Post-processing pipeline

CS 178, Spring 2010

Begun 5/27/10. Finished 6/1/10, and recap slides added throughout.

Marc Levoy
Computer Science Department
Stanford University

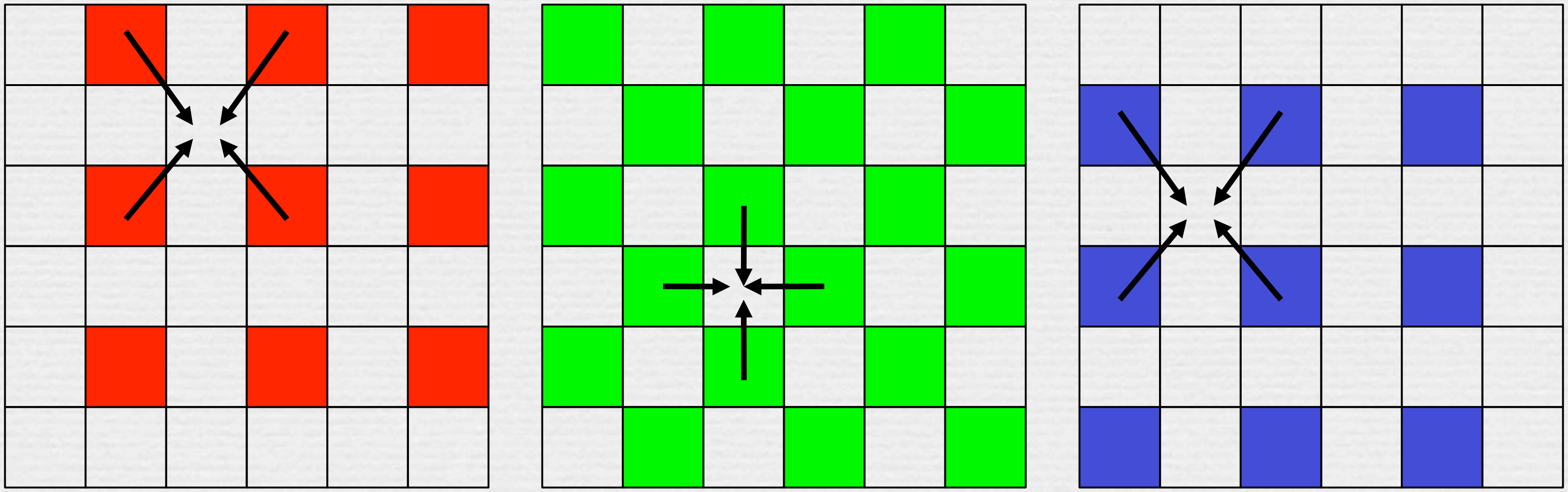
Camera pixel pipeline

sensor → analog to digital conversion (ADC) → processing: demosaicing, tone mapping & white balancing, denoising & sharpening, compression → storage

- every camera uses different algorithms
- the processing order may vary
- most of it is proprietary

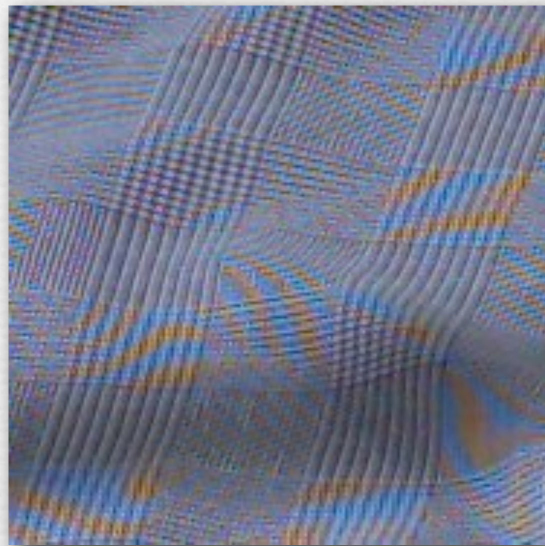
Demosaicing (review)

- linear interpolation
  - average of the 4 nearest neighbors of the same color
- cameras typically use more complicated scheme
  - try to avoid interpolating across feature boundaries
  - demosaicing is often combined with denoising, sharpening...



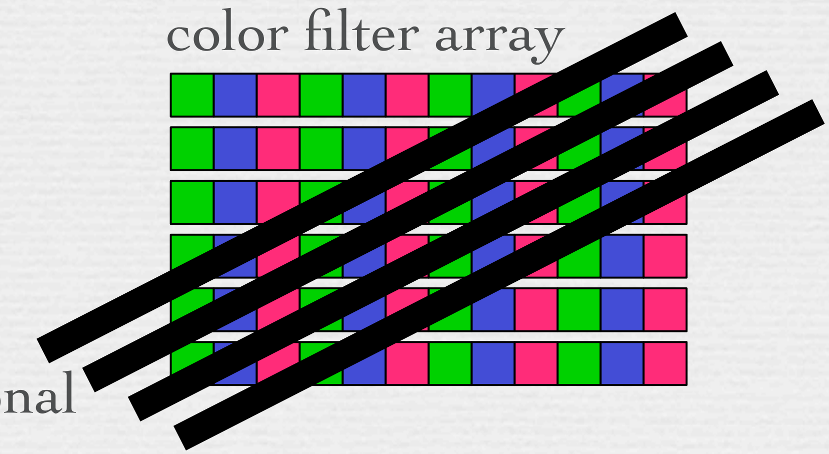
Demosaicing errors

- color fringes or moiré



- the cause of color moiré
  - fine black and white detail in scene is mis-interpreted by interpolation algorithm as color information

simplistic 1D color filter array



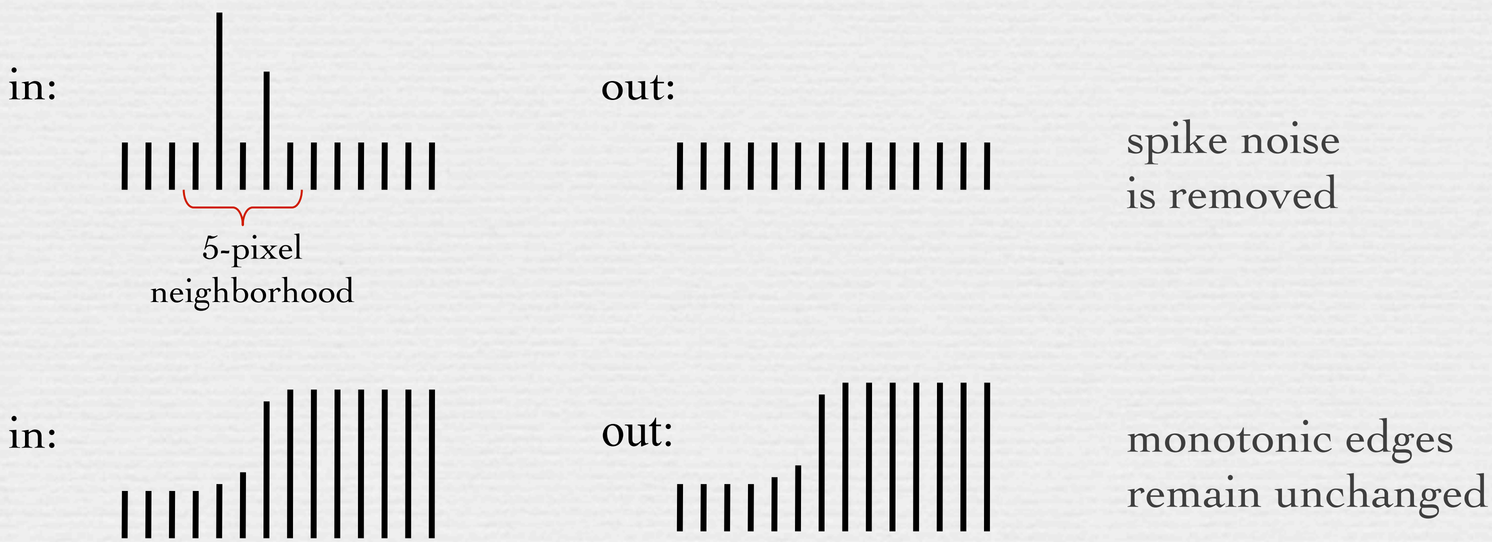
fine diagonal B&W stripes

Common solution: low-pass filter chrominance signal

- color artifacts are places where chrominance changes abruptly (due to interpolation errors) but only transiently

Common solution: low-pass filter chrominance signal

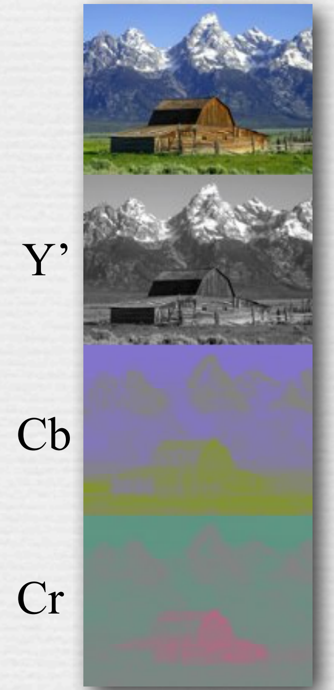
- color artifacts are places where chrominance changes abruptly (due to interpolation errors) but only transiently
- use a median filter on chrominance to remove outlier transient chrominance changes [Freeman 1988]
  - replace the chrominance of each pixel by the median value in a neighborhood
  - this is a non-linear filter



Common solution: low-pass filter chrominance signal

- color artifacts are places where chrominance changes abruptly (due to interpolation errors) but only transiently
- use a median filter on chrominance to remove outlier transient chrominance changes [Freeman 1988]
  - replace the chrominance of each pixel by the median value in a neighborhood
  - this is a non-linear filter
- summary of algorithm
  - 1. apply naive interpolation
  - 2. convert to Y'CbCr
  - 3. median filter Cr & Cb
  - 4. reconstruct R, G, B from sensor value and filtered Cr & Cb

(wikipedia)



Comparison

(crop from larger image)

linear interpolation

median-filtered interpolation

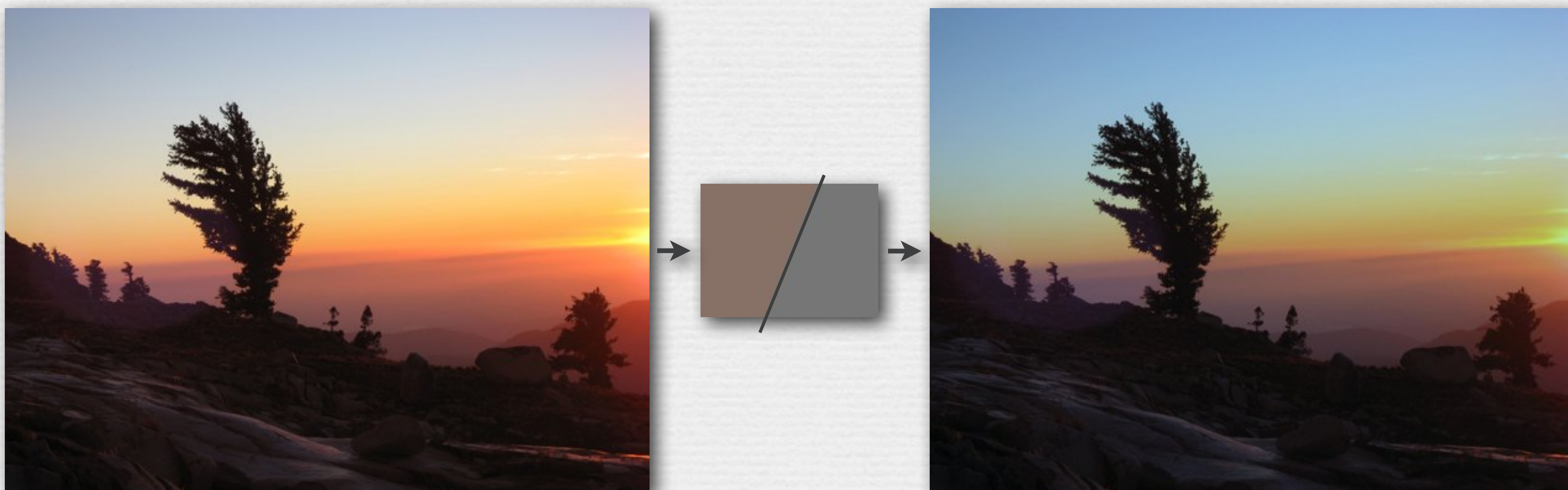
- take-home lesson: 2/3 of your data is made up
- there are better and worse ways to do this

Camera pixel pipeline

sensor → analog to digital conversion (ADC) → processing: demosaicing, tone mapping & white balancing, denoising & sharpening, compression → storage

White balancing (review)

- 1. find the color temperature of the illumination as an (R,G,B)
- 2. scale the RGB values of all pixels in the photograph up or down so that the chosen (R,G,B) becomes (1,1,1)
- the appearance of (1,1,1) depends on the camera's reference white
    - for sRGB cameras, chosen (R,G,B) will match D65 (6500°K)



(Marc Levoy)

Gamma and gamma correction

(FLASH DEMO)

http://graphics.stanford.edu/courses/cs178/applets/gamma.html

- the goal of digital imaging is to accurately reproduce relative scene luminances on a display screen
  - absolute luminance is impossible to reproduce
  - humans are sensitive to relative luminance anyway
- in some workflows, pixel value is made proportional to scene luminance, in other systems to perceived brightness
  - in CRTs luminance is proportional to voltage$^{\gamma}$ with $\gamma \approx 2.5$, so TV cameras must be designed to output scene luminance$^{1/\gamma}$
  - in NTSC cameras have $1/\gamma = 0.5$ to provide a residual *system gamma*
  - pixel value ∝ perceived brightness is perceptually uniform, so in CG and digital photography it's a good space for quantization, JPEG, etc.

System gamma

- why the gamma of cameras (0.5) in NTSC is higher than the inverse of the gamma of CRTs (2.5)

<u>viewer outdoors looking at a scene</u>

$B \propto L_{scene}^{0.5}$

perceived brightness

luminance

bright-adapted eye

<u>viewer in dark living room watching outdoor scene on TV</u>

$B \propto L_{display}^{0.4}$

$\propto L_{scene}^{0.5 \times 2.5 \times 0.4}$

$\propto L_{scene}^{0.5}$

dark-adapted eye

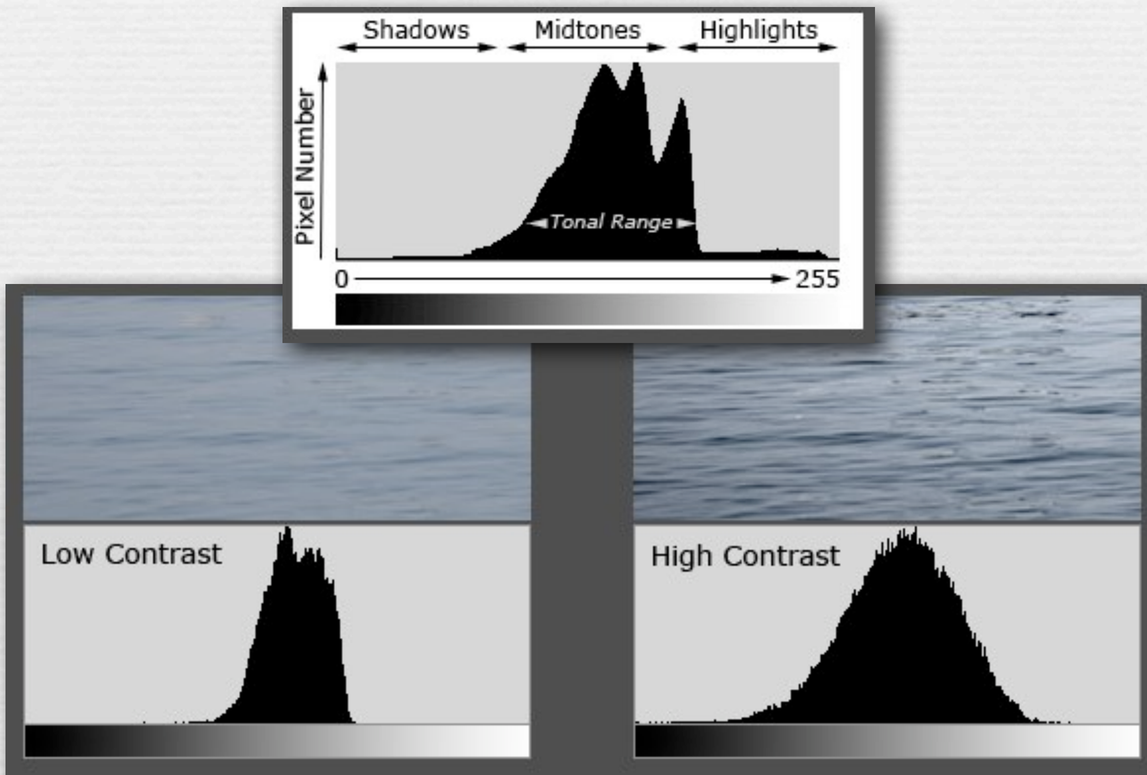
camera

CRT

- camera γ × display γ = 0.5 × 2.5 = 1.25 is the *system gamma*

Contrast correction (a.k.a. tone mapping)

- manual editing
  - capture image in RAW mode, then fiddle with histogram in Photoshop, `dcraw`, Canon Digital Photo Professional, etc.



(cambridgeincoulour.com)

Contrast correction (a.k.a. tone mapping)

- manual editing
  - capture image in RAW mode, then fiddle with histogram in Photoshop, `dcraw`, Canon Digital Photo Professional, etc.
- gamma transform (in addition to RAW→JPEG gamma)
  - output = input$^{\gamma}$ (for $0 \le I_i \le 1$)
  - simple but crude

original

$\gamma = 0.5$

$\gamma = 2.0$

Contrast correction (a.k.a. tone mapping)

- manual editing
  - capture image in RAW mode, then fiddle with histogram in Photoshop, `dcraw`, Canon Digital Photo Professional, etc.
- gamma transform (in addition to RAW→JPEG gamma)
  - output = input$^{\gamma}$ (for $0 \leq I_i \leq 1$)
  - simple but crude
- histogram equalization

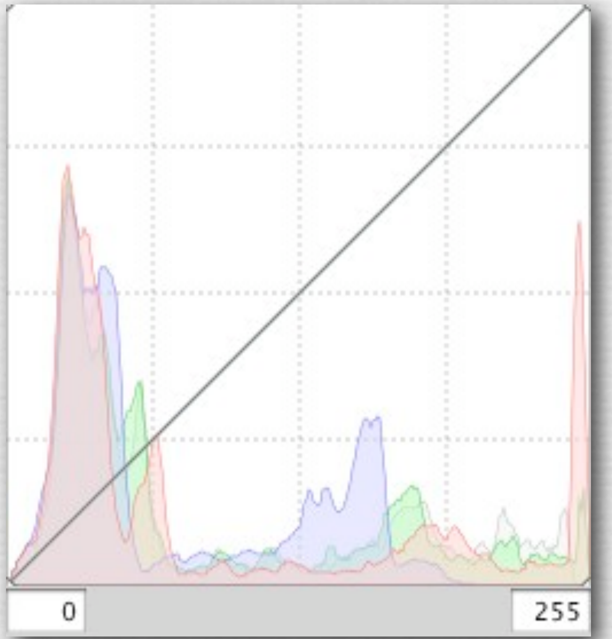
Histogram equalization

1. convert image to L*a*b* in range [0,1]

2. calculate histogram of L* channel $pdf(i) = \frac{N_i}{N}$,
where N_i is the number of pixels of intensity i,
and N is the total number of pixels

image

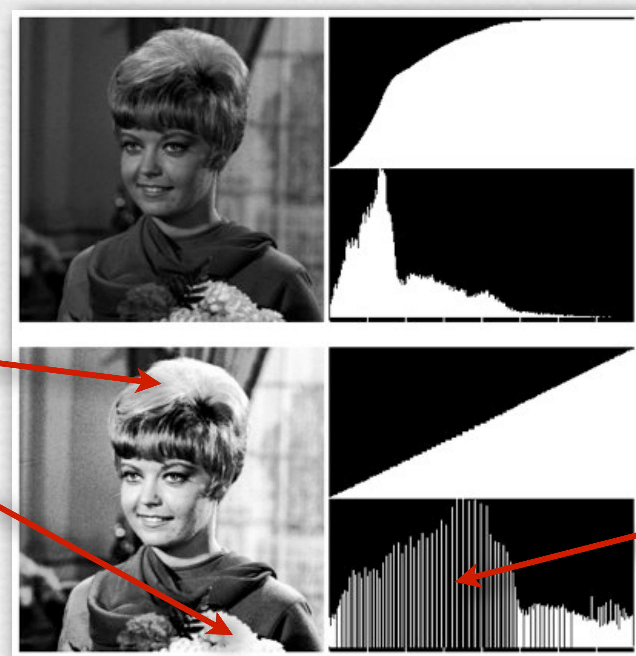
relative frequency



histogram

Histogram equalization

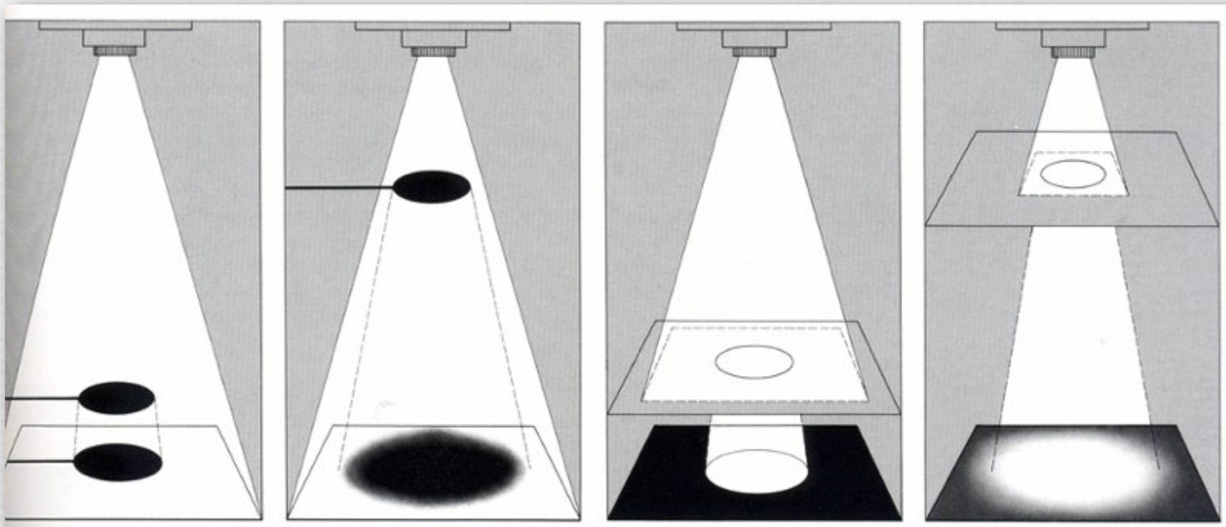
1. convert image to L*a*b* in range [0,1]
2. calculate histogram of L* channel $pdf(i) = \frac{N_i}{N}$,
where N_i is the number of pixels of intensity i,
and N is the total number of pixels
3. calculate cumulative density function $cdf(i) = \sum_{j=0}^{i} pdf(j)$
4. re-map each pixel using $I_{out} = cdf(I_{in}) \times 255 / N$ (for 8-bit pixels)



Contrast correction (a.k.a. tone mapping)

- manual editing
  - capture image in RAW mode, then fiddle with histogram in Photoshop, `dcraw`, Canon Digital Photo Professional, etc.
- gamma transform (in addition to RAW→JPEG gamma)
  - output = input$^{\gamma}$ (for $0 \leq I_i \leq 1$)
  - simple but crude
- histogram equalization
- global versus local transformations

Traditional dodging and burning

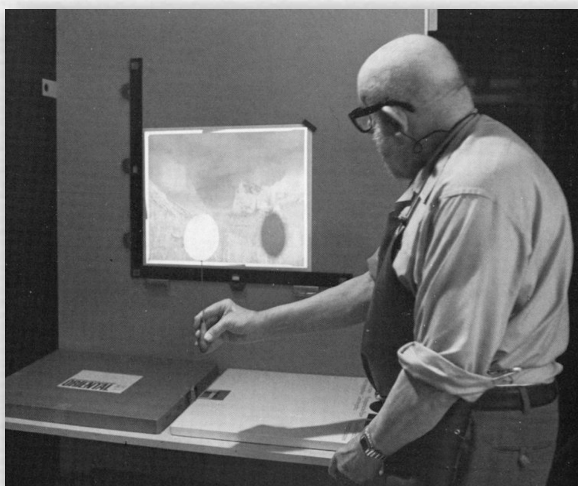


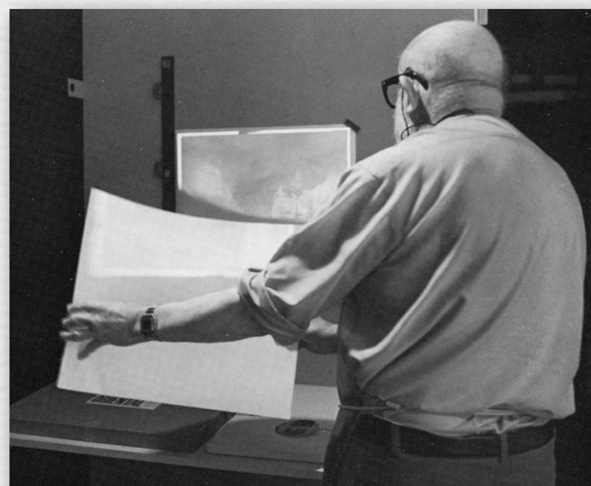
(Rudman)

dodging
(leaves print lighter)

burning
(makes print darker)

Ansel Adams in
his darkroom





(Adams)

straight
print

Ansel Adams, Clearing Winter Storm, 1942

toned
print

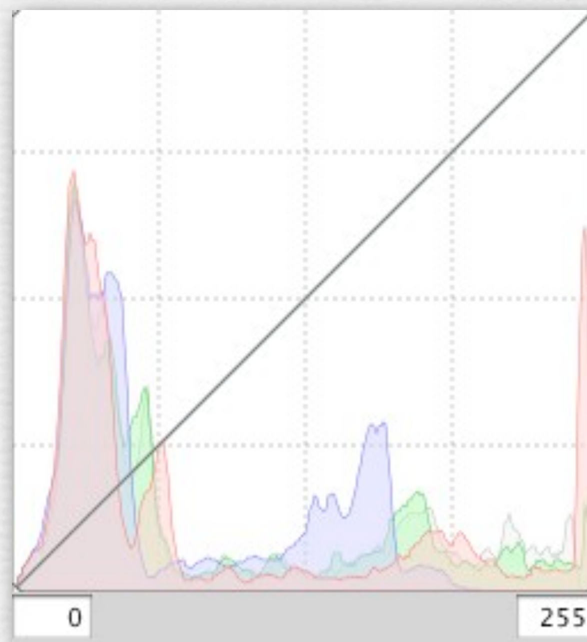
Ansel Adams, Clearing Winter Storm, 1942

Histogram equalization

1. convert image to L*a*b* in range [0,1]
2. calculate histogram of L* channel $pdf(i) = \frac{N_i}{N}$,

where N_i is the number of pixels of intensity i,
and N is the total number of pixels

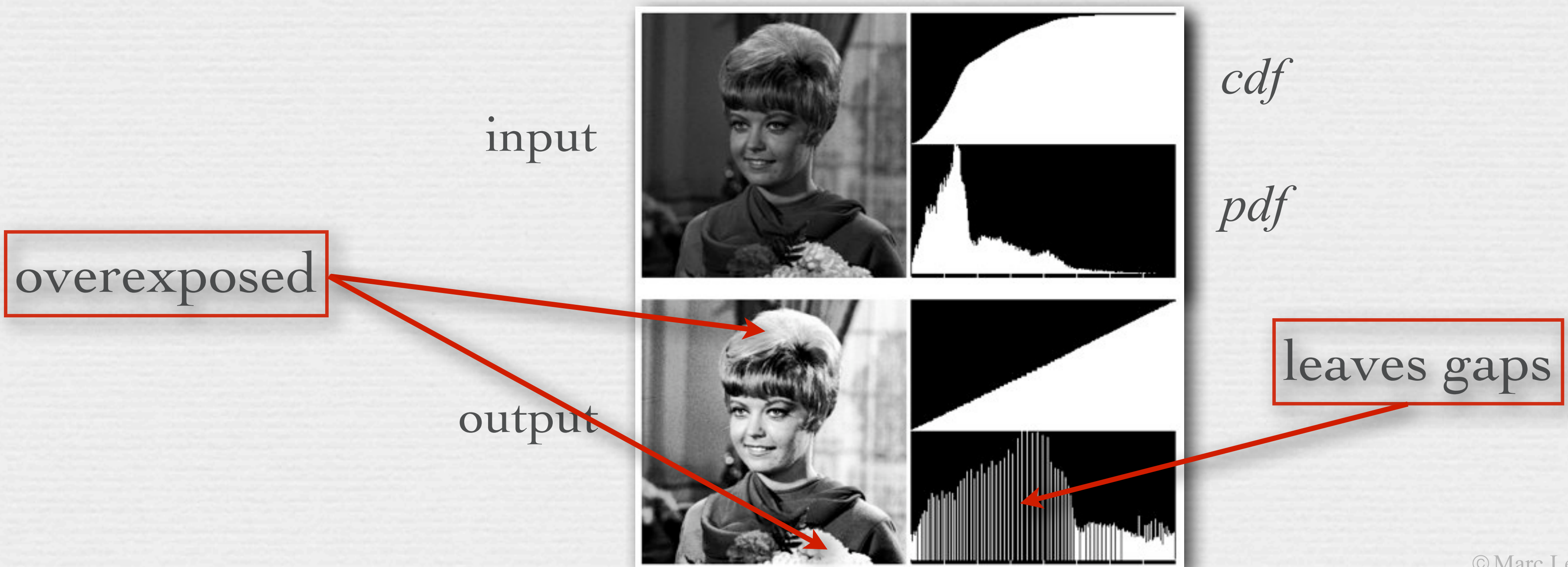
image



histogram

Histogram equalization

1. convert image to L*a*b* in range [0,1]
2. calculate histogram of L* channel $pdf(i) = \frac{N_i}{N}$, where N_i is the number of pixels of intensity i, and N is the total number of pixels
3. calculate cumulative density function $cdf(i) = \sum_{j=0}^{i} pdf(j)$
4. re-map each pixel using $I_{out} = cdf(I_{in}) \times 255 / N$ (for 8-bit pixels)

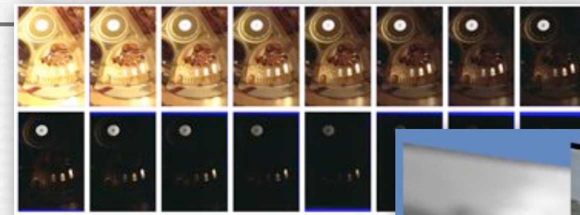


Recap

- in CRTs luminance = voltage$^{\gamma}$ where $\gamma \approx 2.5$, so television cameras output luminance$^{1/\gamma}$ to compensate
  - NTSC cameras use luminance$^{0.5}$, yielding a *system gamma*, to compensate for human *dark adaptation* during viewing
- digital cameras also gamma transform sensed pixels before storing them in JPEG files
  - while this matches television cameras, another good reason is perceptual uniformity, thereby reducing quantization artifacts
  - for sRGB cameras, γ = 1/2.2
- *tone mapping* methods may include
  - contrast expansion
  - additional gamma mapping
  - histogram equalization
  - local methods, like dodging & burning

Questions?

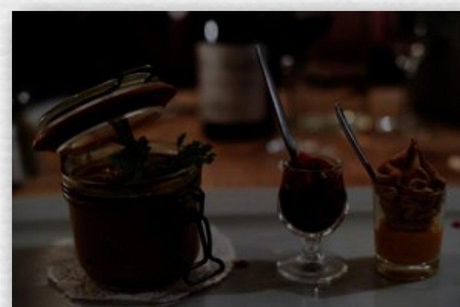
High dynamic range imaging (review)



- step 1: capturing HDR images
- step 2a: direct display of HDR images, or
- step 2b: tone mapping to create an LDR image for display

you’re not responsible for HDR imaging on your final

- goals of tone mapping
  - squeeze 12 bits of sensor into 8 bits of JPEG
  - or squeeze >12 of HDR image into 8 bits of JPEG
  - apply mapping for human adaption if scene was very dark
  - or bright...



Cathedral,
Valencia

(Marc Levoy)

Cathedral,
Valencia

(Marc Levoy)

Cathedral,
Valencia

(Marc Levoy)

tone mapping by exposure and gamma

Cathedral, Valencia

(Marc Levoy)

tone mapping by histogram equalization

Cathedral, Valencia

(Marc Levoy)

Tone mapping techniques
(slides from Fredo Durand)

- image has 10,000:1 dynamic range, projector has ~200:1
- how can we compress the image's dynamic range?

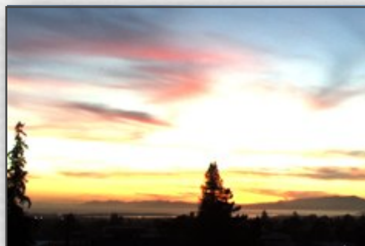
Global tone mapping operators

- gamma compression applied independently on R,G,B

  output = input$^\gamma$ (γ = 0.5 here)

- colors become washed out

input



output

$(1.0, 0.4, 0.2)^{0.5} = (1.0, 0.63, 0.44)$

(try it yourself in Photoshop)

Global tone mapping operators

- ✦ gamma compression on intensity only
- ✦ saturated but light colors become garish

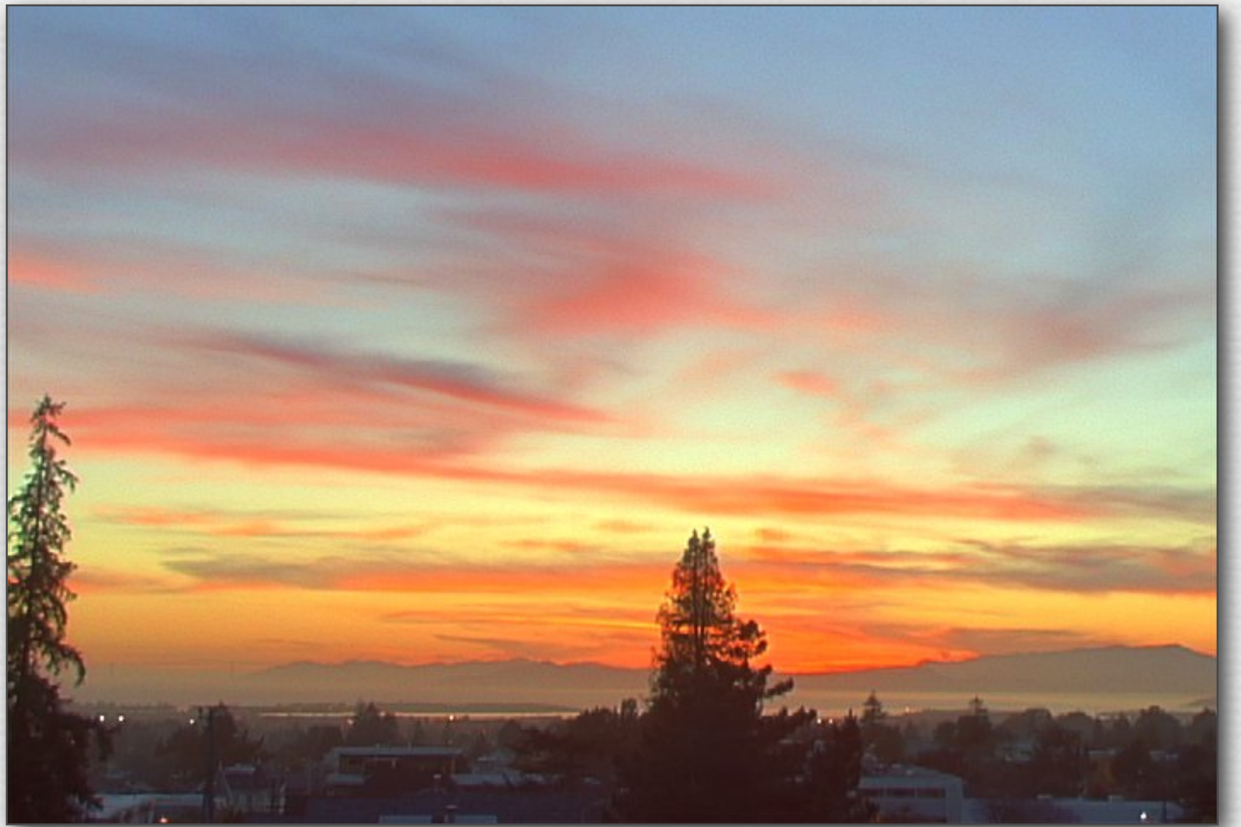
HSL double cone

luminance

(e.g. 0.3R+0.7G+0.01B)

chrominance

(e.g. R/lum, G/ lum, B/lum)



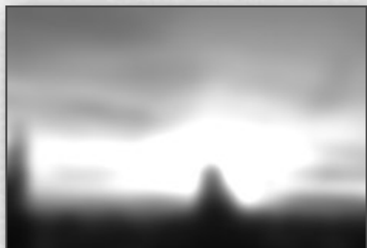
(from our convolution applet)

0.00	-2.00	0.00
-2.00	9.00	-2.00
0.00	-2.00	0.00

Local tone mapping operators

- reduce contrast of low frequencies, while preserving high frequencies [Oppenheim 1968, Chiu et al. 1993]
- produces halos!

low frequency



(e.g. Gaussian blur)

high frequency

(e.g. original - Gaussian)

chrominance

Local tone mapping operators

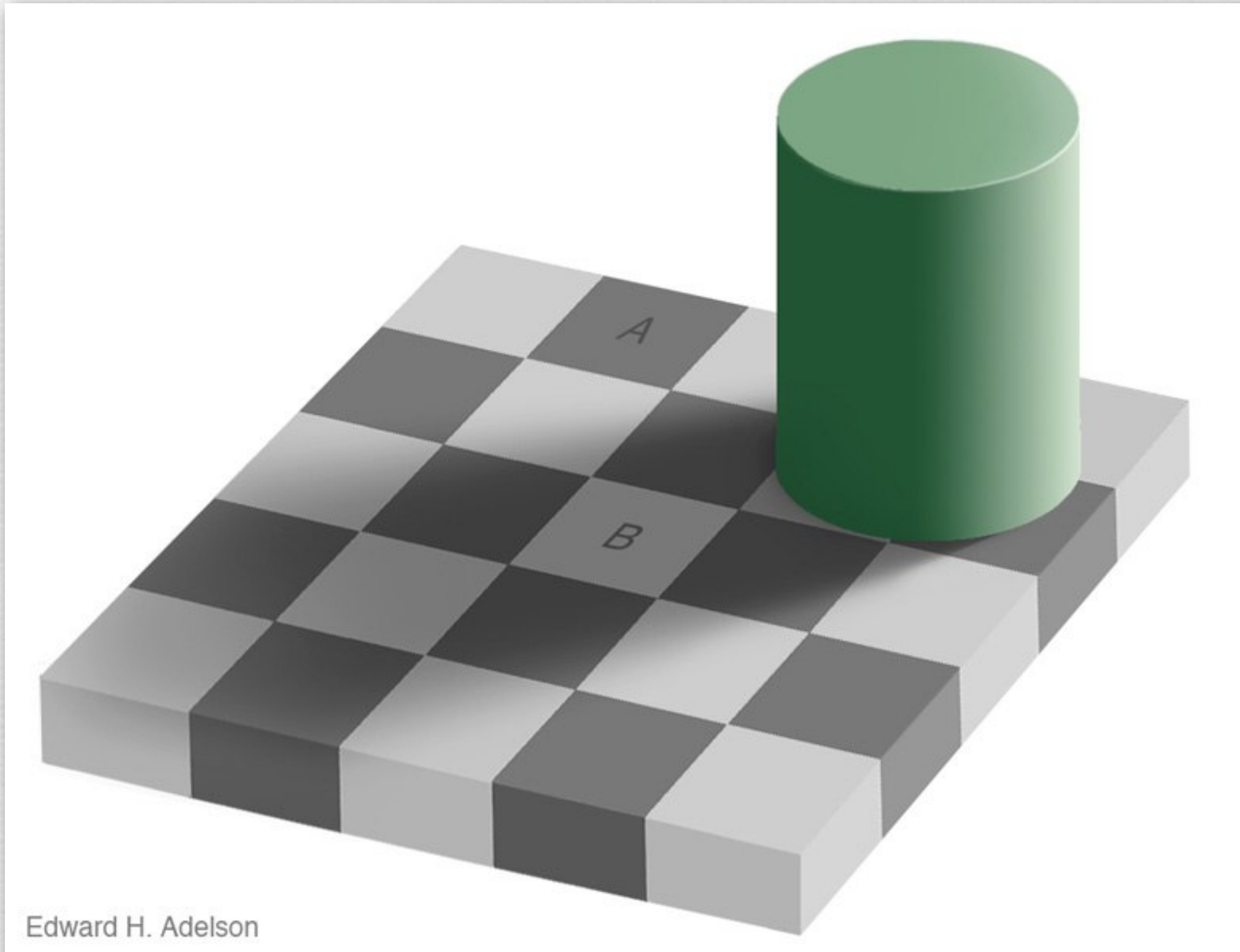
- bilateral filtering to compute large scale image without blurring across edges, remainder is detail image (no halos!); reduce contrast of large scale, while preserving details
[Durand and Dorsey SIGGRAPH 2002]

large scale

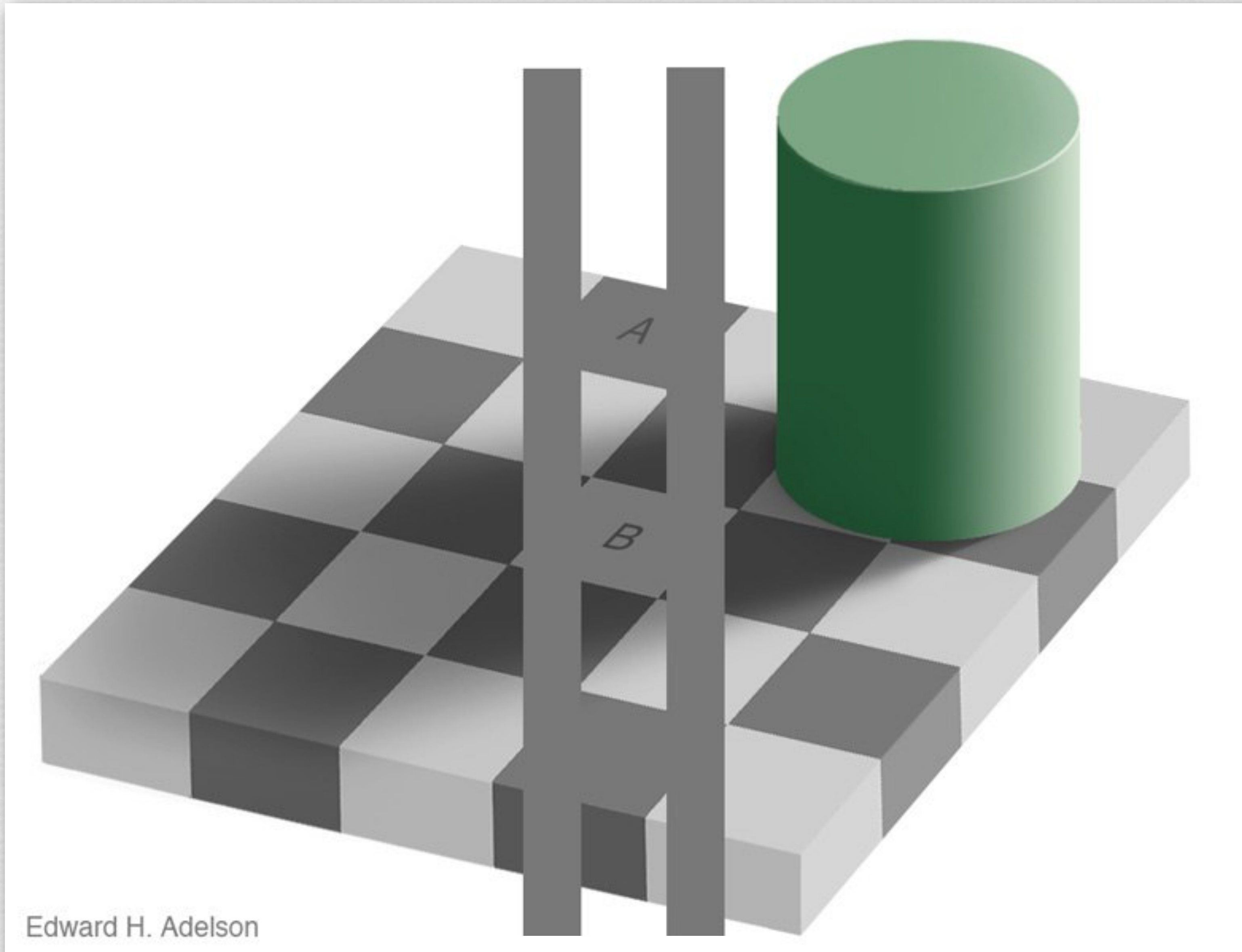
detail

chrominance

The importance of local contrast

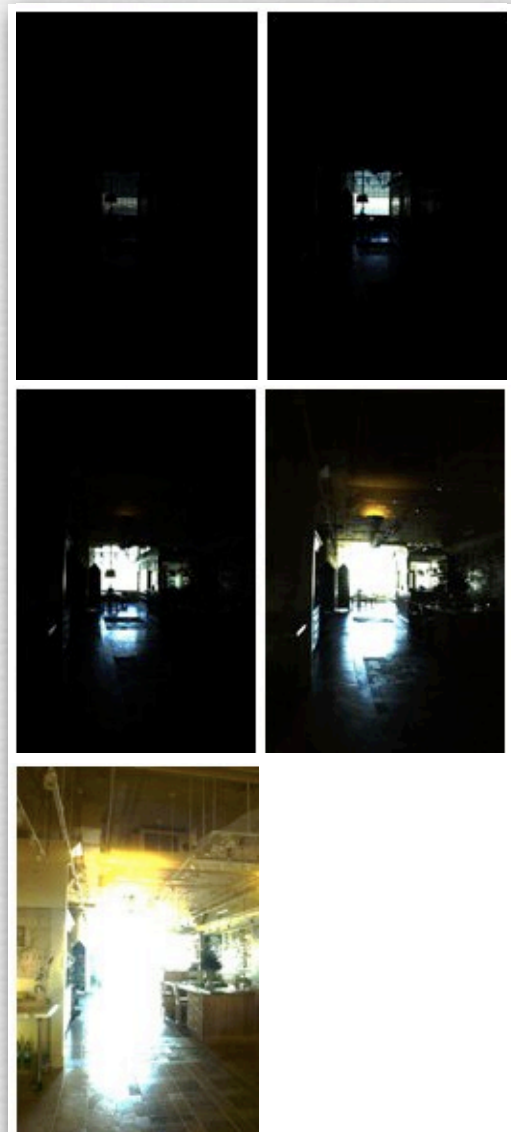


The importance of local contrast

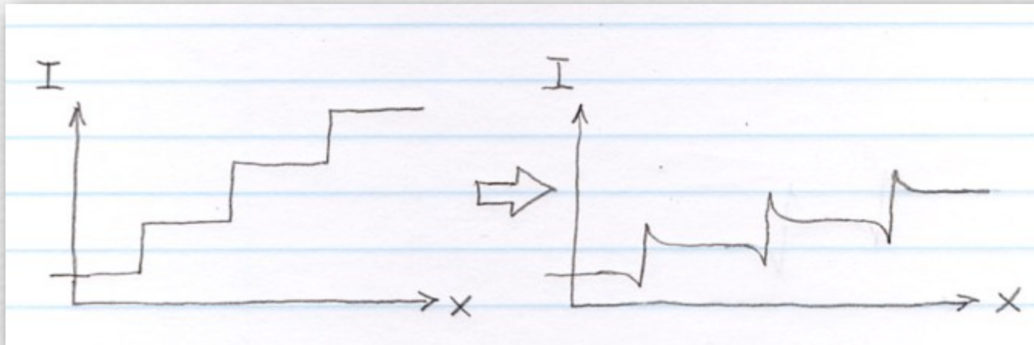


Tone mapping using bilateral filters

[Durand and Dorsey SIGGRAPH 2002]



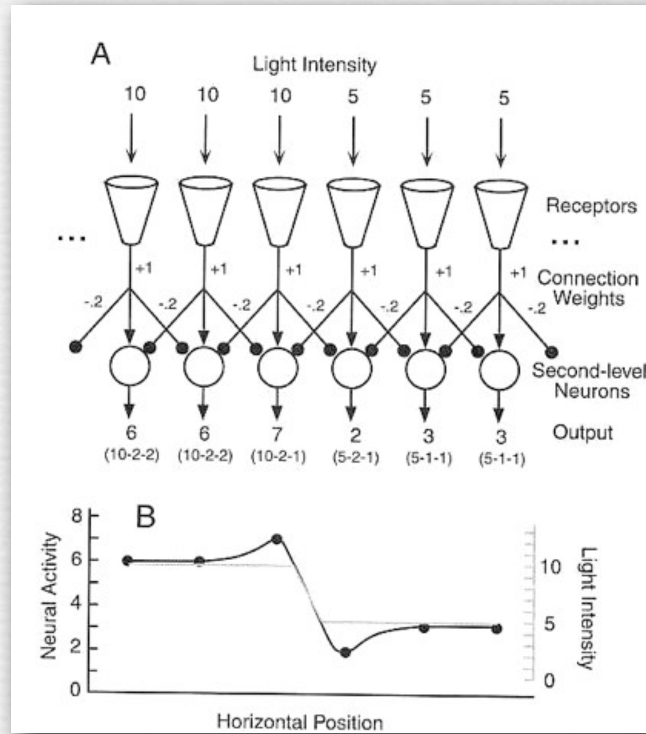
Why might tone mapping look cartoony? (contents of whiteboard)



- a step wedge (at left) is converted by a tone mapping operator that enhances local contrast to the plot at right
  - the human eye does this internally due to lateral inhibition, but that doesn't necessarily mean we want to present an image like this to the human eye!

Mach bands and lateral inhibition

the Mach band illusion: each wedge should appear brighter on its right side



(Goldstein or Wolfe)

- lateral inhibition among receptive fields in the retina is equivalent to image convolution with a sharpening kernel

La Grande Jatte, Georges Seurat, 1884

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(Panasonic ZS3,
1/30s, ISO 125)

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(Panasonic ZS3,
1/30s, ISO 250)

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(Panasonic ZS3,
1/25s, ISO 400)

(Marc Levoy)

Commissary, Fort Ross, CA, 2010

(Panasonic ZS3, 1/13s, ISO 400)

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(Panasonic ZS3,
1/8s, ISO 400)

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(tone mapped HDR using Photomatix v3.3.2's "detail enhancer" algorithm)

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(tone mapped HDR using Photomatix v3.3.2's "tone compressor" algorithm)

(Marc Levoy)

Commissary,
Fort Ross, CA,
2010

(tone mapped HDR using Photomatix v3.3.2's "exposure fusion" algorithm)

(Marc Levoy)

based on [Mertens 2007]

- directly blends original images, without first computing an HDR image
- downweights noisy and saturated pixels
- multi-band blending to avoid seams
- not physically based, but simple and fast

Commissary,
Fort Ross, CA,
2010

(tone mapped HDR using Photomatix v3.3.2's "exposure fusion" algorithm)

(Marc Levoy)

Stanford Arcade, 2009 (1/160s, f/6.3, ISO 100)

(Marc Levoy)

Stanford Arcade, 2009 (1/125s, f/5.6, ISO 100)

(Marc Levoy)

Stanford Arcade, 2009 (1/100s, f/5.4, ISO 100)

(Marc Levoy)

Stanford Arcade, 2009 (Photomatix 3.3.2, “tone compressor” algorithm)

(Marc Levoy)

Stanford Arcade, 2009 (1/125s, f/5.6, ISO 100)

(Marc Levoy)

Stanford Arcade, 2009 (1/125s, f/5.6, ISO 100)

(Marc Levoy)

Stanford Arcade, 2009 (Photomatix 3.3.2, “tone compressor” algorithm)

(Marc Levoy)

The HDR "look"

(Trey Ratcliff, http://www.stuckincustoms.com)

The HDR "look"

(Trey Ratcliff, http://www.stuckincustoms.com)

The HDR “look”

(Trey Ratcliff, http://www.stuckincustoms.com)

Recap

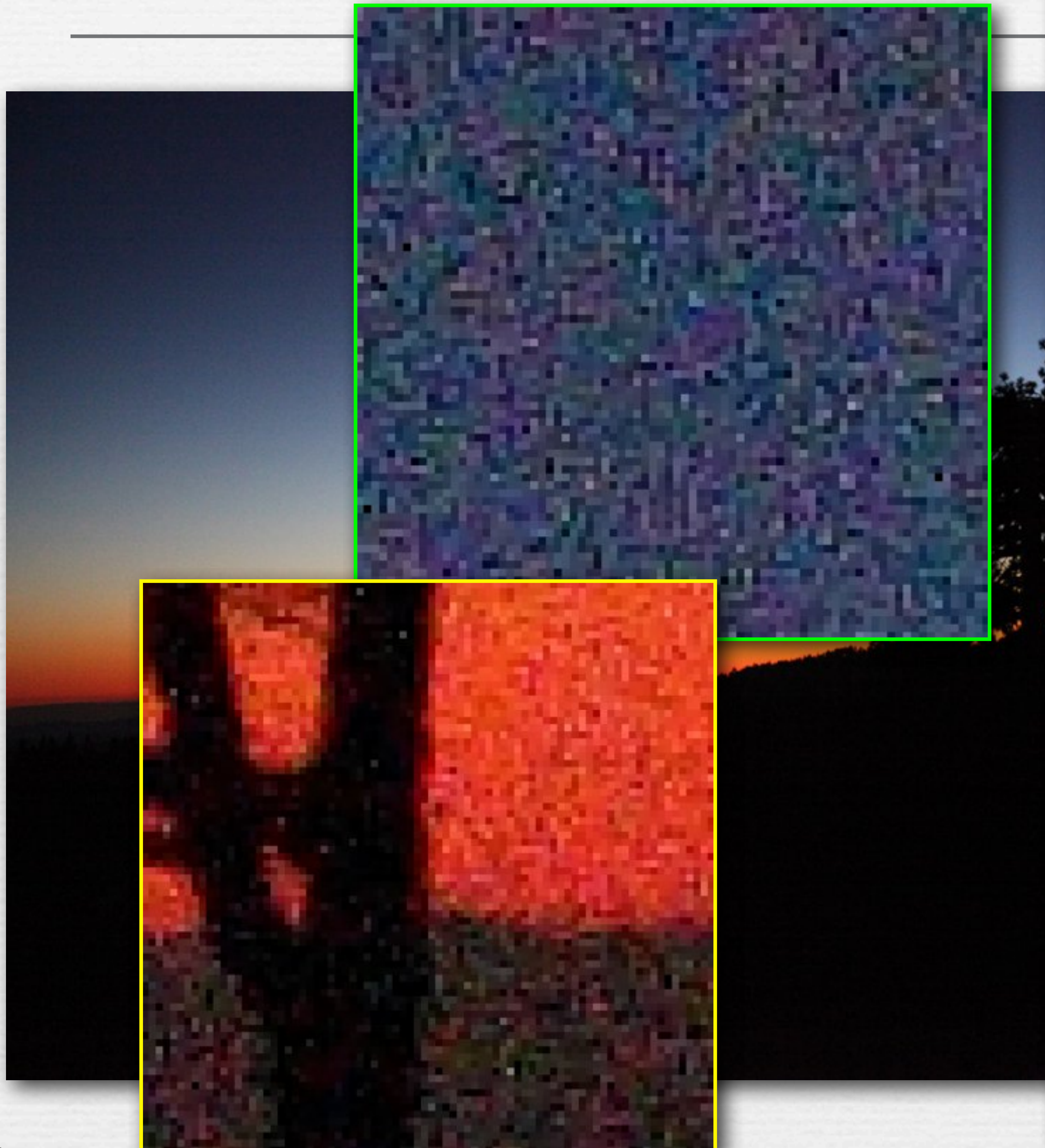
- high dynamic range (HDR) imaging is useful, and a new aesthetic, but is not necessary in most photographic situations
  - SLRs have more useful dynamic range (~12 bits) than point-and-shoot cameras or cell phones, i.e. w/o shadows being noisy
- low dynamic range (LDR) tone mapping methods apply to HDR, but reducing 12 bits to 8 bits using only global methods is hard
  - the reduction is needed for JPEG, display, and printing
- successful methods reduce large-scale luminance changes (across the image) while preserving *local contrast* (across edges)
  - use bilateral filtering to isolate large-scale luminance changes
- these methods mimic *lateral inhibition* in the human visual system
  - but this may not justify applying them to an image before sensing

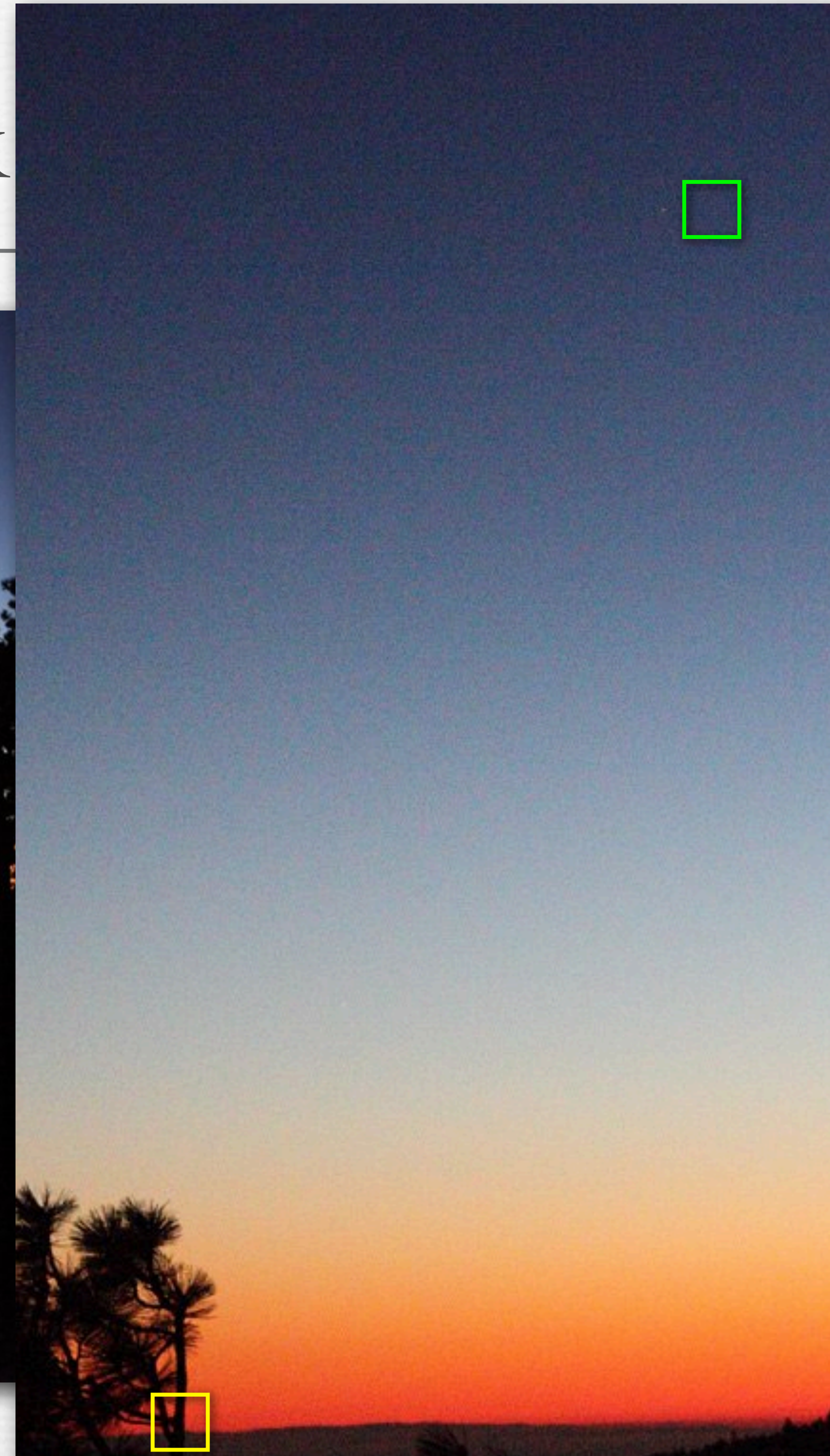
Questions?

Camera pixel pipeline

sensor → analog to digital conversion (ADC) → processing: demosaicing, tone mapping & white balancing, denoising & sharpening, compression → storage

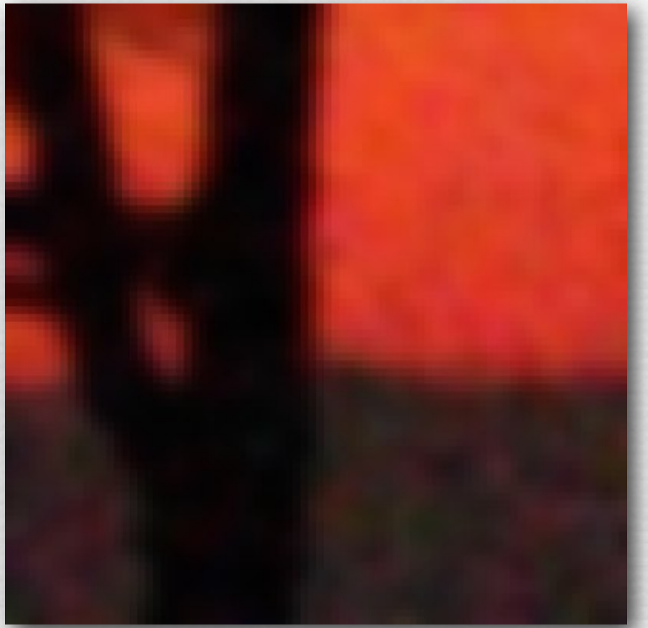
Canon 5D II at dusk





Denoising

RAW (ISO 6400)



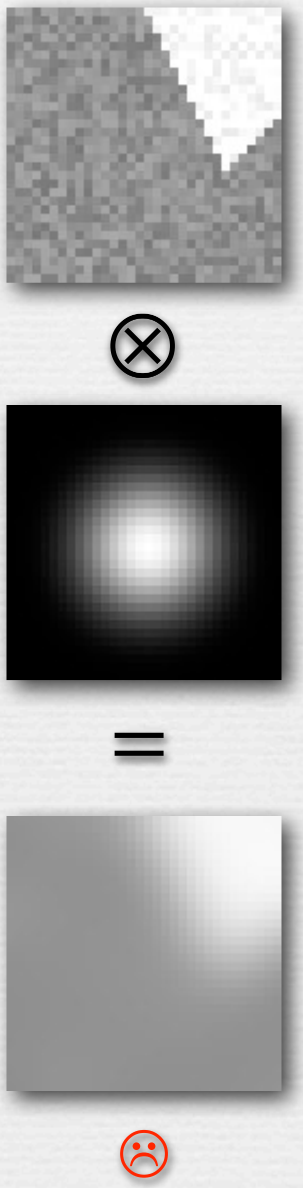
Gaussian blur, radius = 1.3

Canon denoising

- goal is to remove sensor noise
  - blurring works, but also destroys edges
  - I don't know what Canon does, but here's something that works...

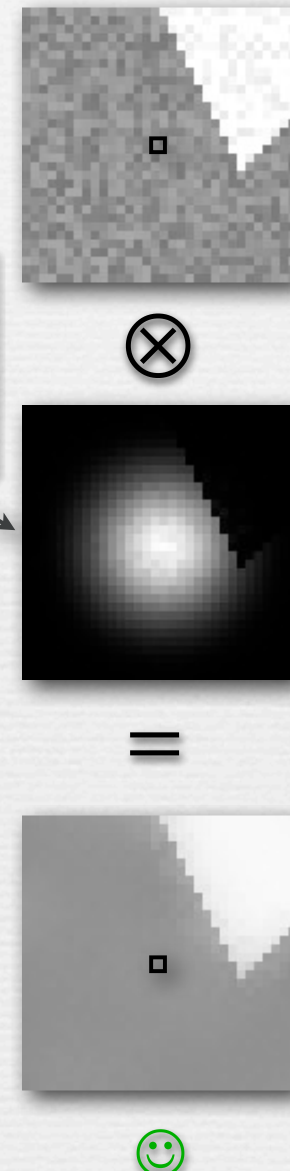
Bilateral filtering [Tomasi ICCV 1998]

- assume the image is piecewise constant with noise added
- therefore, nearby pixels are probably a different noisy measurement of the same data
- blurring doesn't work
- we should do a weighted blur where the weight is the probability a pixel is from the same piece of the scene



Bilateral filtering

- if the pixels are similar in intensity, the probability they are from the same piece of the scene is high
- so perform a convolution where the weight assigned to nearby pixels falls off
  - with increasing *(x,y)* distance from the pixel being blurred
  - with increasing intensity difference from the pixel being blurred
- i.e. blur in *domain* and *range* dimensions!

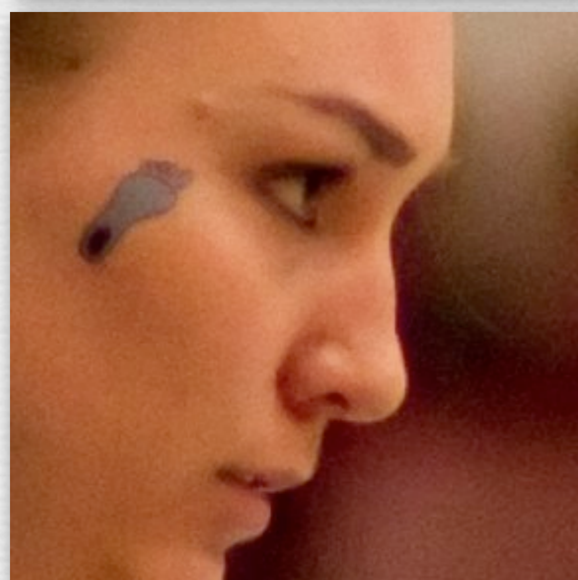


Example of bilateral filtering

Women's gymnastics

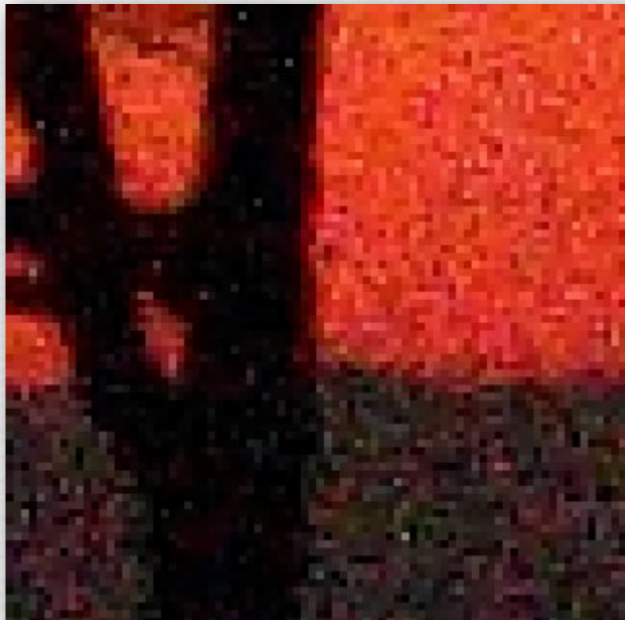
(Canon 7D, 1/1000 sec, ISO 3200, f/1.8, 85mm)

original



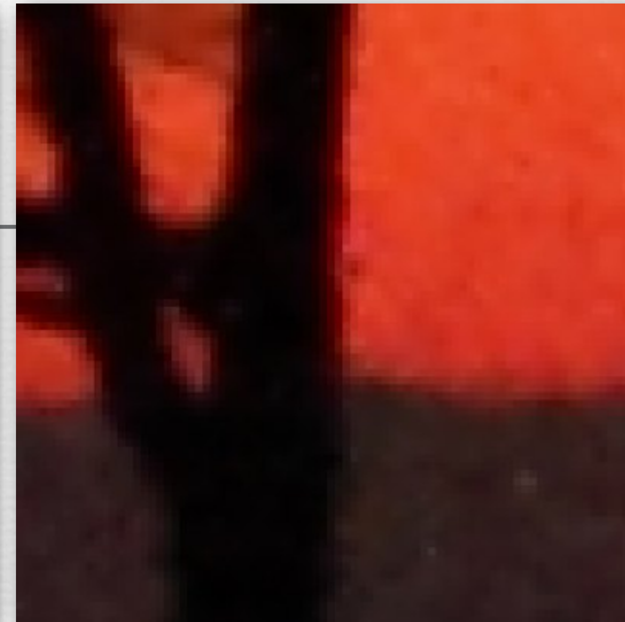
denoised in Noise Ninja

Denoising

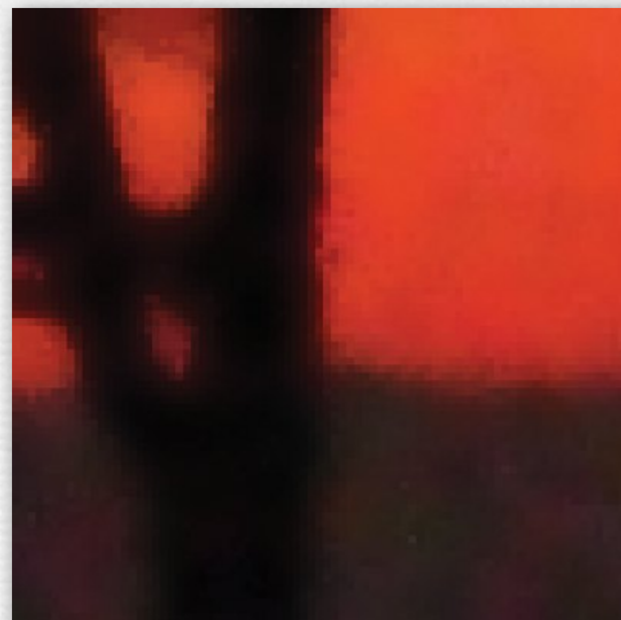


RAW (ISO 6400)

Gaussian blur, radius = 1.3



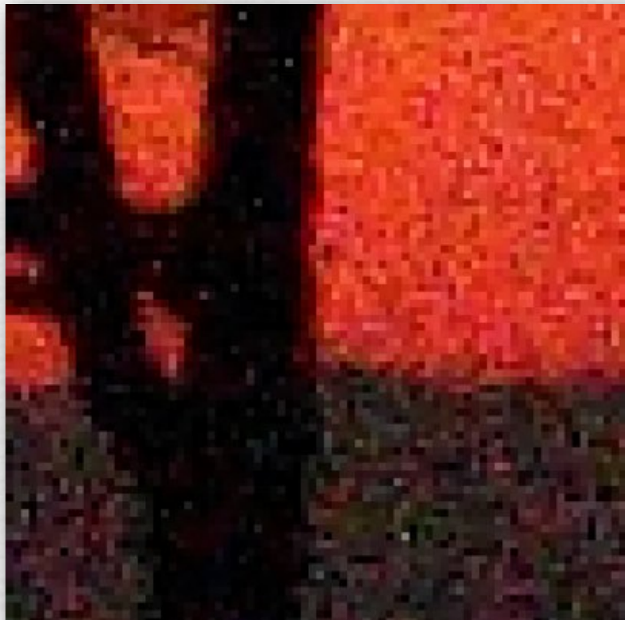
Canon denoising



bilateral filtering

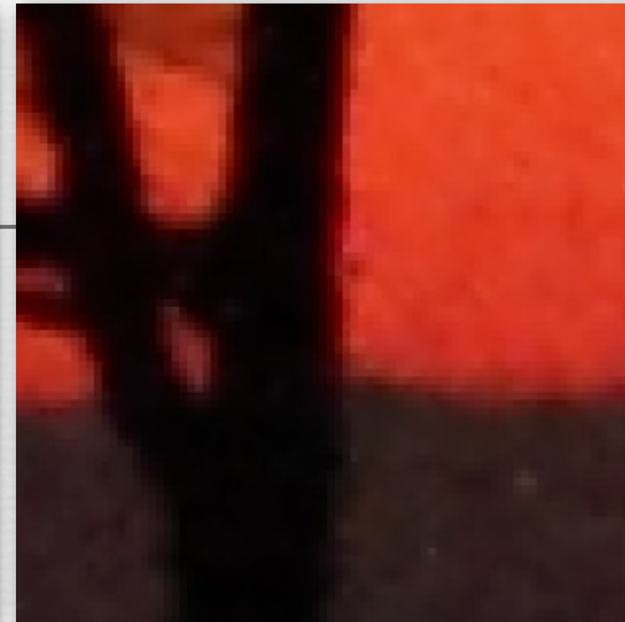
- bilateral filtering removes sensor noise without blurring edges
- can easily be extended to RGB

Denoising

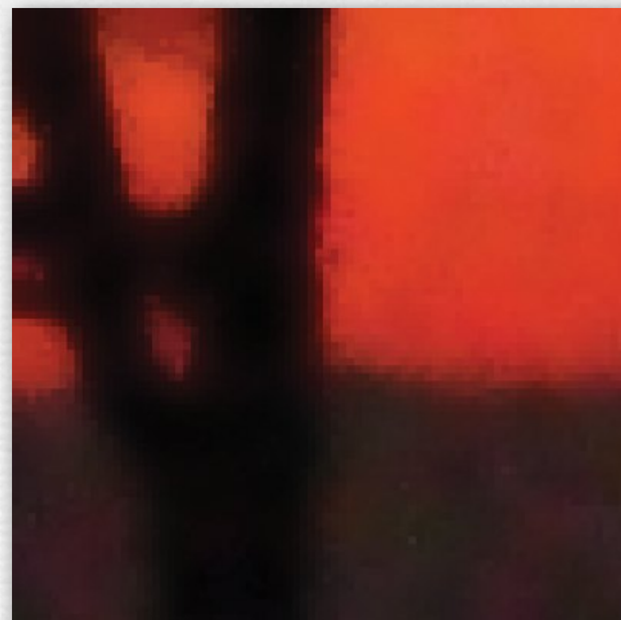


RAW (ISO 6400)

Gaussian blur, radius = 1.3



Canon denoising



bilateral filtering

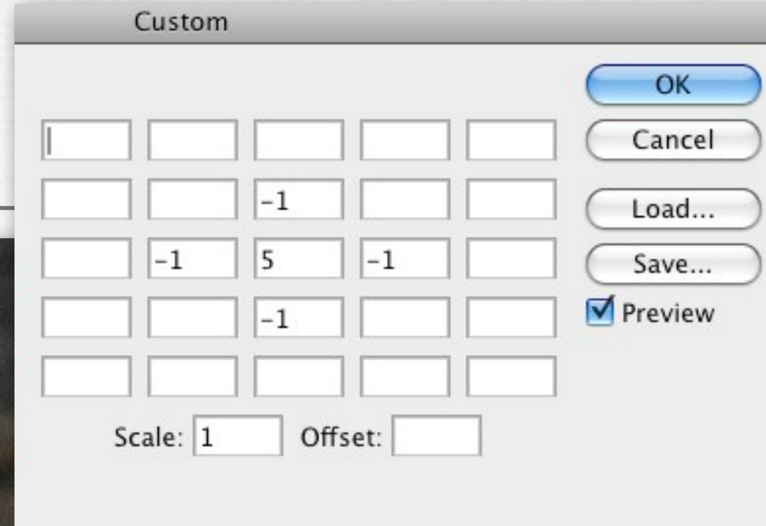
- can be applied more (or less) strongly to chrominance than luminance
- can be combined with demosaicing
- active area of research...

Sharpening

original

(Marc Levoy)

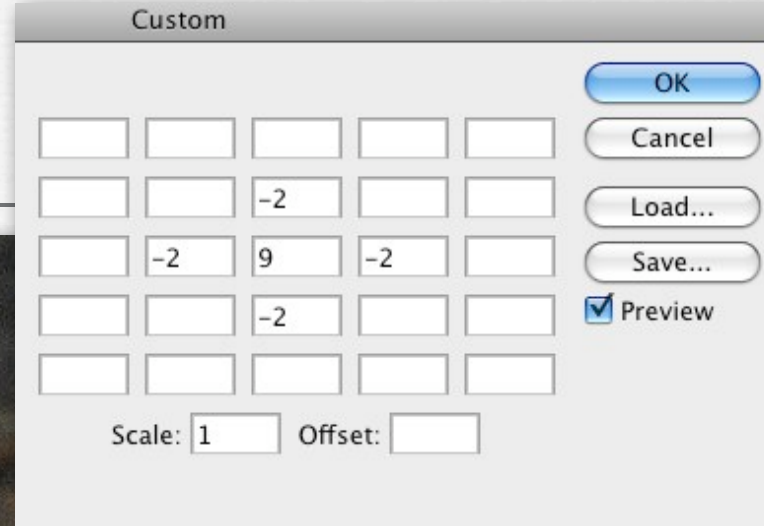
Sharpening



Filter/Other/Custom
in Photoshop CS4

(Marc Levoy)

Sharpening



Filter/Other/Custom
in Photoshop CS4

(Marc Levoy)

Sharpening

original

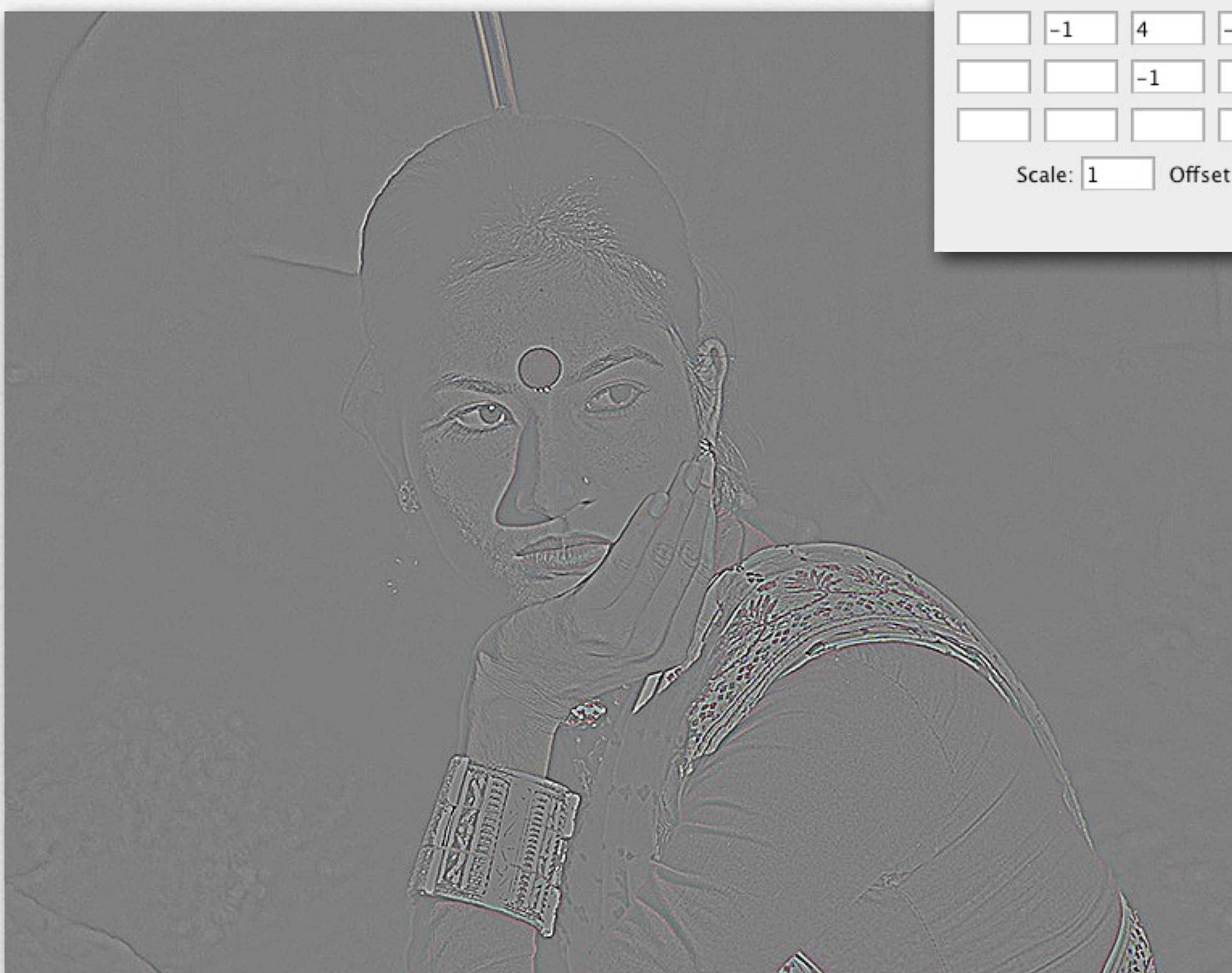
(Marc Levoy)

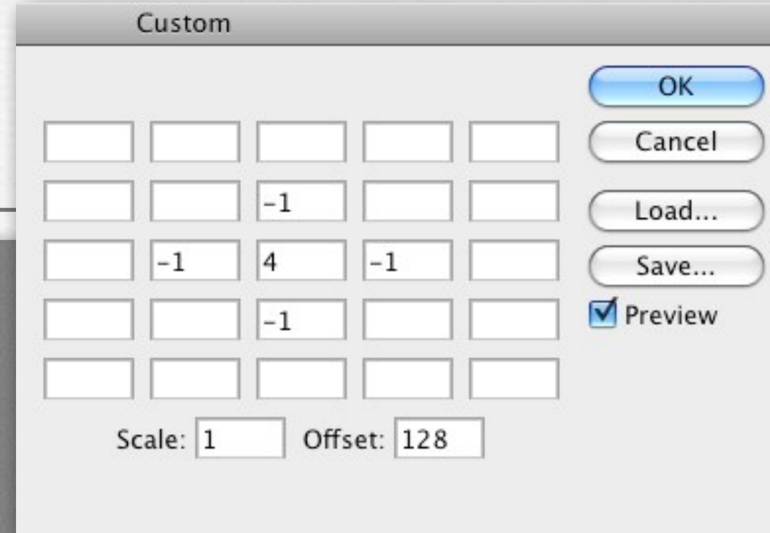
Sharpening

1st layer is original, 2nd layer is sharpened, blend w. 30% opacity

(Marc Levoy)

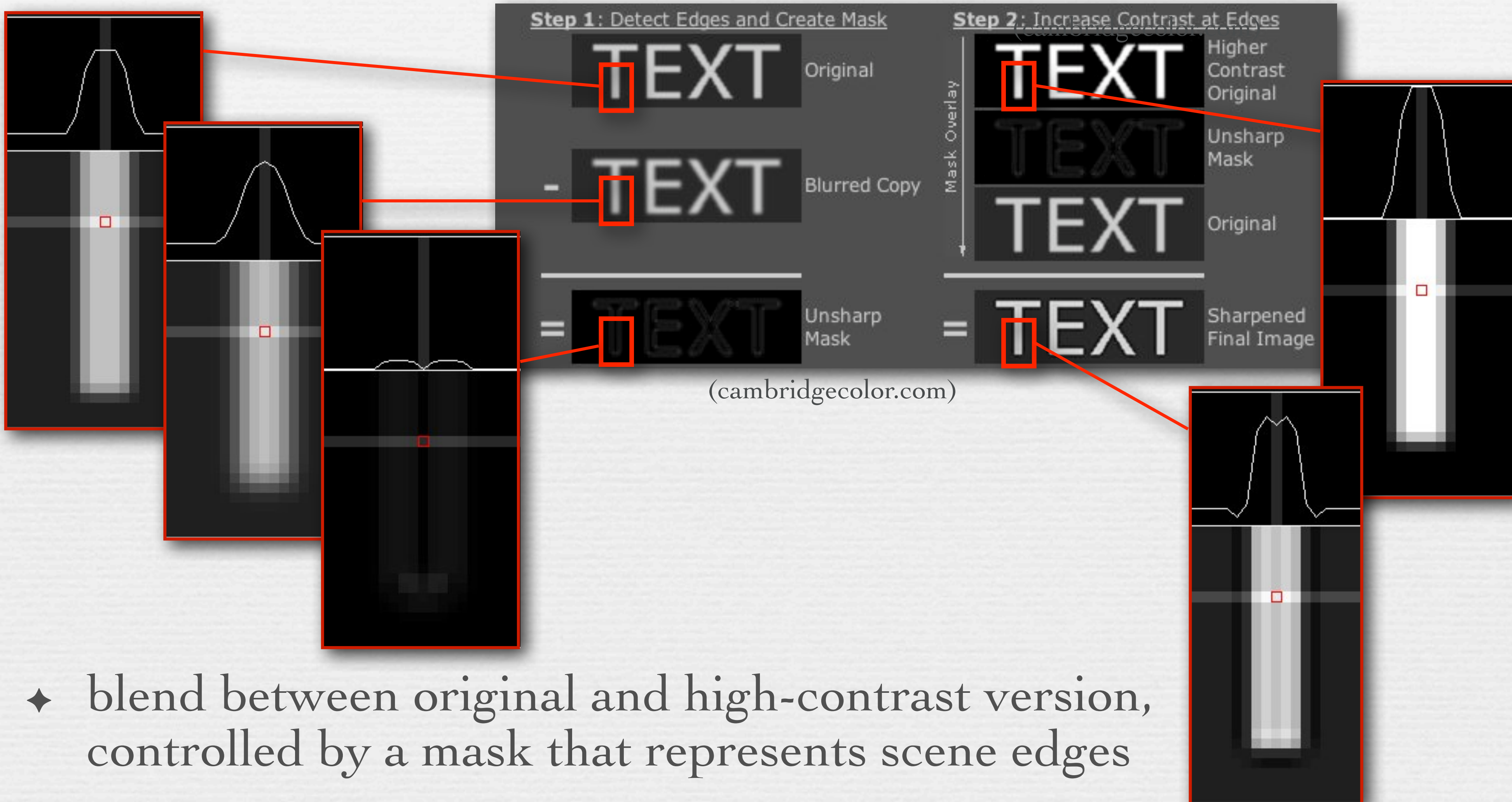
Sharpening





Filter/Other/Custom
in Photoshop CS4

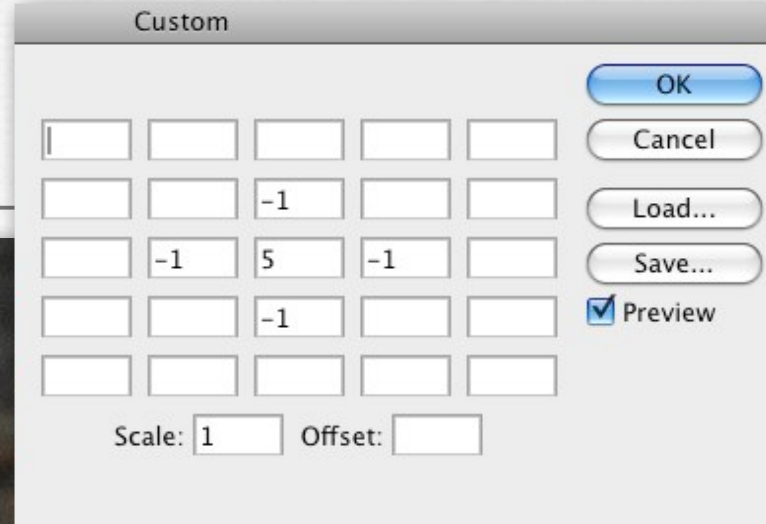
Unsharp masking



(cambridgecolor.com)

- blend between original and high-contrast version, controlled by a mask that represents scene edges
- dropping (thresholding) the darkest mask pixels avoids sharpening noise, and makes the filter non-linear

Sharpening



Filter/Other/Custom in Photoshop CS4

Sharpening

Filter/Sharpen/
Unsharp Mask in CS4

Sharpening

original

Recap

- *bilateral filtering* reduces noise while preserving edges
  - replaces each pixel with a weighted sum of its neighbors, where the weight drops with increasing distance from the pixel in X and Y and with increasing intensity difference
- *unsharp masking* is preferred to sharpening by convolution
  - replaces each pixel with a weighted sum of the original and a contrast-enhanced version, using the latter along edges, where this edge mask is derived from thresholding of original - blurred version
- both are non-linear filters
  - i.e. they are not convolutions by a spatially invariant filter kernel

Questions?

Camera pixel pipeline

sensor → analog to digital conversion (ADC) → processing: demosaicing, tone mapping & white balancing, denoising & sharpening, compression → storage

JPEG files

- Joint Photographic Experts Group
  - organized 1986, standard adopted 1994
- defines how an image is to be compressed (*codec*) into a stream of bytes and the file format for storing that stream
  - file format is JFIF, but people use .JPG or .JPEG extensions
- good for compressing images of natural scenes
  - not so good for compressing drawings or graphics
- *lossy*, so loses quality <u>each</u> time you open → edit → save
  - especially if you crop or shift pixels (hence block boundaries)
  - for *lossless* compression, use PNG or TIFF

EXIF data

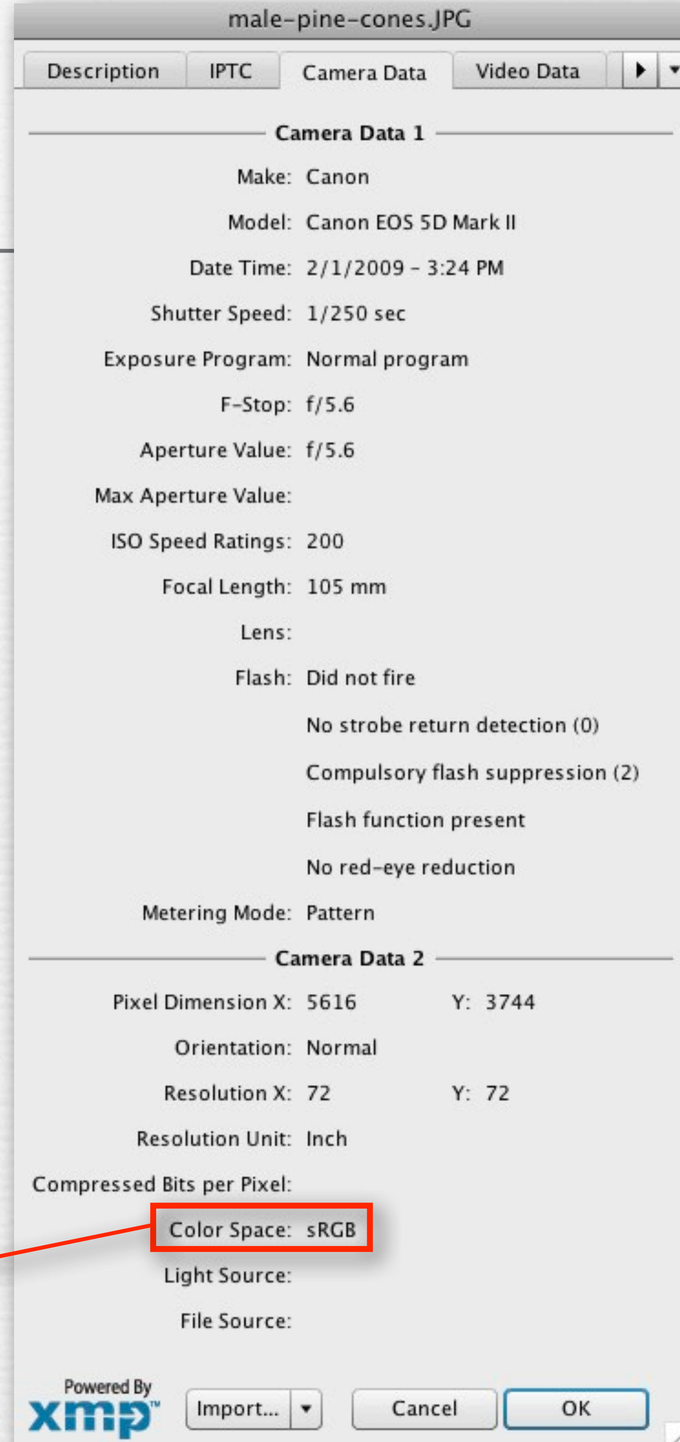
- Exchangeable Image File Format
  - created by Japan Electronic Industries Development Assoc.
- used by nearly every digital camera manufactured today
  - actually a file format
  - JPEG or TIFF file + metadata about the camera and shot
  - .JPG or .JPEG extension is used, not .EXIF

EXIF data

File/File Info in Photoshop CS4

(Marc Levoy)

shot with Canon 5D Mark II



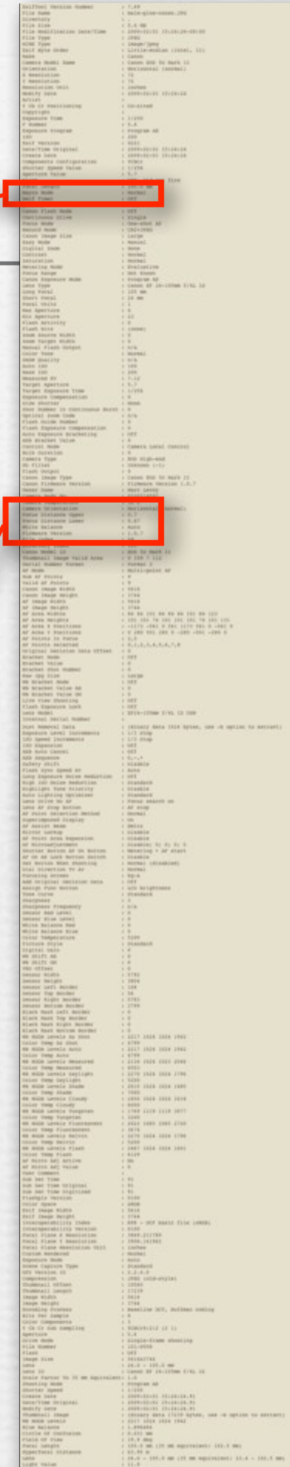
Color Space: sRGB

EXIF data

exiftool

```
Focal Length                              : 105.0 mm
```

shot with Canon 5D Mark II

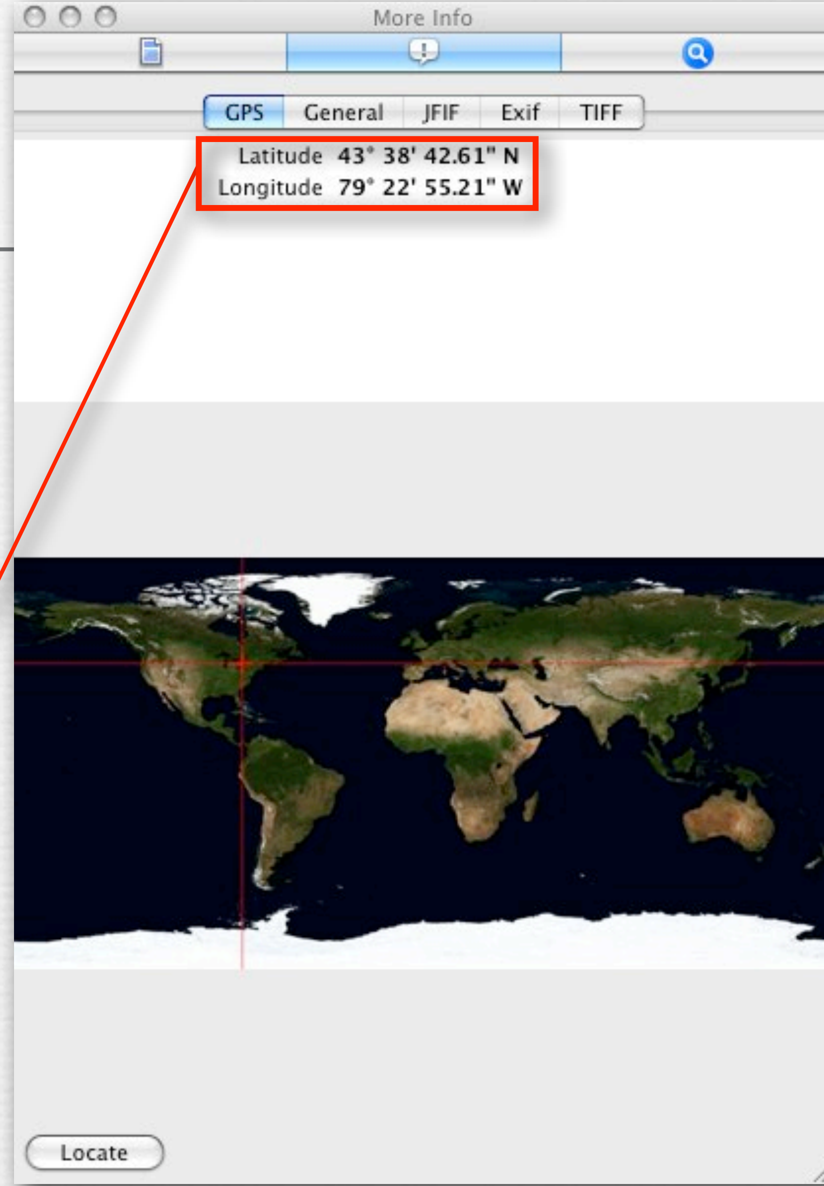


```
Focus Distance Upper                      : 0.7
Focus Distance Lower                      : 0.67
```

EXIF data

Mac Preview

shot with iPhone 3G



Latitude 43° 38' 42.61" N
Longitude 79° 22' 55.21" W

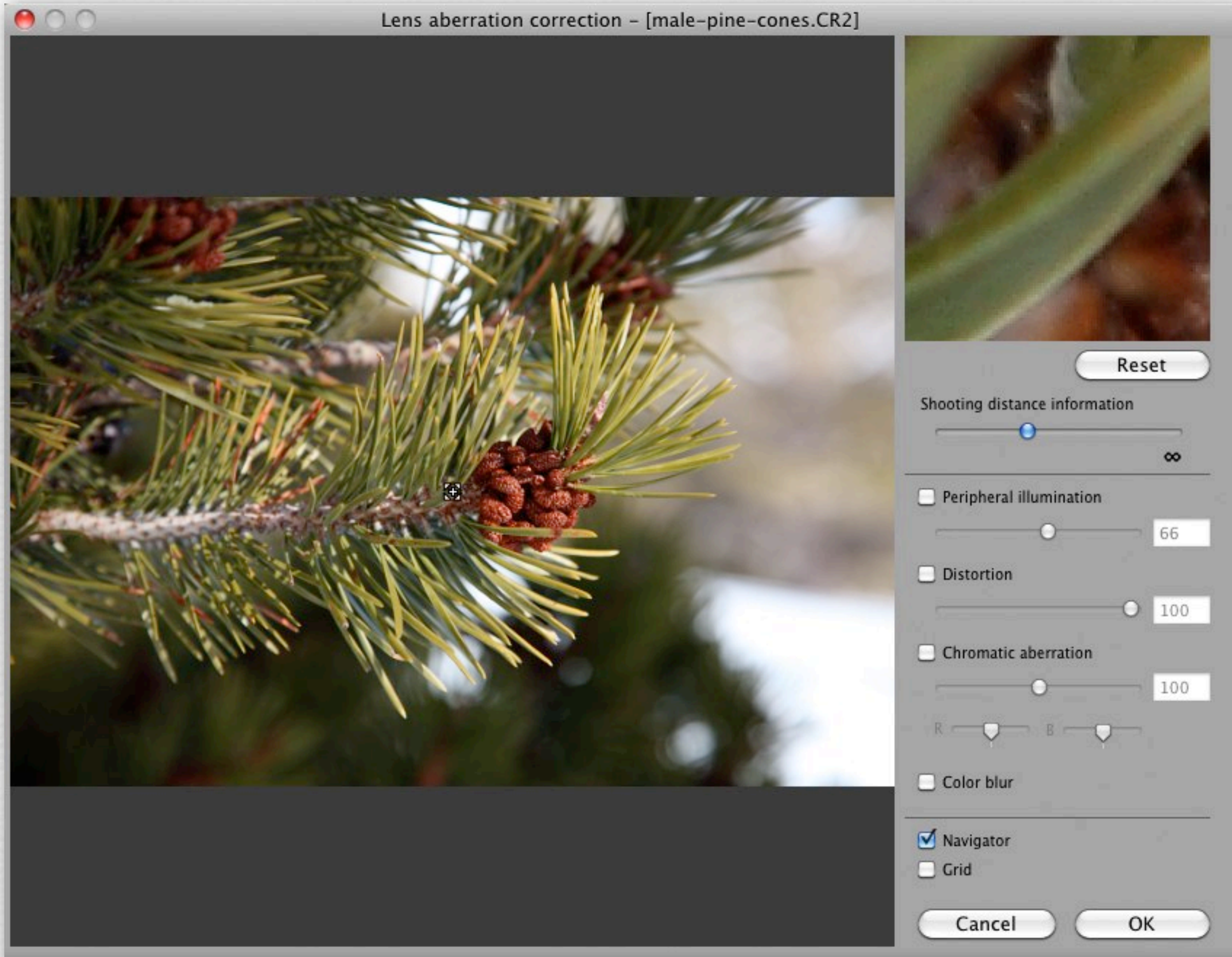
RAW files

- minimally processed images, not even demosaiced
- uncompressed or losslessly compressed
- includes metadata, possibly encrypted
- file format varies by manufacturer
- example extensions: .CR2, .NEF, .RW2

- processed and converted to a JPEG file using
  - proprietary software (e.g. Canon Digital Photo Professional)
  - Photoshop or Lightroom (if they support your camera)
  - freeware programs like `dcraw`
  - but their processing algorithms are all different!

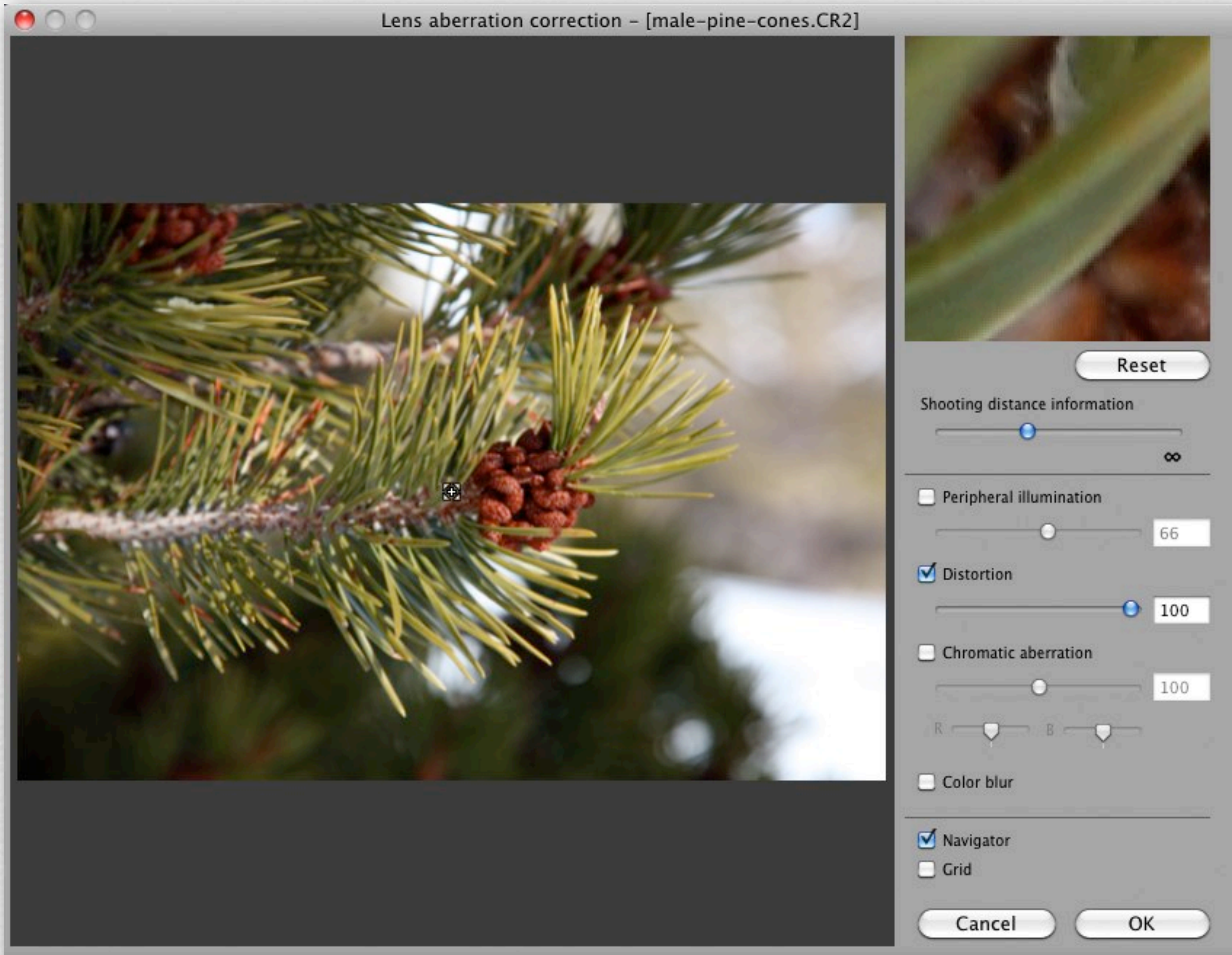
RAW file processor

Lens aberration correction panel in Canon Digital Photo Professional



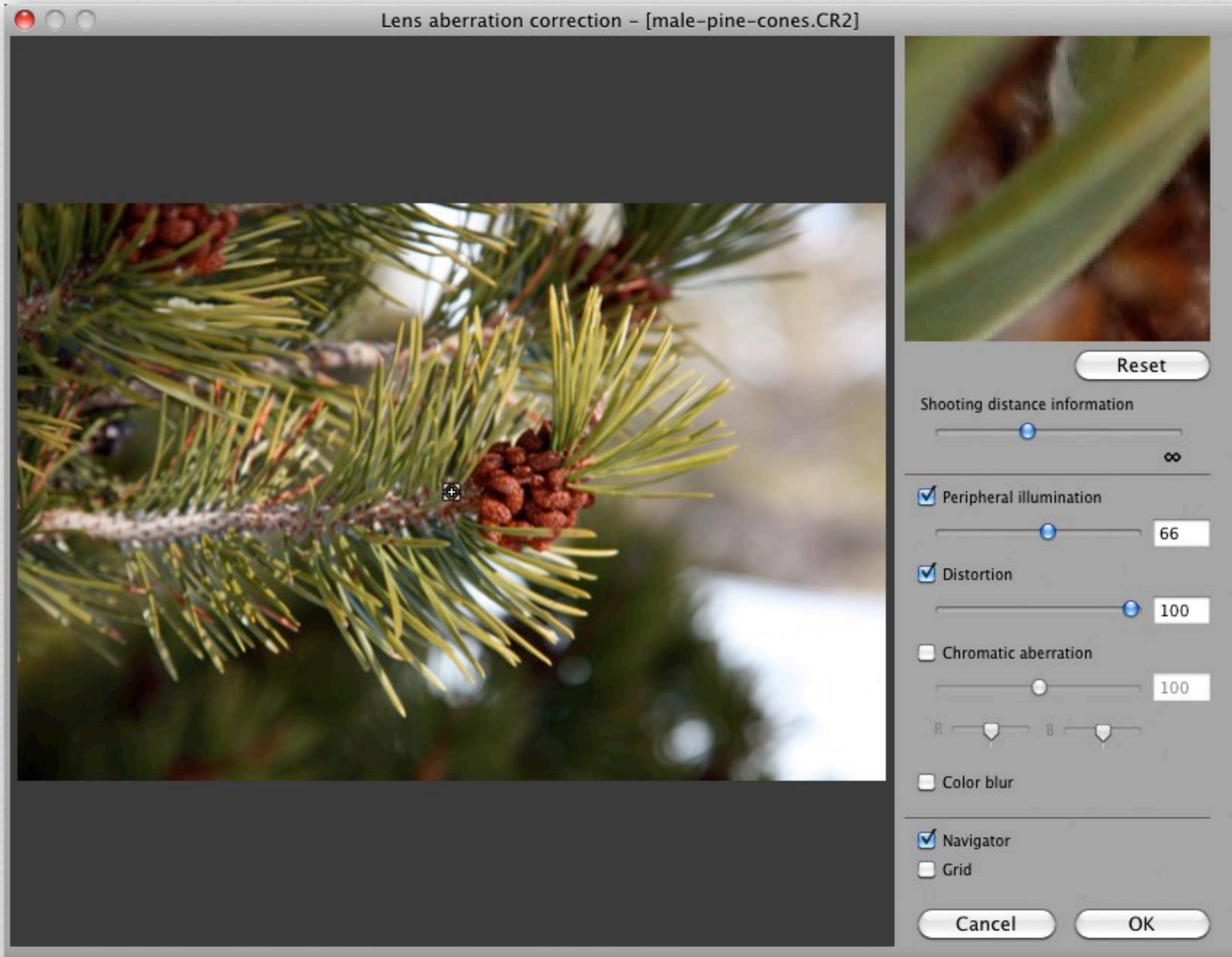
RAW file processor

Lens aberration correction panel in Canon Digital Photo Professional



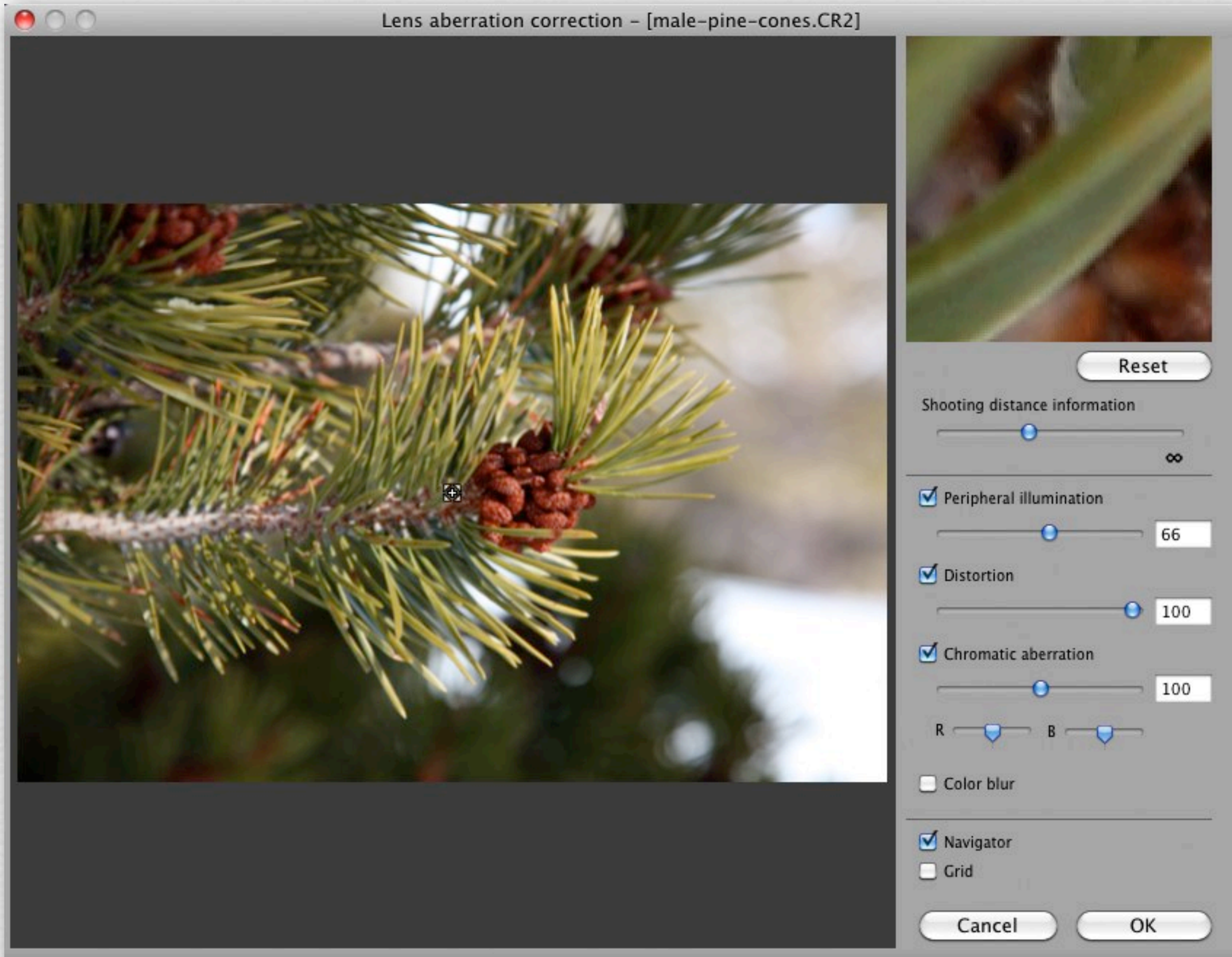
RAW file processor

Lens aberration correction panel in Canon Digital Photo Professional



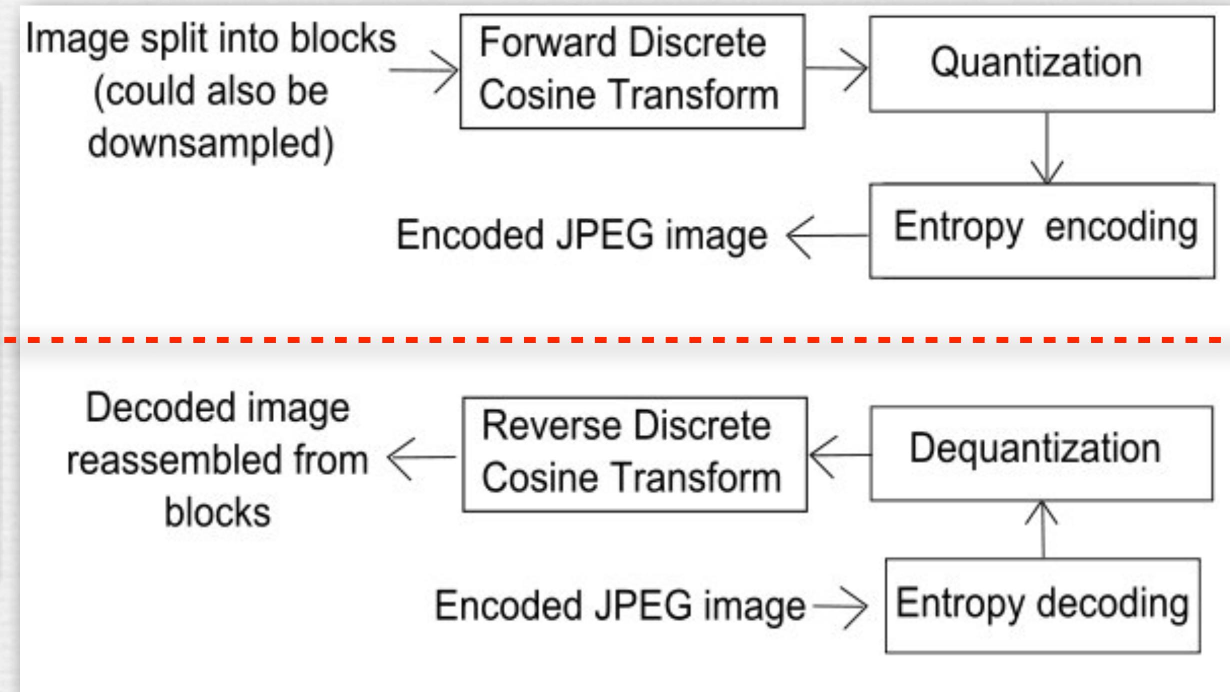
RAW file processor

Lens aberration correction panel in Canon Digital Photo Professional



JPEG image compression

- compression (in camera)



(wikipedia)

- decompression (for display)

- input is Y'CbCr
- Cb and Cr typically downsampled by 2× in X and Y
- each component is compressed separately

JPEG image compression

−76	−73	−67	−62	−58	−67	−64	−55
−65	−69	−73	−38	−19	−43	−59	−56
−66	−69	−60	−15	16	−24	−62	−55
−65	−70	−57	−6	26	−22	−58	−59
−61	−67	−60	−24	−2	−40	−60	−58
−49	−63	−68	−58	−51	−60	−70	−53
−43	−57	−64	−69	−73	−67	−63	−45
−41	−49	−59	−60	−63	−52	−50	−34

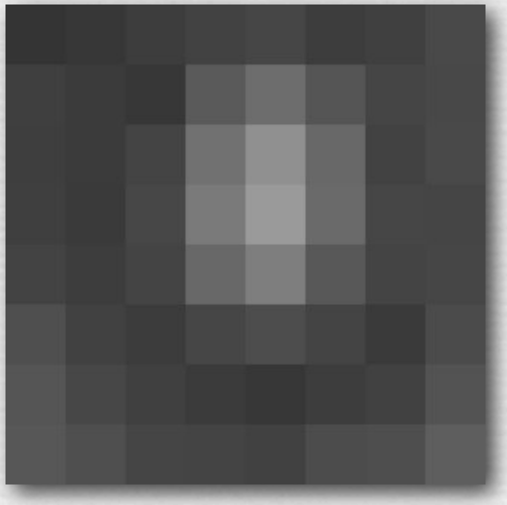
(x →, y ↓)

zero-mean image

52	55	61	66	70	61	64	73
63	59	55	90	109	85	69	72
62	59	68	113	144	104	66	73
63	58	71	122	154	106	70	69
67	61	68	104	126	88	68	70
79	65	60	70	77	68	58	75
85	71	64	59	55	61	65	83
87	79	69	68	65	76	78	94

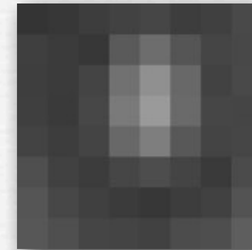
8-bit image

- <u>step #1</u>: split into 8×8 pixel blocks
- <u>step #2</u>: quantize to 8 bits / pixel
- <u>step #3</u>: convert to zero-mean



8×8 pixel block

JPEG image compression



$$\overset{x}{\longrightarrow}\quad \begin{bmatrix} -76 & -73 & -67 & -62 & -58 & -67 & -64 & -55 \\ -65 & -69 & -73 & -38 & -19 & -43 & -59 & -56 \\ -66 & -69 & -60 & -15 & 16 & -24 & -62 & -55 \\ -65 & -70 & -57 & -6 & 26 & -22 & -58 & -59 \\ -61 & -67 & -60 & -24 & -2 & -40 & -60 & -58 \\ -49 & -63 & -68 & -58 & -51 & -60 & -70 & -53 \\ -43 & -57 & -64 & -69 & -73 & -67 & -63 & -45 \\ -41 & -49 & -59 & -60 & -63 & -52 & -50 & -34 \end{bmatrix} \Big\downarrow y$$

zero-mean image

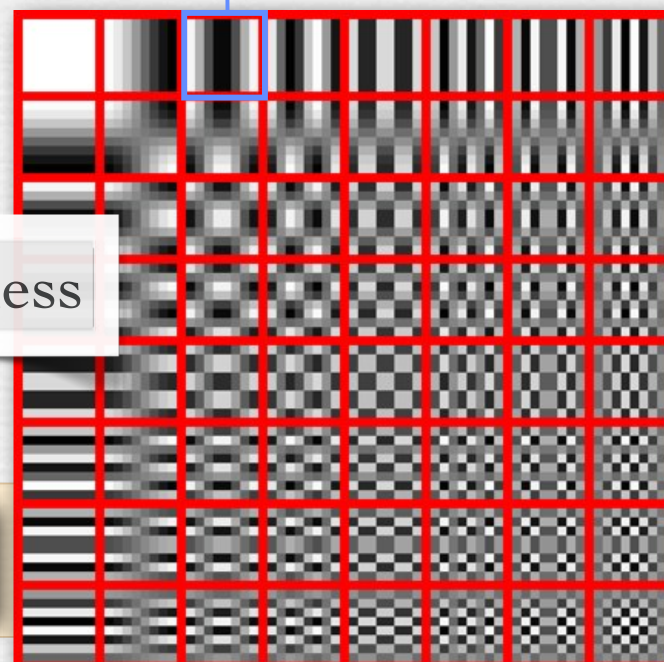
$$\overset{u}{\longrightarrow}\quad \begin{bmatrix} -415 & -30 & -61 & 27 & 56 & -20 & -2 & 0 \\ 4 & -22 & -61 & 10 & 13 & -7 & -9 & 5 \\ -47 & 7 & 77 & -25 & -29 & 10 & 5 & -6 \\ -49 & 12 & 34 & -15 & -10 & 6 & 2 & 2 \\ 12 & -7 & -13 & -4 & -2 & 2 & -3 & 3 \\ -8 & 3 & 2 & -6 & -2 & 1 & 4 & 2 \\ -1 & 0 & 0 & -2 & -1 & -3 & 4 & -1 \\ 0 & 0 & -1 & -4 & -1 & 0 & 1 & 2 \end{bmatrix} \Big\downarrow v$$

discrete cosine transform (DCT)

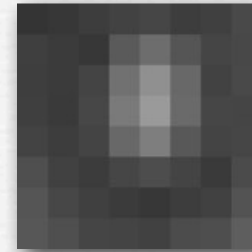
- any 8×8 pixel zero-mean image can be represented by a weighted sum of the 64 different 8×8 pixel *basis functions* shown at right
- step #4: compute the weighting for each basis function using:

lossless

$$G_{u,v} = \alpha(u)\alpha(v)\sum_{x=0}^{7}\sum_{y=0}^{7} g_{x,y}\cos\left[\frac{\pi}{8}\left(x+\frac{1}{2}\right)u\right]\cos\left[\frac{\pi}{8}\left(y+\frac{1}{2}\right)v\right]$$



JPEG image compression



16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

bin size for each coefficient

$u \longrightarrow$, $v \downarrow$

−415	−30	−61	27	56	−20	−2	0
4	−22	−61	10	13	−7	−9	5
−47	7	77	−25	−29	10	5	−6
−49	12	34	−15	−10	6	2	2
12	−7	−13	−4	−2	2	−3	3
−8	3	2	−6	−2	1	4	2
−1	0	0	−2	−1	−3	4	−1
0	0	−1	−4	−1	0	1	2

discrete cosine transform (DCT)

- the human visual system is more sensitive to low frequencies than high frequencies, so quantize the latter coarsely

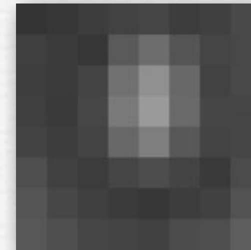
- step #5: quantize the DCT coefficients using bins whose size increases with frequency

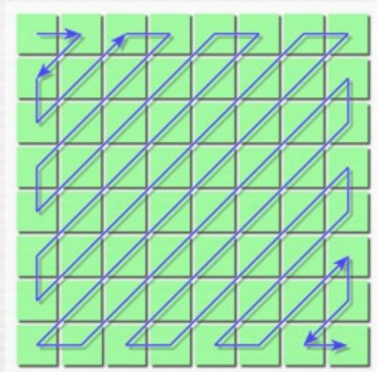
lossy

−26	−3	−6	2	2	−1	0	0
0	−2	−4	1	1	0	0	0
−3	1	5	−1	−1	0	0	0
−4	1	2	−1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

quantized DCT coefficients

JPEG image compression





zig-zag ordering

-26					
-3	0				
-3	-2	-6			
2	-4	1	-4		
1	1	5	1	2	
-1	1	-1	2	0	0
0	0	0	-1	-1	EOB

- step #6: arrange the non-zero coefficients in zig-zag order

lossless

- step #7: use run-length encoding to remove repeated elements
- step #8: apply Huffman coding to reduce number of bits needed for each coefficient

$$\begin{bmatrix} -26 & -3 & -6 & 2 & 2 & -1 & 0 & 0 \\ 0 & -2 & -4 & 1 & 1 & 0 & 0 & 0 \\ -3 & 1 & 5 & -1 & -1 & 0 & 0 & 0 \\ -4 & 1 & 2 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

quantized DCT coefficients

JPEG image compression

Q = 100

2.6 : 1

Q = 25

23 : 1

Q = 1

144 : 1

↓

144:1 looks fine if it's displayed small enough

- not easily comparable to Photoshop quality numbers, since Adobe uses its own (proprietary) encoder

Recap

- RAW files is the direct output of the camera sensor
  - not demosaiced, 16 bits per pixel, losslessly compressed
  - contains metadata, usually proprietary
- JPEG files are a standard format for storing images
  - typically 8 bits per pixel, lossy compression
  - contains metadata in EXIF format
- JPEG's compression format is designed to discard details
  - images are partitioned into blocks of 8 × 8 pixels
  - each block is represented by a weighted sum of sinusoids (DCT)
  - the coefficients of high frequency sinusoids are heavily quantized, which reduces # of bits, hence file size, but also loses images quality
  - these coefficients are losslessly compressed using Huffman coding

Questions?

Slide credits

- Fredo Durand

- Wandell, B., *Foundations of Vision*, Sinauer, 1995.
- Tanser and Kleiner, *Gardner's Art Through the Ages* (10th ed.), Harcourt Brace, 1996.
- Rudman, T., *Photographer's Master Printing Course*, Focal Press, 1998.
- Adams, A., *The Print*, Little, Brown and Co., 1980.
- Goldstein, B.E., *Sensation and Perception,* Wadsworth, 1999.
- Wolfe, J.M., *Sensation and Perception,* Sinauer, 2006.