length	Default	Min	Max	Higher value	Lower value
16	All 2	0	128	Stronger suppression	Weaker suppression

- **c\_thr1\_lut[i] must be less than (<) c\_thr2\_lut[i]**



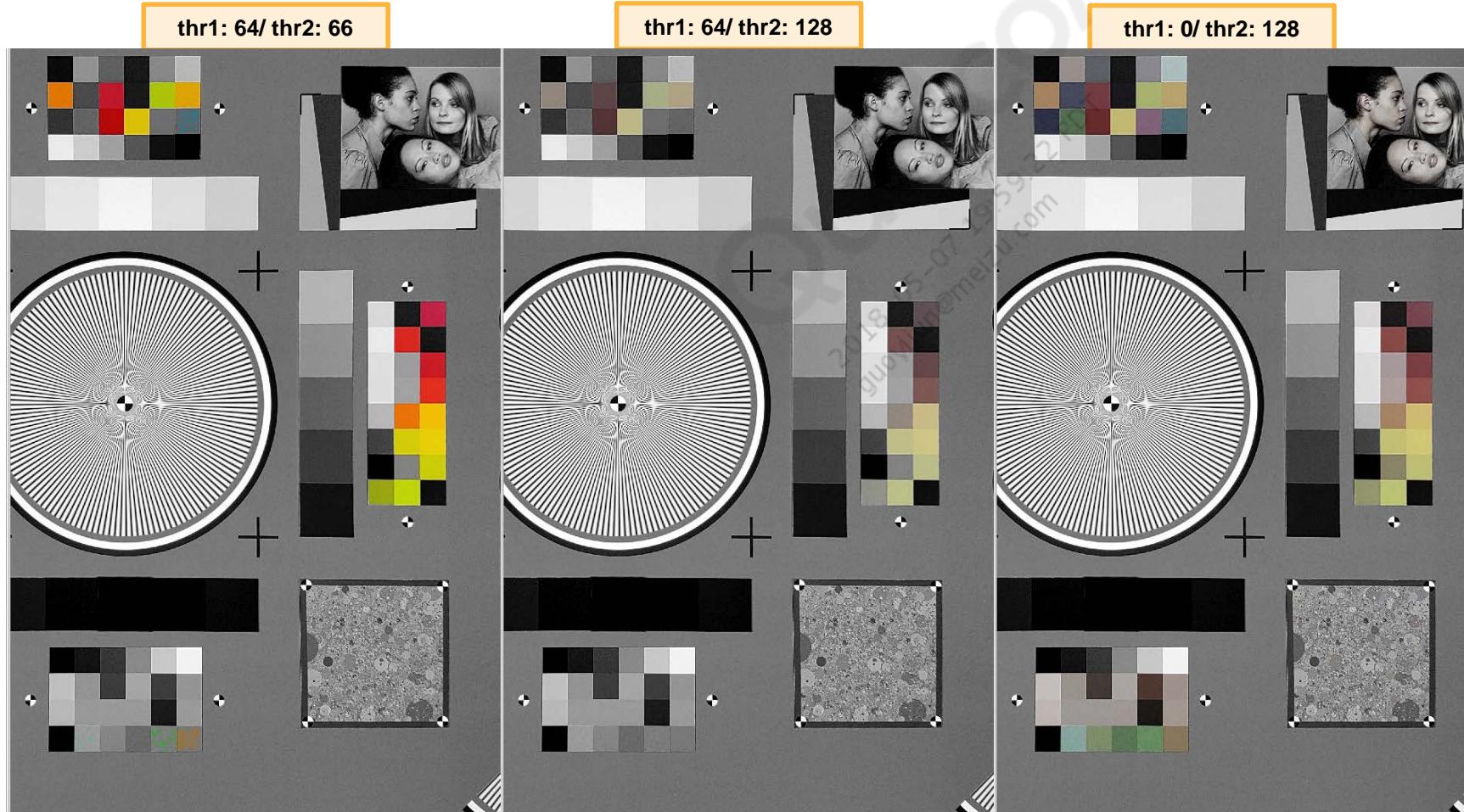
## Step 1.1: c\_thr1\_lut/c\_thr2\_lut (cont.)

- **c\_thr1\_lut/c\_thr2\_lut**
  - When thr1 increases, chroma is suppressed

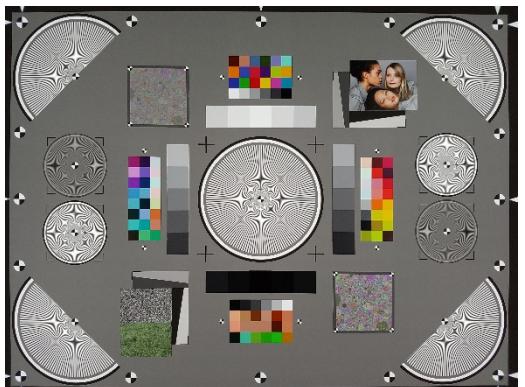


## Step 1.1: c\_thr1\_lut/c\_thr2\_lut (cont.)

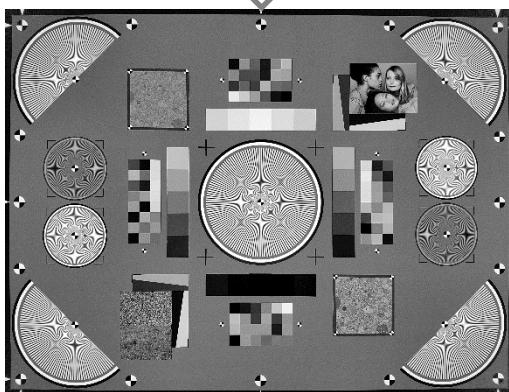
- **c\_thr1\_lut/c\_thr2\_lut**
  - When  $\text{diff}(\text{c\_thr2\_lut} - \text{c\_thr1\_lut})$  is bigger, chroma is suppressed more smoothly



## Step 2: y\_weight\_lut



y\_weight\_lut all 0

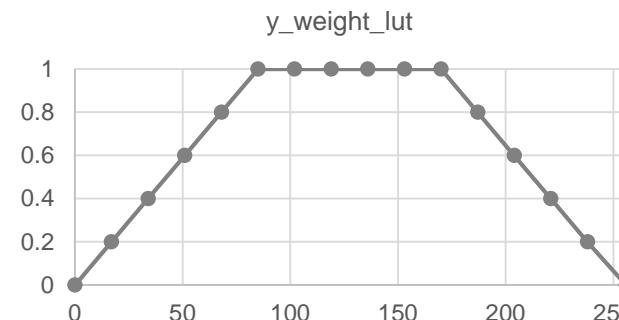


- **y\_weight\_lut[knee\_points]**

- Define luma weight for chroma suppression at luma knee points

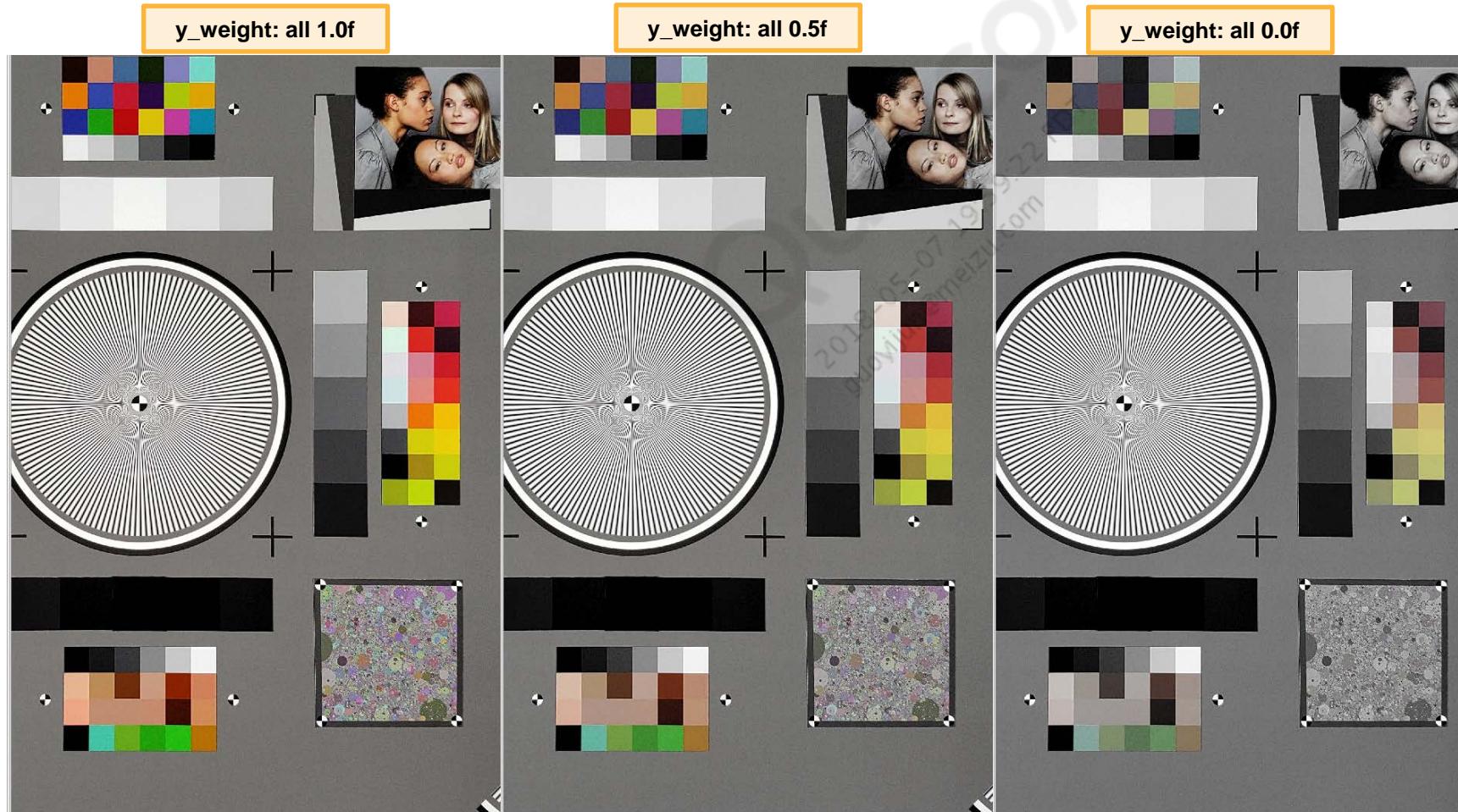
Length	Default	Min	Max	Higher value	Lower value
16	All 0	0.0f	1.0f	Weaker suppression	Stronger suppression

- When y\_weight\_lut is “0.0f”, chroma suppression block is affected fully by c\_thr1\_lut and c\_thr2\_lut
  - When y\_weight\_lut is all “1.0f”, it is as same as disabling chroma suppression block
  - Recommend to apply strong suppression for dark/highlight area and try not to touch middle tone color



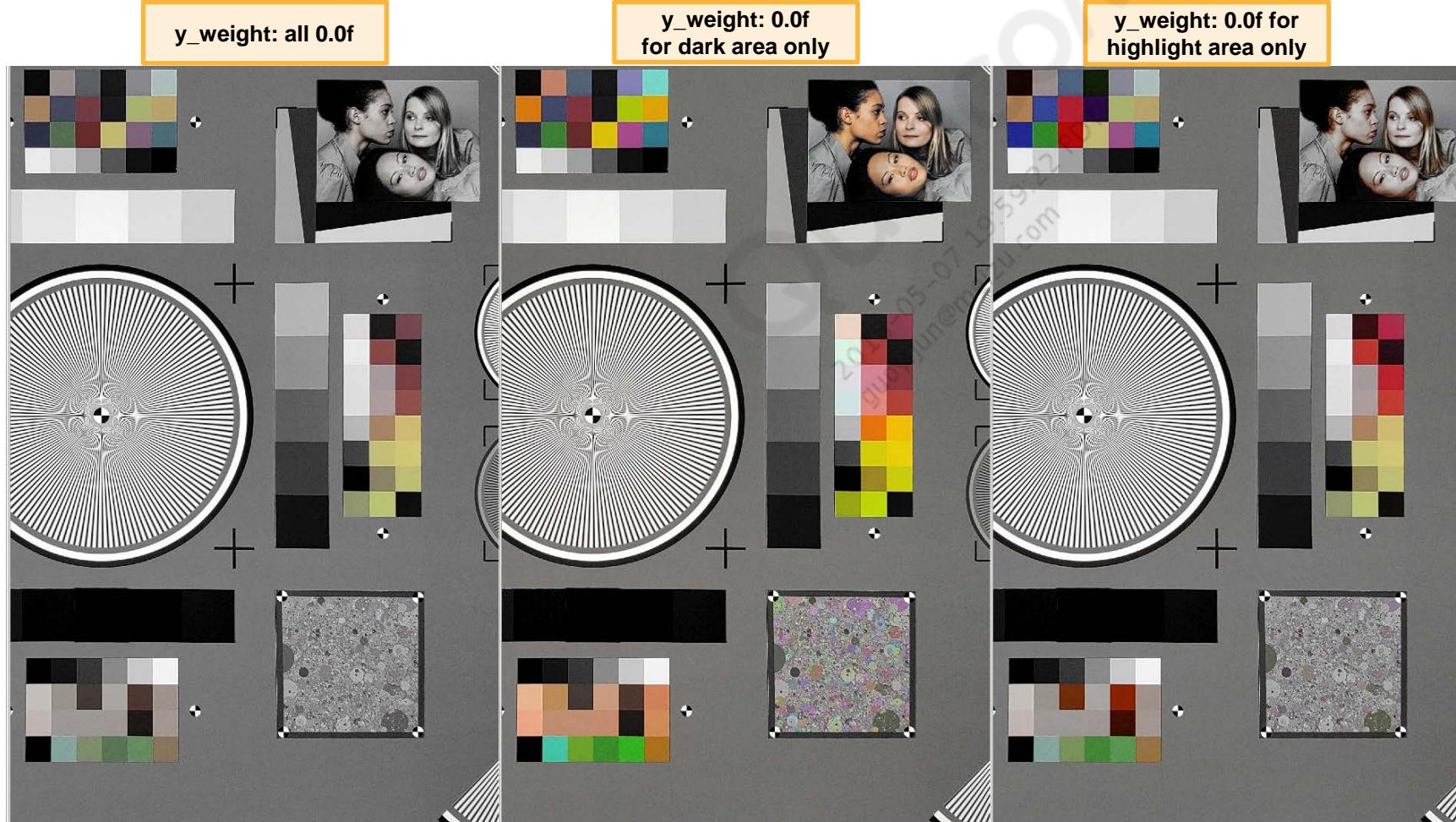
## Step 2: y\_weight\_lut (cont.)

- **y\_weight\_lut**
  - Lower value can suppress more



## Step 2: y\_weight\_lut (cont.)

- **y\_weight\_lut**
  - Suppression strength can be controlled by intensity freely



# How to Coordinate with Other Modules

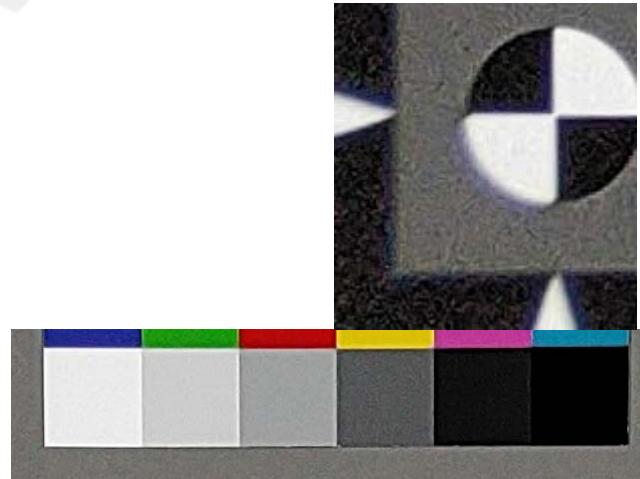
---

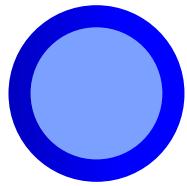
- Tune as a last module for color
- Disable CS with defaults when tuning **color** modules
- To suppress strong color for dark and highlight only, tuning with color NR
- To remove color tint, tune color NR firstly and try CS

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Test Plan

- Enable/disable flags
  - chroma\_suppression\_enable
    - if disabled
    - CS totally disable
- To bypass CS using parameters
  - y\_weight\_lut = all 1
  - The effect is same as CS disable
- How to confirm this module works normally
  - Take TE42 image
  - Check if color noise is not observed for dark/highlight area using this block
  - Check if color tint is observed on bright/ dark area
  - Check if discontinuity color data is observed by this block
  - Check if original colors are not suppressed by this block
- Test images
  - TE42 chart
  - Outdoor with saturated/ near saturated lightness and backlight dark scenes
    - Light sources, the bright sky, white walls, whiteboards, colorful objects, shade, dark backlight with building





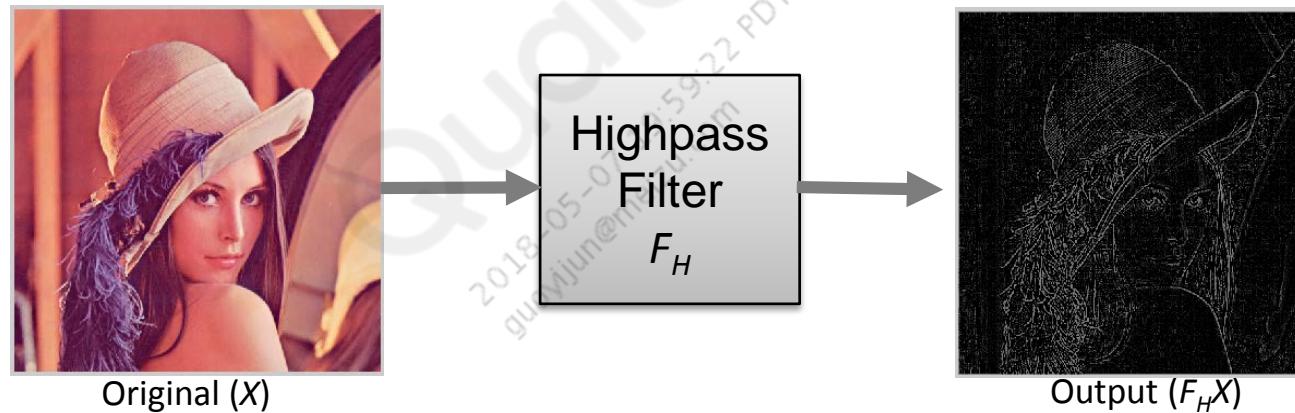
# Adaptive Spatial Filter (ASF)

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Adaptive Spatial Filter (ASF) – Introduction

## Problem statement

- Edge enhancement is a common practice in image processing that increases the local contrast at boundaries, therefore making the image appear sharper. The ASF module is located in PPS of IPE
- Sharpening = Original image + highpass filtered image



High-emphasis filter enhances the details and edges.



# Adaptive Spatial Filter (ASF) – Introduction (cont.)

- Sharpening strength :

$$X + \alpha(F_H X) \rightarrow Y$$



Original ( $X$ )

+

$\alpha$

Sharpening  
Strength



Highpass filtered  
( $F_H X$ )

big  $\alpha$



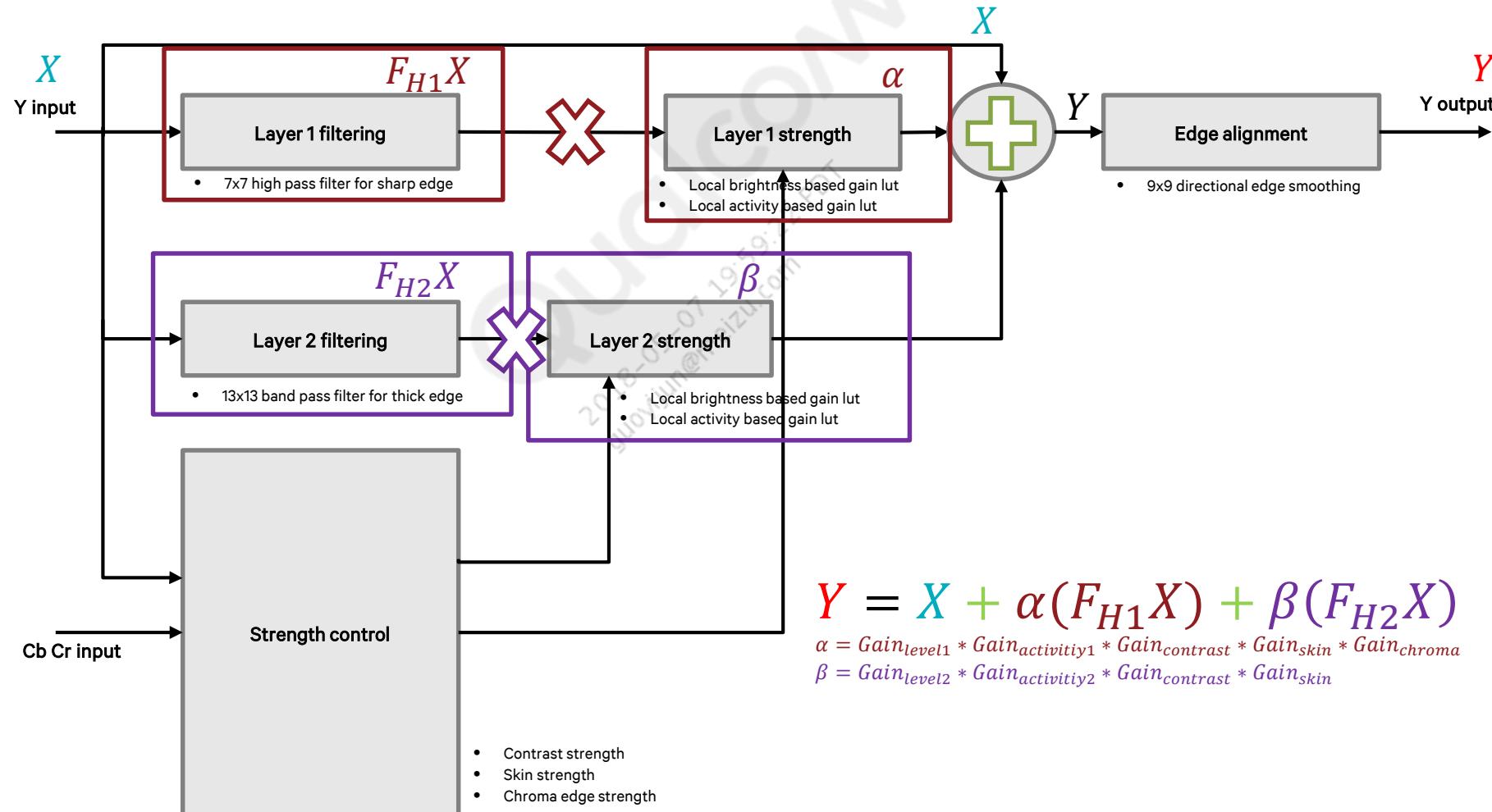
Output ( $Y$ )

small  $\alpha$



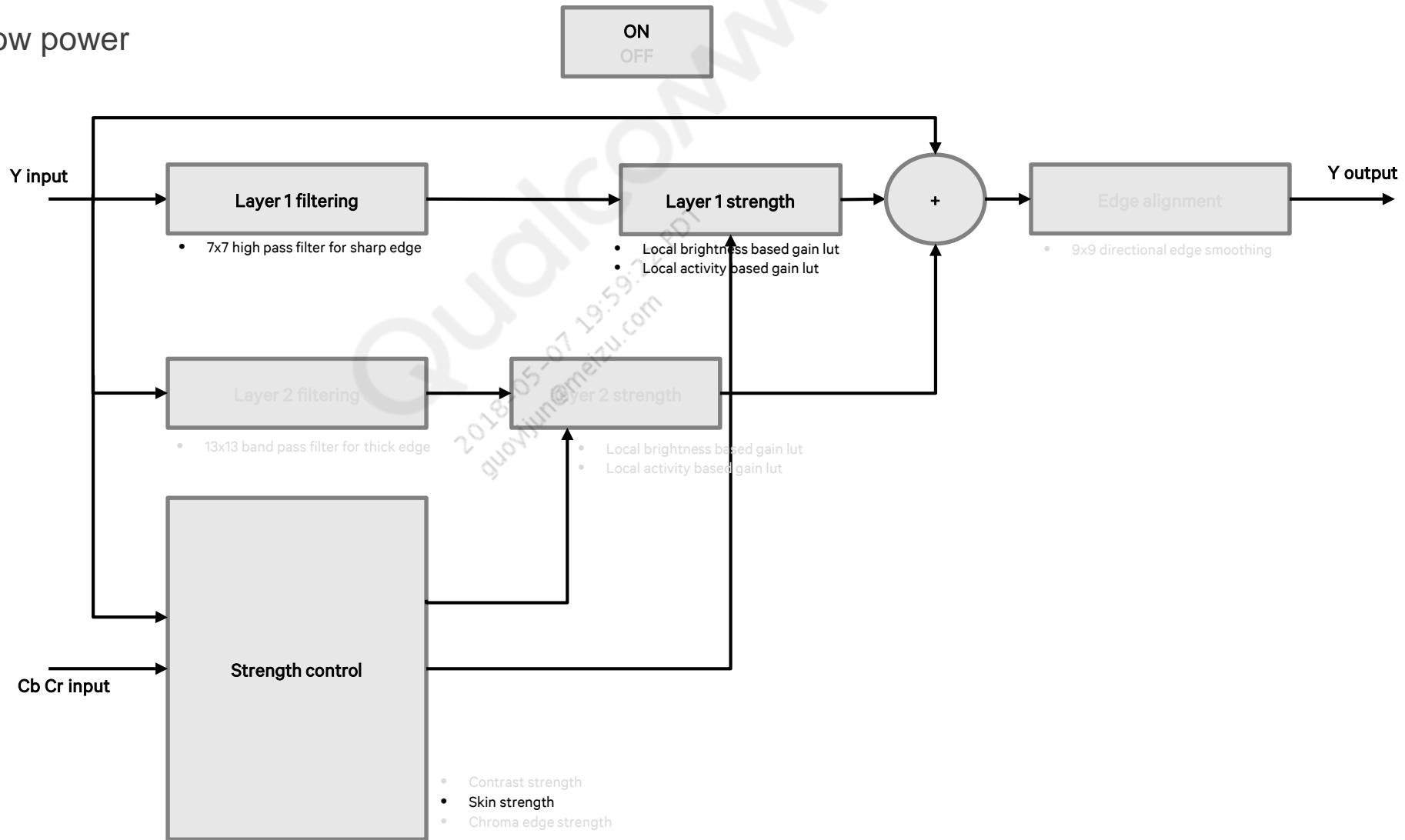
# Overview

## Block diagram



# Overview (cont.)

- Block diagram – Low power

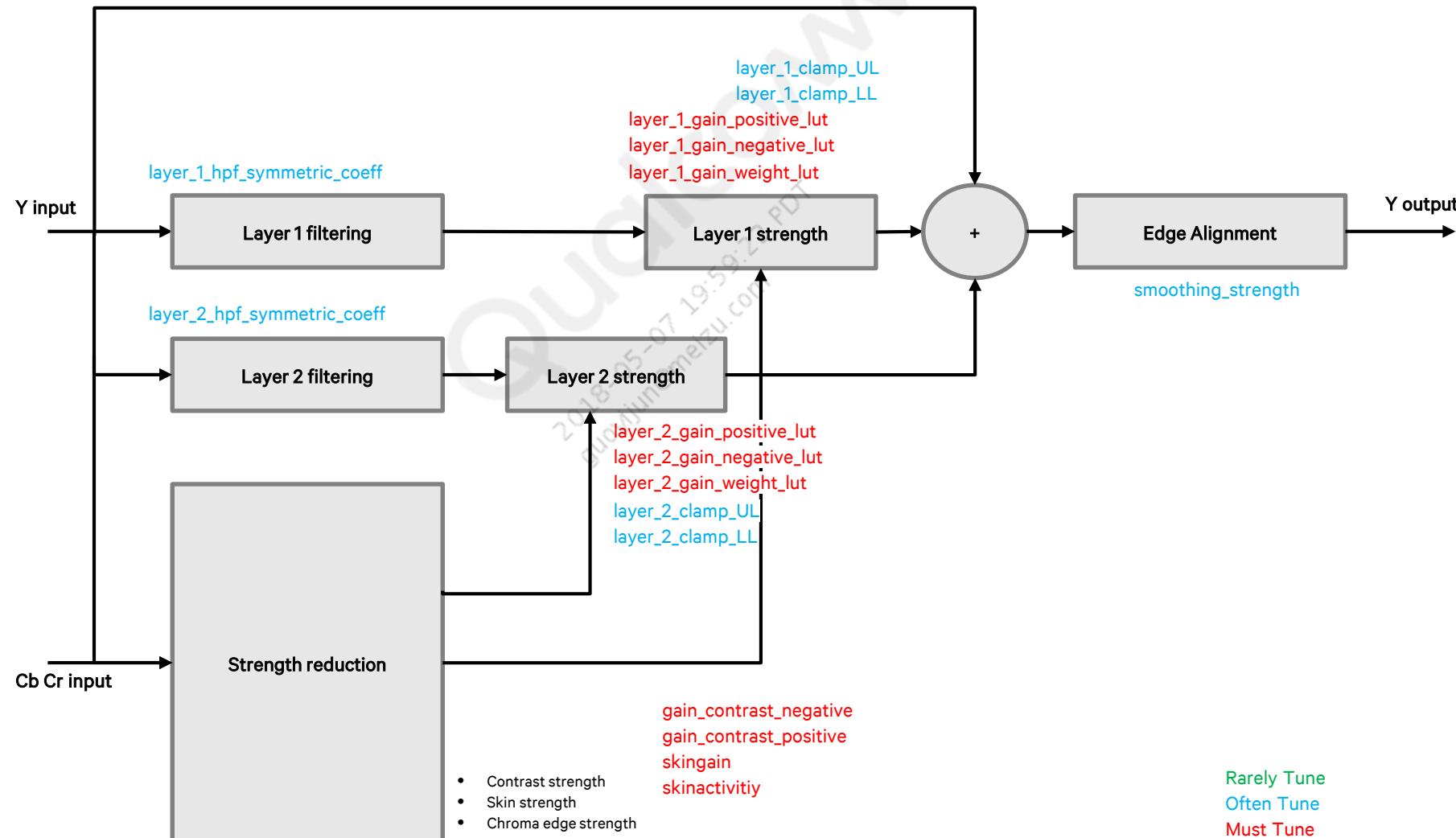


# Tuning Procedure

---

1. Set every parameter as default
  - For radial and skin tuning, follow the instruction of each calibration
2. Tune **Must tune** and **Often tune** parameters
  - a) Set layer 1 symmetric kernel based on light condition
  - b) Tune layer 1 gain strength based on level/activity to enhance steep edges and textures
  - c) Set layer 2 kernel as default
  - d) Tuning layer 2 gain strength based on level/activity to enhance thick edges and more local edge enhancement
  - e) If you see halo, tune gain\_contrast to reduce gain strength around edge
  - f) If you see dirty skin, tune skin\_gain to reduce sharpening on skin
  - g) Tune smoothing strength to make edge smoothly
  - h) Tune clamp if you still have halo after gain\_contrast tuning
3. Tune **Rarely tune** for fine tuning
  - a) When corner is noisier or less detail, tune radial parameters to make sharpening weaker or stronger.
  - b) When halo is observed around chroma edge, tune gain\_chroma to remove halo around chroma when contrast is not high
  - c) When texture is connected by edge alignment unnaturally, tuning thresholds in edge alignment not to detect texture as edge

# Effects of Parameters: Must Tune and Often Tune

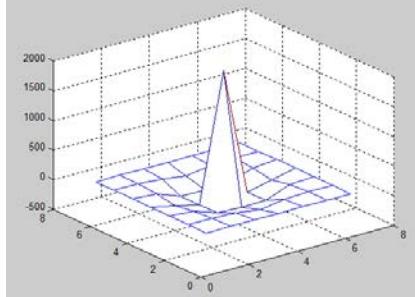


# Step 2.1: Symmetric Kernel

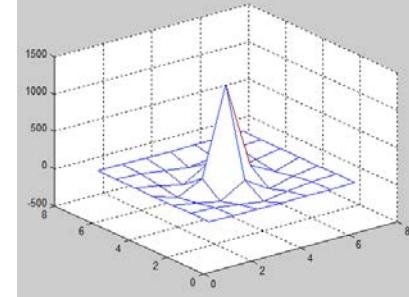
- **layer\_1\_hpf\_symmetric\_coeff**

- 1<sup>st</sup> layer 7x7 sharpening coefficients
  - Length: 10; Q10
  - Default
    - Thin Kernel (for outdoor)
      - 0, 0, 0, -1, -2, -25, -86, -173, -180, 1968
    - Mid Kernel (for indoor)
      - 0, 0, -3, -5, -10, -57, -106, -132, 65, 1232
    - Thick Kernel (for low light)
      - 0, -2, -9, -15, -22, -67, -89, -46, 144, 736

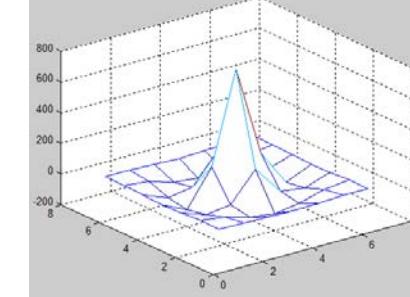
**Thin kernel**



**Mid kernel**



**Thick kernel**

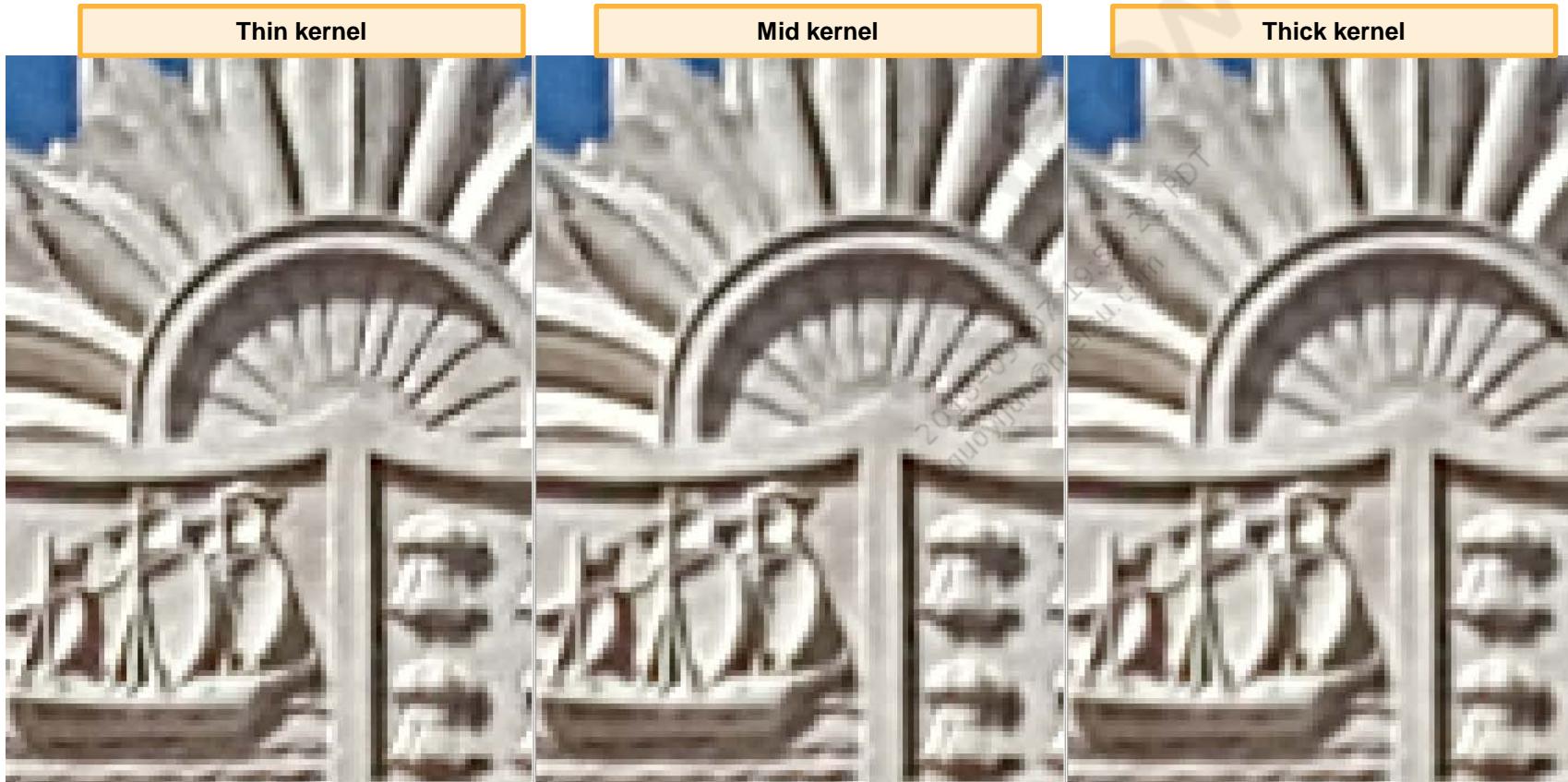


- The sum of kernel should be 0
  - Not recommended to change kernel manually

Qualcomm  
2018-05-07 19:59:22 PDT  
guojun@meizu.com

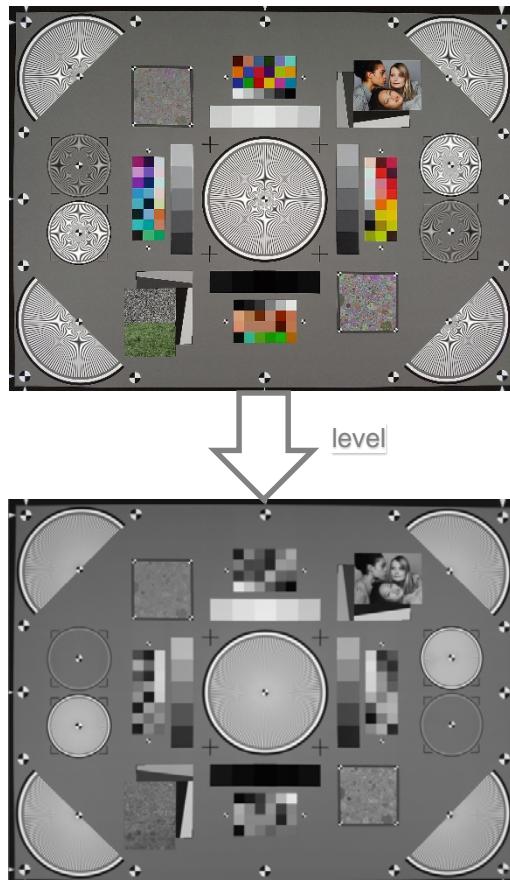
## Step 2.1: Symmetric Kernel (cont.)

- `layer_1_hpf_symmetric_coeff`



- Thin kernel helps to show small details like a tree and makes the image sharp
- Thick kernel helps to show edges smoothly and less zappy

## Step 2.2: gain\_positive/negative\_lut



level = Y intensity value/4

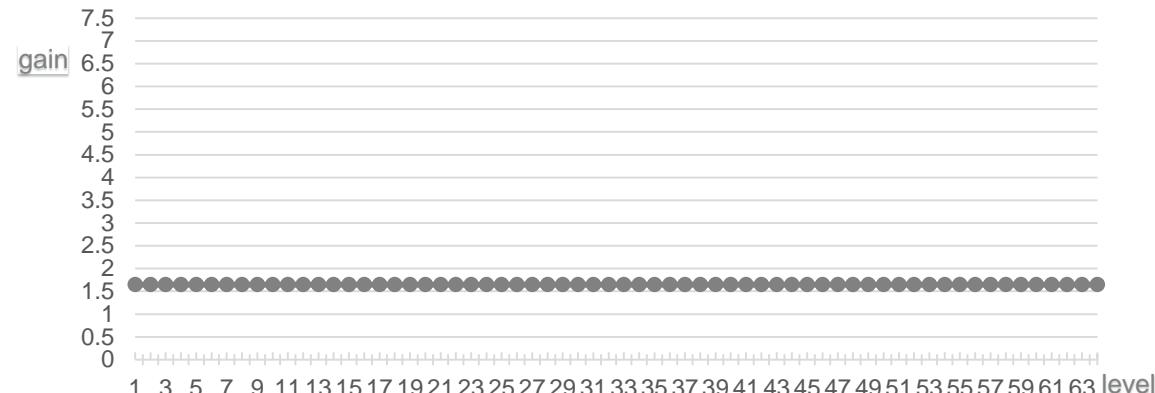
- **layer\_1\_gain\_positive\_lut[level]**
  - Level-based sharpening gain LUT for **positive** halo

Length	Default	Min	Max	Higher value	Lower value
64	All 1.65f	0.0f	7.9f	Stronger sharpening	Weaker sharpening

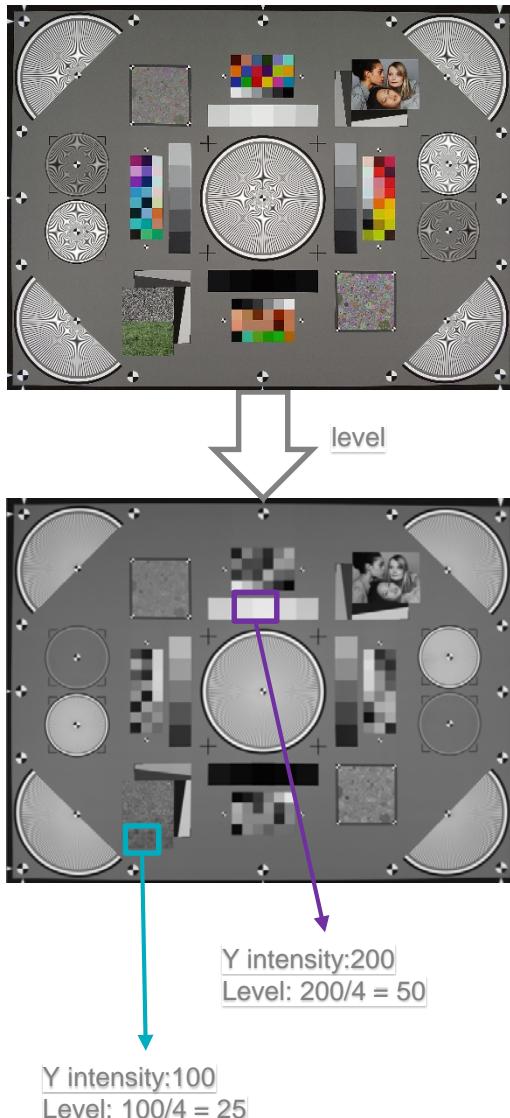
- **layer\_1\_gain\_negative\_lut[level]**
  - Level-based sharpening gain LUT for **negative** halo

Length	Default	Min	Max	Higher value	Lower value
64	All 1.65f	0.0f	7.9f	Stronger sharpening	Weaker sharpening

- Level is from **layer\_1\_lpf\_coeff**  
Default positive/negative gain lut

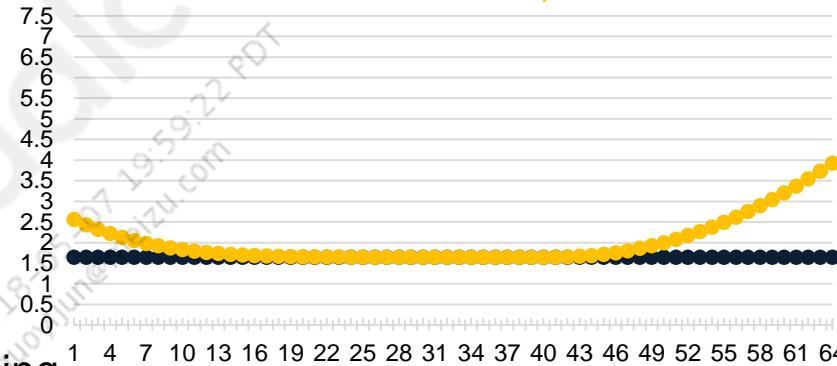


## Step 2.2: gain\_positive/negative\_lut (cont.)



- General setting for gain\_positive/negative lut
  - Bright/ Normal light: U-shape curve, stronger enhancement for highlight & shadow
  - Low light: flat curve, same enhancement for all intensity levels (Default: all 0.6f)

U-shape and flat curve



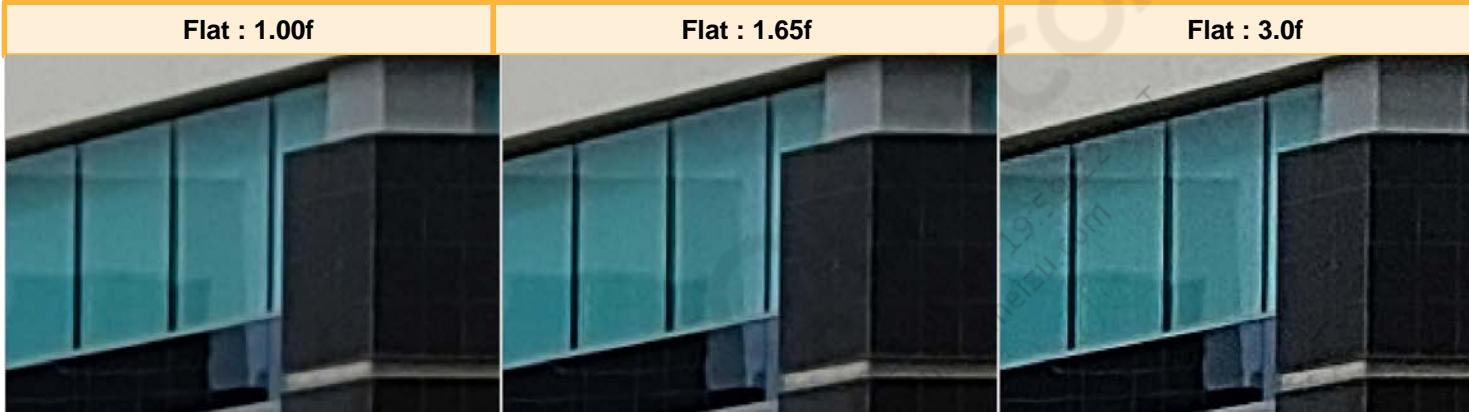
- Manual curve tuning
  - Get level data of the part want to enhance or release sharpening and tune manually on the level.

Tuned positive gain lut

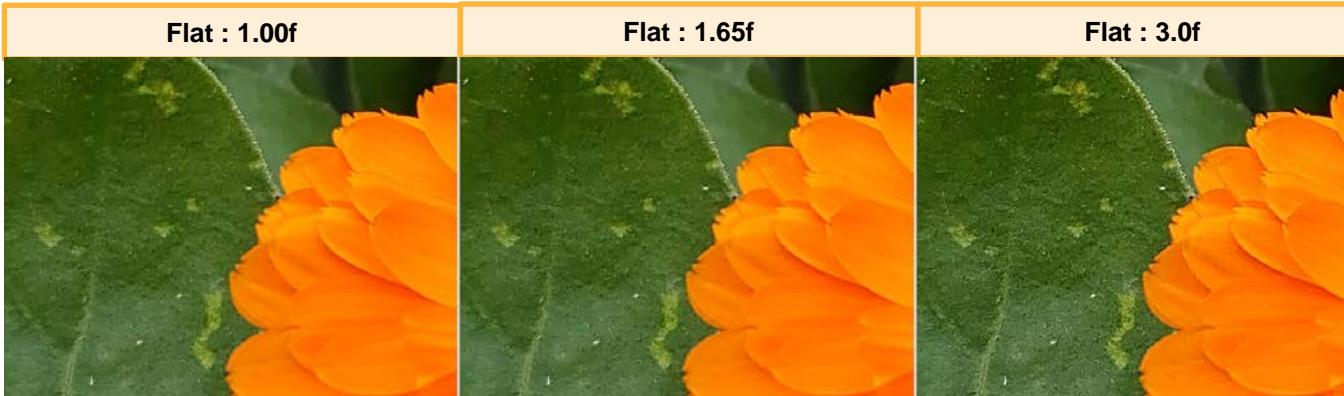


## Step 2.2: gain\_positive/negative\_lut (cont.)

- layer\_1\_gain\_positive\_lut/layer\_1\_gain\_negative\_lut
  - Sharpness on edge getting stronger by setting gain\_lut higher

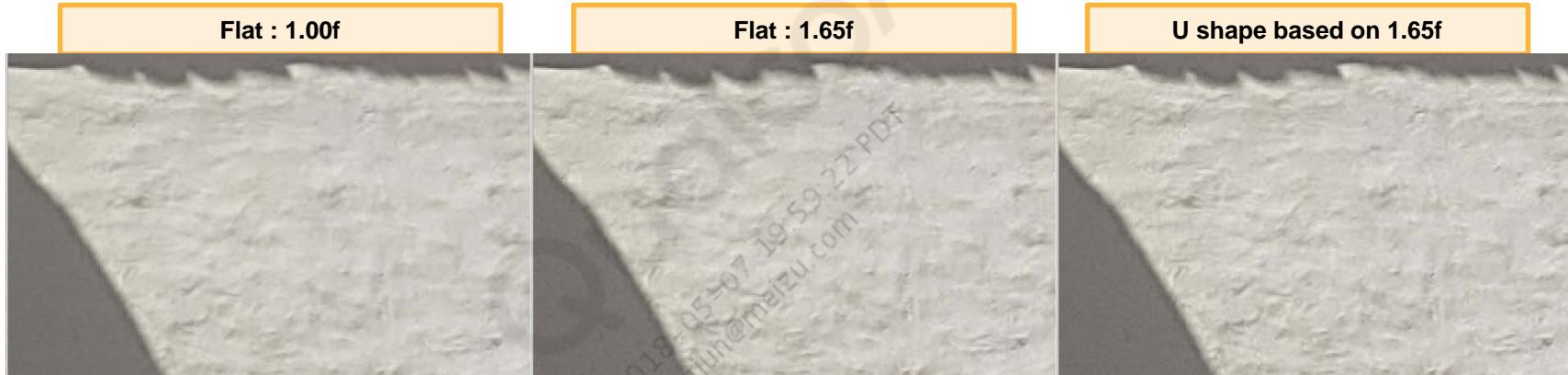


- Details getting stronger by setting gain\_lut higher but it increases noise as well



## Step 2.2: gain\_positive/negative\_lut (cont.)

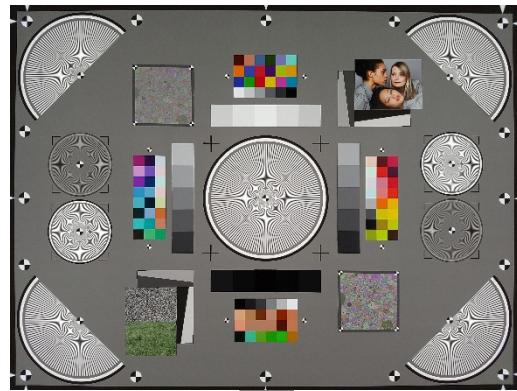
- **layer\_1\_gain\_positive\_lut/layer\_1\_gain\_negative\_lut**
  - U shape can generate stronger sharpening in highlight



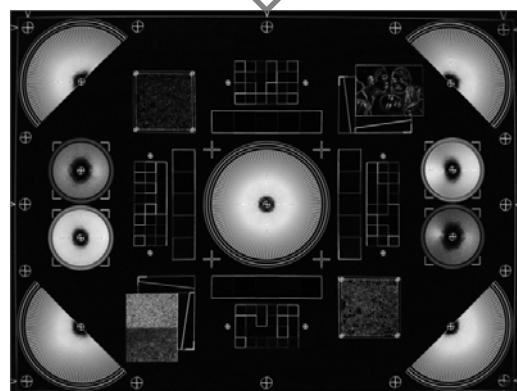
- Manual shape also can make stronger sharpening as same as U shape curve



## Step 2.2: gain\_weight\_lut



local variation



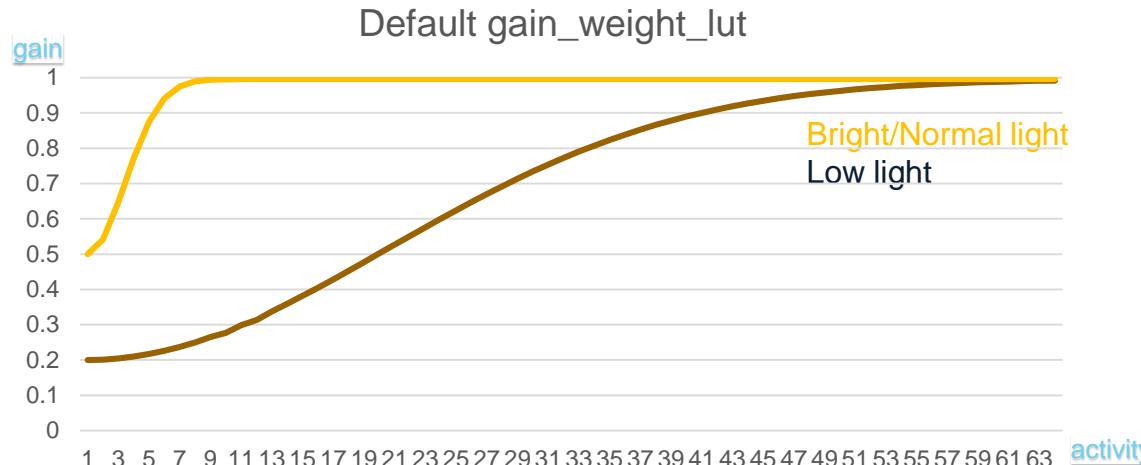
local variation(Q8)=  
how active the pixel is  
using band pass filter

activity = local variation/4

- **layer\_1\_gain\_weight\_lut[activity]**
  - Normalized activity-based sharpening gain LUT

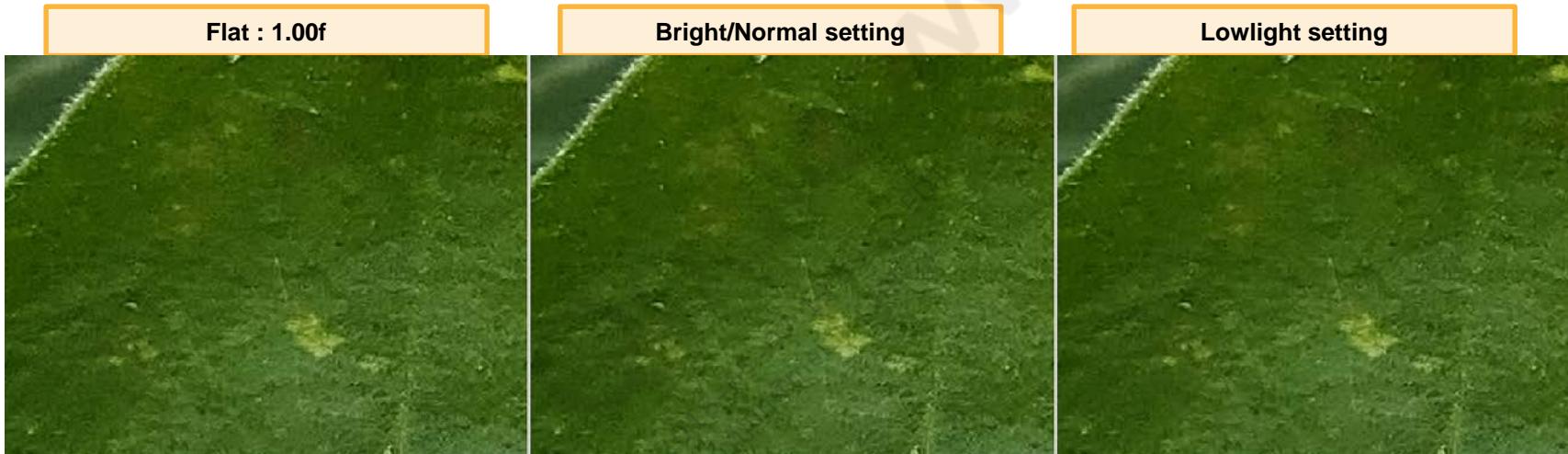
Length	Default	Min	Max	Higher value	Lower value
64	From 0.5 to 1.0	0.0f	0.996f	Stronger sharpening	Weaker sharpening

- Activity: Normalized 5x5 BPF and LPF of input
  - BPF using `layer_1_activity_band_pass_coeff`
  - Scaled using `layer_1_norm_scale` and `layer_1_L2_norm_en`
  - Clamped using `layer_1_activity_clamp_threshold`
  - Normalization using `layer_1_activity_normalization_lut`

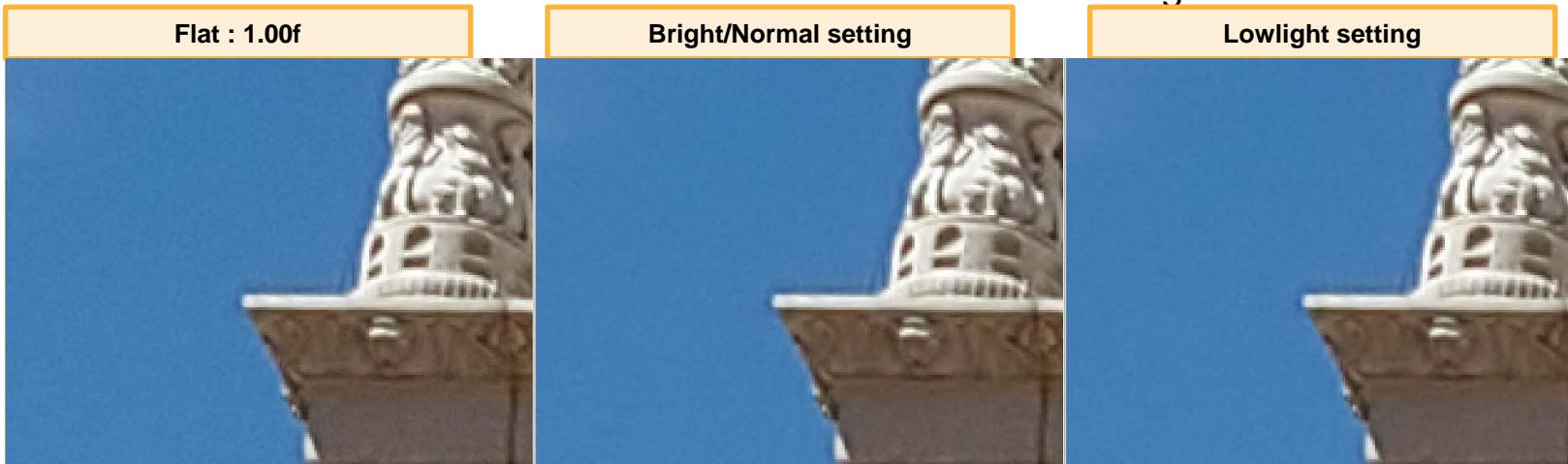


## Step 2.2: gain\_weight\_lut (cont.)

- **layer\_1\_gain\_weight\_lut**

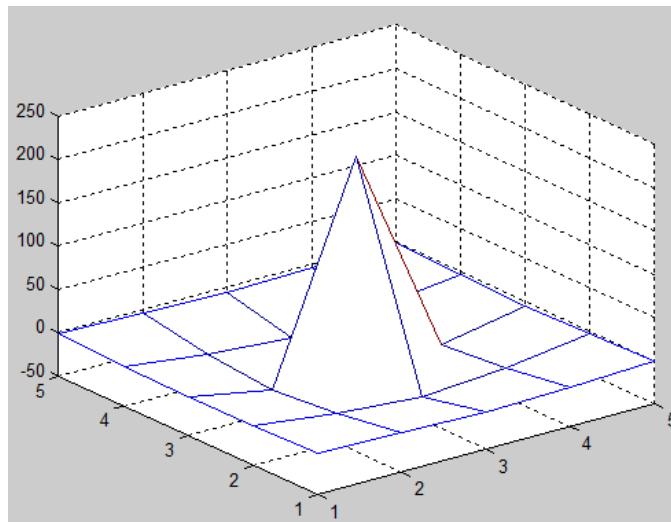


- Details are more clearer when it is all 1.0 but noise on flat also comes together



## Step 2.3: Layer 2 Kernel

- [layer\\_2\\_hpf\\_symmetric\\_coeff](#)
  - 2<sup>nd</sup> layer sharpening coefficients
  - Effect: 5x5 kernel on x2 downscaled image and ×2 up-sampled
  - Length: 6; Q10
  - Default:
    - -1, -4, -6, -16, -24, 220



- The sum of kernel should be 0
- Not recommended to change kernel manually

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

## Step 2.4: Layer 2 Luts

---

- **layer\_2\_gain\_positive\_lut[level]**

- Level-based sharpening gain LUT for **positive** halo

Length	Default	Min	Max	Higher value	Lower value
64	All 0.55f	0.0f	7.9f	Stronger sharpening	Weaker sharpening

- Weaker gain to avoid thick halo

- **layer\_2\_gain\_negative\_lut[level]**

- Level-based sharpening gain LUT for **negative** halo

Length	Default	Min	Max	Higher value	Lower value
64	All 0.55f	0.0f	7.9f	Stronger sharpening	Weaker sharpening

- Weaker gain to avoid thick halo

- **layer\_2\_gain\_weight\_lut[activity]**

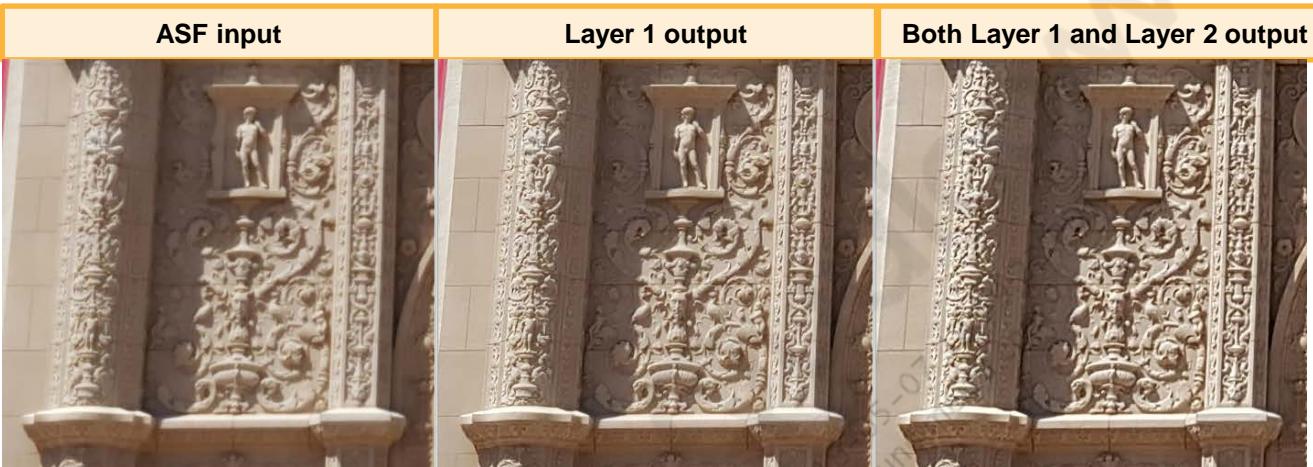
- Normalized activity-based sharpening gain LUT

Length	Default	Min	Max	Higher value	Lower value
64	From 0.5 to 1.0	0.0f	0.996f	Stronger sharpening	Weaker sharpening

- Can make lut same as layer 1

## Step 2.4: Layer 2 Examples

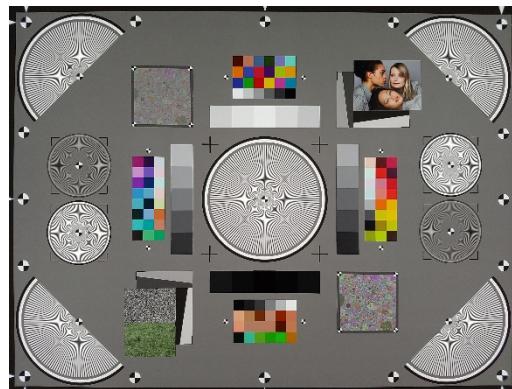
- **Layer 1 and layer 2** parameters are the same with different effects



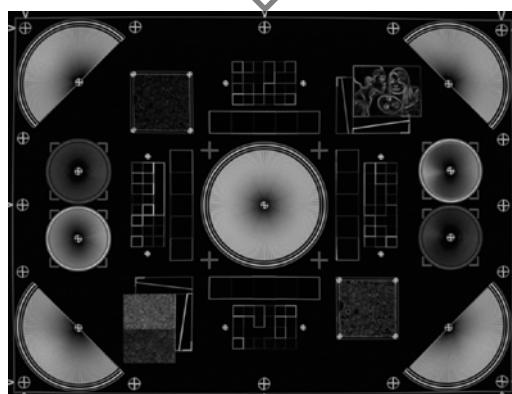
- Details can be more observed when it is all 1.0 but noise on flat also comes together.



## Step 2.5: gain\_contrast\_positive/negative



Y min max diff



Y min max diff (Q8)

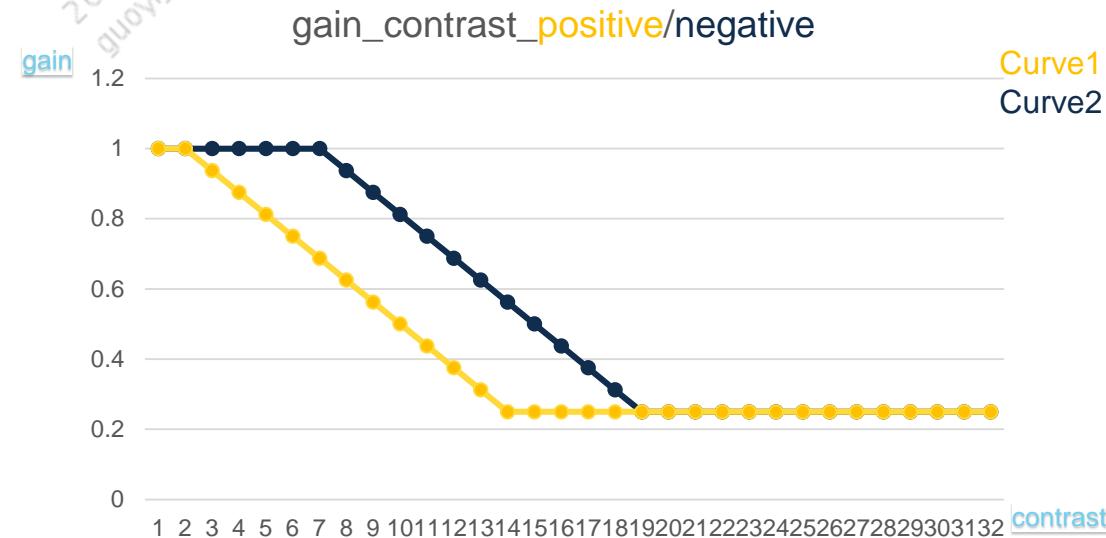
contrast = Y min max diff/4

- **gain\_contrast\_positive/negative [contrast]**

- Contrast-based sharpening gain LUT

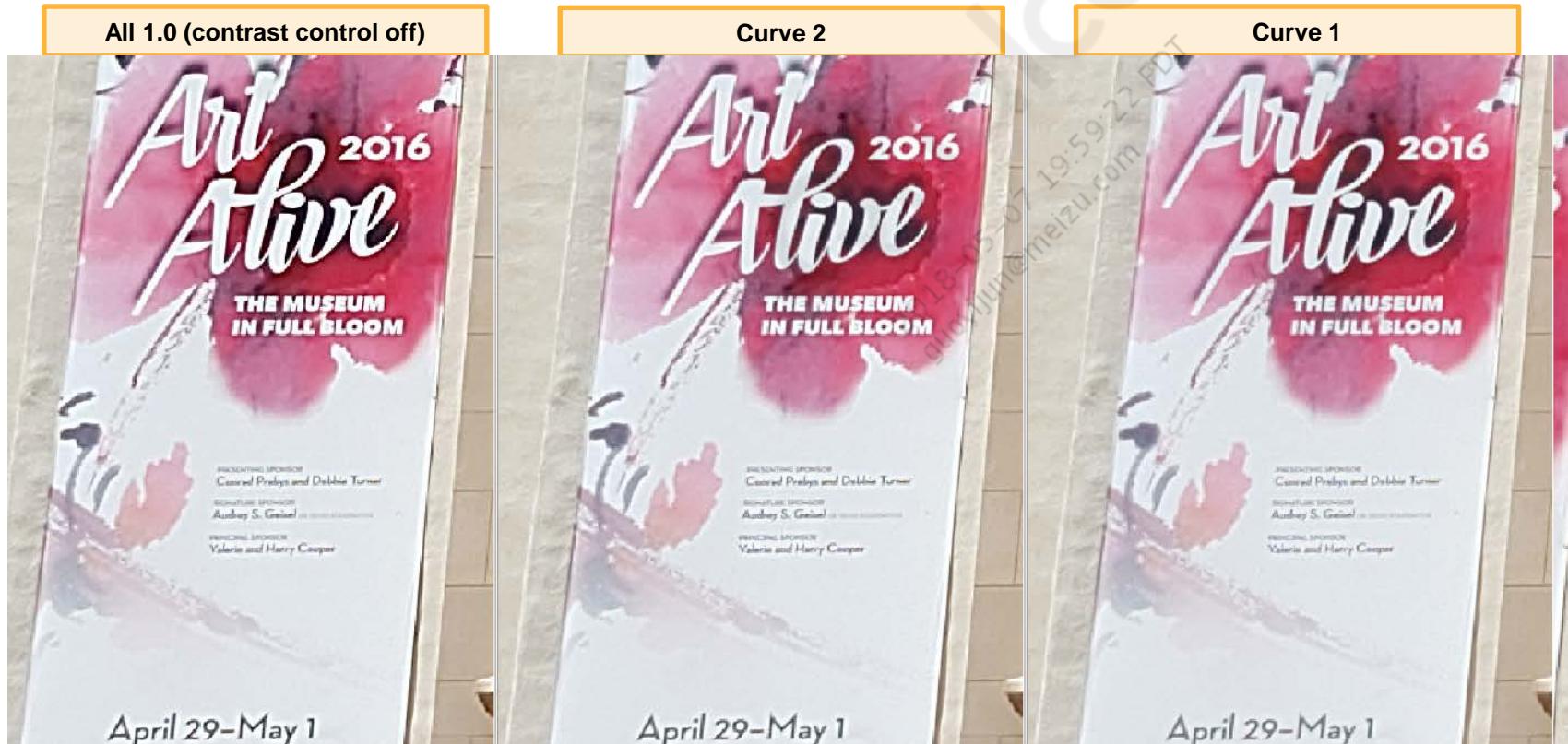
Length	Default	Min	Max	Higher value	Lower value
32	From 1.0 to 0.25	0.004f	1.00f	Stronger sharpening	Weaker sharpening

- Strength control based on contrast
    - Define which area could have halo.
    - Reduce strength on high contrast area.
    - Usually positive halo looks worse than negative halo



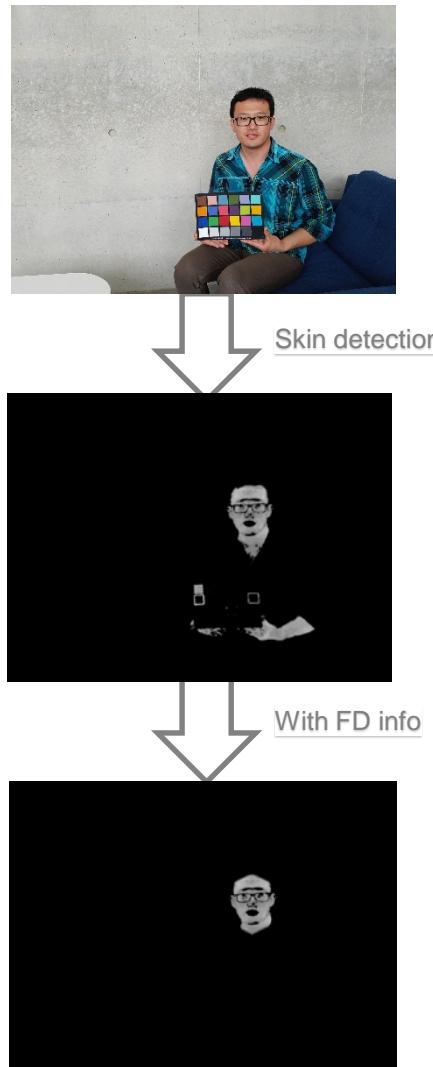
## Step 2.5: gain\_contrast\_positive/negative (cont.)

- **gain\_contrast\_positive/negative**
  - Curve 1 for gain\_contrast\_positive
  - Curve 2 for gain\_contrast\_negative



- Try to keep negative edge and reduce only positive by 2 shape of curve on each

## Step 2.6: Skingain

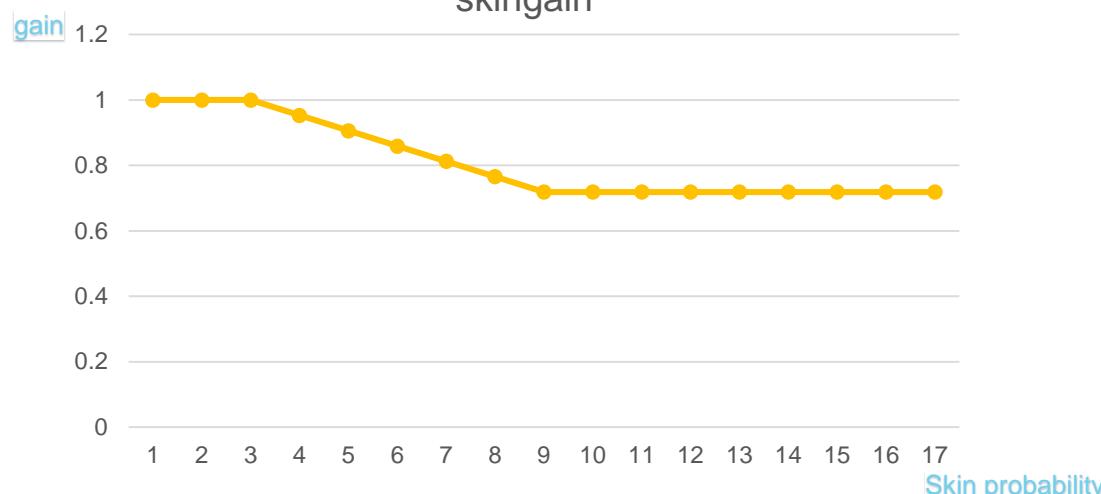


- **skingain**

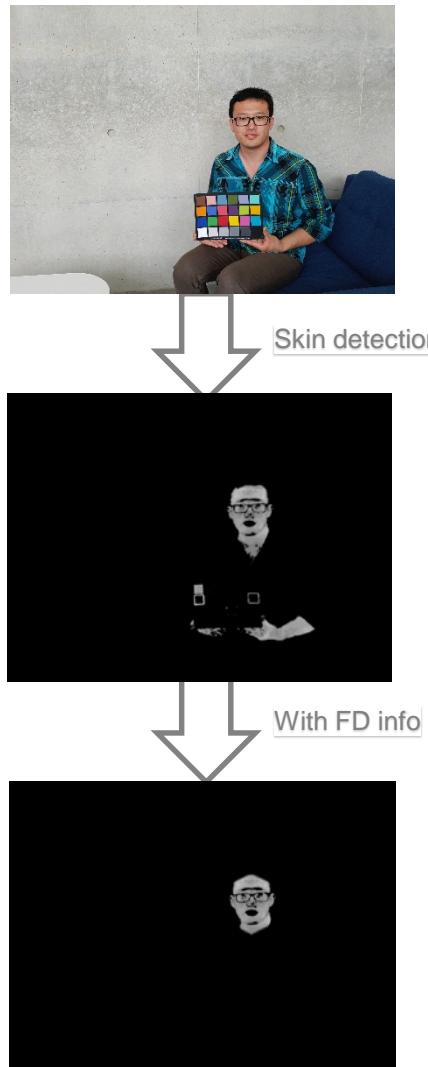
- Skin color-based sharpening gain LUT

Length	Default	Min	Max	Higher value	Lower value
17	From 1.0 to 0.76	0.004f	1.00f	Stronger sharpening	Weaker sharpening

- Strength control based on skin color
    - Define skin color and face
    - Reduce **gain strength** on face
    - Face can be smoothed by FD information
    - Set by **face\_boundary** and **face\_transition** skingain



## Step 2.7: Skinactivity

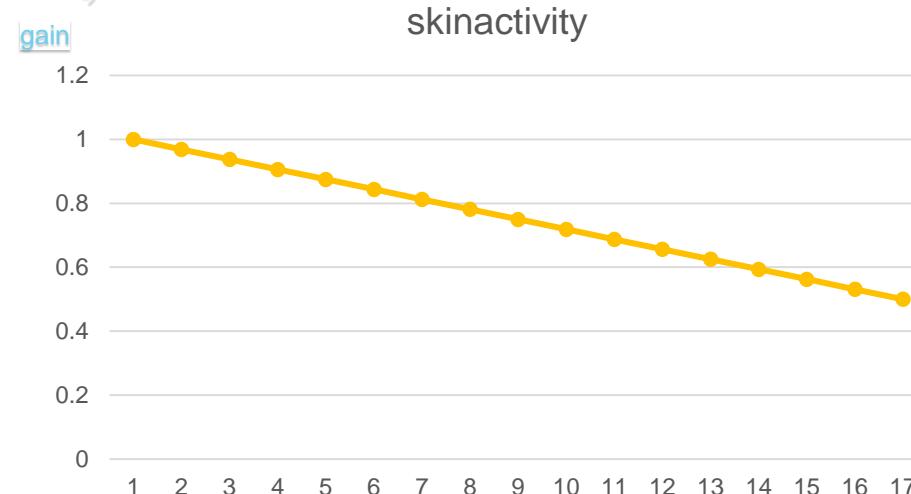


- **skinactivity**

- Skin color-based sharpening gain activity LUT

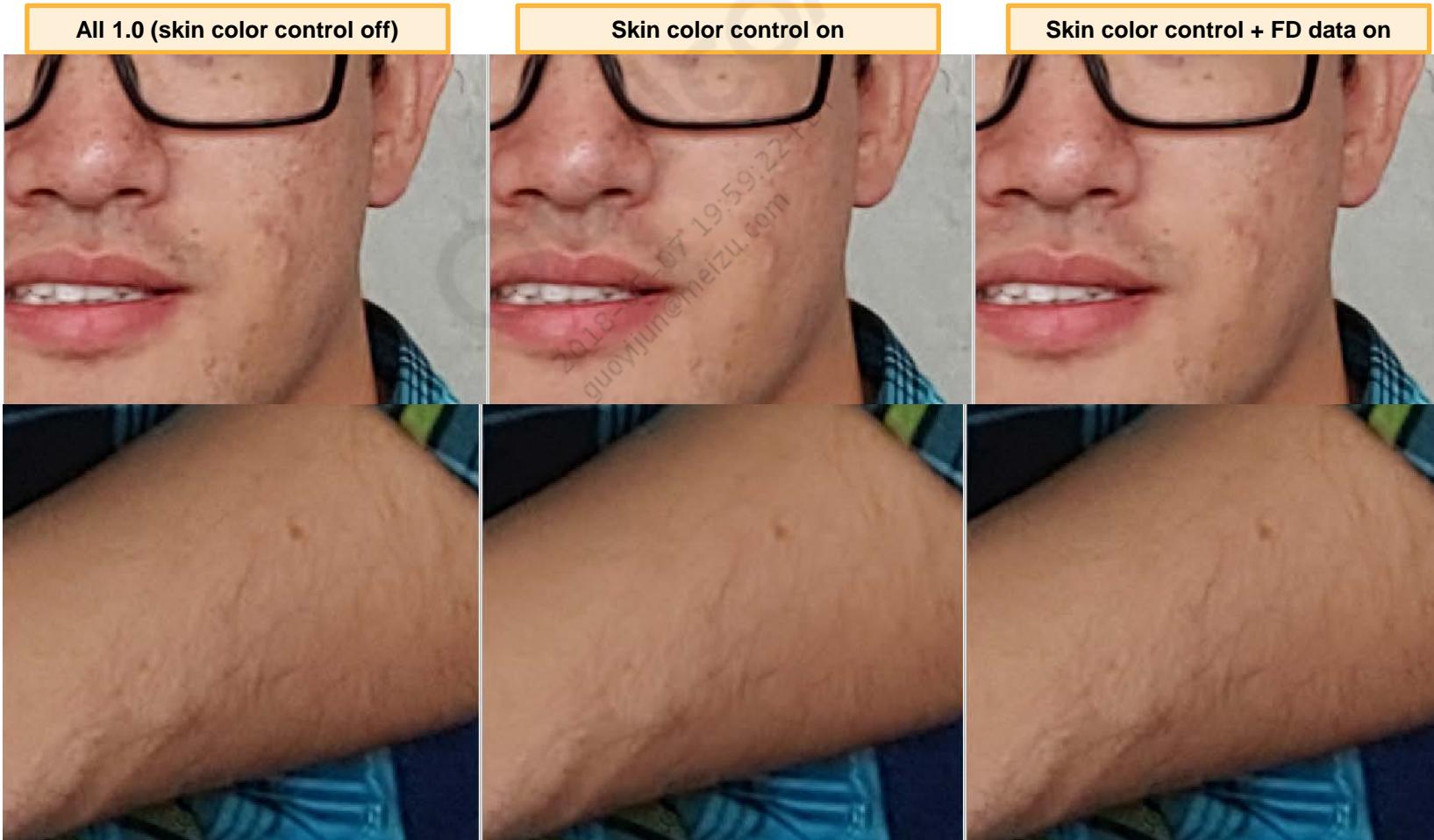
Length	Default	Min	Max	Higher value	Lower value
17	From 1.0 to 0.5	0.004f	1.00f	Stronger sharpening	Weaker sharpening

- Activity strength control based on skin color
    - Define skin color and face
    - Reduce **activity data** on face
    - Face can be smoothed by FD information
      - Set by **face\_boundary** and **face\_transition**

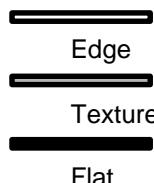


## Step 2.8: Skingain and Skinactivity

- **skingain and skinactivity**
  - Smooth face and keep details on other skin color using FD information



## Step 2.9: smoothing\_strength



- **smoothing\_strength**

- Blending factor between original pixel and edge smoothed pixel

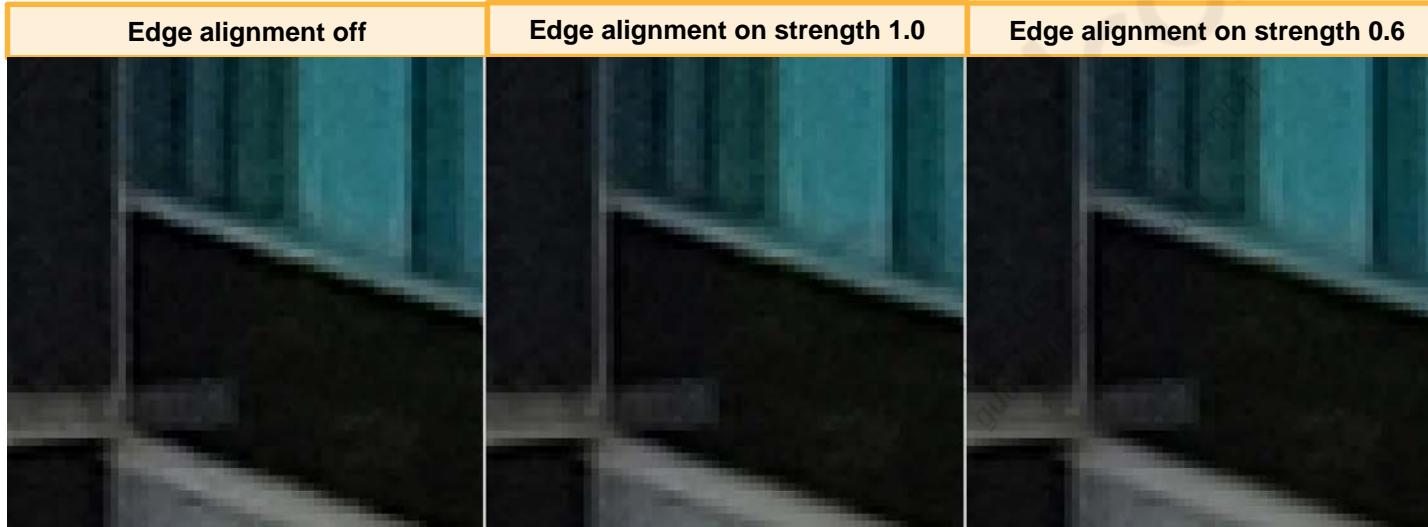
Length	Default	Min	Max	Higher value	Lower value
1	0.5f	0.0f	0.999f	Stronger smoothing	Weaker smoothing

- Edge will be smoothed by direction

- If edge is not detected well by other parameters, this parameter does not work
  - In the left edge map, each color of edge define the each direction of edge

## Step 2.9: smoothing\_strength (cont.)

- **smoothing\_strength**
  - Coarse edges are aligned without reducing texture



- Details don't get hurt by edge alignment



## Step 3.1: Clamp

---

- [layer\\_1\\_clamp\\_UL/ layer\\_2\\_clamp\\_UL](#)

- Manual fixed positive clamping level sharpening

Length	Default	Min	Max	Higher value	Lower value
1	255	0	255	Stronger sharpening	Weaker sharpening

- [layer\\_1\\_clamp\\_LL/ layer\\_2\\_clamp\\_LL](#)

- Manual fixed negative clamping level sharpening

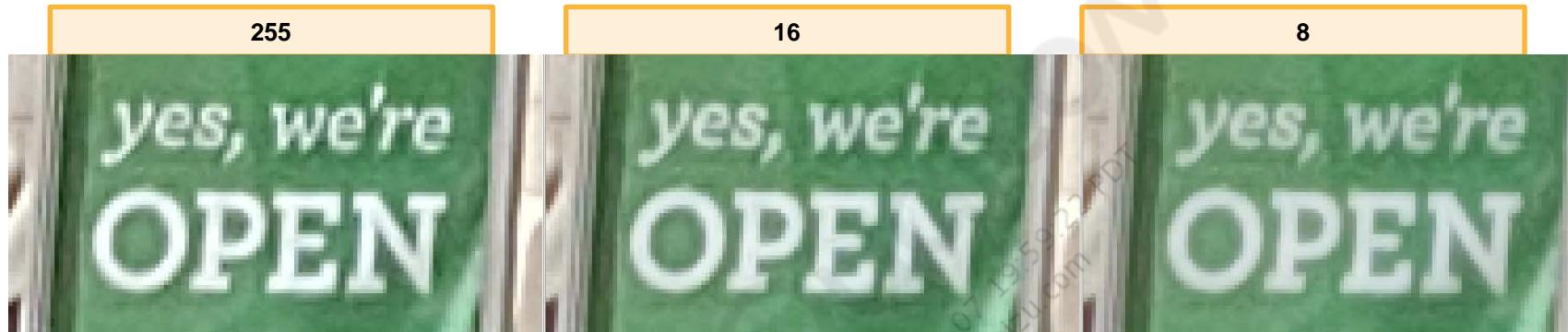
Length	Default	Min	Max	Higher value	Lower value
1	-255	-255	0	Weaker sharpening	Stronger sharpening

- Halo control parameters are preferred to be used first instead of clamp

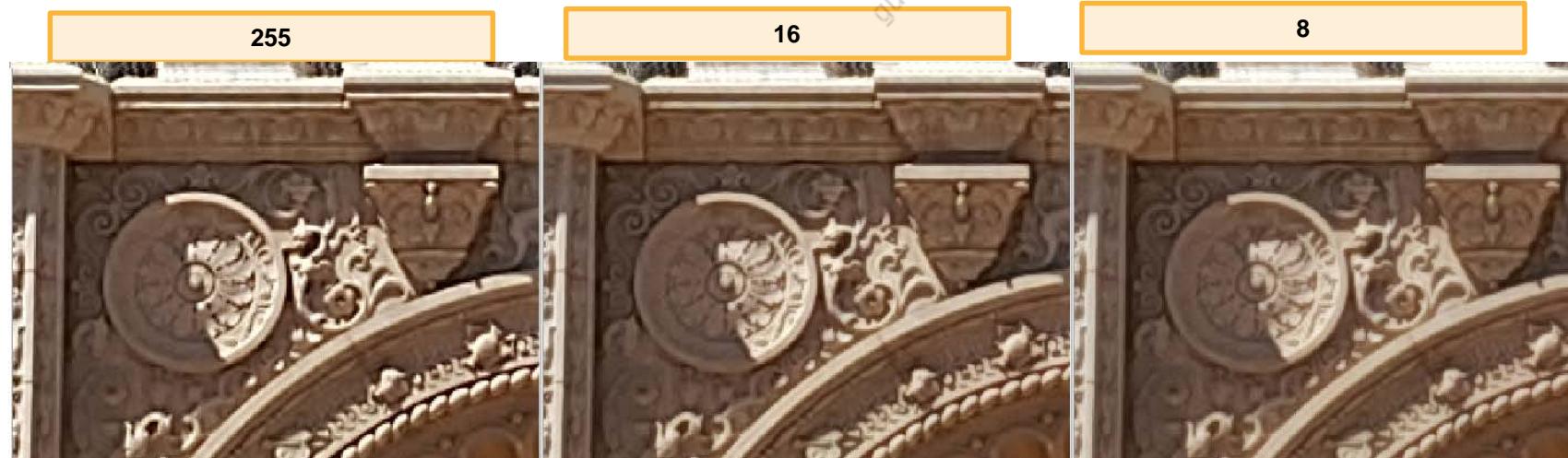
- Users can tighten clamps as needed at the final step of ASF tuning. Tighter clamps (that is smaller absolute value) can reduce halo, producing more natural images, especially for normal/low light conditions.

## Step 3.1: Clamp (cont.)

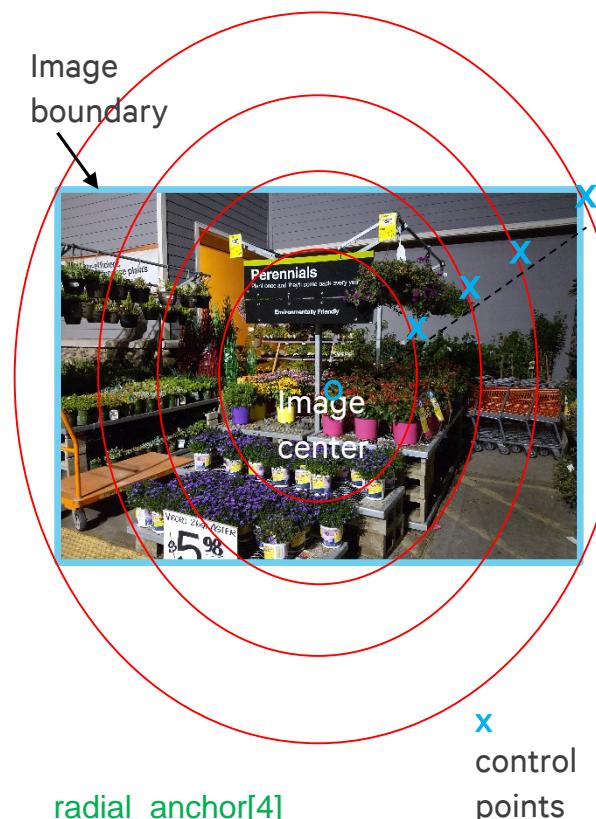
- layer\_1\_clamp\_UL



- layer\_1\_clamp\_LL



## Step 3.2: Radial



radial\_anchor[4]

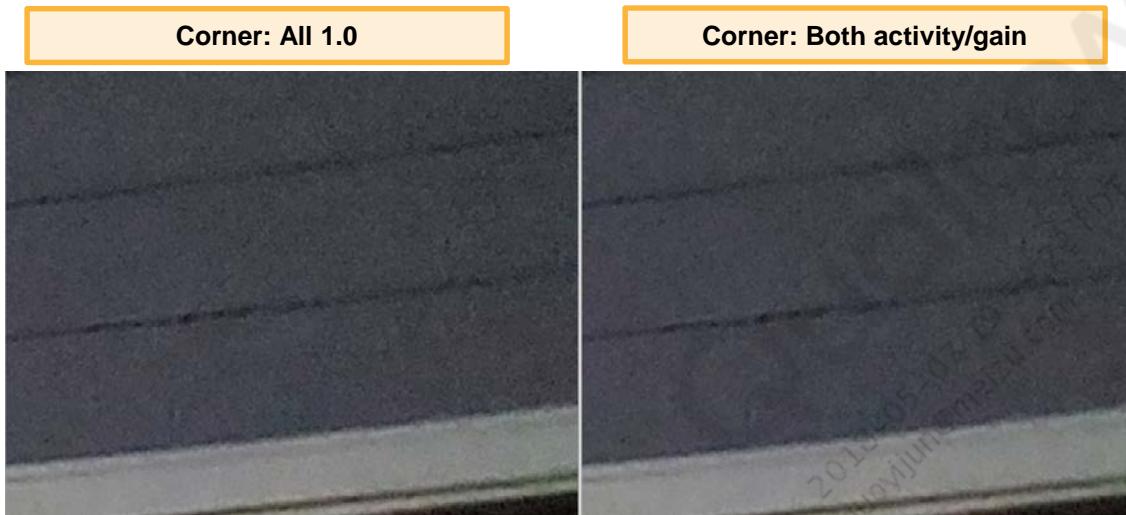
```
= {r1, r2, r3, r4}  
= {0.2500, 0.5, 0.75, 1.00}  
// percentage of the image
```

r4 is always at the corner which is 1

- **radial\_activity\_adj**
  - Correction factor for activity based on radial distance
- | Length | Default         | Min  | Max   | Higher value        | Lower value       |
|--------|-----------------|------|-------|---------------------|-------------------|
| 4      | From 1.0 to 0.6 | 0.0f | 7.96f | Stronger sharpening | Weaker sharpening |
- Apply to activity
  - {1.0, 1.0, 0.8, 0.6}
- **radial\_gain\_adj**
  - Correction factor for gain based on radial distance
- | Length | Default         | Min  | Max   | Higher value        | Lower value       |
|--------|-----------------|------|-------|---------------------|-------------------|
| 4      | From 1.0 to 1.0 | 0.0f | 7.96f | Stronger sharpening | Weaker sharpening |
- Apply to gain
  - {1.0, 1.0, 1.0, 1.0}
- Both parameters can be increased to enhance edge at corner

## Step 3.2: Radial (cont.)

- Center details are the same and corner noises are removed



`radial_activity_adj`  
 $\{1.0, 0.8, 0.6, 0.2\}$   
`radial_gain_adj`  
 $\{1.0, 0.8, 0.6, 0.2\}$



**Note:** Refer radial calibration document for detail

## Step 3.2: Radial (cont.)

`radial_activity_adj`

{1.0, 1.0, 1.0, 1.0}

`radial_gain_adj`

{1.0, 0.8, 0.6, 0.2}

`radial_activity_adj`

{1.0, 0.8, 0.6, 0.2}

`radial_gain_adj`

{1.0, 1.0, 1.0, 1.0}

`radial_activity_adj`

{1.0, 0.8, 0.6, 0.2}

`radial_gain_adj`

{1.0, 0.8, 0.6, 0.2}

Corner: Gain adjust only



Corner: Activity adjust only

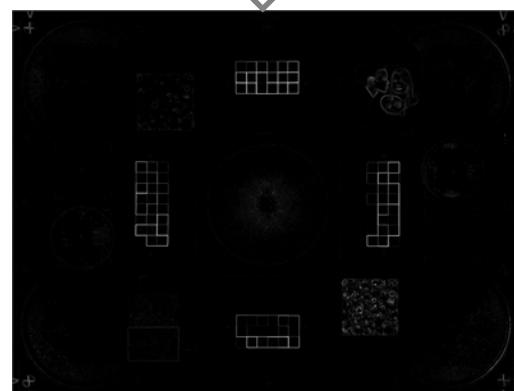
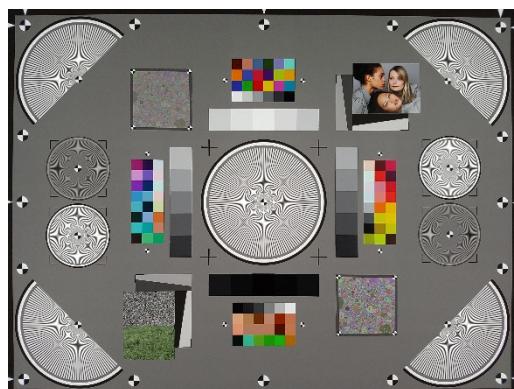


Corner: Both activity/ gain adjust



- To preserve details at image corner, using of activity adjustment for radial processing is recommended

## Step 3.3: gain\_chroma\_positive/negative



- **gain\_chroma\_positive/negative[chroma\_halo]**

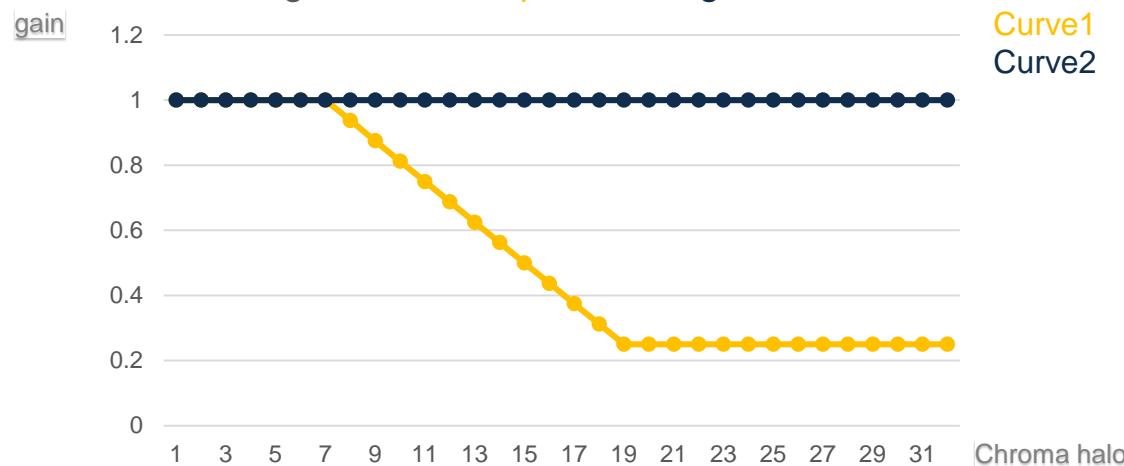
- Contrast-based sharpening gain LUT

Length	Default	Min	Max	Higher value	Lower value
32	From 1.0 to 0.25	0.004f	1.00f	Stronger sharpening	Weaker sharpening

- Strength control based on chroma edge

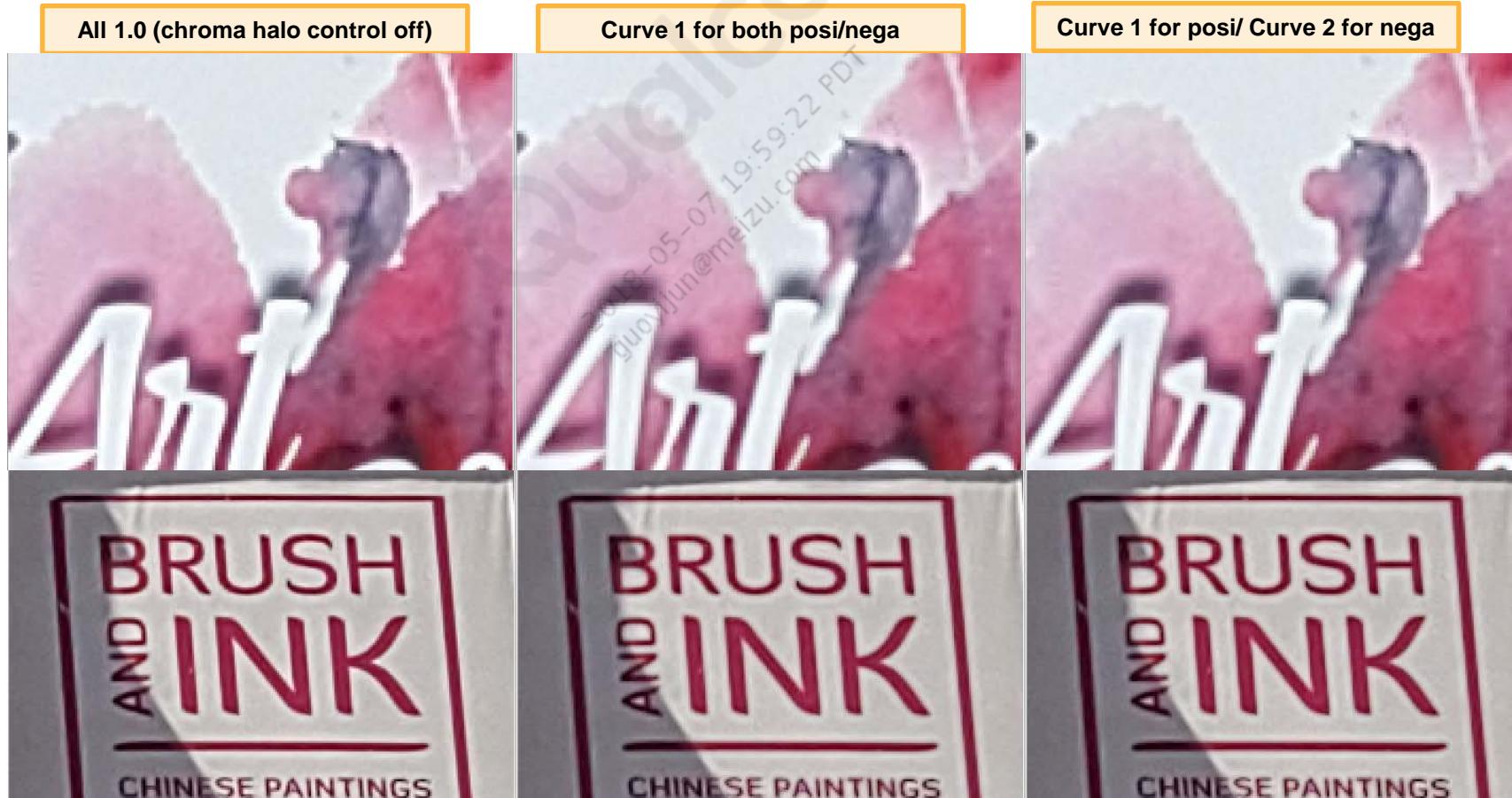
- Define which area could have halo around chroma
    - Reduce strength around chroma edge when there is no contrast (usually chroma to chroma)
    - Mainly used for positive halo control around chroma

### gain\_chroma\_positive/negative



## Step 3.3: gain\_chroma\_positive/negative (cont.)

- **gain\_chroma\_positive/negative**
  - Curve 1 for gain\_chroma\_positive/Curve 2 for gain\_chroma\_negative
  - Negative halo helps to keep contrast



## Step 3.4: Edge Alignment Threshold



Edge direction



Edge

Texture

Flat

- **flat\_threshold**

- Apply edge smoothing only when max of each edge value is larger or equal to flat\_threshold

Length	Default	Min	Max	Higher value	Lower value
1	8	0	255	Less edge detection - Less smoothing	More edge detection - More smoothing

- If you increase this threshold, edge will not be detected and edge smoothing will not work on flat

## Step 3.4: Edge Alignment Threshold (cont.)



similarity\_threshold (corner\_threshold)  
2 -> 10



More texture observed than edge  
Edge  
Texture  
Flat

- **similarity\_threshold (corner\_threshold)**

- Edge smoothing value is applied only when  $\text{max} \geq \text{min} * \text{similarity\_threshold}$  (corner\_threshold)

Length	Default	Min	Max	Higher value	Lower value
1	2	0	63.999f	Less edge detection - Less smoothing	More edge detection - More smoothing

- Gray color is edge similarity with other direction so it does not get smoothed. Considered as texture.

- **texture\_threshold (max\_smoothing\_clamp)**

- Edge smoothing value change is clamped by texture\_threshold (max\_smoothing\_clamp)

Length	Default	Min	Max	Higher value	Lower value
1	8	0	255	Stronger smoothing	Weaker smoothing

- Clamping of edge smoothing

# How to Coordinate with Other Modules

---

- Tune as a last module
- Enable ASF with defaults when tuning **NR** modules
  - To check artifacts and noise patterns more obvious
- To check if how much noise level is amplified by ASF
  - Maybe disable ASF during **NR** tuning
  - The starting level of activity LUT determines the high-frequency noise amplification level in ASF. This value needs to be considered together with **NR** modules.
- To remove zigzag pattern amplified by ASF
  - Tune **GIC** and **ABF** blocks to remove it in bayer firstly
- To enhance edge more strong with max ASF gain
  - Reduce **NR** strength near edge
- To enhance corner details
  - Reduce radial noise reduction in **NR** blocks firstly
- To make flat area noise pattern more natural
  - Use **Grain** block instead of increasing ASF gain on flat area

# Test Plan

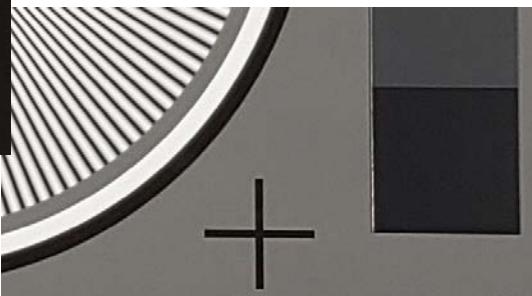
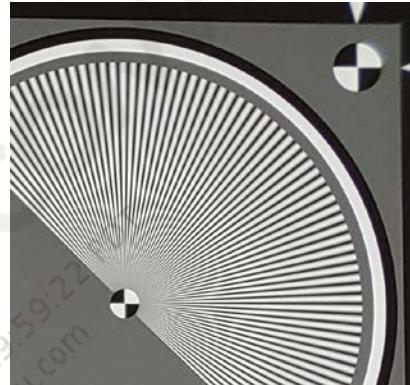
---

- **Enable/Disable flags**
  - layer\_1\_enable;
  - layer\_2\_enable;
  - radial\_enable;
  - contrast\_enable;
  - chroma\_gradient\_enable;
  - skin\_enable;
  - face\_enable;
  - edge\_alignment\_enable;
- If disabled
  - 1<sup>st</sup> layer totally disable
  - 2<sup>nd</sup> layer totally disable
  - do not apply gain for radial location
  - do not apply gain for contrast
  - do not apply gain for chroma gradient
  - do not apply gain for skin
  - do not use face detection data for skin
  - do not smooth edge
- **To bypass ASF using parameters**
  - layer\_1\_clamp\_UL/ layer\_2\_clamp\_UL = 0
  - layer\_1\_clamp\_LL/ layer\_2\_clamp\_LL = 0
  - layer\_1\_sp = 0
  - smoothing\_strength = 0
  - The effect is same as ASF disable

# Test Plan (cont.)

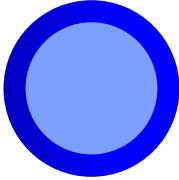
To confirm if this module works normally:

1. Take TE42 image
2. Check if flat area has no dots or artifact by this block
3. Check if details are good enough to see
4. Check if the edge is strong enough
5. Check if corner looks not noisy or less sharpened
6. Check if halo is not observed
7. Check if skin looks smooth



Test images:

- Resolution chart
- All light conditions (outdoor/ indoor/ lowlight/ very lowlight from 1500lux to 4lux)
  - This module is sensitive to noise. All light condition should be carefully tuned.
- High frequency components
  - trees, bushes, skies, roads, and rocks to check for artifacts

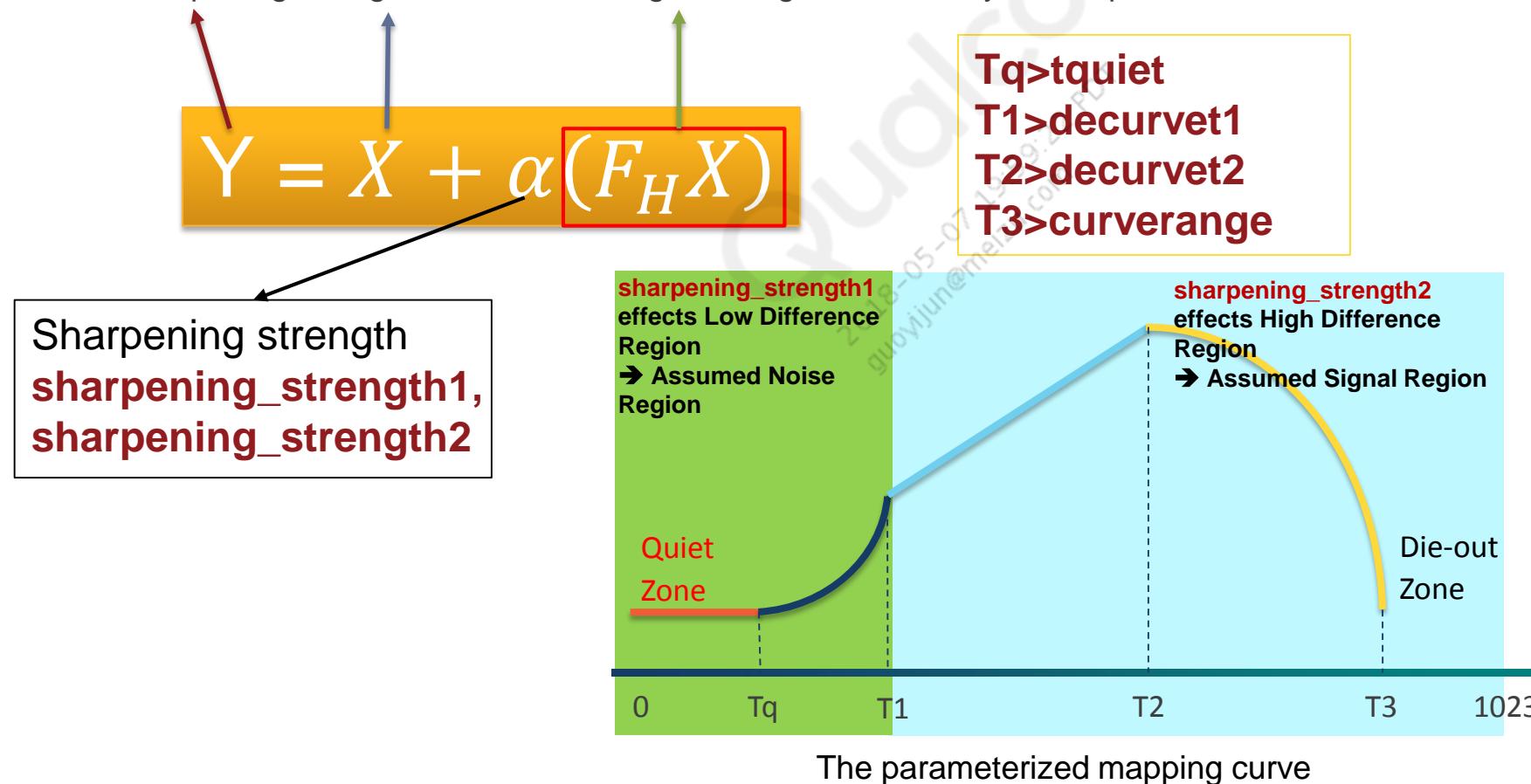


# Upscaler

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

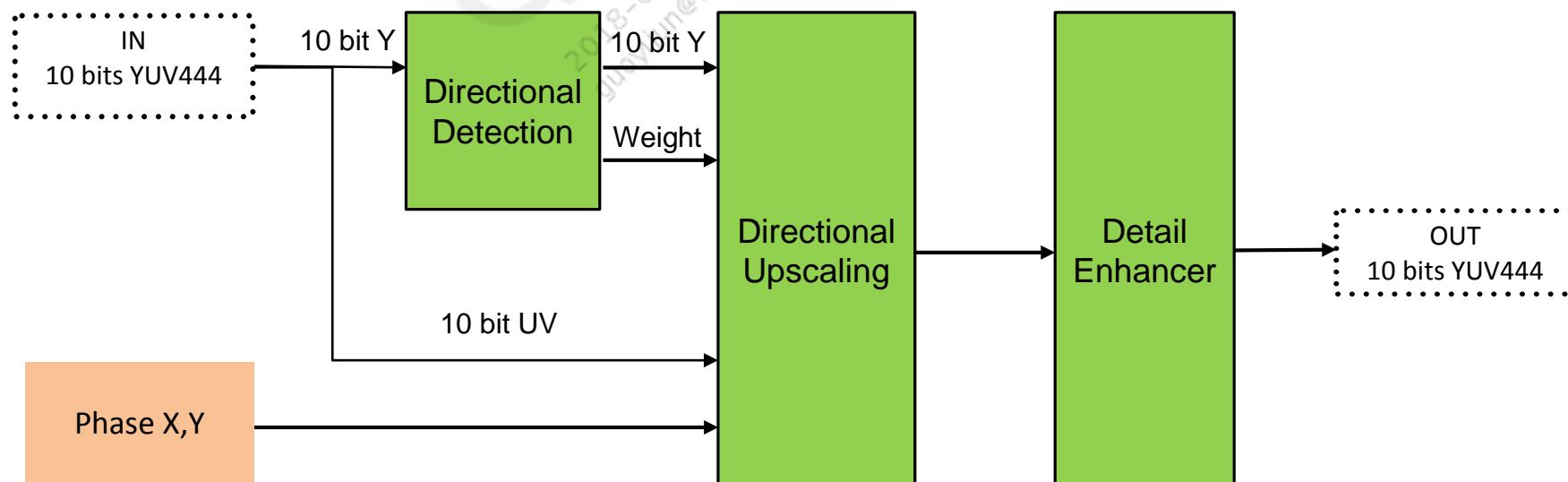
# Background

- Detail enhancer in upscaler
  - Detail enhancer, based on direction detection
    - Sharpening = Original + details in original image extracted by unsharp mask



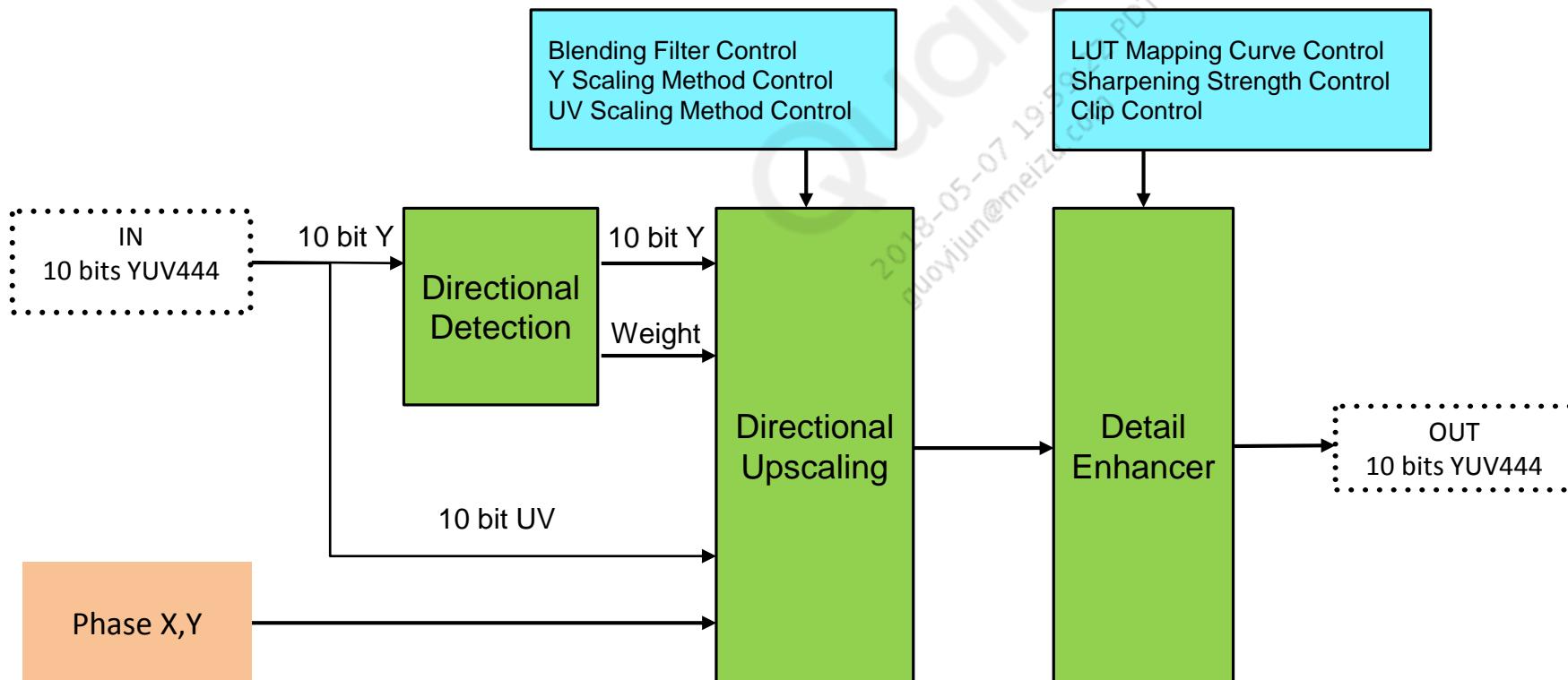
# Overview of Algorithm

- Upscaler 2.0 in PPS
  - Performing filtering/scaling/sharpening
  - Using directional filtering to improve quality
  - Sharpening/Smoothing image in detail enhancer
- Supported scaling range is 1x-20x upscaling
- 10bit YUV444 input and 10 bit YUV444 output



## Overview of Algorithm (cont.)

- 2D filtering controls: Blending filter, Y scaling method, UV scaling method
- Detail enhancer controls: LUT mapping curve, sharpening strength, clip



# Tuning Procedure

---

1. Set every parameter as default
2. Enable desired features in enable section
3. Tuning important parameters:
  - Upscaling ratio adjustment: **out\_width, out\_height**
  - Choice of edge-directed blending filter method: **blend\_filter**
  - Choice of YUV scaling method: **comp0\_filter\_method, comp1\_2\_filter\_method**
  - Detail enhancer LUT mapping curve adjustment: **tquiet, decurvet1, decurvet2, curverange**
  - High and low frequency region adjustment: **decurvet1**
  - Sharpening Strength adjustment: **sharpening\_strength1, sharpening\_strength2**
  - Prevent overshoot: **de\_clip\_shift**

# Step 1: Parameters for Enable Controls

---

- Parameters in enable section:
  - **upscale\_enable**: Upscaler enable
  - **enable**: Detail enhancer enable

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

## Step 2: Upscaling Ratio Adjustment

---

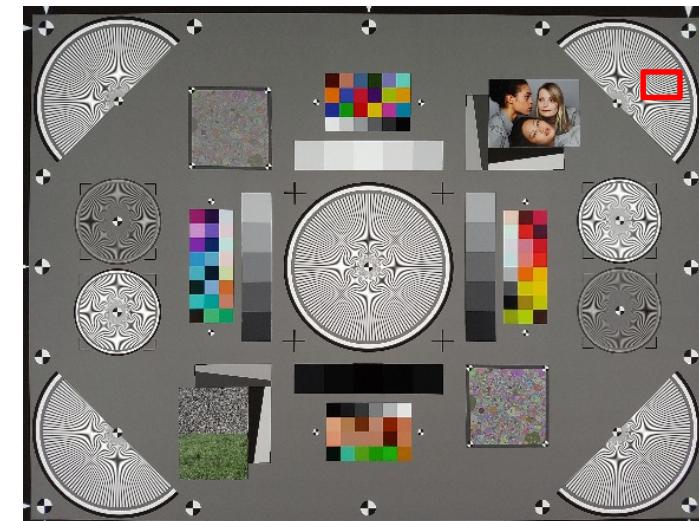
- **out\_width, out\_height**
  - Decide upscaling ratio
  - Supported scaling range is 1x-20x upscaling
  - Length: 1
  - Default: Same as input size
  - Effect: Get upscaled image what it is setting

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Step 2.1: Choice of Edge-directed Blending Filter Method

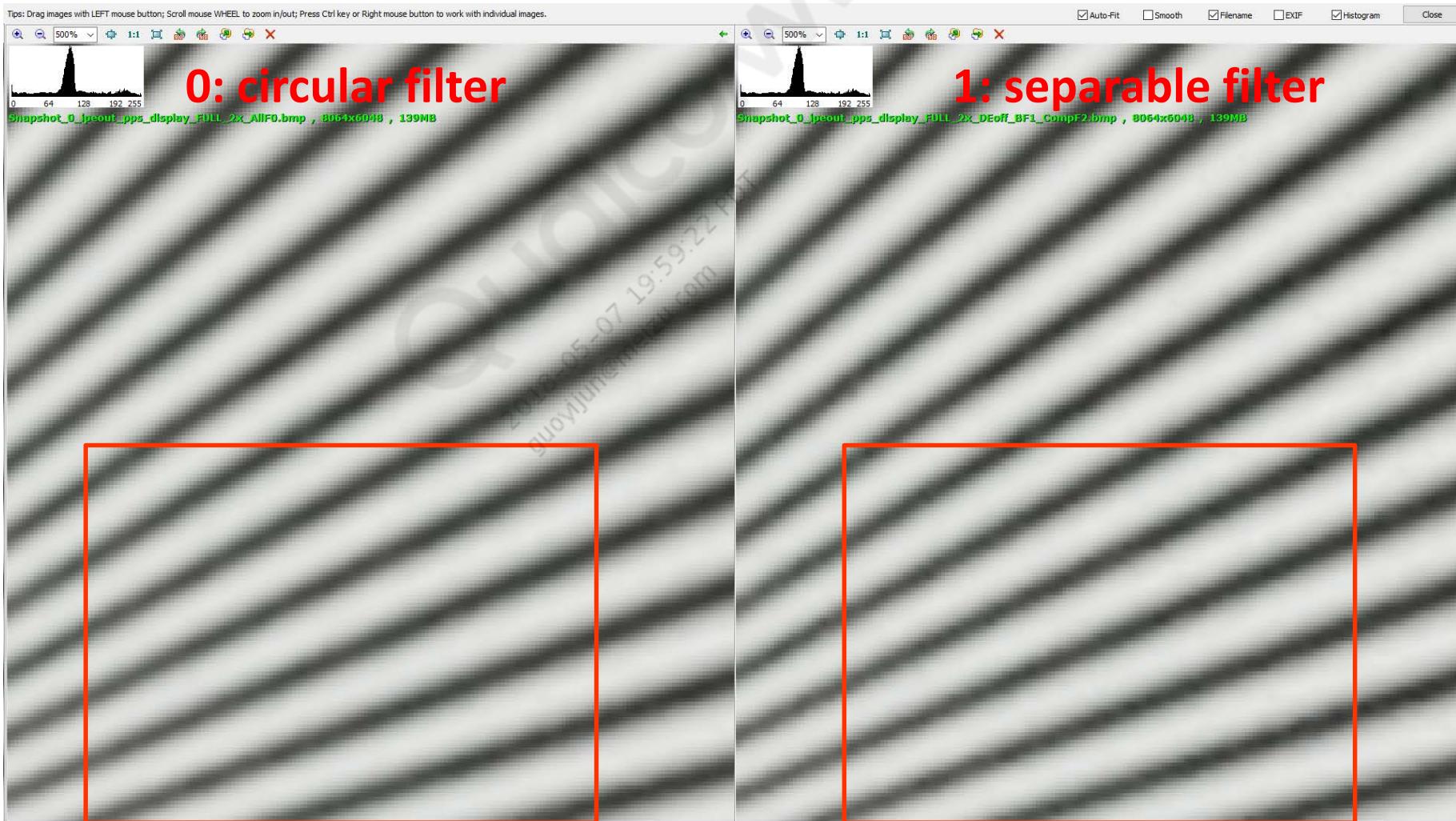
- **blend\_filter**

- Chooses the filter that is combined with directional filter to generate filter coefficients for edge-directed interpolation
- Length: 1
- Default: 0
- Min: 0; Max: 1
- Effect - 0: Circular filter, more directional edge preserved
- Effect -1:Separable filter, less directional edge preserved
- Calibration: None
- Suggestion: To preserve directional edge, set 0
- Example: With  $\times 2$  scaling ratio, circular filter vs. separable filter



## Step 2.1: Choice of Edge-directed Blending Filter Method (cont.)

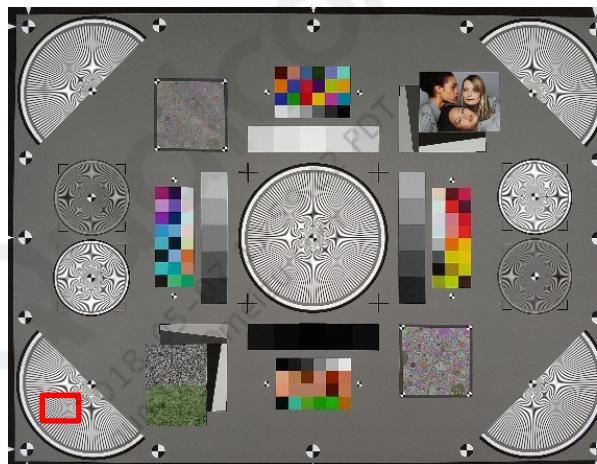
- blend\_filter



## Step 2.3: Choice of YUV Scaling Method

- **comp0\_filter\_method, comp1\_2\_filter\_method**

- specifies the scaling method that is applied to Y (**comp0\_filter\_method**), UV (**comp1\_2\_filter\_method**) component of YUV format
- Length: 1
- Default: 0
- Min: 0; Max: 2



- Effect - 0: 2D 4x4 filter > more directional edge preserved
- Effect - 1: 2D circular filter > less directional edge preserved than '0'
- Effect - 2: 1D Separable filter > less directional edge preserved than '1'
- Calibration: None
- Suggestion: To preserve directional edge, set 0
- Example: With x2 scaling ratio, 4x4 (directional) vs circular vs separable filter

## Step 2.4: Choice of Edge-directed Blending Filter Method

- **comp0\_filter\_method, comp1\_2\_filter\_method**



## Step 2.5: Detail Enhancer LUT Mapping Curve Adjustment

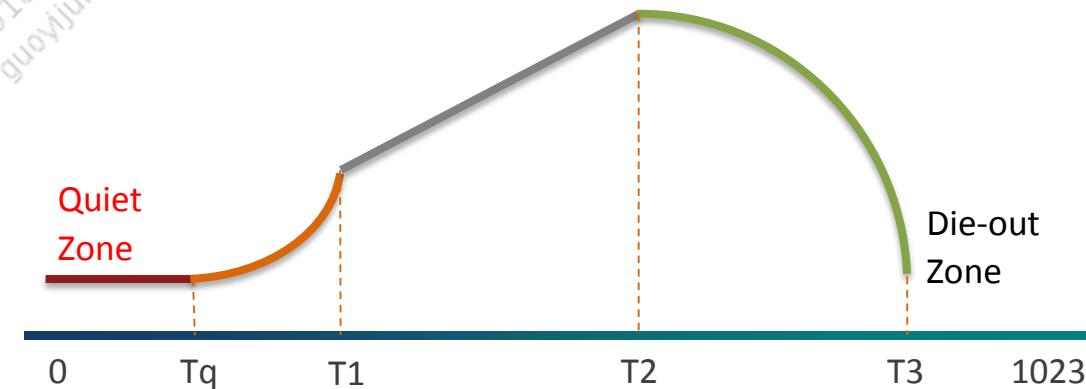
- **tquiet, decurvet1, decurvet2, curverange**

- Decide LUT curve for detail enhancer

Name	Length	Default	Min	Max
tquiet	1	20	0	255
decurvet1	1	70	tquiet	255
decurvet2	1	150	decurvet1	1023
curverange	1	400	decurvet2	1023

- Effect: Lower values make stronger sharpness
  - Calibration: None
  - Suggestion: Consider the range of input value

**Tq>tquiet  
T1>decurvet1  
T2>decurvet2  
T3>curverange**



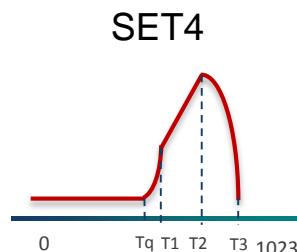
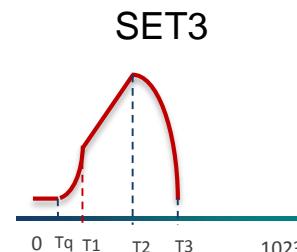
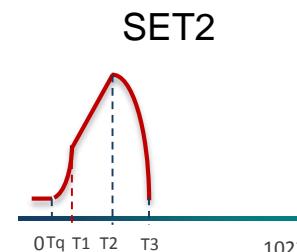
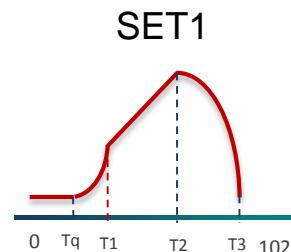
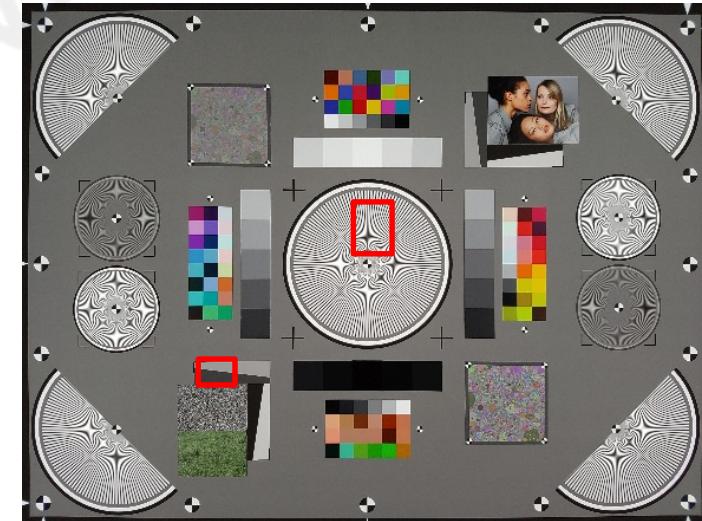
## Step 2.5: Detail Enhancer LUT Mapping Curve Adjustment (cont.)

- **tquiet, decurvet1, decurvet2, curverange**

- Examples

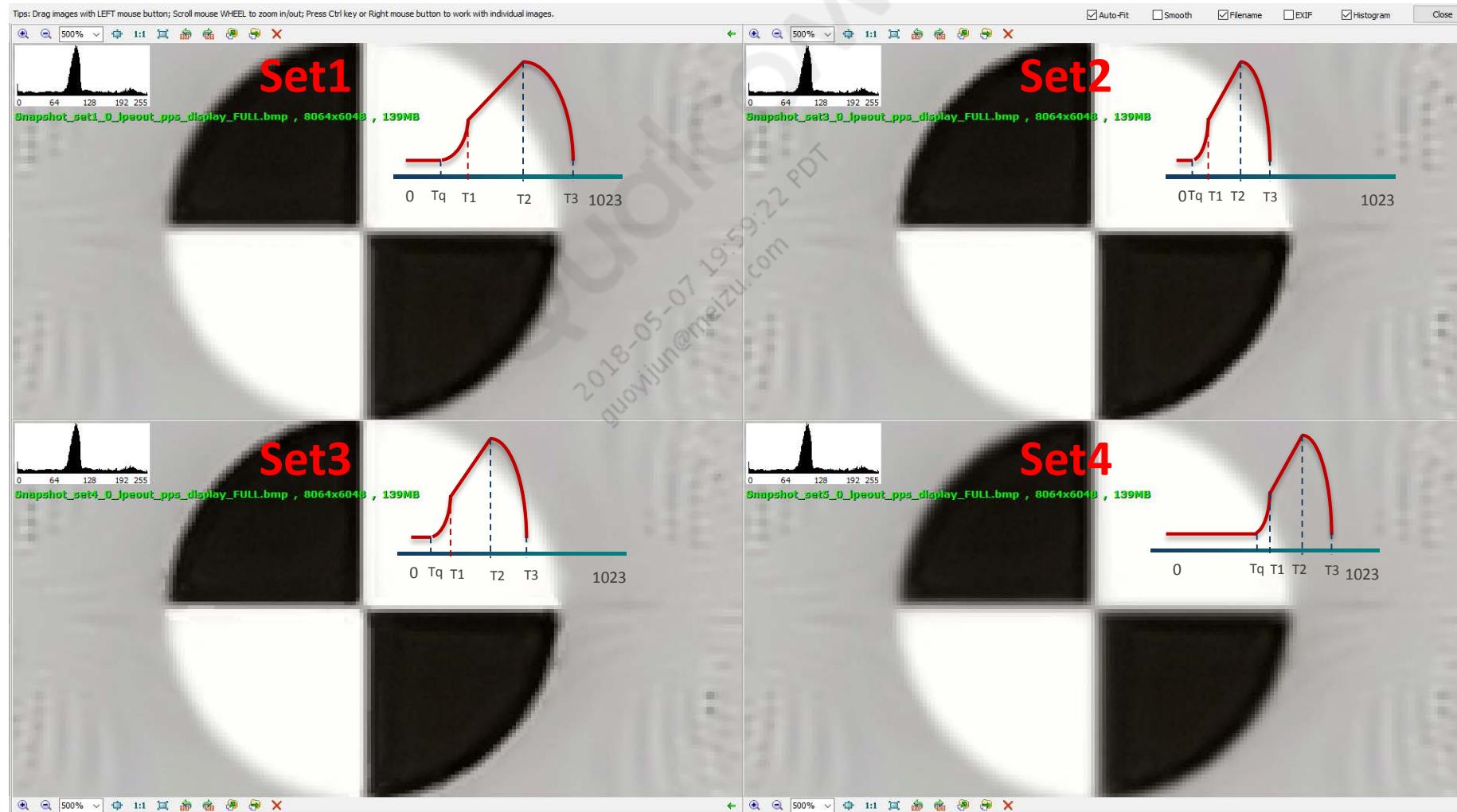
	SET1	SET2	SET3	SET4
Tquiet	10	10	10	400
Decurvet1	150	50	70	600
Decurvet2	550	250	450	850
curverange	950	450	700	950

- Test images on  $\times 2$  scaling ratio
  - Fix **Sharpening Strength1,2** values on 60



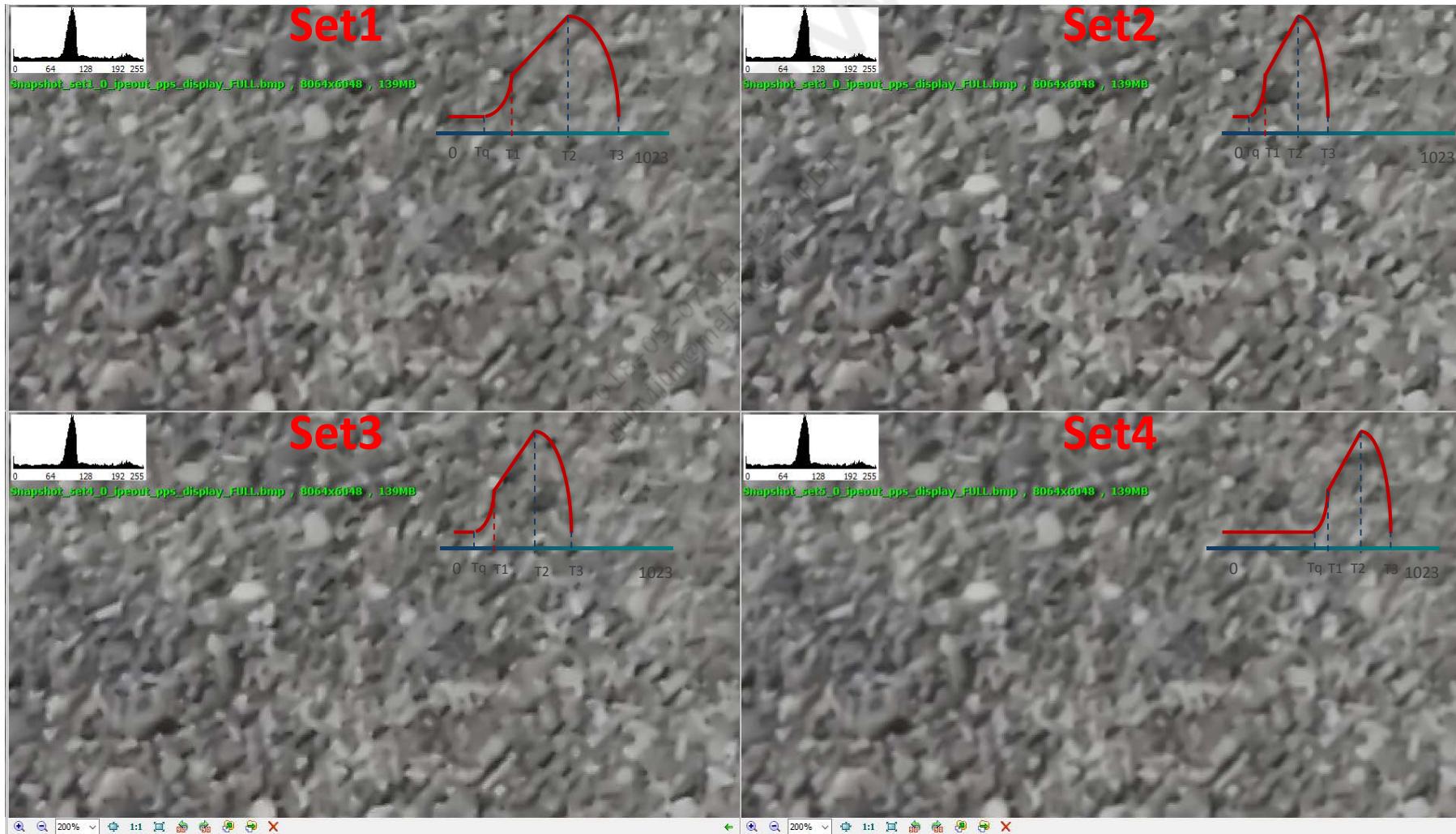
## Step 3.4: Detail Enhancer LUT Mapping Curve Adjustment

- **tquiet, decurvet1, decurvet2, curverange**



## Step 2.5: Detail Enhancer LUT Mapping Curve Adjustment

- **tquiet, decurvet1, decurvet2, curverange**

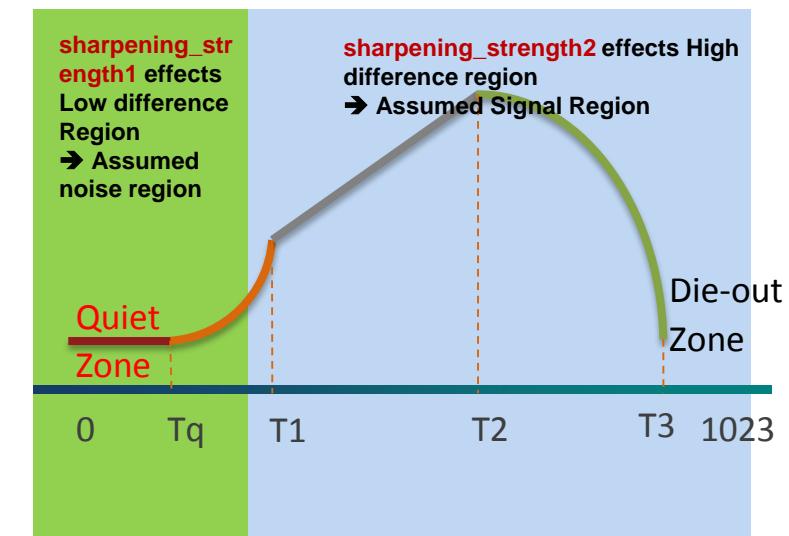


## Step 2.6: High and Low frequency Region Adjustment

- **decurvet1**

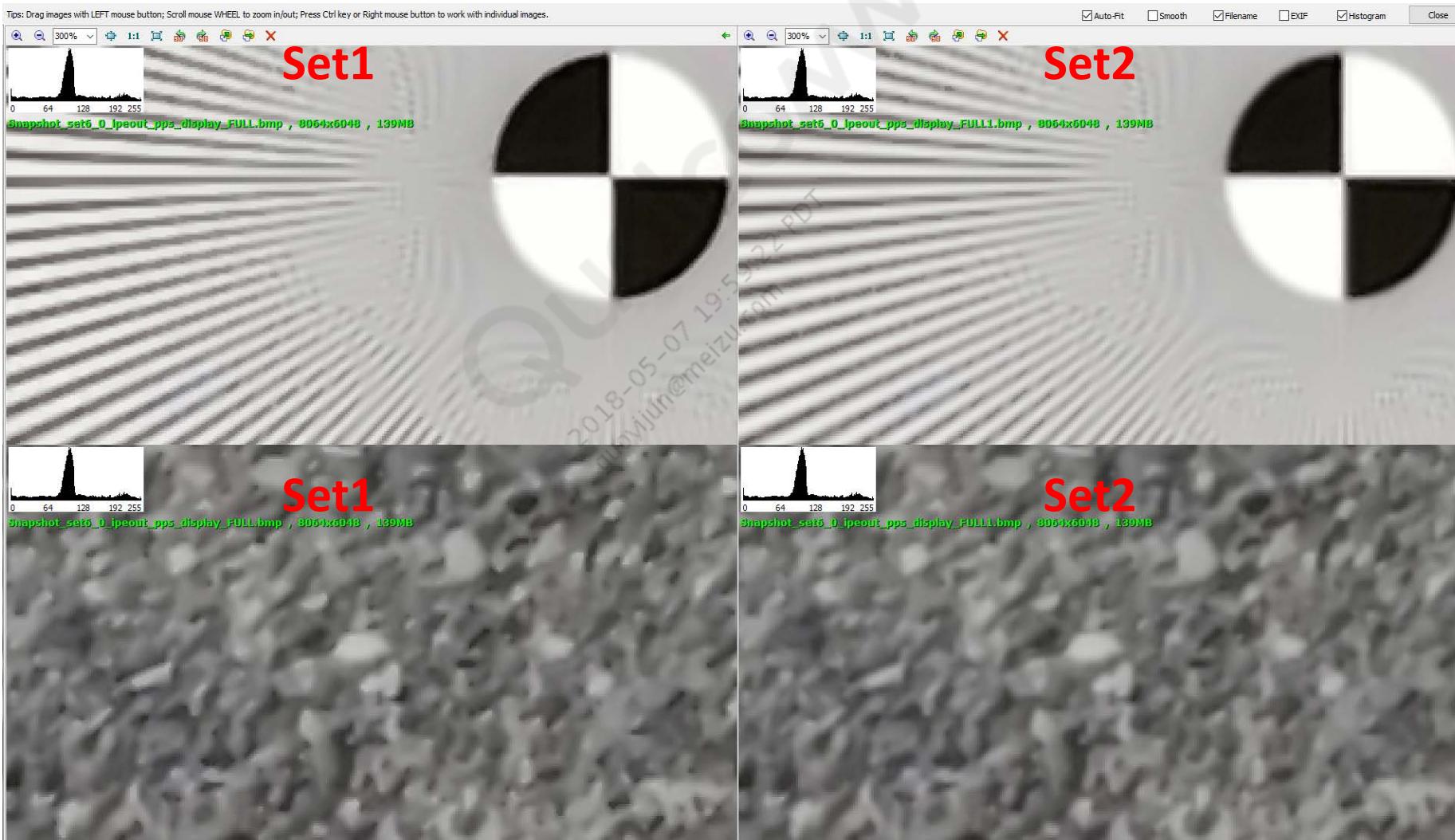
- Decide the image region between low and high frequency to control sharpness strength
  - ➔ In case of LUT pixel value is  $\leq$  **decurvet1, Sharpening\_Strength1 effect**
  - ➔ In case of LUT pixel value is  $>$  **decurvet1, Sharpening\_Strength2 effect**
- Length: 1
- Default: 70
- Min: 0; Max: 255
- Effect: The low and high difference regions are classified by **decurvet1**
- Calibration: None
- Suggestion: Consider low and high diff. region
- Examples:

	SET1	SET2
Tquiet	10	10
Decurvet1	50	90
Decurvet2	150	150
curverange	400	400
Sharpening_Strength_1	1	1
Sharpening_Strength_2	50	50



## Step 2.7: Sharpening Strength Adjustment

- **decurvet1**



## Step 2.8: High and Low Frequency Region Adjustment

- **Sharpening\_Strength1, Sharpening\_Strength2**

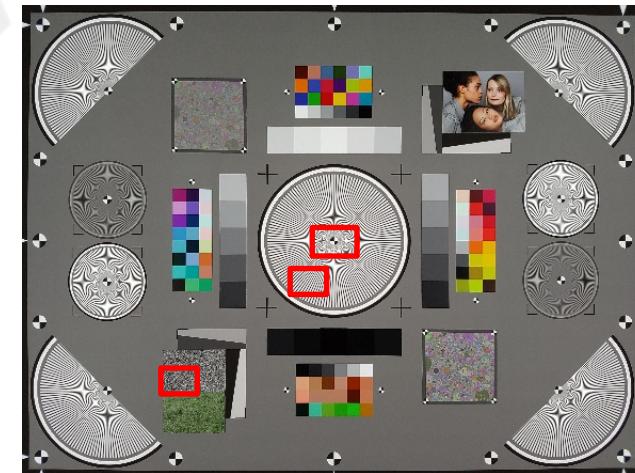
- Adjust strength according to **decurvet1**
- Length: 1
- Default: SS1:40 SS2:40
- Min: -32; Max: 160
- Effect(+value): Higher value stronger sharpening
- Effect(-value): Lower value stronger smoothing
- Suggestion:

**Sharpening\_Strength1** > only effect in the image region of low difference

**Sharpening\_Strength2** > only effect in the image region of high difference

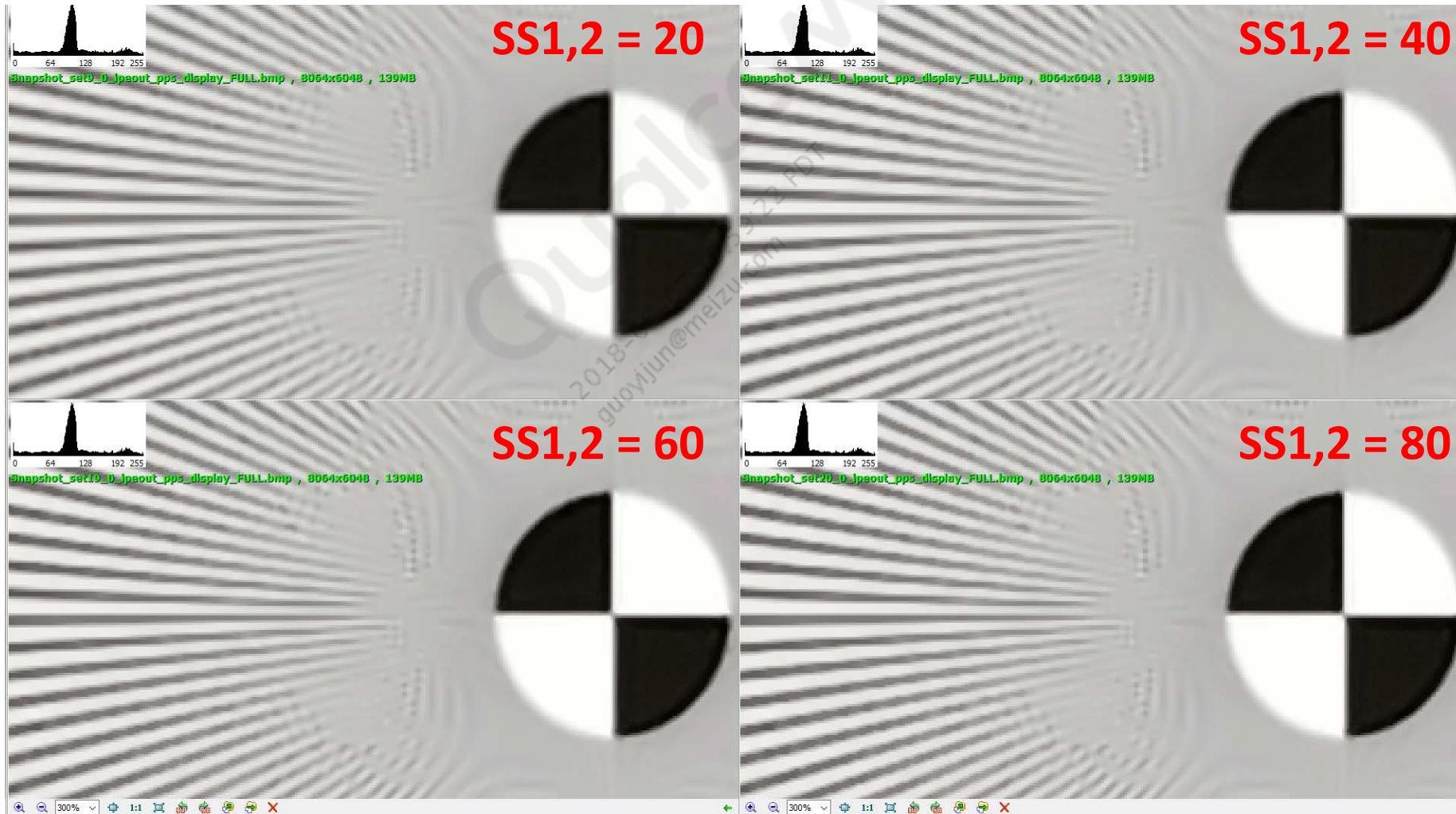
- Calibration: None
- Examples:

	SET1
Tquiet	10
Decurvet1	70
Decurvet2	150
curverange	400
Sharpening_Strength_1	??
Sharpening_Strength_2	??



## Step 2.9: Sharpening Strength Adjustment

- **Sharpening\_Strength1, Sharpening\_Strength2**



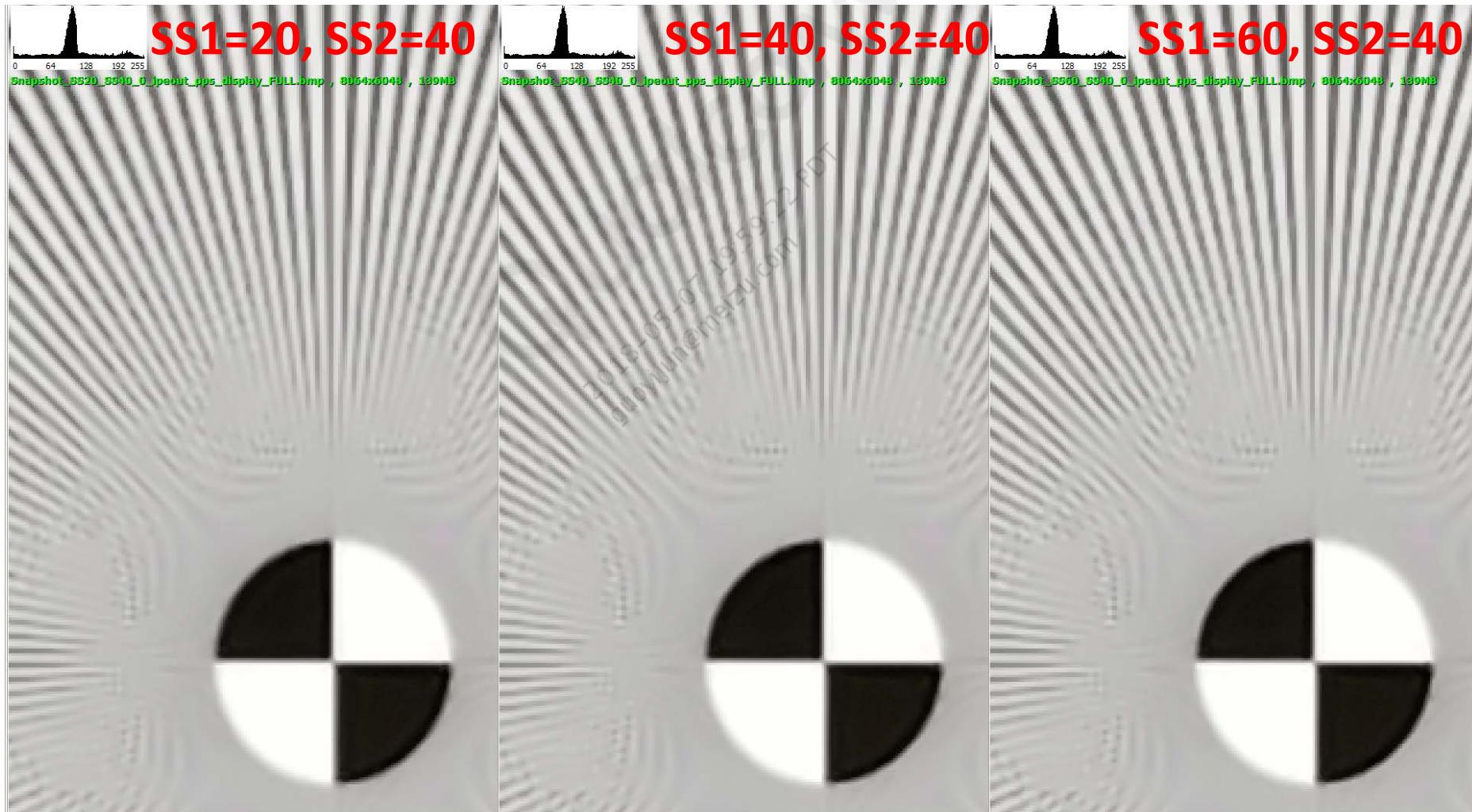
## Step 2.9: Sharpening Strength Adjustment (cont.)

- **Sharpening\_Strength1, Sharpening\_Strength2**



## Step 2.9: Sharpening Strength Adjustment (cont.)

- **Sharpening\_Strength1, Sharpening\_Strength2**



## Step 2.9: Sharpening Strength Adjustment (cont.)

- **Sharpening\_Strength1, Sharpening\_Strength2**



# Step 3: Prevent Overshoot

- **de\_clip\_shift**

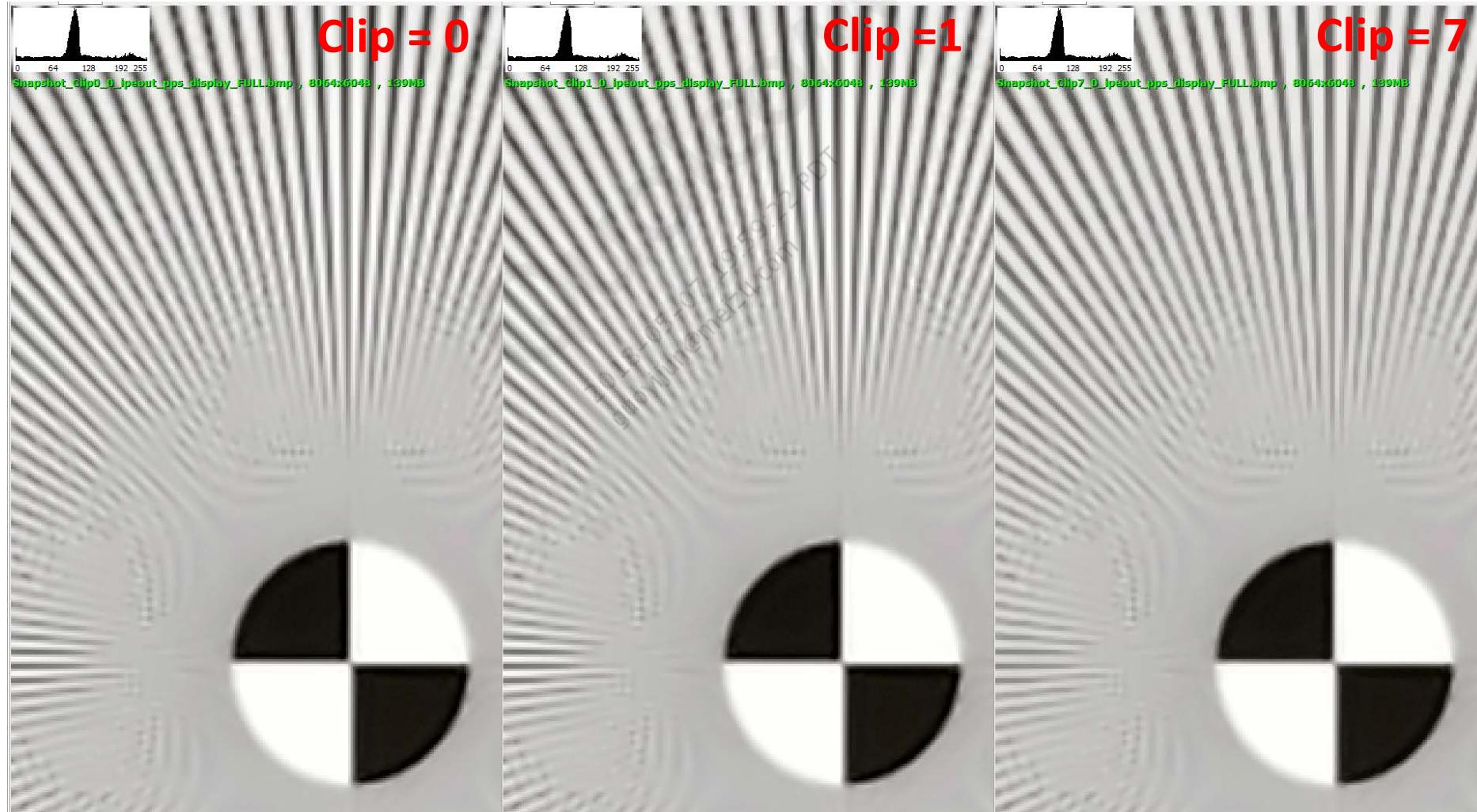
- Prevent overshoot problems
- Length: 1
- Default: 7
- Min: 0; Max: 7
- Effect: Higher value more prevent overshoot
- Suggestion: Consider the tradeoff sharpness
- Calibration: None
- Examples:

	SET1
Tquiet	10
Decurvet1	70
Decurvet2	150
curverange	400
Sharpening_Strength_1	40
Sharpening_Streingth_2	40
de_clip_shift	0,1,7



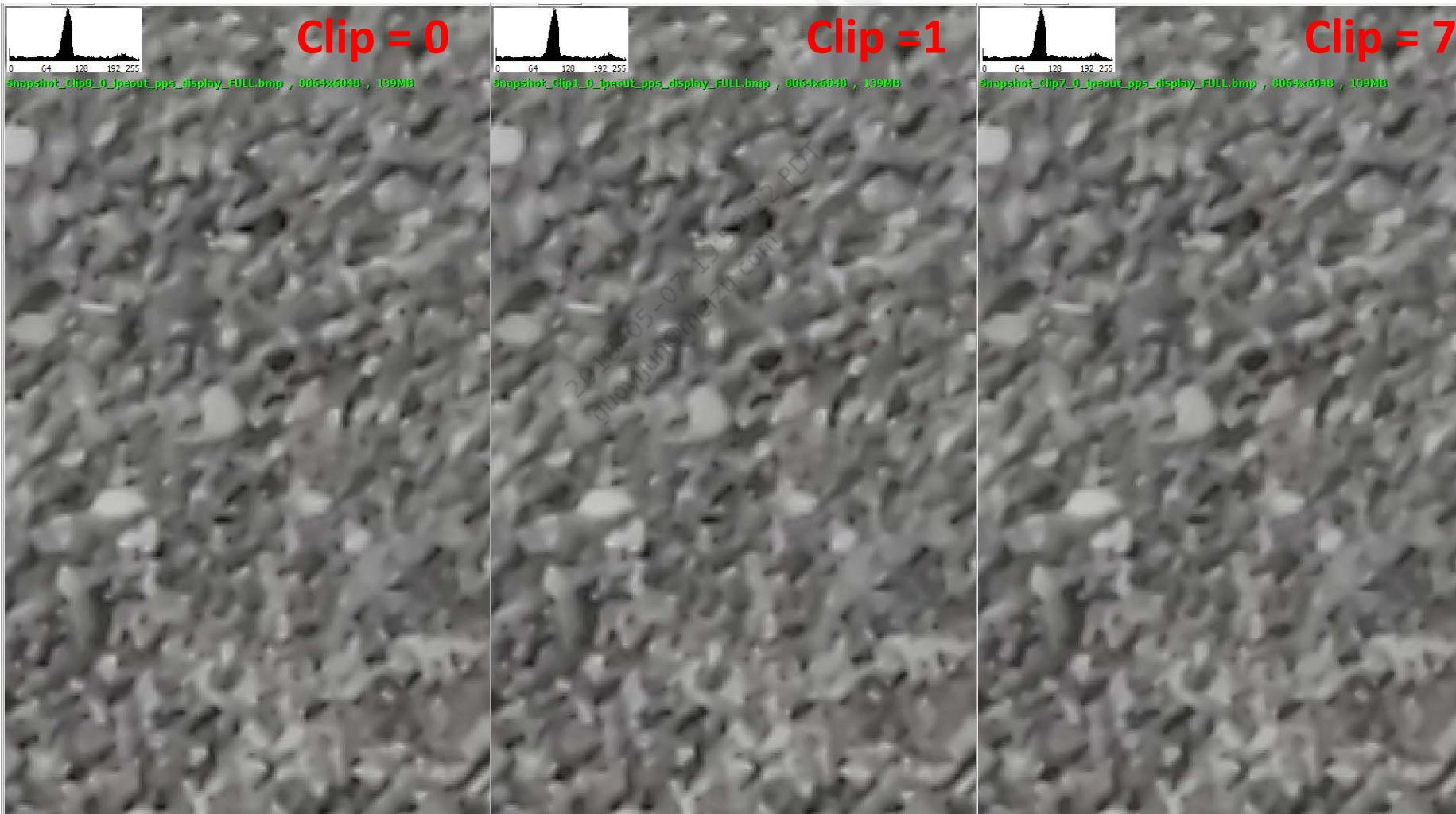
## Step 3: Prevent Overshoot (cont.)

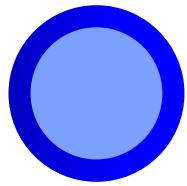
- **de\_clip\_shift**



## Step 3: Prevent Overshoot (cont.)

- **de\_clip\_shift**



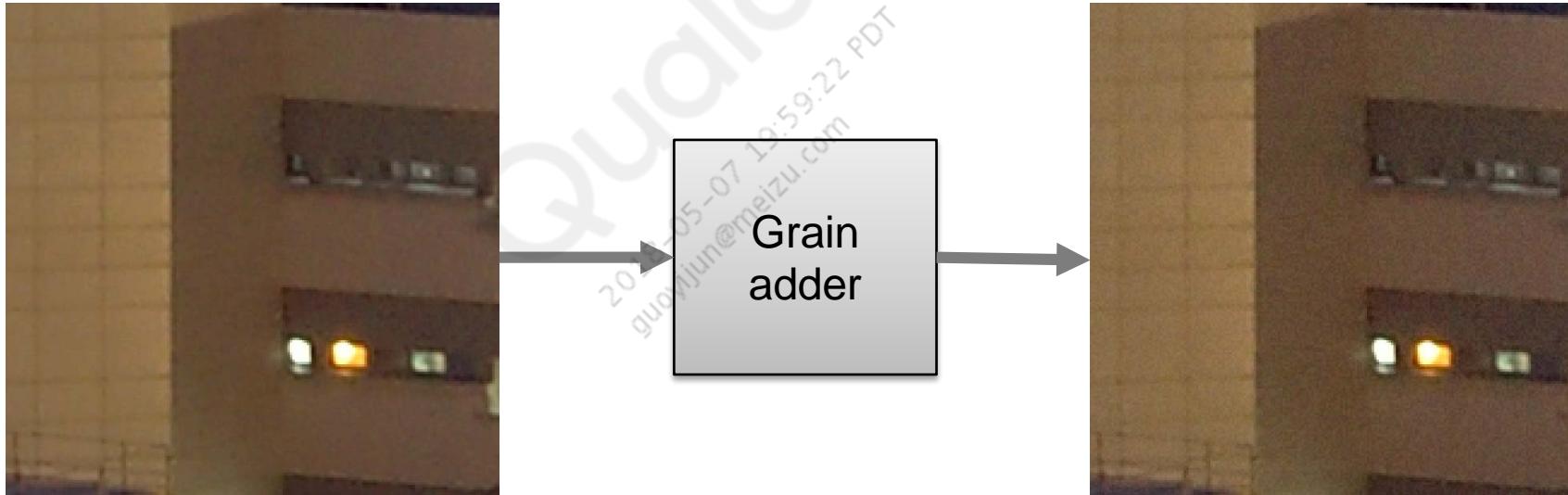


# Grain Adder (GRA)

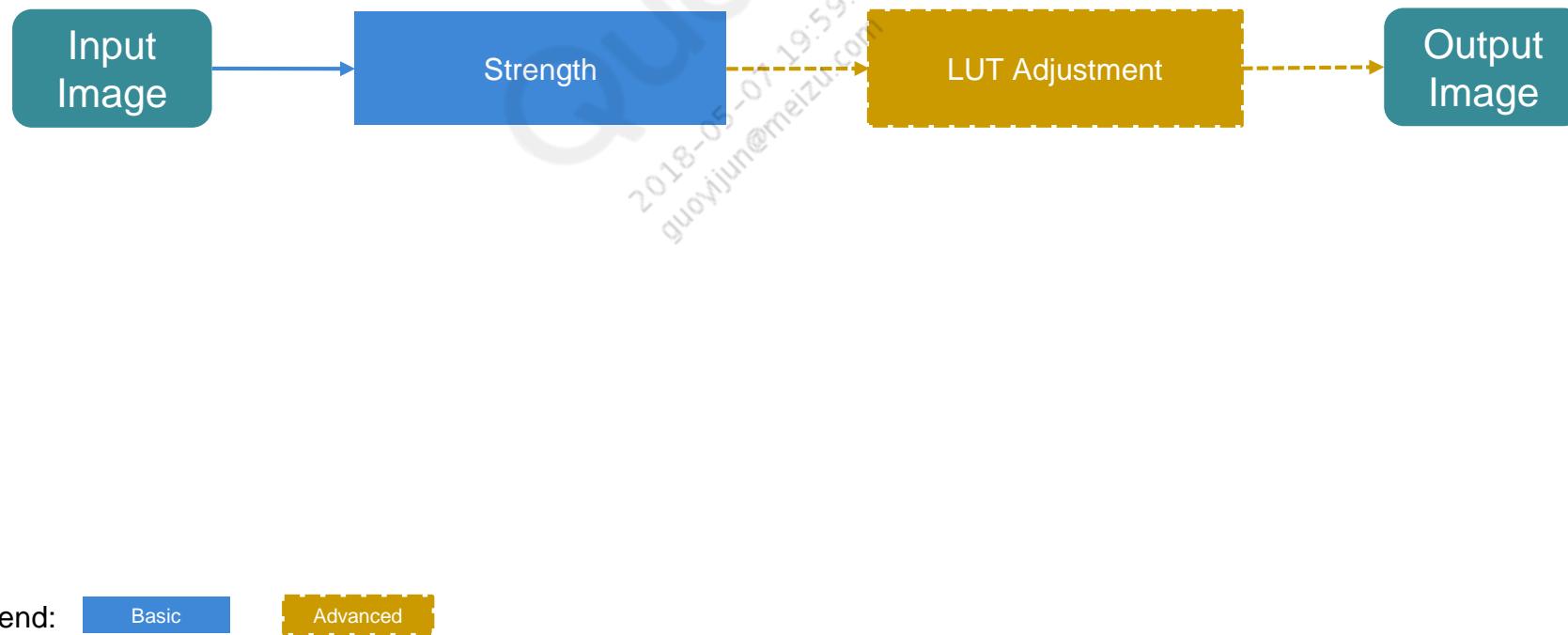
Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Grain Adder – Background

- Grain adder
  - Adds fine granularity to images
  - Can perform dithering



# Tuning Flow Diagram





# Tuning Steps – Strength Basic

- **Strength**
  - Controls the amount of grain added to the image
    - Higher value = More grain
    - Lower value = Less grain



Strength low



Strength mid



Strength high





# Tuning Steps – Strength Advanced

- **y\_weight\_lut**
  - 32 entries LUT
  - Change grain strength according to pixel luminance value
  - Higher value= More grain
- **cb\_weight\_lut** and **cr\_weight\_lut**
  - 32 entries each LUT
  - Change grain strength according to pixel chroma value
  - May be used for applying less grain for example for skin and sky colors
  - Higher value = More gain

2018-05-07 19:59:22 PDT  
guovjuncqiezu.com

# How to Coordinate with Other Modules

---

- Module tuning
  - Should be tuned as the last module in the pipe
- Other module influence
  - Making a change in noise reduction modules (ANR, HNR, TF) may require different grain strength

Qualcomm  
2018-05-07 19:59:22 PDT  
guovjun@meizu.com

# Questions?

For additional information or to submit technical questions, go to <https://createpoint.qti.qualcomm.com>