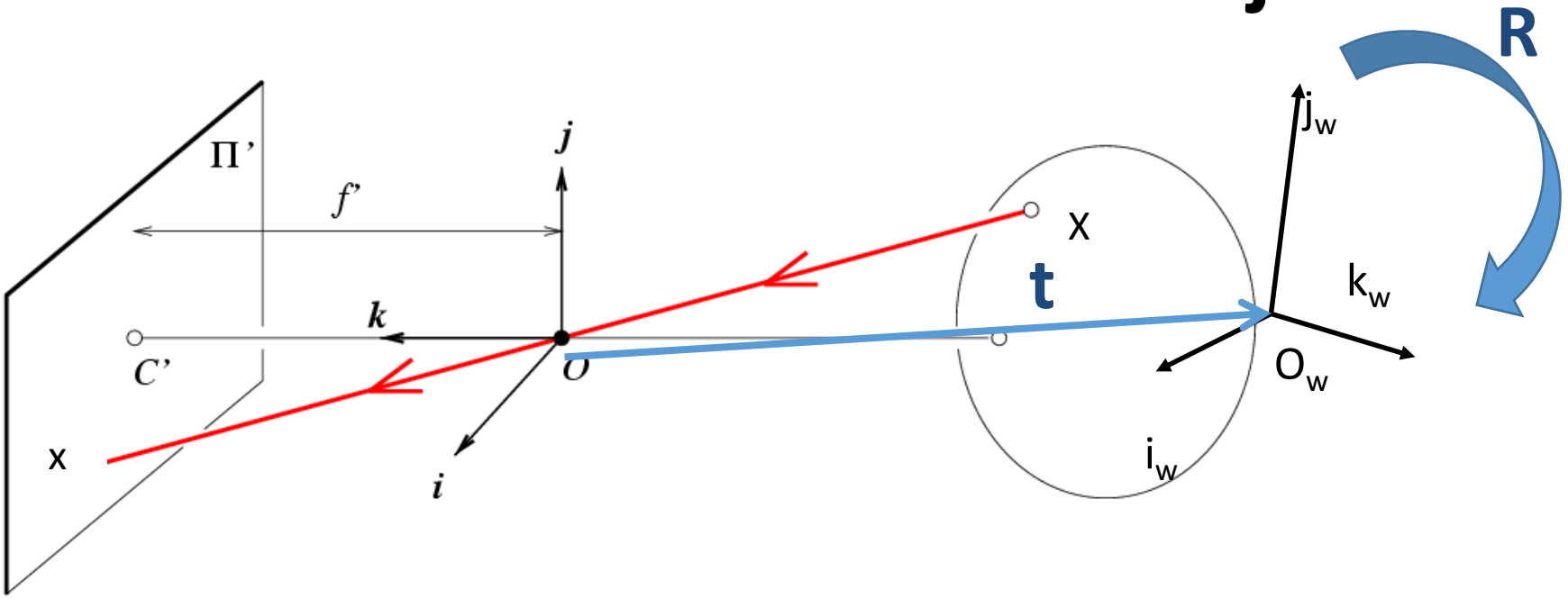
ISP: Image Signal Processing

Jianping Fan
Dept of Computer Science
UNC-Charlotte

Course Website:
http://webpages.uncc.edu/jfan/itcs5152.html

Review: Matrix for Geometric Projection

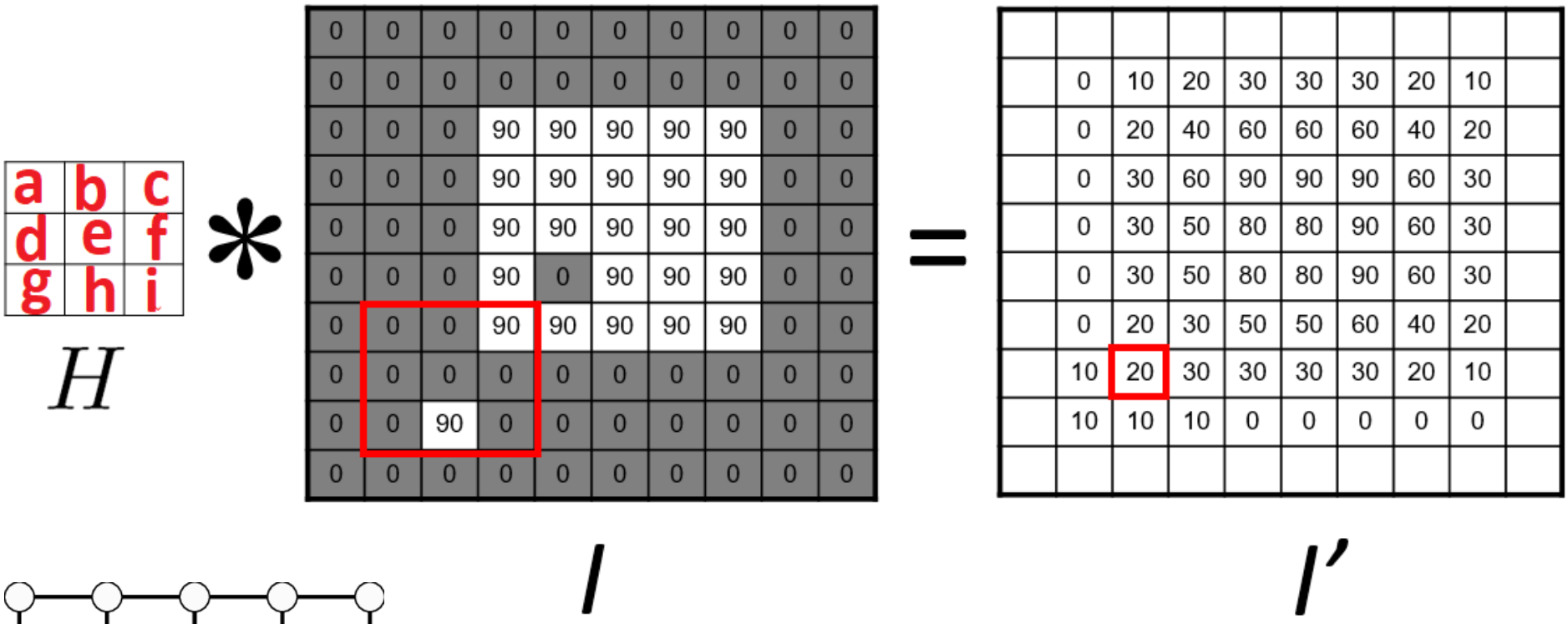


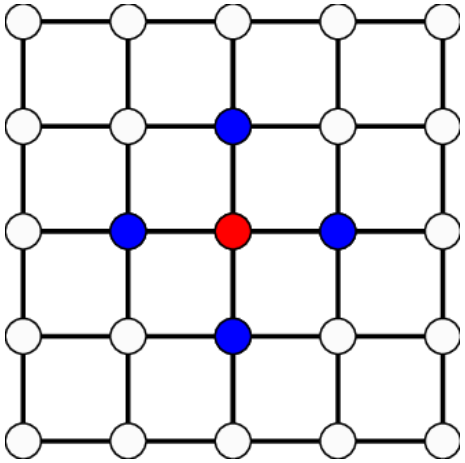
$$\mathbf{x} = \mathbf{K}\begin{bmatrix}\mathbf{R} & \mathbf{t}\end{bmatrix}\mathbf{X}$$

5 6

$$w\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

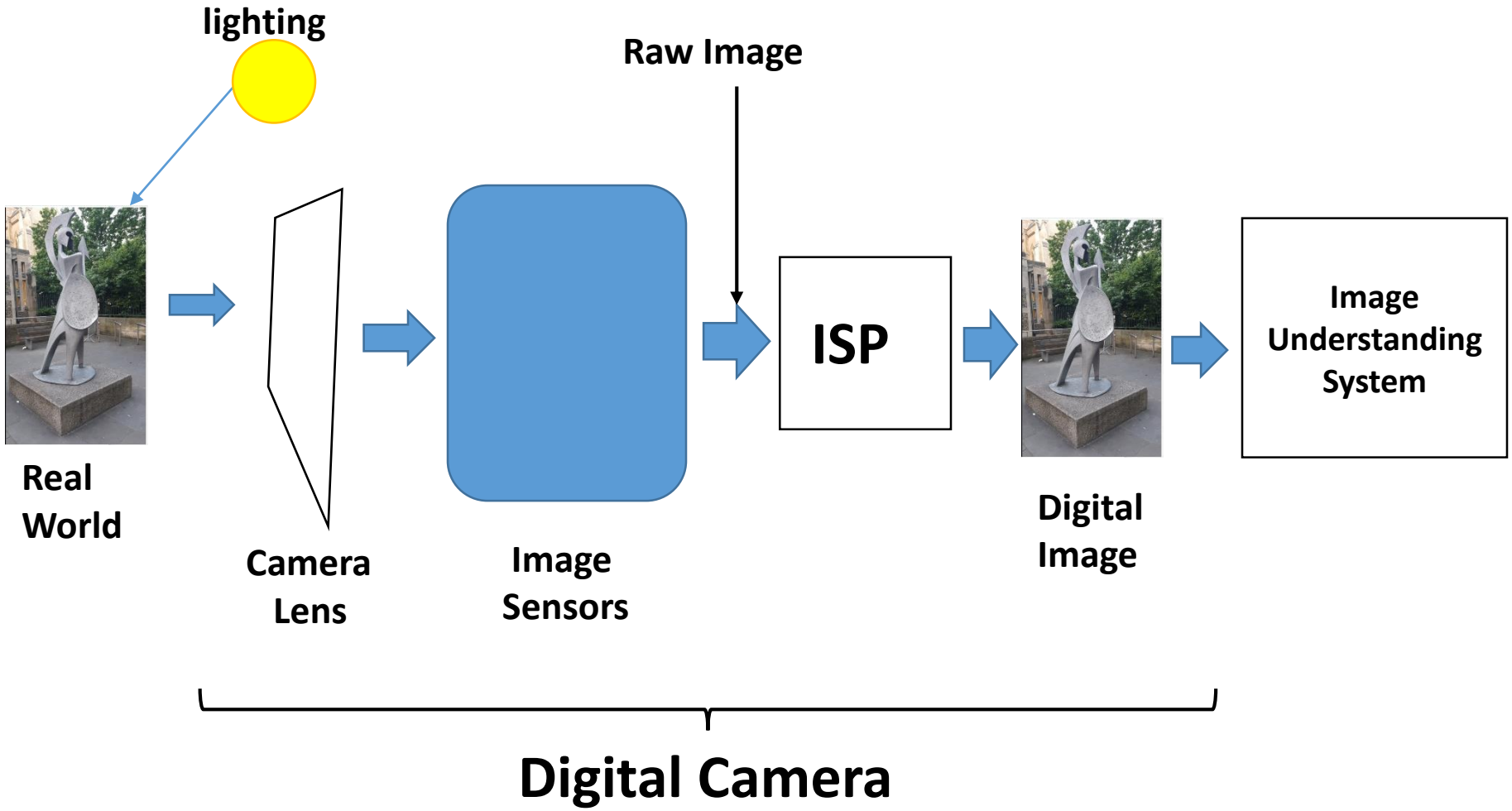
Review: Image Filter



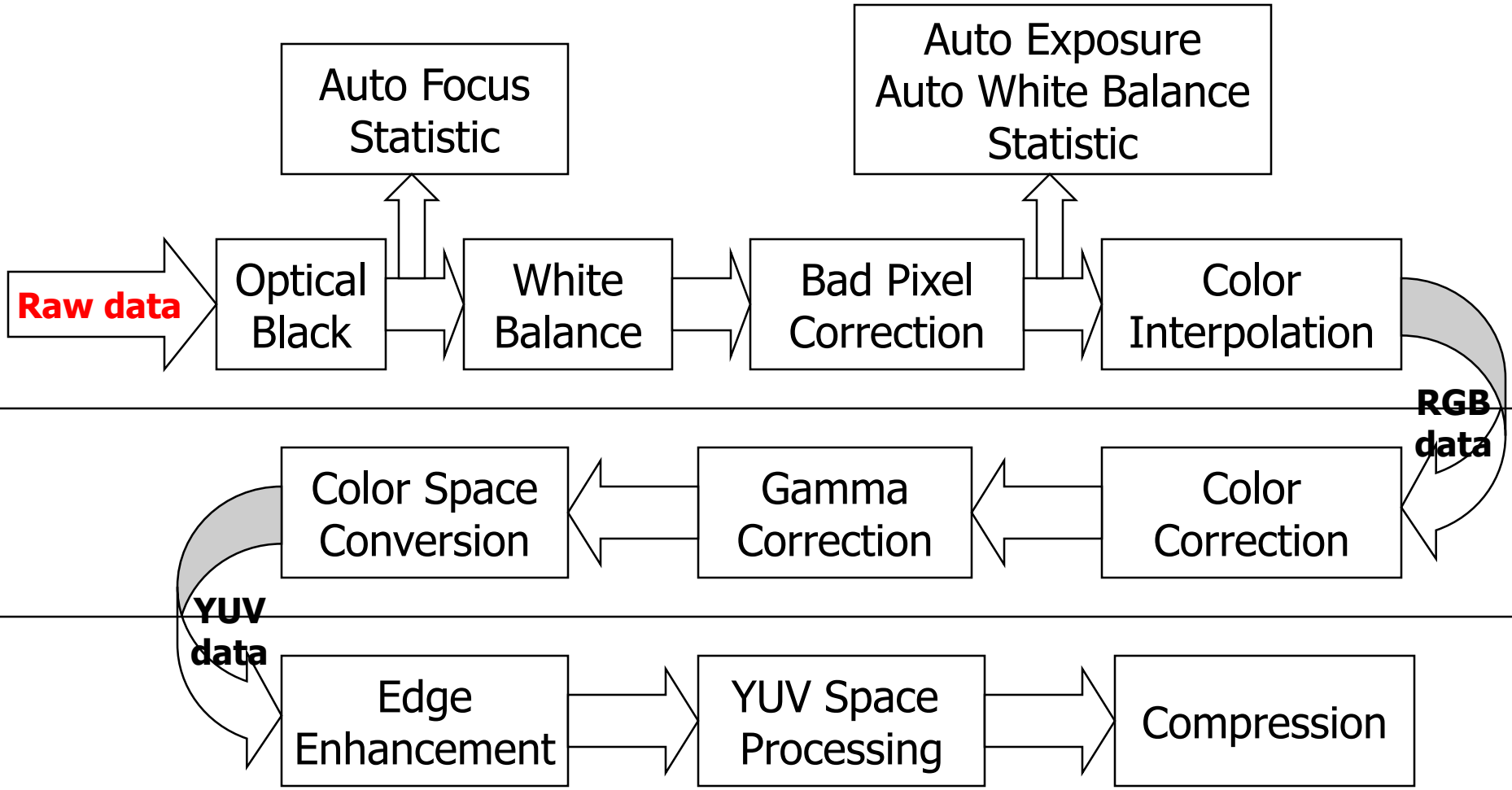


First project is also assigned!

Pipeline for Computer Vision Systems



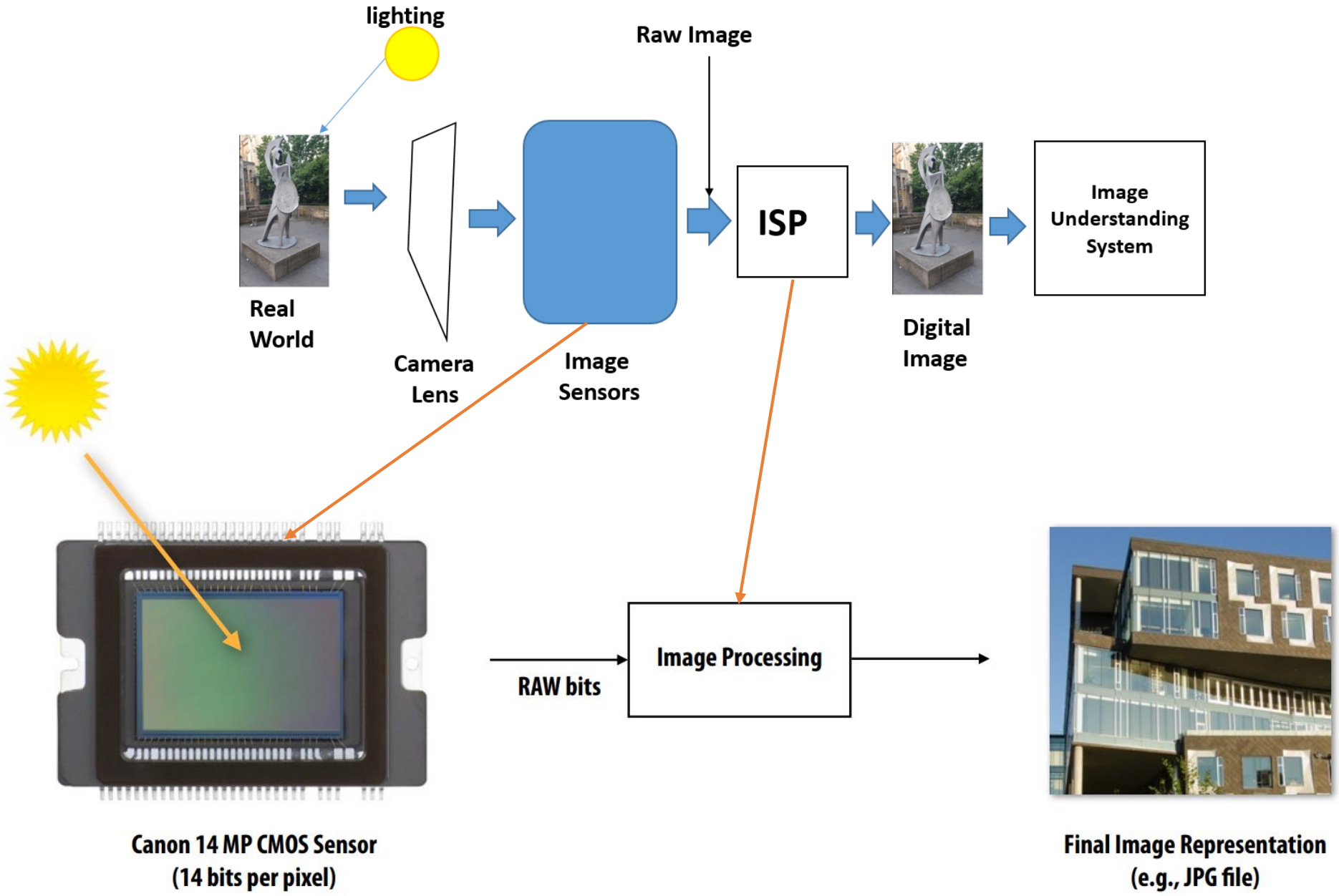
A Typical Image Pipeline for Digital Camera



www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

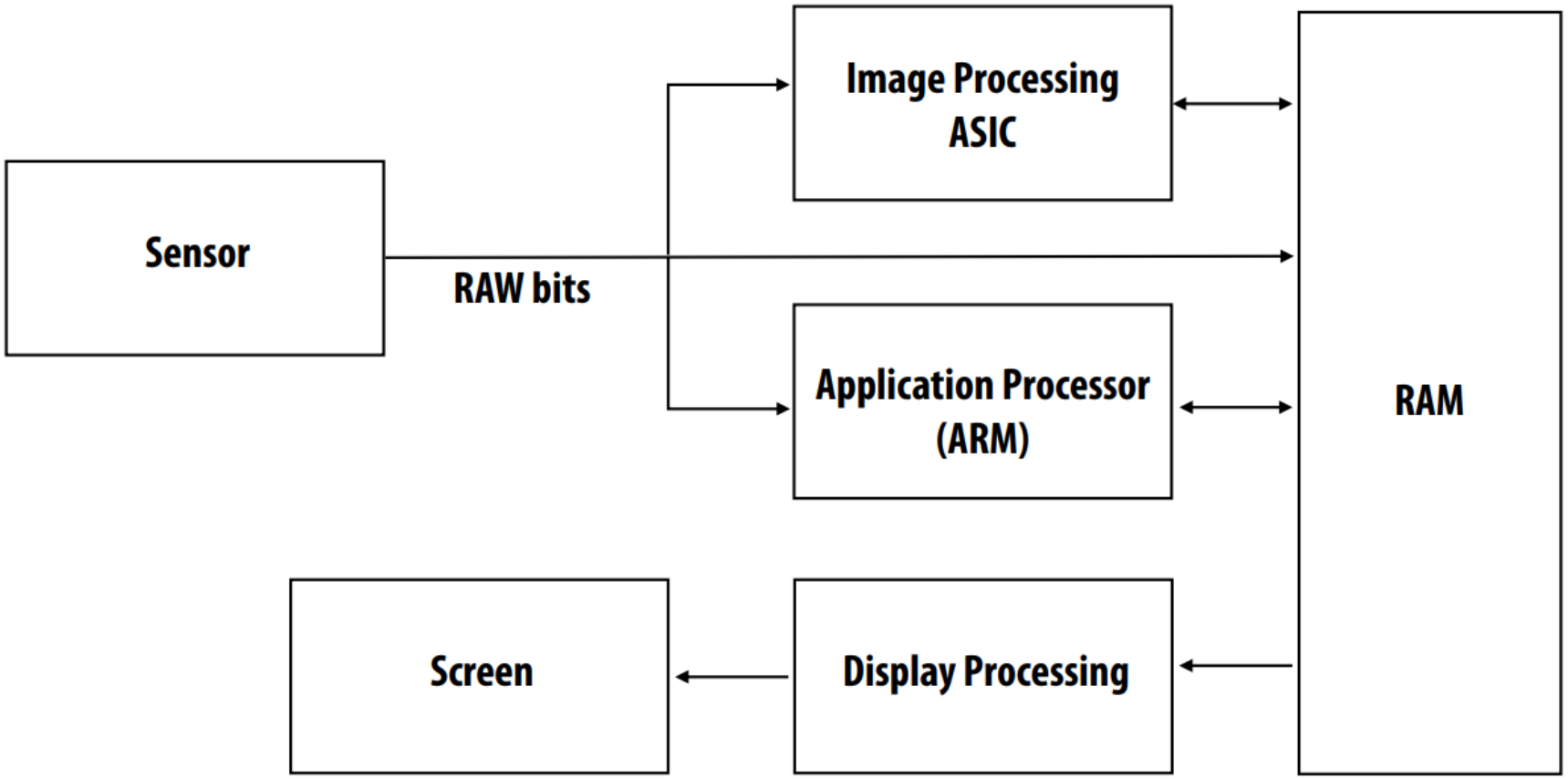
Pipeline Goals

- Correct hardware imperfections
- Facilitate inter-device communication
- Address appearance (aesthetics) issues
    - Reproduce an image whose appearance matches the original up to display intensity and gamut limitations, or
    - Make a nice picture, guided by the original
    - Preserve certain image metrics (edges, geometry)



Kayvon Fatahalian, Graphics and Imaging Architectures (CMU 15-869, Fall 2011)

Generic camera: system overview



Kayvon Fatahalian, Graphics and Imaging Architectures (CMU 15-869, Fall 2011)

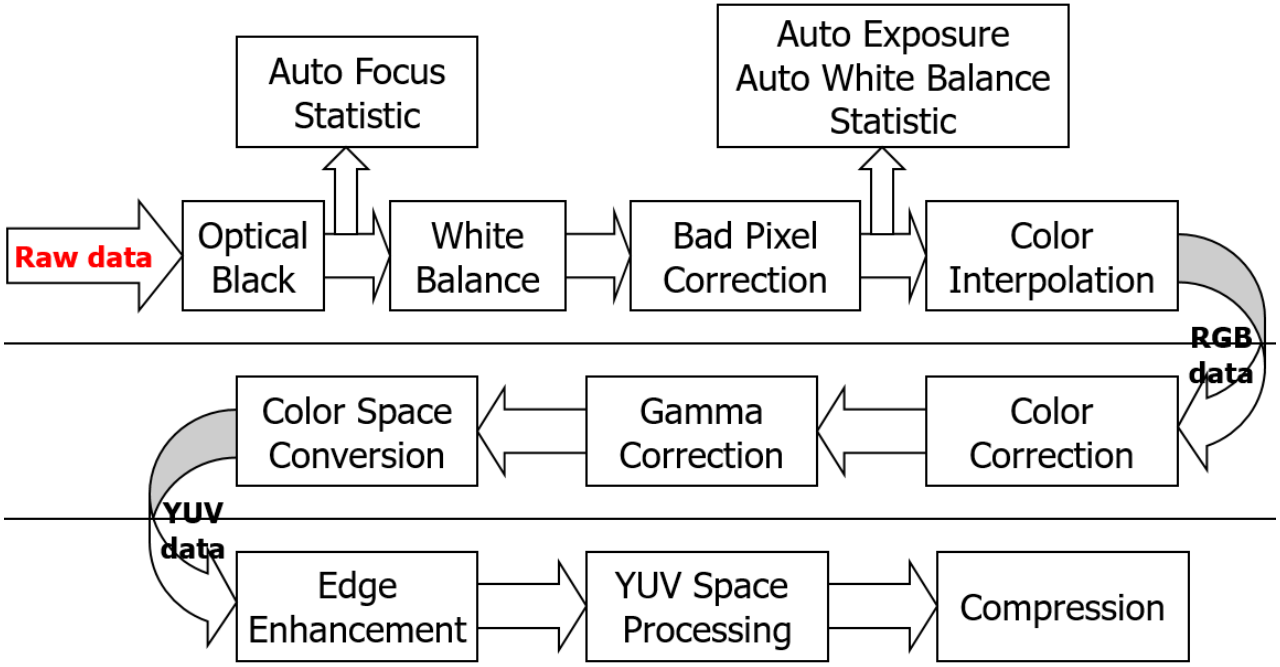
Image Processing Pipeline(ISP) Overview

- From **sensor raw data** to **final high-quality image**
- Targeted at **matching human perception**
- Pipeline approach (step-by-step & back-and-forth)
- Linear or nonlinear
- Color depth consideration
- **Calibration, Compensation, Correction, and Concealment**

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

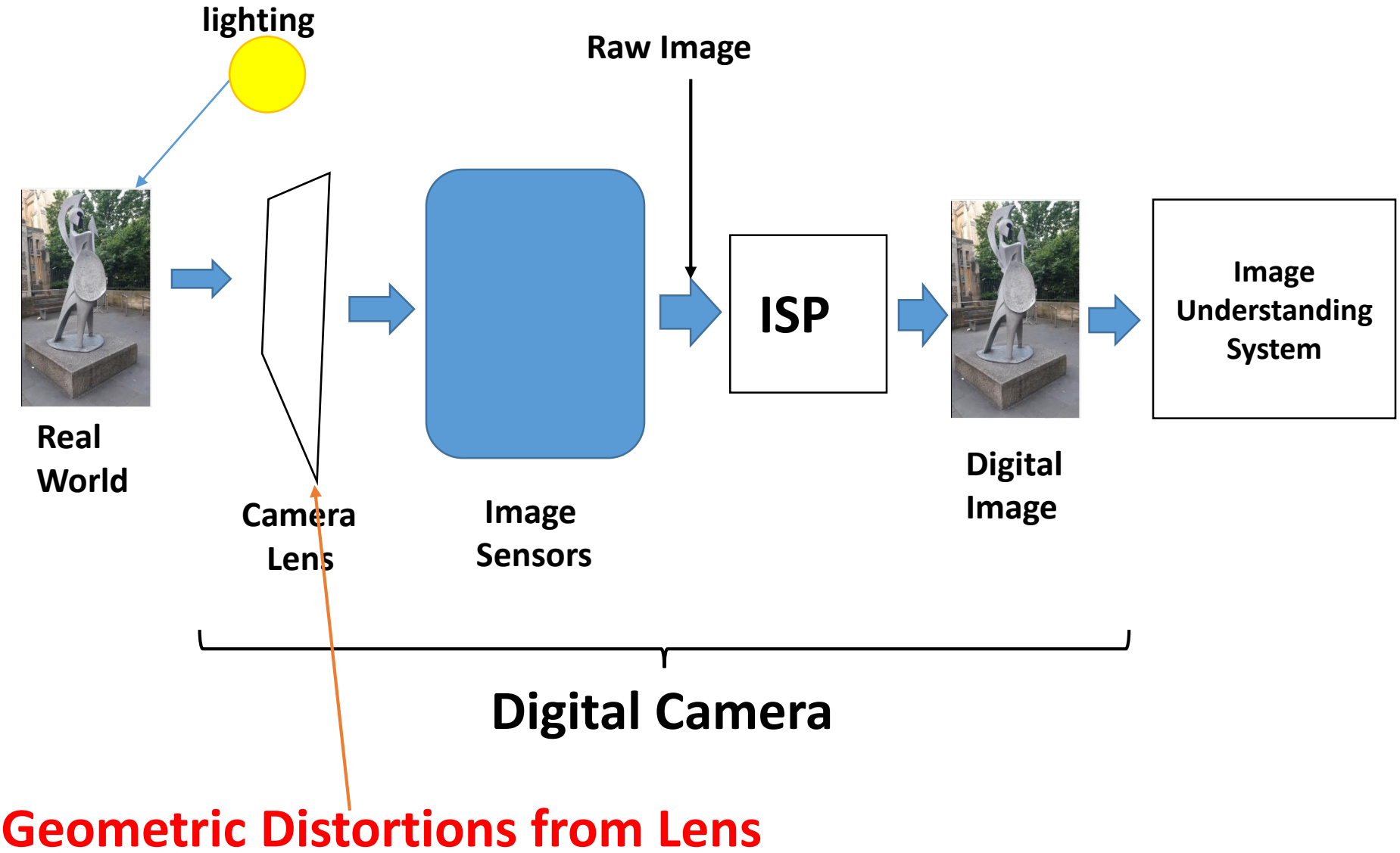
ISP Pipeline Step by Step

- **Each step has its purposes**.
- Instead of describing the techniques for each process, we like to focus on their general approaches and their meanings on the image pipeline.
- We will introduce the general why and how.

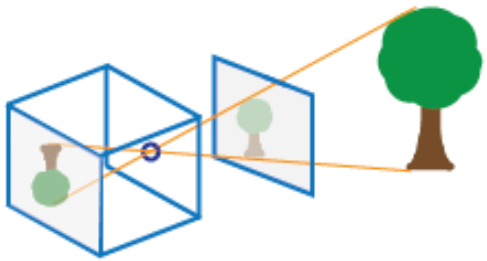


www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

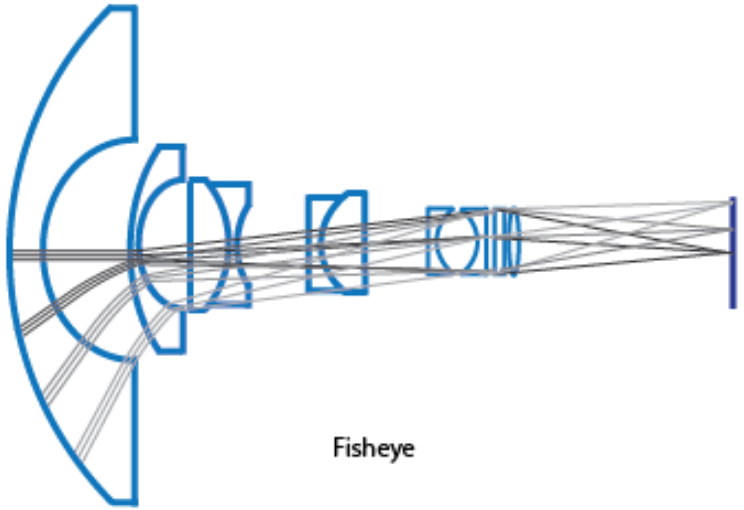
Pipeline for Computer Vision Systems



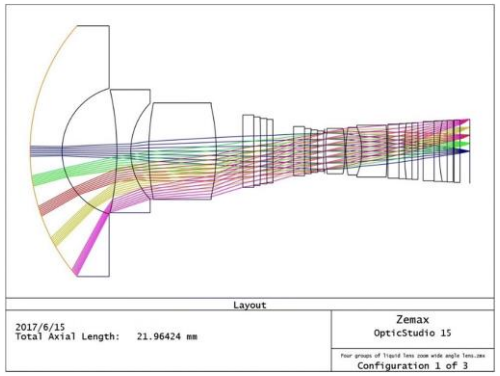
Geometric Distortions from Lens

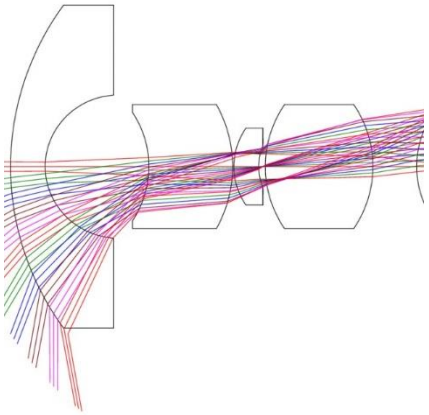


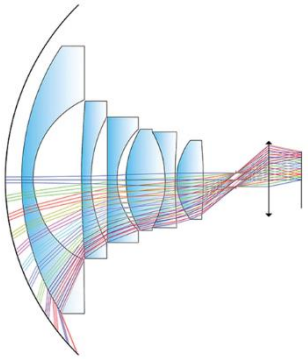
Pinhole



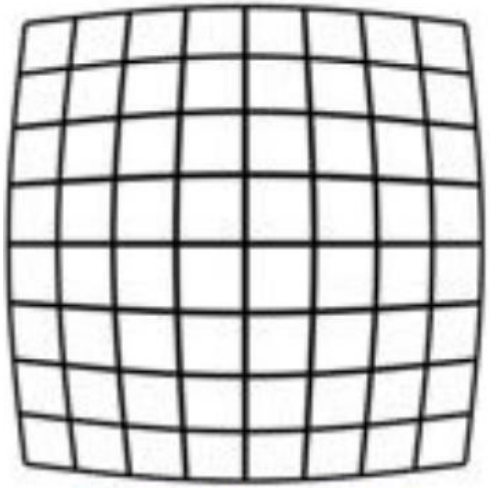
Fisheye



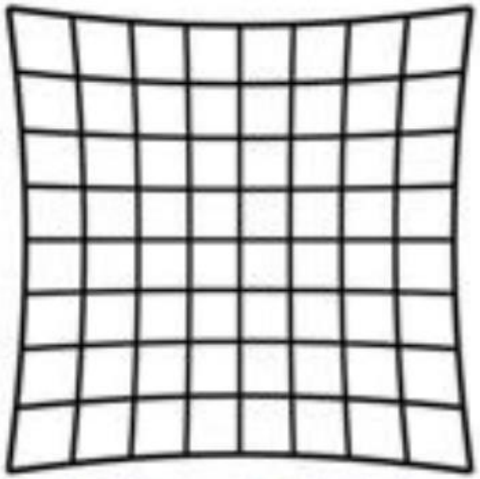




Geometric Distortions from Lens



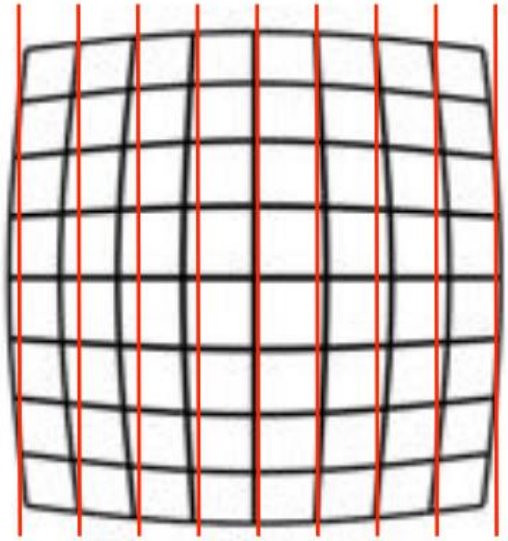
Barrel Distortion



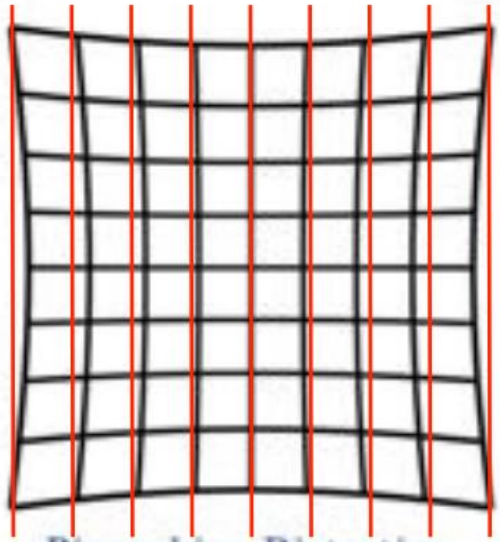
Pincushion Distortion

$$x_{\mathrm{u}} = x_{\mathrm{c}} + \frac{x_{\mathrm{d}} - x_{\mathrm{c}}}{1 + K_1 r^2 + K_2 r^4 + \cdots}$$

$$y_{\mathrm{u}} = y_{\mathrm{c}} + \frac{y_{\mathrm{d}} - y_{\mathrm{c}}}{1 + K_1 r^2 + K_2 r^4 + \cdots},$$



Barrel Distortion



Pincushion Distortion

Stanford, ColorImageProcessing_Joyce.pdf

Software Correction of Lens Distortions

$$x_u = x_d + (x_d - x_c)(K_1 r^2 + K_2 r^4 + \cdots) + (P_1(r^2 + 2(x_d - x_c)^2) + 2P_2(x_d - x_c)(y_d - y_c))(1 + P_3 r^2 + P_4 r^4 \cdots)$$

$$y_u = y_d + (y_d - y_c)(K_1 r^2 + K_2 r^4 + \cdots) + (2P_1(x_d - x_c)(y_d - y_c) + P_2(r^2 + 2(y_d - y_c)^2))(1 + P_3 r^2 + P_4 r^4 \cdots),$$

where:

$(x_d,\ y_d)$ = distorted image point as projected on image plane using specified lens,

$(x_u,\ y_u)$ = undistorted image point as projected by an ideal pinhole camera,

$(x_c,\ y_c)$ = distortion center,

K_n = n^{th} radial distortion coefficient,

P_n = n^{th} tangential distortion coefficient,

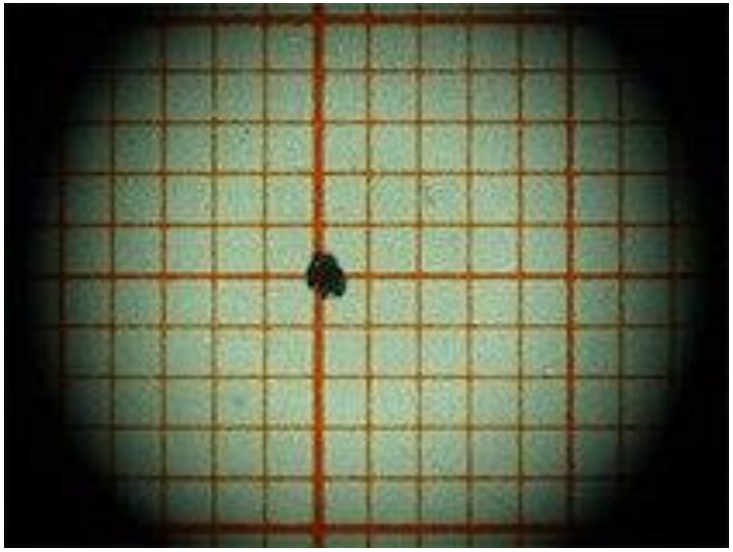
r = $\sqrt{(x_d - x_c)^2 + (y_d - y_c)^2}$, and

$\ldots$ = an infinite series.

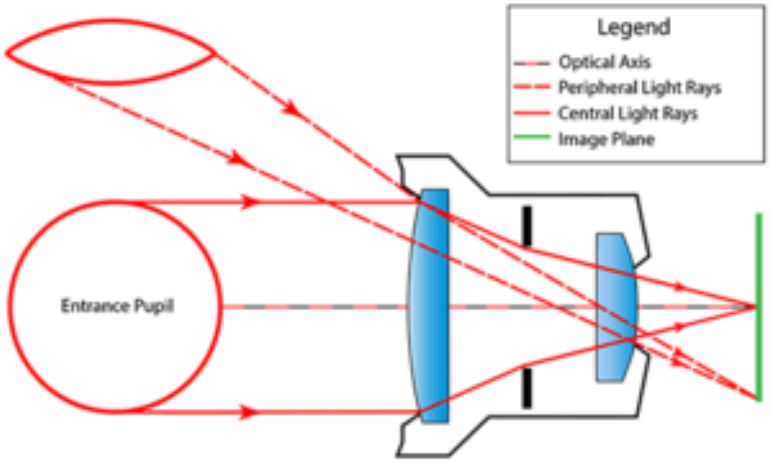
$$x_u = x_c + \frac{x_d - x_c}{1 + K_1 r^2 + K_2 r^4 + \cdots}$$

$$y_u = y_c + \frac{y_d - y_c}{1 + K_1 r^2 + K_2 r^4 + \cdots},$$

Lens Vignetting

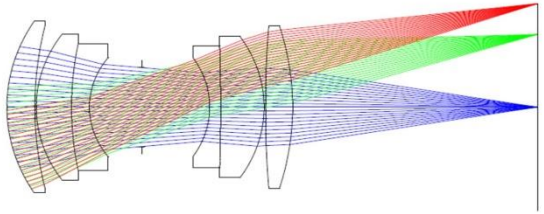


Optical Vignetting



Types of vignetting

Optical vignetting: less light reaches edges of sensor due to physical obstruction in lens



Pixel vignetting: light reaching pixel at an oblique angle is less likely to hit photosensitive region than light incident from straight above (e.g., obscured by electronics)

- **Microlens reduces pixel vignetting**

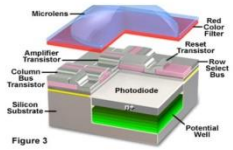


Image credit: Mark Butterworth


```
output_pixel = gain + offset * input_pixel
```

Lens shading compensation

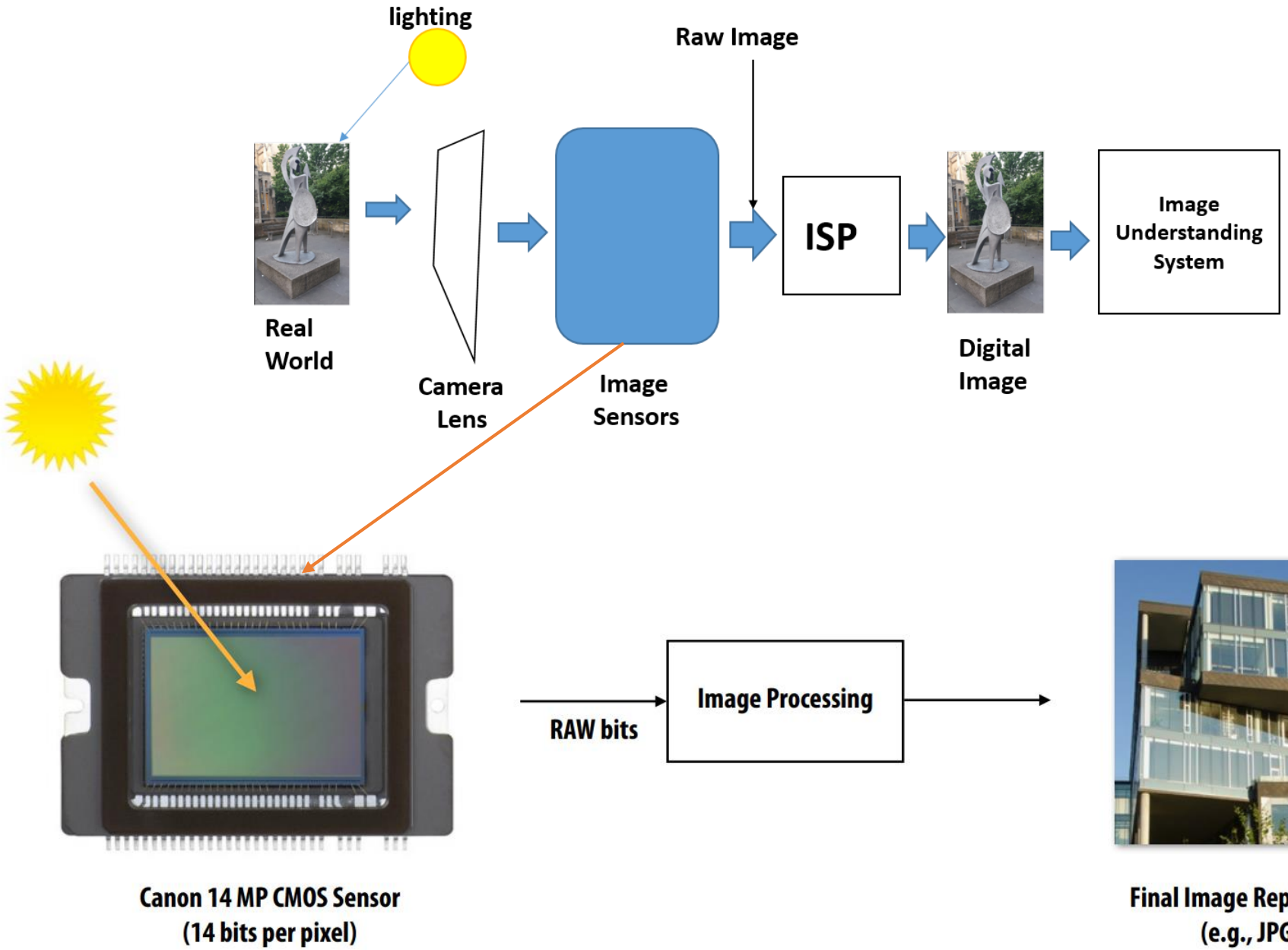
- Correct for vignetting
- Use 2D buffer stored in memory
  - Lower res buffer, upsampled on-the-fly

```
offset = upsample_compensation_offset_buffer(current_pixel_xy);
gain = upsample_compensation_gain_buffer(current_pixel_xy);

output_pixel = gain + offset * input_pixel;
```

Optional dark frame subtract

- Similar computation to lens shading compensation

```
output_pixel = input_pixel - dark_frame[current_pixel_xy];
```



Exposure for Photography

Concept of Exposure Value

1. Different combinations of aperture and integration times produce the same exposure value

2. Exposure value accounts for both F-number and integration time

3. Algorithms are based on a definition of Exposure Value (EV) and estimates of the current image brightness (B).

$$EV = \log_2\left(\frac{F^2}{T}\right) = 2\log_2(F) - \log(T)$$

F : f-number

T : exposure time

Stanford, ColorImageProcessing_Joyce.pdf

Concept of Exposure Value

$$EV = \log_2(\frac{F^2}{T}) = 2\log_2(F) - \log(T)$$

F : f-number

T : exposure time

1. Smaller F-number = more light
   - The bigger the aperture, the smaller the F-number
   - F=focal length/aperture diameter
2. Larger T (exposure time) = more light
3. Hence, smaller exposure value is better

Stanford, ColorImageProcessing_Joyce.pdf

Exposure Control

Increase or decrease the amount of light by varying

1. Aperture size
   - When focal length is fixed, the f-number is determined by the diameter of the aperture

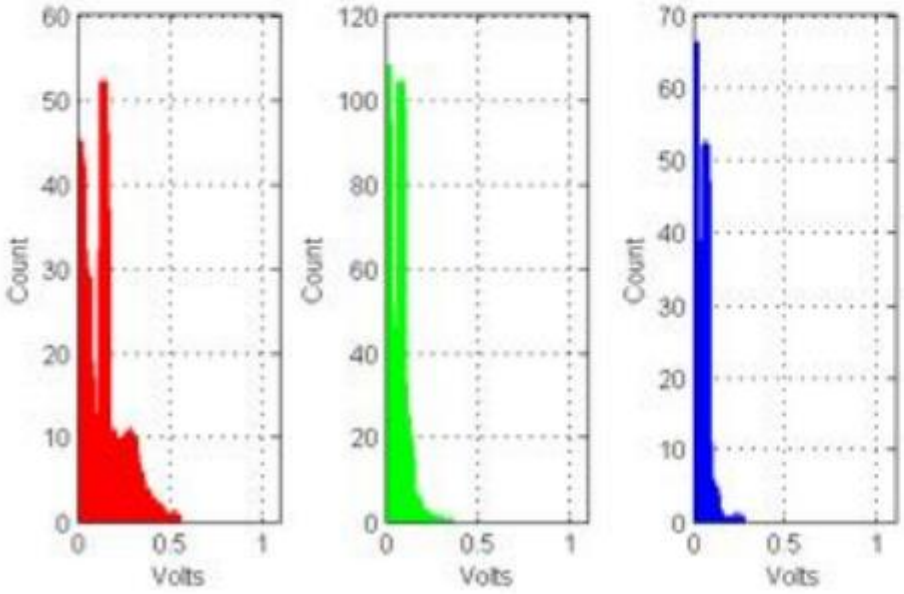
     F = focal length/aperture diameter

2. Exposure duration

Set the f-number (F) and exposure time (T) so that the image pixels accumulate a charge that reaches, but does not exceed, the well-capacity of the pixels (sensor range).

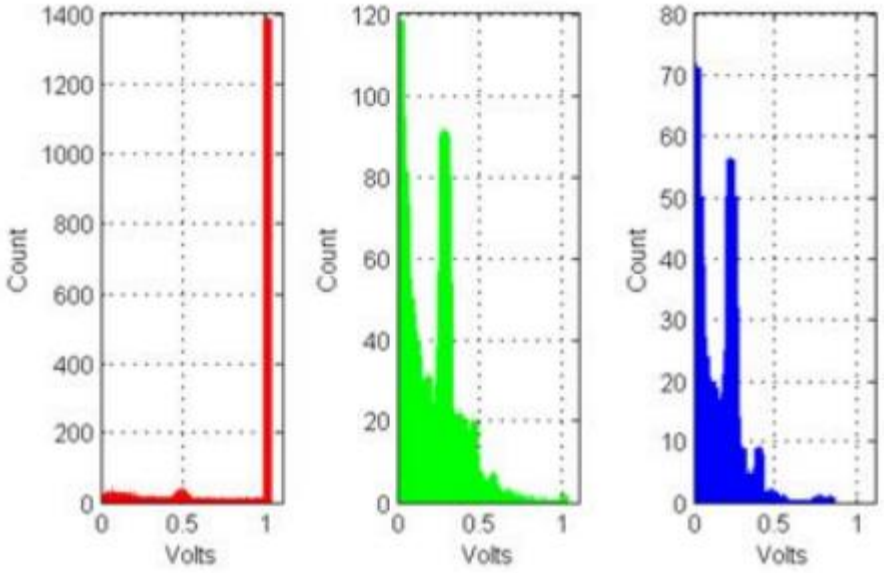
Stanford, ColorImageProcessing_Joyce.pdf

Exposure Control: Underexposed



Stanford, ColorImageProcessing_Joyce.pdf

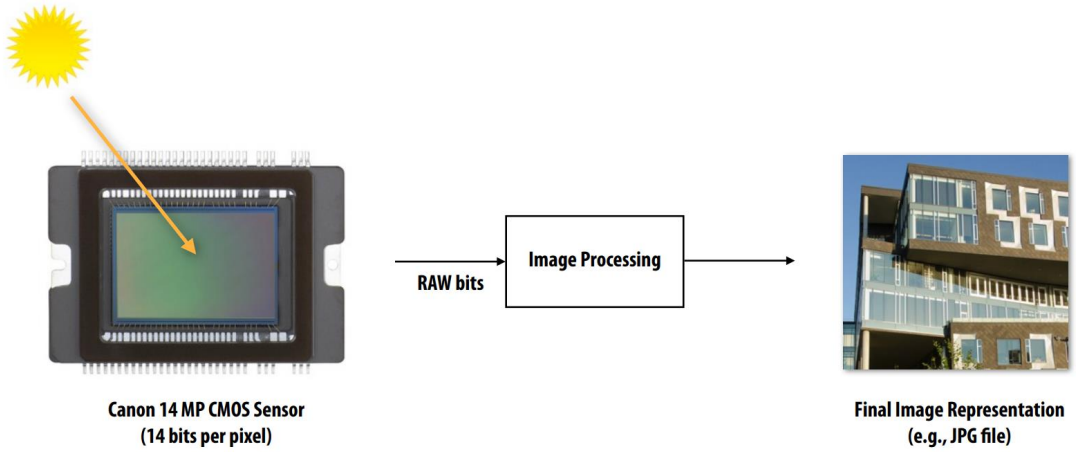
Exposure Control: Overexposed



Stanford, ColorImageProcessing_Joyce.pdf

Image Sensor Output

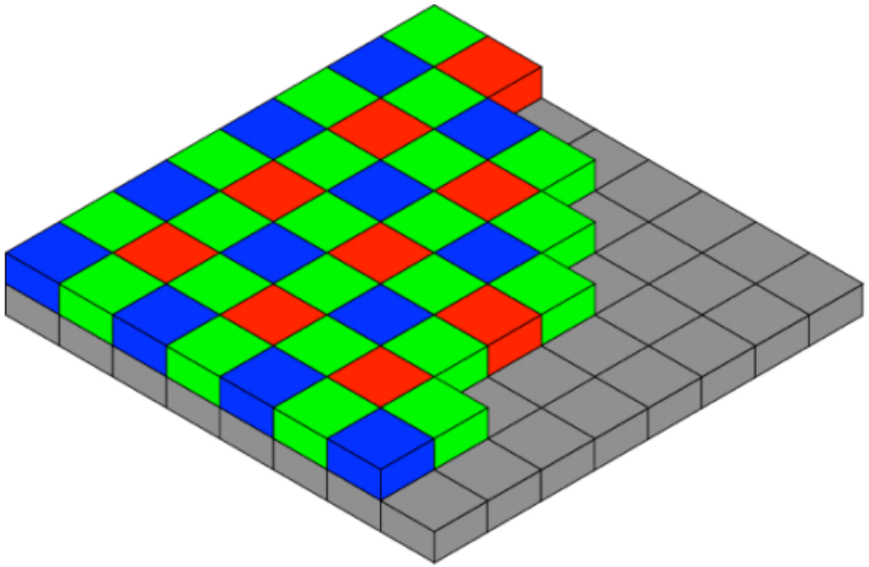
- CCD (Charge-Coupled Device)
  - Monitor mode (preview): full width * 2XX sub-sampled line
  - Capture mode (still): full frame output
  - AF mode: faster frame rate with less lines output
- CMOS (Complementary Metal-Oxide-Semiconductor)
  - Monitor mode: **Down sampling output by two** or ...
  - Capture mode: full frame output
  - Special mode: window output with panning feature



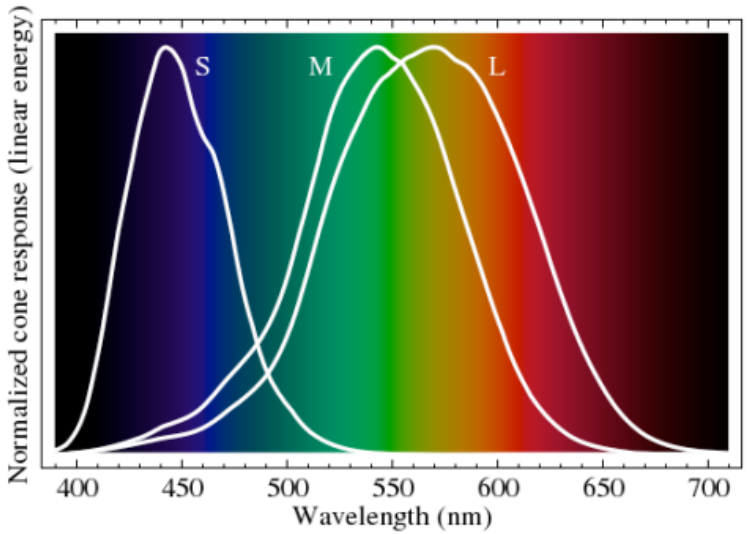
www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Bayer filter mosaic

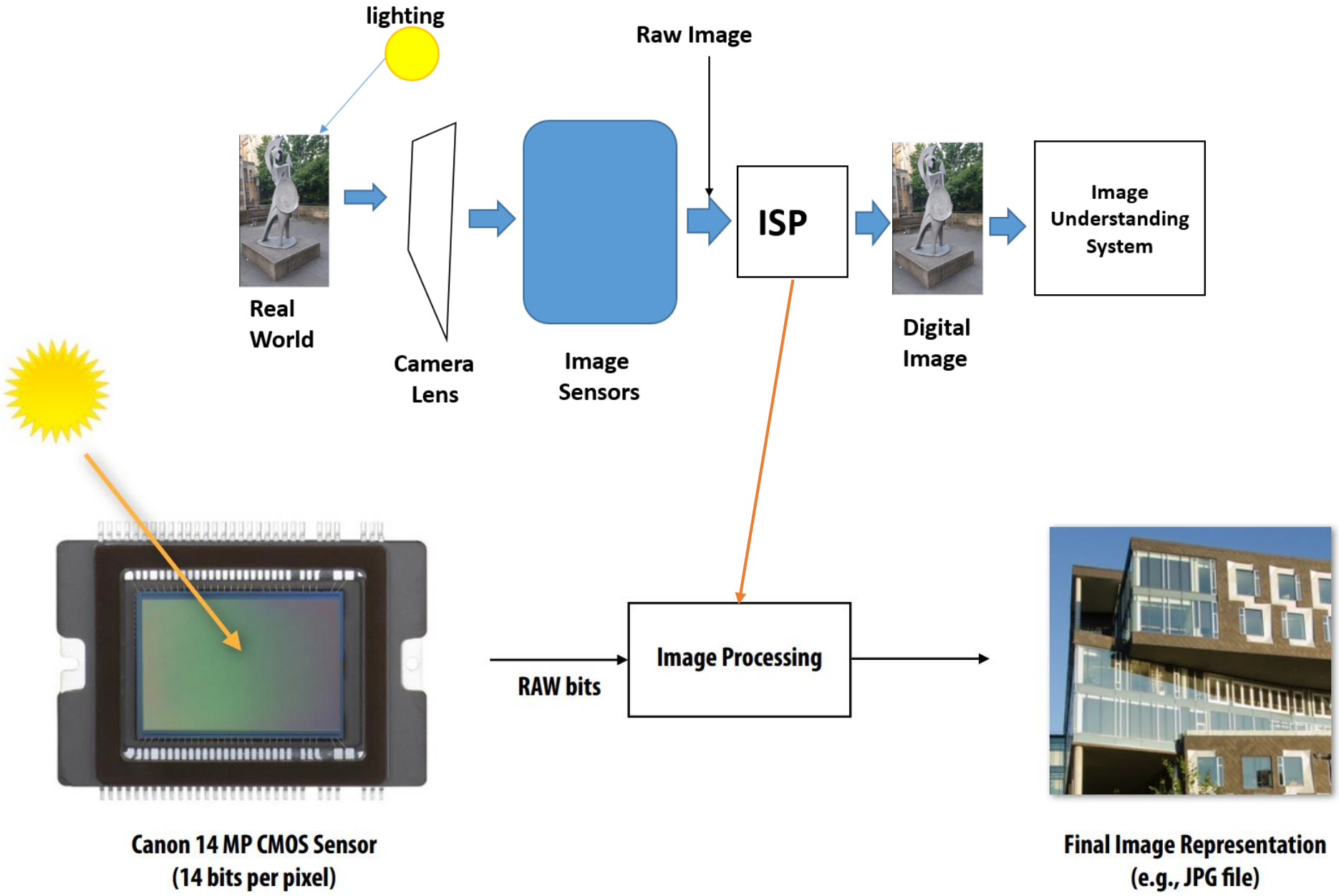
- Color filter array placed over sensor
- Result: each pixel measures incident red, green, or blue light
- 50% of pixels are green pixels
  - Human visual perception most sensitive to green light (in normal light levels)



Traditional Bayer mosaic
(other filter patterns exist: e.g., Sony's RGBE)

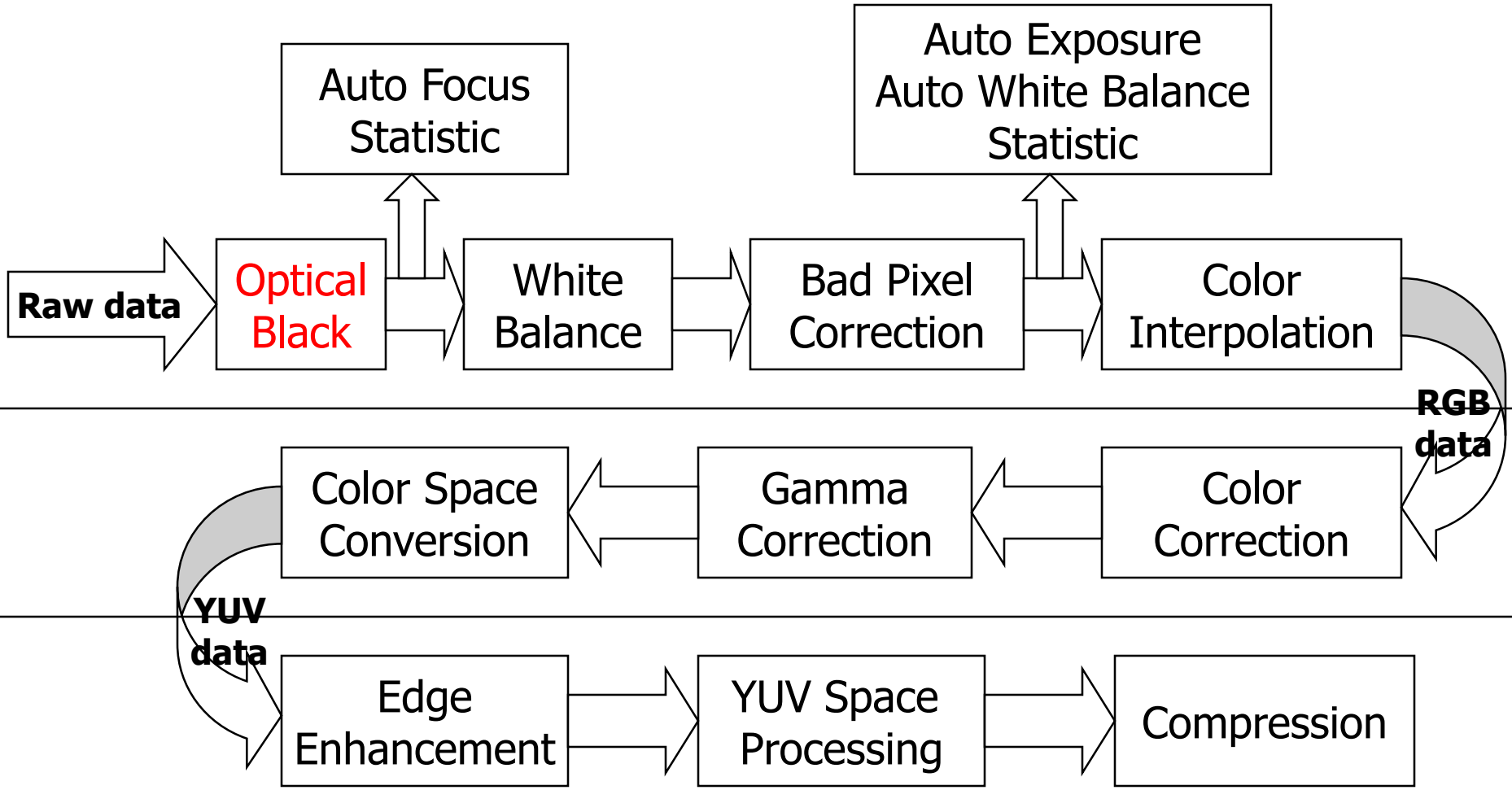


Human eye: cone spectral response



Kayvon Fatahalian, Graphics and Imaging Architectures (CMU 15-869, Fall 2011)

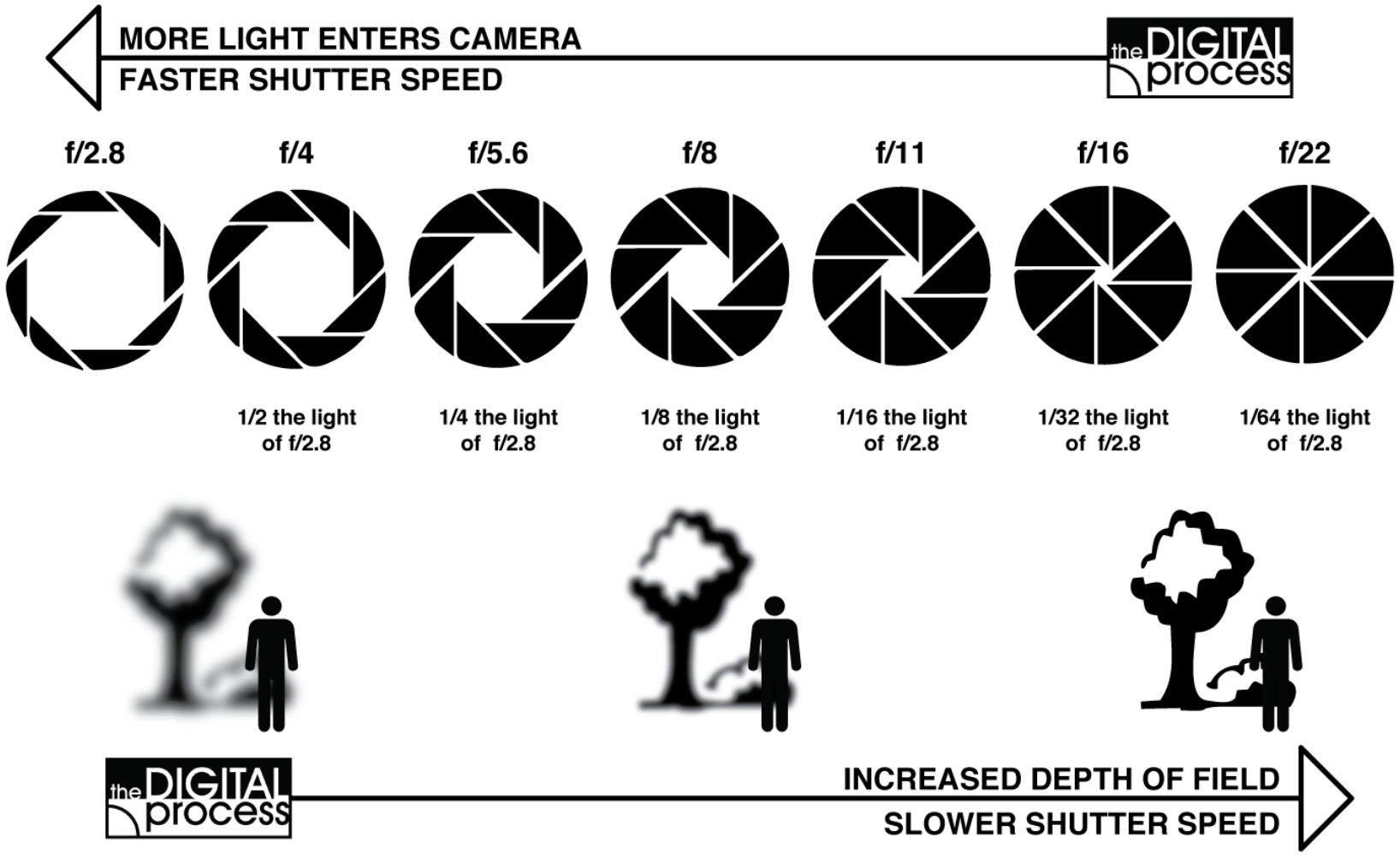
A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

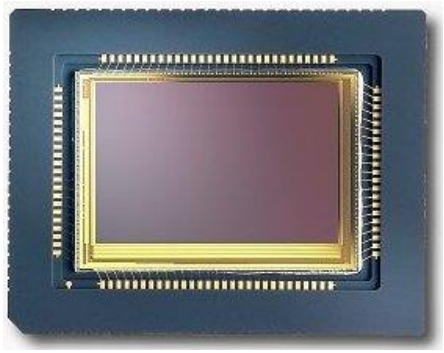
www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

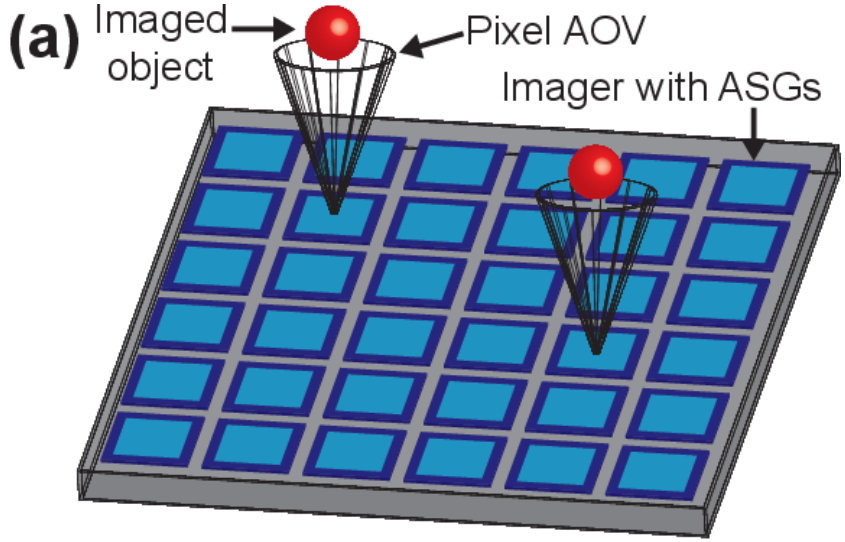
Optical Black Clamping

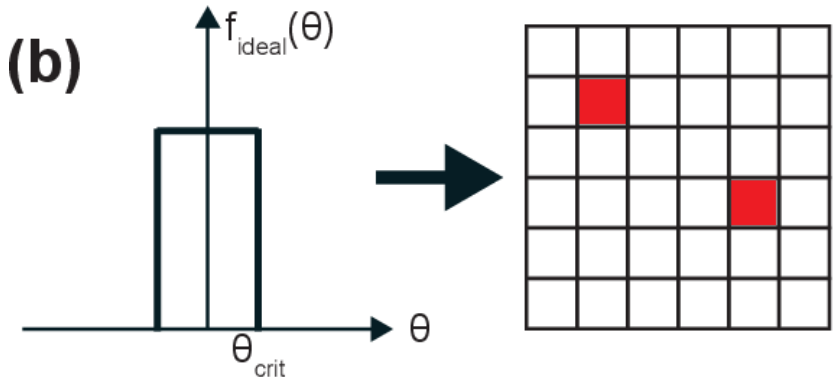


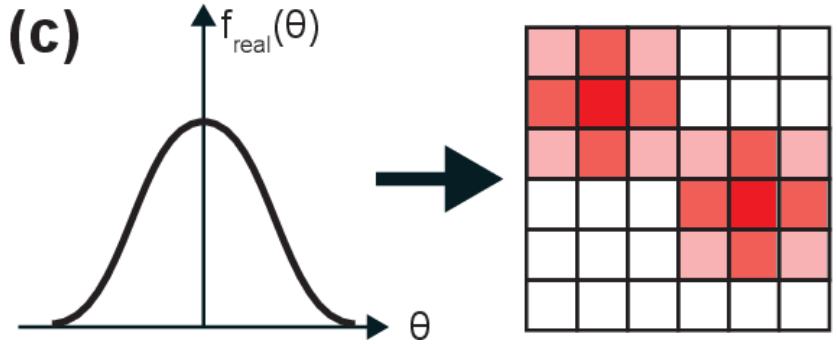
Optical Black Clamping

Optical Black Clamping

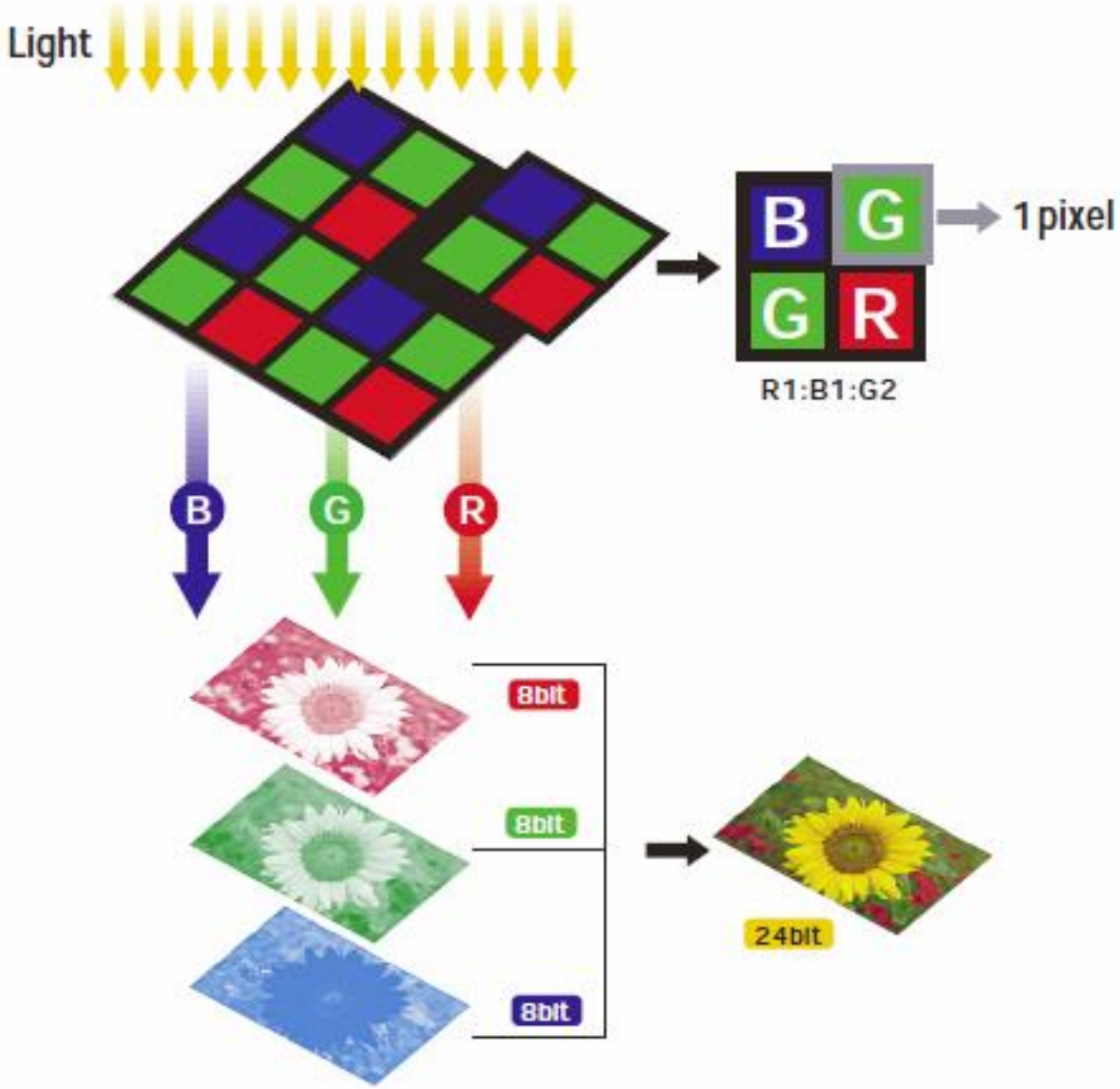






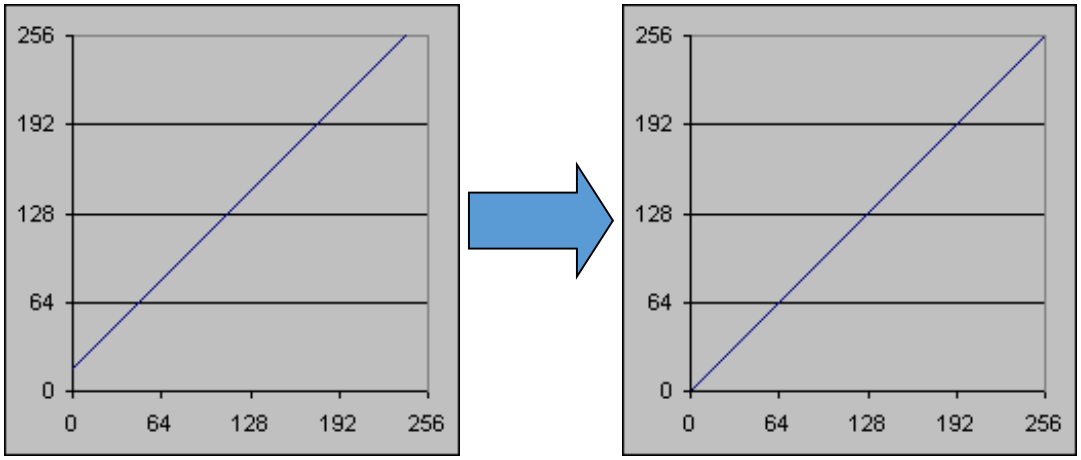


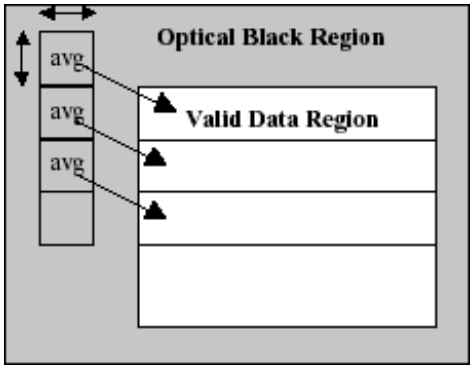
Optical Black Clamping



Optical Black Clamping

- Compensate image sensors' **dark signal**
- Subtract OB (Optical Black) from pixel signal
- OB value
  - Computed by DSP from image sensor's OB area
  - Manually set by firmware

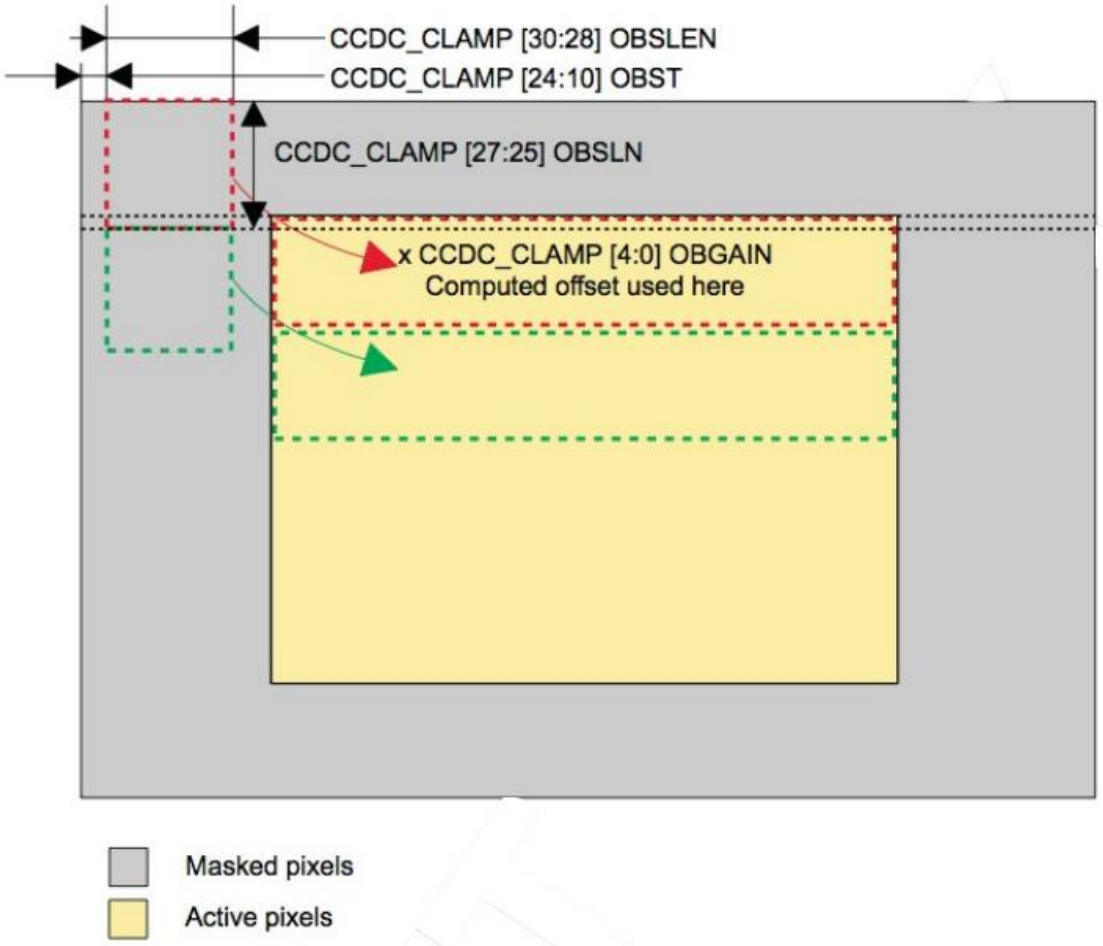





```
output_pixel = input_pixel - [average of pixels from optically black region]
```

Optical clamp: remove sensor offset bias

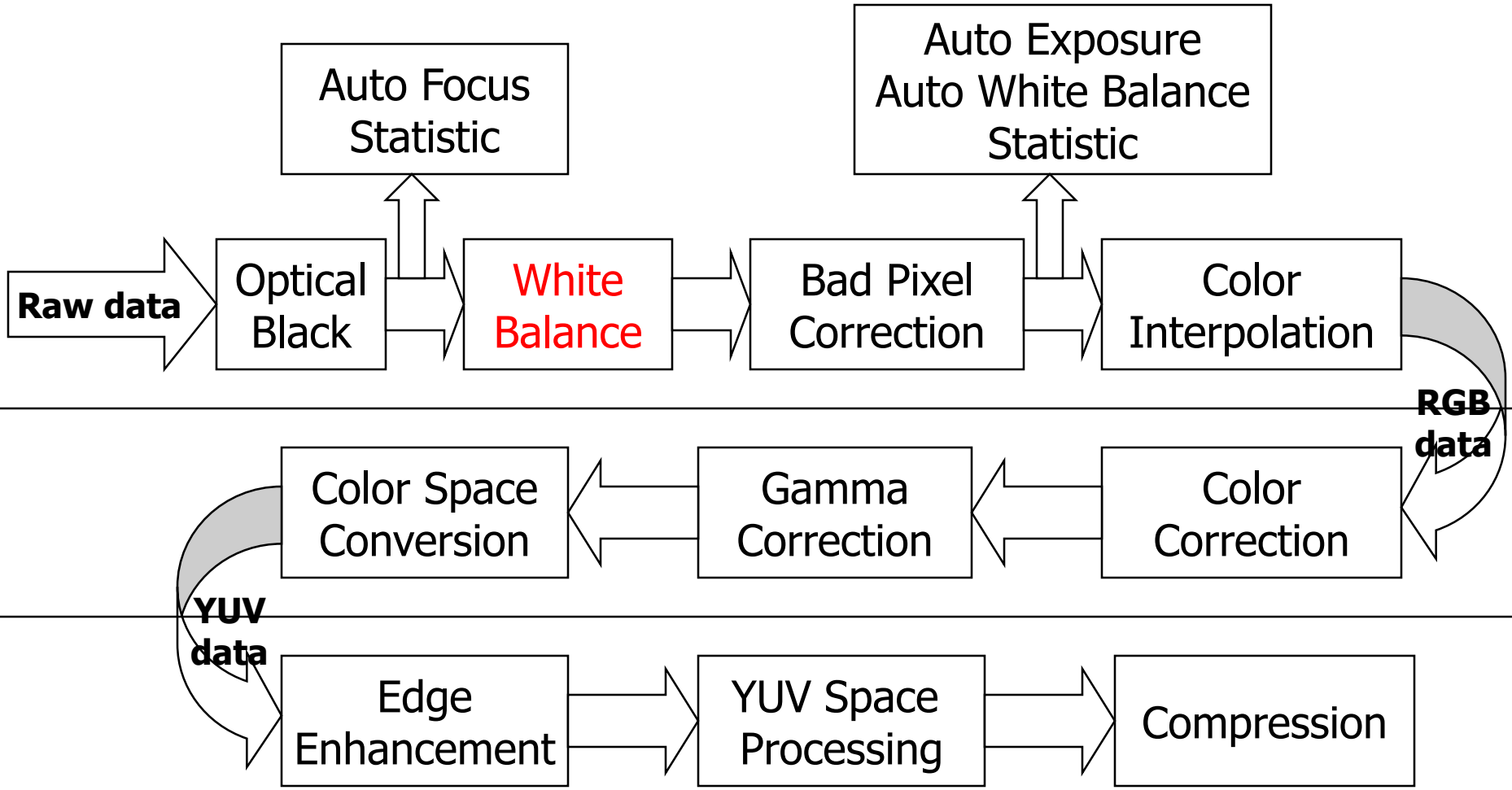
```
output_pixel = input_pixel - [average of pixels from optically black region]
```



Remove bias due to sensor black level

(from nearby sensor pixels at time of shot)

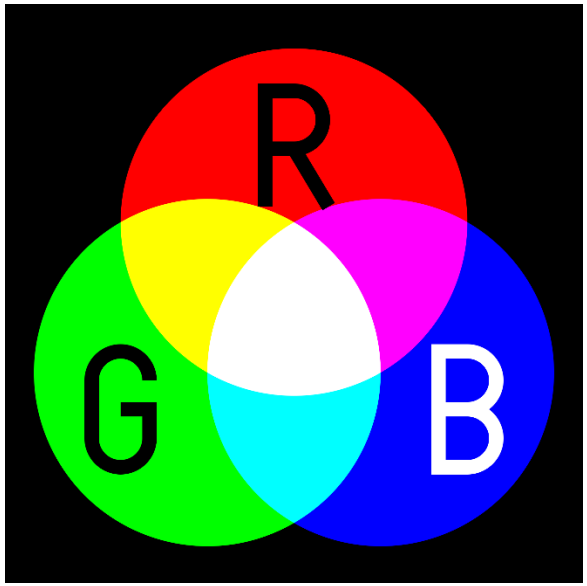
A Typical Image Pipeline for Digital Camera

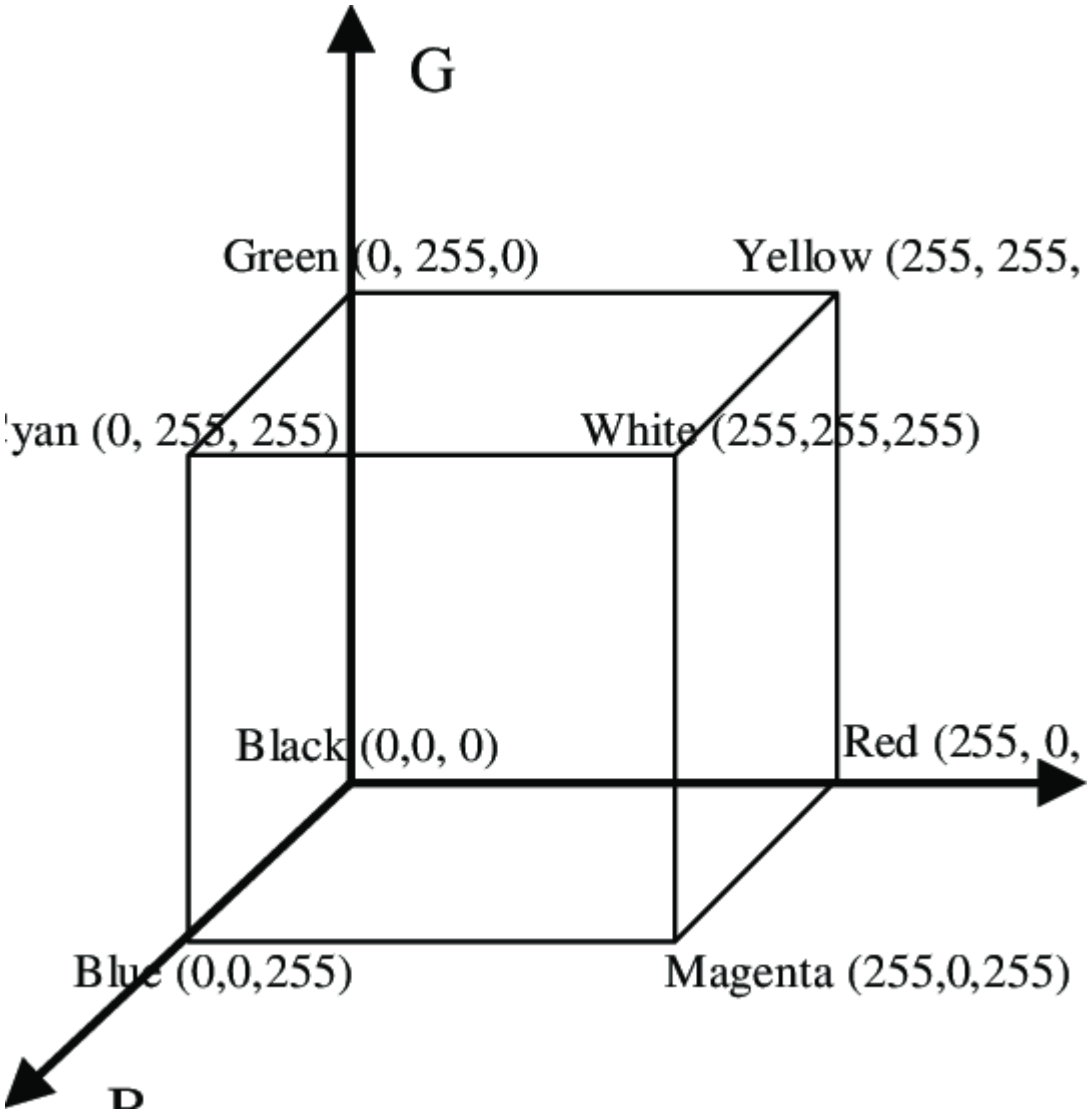


ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

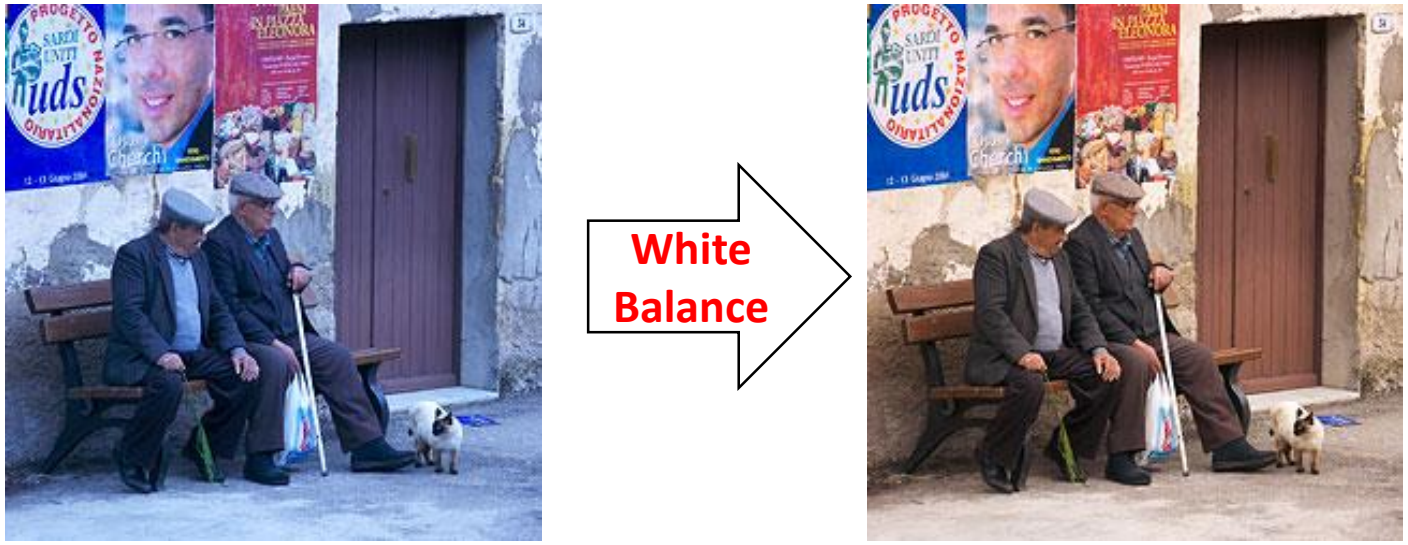
Color Balance





White Balance: Matching Human Perception

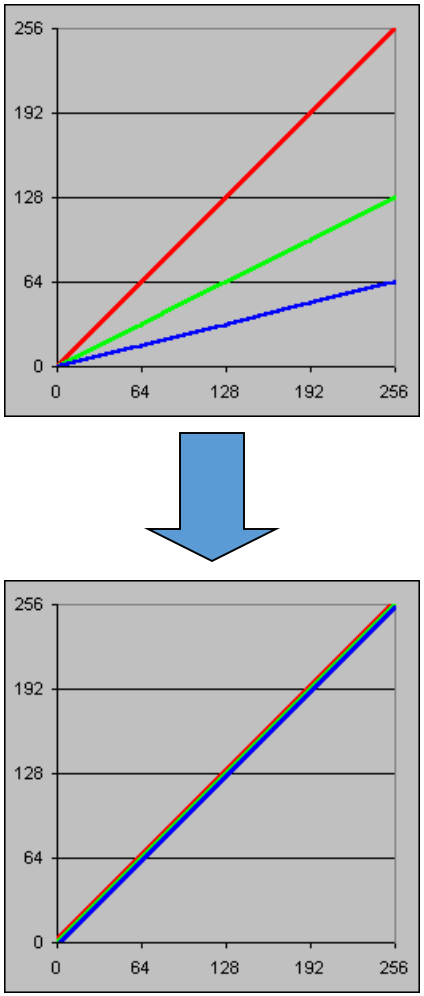
White balance (WB) is the process of **removing unrealistic color casts**, so that objects which appear white in person are rendered white in your photo. Proper camera white balance has to take into account the "color temperature" of a **light source**, which refers to the relative warmth or coolness of **white light**. Our eyes are very good at judging **what is white under different light sources**, but digital cameras often have great difficulty with **auto white balance** (AWB) — and can create unsightly blue, orange, or even green color casts. Understanding digital white balance can help you avoid these color casts, thereby improving your photos under a wider range of lighting conditions.

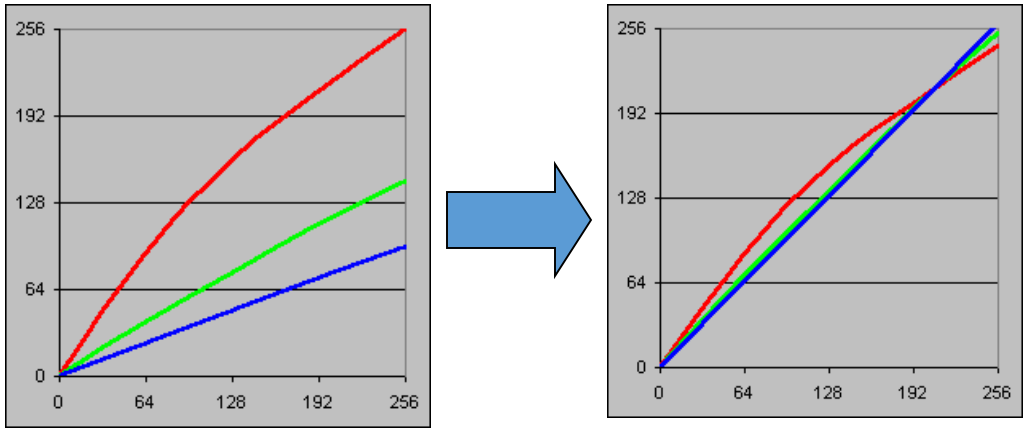



```
output_pixel = white_balance_coeff * input_pixel
```

White Balance: Matching Human Perception

- **To simulate human eyes white balance**: adjusting colors so that the image looks more natural
- **Adjustable channel gain** for each color channel
- General approaches
  - Gray world assumption
  - Perfect reflector assumption
  - Calibration based approaches
- What if data are nonlinear?



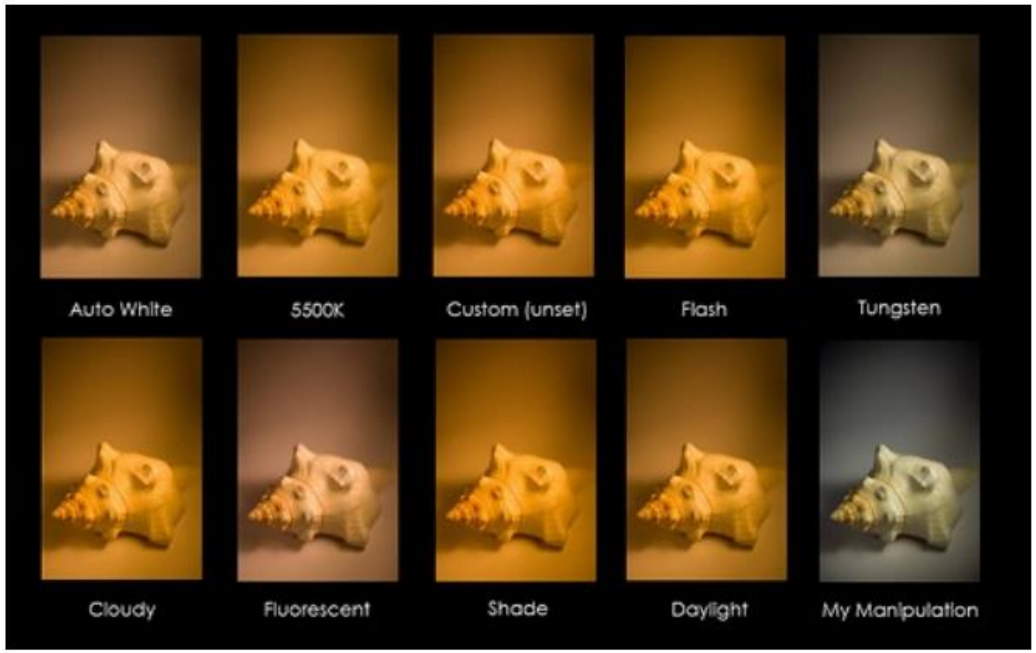


White balance

- **Adjust relative intensity of rgb values (usually so neutral tones appear neutral)**

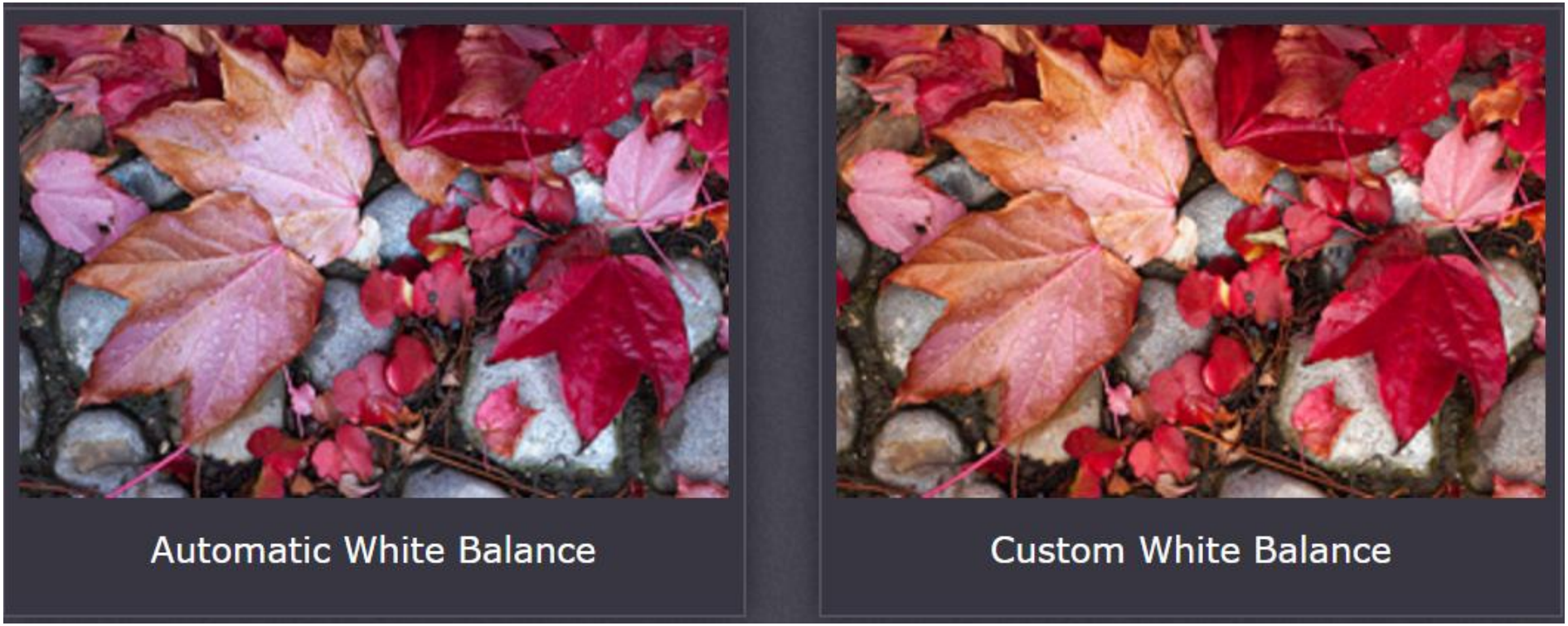
```
output_pixel = white_balance_coeff * input_pixel

note: white_balance_coeff depends on whether pixel is red, green, or blue pixel
```

- **Setting white balance coefficients:**
  - **Example naive auto-white balance algorithms**
    - **Gray world assumption: make average of all pixels gray**
    - **Find brightest region of image, make it white**
- **Modern cameras have sophisticated, heuristic white-balance algorithms (proprietary)**



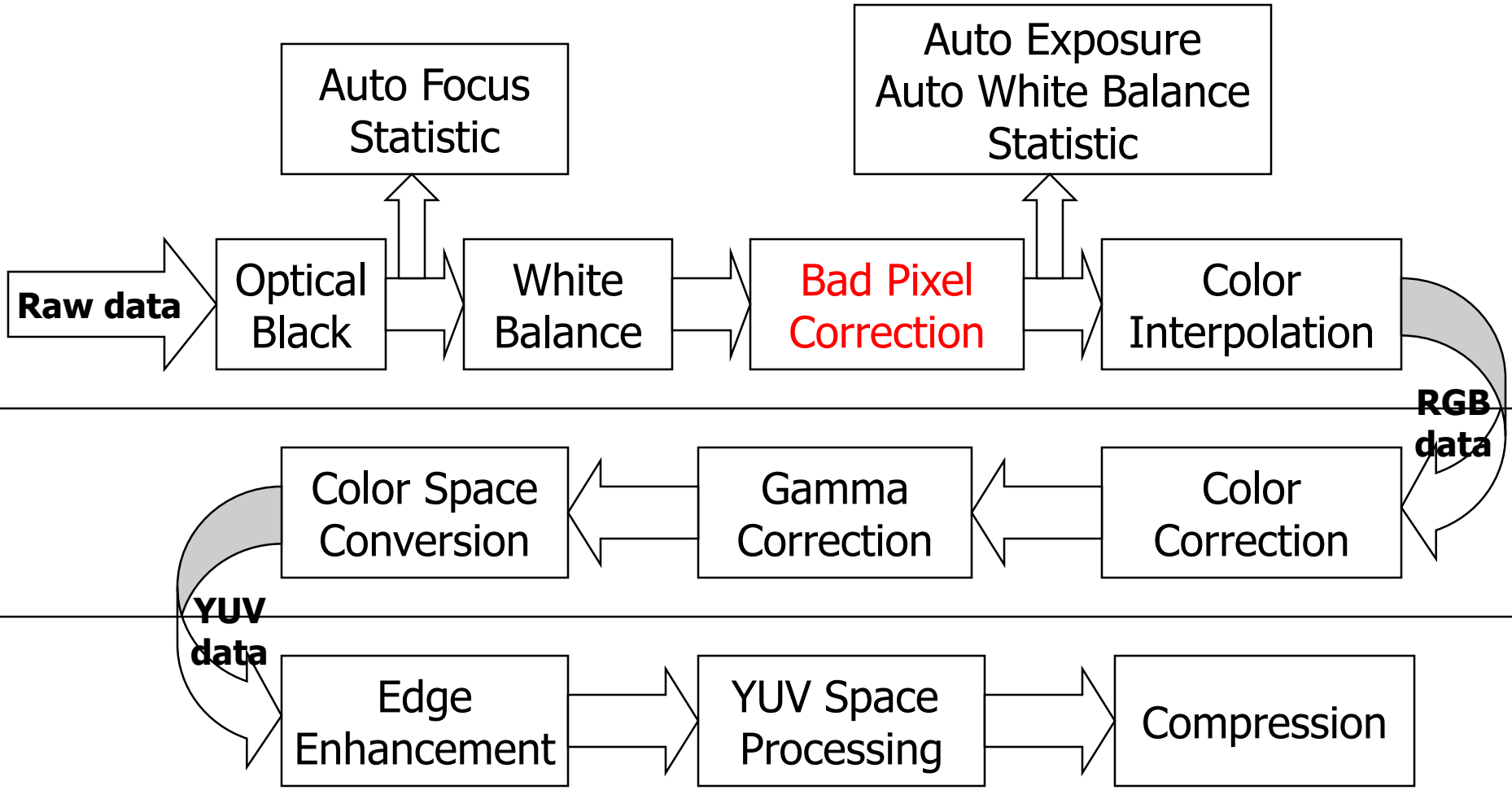
Kayvon Fatahalian, Graphics and Imaging Architectures (CMU 15-869, Fall 2011)

White Balance




```
output_pixel = white_balance_coeff * input_pixel
```

A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

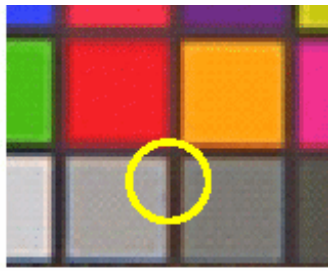
www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Bad Pixel Correction

- **Non-perfect image sensors**
- More than what the spec. claims
  - Judgment standards are different!
- **Must be done in raw data space** to prevent bad pixels from polluting neighborhood
- Considering **edge and gradient** information

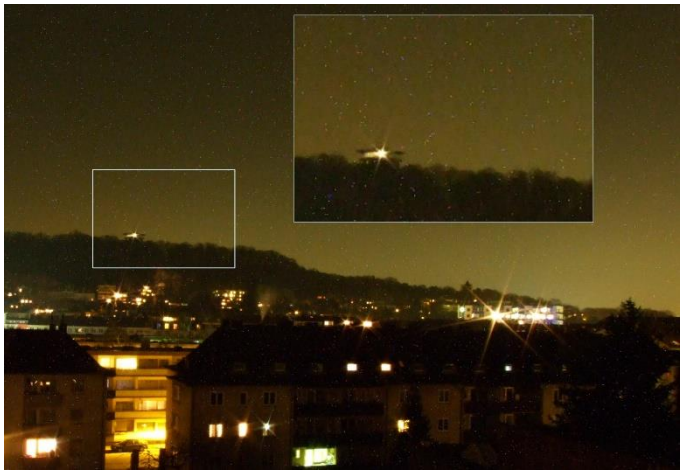
Error pixel
(Uncorrected)

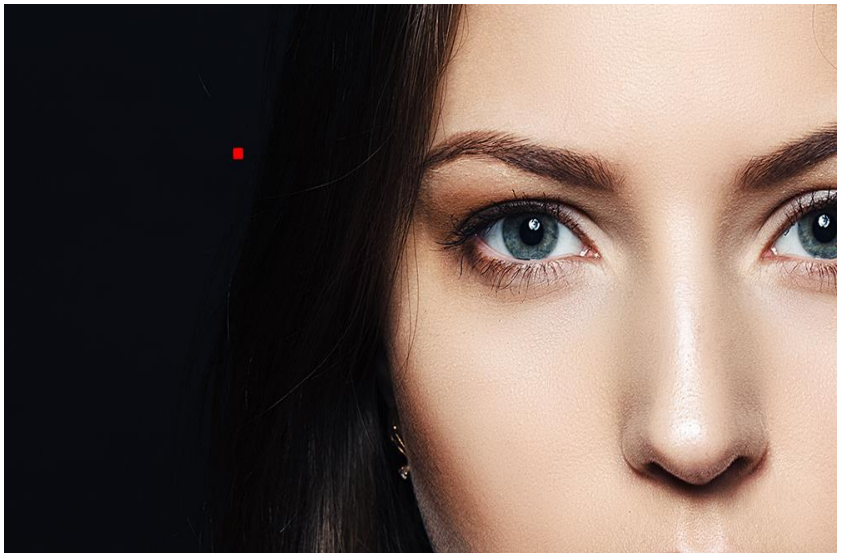
Typical pixel
error correction



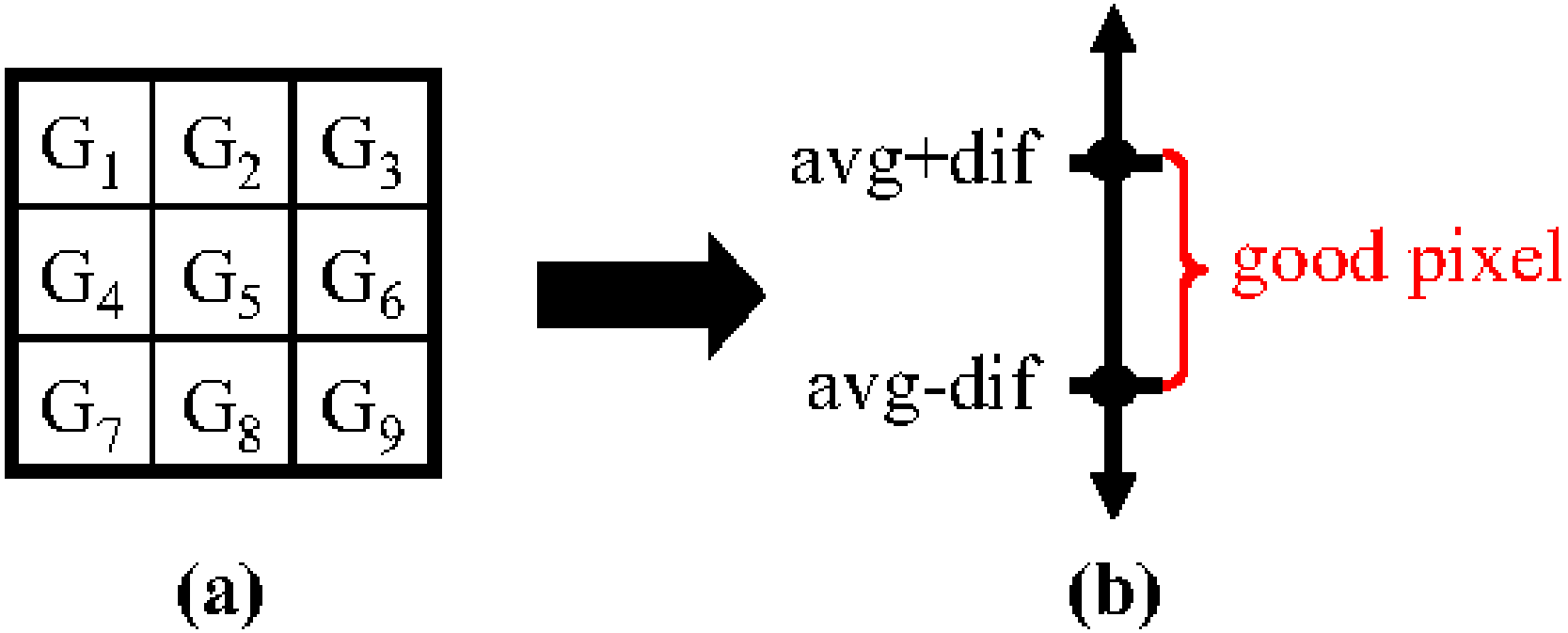
NuCORE pixel
error correction

Bad Pixels





Bad Pixel Correction



Bad Pixel Correction

	136	
147	255	144
	255	

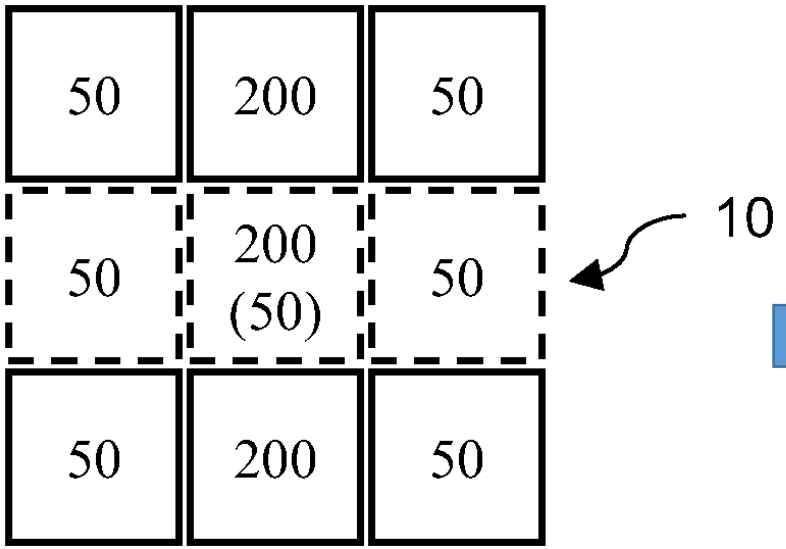
147	145	144

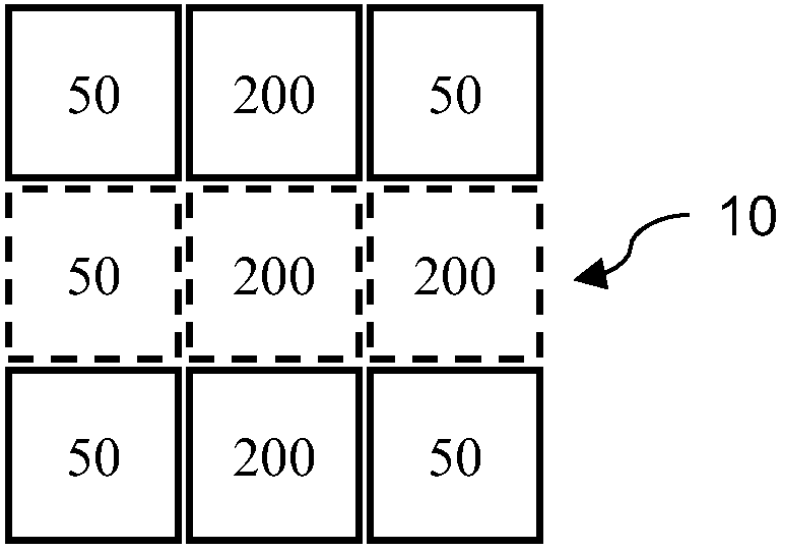
$est = (147 + 144) / 2 = 145$

$dif = |145 - 255| = 110$

$avg = (136 + 147 + 255x4 + 144 + 255) / 8$
$= 213$

Original Output **Estimated P_5**





Bad Pixel Correction

	136	
147	0	144
	0	

Original Output

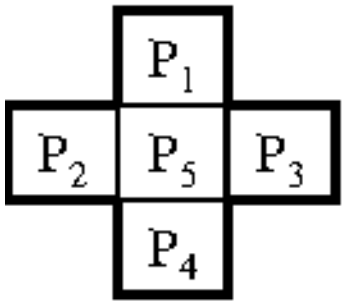
147	145	144

Estimated P_5

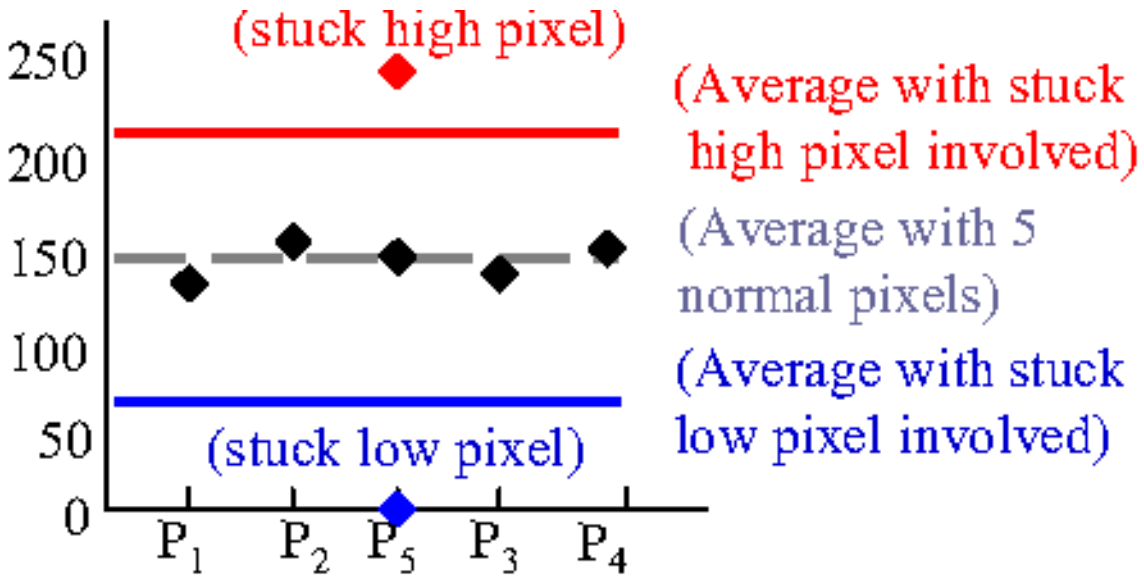
$est = (147 + 144) / 2 = 145$

$dif = |145 - 0| = 145$

$avg = (136 + 147 + 0x4 + 144 + 0) / 8 = 53$

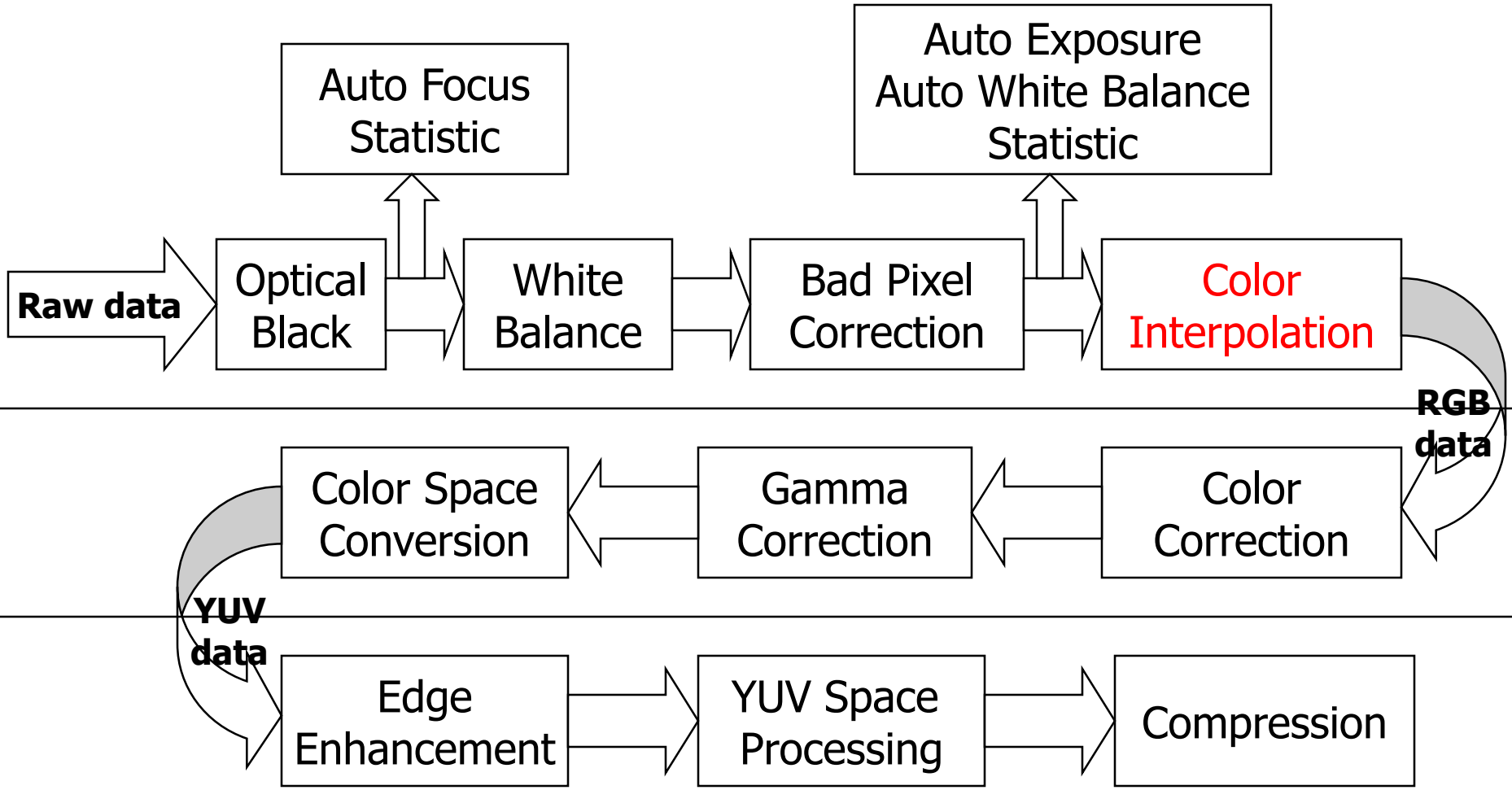


(a)



(b)

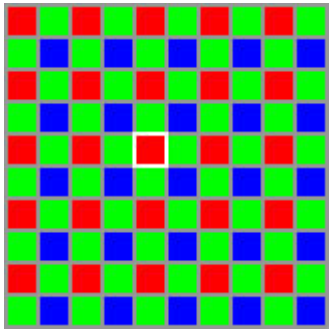
A Typical Image Pipeline for Digital Camera



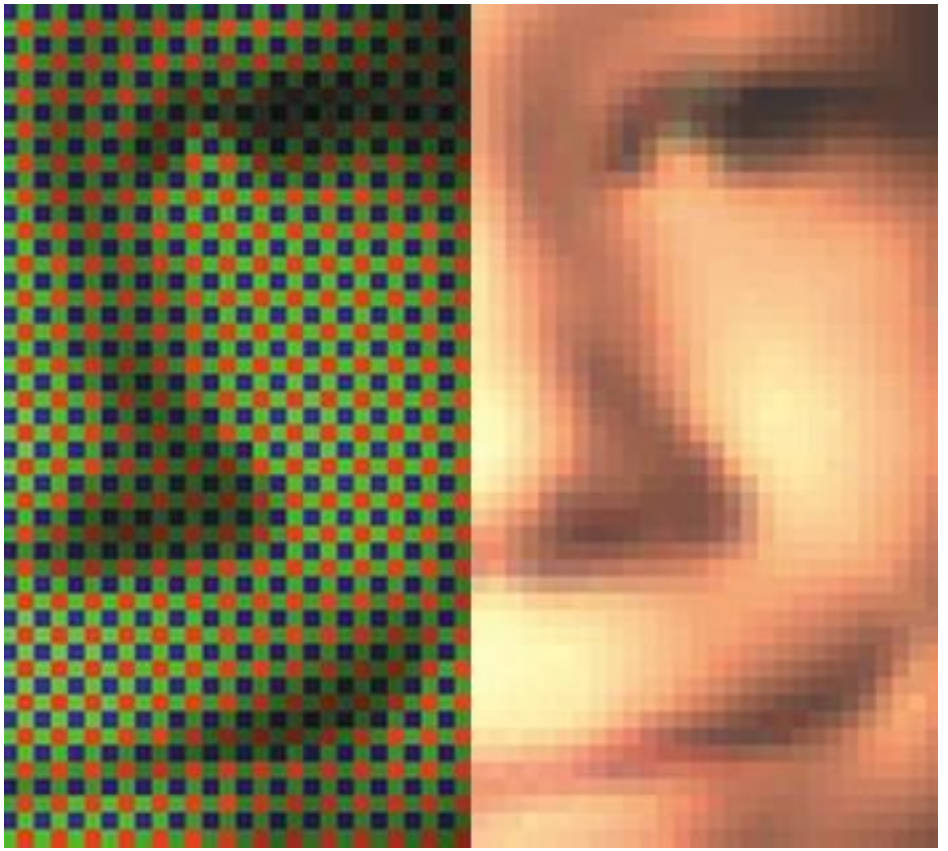
ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Color Interpolation



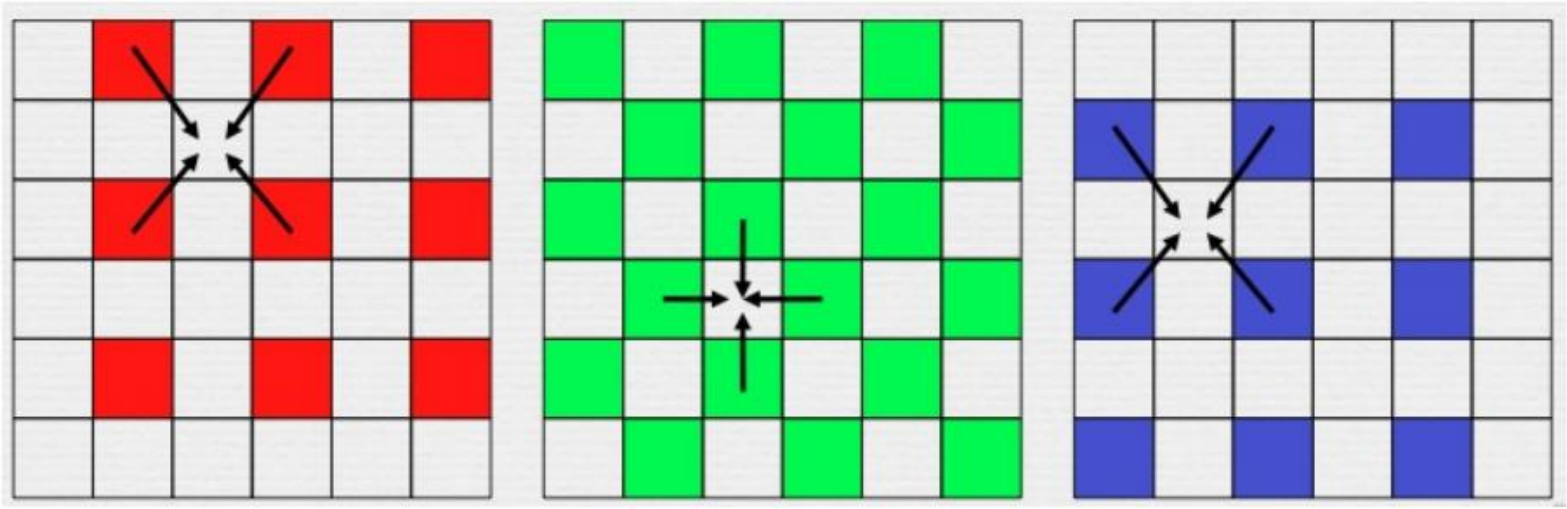
- Also called **de-mosaic / raw2rgb**...
- Guess missing channels for each pixel by the following:
  - Neighbor pixels
  - Edge
  - Gradient
  - …
- Avoid zigzag and false color artifacts



Color Interpolation

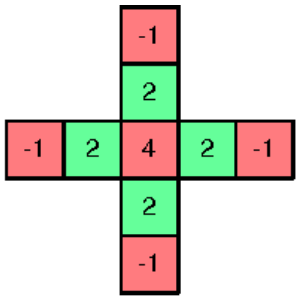
Demosiac

- Produce a RGB image from mosaiced input
- Basic algorithm: linear interpolation of mosaiced values
- More advanced algorithms: attempt to preserve edges

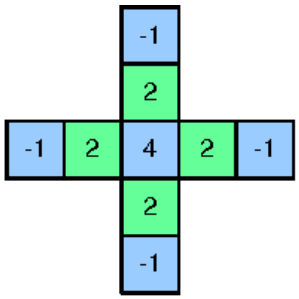


Kayvon Fatahalian, Graphics and Imaging Architectures (CMU 15-869, Fall 2011)

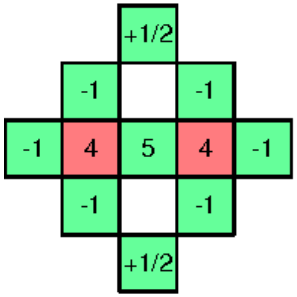
Color Interpolation



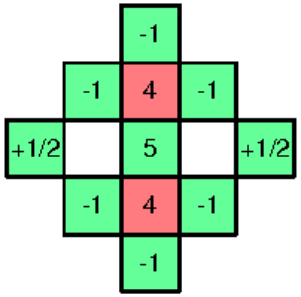
G at R locations



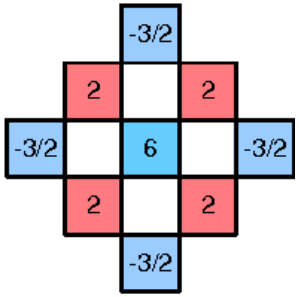
G at B locations



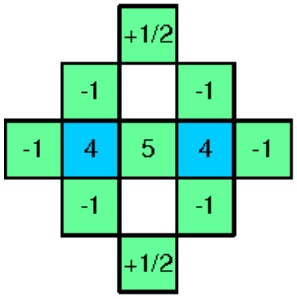
R at green in
R row, B column



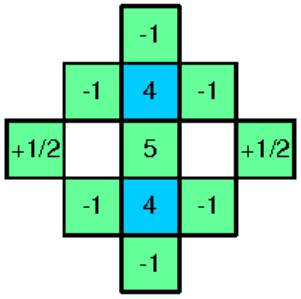
R at green in
B row, R column



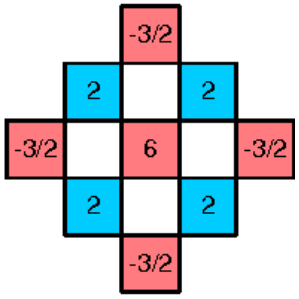
R at blue in
B row, B column



B at green in
B row, R column



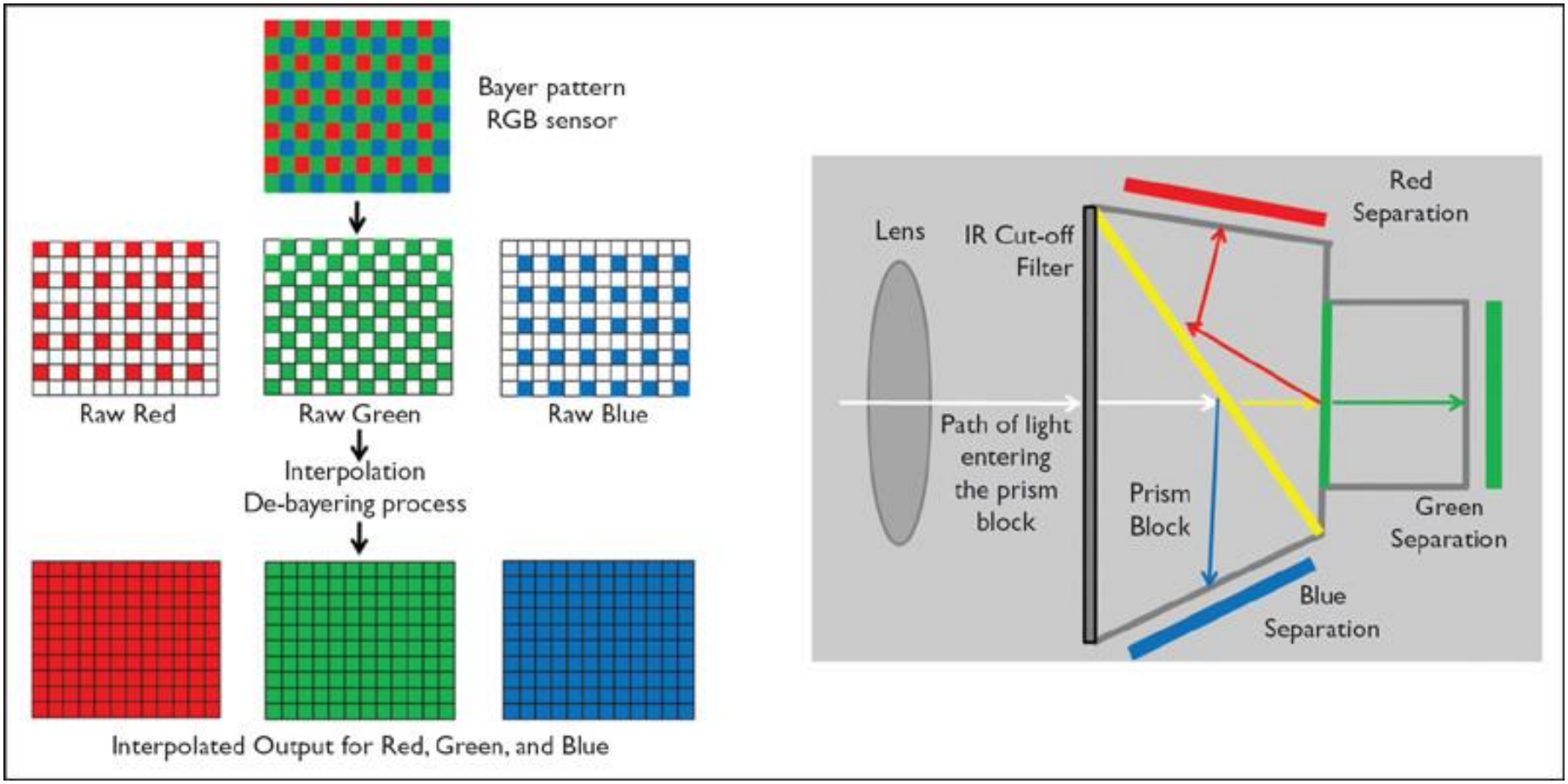
B at green in
R row, B column



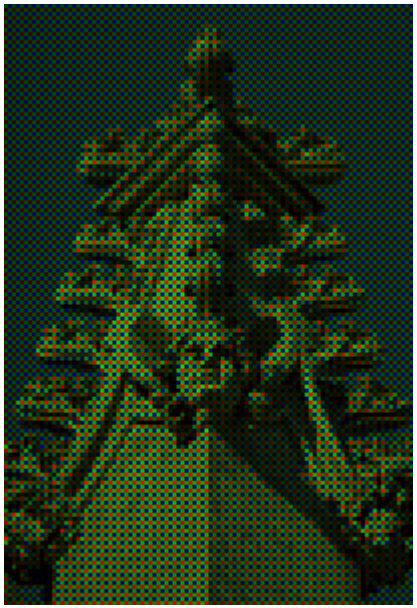
B at red in
R row, R column

BAYER DEMOSAICING

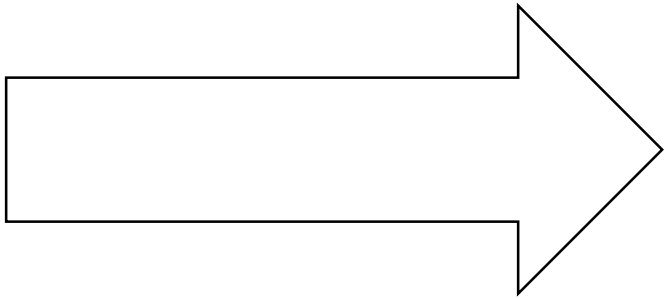
Color Interpolation



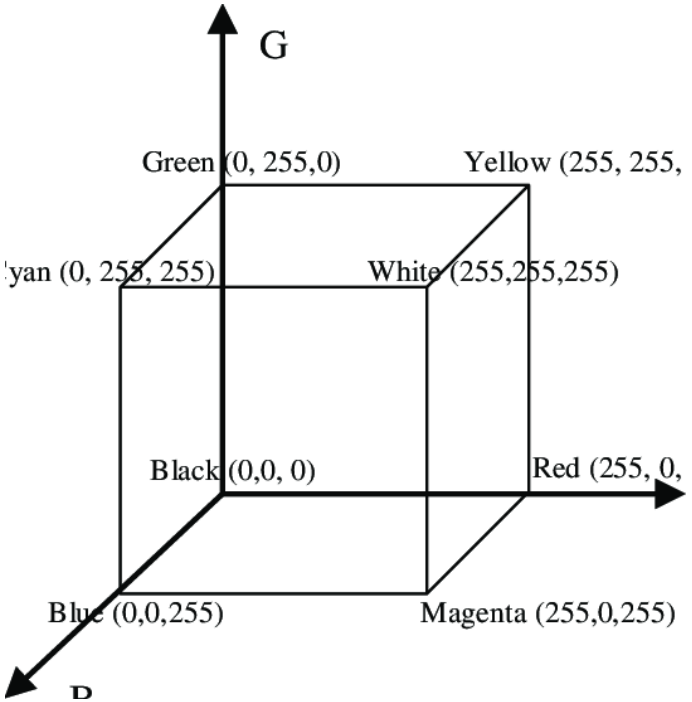
Color Interpolation: raw2rgb



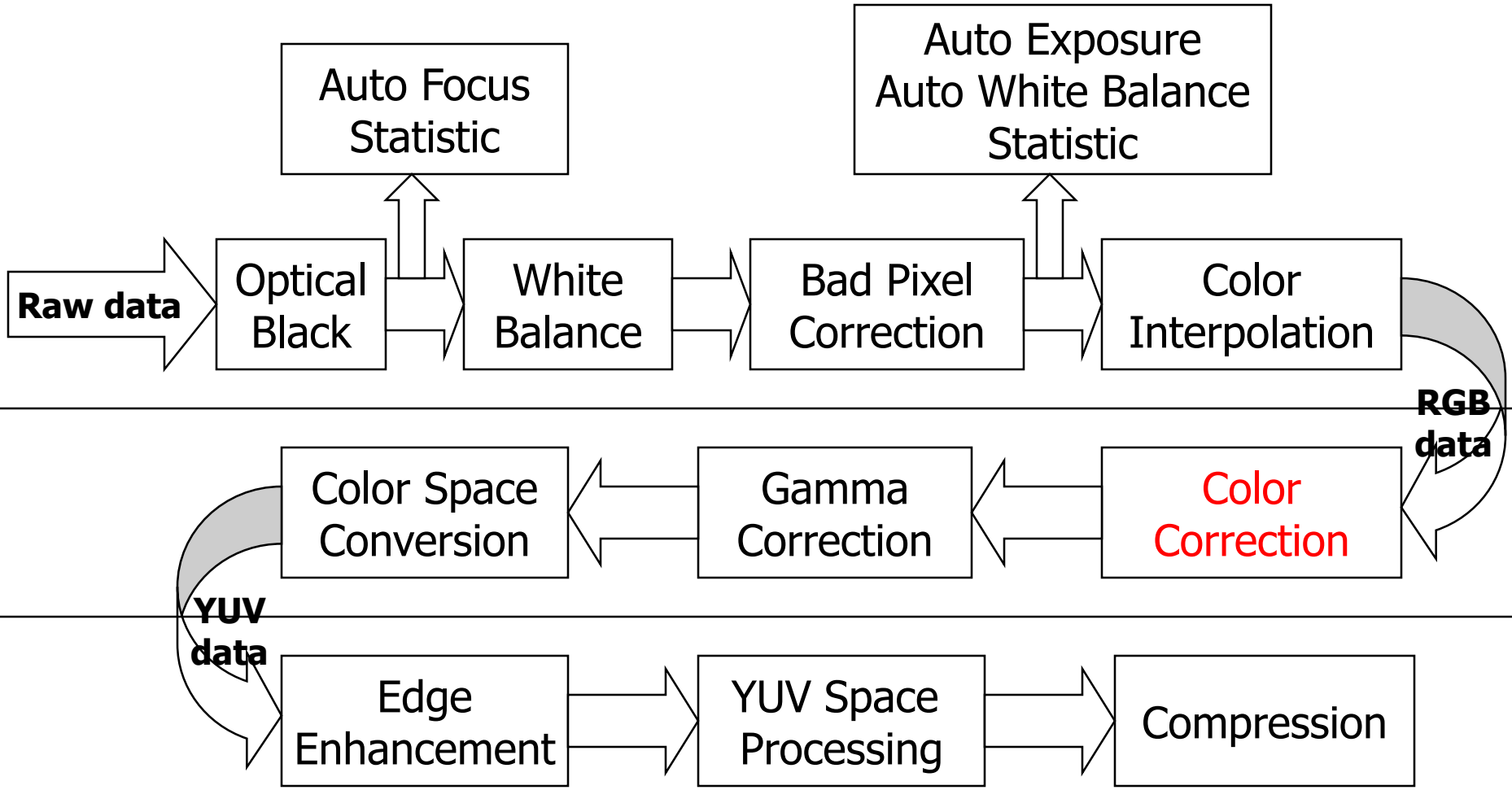
Raw Image



RGB Image



A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Color Correction

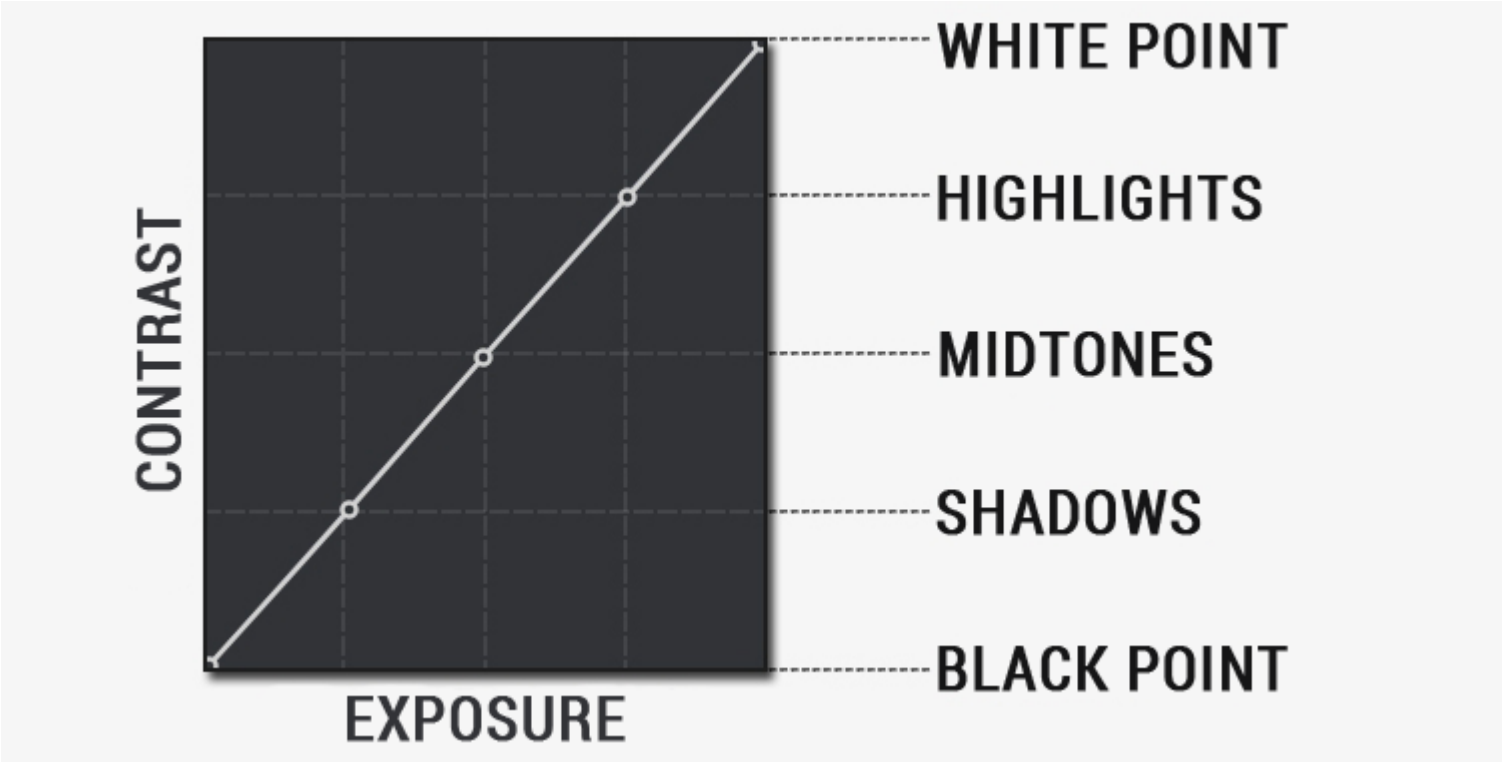
ISP targets on matching human perception

Color Correction

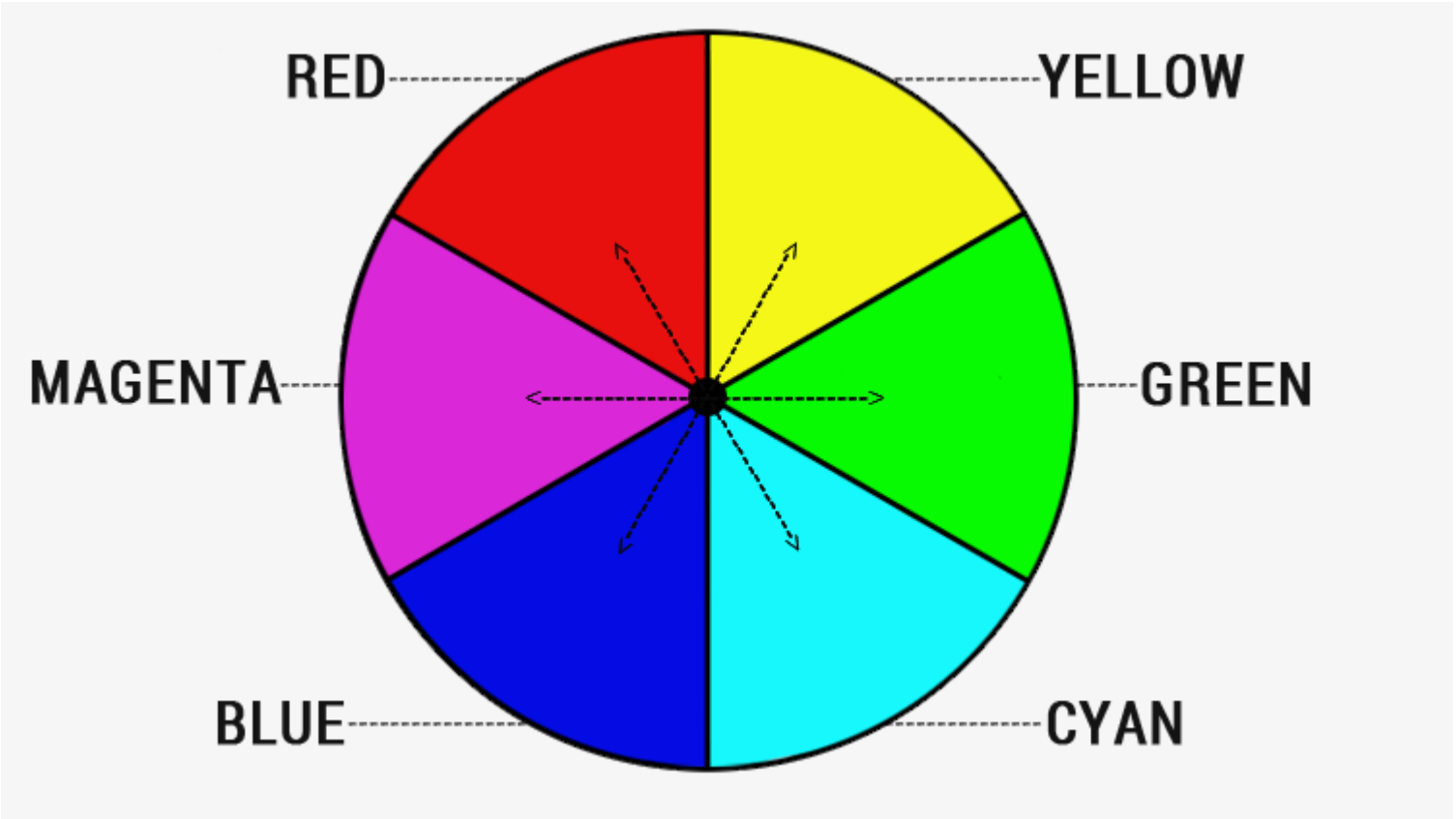
ISP targets on matching human perception

Color Correction

Color Correction



Color Correction



Color Correction

Tone Mapping

Tone mapping is a technique used in image processing and computer graphics to **map one set of colors to another** to approximate the appearance of high-dynamic-range (HDR) images in a medium that has a more limited dynamic range. Point-outs, CRT, or LCD monitors, and projectors all have a limited dynamic range that is inadequate to reproduce the full range of light intensities present in natural scenes.

Tone mapping addresses the problem of strong contrast reduction from the scene radiance to the displayable range while **preserving the image details and color appearance important** to appreciate the original scene content.

Tone Mapping

The goals of tone mapping can be differently stated depending on the particular application. In some cases **producing just aesthetically pleasing images** is the main goal, while other applications might emphasize **reproducing as many image details as possible**, or **maximizing the image contrast**.

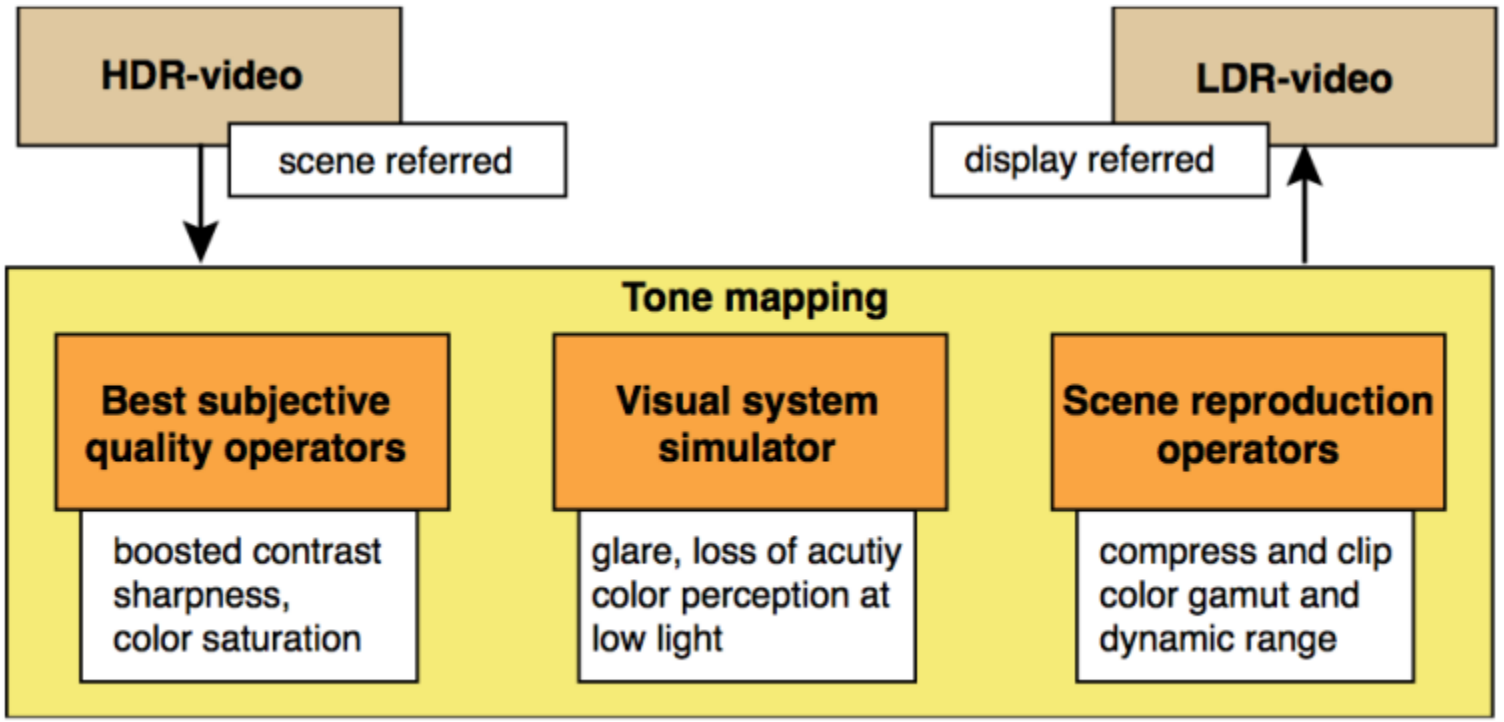
Various tone mapping operators have been developed in the recent years. They all can be divided in two main types:
(a) ***global* (or *spatially uniform*) operators**: they are non-linear functions based on the luminance and other global variables of the image.
(b) ***local* (or *spatially varying*) operators**: the parameters of the non-linear function change in each pixel, according to features extracted from the surrounding parameters.

Tone Mapping

- Map tone curve to **get better image**
- Similar to histogram adjustment or Photoshop's curve function
- For Y channel only

Three intents of tone-mapping

1. Scene reproduction operator
2. Visual system simulator
3. Best subjective quality



Rafał Mantiuk, Tone Mapping, http://www.bangor.ac.uk/mantiuk/

Intent #1: Scene reproduction problem

Real-world

Display

Goal: map colors to a restricted color space

Rafał Mantiuk, Tone Mapping, http://www.bangor.ac.uk/mantiuk/

Intent #2: Visual system simulator

Real-world

The eye adapted to the display viewing conditions

Display

The eye adapted to the real-world viewing conditions

Goal: match color appearance

Rafał Mantiuk, Tone Mapping, http://www.bangor.ac.uk/mantiuk/

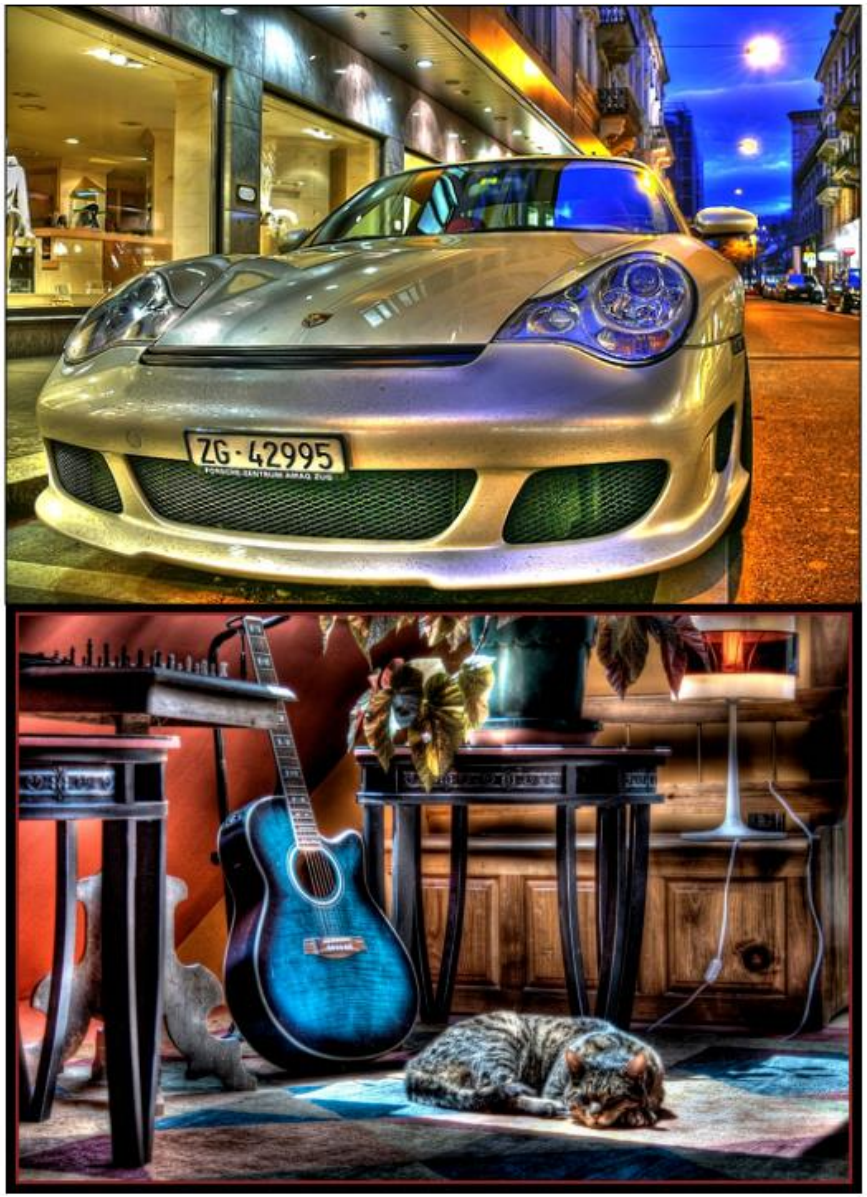
Visual system simulator - example

- Simulation of glare

Rafał Mantiuk, Tone Mapping, http://www.bangor.ac.uk/mantiuk/

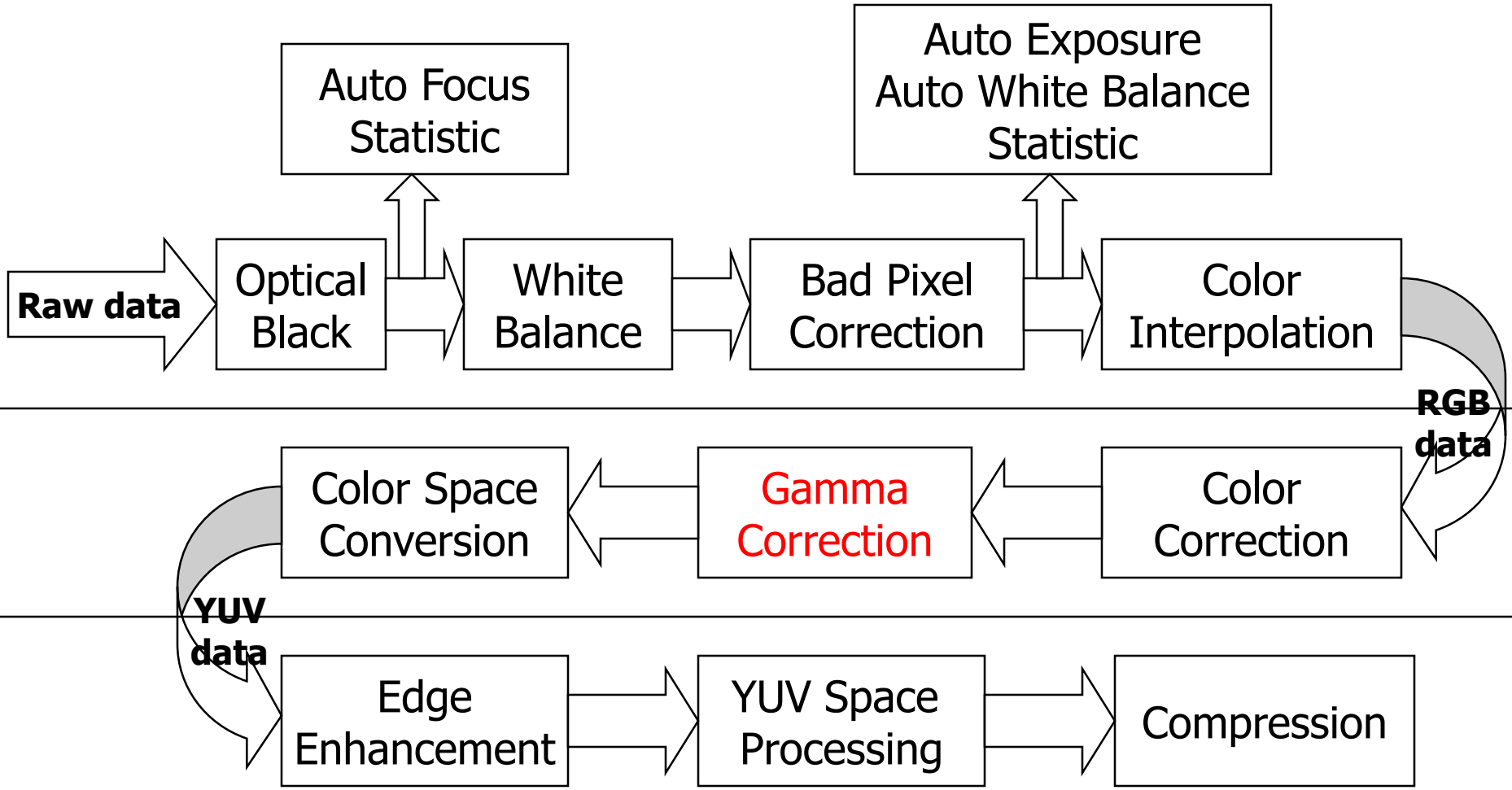
Intent #3: Best subjective quality

- Tools
  - Photoshop
  - Lightroom
  - Photomatix
- Techniques
  - Color-grading
- Often artistic intent



Rafał Mantiuk, Tone Mapping, http://www.bangor.ac.uk/mantiuk/

A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Gamma Correction

Gamma is an important but seldom understood characteristic of virtually all digital imaging systems. It defines the **relationship** between a **pixel's numerical value** and its **actual luminance**.

Without gamma, **shades** captured by digital cameras wouldn't appear as they did to our eyes (on a standard monitor). It's also referred to as gamma correction, gamma encoding or gamma compression, but these all refer to a similar concept.

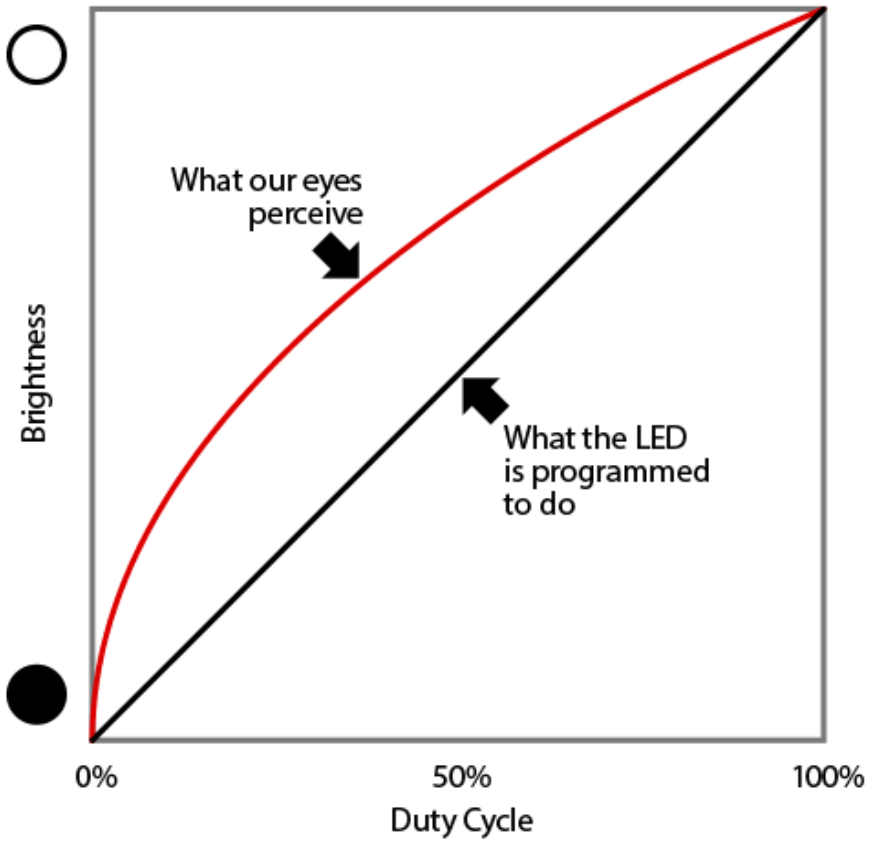
Understanding how gamma works can improve one's exposure technique, in addition to helping one make the most of image editing.

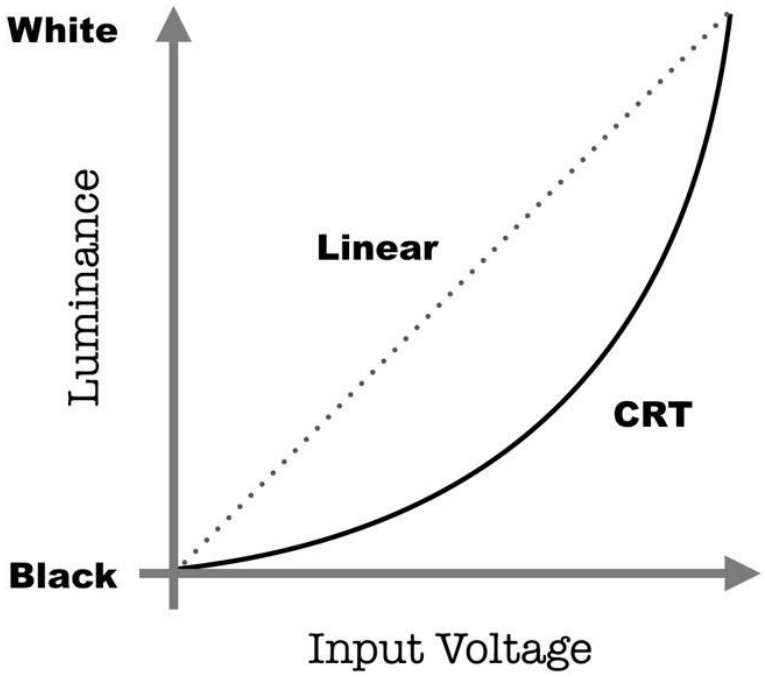
Gamma Correction

1. Our eyes do not perceive light the way cameras do. With a digital camera, when twice the number of photons hit the sensor, it receives twice the signal (a "linear" relationship). That's not how our eyes work. Instead, we perceive twice the light as being only a fraction brighter — and increasingly so for higher light intensities (a "nonlinear" relationship).

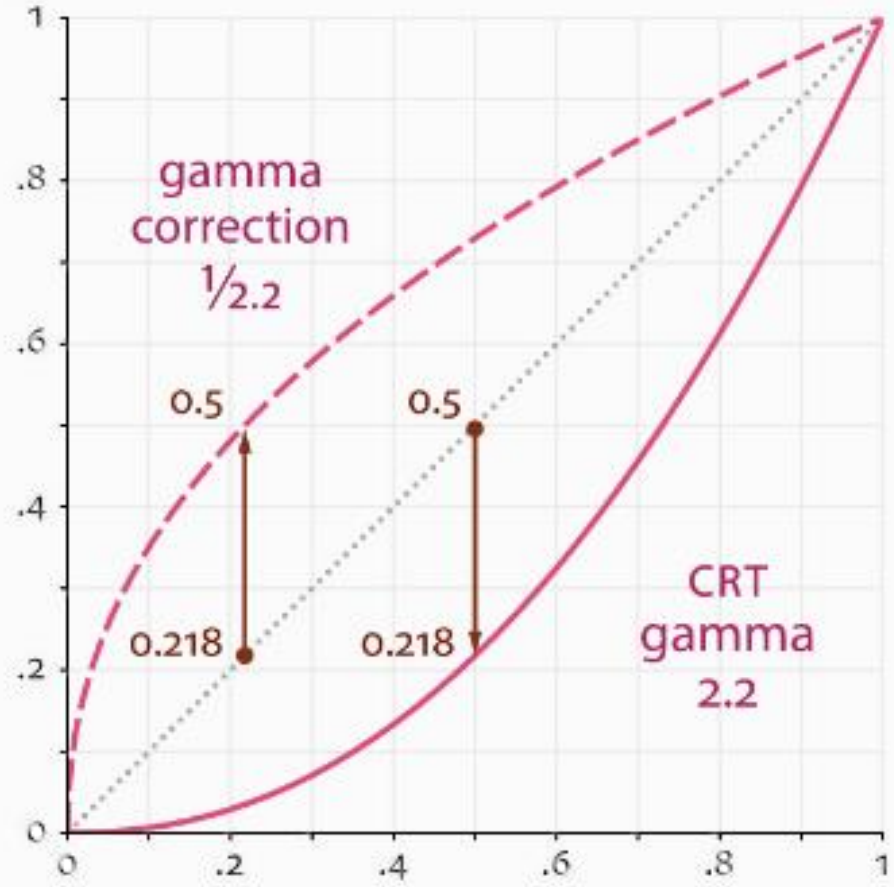
2. **Gamma encoded images store tones more efficiently**. Since gamma encoding redistributes tonal levels closer to how our eyes perceive them, fewer bits are needed to describe a given tonal range. Otherwise, an excess of bits would be devoted to describe the brighter tones (where the camera is relatively more sensitive), and a shortage of bits would be left to describe the darker tones.

Gamma Correction

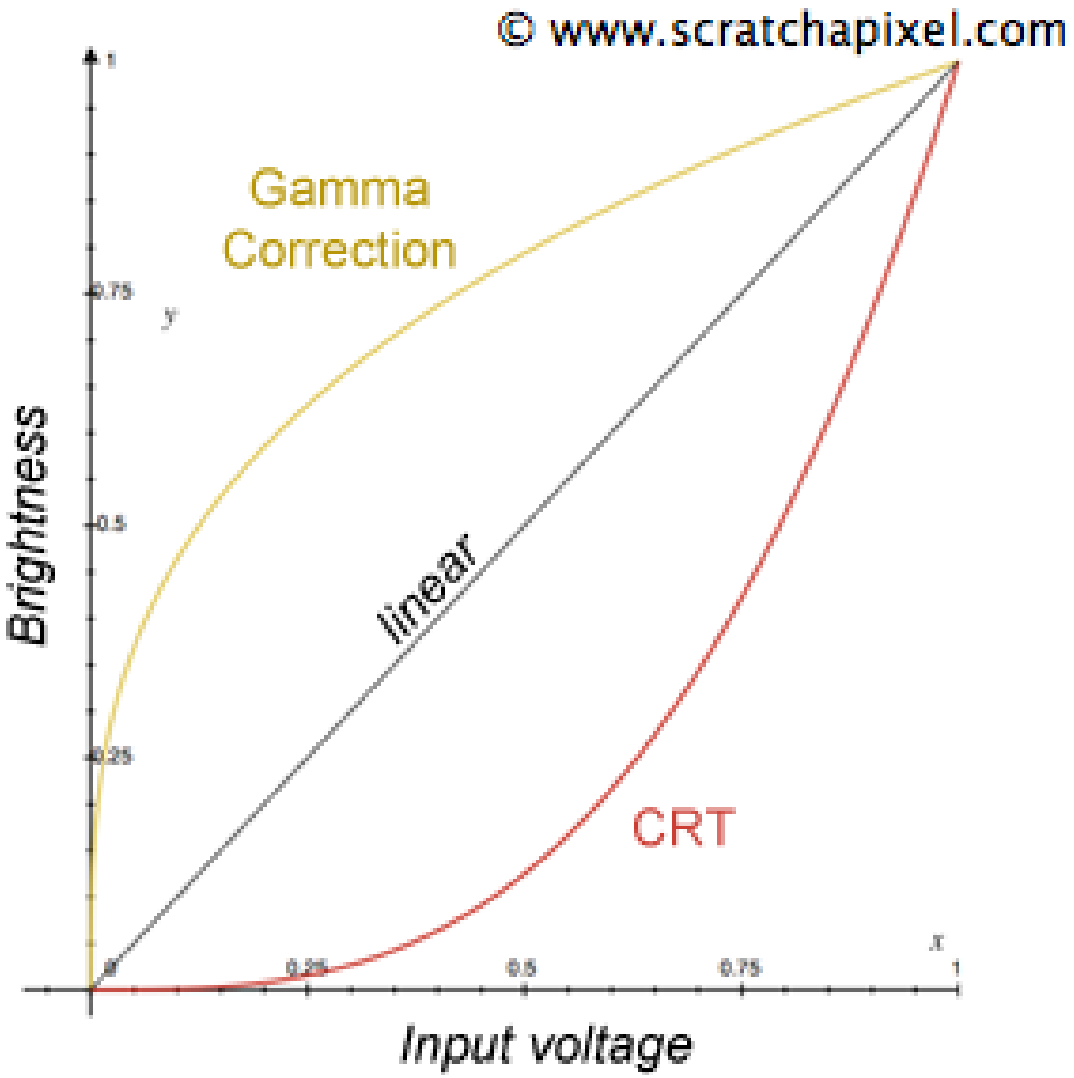




Gamma Correction



Gamma Correction



Gamma Correction

Linear RAW Image
(image gamma = 1.0)

Gamma Encoded Image
(image gamma = 1/2.2)

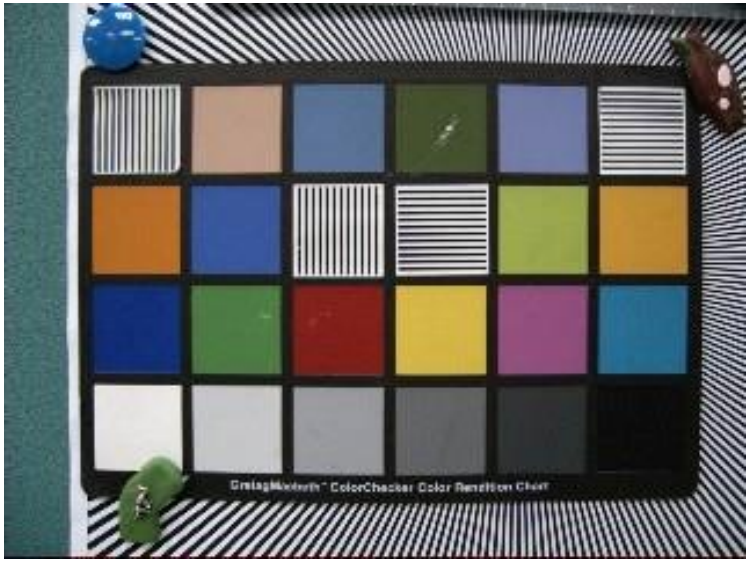
Color and Gamma Correction

- Image sensor's color sensitivity is different from human eyes.
- A 3x3 matrix multiplication and a nonlinear gamma mapping are used to correct color to match TRUE color. However, **there is no TRUE color**.
- Color target is used to replace TRUE color.
- **Correction means solving best matrix and gamma**.
- sRGB assumes 2.2 gamma correction but...
  - No one really cares.
  - **Perceptive correctness is the truth**.

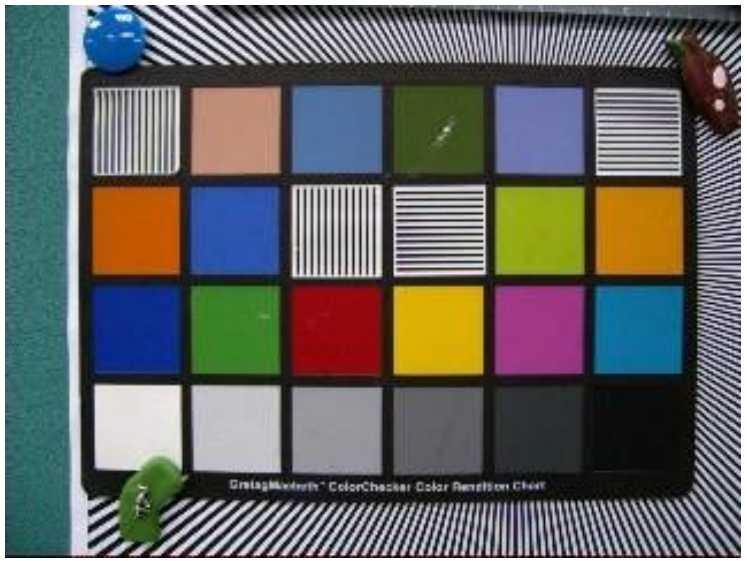
Influence of Color and Gamma

Matrix

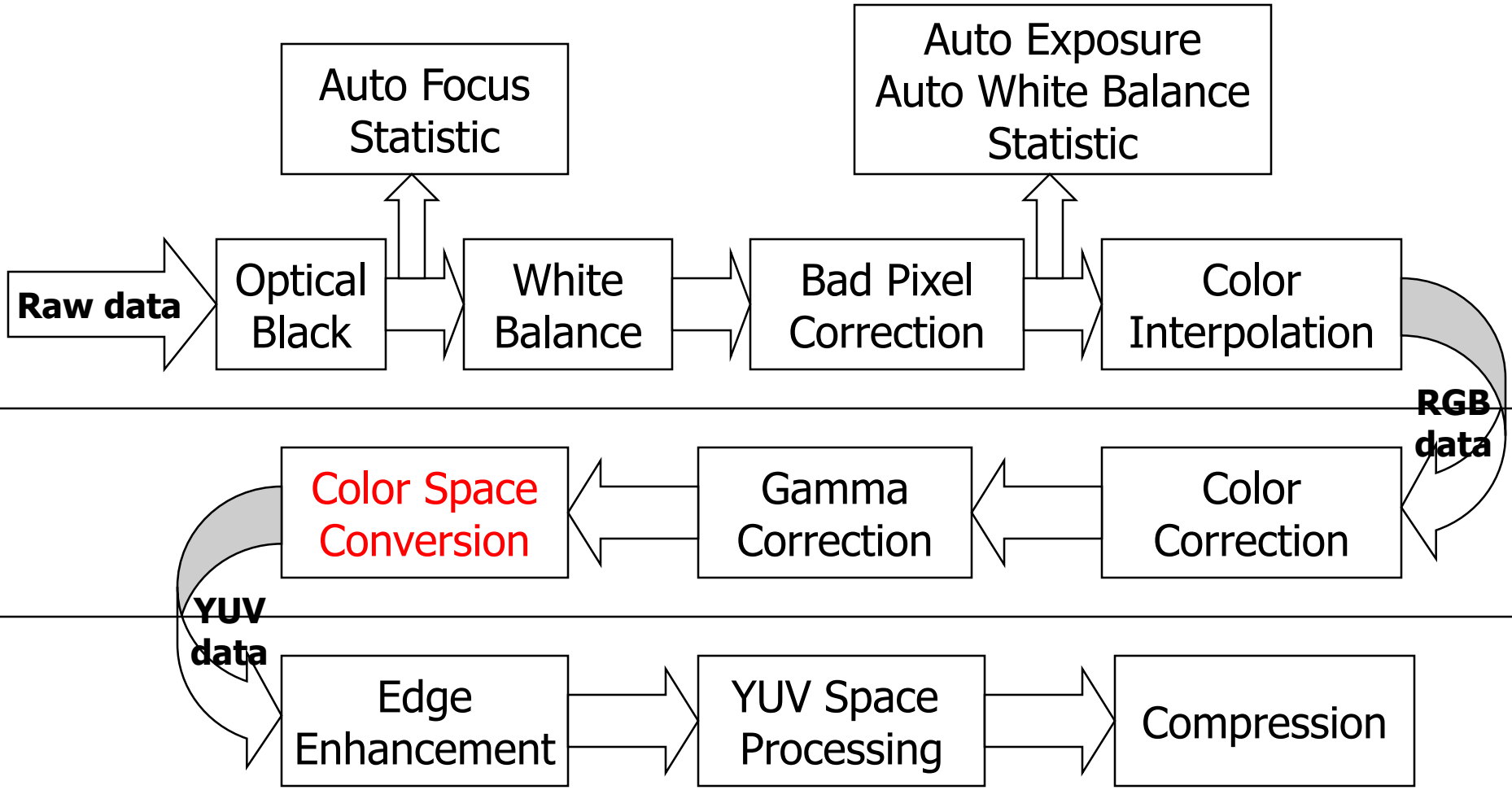
Gamma



Gamma



A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

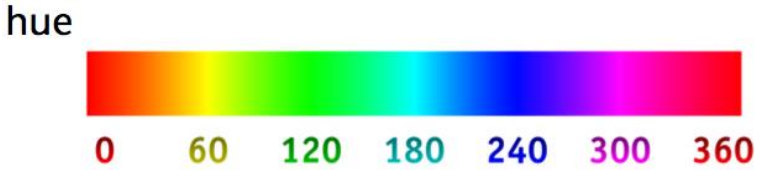
Color Space Conversion

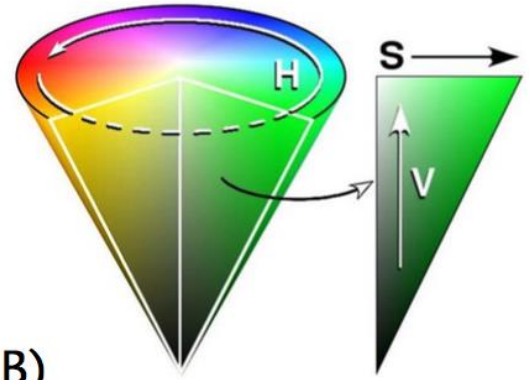
- RGB <-> YUV
- Prepare for brightness/contrast/hue/saturation adjustment and JPEG (Joint Photographic Experts Group) compression
- Typically done by 3x3 matrix multiplication

$$\begin{pmatrix} R \\ G \\ B \end{pmatrix} = \begin{pmatrix} 3.1406 & -1.5372 & -0.4986 \\ -0.9689 & 1.8758 & 0.0415 \\ 0.0557 & -0.2040 & 1.0570 \end{pmatrix} \begin{pmatrix} X_{D65} \\ Y_{D65} \\ Z_{D65} \end{pmatrix}$$

Color Space Conversion

- RGB <-> HSV





- colour cone
  - H = hue / colour in degrees $\in$ [0,360]
  - S = saturation $\in$ [0,1]
  - V = value $\in$ [0,1]
- conversion RGB $\rightarrow$ HSV
  - V = max = max (R, G, B), min = min (R, G, B)
  - S = (max – min) / max (or S = 0, if V = 0)
  - $$H = 60 \times \begin{cases} 0 + (G - B) / (\max - \min), & \text{if } \max = R \\ 2 + (B - R) / (\max - \min), & \text{if } \max = G \\ 4 + (R - G) / (\max - \min), & \text{if } \max = B \end{cases}$$

    $H = H + 360$, if $H < 0$

Color Space Conversion

- RGB <-> Lab

LAB $\rightarrow$ RGB color space conversion

$\triangleright$ convert LAB $\rightarrow$ XYZ

$\triangleright$ convert XYZ $\rightarrow$ RGB [38]

$$\begin{pmatrix} R_s \\ G_s \\ B_s \end{pmatrix} = \begin{pmatrix} 3.2410 & -1.5374 & -0.4986 \\ -0.9692 & 1.8760 & 0.0416 \\ 0.0556 & -0.2040 & 1.0570 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

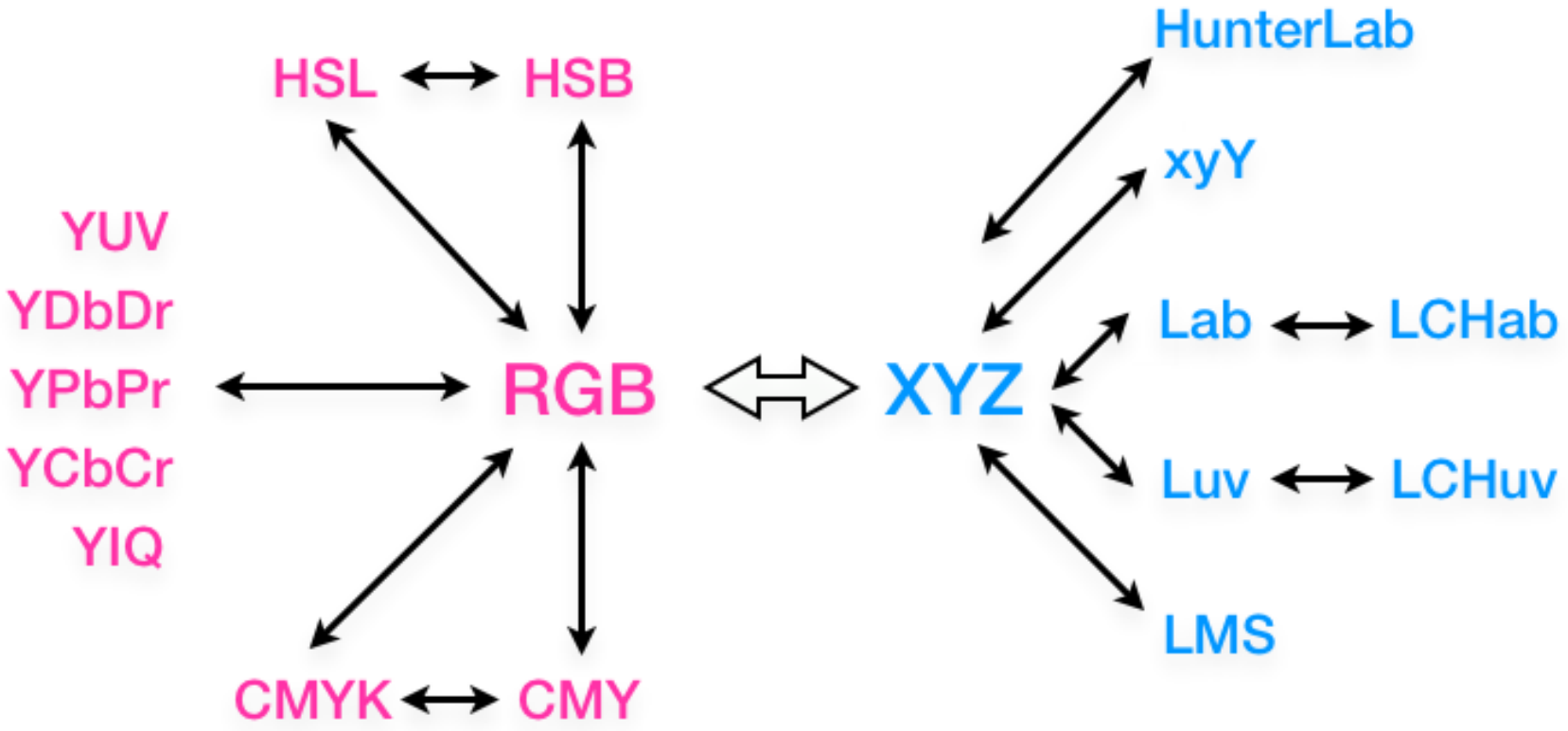
sRGB gamma correction

$$R' = \begin{cases} 12.92R_s & \text{if } R_s \leq 0.00304 \\ 1.055R_s^{(1.0/2.4)} - 0.055 & \text{otherwise} \end{cases}$$

Similarly for G' and B'. Finally,

$$\begin{aligned} R &= 255.0 \cdot R' \\ G &= 255.0 \cdot G' \\ B &= 255.0 \cdot B' \end{aligned}$$

Color Space Conversion



Color Space Conversion

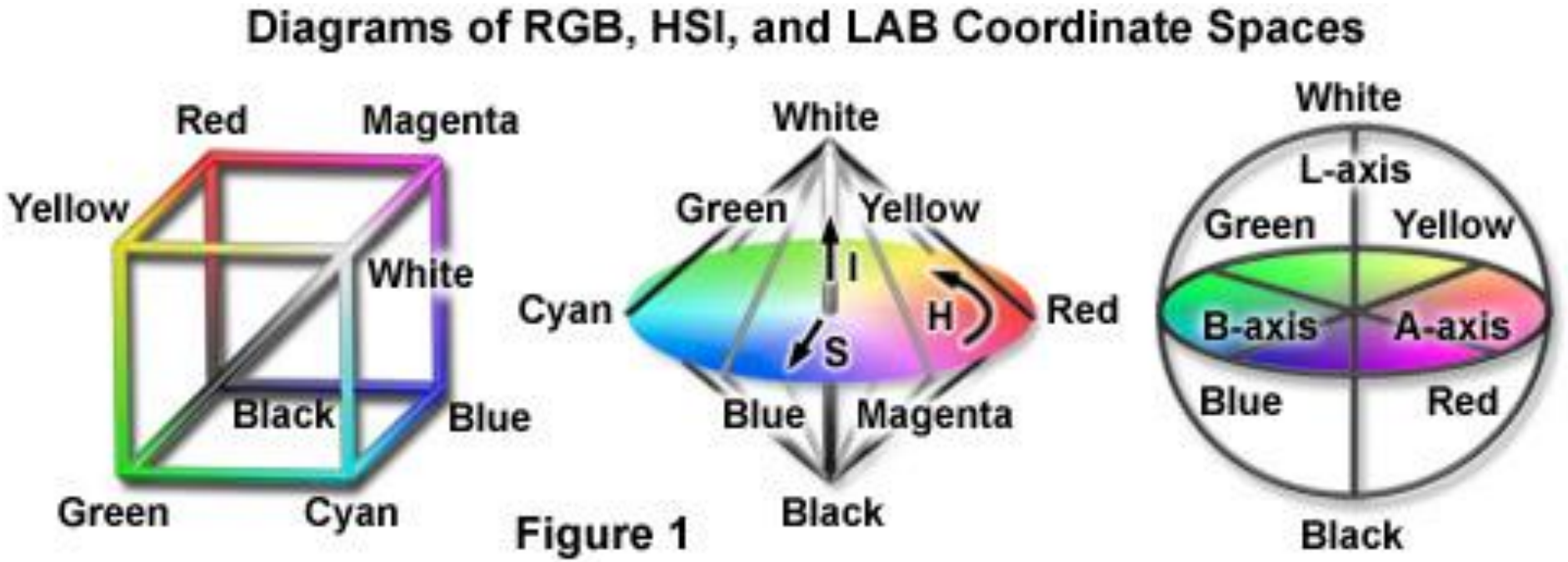
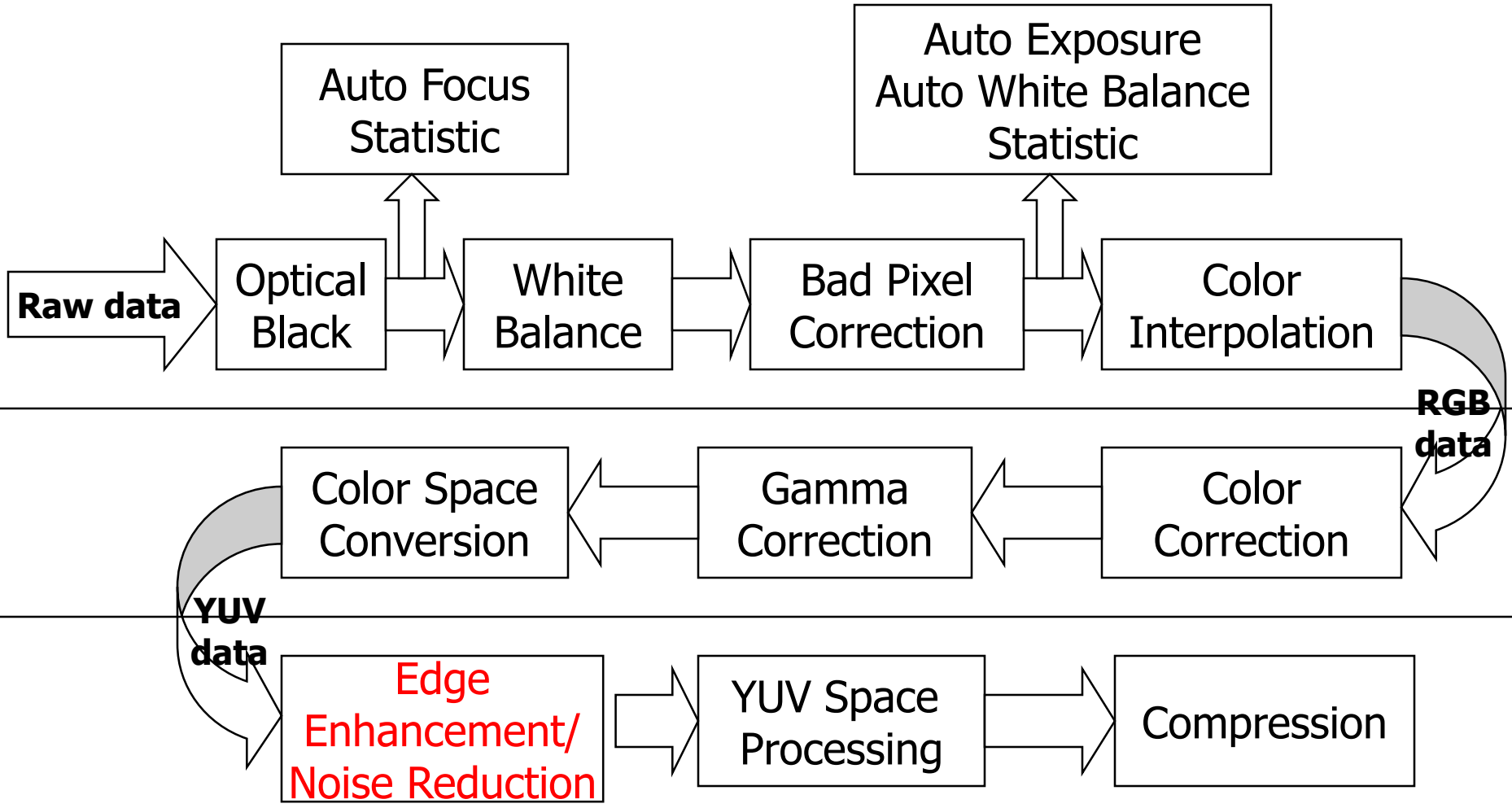


Figure 1

A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

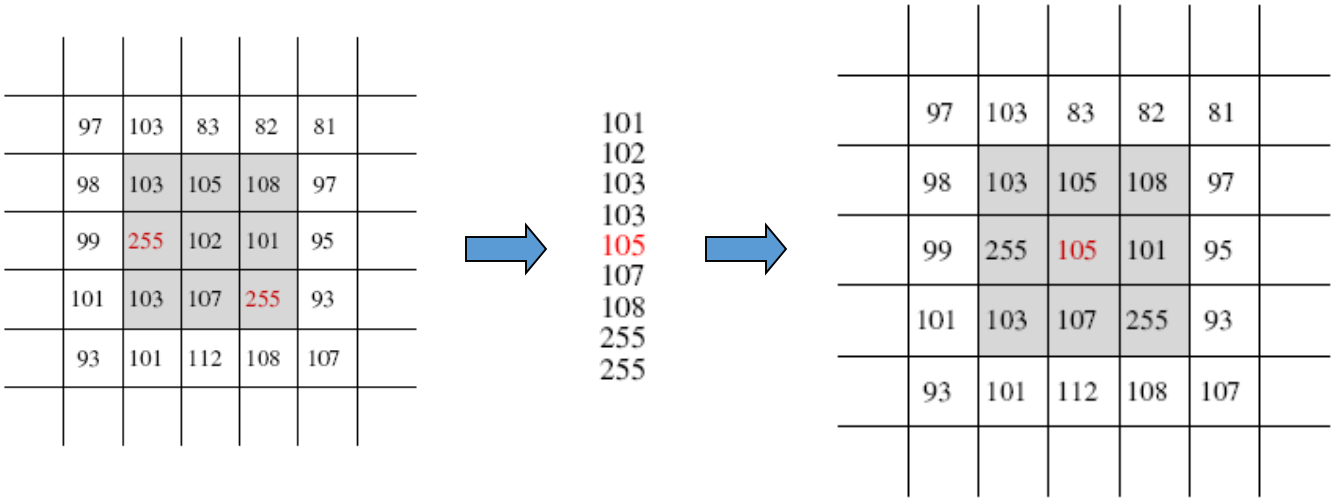
www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Noise Reduction

- What is noise?
  - **Unnatural artifacts**: power, readout, flicker...
  - Too many possible noise sources
  - Focus on removing noise sources first
- Should we reduce noise in raw or YUV space?
  - Raw data space
    - Prevent noise from going into remaining steps of image pipeline and being magnified further
  - YUV space
    - More information to reduce noise correctly.

Noise Reduction via Median Filter

- For each neighbor in image, sliding the window
- Sort pixel values
- Set the center pixel to the median



Noise Reduction via Mean Filtering

Median Filter

- Problem with Averaging Filter
  - Blur edges and details in an image
  - Not effective for impulse noise (Salt-and-pepper)
- Median filter:
  - Taking the median value instead of the average or weighted average of pixels in the window
    - Median: sort all the pixels in an increasing order, take the middle one
  - The window shape does not need to be a square
  - Special shapes can preserve line structures
- Order-statistics filter
  - Instead of taking the mean, rank all pixel values in the window, take the n-th order value.
  - E.g. max or min

Noise Reduction via Mean Filtering

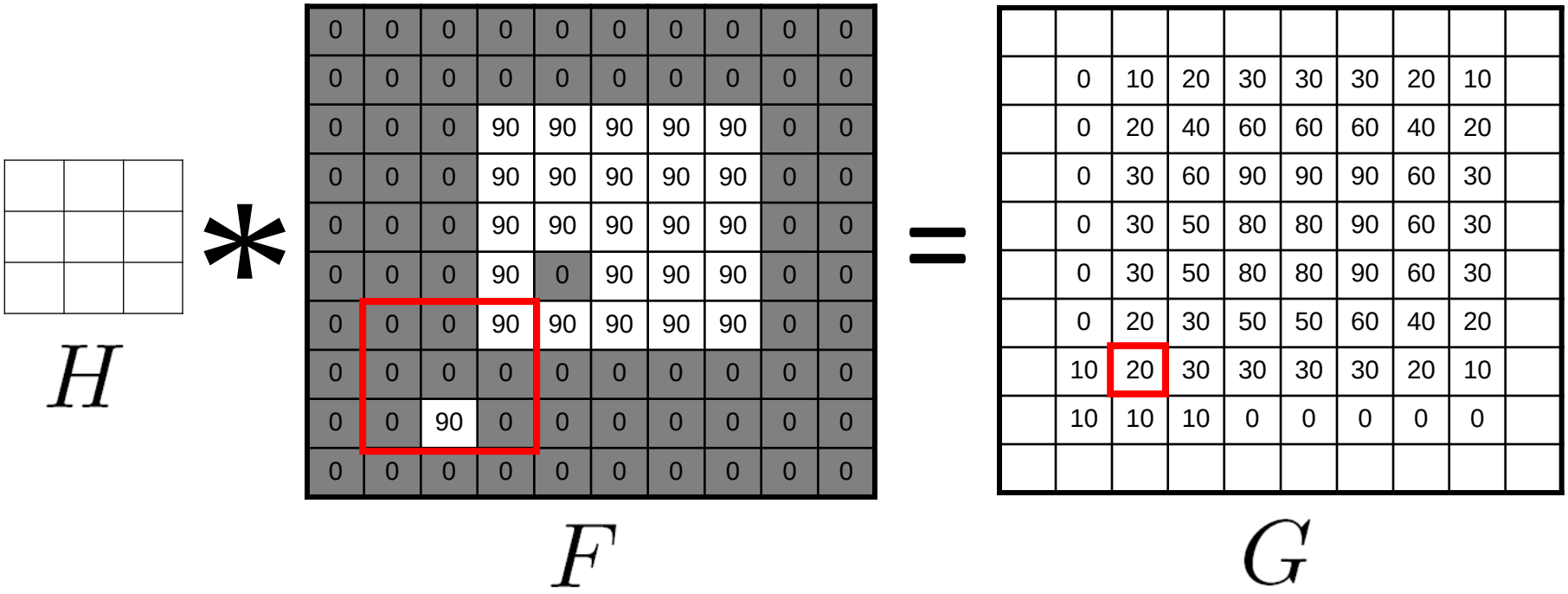
Example: 3x3 Median

100	100	100	100	100
100	200	205	203	100
100	195	200	200	100
100	200	205	195	100
100	100	100	100	100

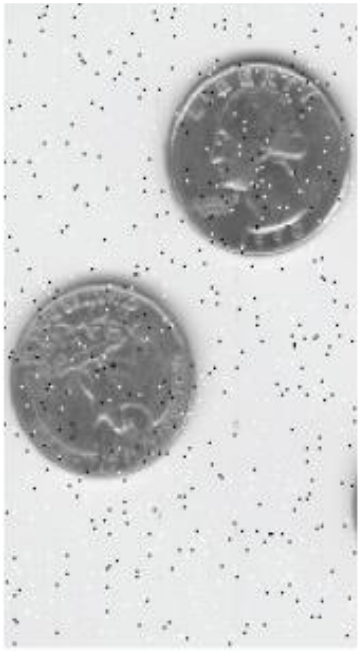
100	100	100	100	100
100	100	200	100	100
100	200	200	200	100
100	100	195	100	100
100	100	100	100	100

Matlab command: medfilt2(A,[3 3])

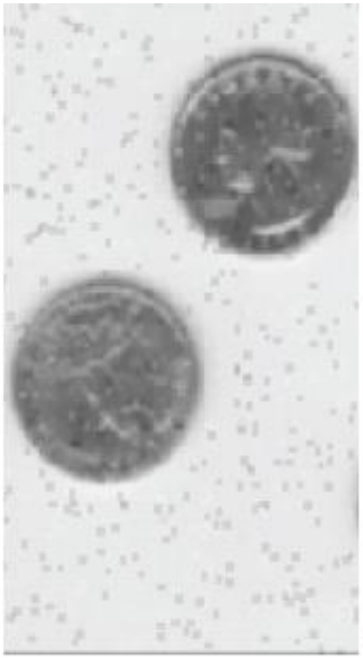
Noise Reduction via Mean Filtering



Median Filter



input



Gaussian filter

Median filter

Median Filter Examples

input

Median 7X7

Median Filter Examples

Median 3X3

Median 11X11

Median Filter Examples

Median 3X3

Median 11X11

Median Filter Properties

Can remove outliers (peppers and salts)

Window size controls size of structure

Preserve some details but sharp corners and edges might get lost

Common Problems

Mean: blurs image, removes simple noise, no details are preserved

Gaussian: blurs image, preserves details only for small σ.

Median: preserves some details, good at removing strong noise

Can we find a filter that not only smooths regions but preserves edges?

- yes, bilateral filter

What Is Bilateral Filter?

Bilateral

- Affecting or undertaken by two sides equally

Property:

- Convolution filter
- Smooth image but preserve edges
- Operates in the domain and the range of image

Noise Reduction via Bilateral Filter

A **bilateral filter** is a non-linear, edge-preserving, and noise-reducing smoothing filter for images. It replaces the intensity of each pixel with a weighted average of intensity values from nearby pixels.

The bilateral filter is defined as

$$I^{\text{filtered}}(x) = \frac{1}{W_p} \sum_{x_i \in \Omega} I(x_i) f_r(\|I(x_i) - I(x)\|) g_s(\|x_i - x\|),$$

and normalization term, W_p, is defined as

$$W_p = \sum_{x_i \in \Omega} f_r(\|I(x_i) - I(x)\|) g_s(\|x_i - x\|)$$

I^{filtered} is the filtered image;

I is the original input image to be filtered;

x are the coordinates of the current pixel to be filtered;

Ω is the window centered in x, so $x_i \in \Omega$ is another pixel;

f_r is the range kernel for smoothing differences in intensities (this function can be a Gaussian function);

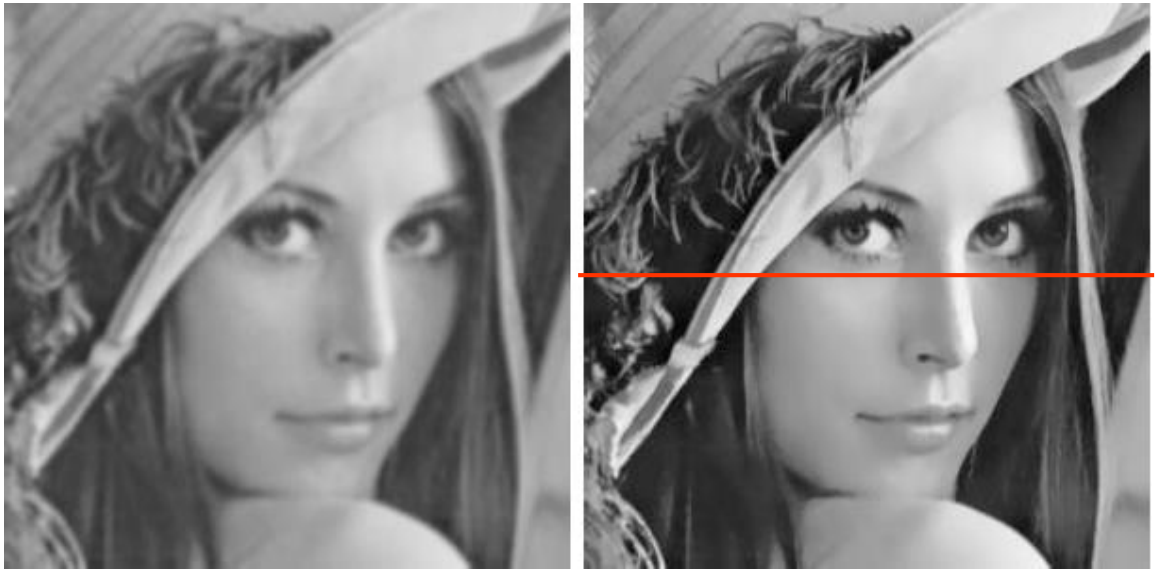
g_s is the spatial (or domain) kernel for smoothing differences in coordinates (this function can be a Gaussian function).

Bilateral Filter Example

Gaussian filter

Bilateral filter

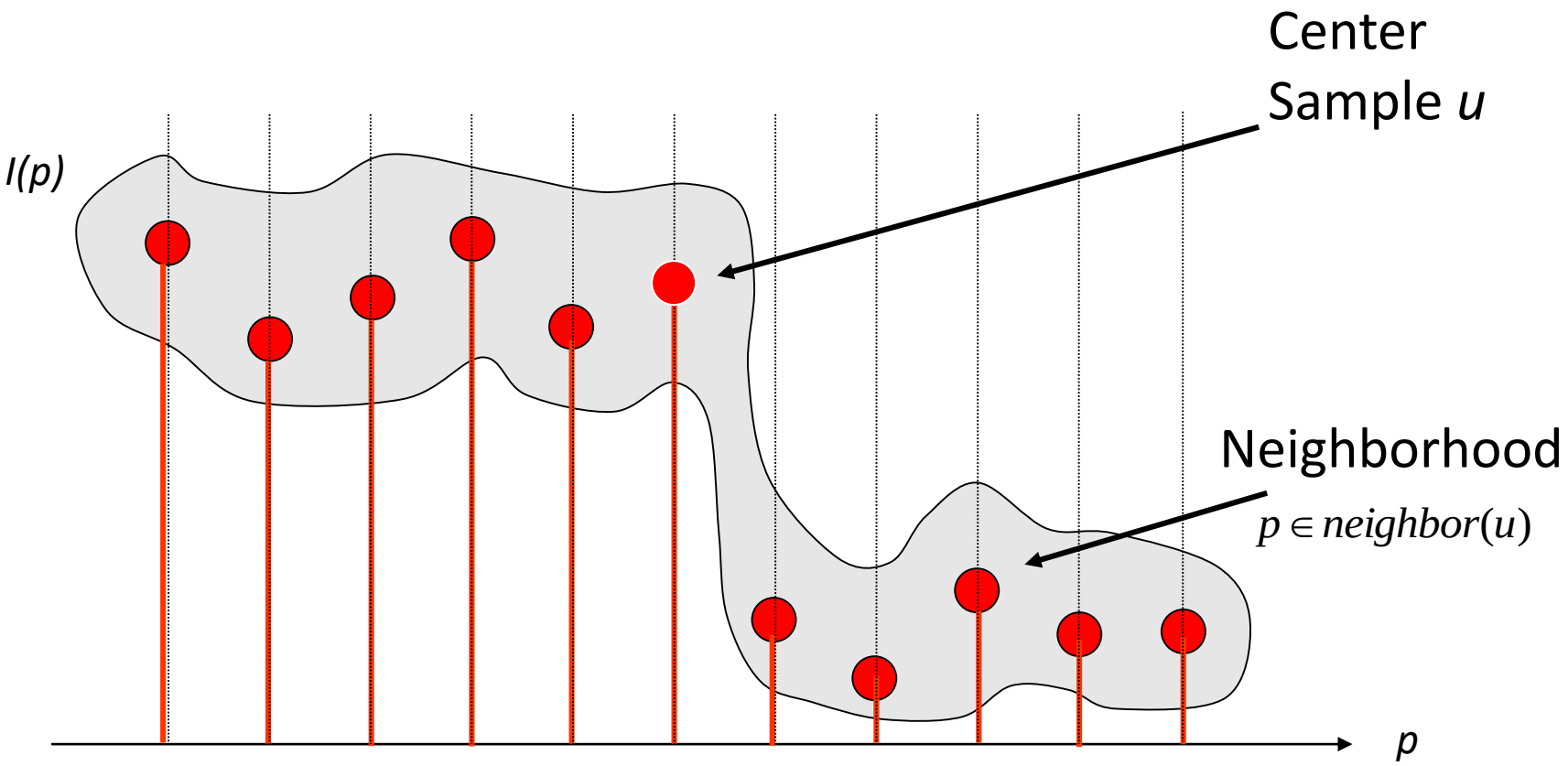
Bilateral Filter Example



Gaussian filter

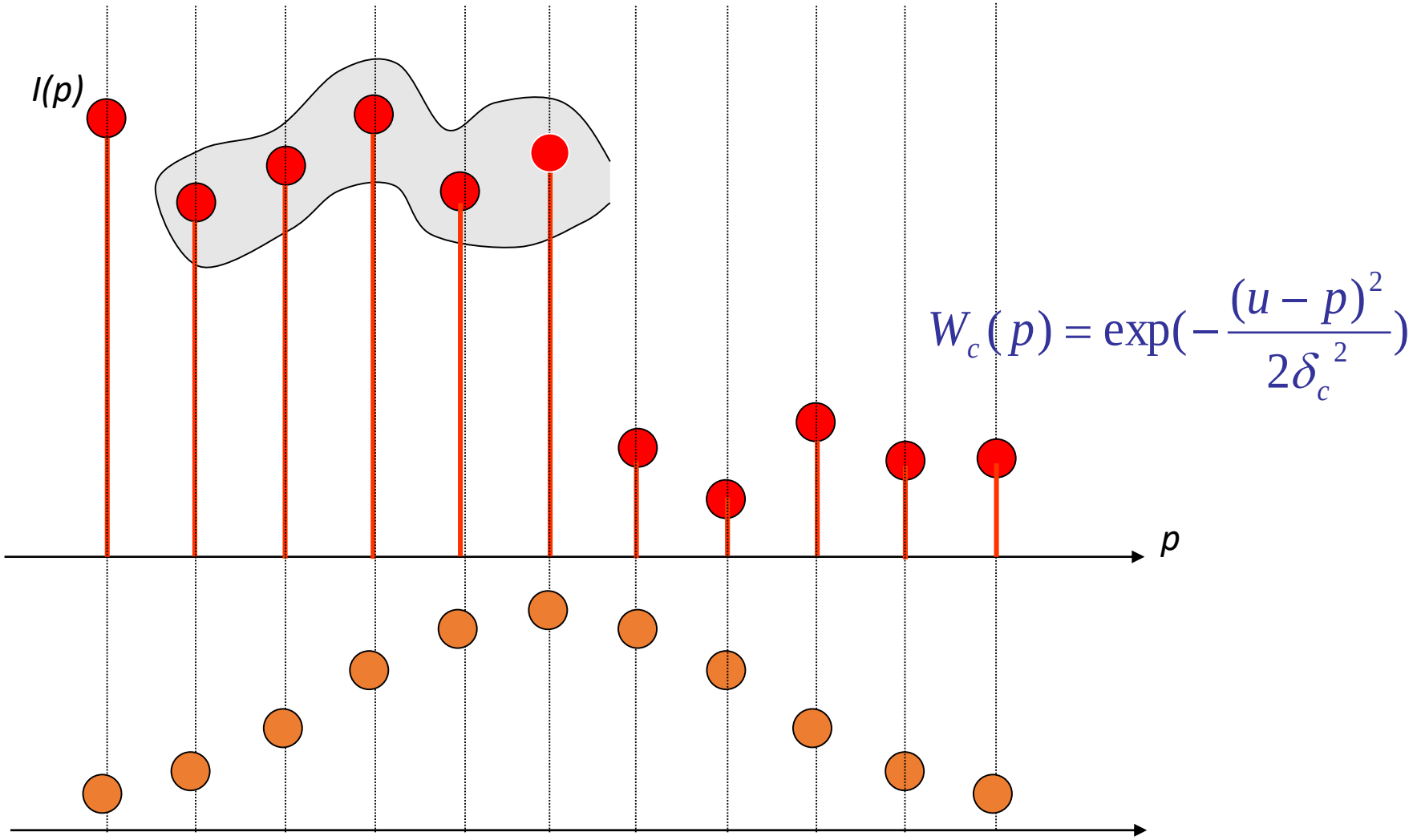
Bilateral filter

1D Graphical Example



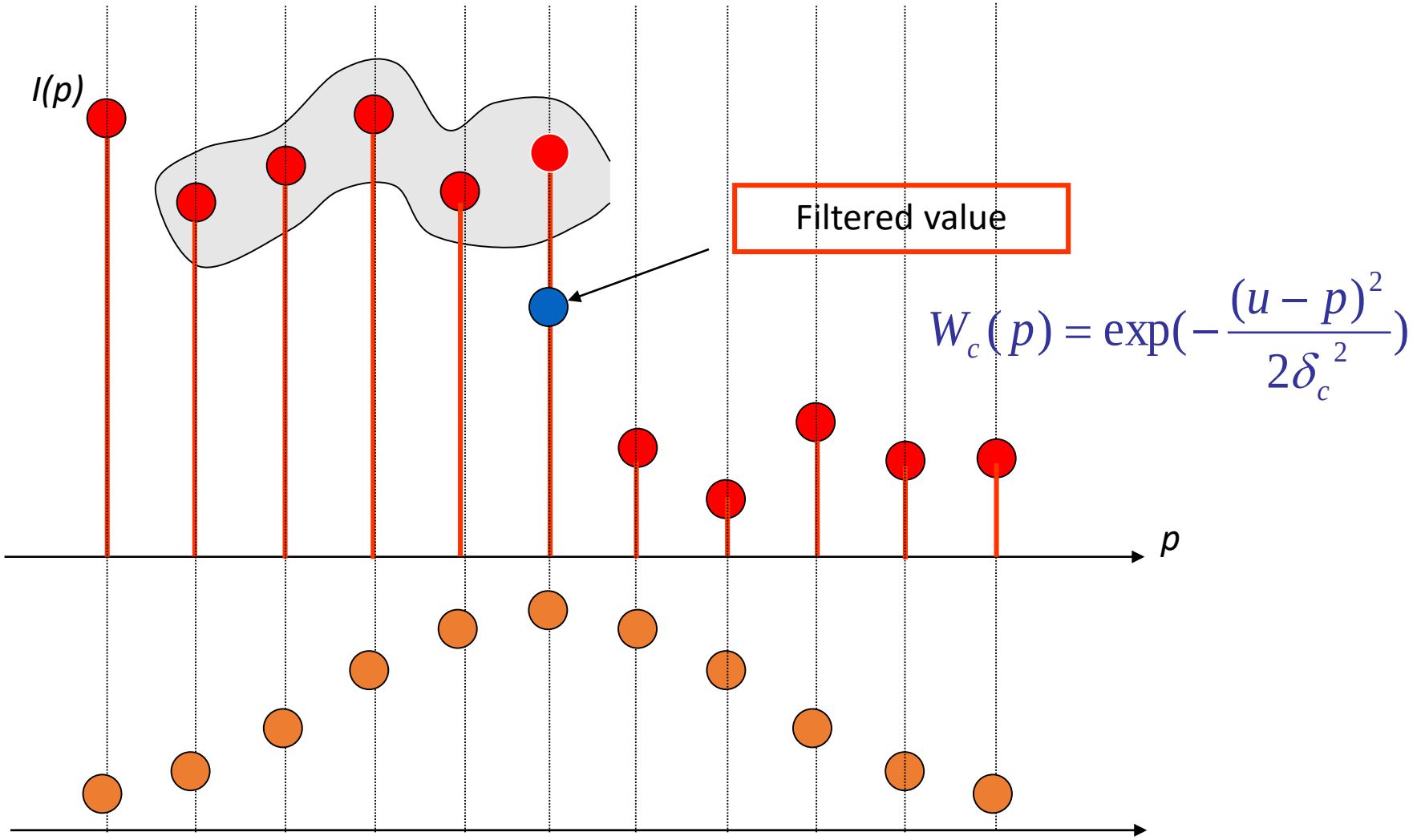
It is clear that in weighting this neighborhood, we would like to preserve the step

The Weights



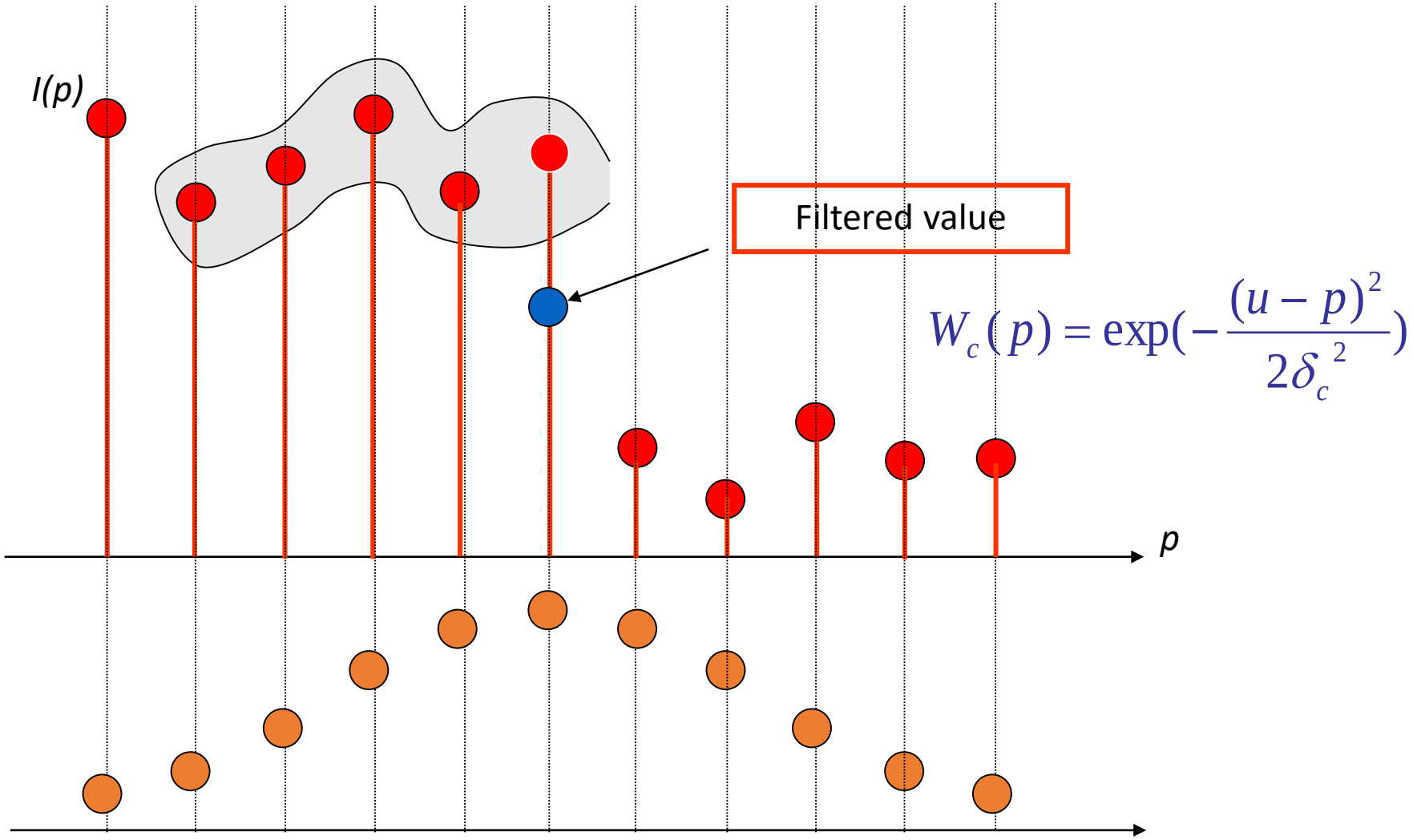
$$W_c(p) = \exp(-\frac{(u-p)^2}{2\delta_c^2})$$

Filtered Values

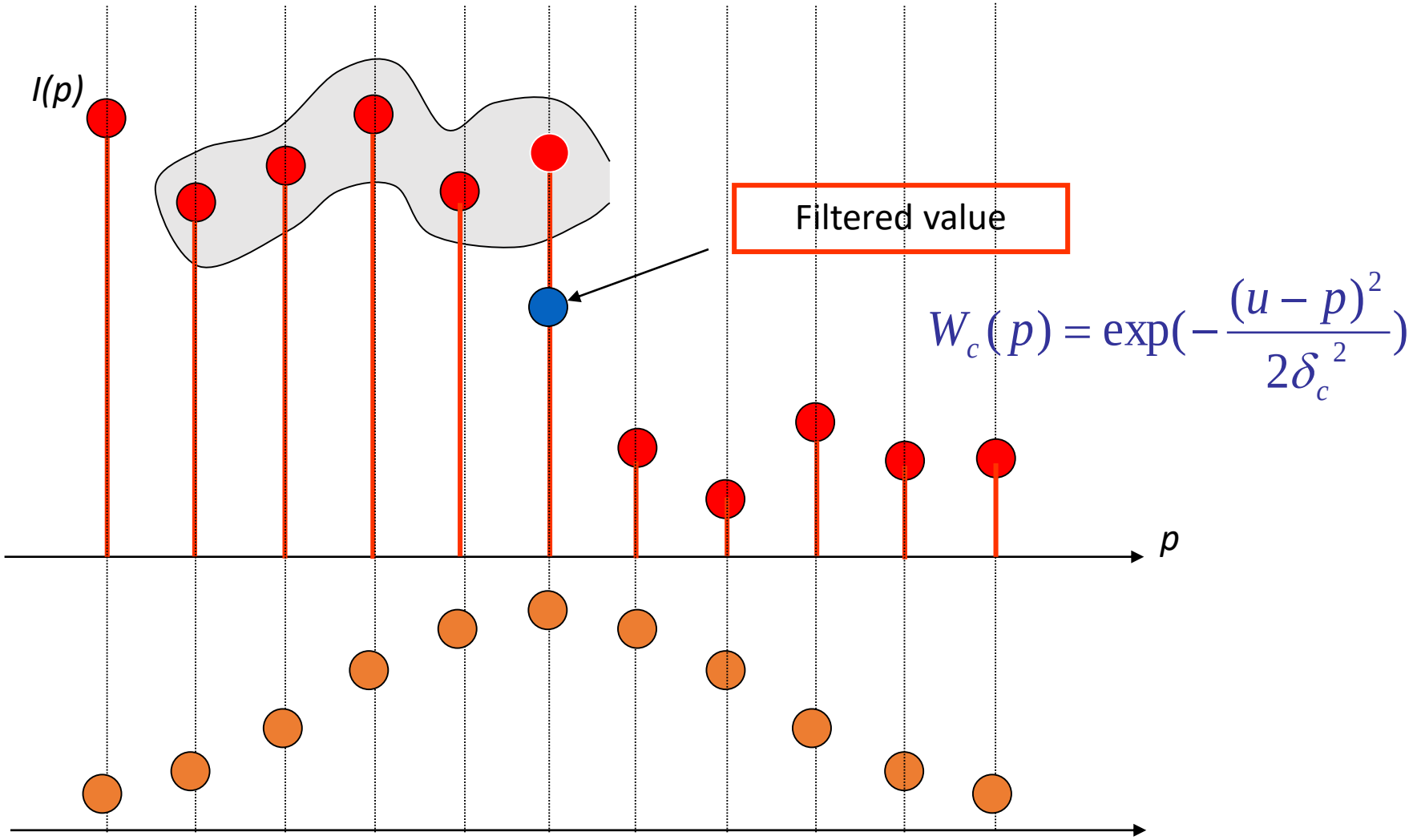


$$W_c(p) = \exp(-\frac{(u-p)^2}{2\delta_c^2})$$

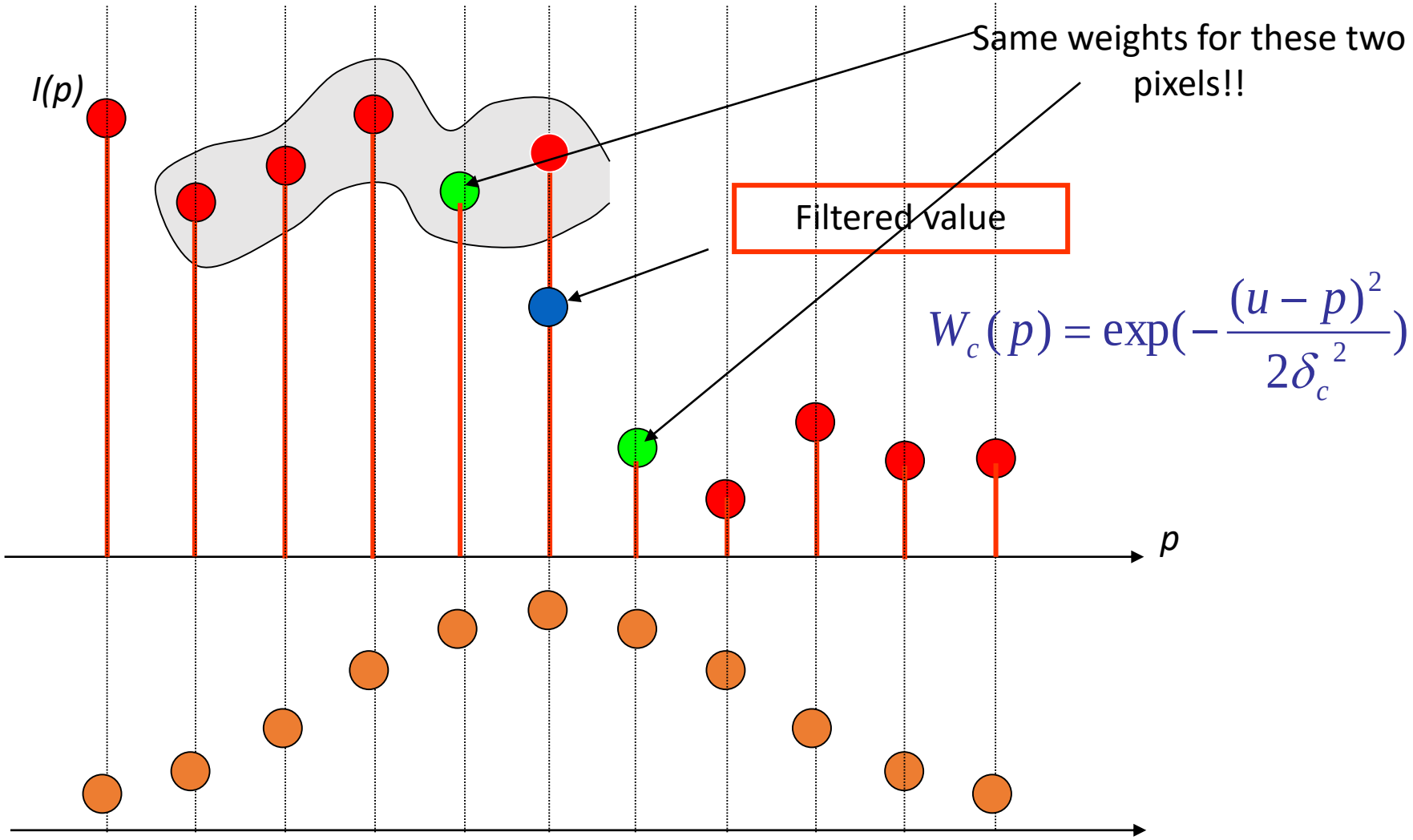
Edges Are Smoothed



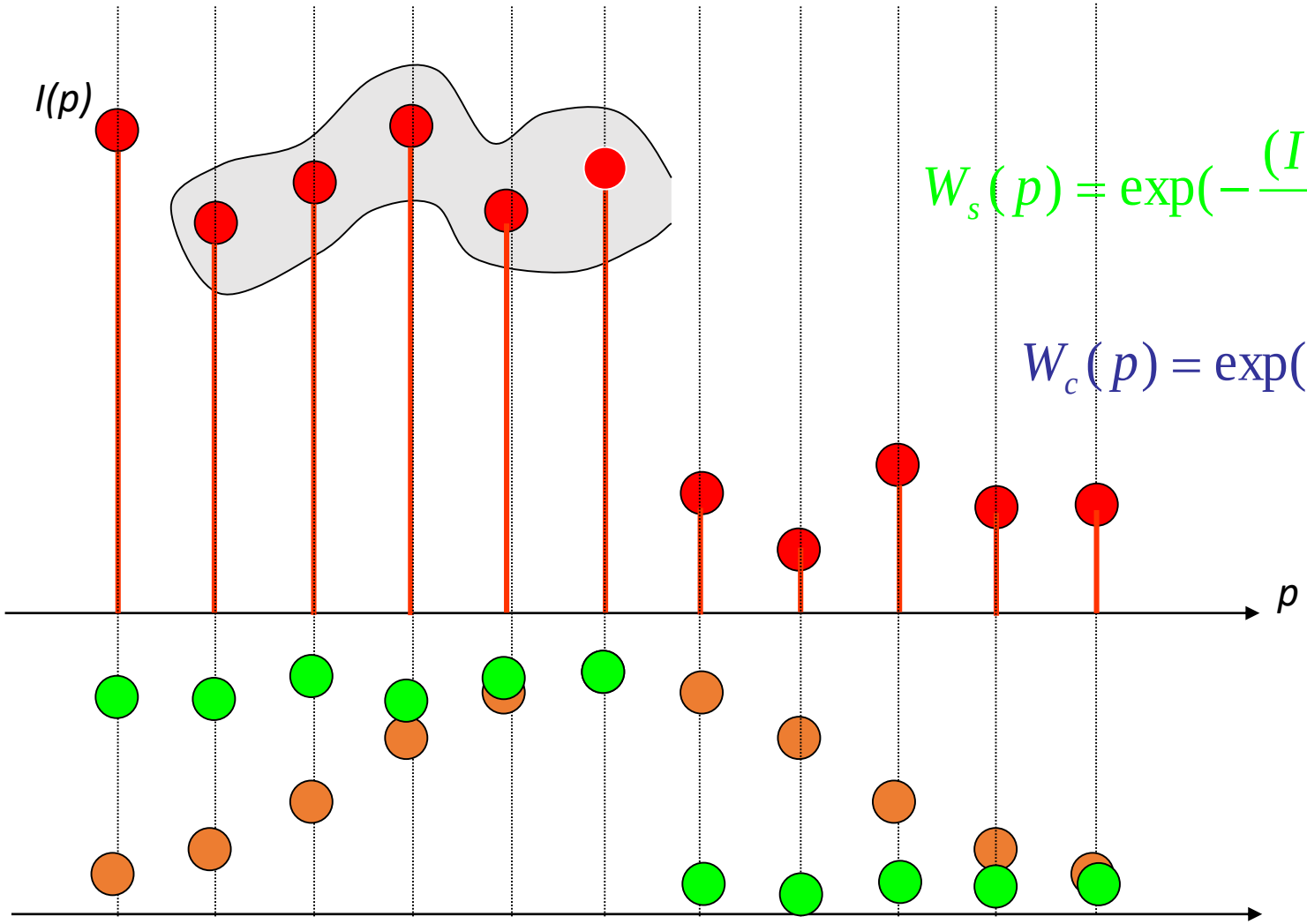
What Causes the Problem?



What Causes the Problem?



The Weights

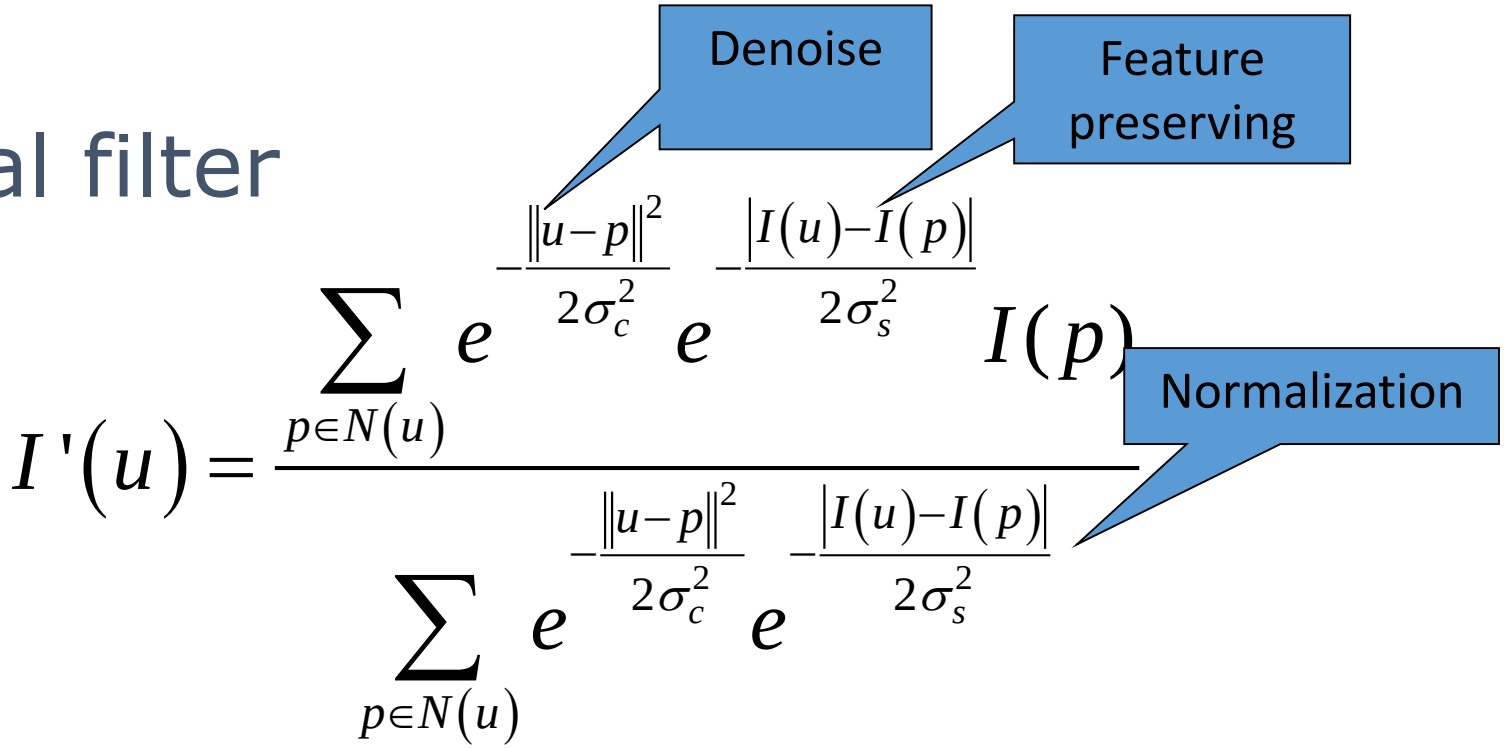


$$W_s(p) = \exp\left(-\frac{(I(u)-I(p))^2}{2\delta_s^{\,2}}\right)$$

$$W_c(p) = \exp\left(-\frac{(u-p)^2}{2\delta_c^{\,2}}\right)$$

Bilateral Filtering

Bilateral filter



Kernel Properties

$$I'(u) = \frac{\sum_p W_c(p) * W_s(p) * I(p)}{\sum_p W_c(p) * W_s(p)}$$

- Per each sample, we can define a 'Kernel' that averages its neighborhood
- This kernel changes from sample to sample!
- The sum of the kernel entries is 1 due to the normalization,
- The center entry in the kernel is the largest,
- Subject to the above, the kernel can take any form (as opposed to filters which are monotonically decreasing).

Filter Parameters

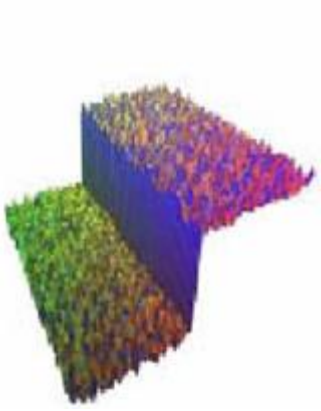
$$I'(u) = \frac{\sum_{p \in N(u)} W_c(p) * W_s(p) * I(p)}{\sum_{p \in N(u)} W_c(p) * W_s(p)}$$

As proposed by Tomasi and Manduchi, the filter is controlled by 3 parameters:

- *N(u)* – The neighbor size of the filter support,
- σ_c – The variance of the spatial distances,
- σ_s – The variance of the value distances,

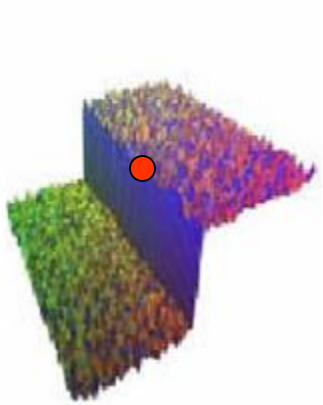
The filter can be applied for several iterations in order to further strengthen its edge-preserving smoothing

Bilateral Filter



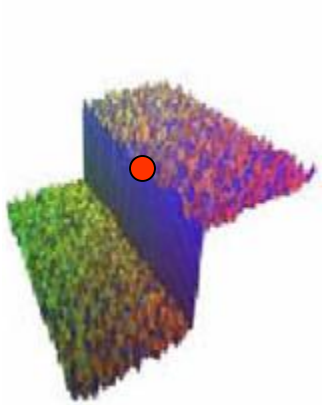
input

Bilateral Filter

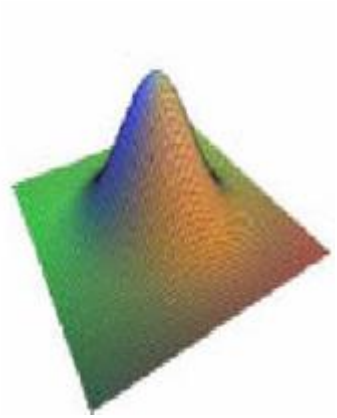


input

Bilateral Filter

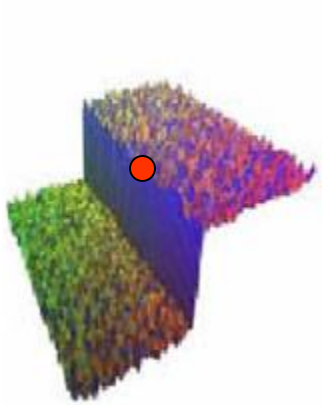


input

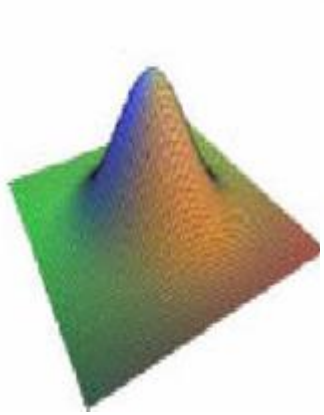


W_c

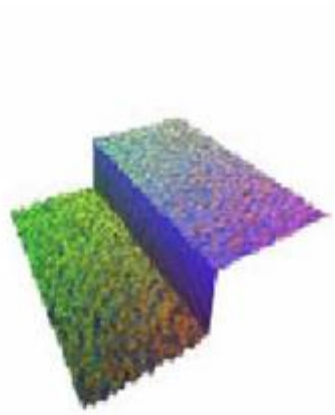
Bilateral Filter



input

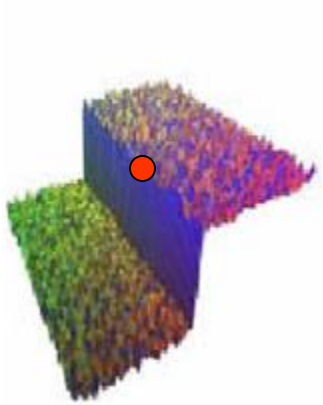


W_c

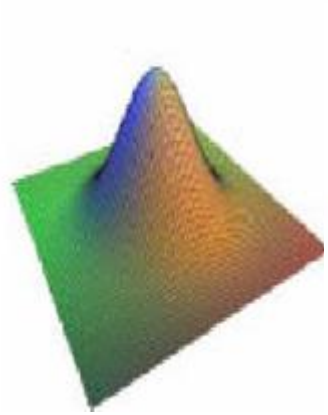


W_s

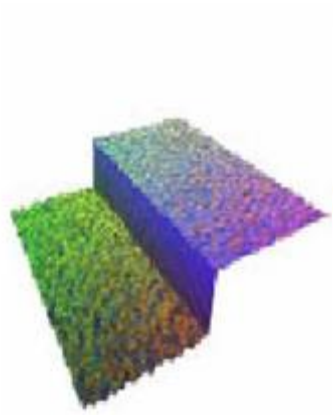
Bilateral Filter



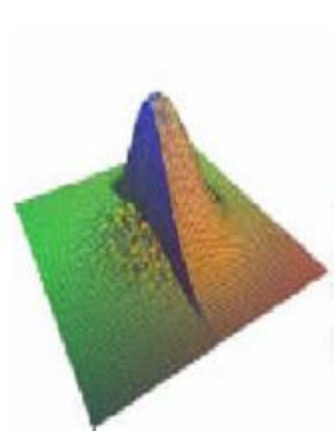
input



W_c

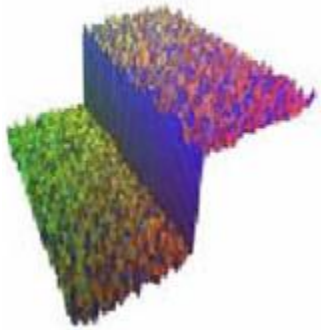


W_s

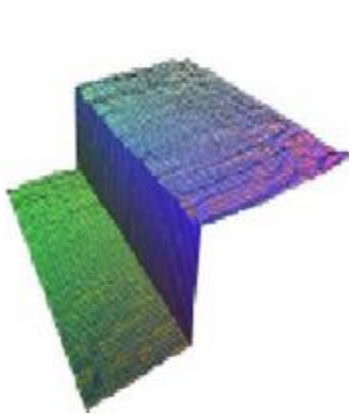


$W_s * W_c$

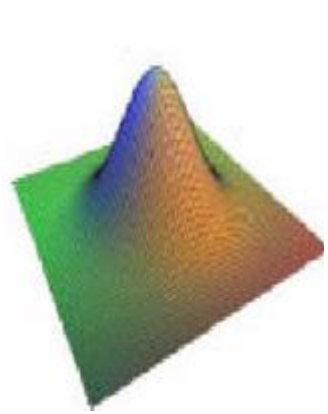
Bilateral Filter



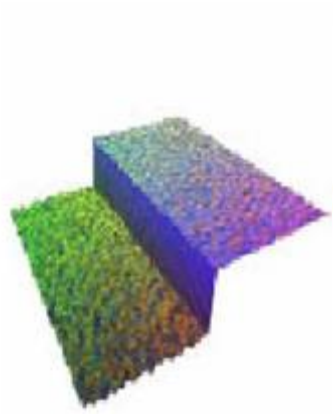
input



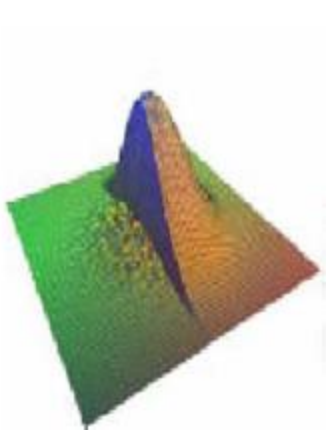
Output



W_c



W_s



$W_s * W_c$

Bilateral Filter Results

Original

Bilateral Filter Results

$\sigma_c = 3, \sigma_s = 3$

Bilateral Filter Results

$\sigma_c = 6, \sigma_s = 3$

Bilateral Filter Results

$\sigma_c = 12, \sigma_s = 3$

Bilateral Filter Results

$\sigma_c = 12, \sigma_s = 6$

Bilateral Filter Results

$\sigma_c = 15, \sigma_s = 8$

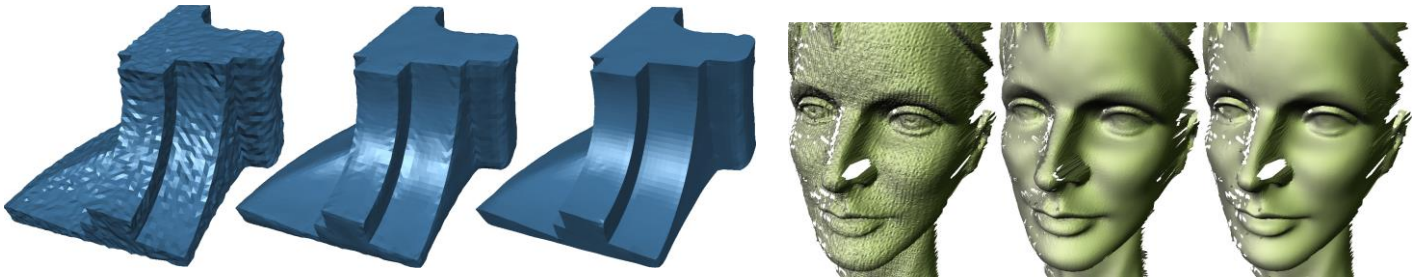
Additional Comments

The bilateral filter is a powerful filter:

- Can work with any reasonable distances function W_s and W_c,

- Easily extended to higher dimension signals, e.g. Images, video, mesh, animation data etc.

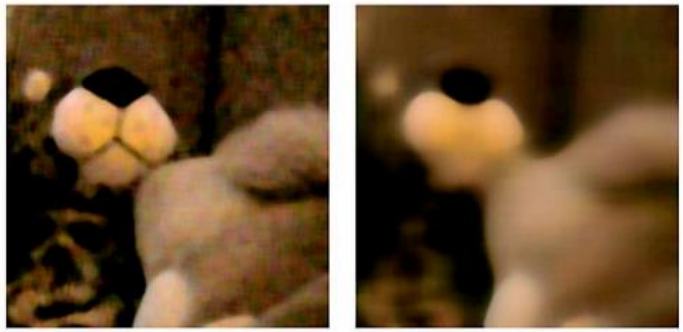
- Easily extended to vectored-signals, e.g. Color images, etc.



Bilateral Mesh Denoising
[Fleishman et al, siggraph 03]

Denoise

original image | 1px median filter



3px median filter | 10px median filter

Median Filter

Bilateral filter: remove noise, preserve edges

Machine Learning for Noise Reduction

To support noise reduction or obtain original image from its noisy image (observed image), we can have **two learning-based approaches**:

(a) learning the denoising function F:

$x\_{original} = F(y\_{observed})$; $F = argmin_F \|F(Y) - X\|_2^2$

(b) learning the noise image (residual image) n:

$x\_{original} = y\_{observed} - n$.

$$\hat{x} = \arg\min_x \frac{1}{2\sigma^2}\|y - x\|^2 + \lambda\Phi(x), \qquad (1)$$

Machine Learning for Joint Noise Reduction and De-Mosaicking

To solve the joint demosaicking-denoising problem, one of the most frequently used approaches in the literature relies on the following linear observation model

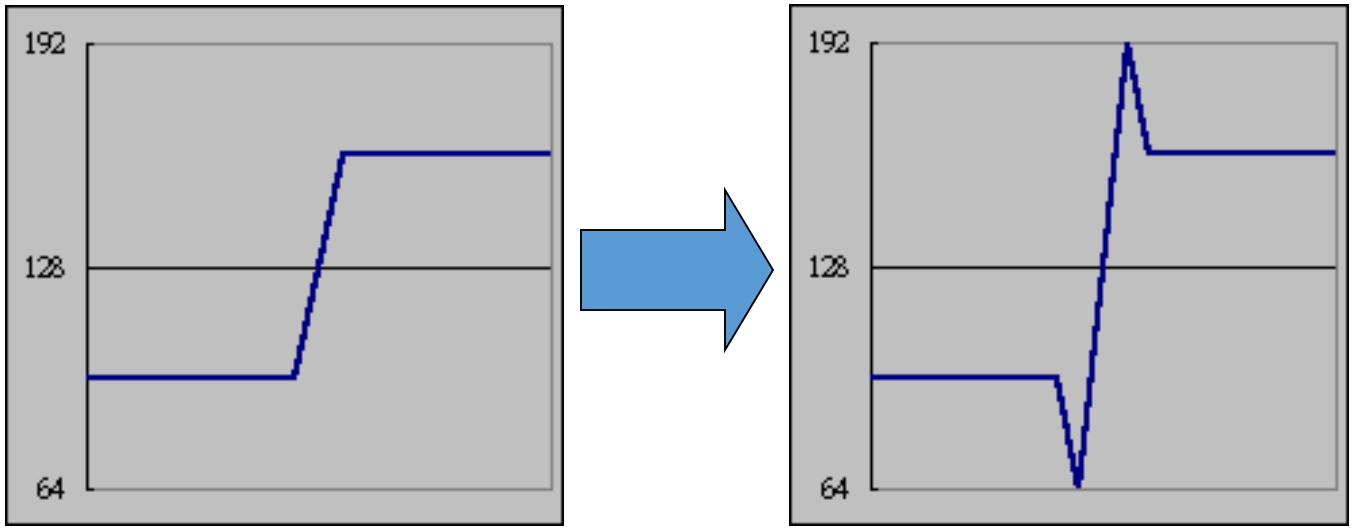
$$\mathbf{y} = \mathbf{M}\mathbf{x} + \mathbf{n}, \tag{1}$$

$$\mathbf{x}^\star = \arg\max_{\mathbf{x}} \log(p(\mathbf{x}|\mathbf{y})) = \arg\max_{\mathbf{x}} \log(p(\mathbf{y}|\mathbf{x})) + \log(p(\mathbf{x})), \tag{2}$$

$$\mathbf{x}^\star = \arg\min_{\mathbf{x}} \frac{1}{2\sigma^2} \|\mathbf{y} - \mathbf{M}\mathbf{x}\|_2^2 + \phi(\mathbf{x}) \tag{3}$$

Edge Enhancement

- A must - all cameras add edges
- General approaches
  - Edge filter: NxN, 1xN+Nx1
  - Edge gain control
  - Edge detection module
- Noise should not be enhanced



Edge Enhancement

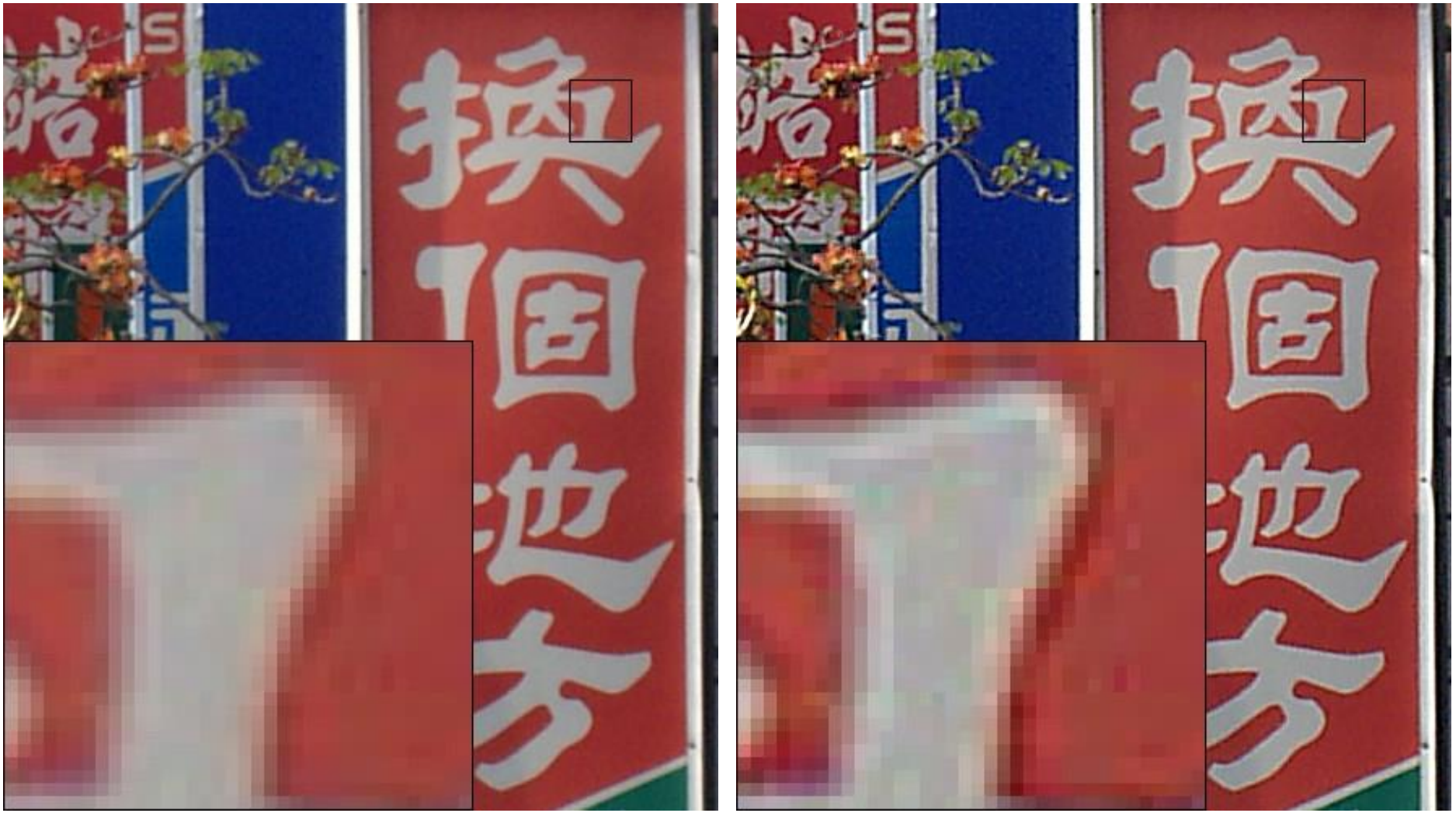
Unlike some forms of image sharpening, **edge enhancement** does not enhance subtle detail which may appear in more uniform areas of the image, such as texture or grain which appears in flat or smooth areas of the image. The benefit to this is that imperfections in the image reproduction, such as grain or noise, or imperfections in the subject, such as natural imperfections on a person's skin, are not made more obvious by the process.
A drawback to this is that the image may begin to look less natural, because the apparent sharpness of the overall image has increased but the level of detail in flat, smooth areas has not.

Edge Enhancement

As with other forms of image sharpening, edge enhancement is only capable of improving the *perceived* sharpness or acutance of an image. The enhancement is not completely reversible, and as such some detail in the image is lost as a result of filtering. Further sharpening operations on the resulting image compound the loss of detail, leading to artifacts such as ringing. An example of this can be seen when an image that has already had edge enhancement applied, such as the picture on a DVD video, has further edge enhancement applied by the DVD player it is played on, and possibly also by the television it is displayed on. Essentially, the first edge enhancement filter creates new edges on either side of the existing edges, which are then further enhanced.

Edge Enhancement

- Normal and strong edge enhancement



Resizing and Cropping

Resizing and Cropping

- Preview display
  - Sensor lines number doesn't match with LCD.
- Video capture
  - Sensor lines number doesn't match with output.
- Digital zoom
  - Crop smaller area centered at original center
- Raw data space or YUV space
- Performance vs. quality
  - Dropping or duplication, bilinear interpolation, bicubic interpolation

AE / AF / AWB (I)

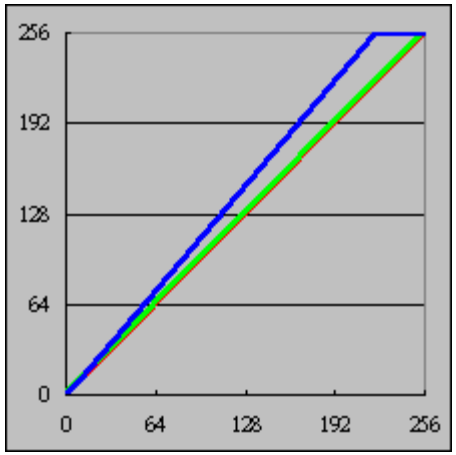
- Closed loop controls for passive 3A (active 3A are totally different)
  - Data collecting + control mechanism + converging algorithm
- AE (auto exposure)
  - Collect luminance before gamma correction
  - Control exposure time, analog gain, iris,...
  - Converge luminance to AE target

AE / AF / AWB (II)

- **AWB (auto white balance)**
  - Collect color statistic after white balancing
  - Control color gain
  - Converge color average to white target
- AF (auto focus)
  - Collect focus value before edge enhancement
  - Control image plane position via AF motor
  - Find position with maximum focus value

More Processing (I)

- To avoid false color
  - Color clamping
  - False color suppression
  - Color noise reduction
- Lens shading correction
- Lens distortion correction



More Processing (II)

- Skin-tone detection

More Processing (III)

- Digital flash (local contrast normalization)
  - Improve the contrast on dark area like a flash light
  - http://www.ukapical.com/products_DSC_inter.htm

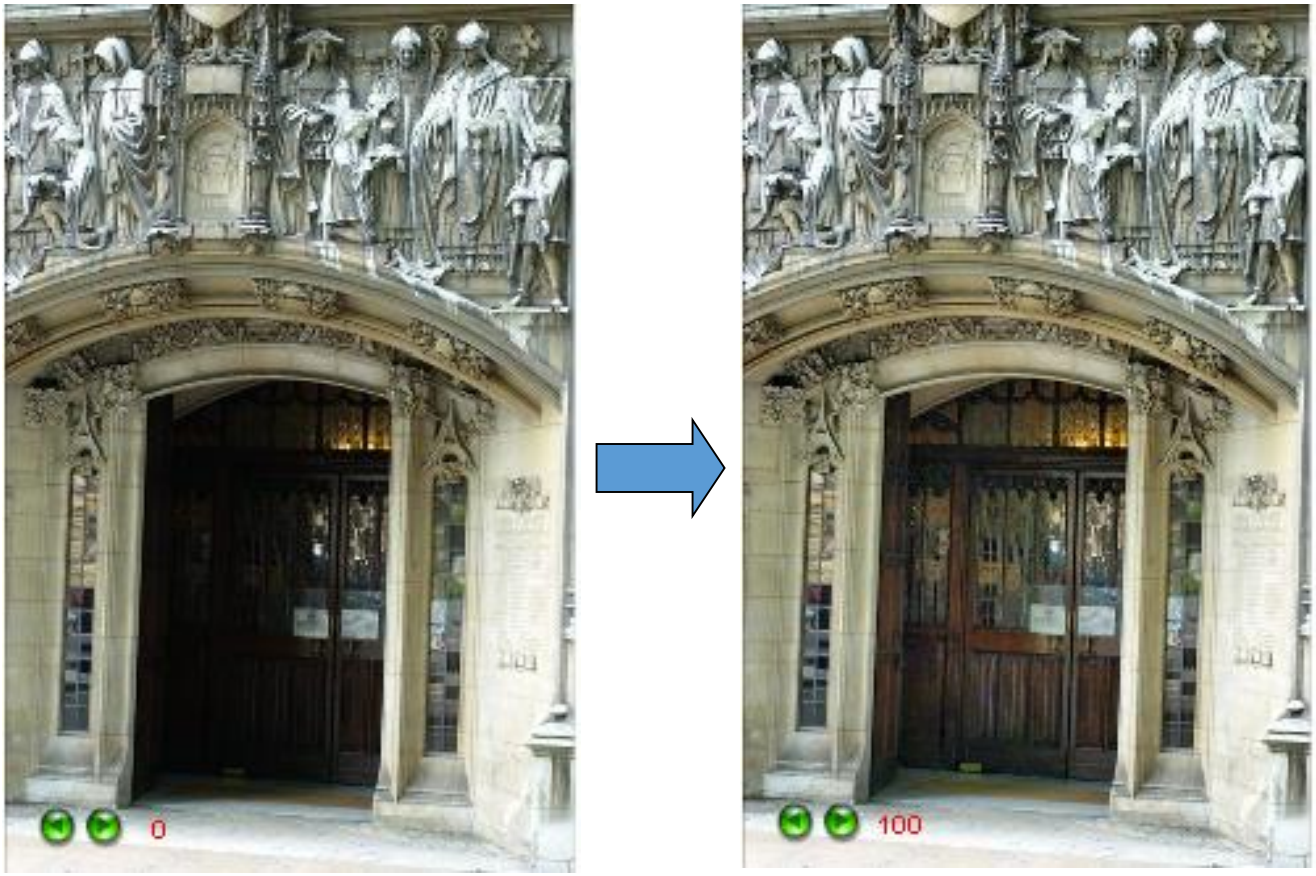


Image Details Degradation

- Image process harms image details
  - Increasing noise
  - Accumulating rounding error
  - Overflow and underflow
- Less processing is better

Image Pipeline Noise (8-bit, δ=0)

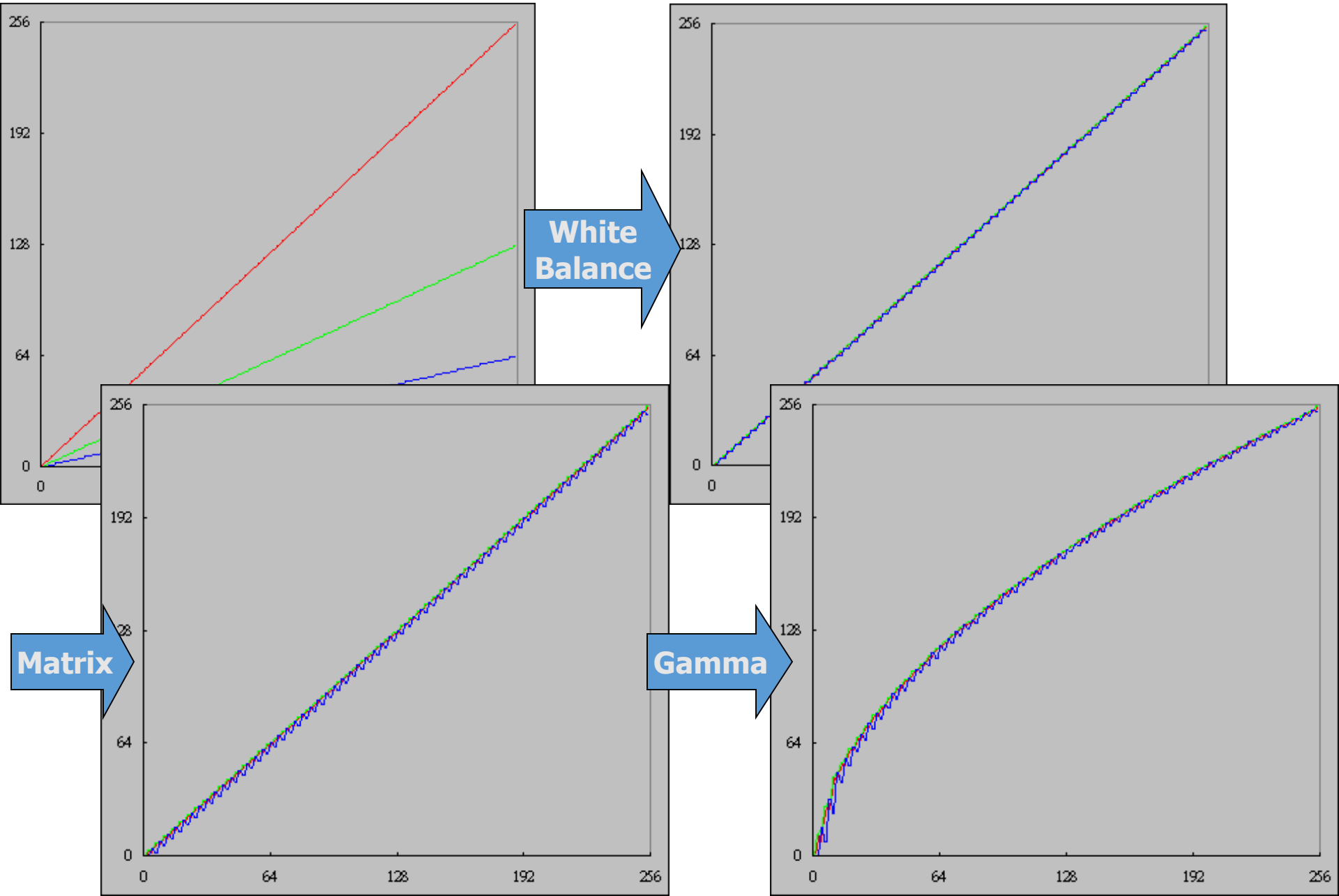


Image Pipeline Noise (8-bit, δ=2)

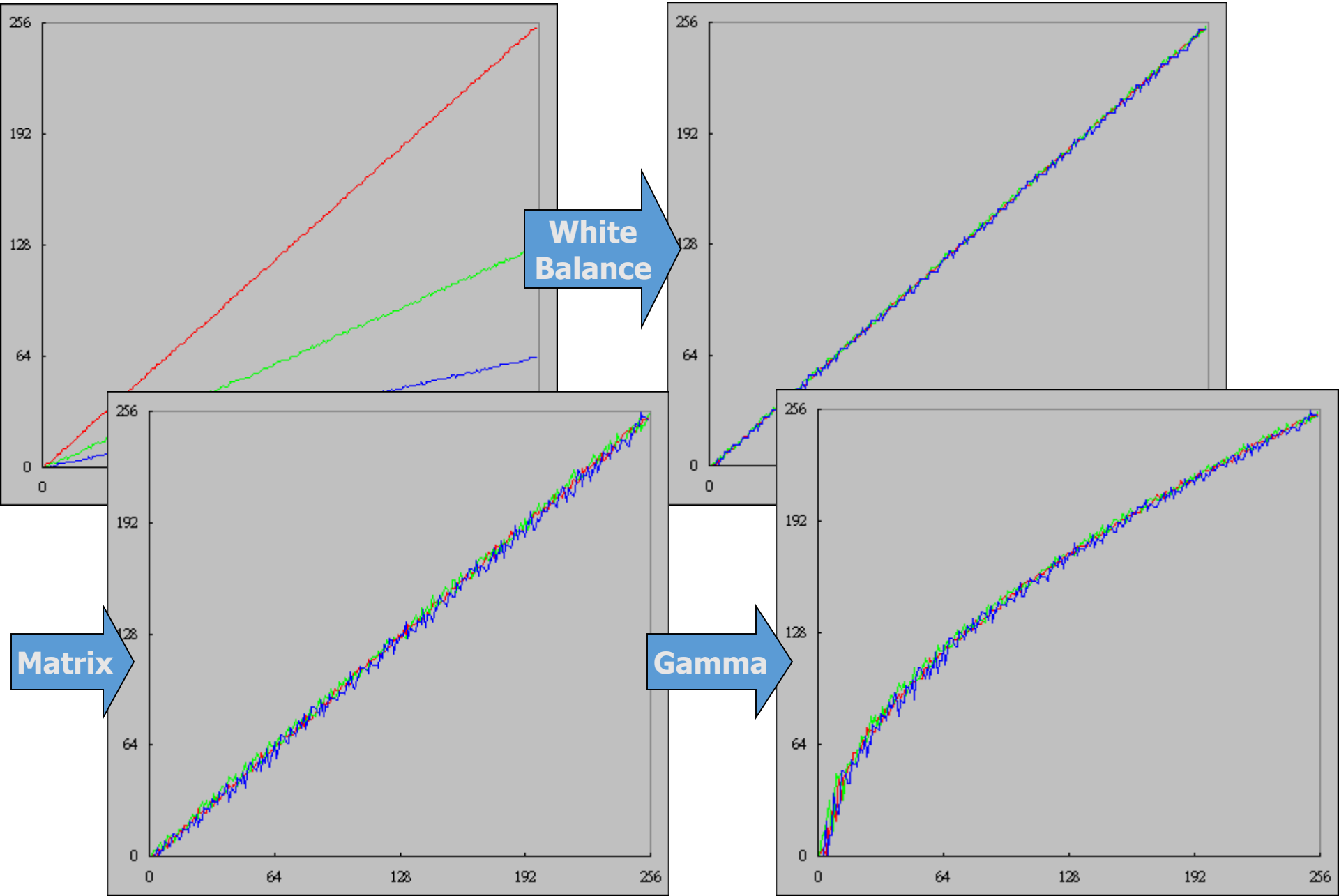
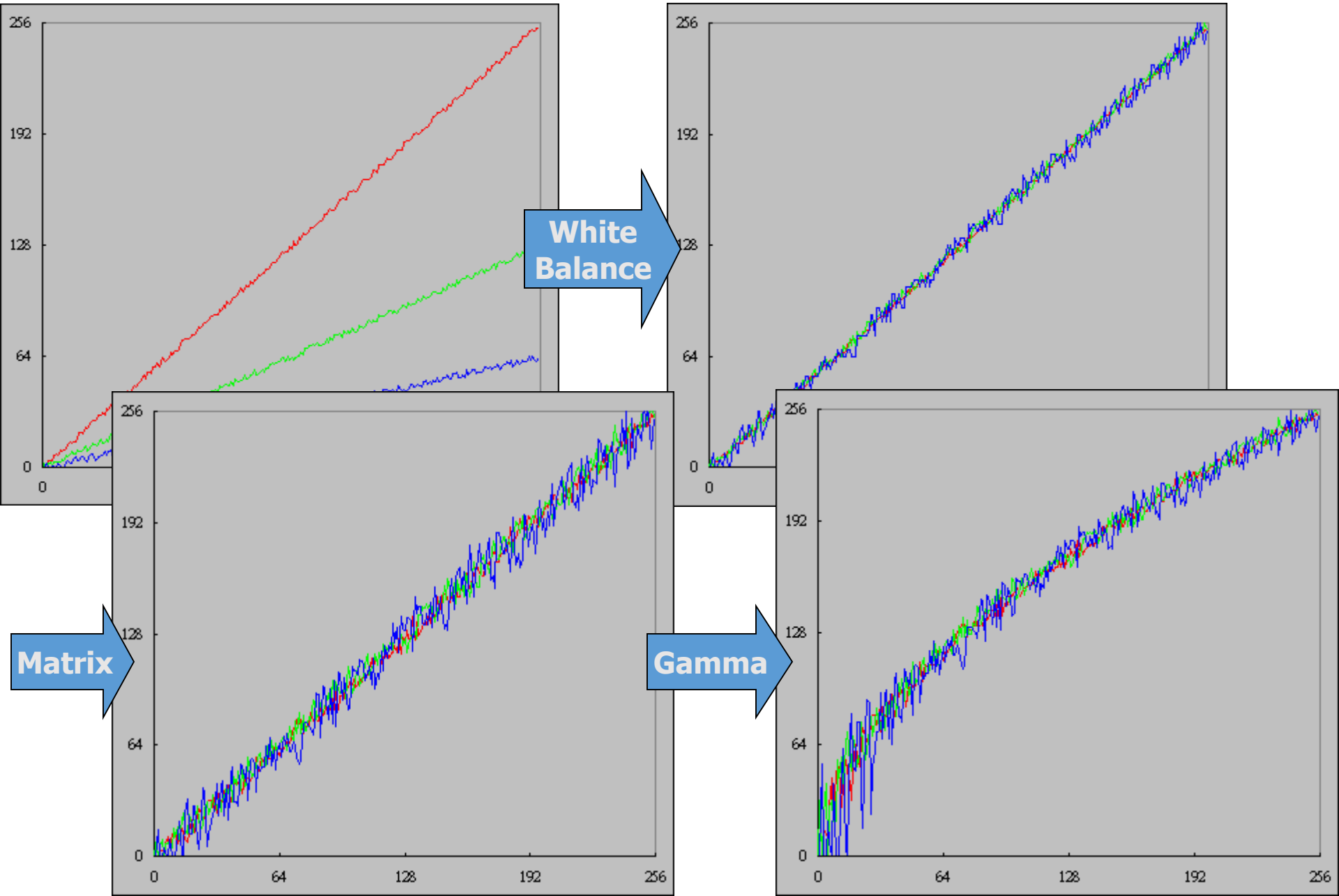


Image Pipeline Noise (8-bit, δ=4)



Performance & Resource Consideration

- Memory buffer size
- DRAM bandwidth
- Computing complexity
- Pipelining: overlapping operation

Image Pipeline Fine-tuning

- From beginning of pipeline
- Step by step
- Back and forth

Zoran COACH 6

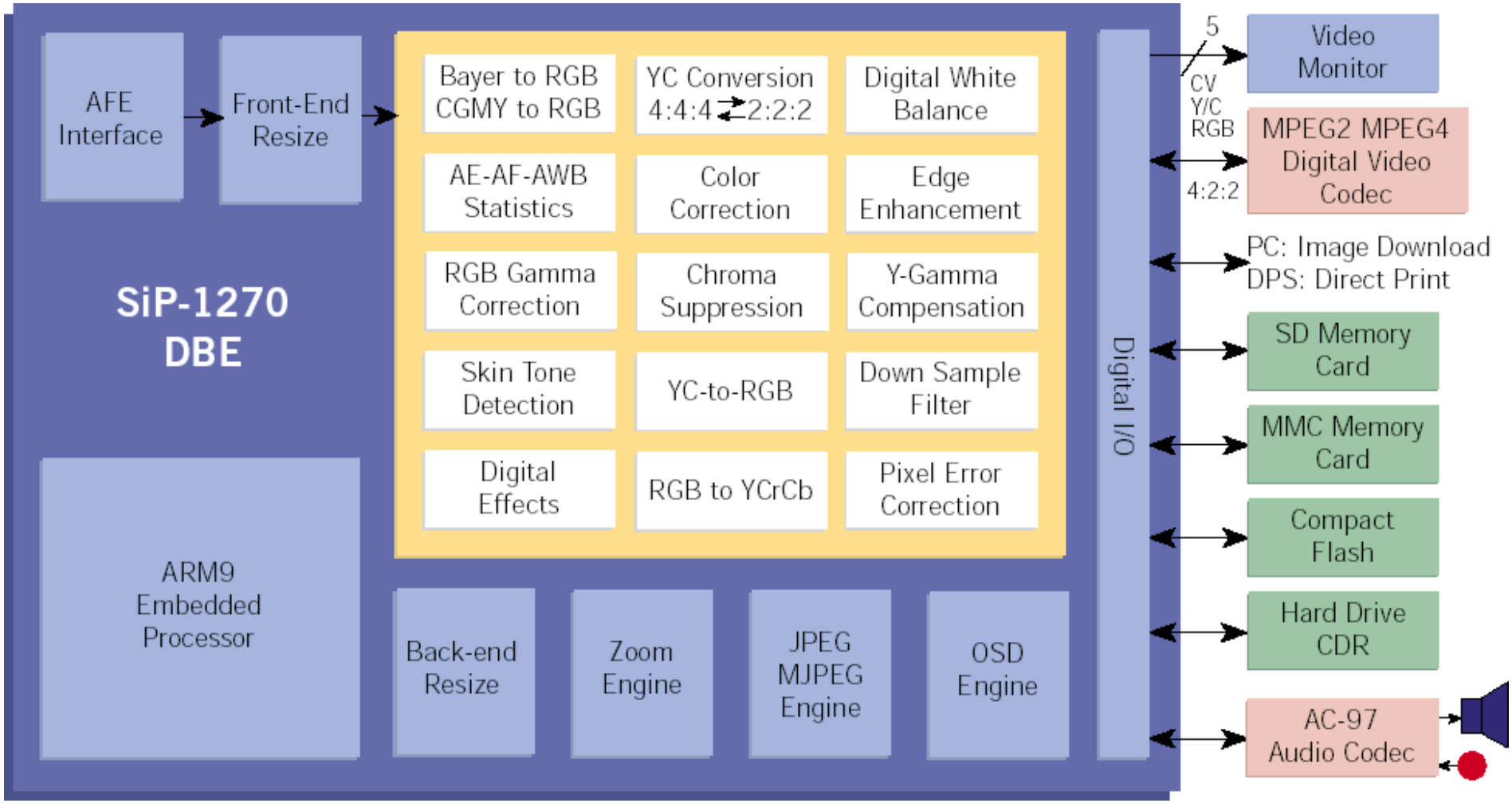
Features

- Direct glueless interface to multiple-field & progressive CCD and CMOS sensors
- Supports image resolutions up to 3 Megapixels for COACH 6e and up to 16 Megapixels for COACH 6p
- Enhanced CCD signal processing with
  - 12-bit processing for high dynamic range
  - Color interpolation and conversion
  - HW based Image scalers (zero penalty on click-to-click times)
  - Automatic White Balance (AWB)
  - Automatic Exposure (AE)
  - Edge enhancement
  - Programmable Lens Shading (Patent Pending) compensation
  - Performance enhancement when using Zoran's CMOS sensor

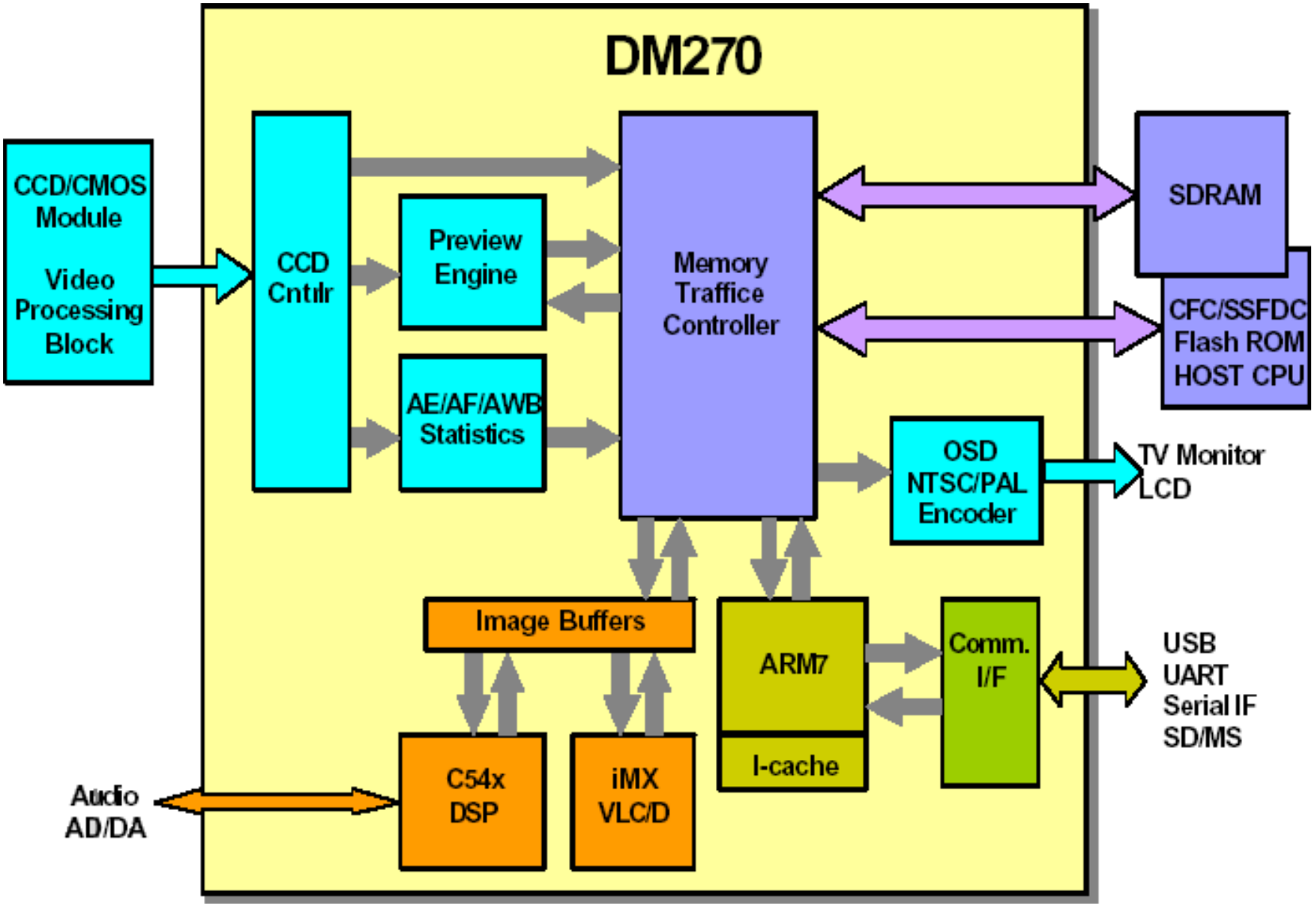
- Programmability for image processing algorithms
- Smooth Digital zoom – complete emulation of optical zoom (view, capture and clip recording)
- Real-time JPEG compression with advanced bit rate control
- Capture of a sequence of still images
- Movie (AVI, with compressed audio) capture and playback to the Flash card, at a rate of up to 30 frames per second (CIF & VGA)

NuCORE Sip-1270

- Work with their AFE NDX-1260



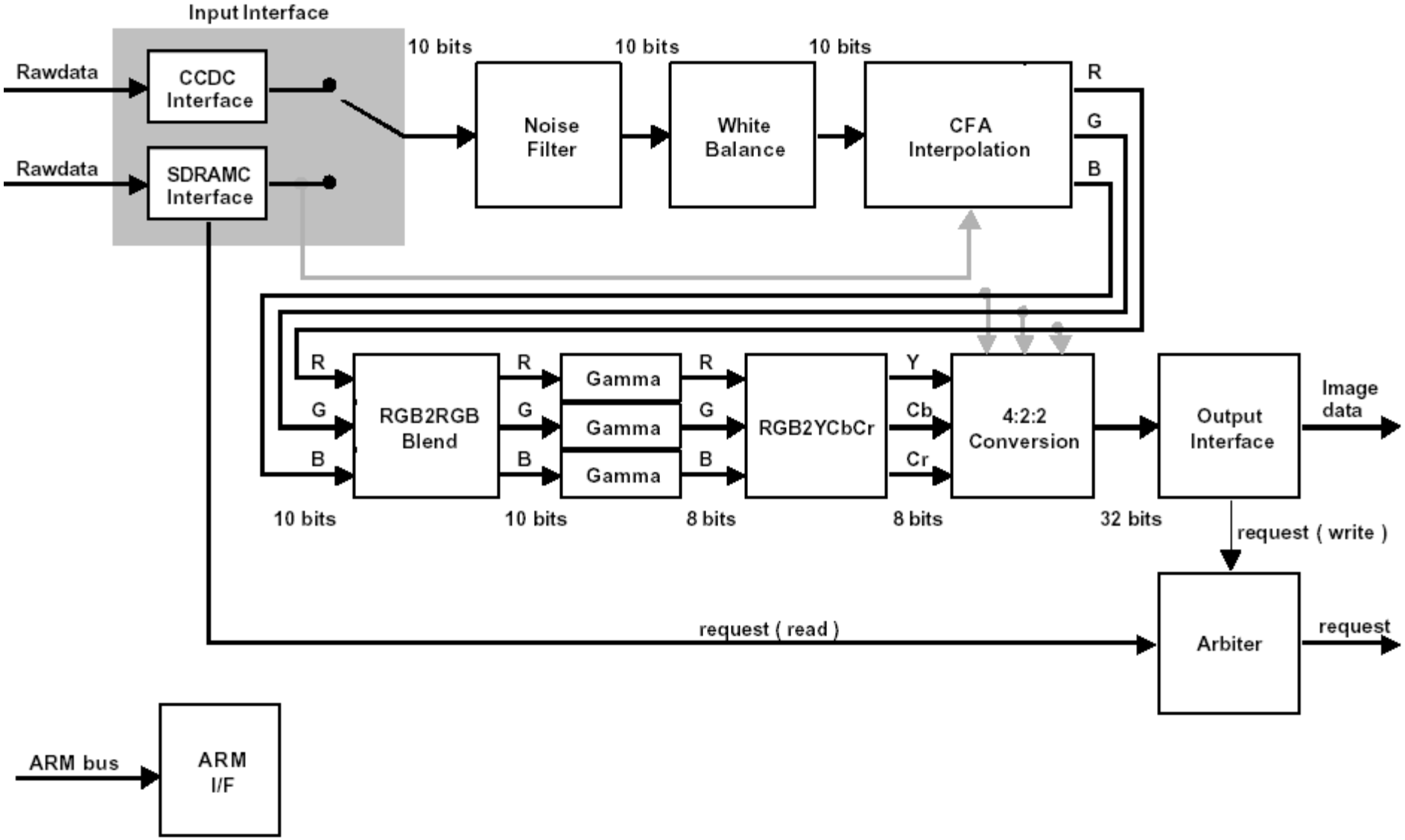
TI TMS320DM270



DM270 CCD Controller

- Digital clamping
- Black level compensation
- Median filter
- Gain and offset
- Output formatter

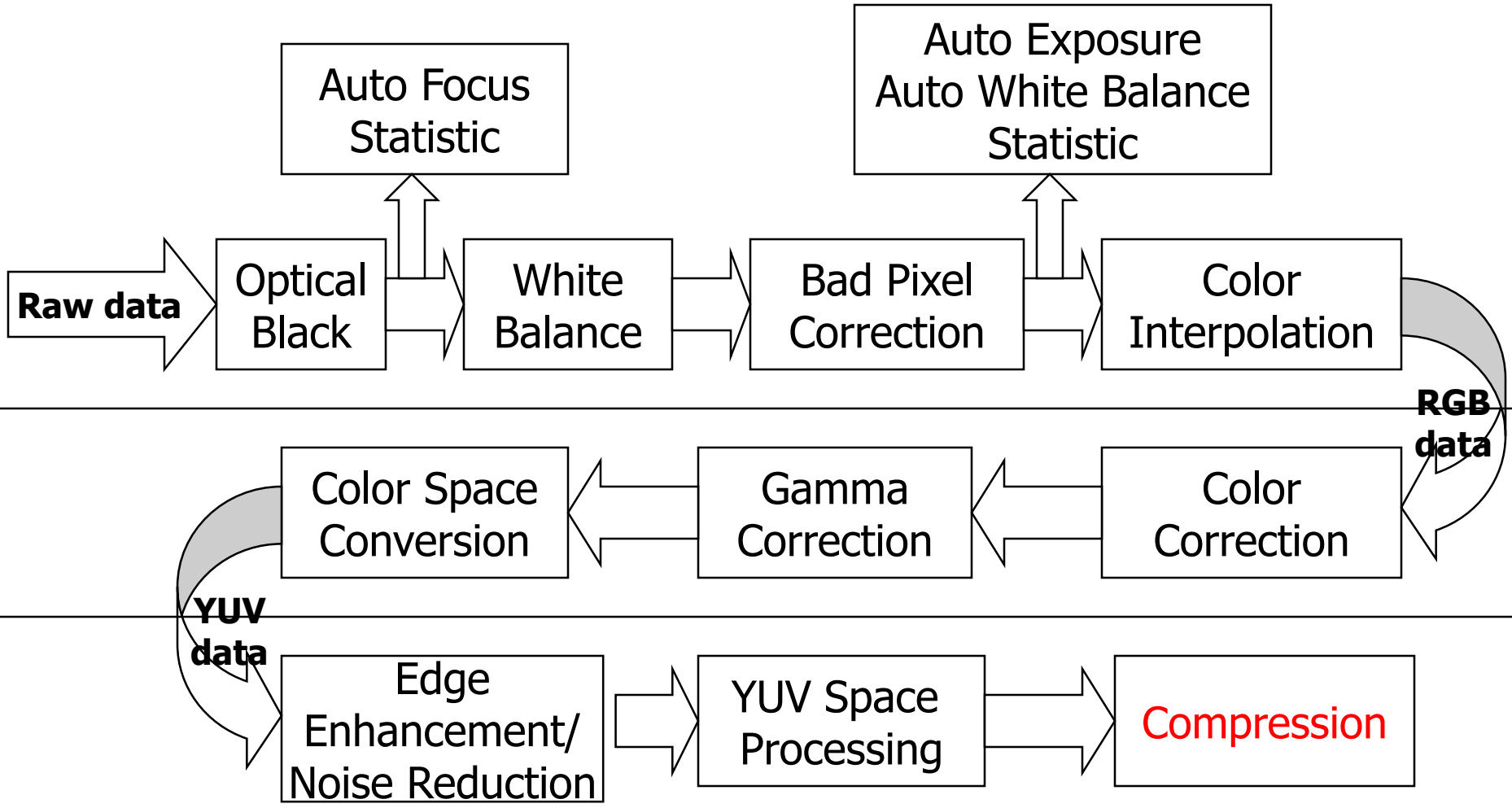
DM270 Preview Engine



DM270 AF/AE/AWB Statistics Engine

- AE / AWB
  - Up to 192 windows
  - Configurable rows and columns number
  - Configurable window size and position
  - Output R, Gr, Gb, B accumulation
- AF
  - Up to 36 windows
  - Configurable window size and position
  - Output G accumulation and focus value (result of two filters for green pixels)

A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

Image Compression

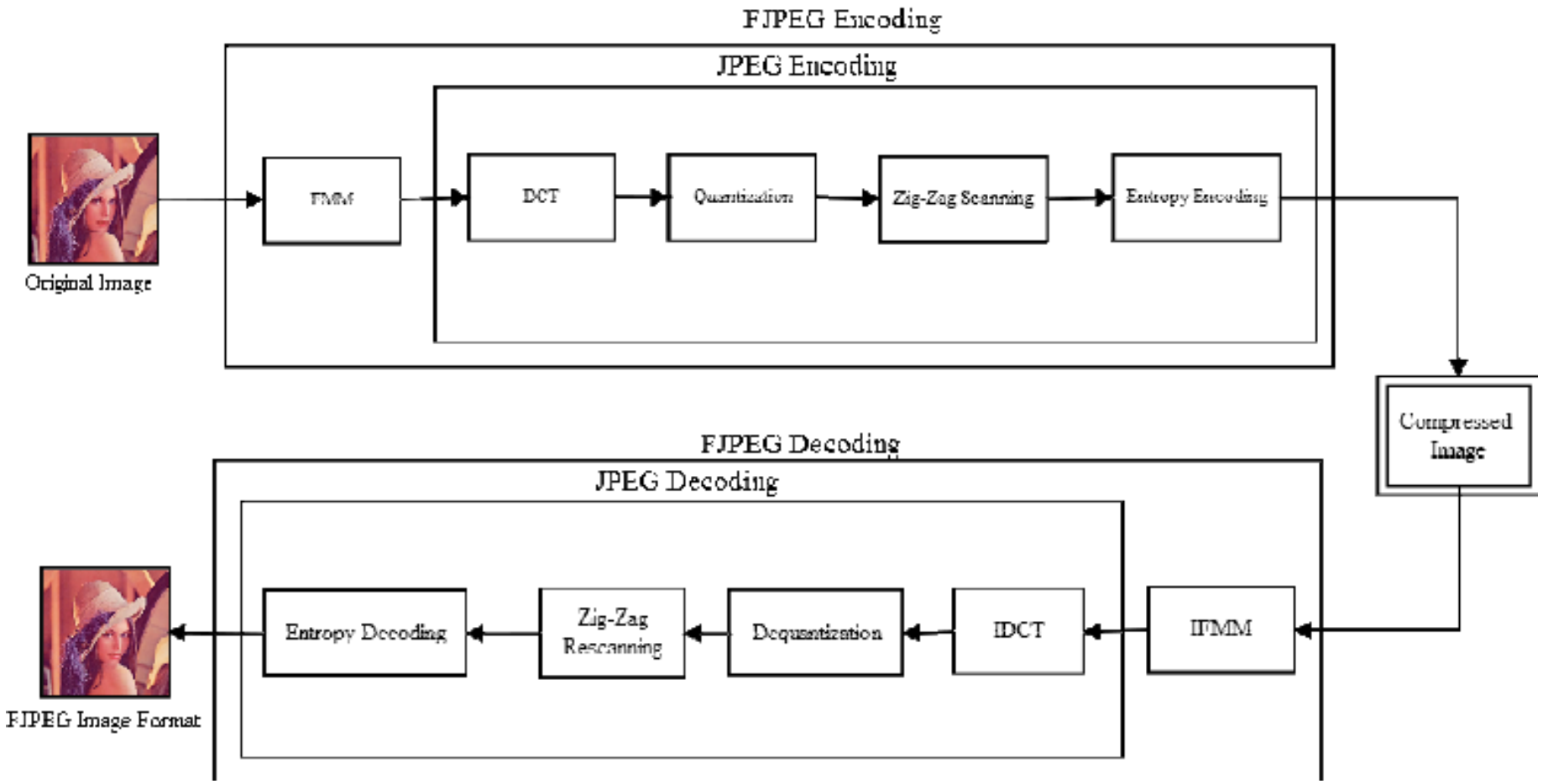
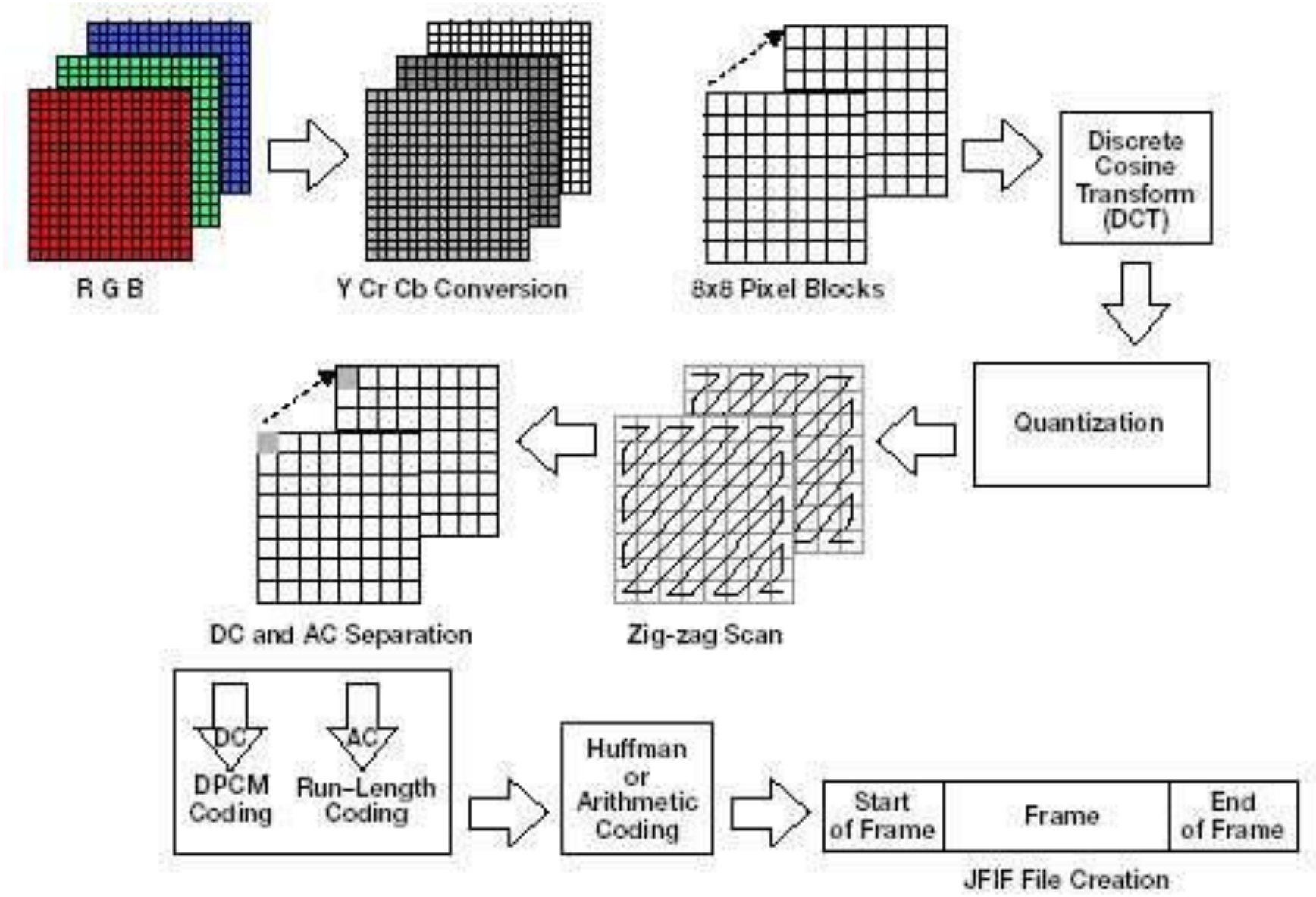


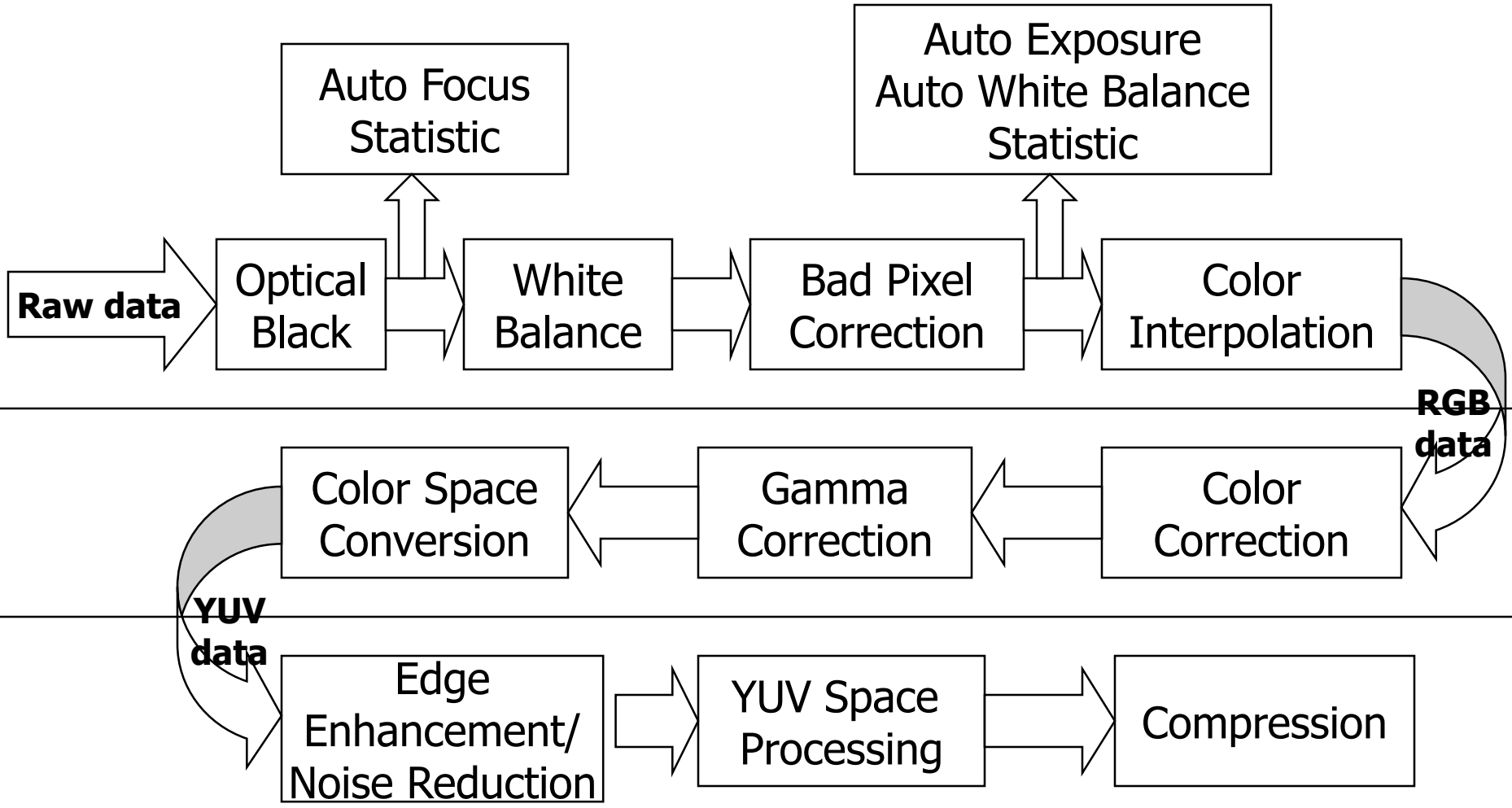
Image Compression



Conclusions

- **Error Propagation**: ISP contains a set of sequential operations (outputs of previous operator are the inputs of current operator), thus the mistakes for previous steps will translate into next step and such mistakes may be enlarged step-by-step;
- **Huge Potential Combinations**: All these operators are optimized in a step-by-step and back-and-forth way, thus when one parameter or one operator is changed, we have to fine tune all the relevant parameters, which is very time-consuming.

A Typical Image Pipeline for Digital Camera



ISP targets on matching human perception

www.csie.ntu.edu.tw › IntroductionToDigitalCameraSignalProcessor

32 parameters & their combinations

Demoisaicking

Parameter	Value	Max
vh slope	190	2^8
vh thresh	220	2^{12}
va slope	175	2^8
va thresh	210	2^{12}
aa slope	170	2^8
aa thresh	100	2^{12}
uu slope	165	2^8
uu thresh	210	2^{12}
sharp alt ld	45	2^8
sharp alt ldu	45	2^8
sharp alt lu	25	2^8
fc alias slope	85	2^8
fc alias thresh	0	2^8
fc slope	130	2^8
np offset	3	2^8

Colorspace Conversion

Parameter	Value	Max
coef a 11	4461	5880
coef a 12	4001	5880
coef a 21	4068	5880
coef a 22	4398	5880
coef a 31	4135	5880
coef a 32	3842	5880

Denoising

Parameter	Value	Max
thresh 1h	5	2^8
strength 1	190	2^8
thresh 4h	10	2^8
strength 4	255	2^8
thresh long	48	2^8

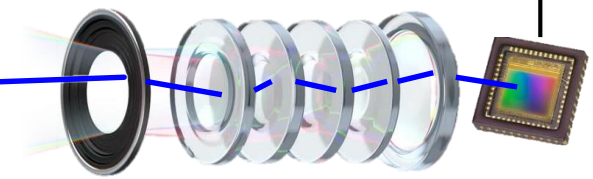
Color & Tone Correction

Parameter	Value	Max
lut knee	207	2^8
lut power	129	2^8
lut shadow	31	2^8

White Balance

Parameter	Value	Max
gain 00	583	2^{12}
gain 01	271	2^{12}
gain 11	587	2^{12}

Current Industry Approach



- **High-dimensional** parameter space and **mixed** between continuous and discrete
- ISP is often **non-differentiable** and **black box**
- Want to tune parameters towards almost **any task metric**

Parameters ?

Hand-Tuning

"Golden Eye"

Few months of dirty works