Tonemapping and bilateral filtering

15-463, 15-663, 15-862
Computational Photography
Fall 2018, Lecture 6

http://graphics.cs.cmu.edu/courses/15-463

Course announcements

- Homework 2 is out.
  - Due September 28th.
  - Requires camera *and* tripod.
  - Still five cameras left if anybody needs one.
  - Start early! Large programming component and generous bonus.

- Any issues with Homework 1?
  - How did you find homework 1 in general?
  - Which part of homework 1 did you enjoy the most?

Overview of today's lecture

- Leftover from lecture 5: optimal weights for HDR merging.
- Color calibration and homography estimation.
- Tonemapping.
- Edge-aware filtering and bilateral filtering.
- Back to tonemapping.
- Some notes about HDR and tonemapping.

Slide credits

Many of these slides were inspired or adapted from:

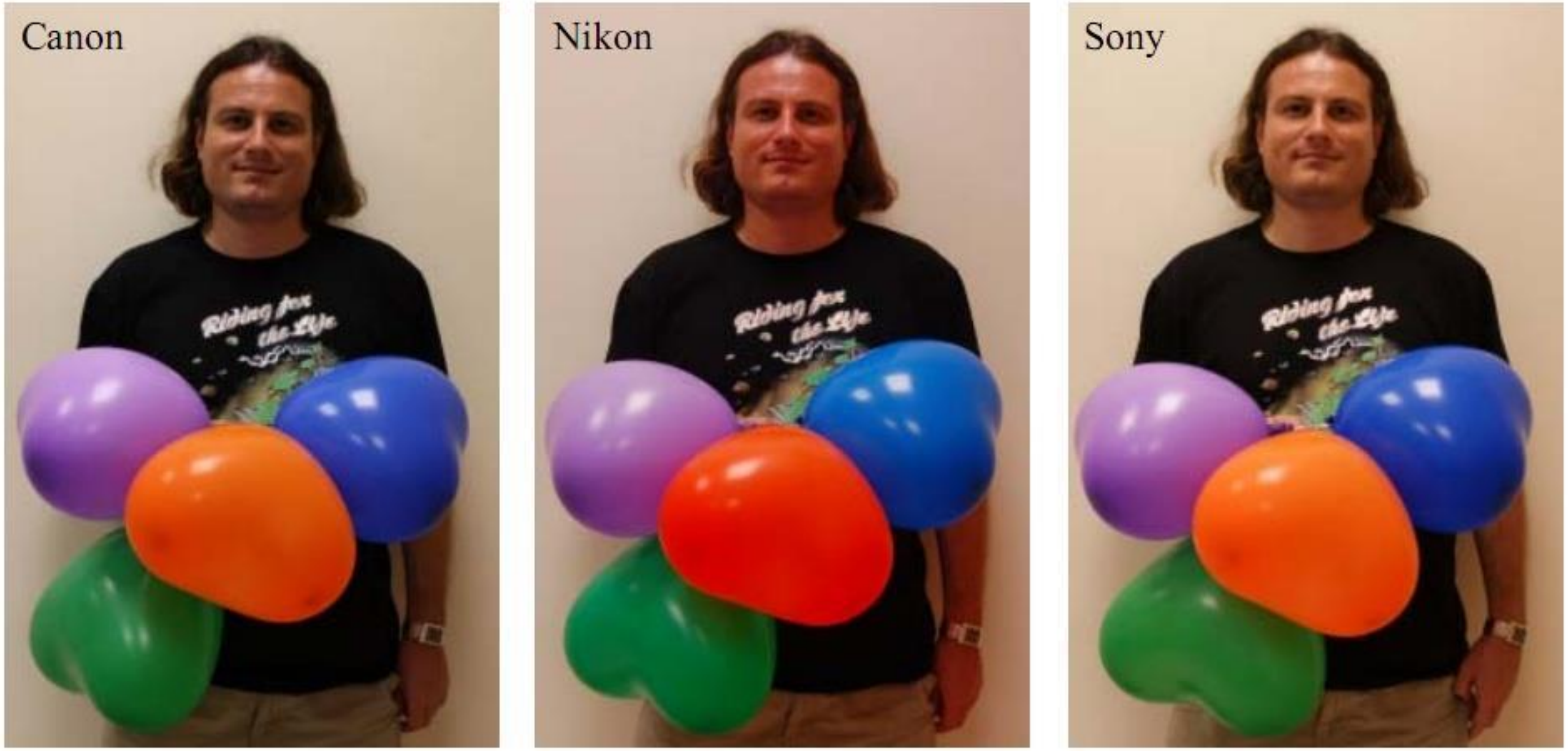
- James Hays (Georgia Tech).
- Fredo Durand (MIT).
- Gordon Wetzstein (Stanford).
- Sylvain Paris (MIT).
- Sam Hasinoff (Google).

Color calibration and homography estimation

Many different spectral sensitivity functions

Each camera has its more or less unique, and most of the time *secret*, SSF.

- Makes it very difficult to correctly reproduce the color of sensor measurements.



Images of the same scene captured using 3 different cameras with identical **sRGB** settings.

Color calibration

Apply linear scaling and translation to RGB vectors in the image:

$$c' = M \cdot c + b$$

transformed RGB vector

original RGB vector

What are the dimensions of each quantity in this equation?

Color calibration

Apply linear scaling and translation to RGB vectors in the image:

$$c' = M \cdot c + b$$

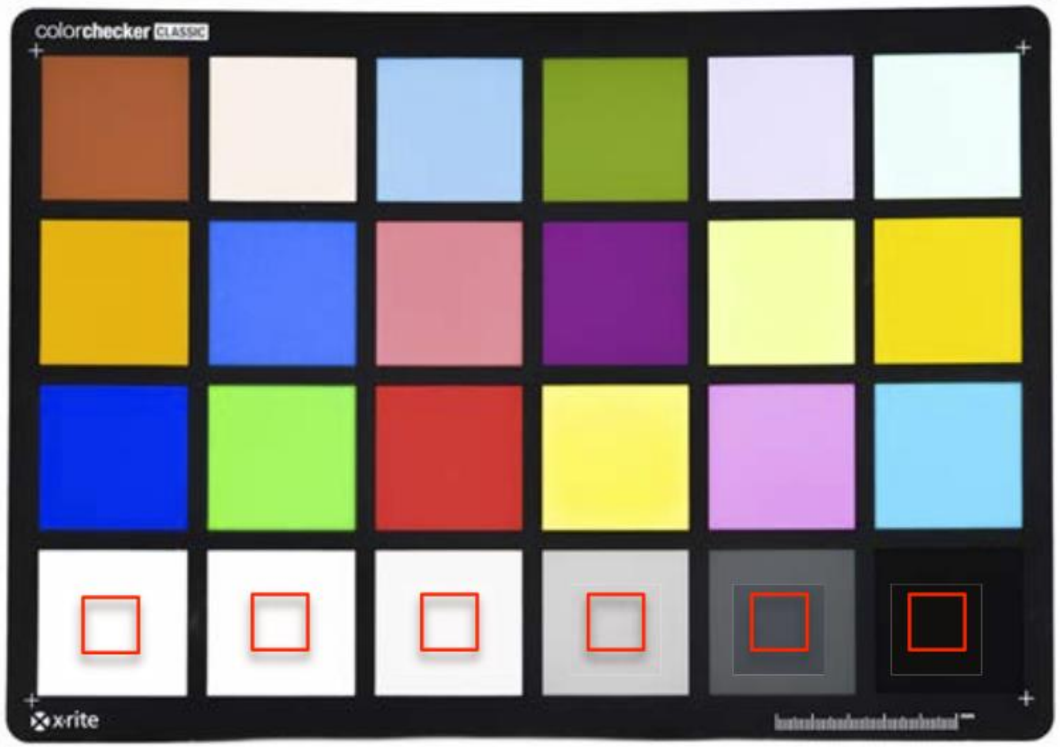
transformed RGB vector

original RGB vector

What are the dimensions of each quantity in this equation?

How do we decide what transformed vectors to map to?

Using (again) a color checker

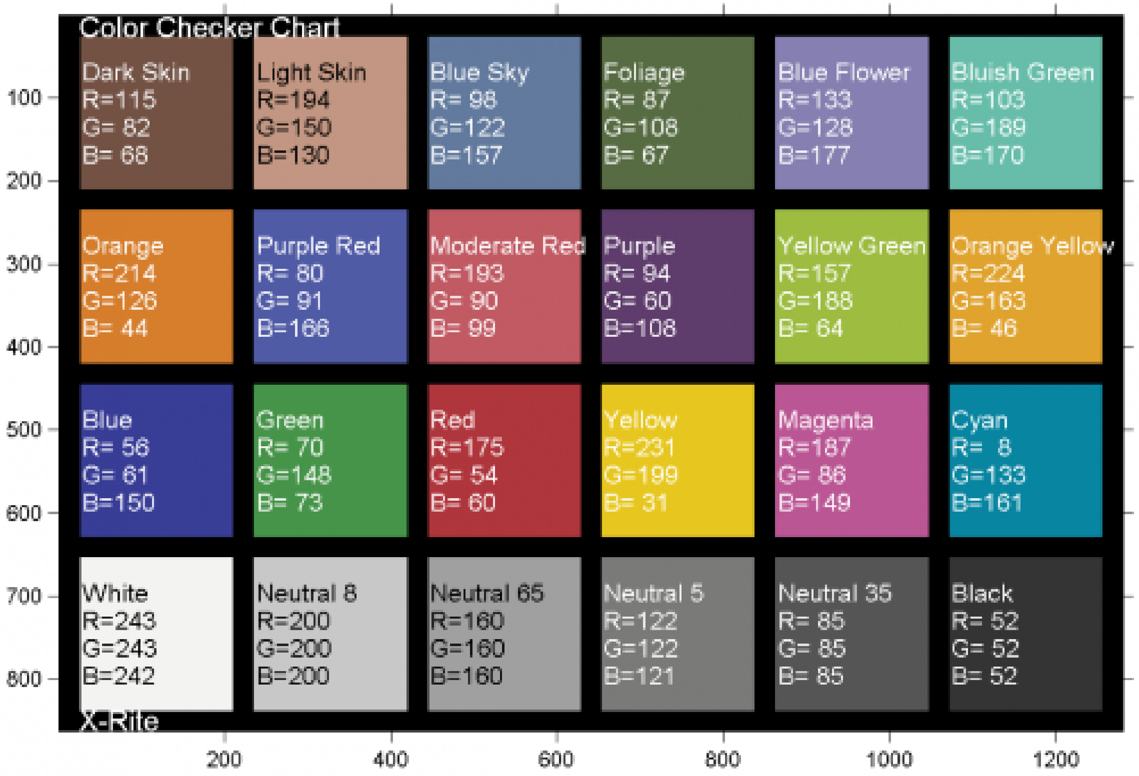


Color patches manufactured to have pre-calibrated XYZ coordinates.

Calibration chart can be used for:

1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row

Using (again) a color checker



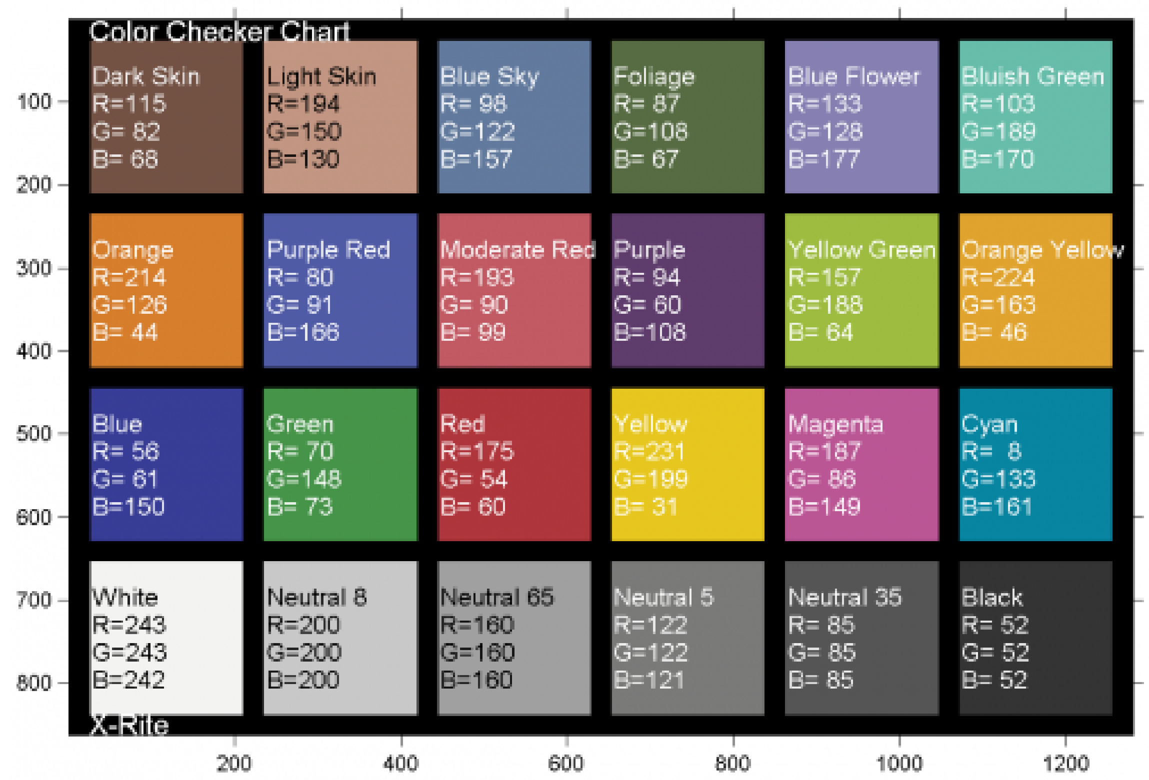
Color patches manufactured to have pre-calibrated XYZ coordinates.

Can we use any color chart image for color calibration?

Calibration chart can be used for:

1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row

Using (again) a color checker



Color patches manufactured to have pre-calibrated XYZ coordinates.

Can we use any color chart image for color calibration?

- It needs to be a *linear* image!
- Do radiometric calibration first.

Calibration chart can be used for:

1. color calibration
2. radiometric calibration (i.e., response curve) using the bottom row

Color calibration

Apply linear scaling and translation to RGB vectors in the image:

$$c' = M \cdot c + b$$

transformed RGB vector

original RGB vector

What are the dimensions of each quantity in this equation?

How do we decide what transformed vectors to map to?

How do we solve for matrix M and vector b?

Color calibration

Apply linear scaling and translation to RGB vectors in the image:

$$\begin{bmatrix} c' \\ 1 \end{bmatrix} = \begin{bmatrix} M & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c \\ 1 \end{bmatrix}$$

Color calibration

Apply linear scaling and translation to RGB vectors in the image:

$$\underbrace{\begin{bmatrix} c' \\ 1 \end{bmatrix}}_{C'} = \underbrace{\begin{bmatrix} M & b \\ 0 & 1 \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} c \\ 1 \end{bmatrix}}_{C}$$

Color calibration

Apply a homography to homogeneous RGB vectors in the image:

$$C' = H \cdot C$$

homogeneous
transformed RGB vector

homogeneous
original RGB vector

How do we solve for a homography transformation?

Determining the homography matrix

Write out linear equation for each color vector correspondence:

$$C' = H \cdot C \quad \text{or} \quad \begin{bmatrix} r' \\ g' \\ b' \\ 1 \end{bmatrix} = a \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \\ h_5 & h_6 & h_7 & h_8 \\ h_9 & h_{10} & h_{11} & h_{12} \\ h_{13} & h_{14} & h_{15} & h_{13} \end{bmatrix} \begin{bmatrix} r \\ g \\ b \\ 1 \end{bmatrix}$$

Determining the homography matrix

Write out linear equation for each color vector correspondence:

$$C' = H \cdot C \qquad \text{or} \qquad \begin{bmatrix} r' \\ g' \\ b' \\ 1 \end{bmatrix} = a \begin{bmatrix} h_1 & h_2 & h_3 & h_4 \\ h_5 & h_6 & h_7 & h_8 \\ h_9 & h_{10} & h_{11} & h_{12} \\ h_{13} & h_{14} & h_{15} & h_{13} \end{bmatrix} \begin{bmatrix} r \\ g \\ b \\ 1 \end{bmatrix}$$

Expand matrix multiplication:

$$r' = a(h_1 r + h_2 g + h_3 b + h_4)$$
$$g' = a(h_5 r + h_6 g + h_7 b + h_8)$$
$$b' = a(h_9 r + h_{10} g + h_{11} b + h_{12})$$
$$1 = a(h_{13} r + h_{14} g + h_{15} b + h_{16})$$

Determining the homography matrix

Divide out unknown scale factor:

$$r'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_1 r + h_2 g + h_3 b + h_4)$$
$$g'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_5 r + h_6 g + h_7 b + h_8)$$
$$b'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_9 r + h_{10} g + h_{11} b + h_{12})$$

Determining the homography matrix

Divide out unknown scale factor:

$$r'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_1 r + h_2 g + h_3 b + h_4)$$
$$g'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_5 r + h_6 g + h_7 b + h_8)$$
$$b'(h_{13}r + h_{14}g + h_{15}b + h_{16}) = (h_9 r + h_{10} g + h_{11} b + h_{12})$$

Rearrange as a linear constraint on entries of H:

$$r'rh_{13} + r'gh_{14} + r'bh_{15} + r'h_{16} - rh_1 - gh_2 - bh_3 - h_4 = 0$$
$$g'rh_{13} + g'gh_{14} + g'bh_{15} + g'h_{16} - rh_5 - gh_6 - bh_7 - h_8 = 0$$
$$b'rh_{13} + b'gh_{14} + b'bh_{15} + b'h_{16} - rh_9 - gh_{10} - bh_{11} - h_{12} = 0$$

Determining the homography matrix

Re-write in matrix form:

$$\mathbf{A}_i \boldsymbol{h} = \mathbf{0}$$

What are the dimensions of each variable in this system?

How many equations from one color vector correspondence?

How many color vector correspondences do we need?

Determining the homography matrix

Re-write in matrix form:

$$\mathbf{A}_i \boldsymbol{h} = \mathbf{0}$$

Stack together constraints from additional color vector correspondences row-wise:

$$\mathbf{A} \boldsymbol{h} = \mathbf{0}$$

Homogeneous linear least squares system.

- How do we solve such systems?

Determining the homography matrix

Re-write in matrix form:

$$\mathbf{A}_i \boldsymbol{h} = \mathbf{0}$$

Stack together constraints from additional color vector correspondences row-wise:

$$\mathbf{A} \boldsymbol{h} = \mathbf{0}$$

Homogeneous linear least squares system.

- How do we solve such systems? $\rightarrow$ Use singular value decomposition (SVD)

General form of total least squares

(**Warning:** change of notation. x is a vector of parameters!)

$$E_{\mathrm{TLS}} = \sum_i (\boldsymbol{a}_i \boldsymbol{x})^2$$

$$= \|\mathbf{A}\boldsymbol{x}\|^2 \qquad \text{(matrix form)}$$

$$\|\boldsymbol{x}\|^2 = 1 \qquad \text{constraint}$$

minimize $\|\mathbf{A}\boldsymbol{x}\|^2$

subject to $\|\boldsymbol{x}\|^2 = 1$

$\Rightarrow$

minimize $\dfrac{\|\mathbf{A}\boldsymbol{x}\|^2}{\|\boldsymbol{x}\|^2}$ (Rayleigh quotient)

Solution is the eigenvector corresponding to smallest eigenvalue of

$$\mathbf{A}^\top \mathbf{A}$$

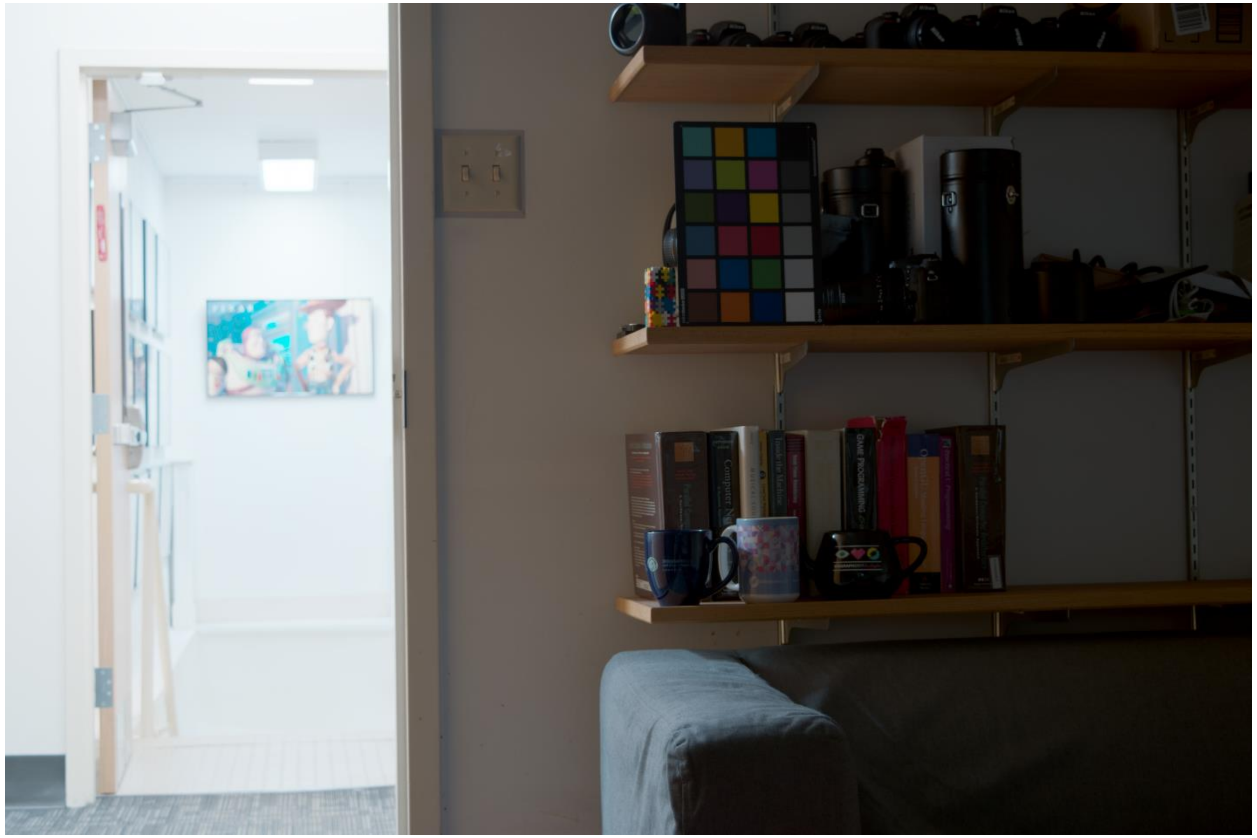
(equivalent)

Solution is the column of **V** corresponding to smallest singular value

$$\mathbf{A} = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^\top$$

An example

original

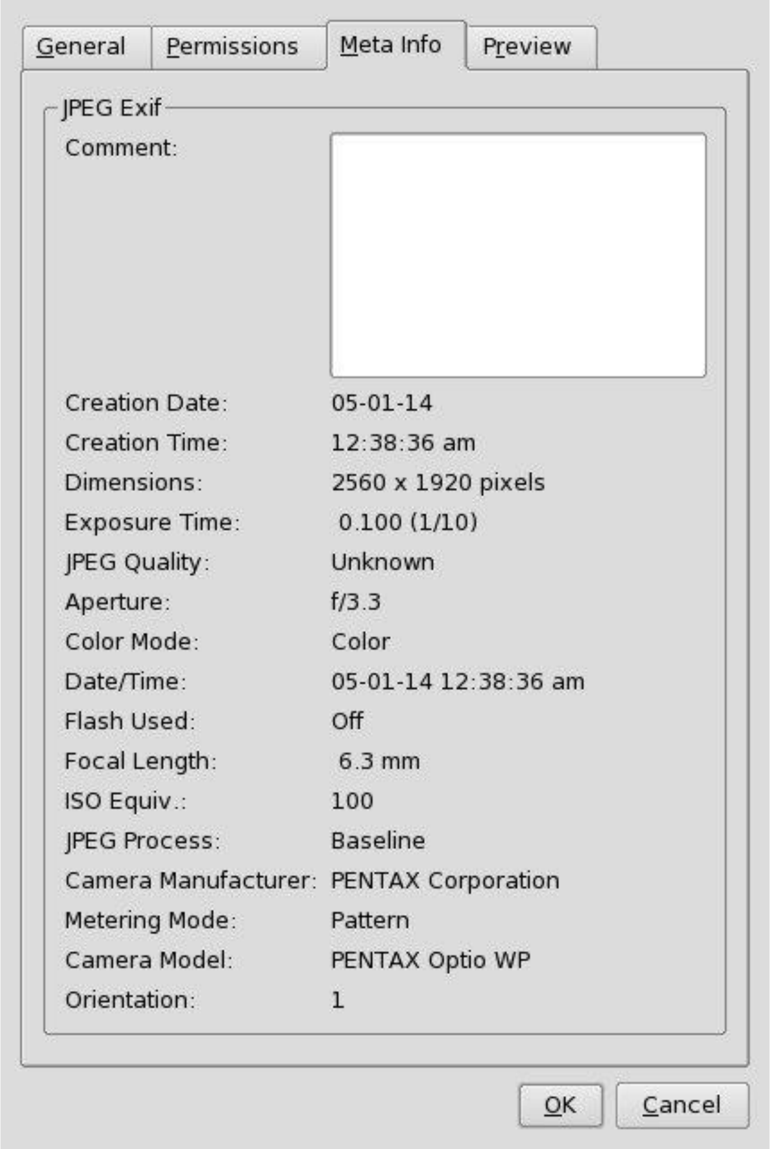


color-corrected

Quick note

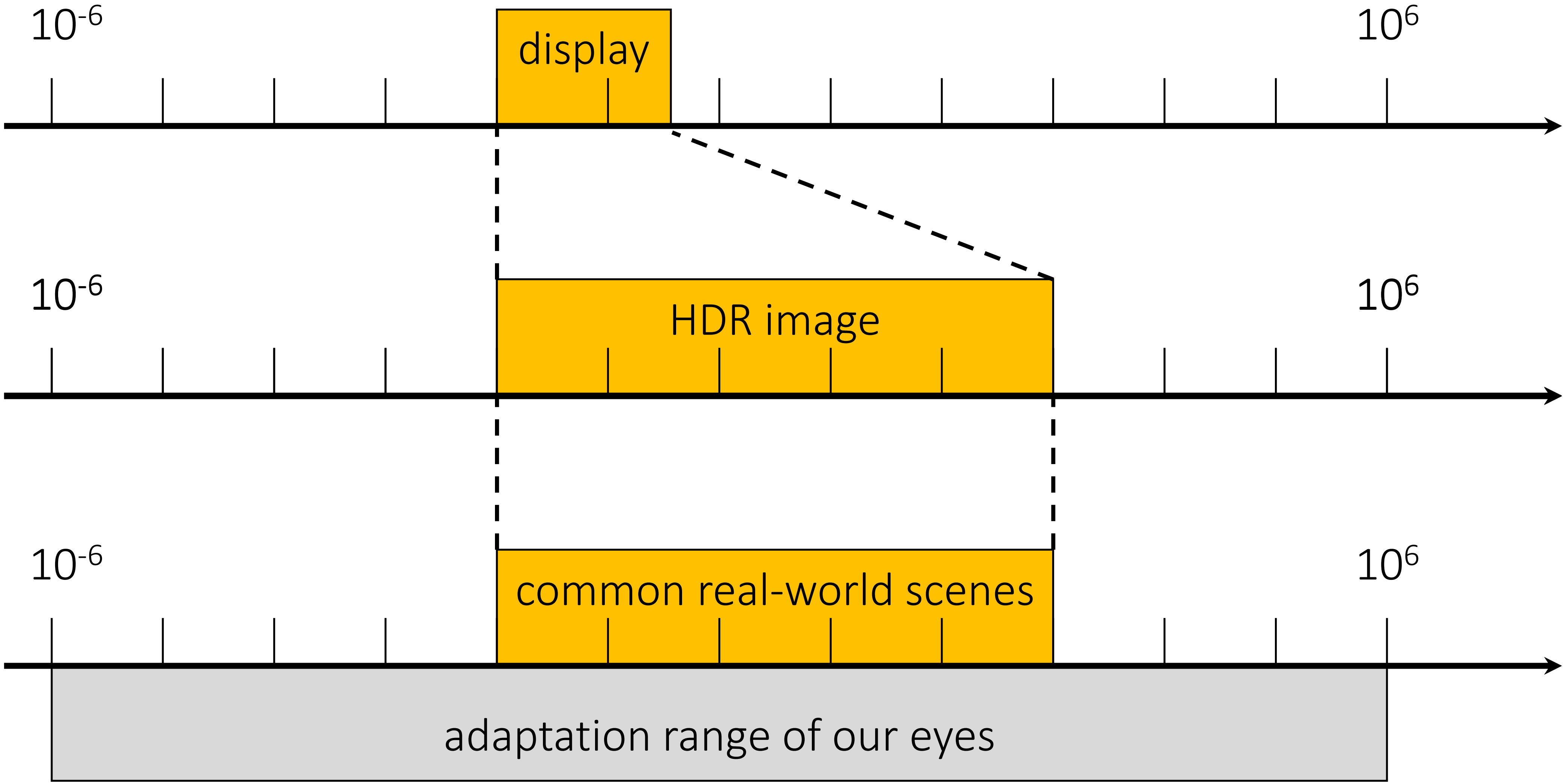
If you cannot do calibration, take a look at the image's EXIF data (if available).

Often contains information about tone reproduction curve <u>and color space</u>.



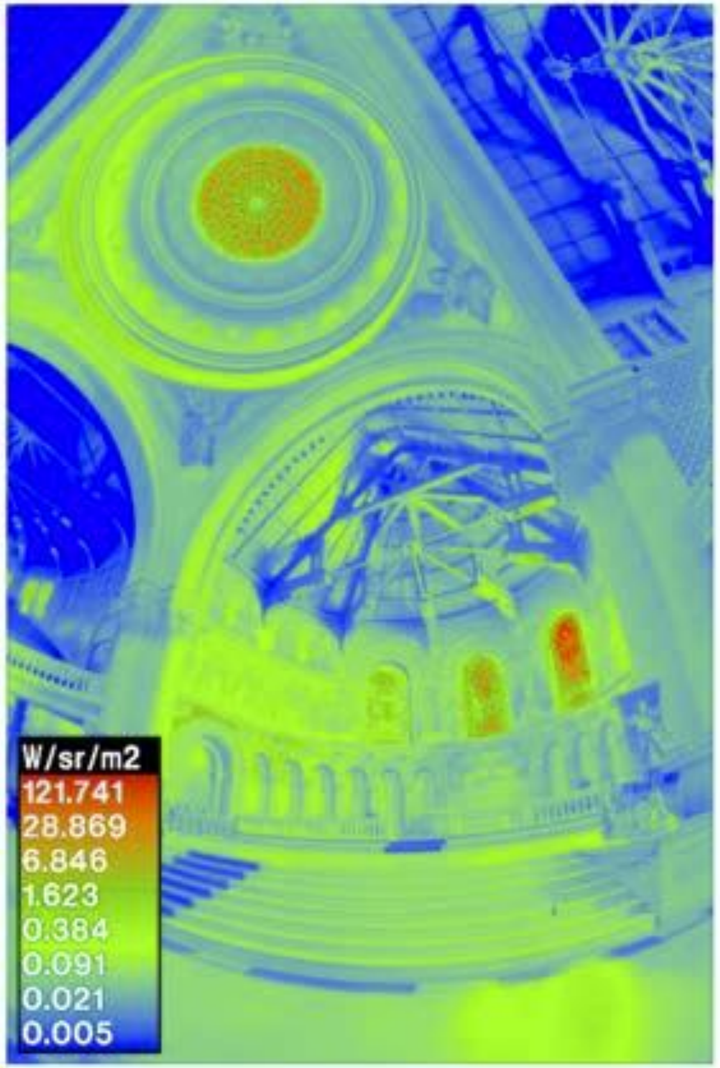
Tonemapping

How do we display our HDR images?



Linear scaling

Scale image so that maximum value equals 1

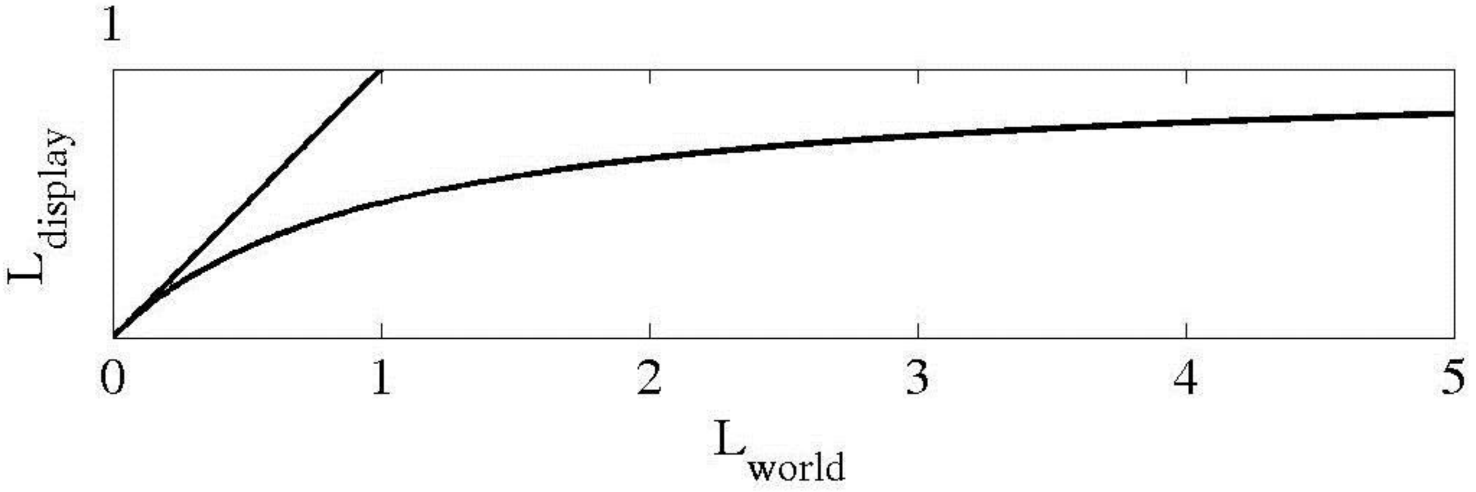


Can you think of something better?

Photographic tonemapping

Apply the same non-linear scaling to all pixels in the image so that:

- Bring everything within range → asymptote to 1
- Leave dark areas alone → slope = 1 near 0



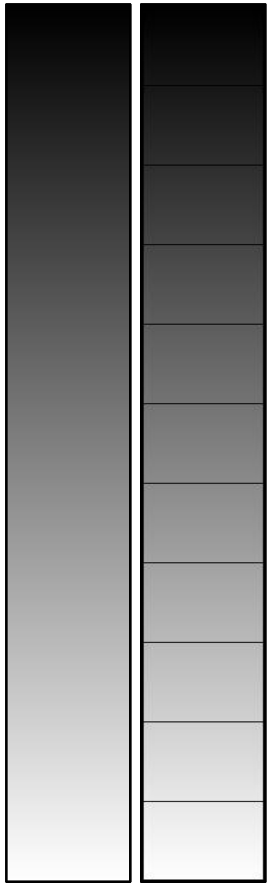
$$I_{display} = \frac{I_{HDR}}{1 + I_{HDR}}$$

(exact formula more complicated)

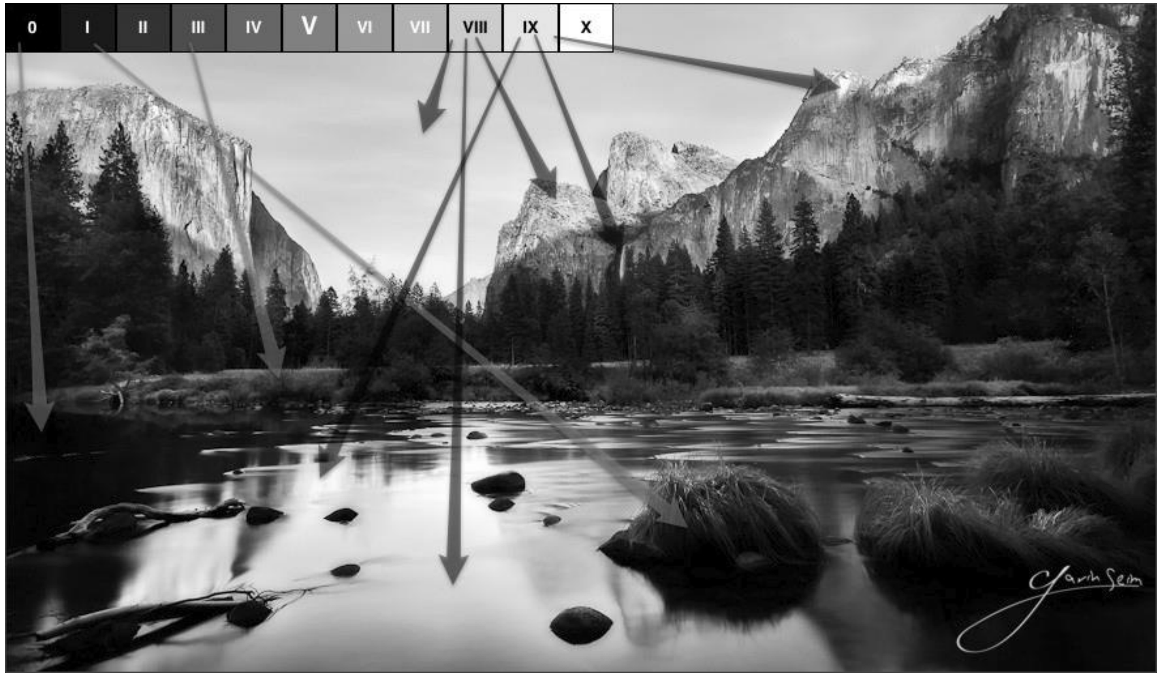
- Photographic because designed to approximate film zone system.
- Perceptually motivated, as it approximates our eye's response curve.

What is the zone system?

- Technique formulated by Ansel Adams for film development.
- Still used with digital photography.



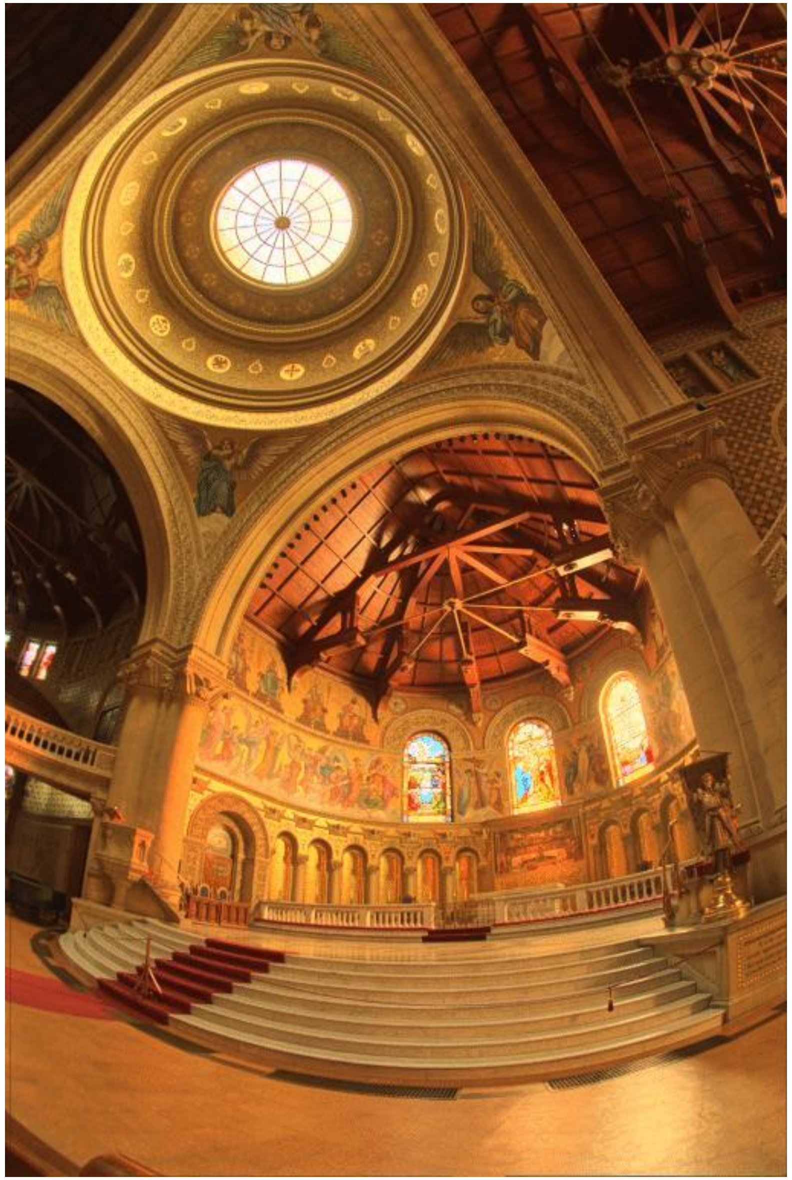
Zone	Description
0	Pure black
I	Near black, with slight tonality but no texture
II	Textured black; the darkest part of the image in which slight detail is recorded
III	Average dark materials and low values showing adequate texture
IV	Average dark foliage, dark stone, or landscape shadows
V	Middle gray: clear north sky; dark skin, average weathered wood
VI	Average Caucasian skin; light stone; shadows on snow in sunlit landscapes
VII	Very light skin; shadows in snow with acute side lighting
VIII	Lightest tone with texture: textured snow
IX	Slight tone without texture; glaring snow
X	Pure white: light sources and specular reflections

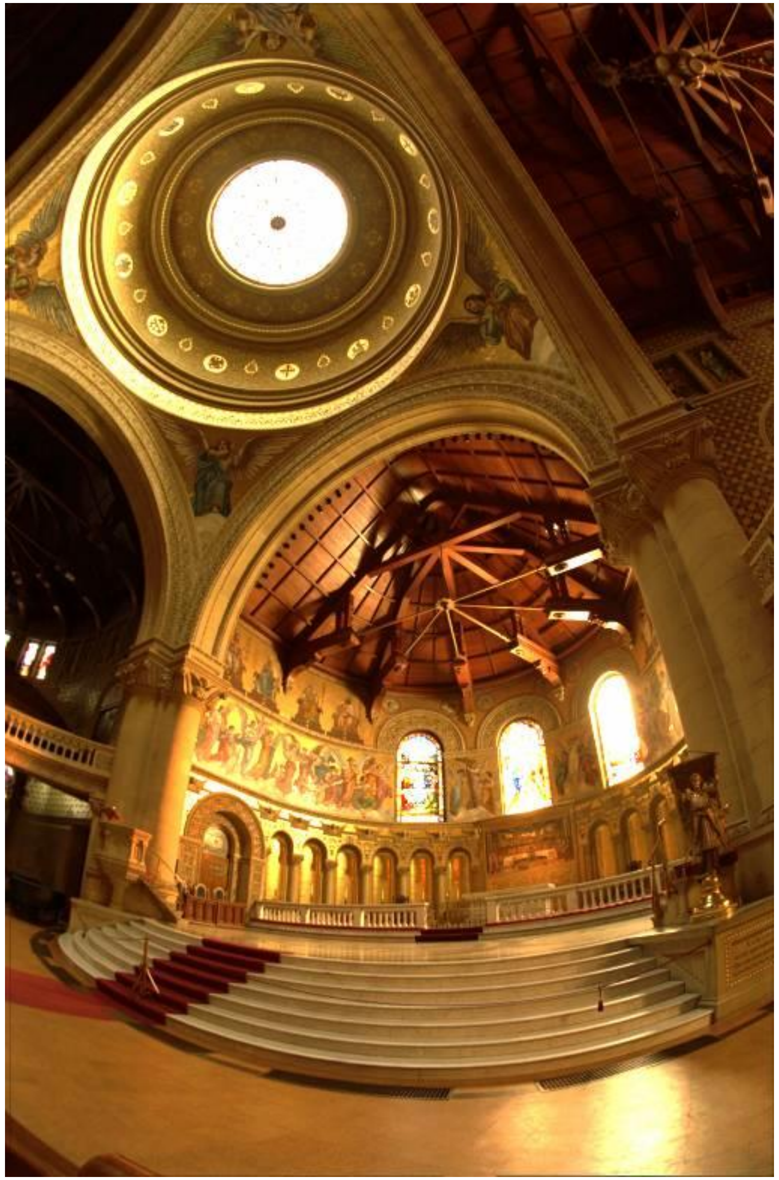


Examples

Examples

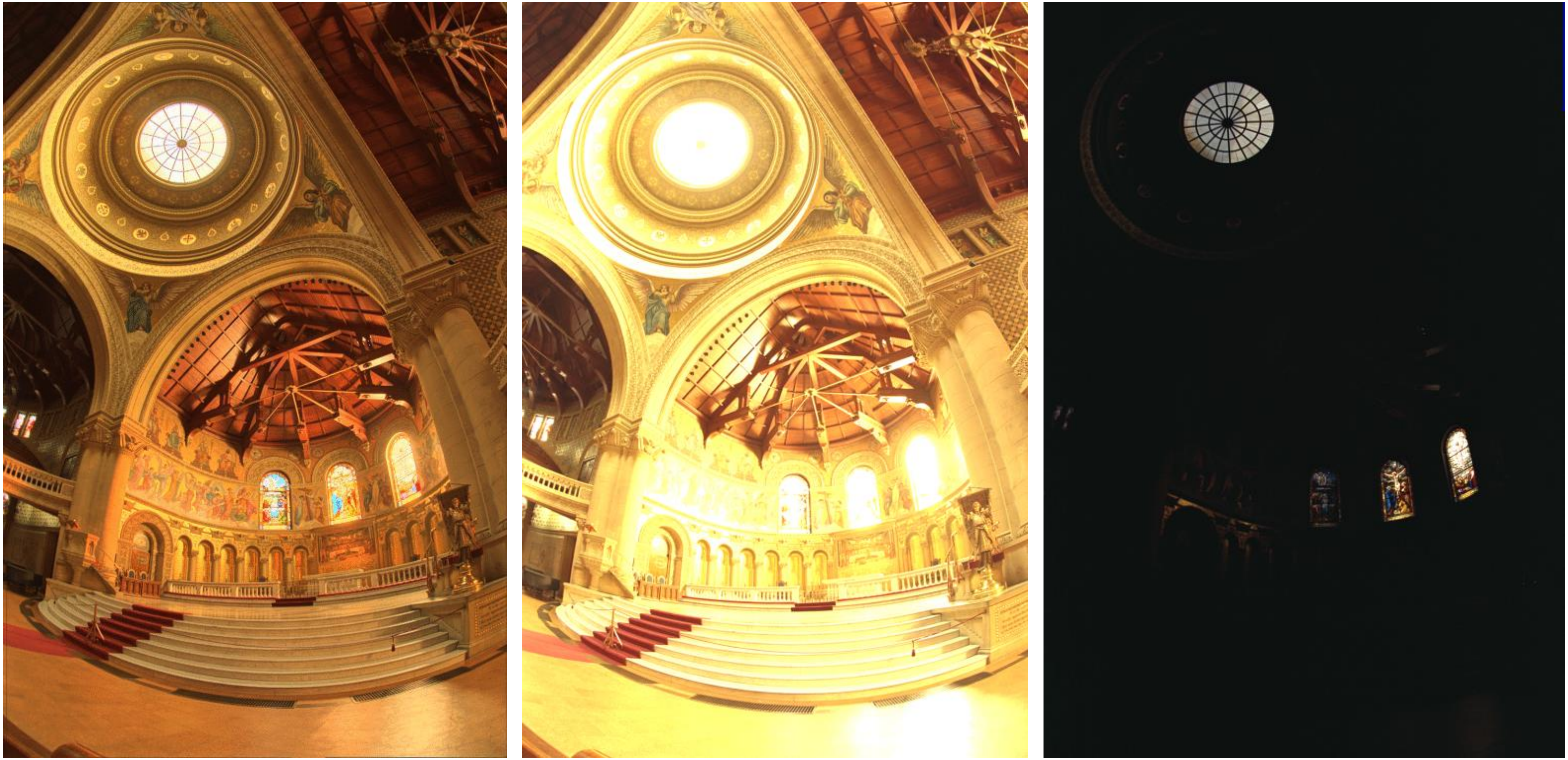
photographic tonemapping





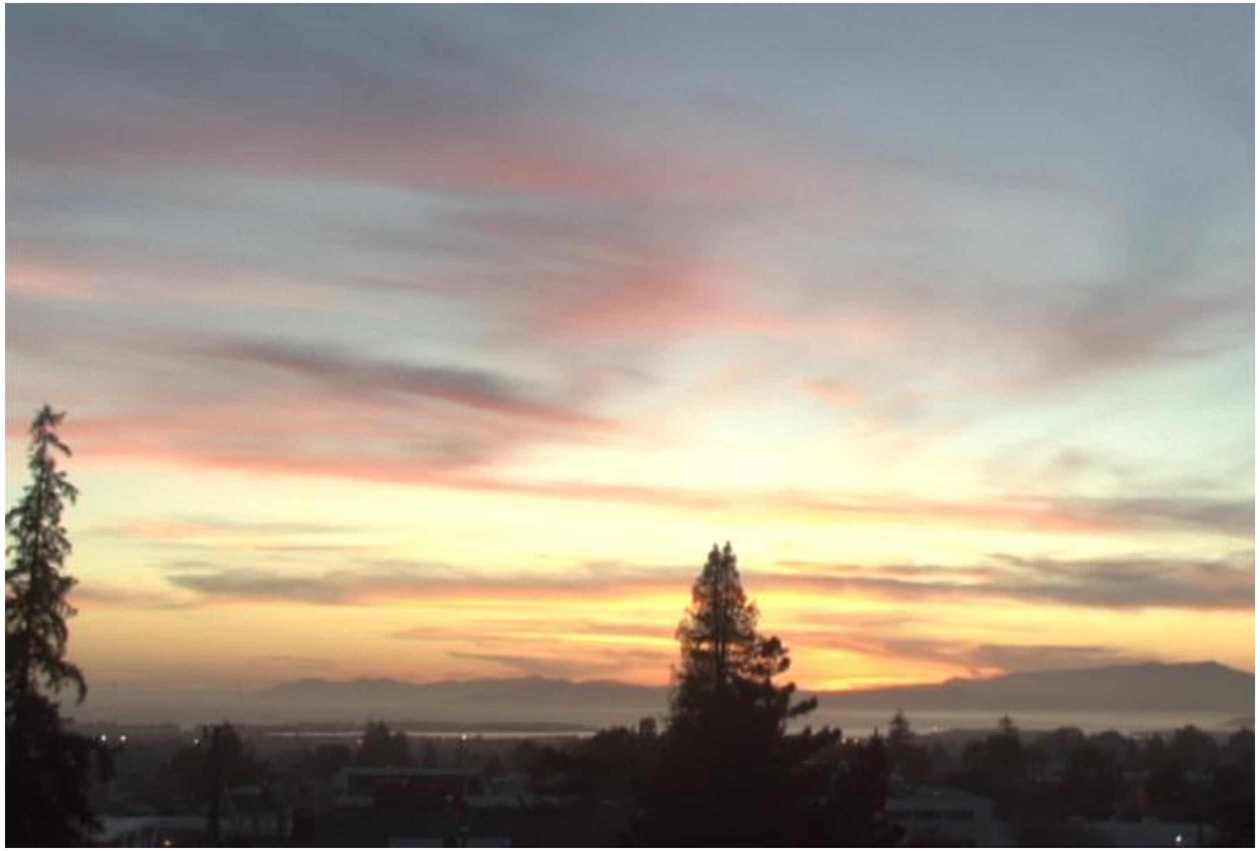
linear scaling (map 10% to 1)

Compare with LDR images



Dealing with color

If we tonemap all channels the same, colors are washed out



Can you think of a way to deal with this?

Intensity-only tonemapping

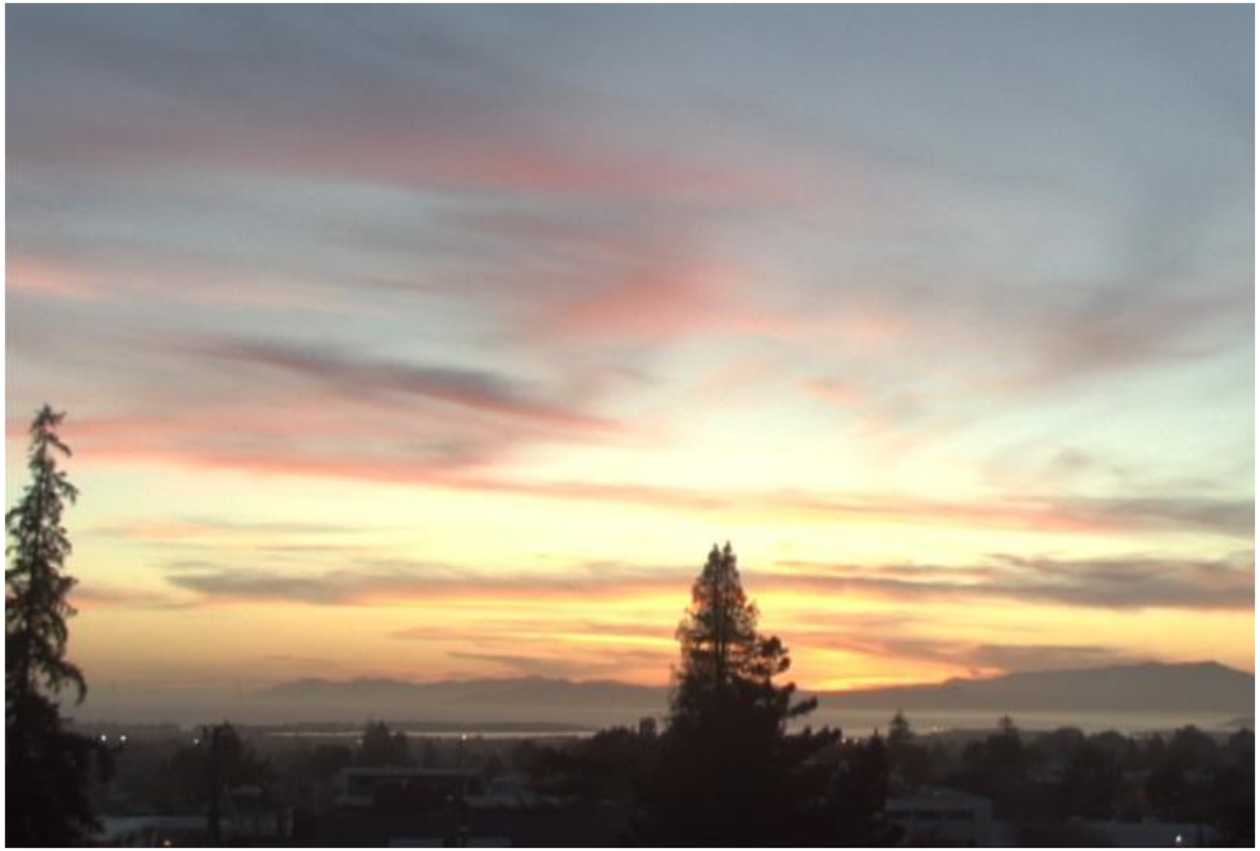
tonemap intensity

leave color the same

How would you implement this?

Comparison

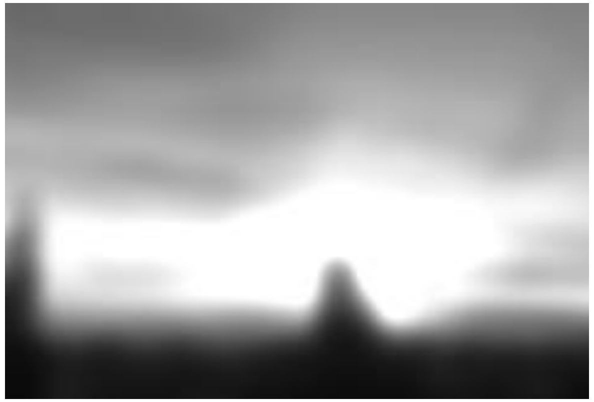
Color now OK, but some details are washed out due to loss of contrast



Can you think of a way to deal with this?

Low-frequency intensity-only tonemapping

tonemap low-frequency intensity component



leave high-frequency intensity component the same

leave color the same

How would you implement this?

Comparison

We got nice color and contrast, but now we’ve run into the halo plague

Can you think of a way to deal with this?

Edge-aware filtering and bilateral filtering

Motivational example

original

Let’s say I want to reduce the amount of detail in this picture. What can I do?

Motivational example

original

Gaussian filtering

What is the problem here?

Motivational example

original

Gaussian filtering

How to smooth out the details in the image without losing the important edges?

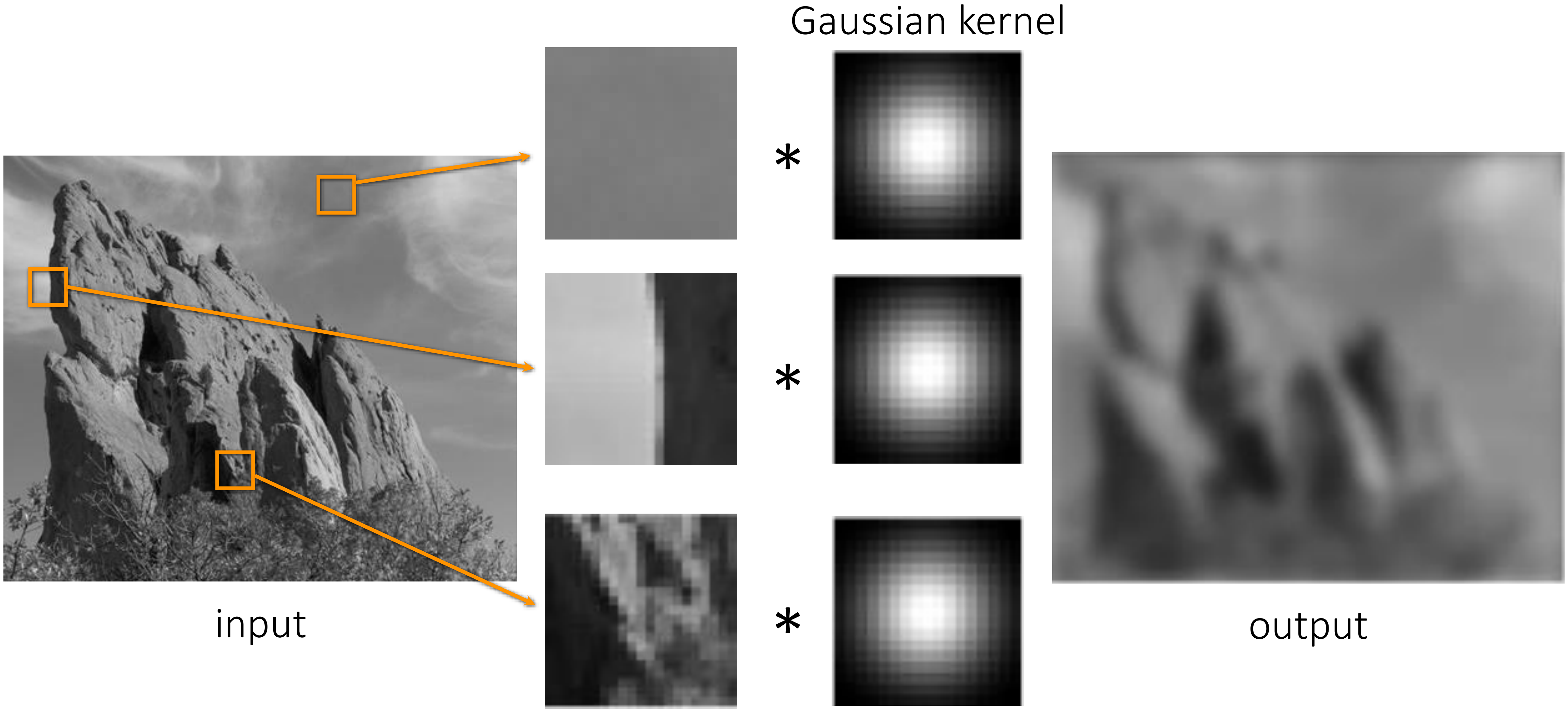
Motivational example

original

Gaussian filtering

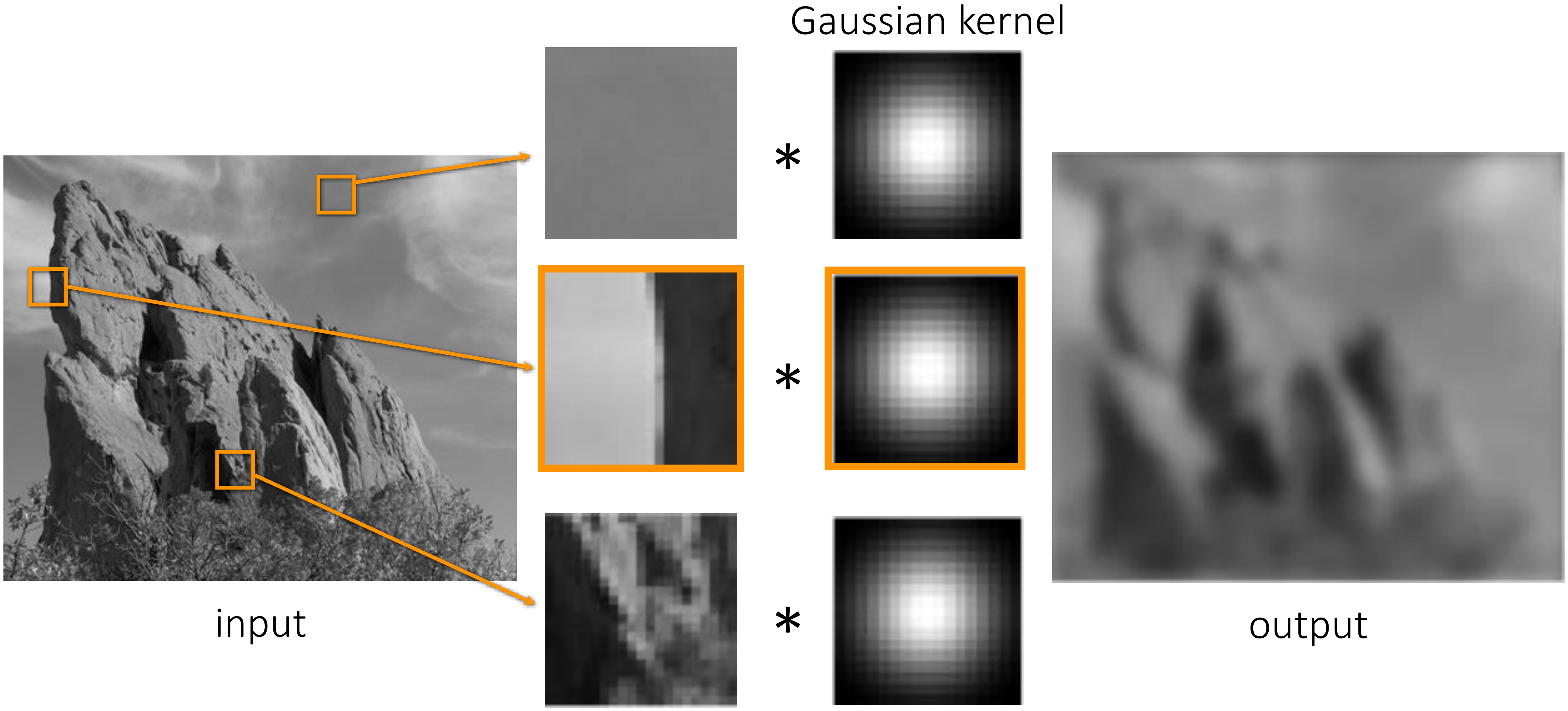
bilateral filtering

The problem with Gaussian filtering



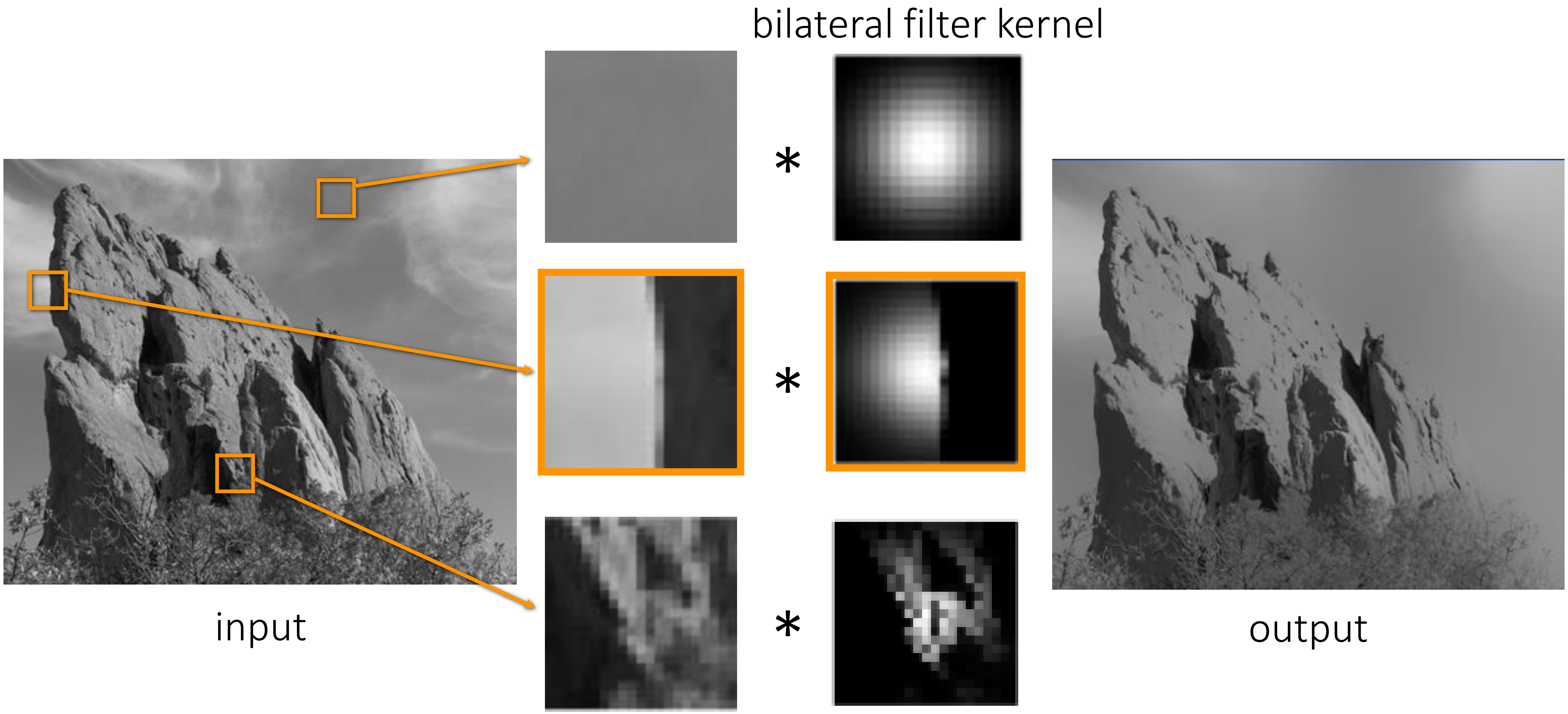
Why is the output so blurry?

The problem with Gaussian filtering



Blur kernel averages across edges

The bilateral filtering solution



Do not blur if there is an edge! How does it do that?

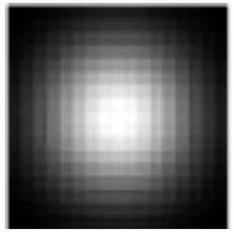
Bilateral filtering

$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

Bilateral filtering

$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} \boxed{g[k,l]} r_{mn}[k,l] f[m+k, n+l]$$

Spatial weighting



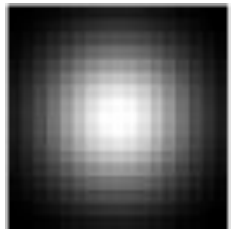
$$\sigma_s$$

Assign a pixel a large weight if: 1) it's nearby

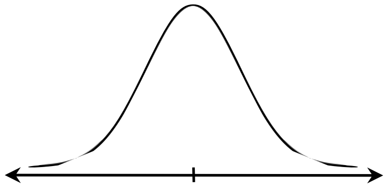
Bilateral filtering

$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

Spatial weighting



Intensity range weighting



$x = f[m,n] - f[m+k, n+l]$

σ_s

σ_r

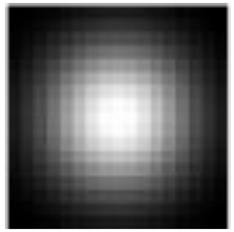
Assign a pixel a large weight if: 1) it's nearby and 2) it looks like me

Bilateral filtering

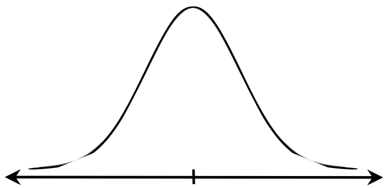
$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l]\, r_{mn}[k,l]\, f[m+k,n+l]$$

Normalization factor

Spatial weighting



Intensity range weighting



$x = f[m,n] - f[m+k,n+l]$

σ_s σ_r

Assign a pixel a large weight if: 1) it's nearby and 2) it looks like me

Bilateral filtering vs Gaussian filtering

Which is which?

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

Bilateral filtering vs Gaussian filtering

Gaussian filtering

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

Bilateral filtering

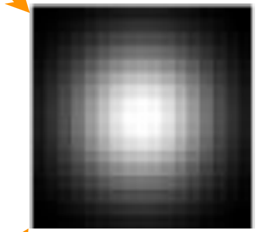
$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

Bilateral filtering vs Gaussian filtering

Gaussian filtering

$$h[m,n] = \sum_{k,l} \boxed{g[k,l]} f[m+k, n+l]$$

σ_s



Spatial weighting: favor *nearby* pixels

Bilateral filtering

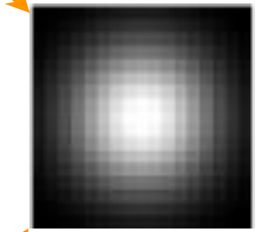
$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} \boxed{g[k,l]} r_{mn}[k,l] f[m+k, n+l]$$

Bilateral filtering vs Gaussian filtering

Gaussian filtering

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

σ_s

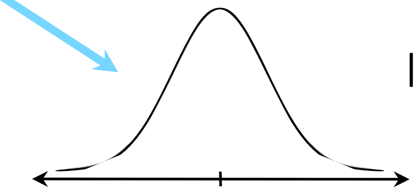


Spatial weighting: favor *nearby* pixels

Bilateral filtering

$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

σ_r



$x = f[m,n] - f[m+k, n+l]$

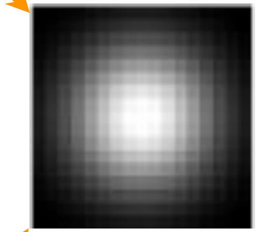
Intensity range weighting: favor *similar* pixels

Bilateral filtering vs Gaussian filtering

Gaussian filtering

$$h[m,n] = \sum_{k,l} g[k,l] f[m+k, n+l]$$

σ_s



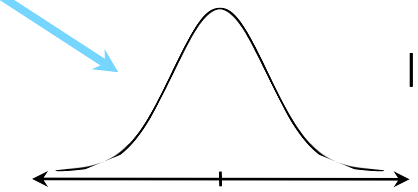
Spatial weighting: favor *nearby* pixels

Bilateral filtering

$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

Normalization factor

σ_r



$x = f[m,n] - f[m+k, n+l]$

Intensity range weighting: favor *similar* pixels

Bilateral filtering vs Gaussian filtering

Gaussian filtering

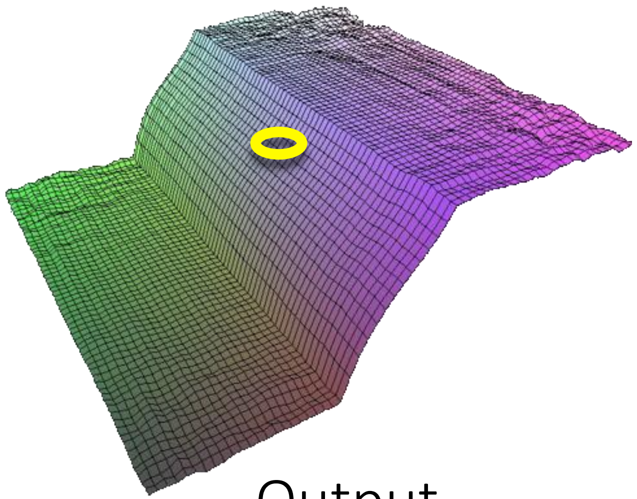
Smooths everything nearby (even edges)
Only depends on *spatial* distance

Bilateral filtering

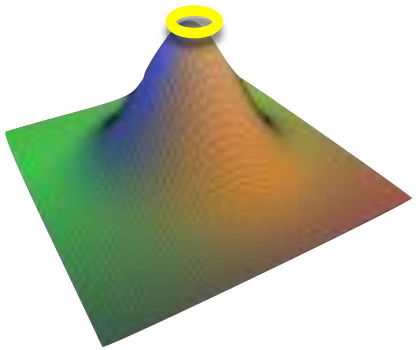
Smooths ‘close’ pixels in space and intensity
Depends on *spatial* and *intensity* distance

Gaussian filtering visualization

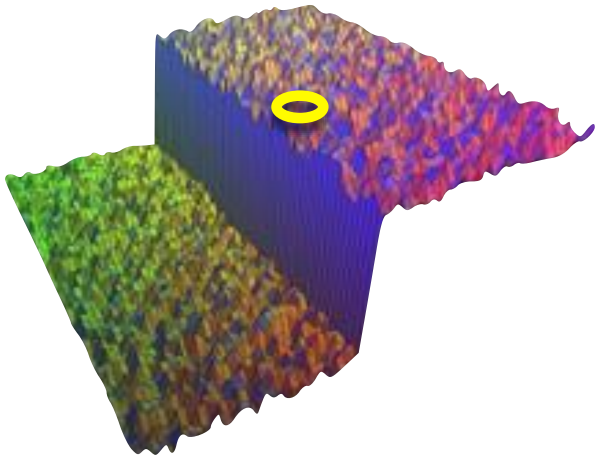
$$h[m,n] = \sum_{k,l} \boxed{g[k,l]} f[m+k, n+l]$$



Output



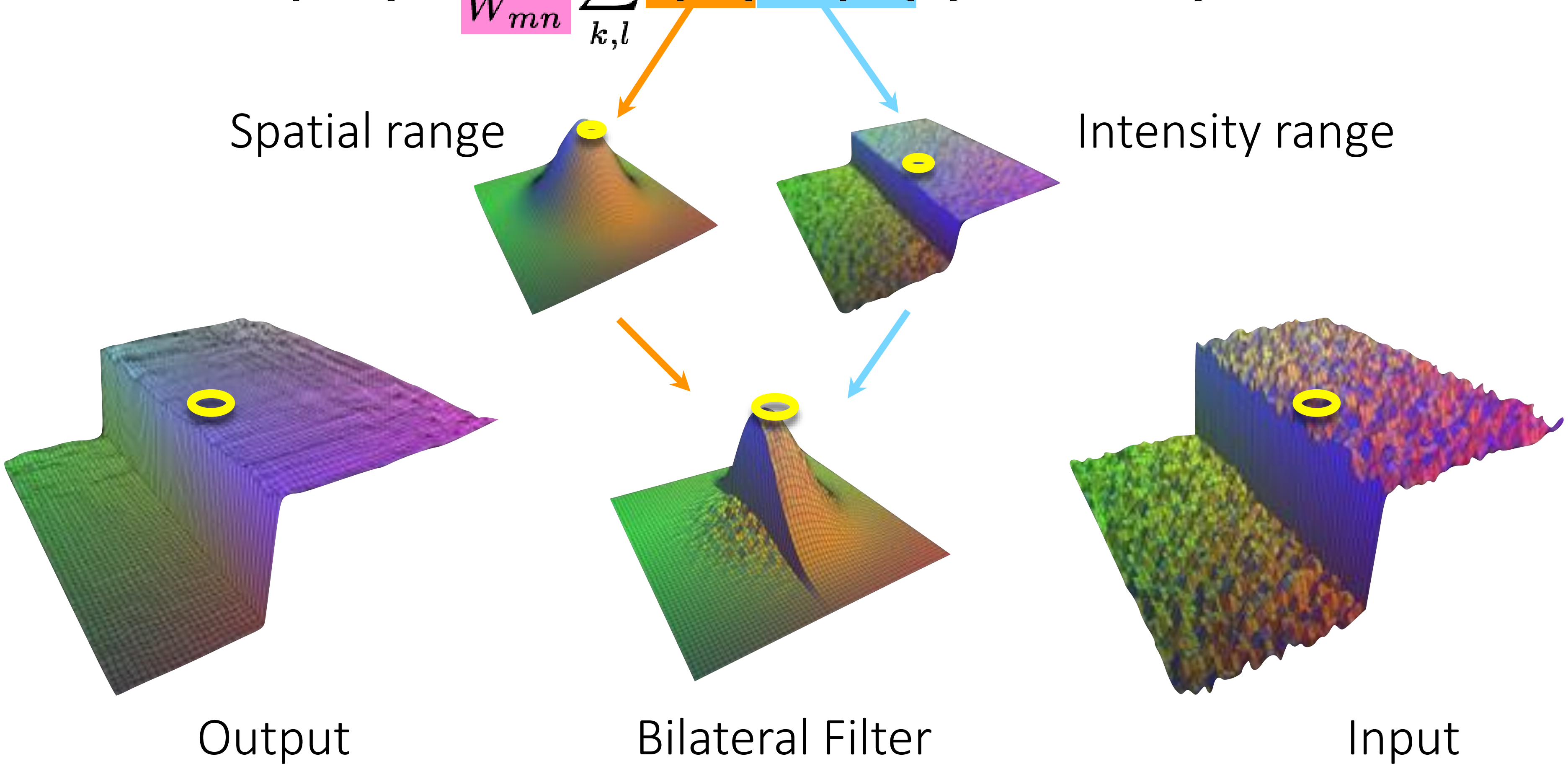
Gaussian Filter



Input

Bilateral filtering visualization

$$h[m,n] = \boxed{\frac{1}{W_{mn}}} \sum_{k,l} \boxed{g[k,l]}\boxed{r_{mn}[k,l]} f[m+k, n+l]$$



Exploring the bilateral filter parameter space

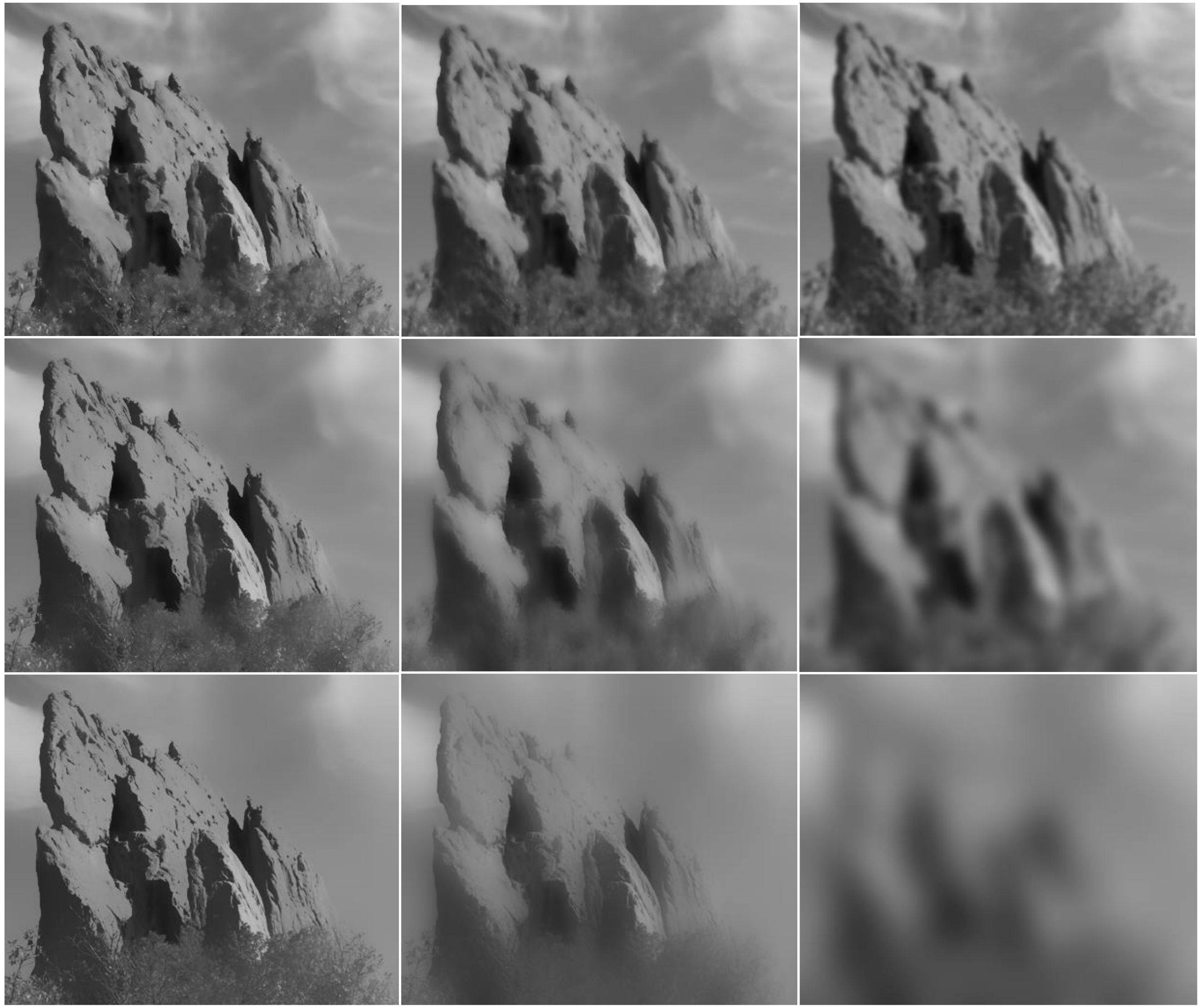
input

$\sigma_r = 0.1$ | $\sigma_r = 0.25$ | $\sigma_r = \infty$ (Gaussian blur)

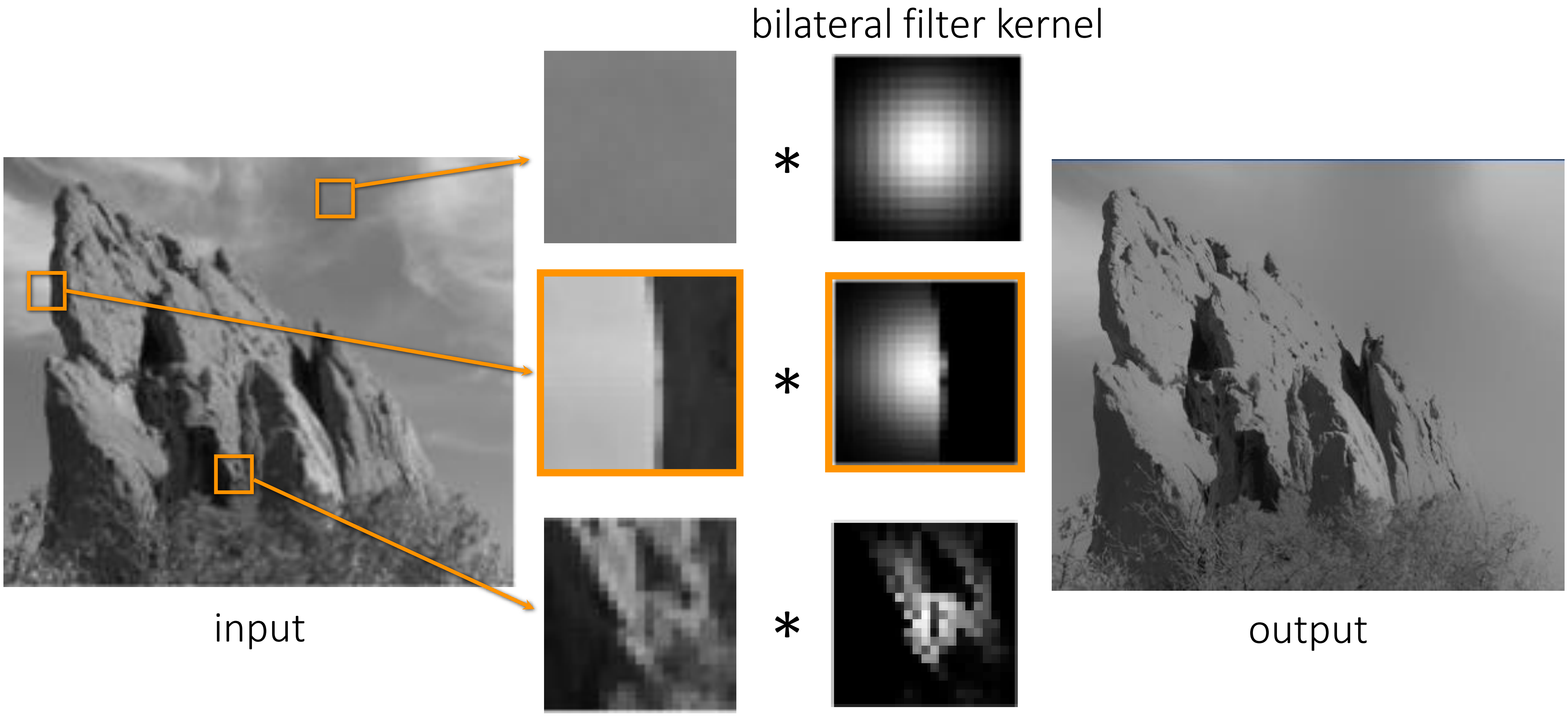
$\sigma_s = 2$

$\sigma_s = 6$

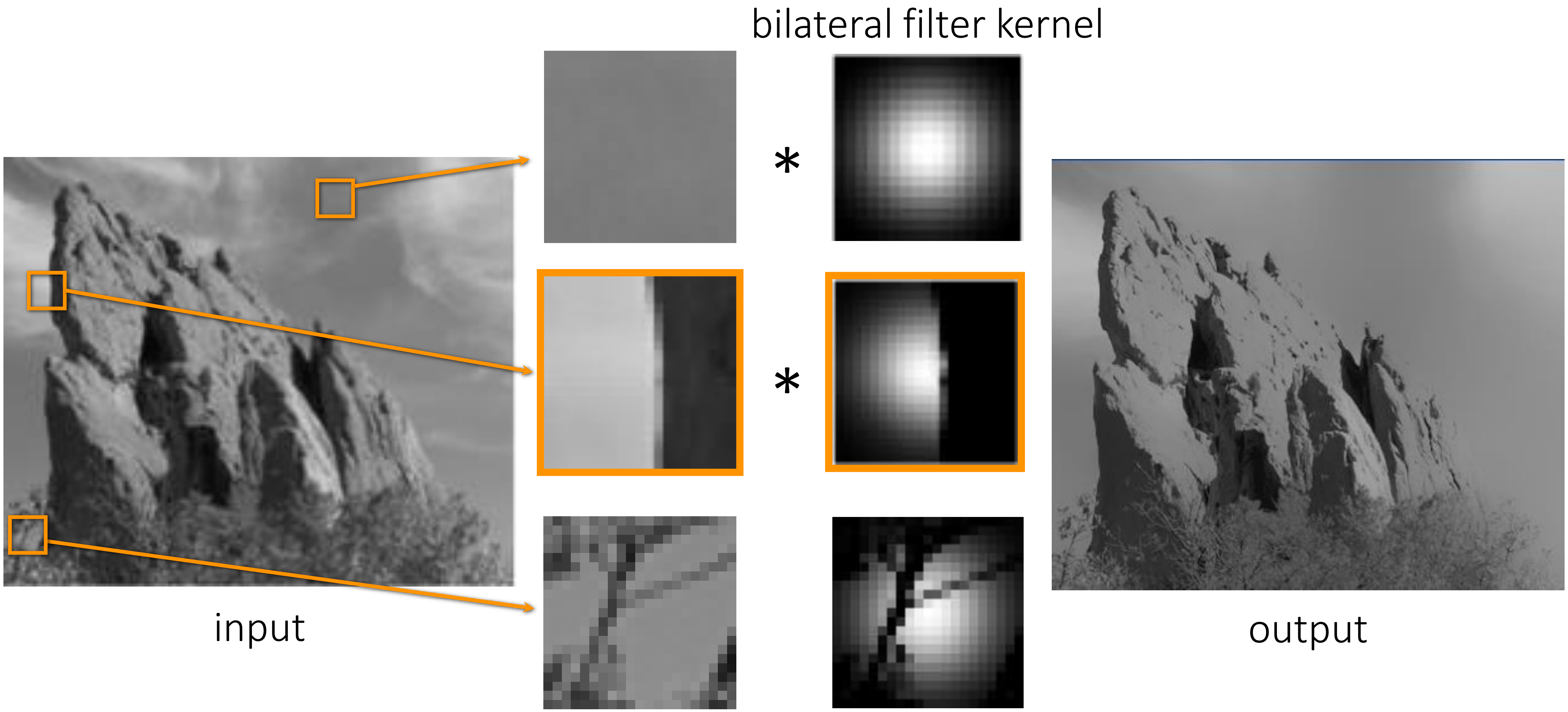
$\sigma_s = 18$



Does the bilateral filter respect all edges?

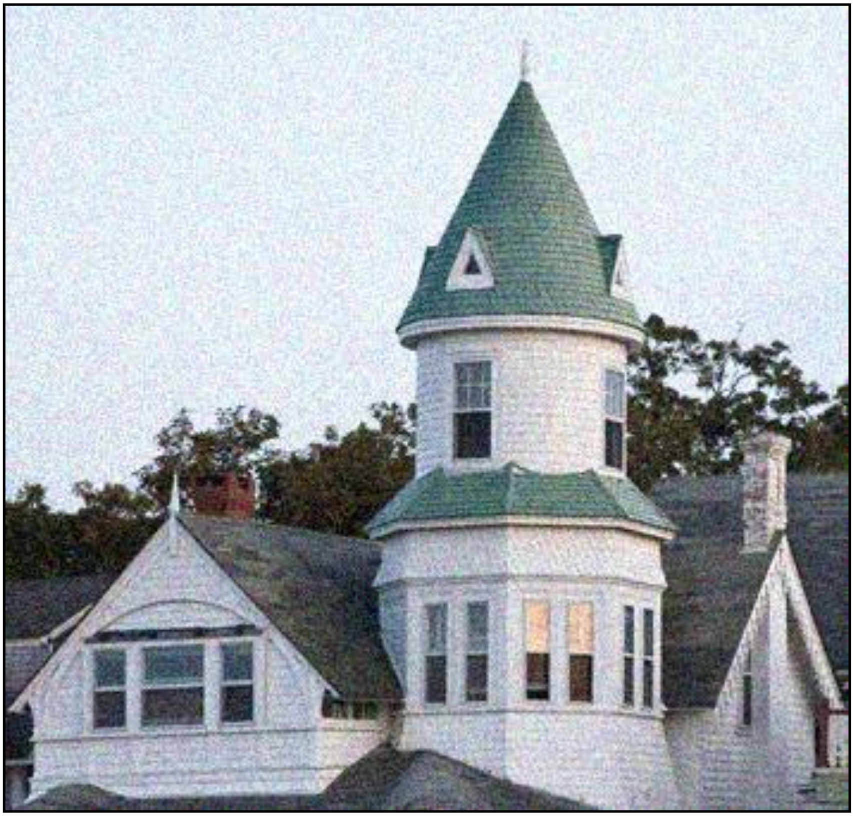


Does the bilateral filter respect all edges?



Bilateral filter crosses (and blurs) thin edges.

Denoising



noisy input

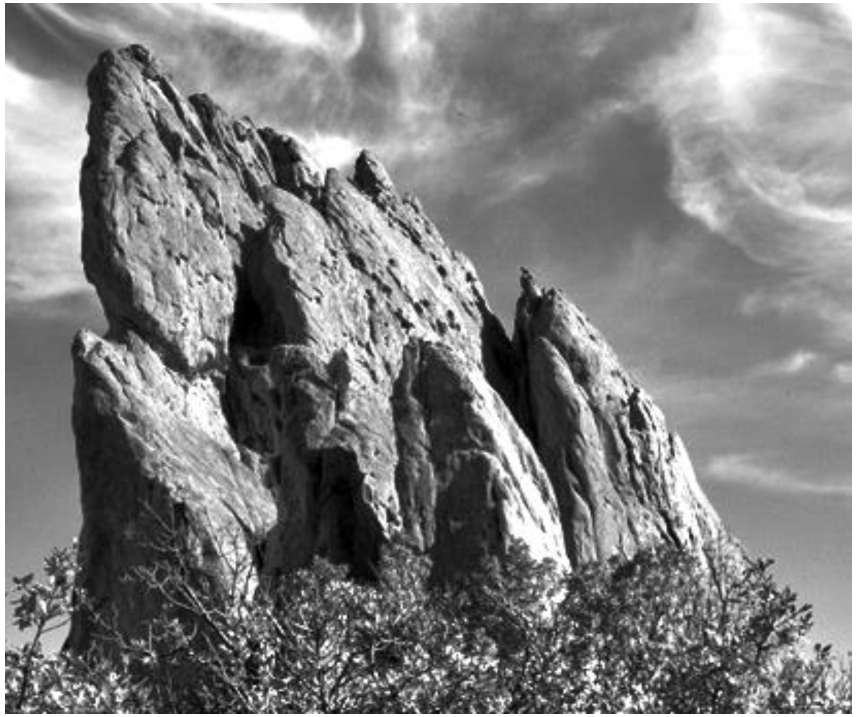
bilateral filtering

median filtering

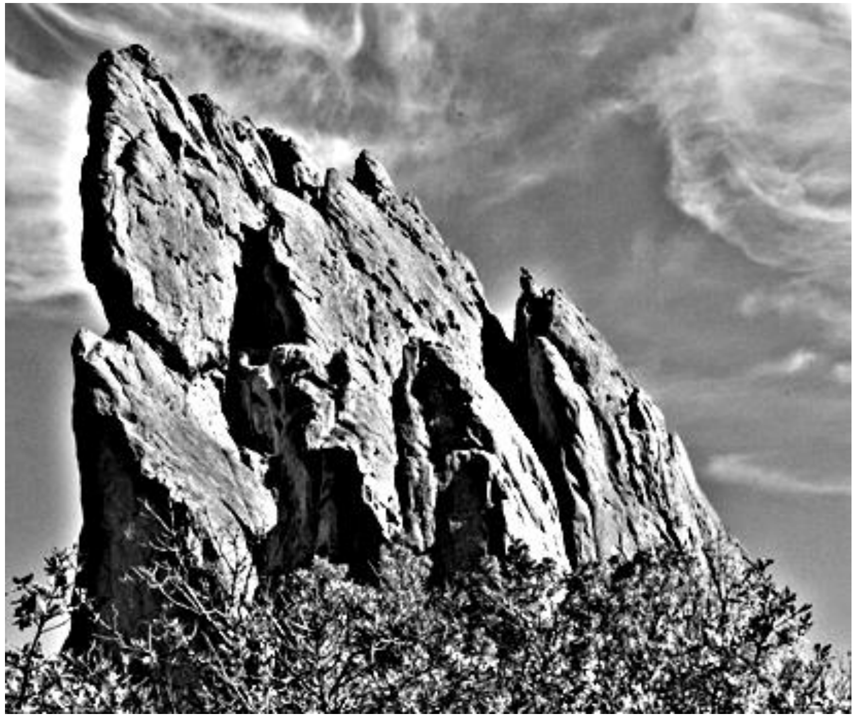
Contrast enhancement

How would you use Gaussian or bilateral filtering for sharpening?

input



sharpening based on bilateral filtering



sharpening based on Gaussian filtering

Photo retouching

Photo retouching

original

digital pore removal (aka bilateral filtering)

Before

After

Close-up comparison

original

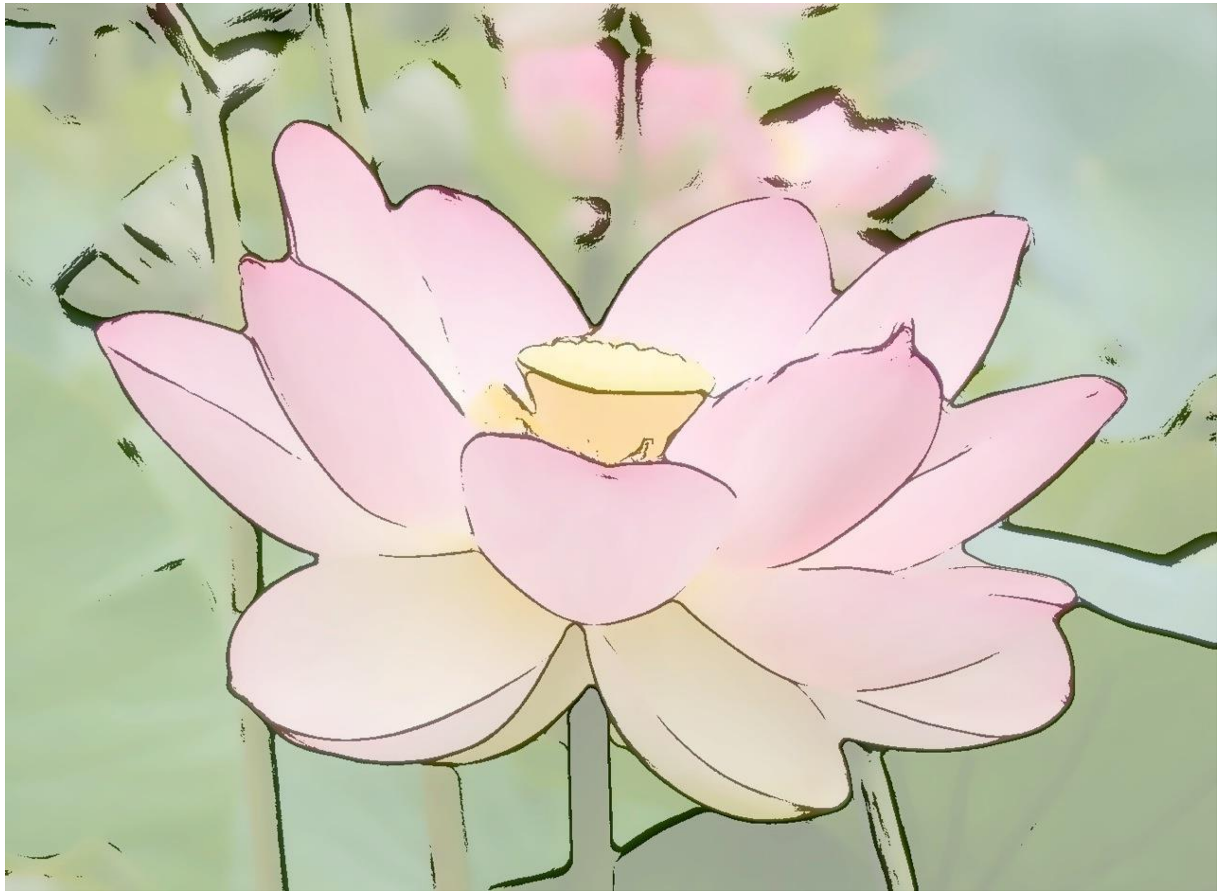
digital pore removal (aka bilateral filtering)

Cartoonization

input

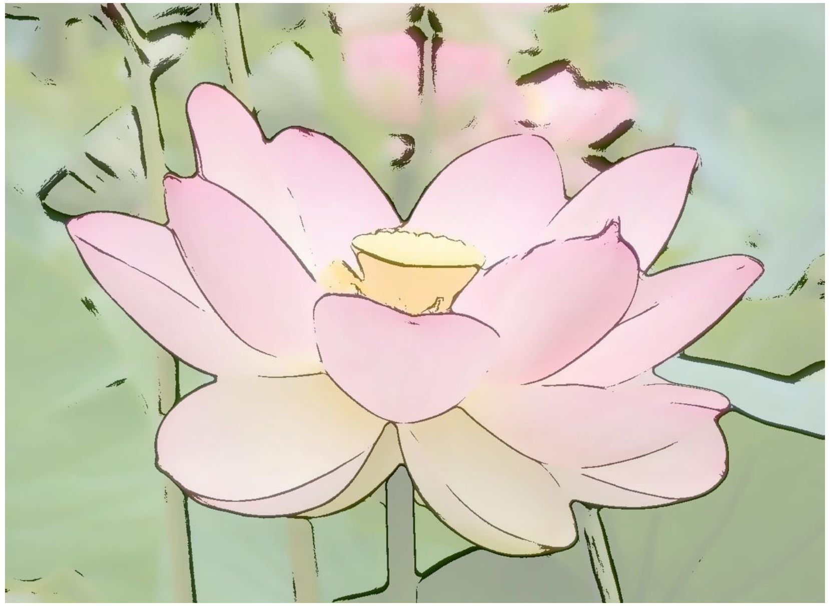
cartoon rendition

Cartoonization



How would you create this effect?

Cartoonization



edges from bilaterally filtered image

+

bilaterally filtered image

=

cartoon rendition

Note: image cartoonization and abstraction are very active research areas.

Is the bilateral filter:

Linear?

Shift-invariant?

Is the bilateral filter:

Linear?

- No.

Shift-invariant?

- No.

Does this have any bad implications?

The bilateral grid

Real-time Edge-Aware Image Processing with the Bilateral Grid

Jiawen Chen   Sylvain Paris   Frédo Durand

Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology

Figure 1: *The bilateral grid enables edge-aware image manipulations such as local tone mapping on high resolution images in real time. This 15 megapixel HDR panorama was tone mapped and locally refined using an edge-aware brush at 50 Hz. The inset shows the original input. The process used about 1 MB of texture memory.*

Data structure for fast edge-aware image processing.

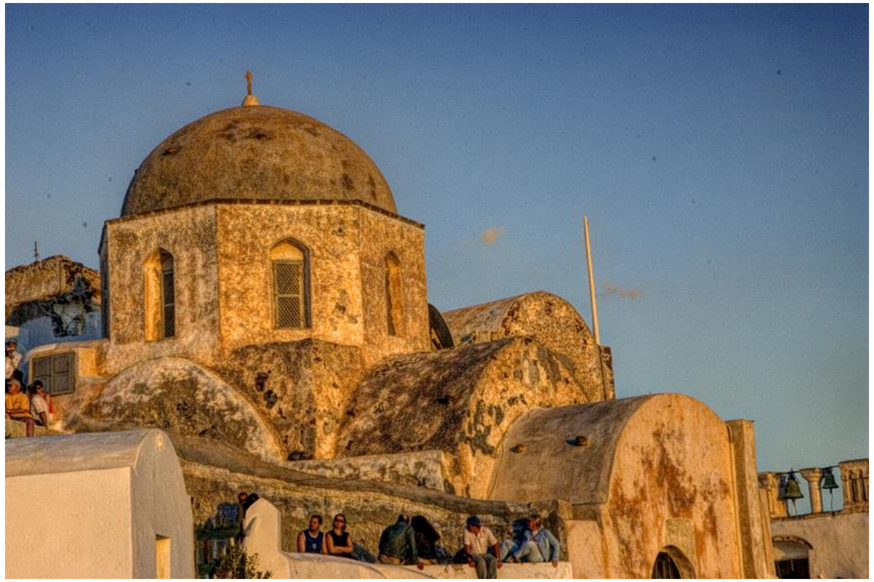
Modern edge-aware filtering: local Laplacian pyramids

Modern edge-aware filtering: local Laplacian pyramids

input

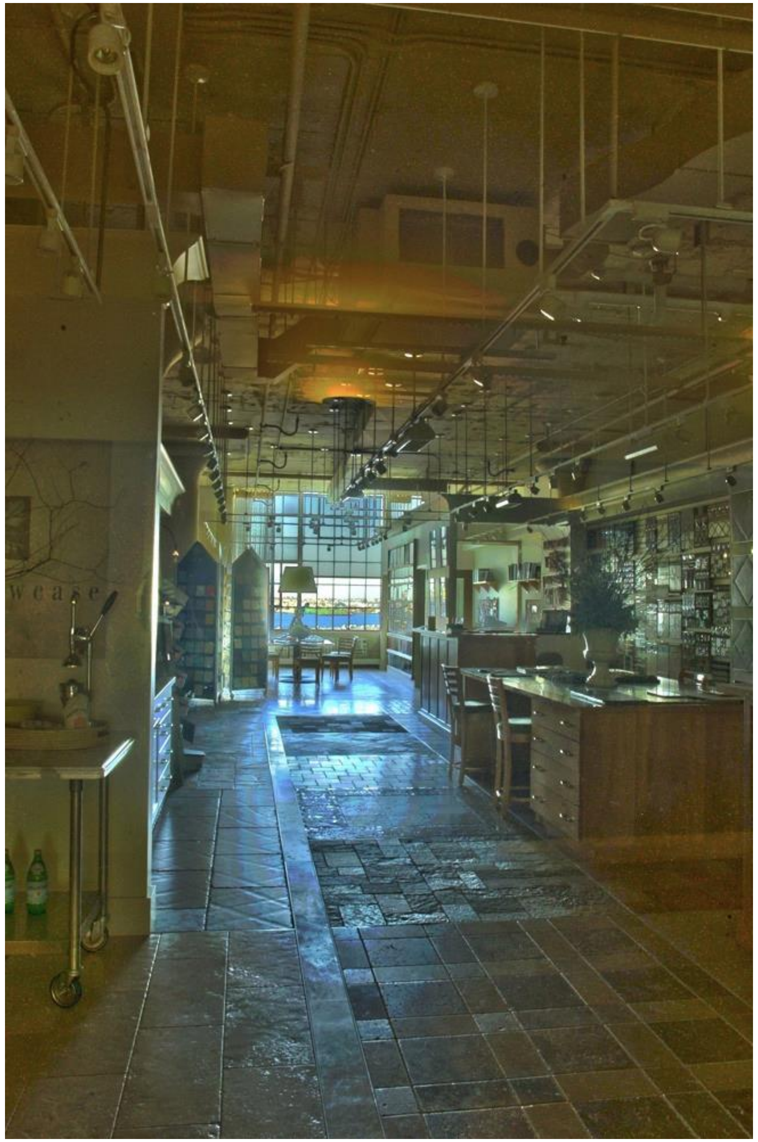
texture increase

texture decrease

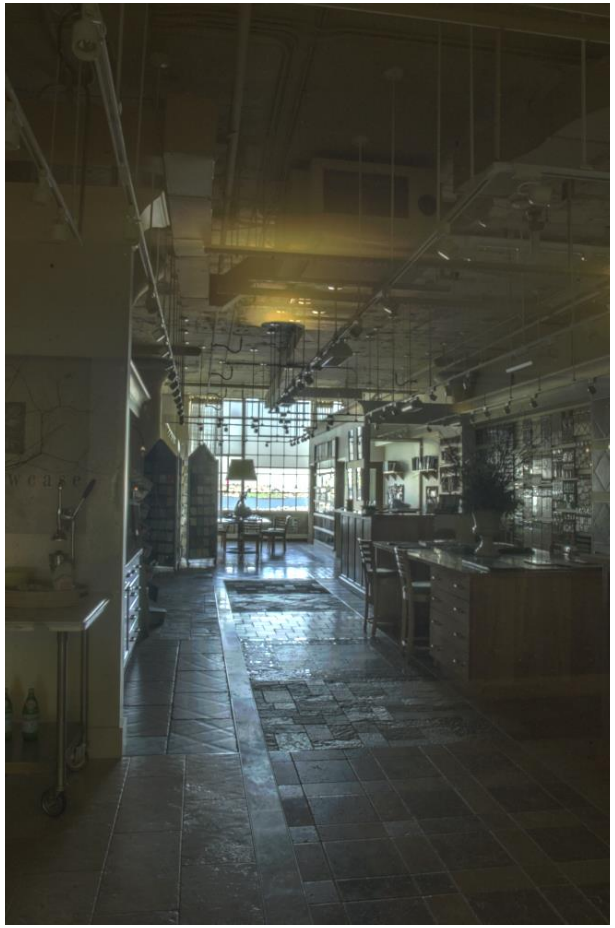


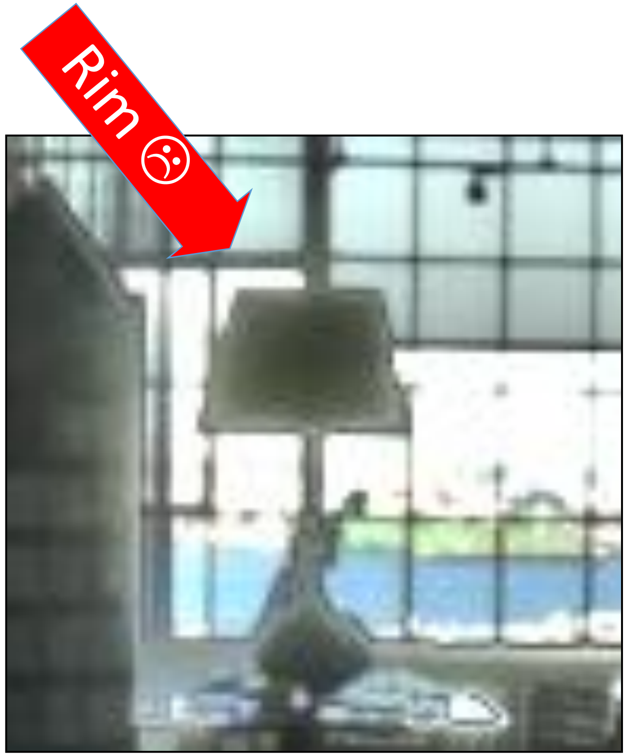
large texture increase

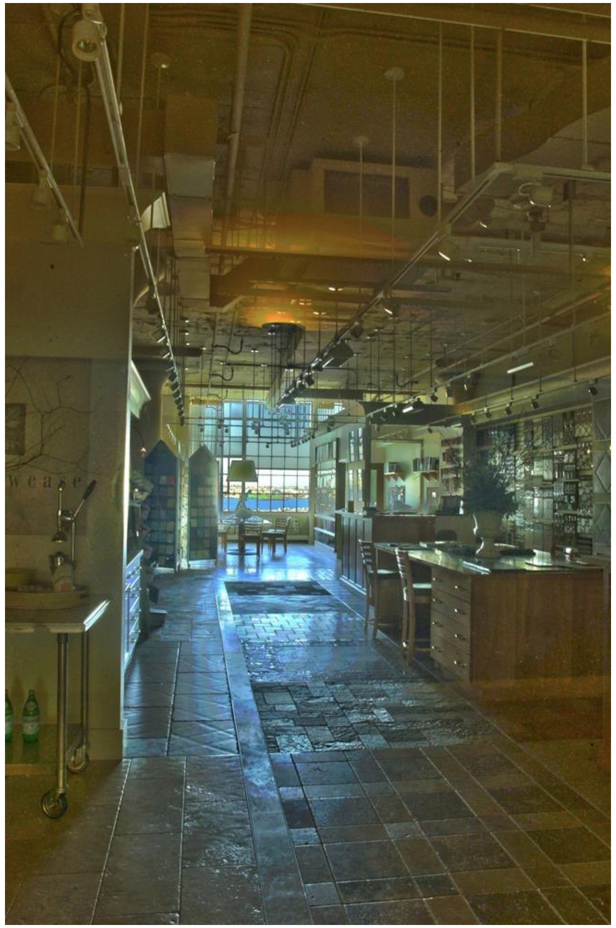
Tonemapping with edge-aware filtering



Tonemapping with edge-aware filtering







local Laplacian pyramids

bilateral filter

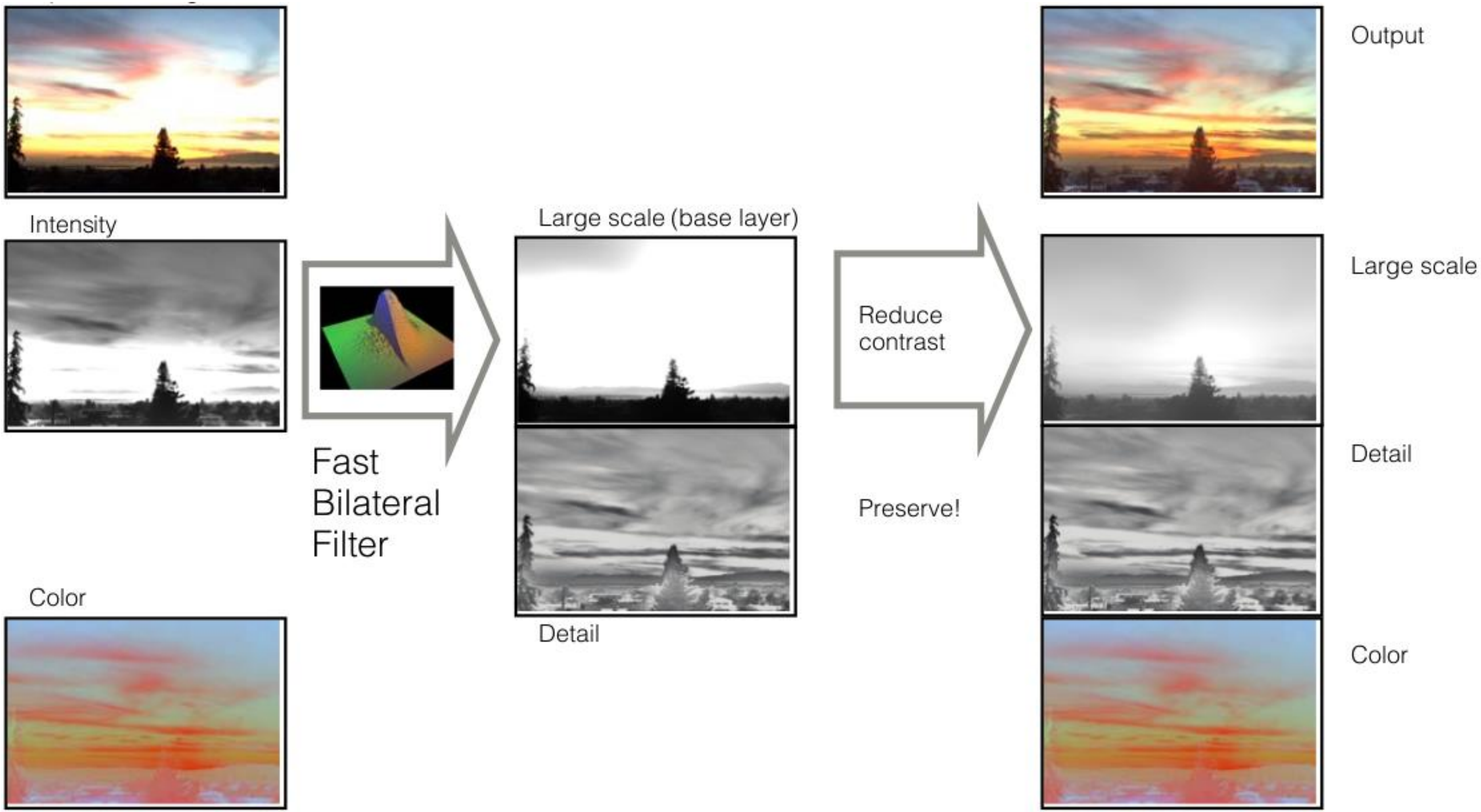
Back to tonemapping

Comparison

We got nice color and contrast, but now we’ve run into the halo plague

Can you think of a way to deal with this?

Tonemapping with bilateral filtering



Comparison

We fixed the halos without losing contrast

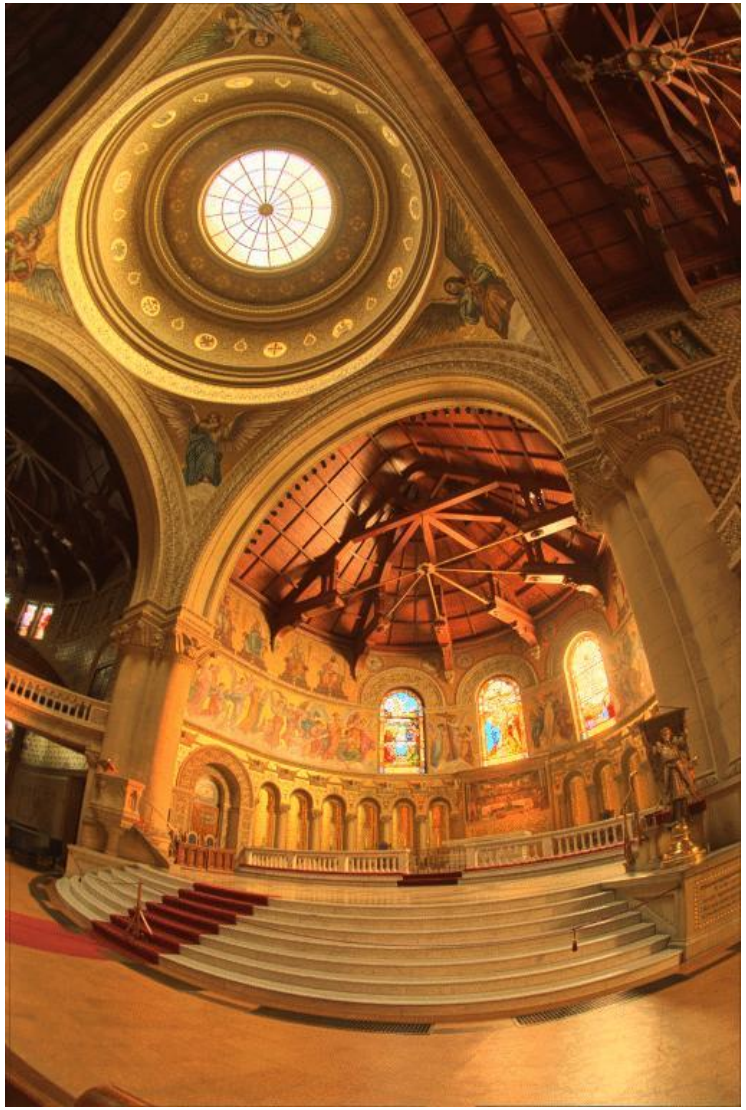
Gradient-domain merging and tonemapping

Compute gradients, scale and merge them, then integrate (solve Poisson problem).

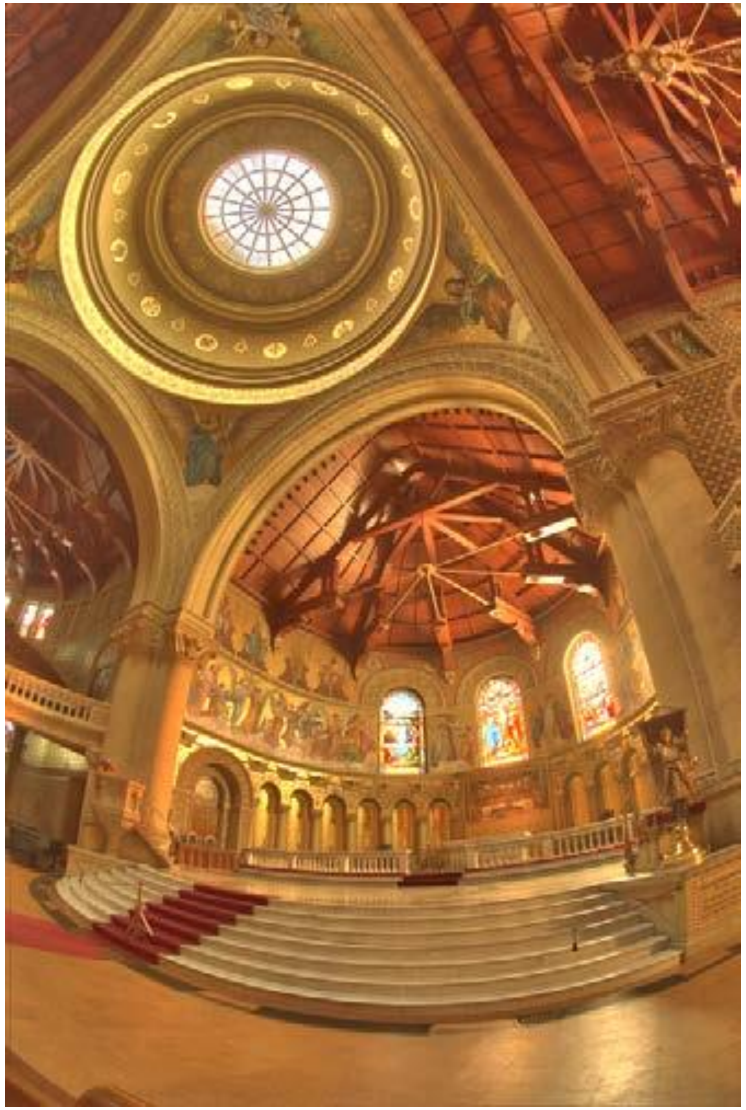
- More in lecture 7.

Gradient-domain merging and tonemapping

Comparison (which one do you like better?)



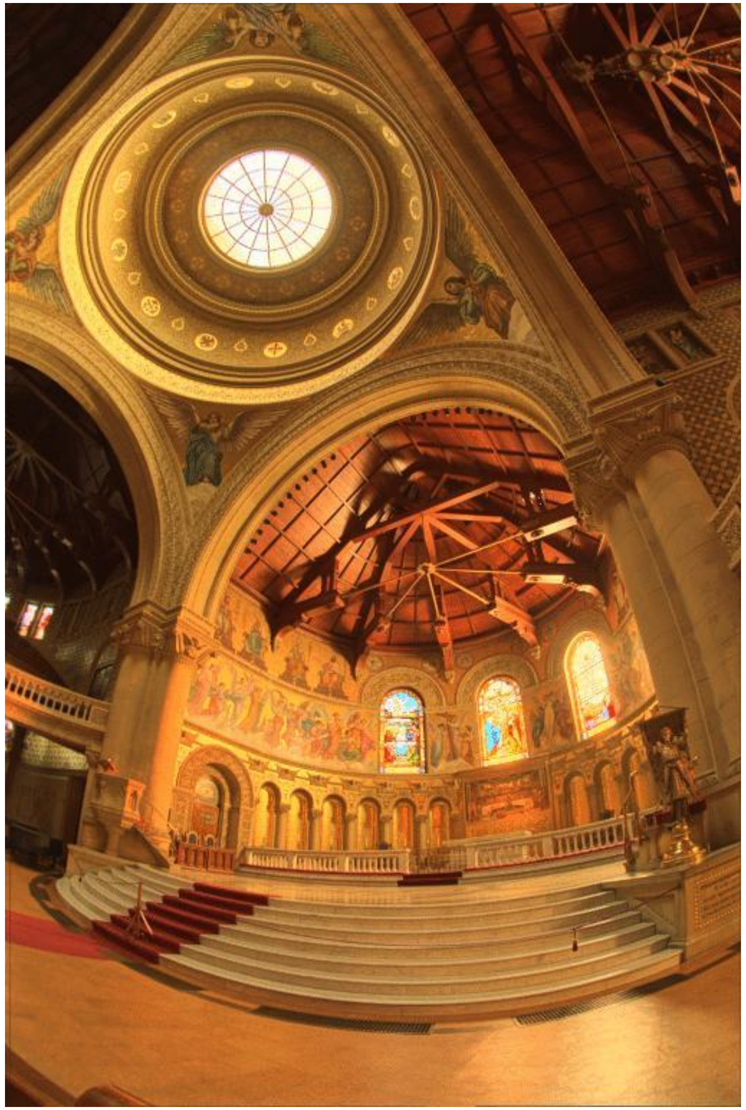
photographic



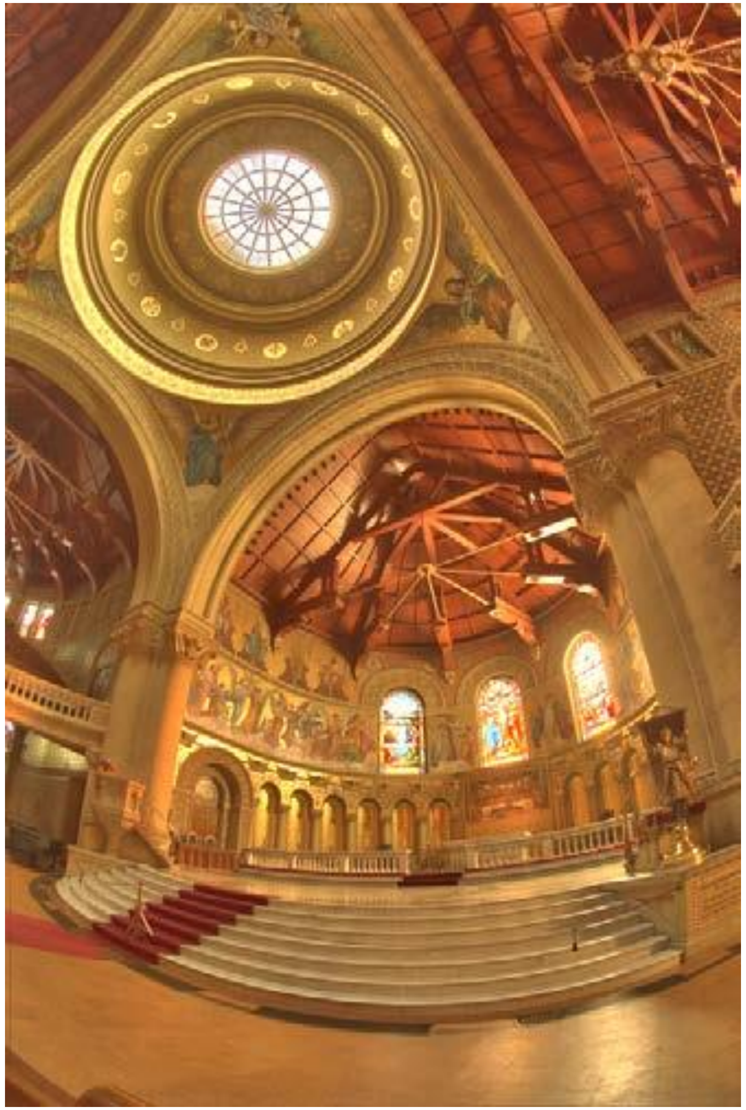
bilateral filtering

gradient-domain

Comparison (which one do you like better?)



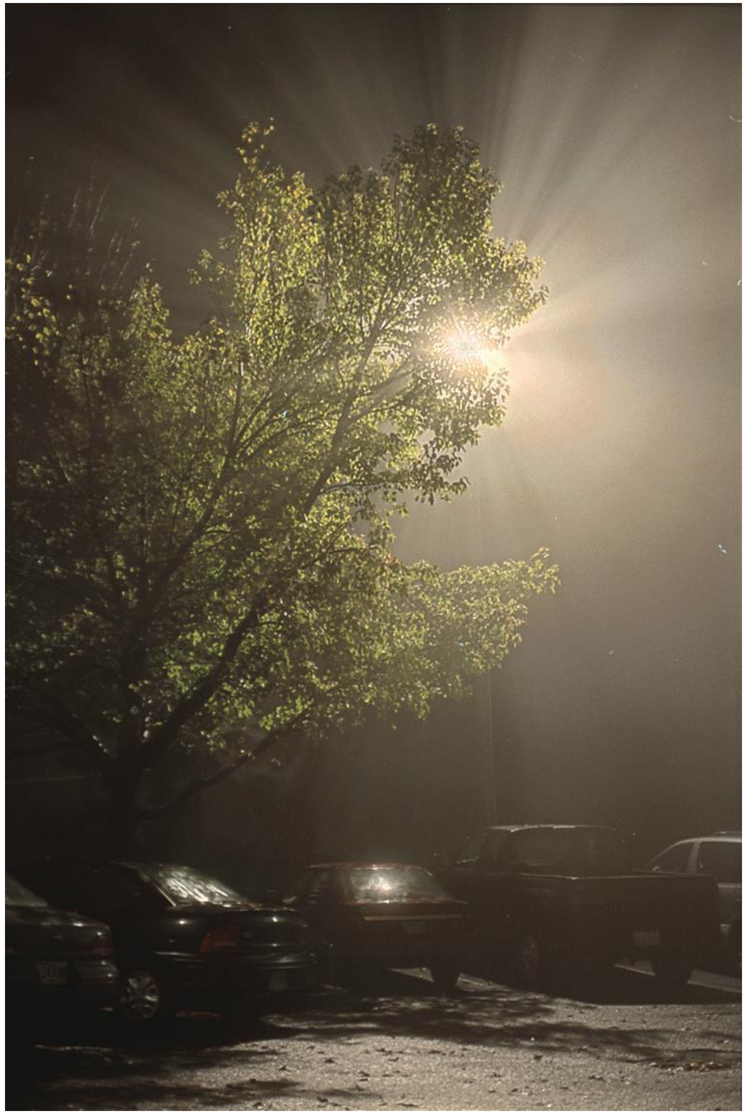
photographic



bilateral filtering

gradient-domain

Comparison (which one do you like better?)

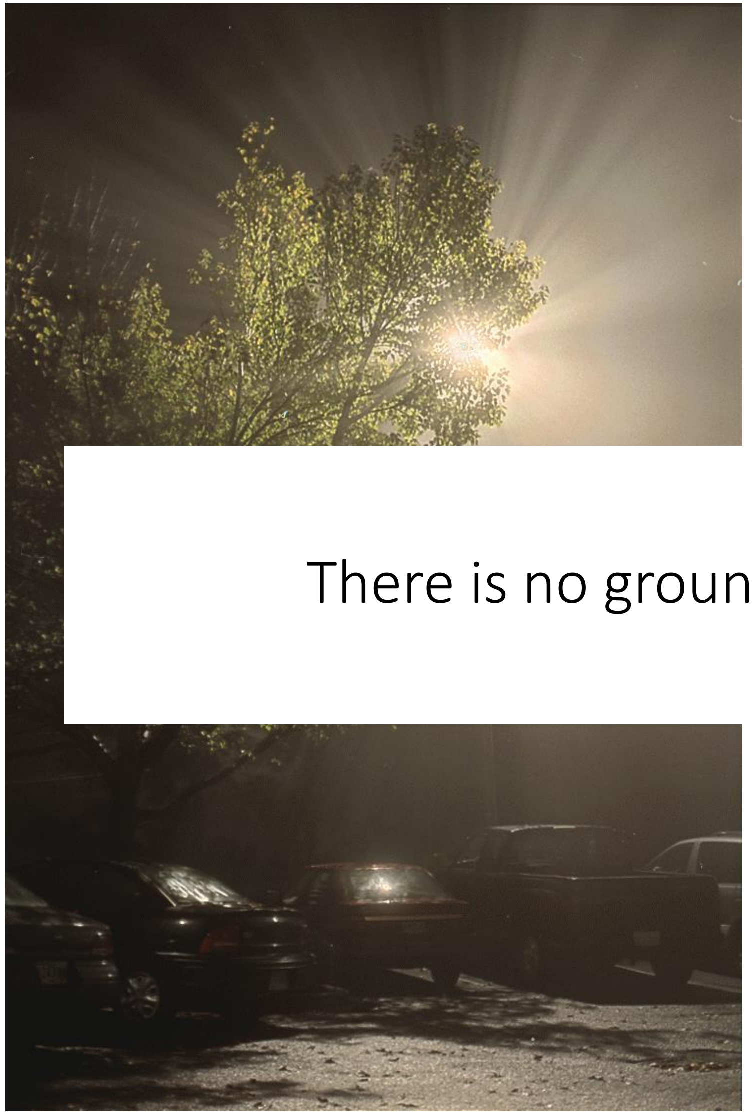


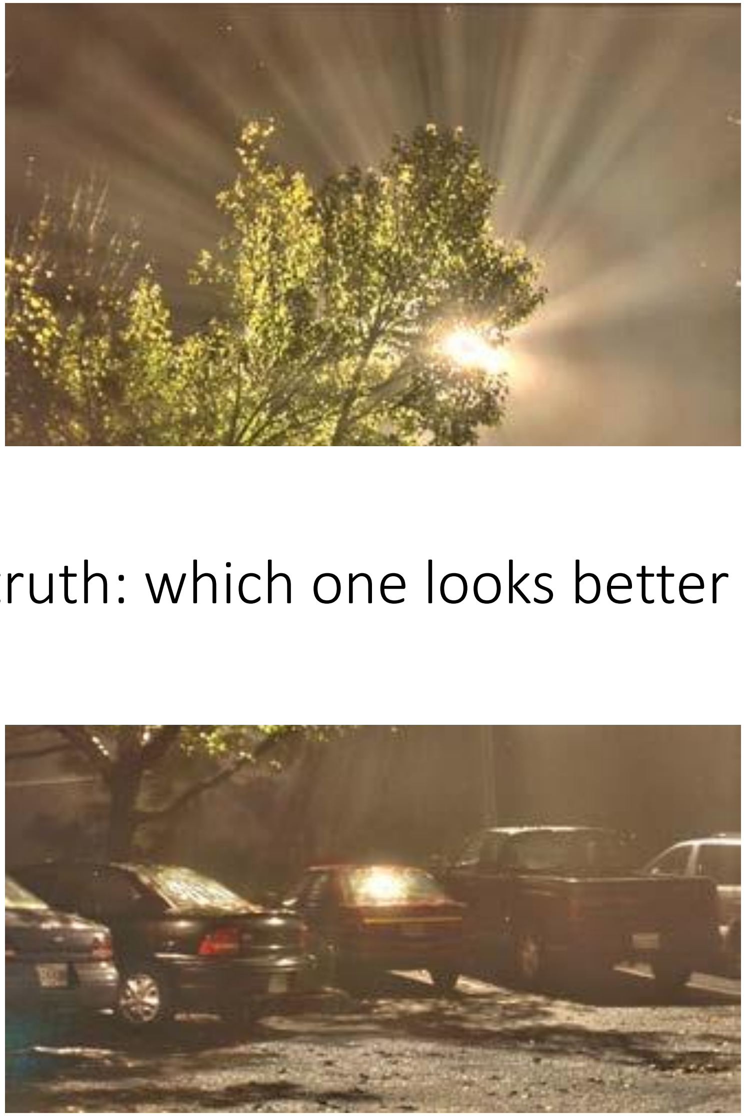
photographic

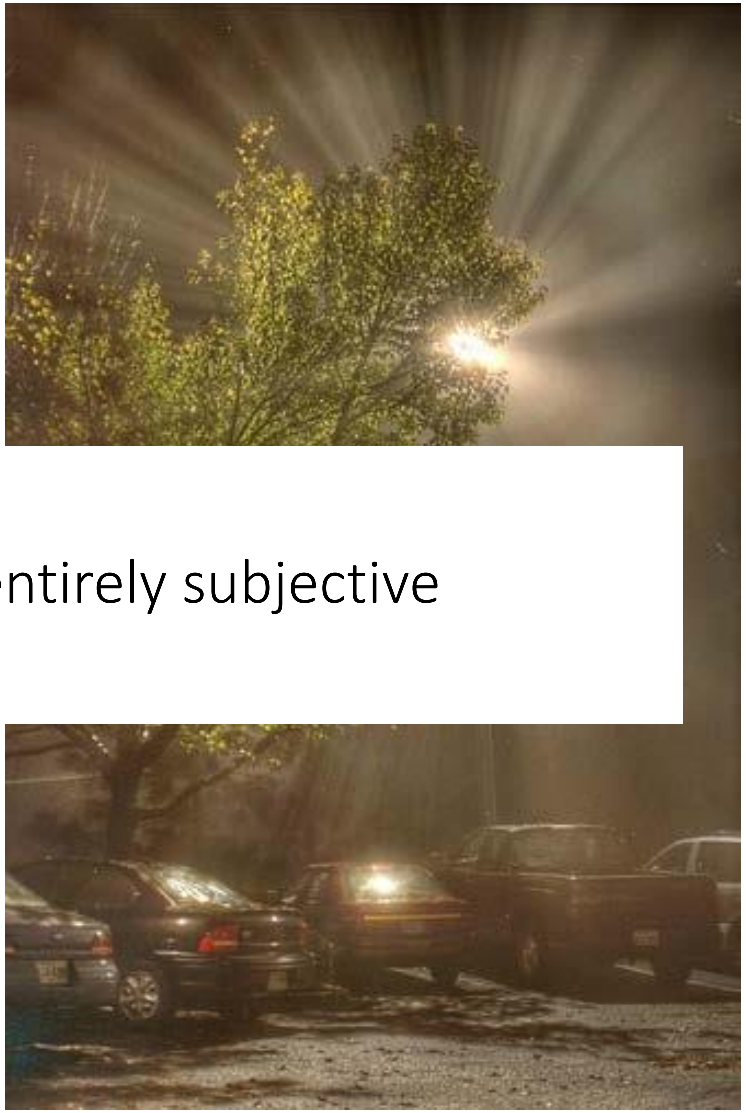
bilateral filtering

gradient-domain

Comparison (which one do you like better?)







There is no ground-truth: which one looks better is entirely subjective

photographic | bilateral filtering | gradient-domain

Tonemapping for a single image

Modern DSLR sensors capture about 3 stops of dynamic range.

- Tonemap single RAW file instead of using camera's default rendering.

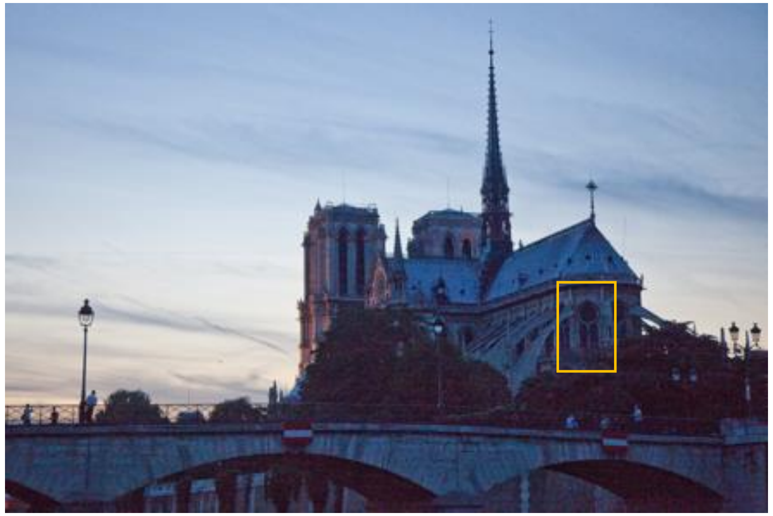
result from image processing pipeline (basic tone reproduction)

tonemapping using bilateral filtering (I think)

Tonemapping for a single image

Modern DSLR sensors capture about 3 stops of dynamic range.

- Tonemap single RAW file instead of using camera's default rendering.



Careful not to "tonemap" noise.

- Why is this not a problem with multi-exposure HDR?

Some notes about HDR and tonemapping

A note of caution

- HDR photography can produce very visually compelling results

A note of caution

- HDR photography can produce very visually compelling results

- It is also a very routinely abused technique, resulting in awful results

A note of caution

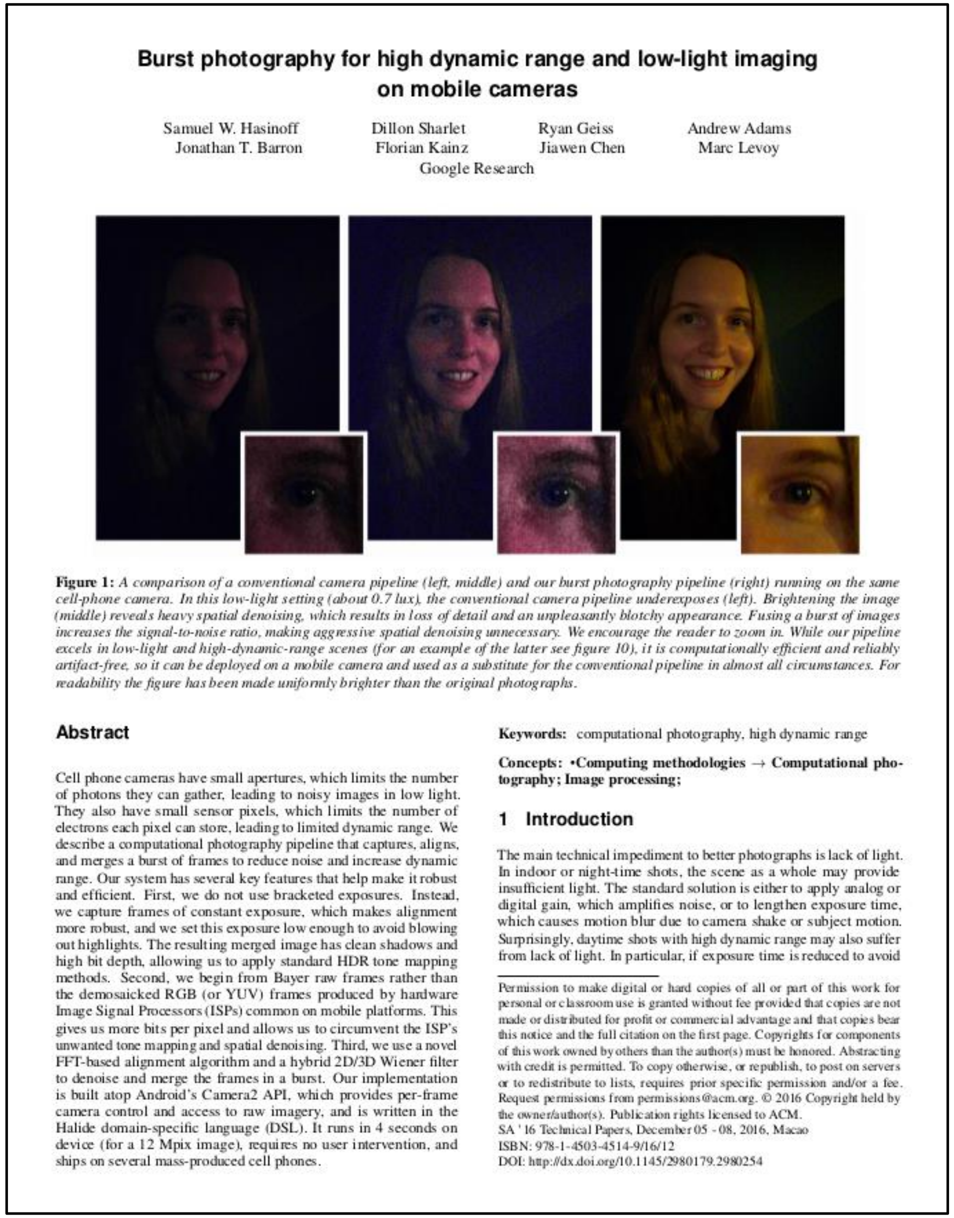
- HDR photography can produce very visually compelling results
- It is also a very routinely abused technique, resulting in awful results
- The problem is tonemapping, not HDR itself

A note about HDR today

- Most cameras (even phone cameras) have automatic HDR modes/apps

- Popular-enough feature that phone manufacturers are actively competing about which one has the best HDR

- The technology behind some of those apps (e.g., Google's HDR+) is published in SIGGRAPH and SIGGRAPH Asia conferences



Burst photography for high dynamic range and low-light imaging on mobile cameras

Samuel W. Hasinoff, Dillon Sharlet, Ryan Geiss, Andrew Adams, Jonathan T. Barron, Florian Kainz, Jiawen Chen, Marc Levoy
Google Research

Figure 1: *A comparison of a conventional camera pipeline (left, middle) and our burst photography pipeline (right) running on the same cell-phone camera. In this low-light setting (about 0.7 lux), the conventional camera pipeline underexposes (left). Brightening the image (middle) reveals heavy spatial denoising, which results in loss of detail and an unpleasantly blotchy appearance. Fusing a burst of images increases the signal-to-noise ratio, making aggressive spatial denoising unnecessary. We encourage the reader to zoom in. While our pipeline excels in low-light and high-dynamic-range scenes (for an example of the latter see figure 10), it is computationally efficient and reliably artifact-free, so it can be deployed on a mobile camera and used as a substitute for the conventional pipeline in almost all circumstances. For readability the figure has been made uniformly brighter than the original photographs.*

Abstract

Cell phone cameras have small apertures, which limits the number of photons they can gather, leading to noisy images in low light. They also have small sensor pixels, which limits the number of electrons each pixel can store, leading to limited dynamic range. We describe a computational photography pipeline that captures, aligns, and merges a burst of frames to reduce noise and increase dynamic range. Our system has several key features that help make it robust and efficient. First, we do not use bracketed exposures. Instead, we capture frames of constant exposure, which makes alignment more robust, and we set this exposure low enough to avoid blowing out highlights. The resulting merged image has clean shadows and high bit depth, allowing us to apply standard HDR tone mapping methods. Second, we begin from Bayer raw frames rather than the demosaicked RGB (or YUV) frames produced by hardware Image Signal Processors (ISPs) common on mobile platforms. This gives us more bits per pixel and allows us to circumvent the ISP's unwanted tone mapping and spatial denoising. Third, we use a novel FFT-based alignment algorithm and a hybrid 2D/3D Wiener filter to denoise and merge the frames in a burst. Our implementation is built atop Android's Camera2 API, which provides per-frame camera control and access to raw imagery, and is written in the Halide domain-specific language (DSL). It runs in 4 seconds on device (for a 12 Mpix image), requires no user intervention, and ships on several mass-produced cell phones.

Keywords: computational photography, high dynamic range

Concepts: **•Computing methodologies → Computational photography; Image processing;**

1 Introduction

The main technical impediment to better photographs is lack of light. In indoor or night-time shots, the scene as a whole may provide insufficient light. The standard solution is either to apply analog or digital gain, which amplifies noise, or to lengthen exposure time, which causes motion blur due to camera shake or subject motion. Surprisingly, daytime shots with high dynamic range may also suffer from lack of light. In particular, if exposure time is reduced to avoid

SA '16 Technical Papers, December 05 - 08, 2016, Macao
ISBN: 978-1-4503-4514-9/16/12
DOI: http://dx.doi.org/10.1145/2980179.2980254

References

Basic reading:

- Szeliski textbook, Sections 10.1, 10.2. Reinhard et al., “Photographic Tone Reproduction for Digital Images,” SIGGRAPH 2002.
  The photographic tonemapping paper, including a very nice discussion of the zone system for film.
- Durand and Dorsey, “Fast bilateral filtering for the display of high-dynamic-range images,” SIGGRAPH 2002.
  The paper on tonemapping using bilateral filtering.
- Paris et al., “A Gentle Introduction to the Bilateral Filter and Its Applications,” SIGGRAPH 2007-08, CVPR 2008
  Short course on the bilateral filter, including discussion of fast implementations,
  https://people.csail.mit.edu/sparis/bf_course/
- Fattal et al., “Gradient Domain High Dynamic Range Compression,” SIGGRAPH 2002.
  The paper on gradient-domain tonemapping.

Additional reading:

- Reinhard et al., “High Dynamic Range Imaging, Second Edition: Acquisition, Display, and Image-Based Lighting,” Morgan Kaufmann 2010.
  A very comprehensive book about everything relating to HDR imaging and tonemapping.
- Kuang et al., “Evaluating HDR rendering algorithms,” TAP 2007.
  One of many, many papers trying to do a perceptual evaluation of different tonemapping algorithms.
- Hasinoff et al., “Burst photography for high dynamic range and low-light imaging on mobile cameras,” SIGGRAPH Asia 2016.
  The paper describing Google’s HDR+.
- Paris et al., “Local Laplacian Filters: Edge-aware Image Processing with a Laplacian Pyramid,” SIGGRAPH 2011 and CACM 2015.
  The paper on local Laplacian pyramids.