Edge-aware and bilateral filtering

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

15-463, 15-663, 15-862
Computational Photography
Fall 2021, Lecture 9

Course announcements

- Homework assignment 3 posted.
    - Due October 18th.
    - Start early: tricky implementation that is easy to get wrong.

- Grades for homework assignment 2 posted.

- Propose and/or vote for topics for this week's reading group.

Overview of today's lecture

- Leftover from lecture 8.
- Back to tonemapping.
- Edge-aware filtering and bilateral filtering.
- Non-local means.
- Flash/no-flash photography.
- Joint bilateral filtering.

Slide credits

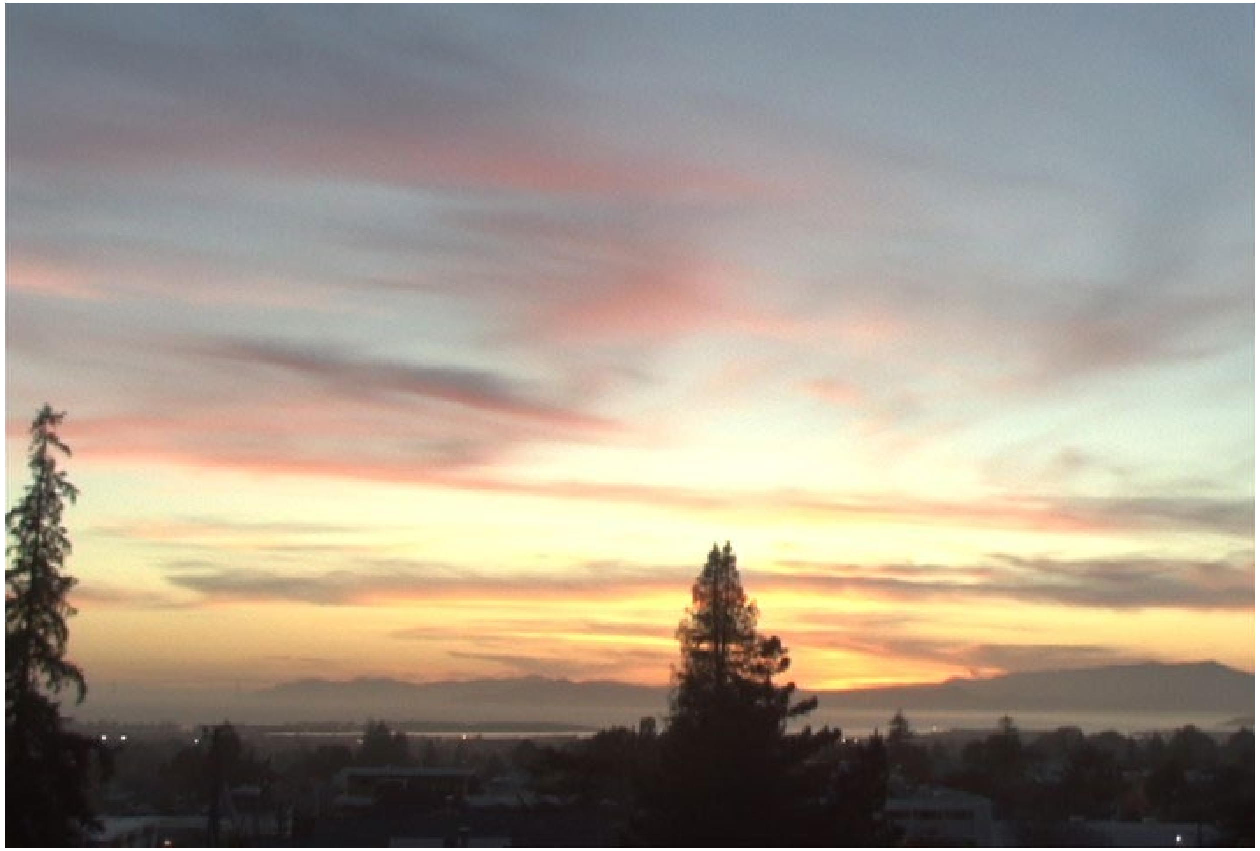
Many of these slides were inspired or adapted from:

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

Back to tonemapping

Dealing with color

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

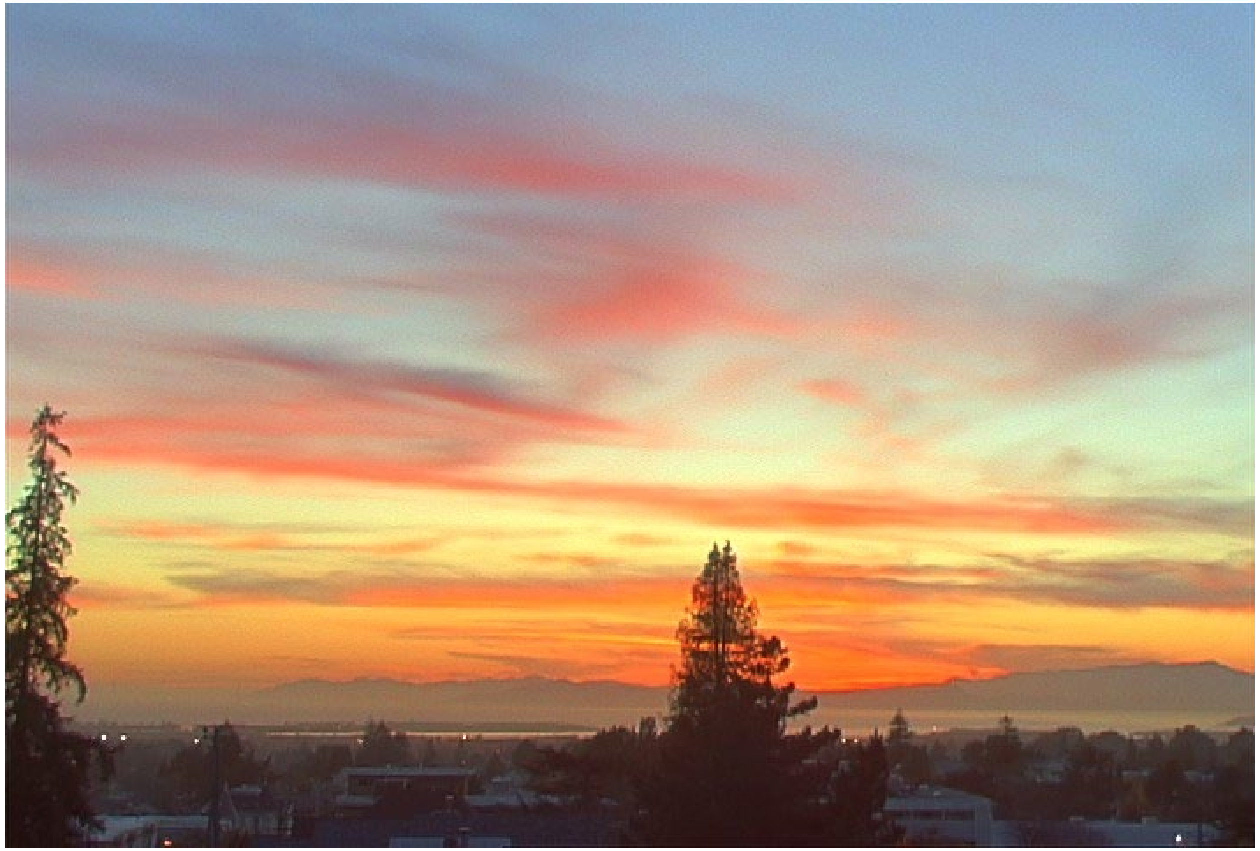


Can you think of a way to deal with this?

Intensity-only tonemapping

tonemap intensity

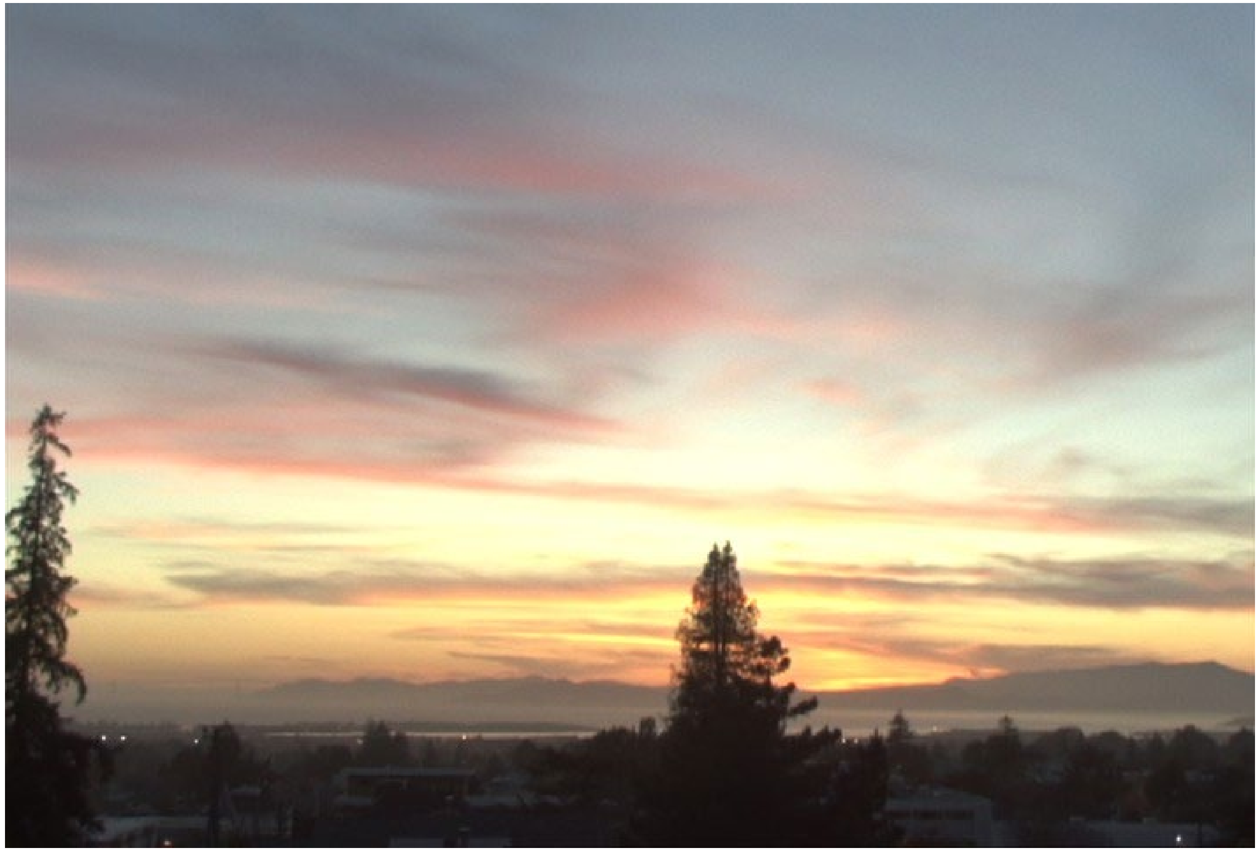
leave color the same

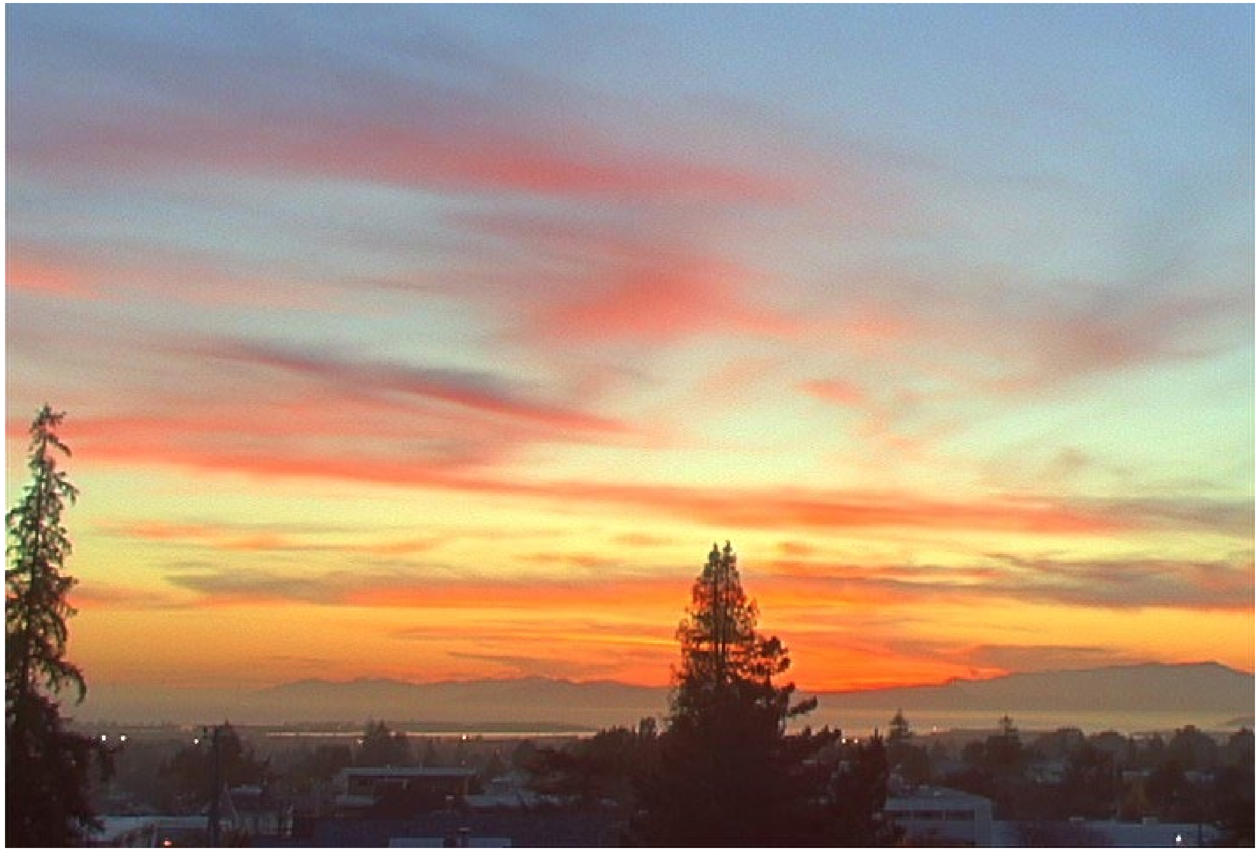


How would you implement this?

Comparison

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

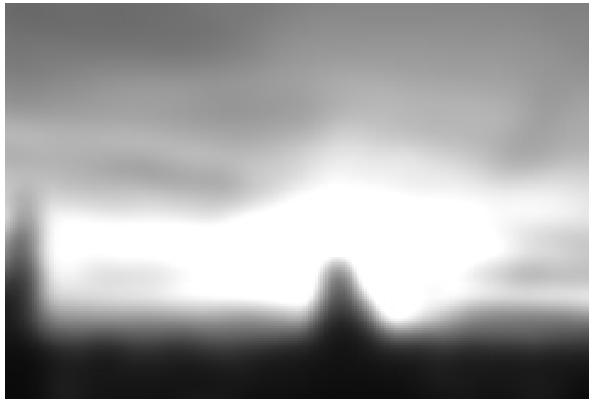




Can you think of a way to deal with this?

Low-frequency intensity-only tonemapping

tonemap low-frequency intensity component



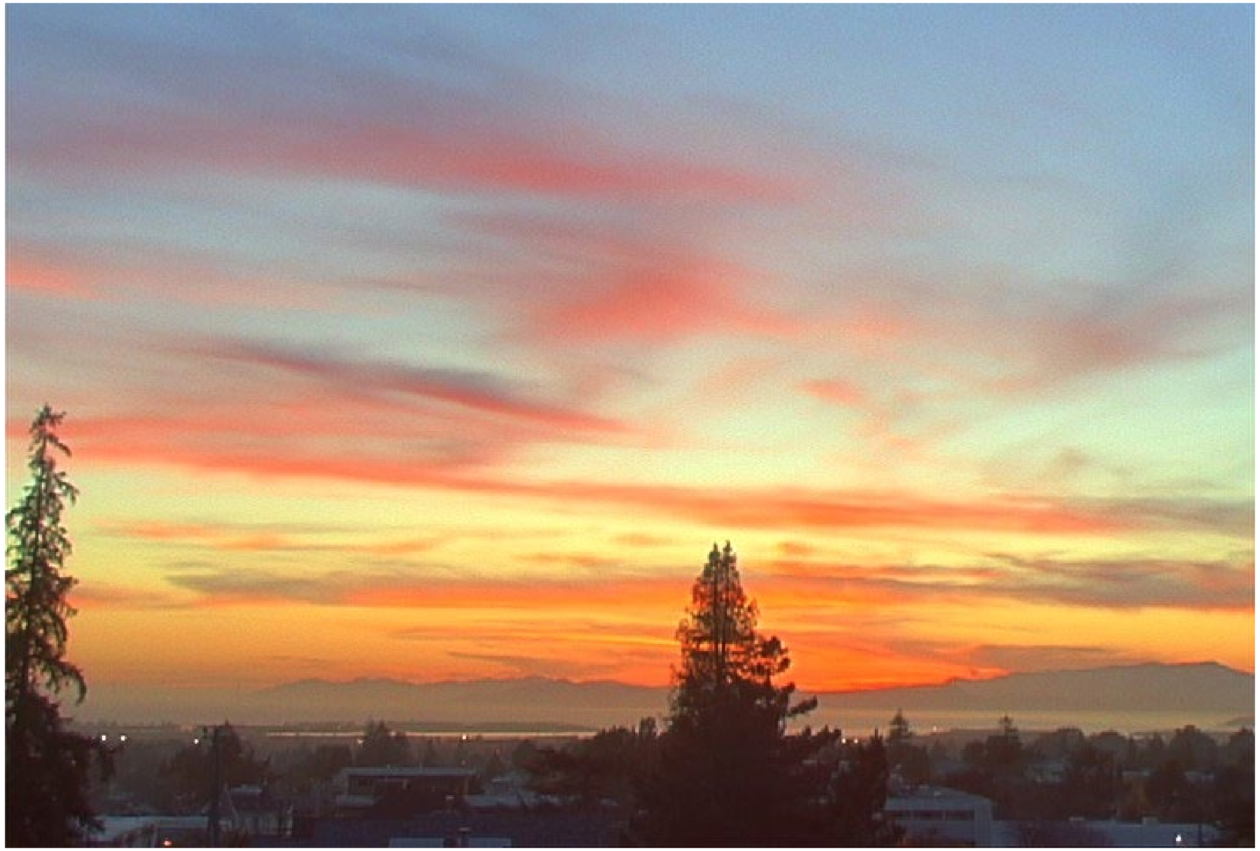
leave high-frequency intensity component the same

leave color the same

How would you implement this?

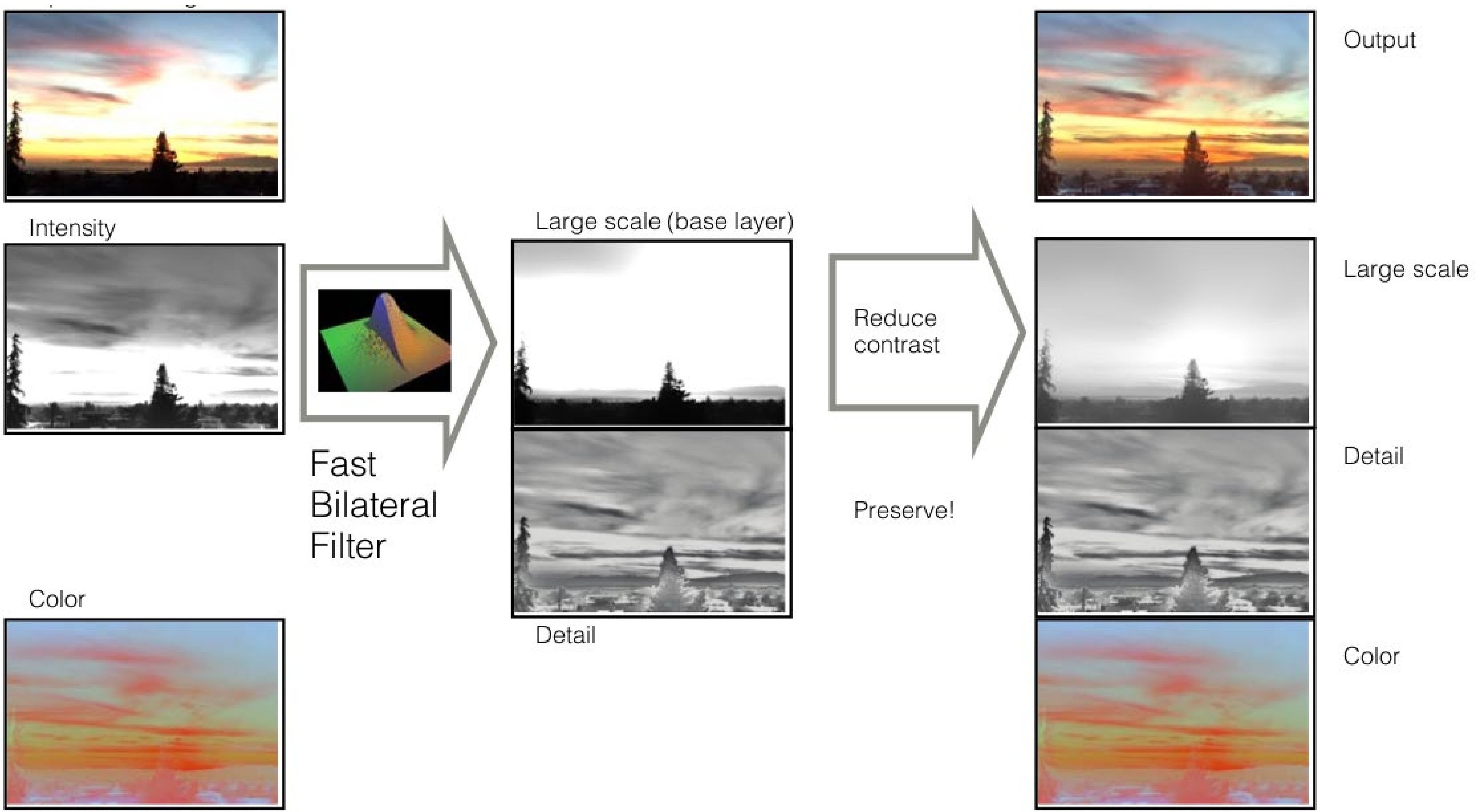
Comparison

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

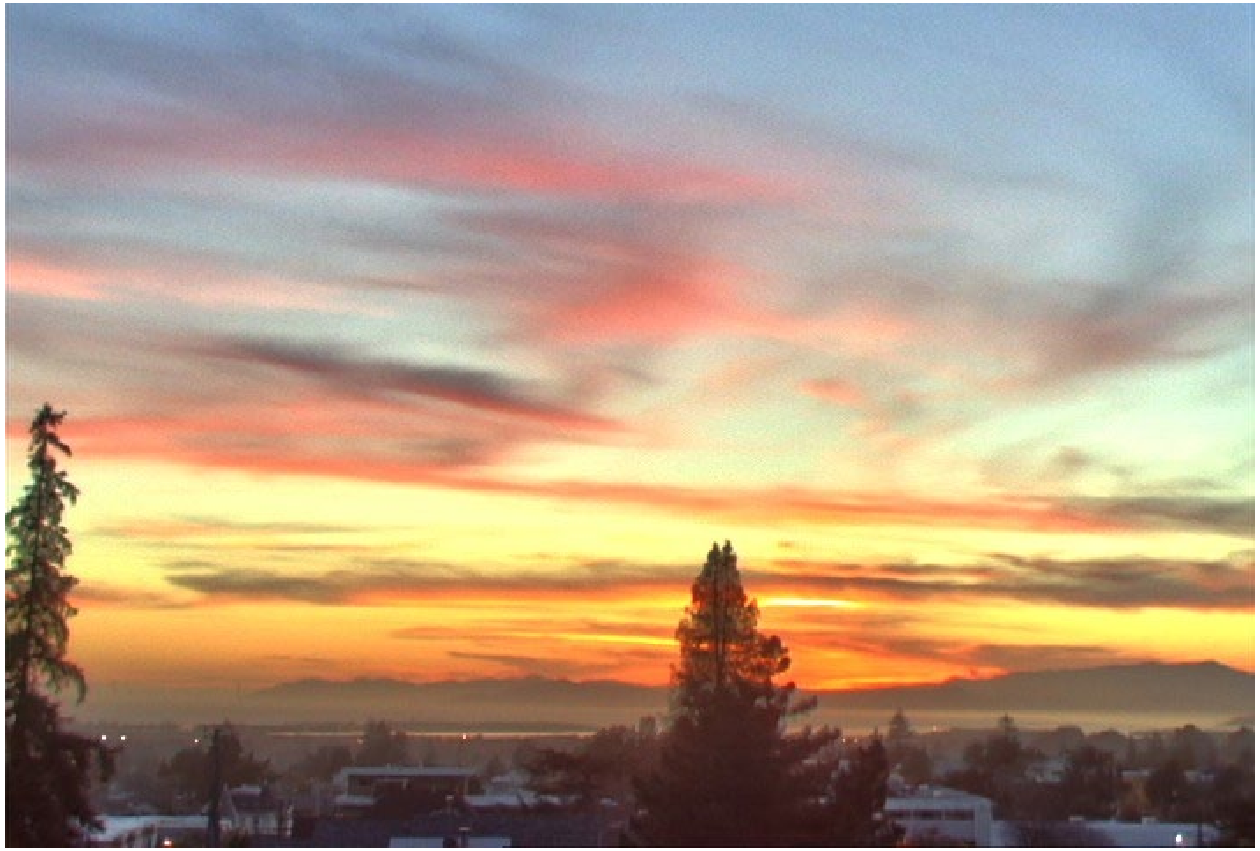
Can you think of a way to deal with this?

Tonemapping with bilateral filtering



Comparison

We fixed the halos without losing contrast



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

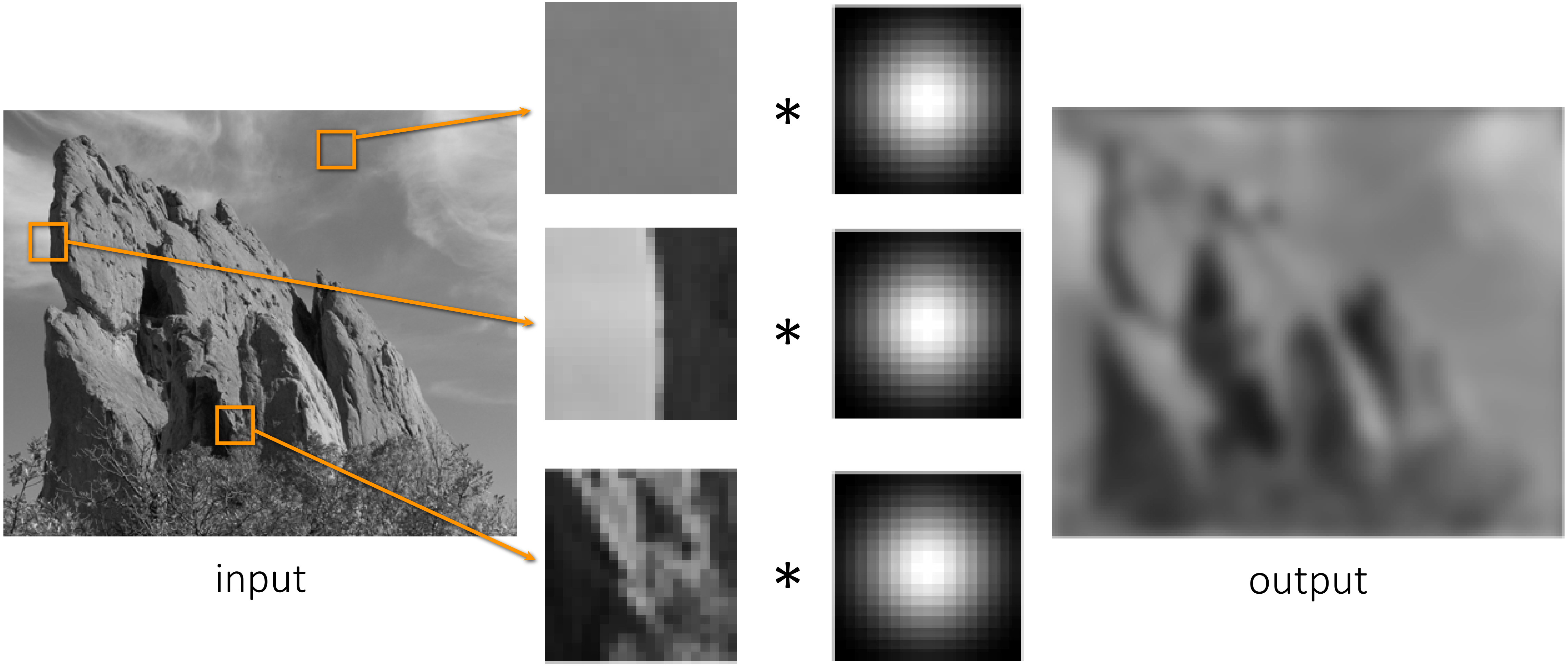
original

Gaussian filtering

bilateral filtering

The problem with Gaussian filtering

Gaussian kernel

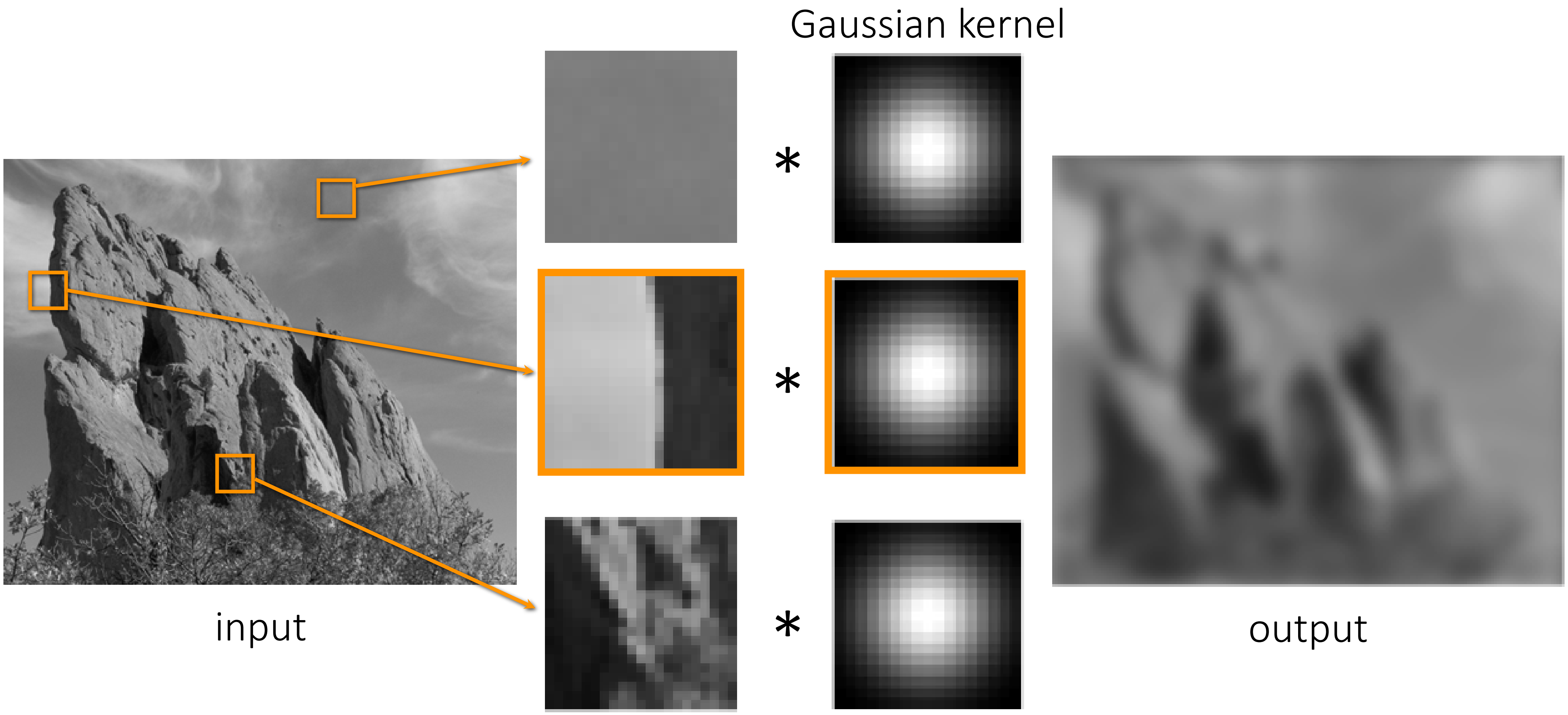


input

output

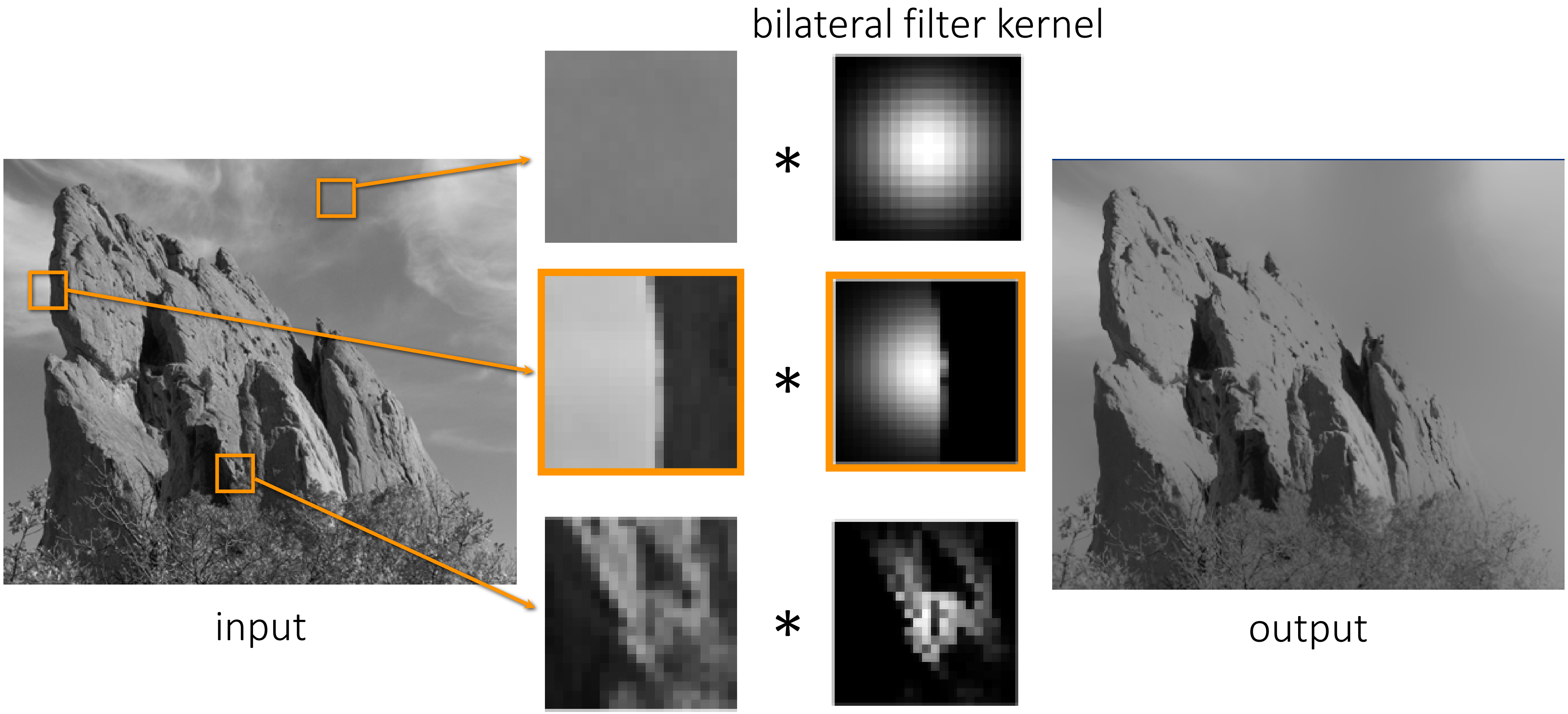
Why is the output so blurry?

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?

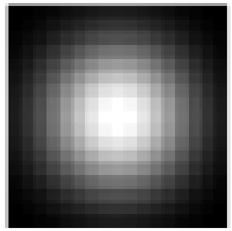
Bilateral filtering

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

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



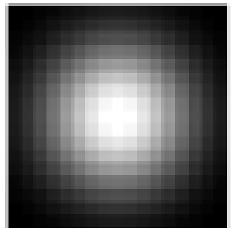
σ_s

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

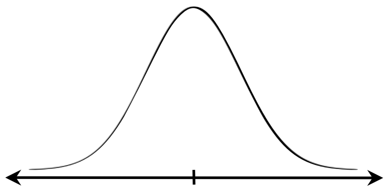
Bilateral filtering

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

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

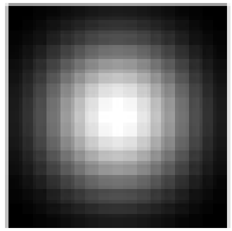
Bilateral filtering

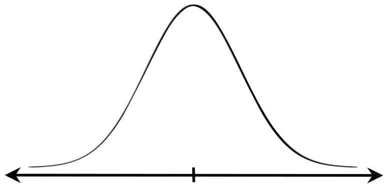
$$h[m,n] = \boxed{\frac{1}{W_{mn}}} \sum_{k,l} \boxed{g[k,l]}\boxed{r_{mn}[k,l]} f[m+k,n+l]$$

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

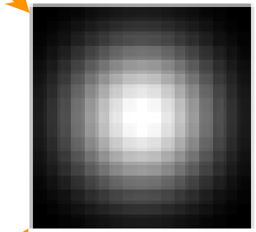
$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} g[k,l] r_{mn}[k,l] f[m+k, n+l]$$

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

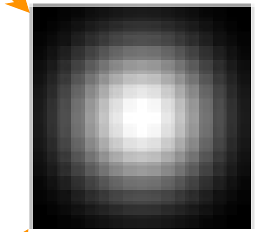
$$h[m,n] = \frac{1}{W_{mn}} \sum_{k,l} \boxed{g[k,l]} r_{mn}[k,l] f[m+k, n+l]$$

Bilateral filtering vs Gaussian filtering

Gaussian filtering

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

σ_s

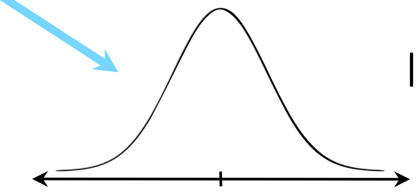


Spatial weighting: favor *nearby* pixels

Bilateral filtering

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

σ_r



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

Intensity range weighting: favor *similar* pixels

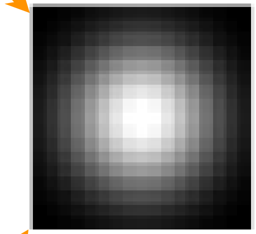
Bilateral filtering vs Gaussian filtering

Gaussian filtering

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

Bilateral filtering

σ_s

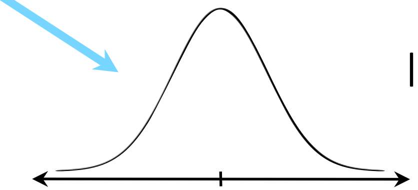


Spatial weighting:
favor *nearby* pixels

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

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

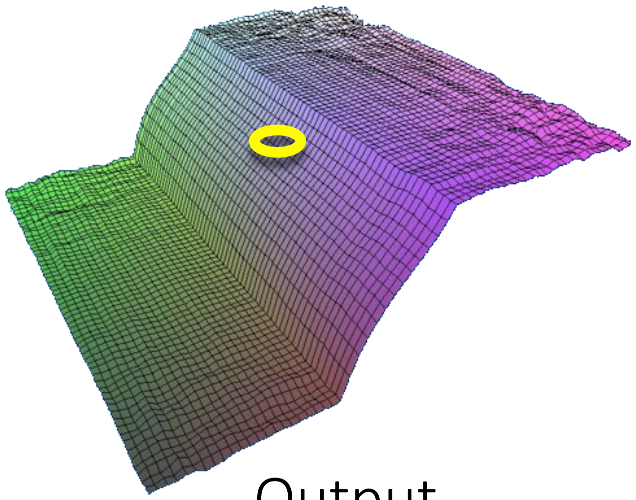
Smooths everything nearby (even edges)
Only depends on *spatial* distance

Bilateral filtering

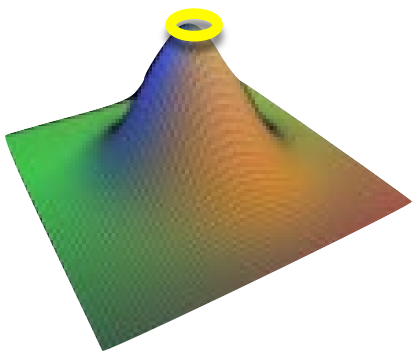
Smooths ‘close’ pixels in space and intensity
Depends on *spatial* and *intensity* distance

Gaussian filtering visualization

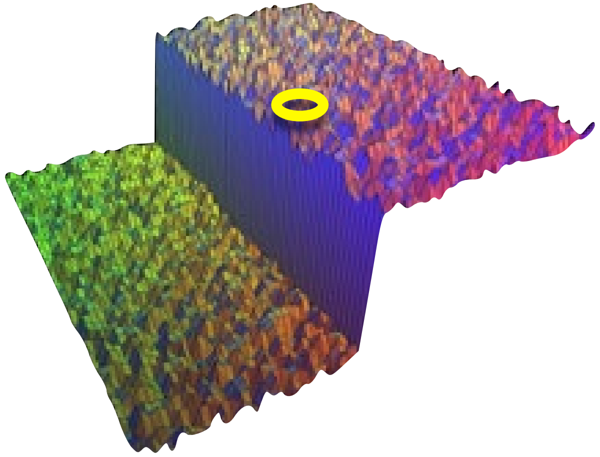
$$h[m,n] = \sum_{k,l} g[k,l] f[m+k,n+l]$$



Output



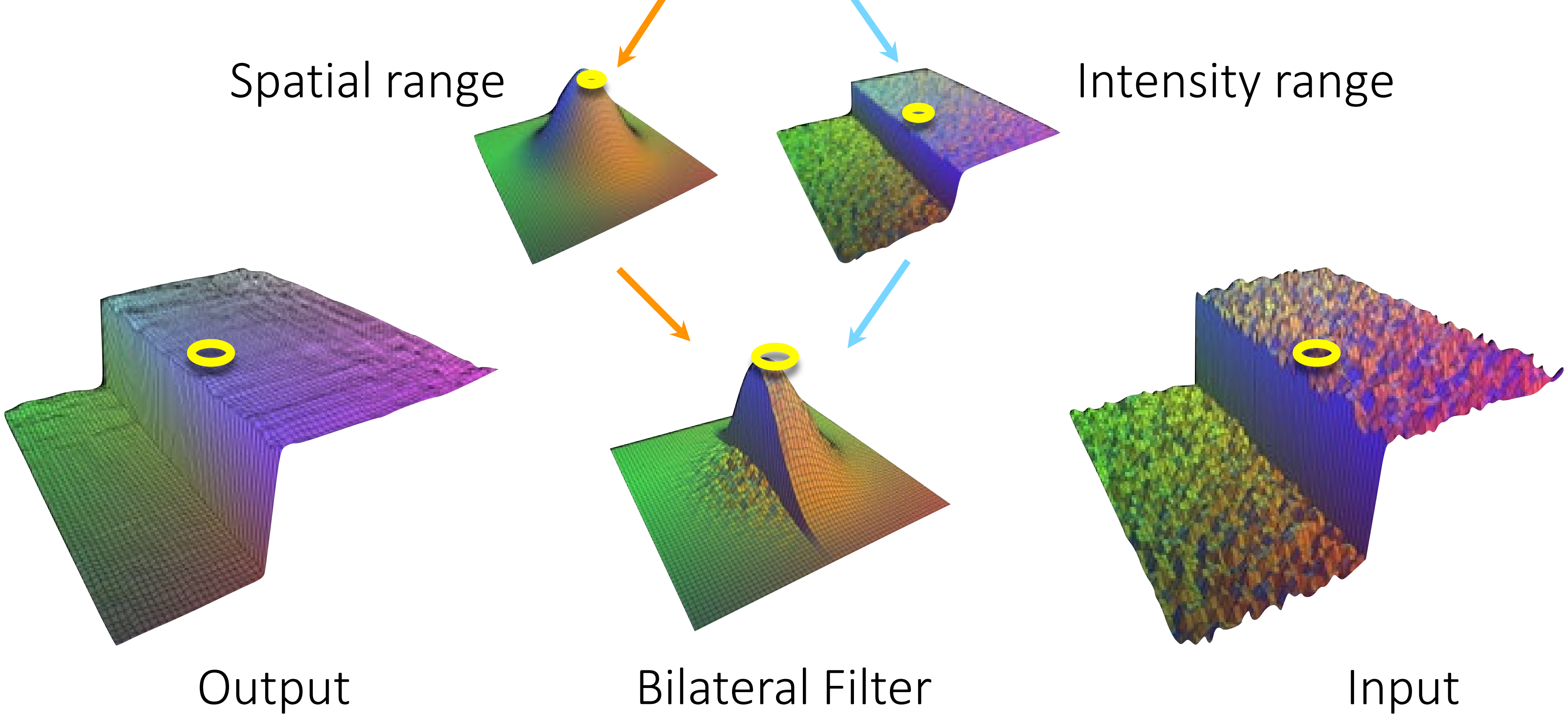
Gaussian Filter



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

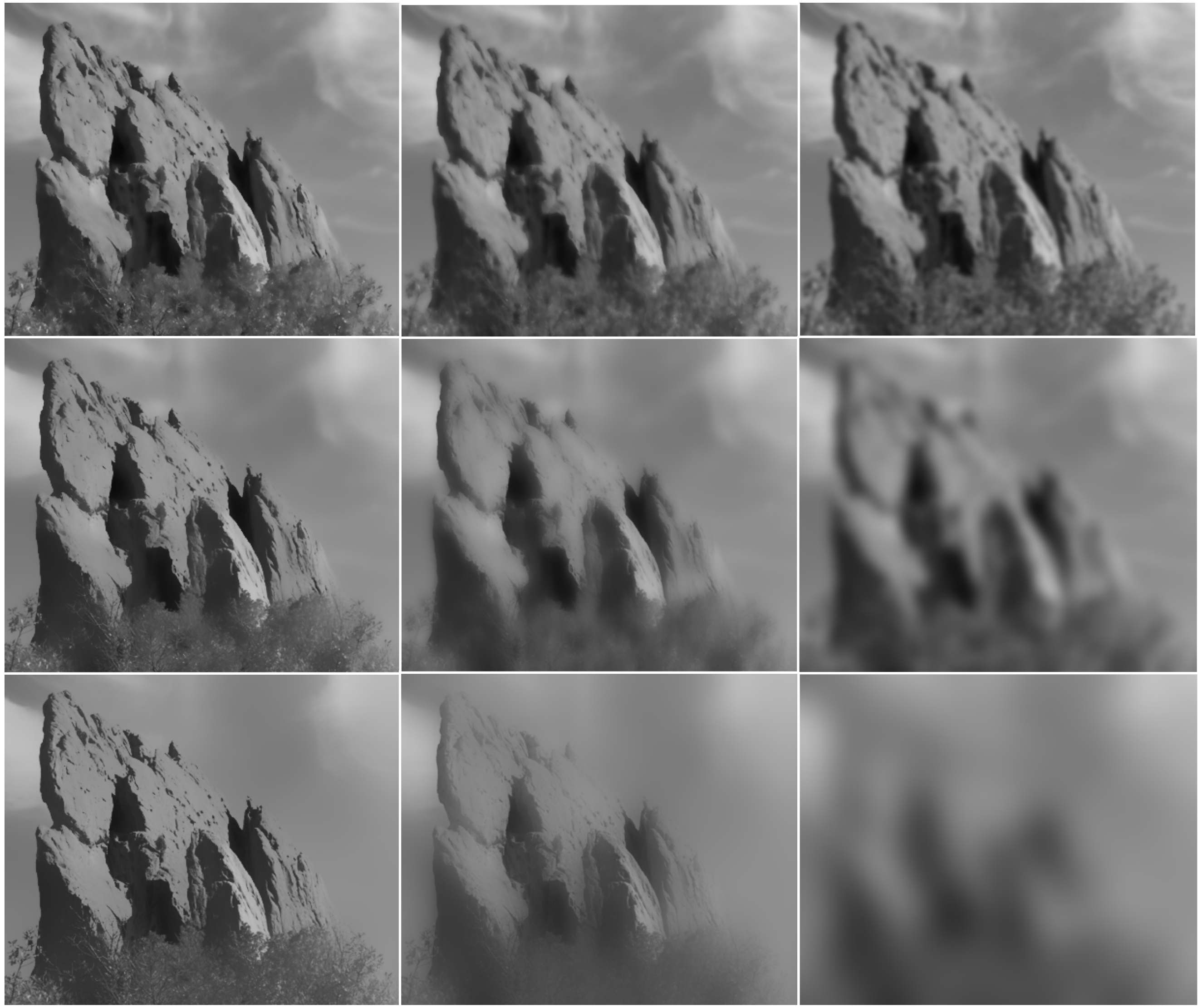
input

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

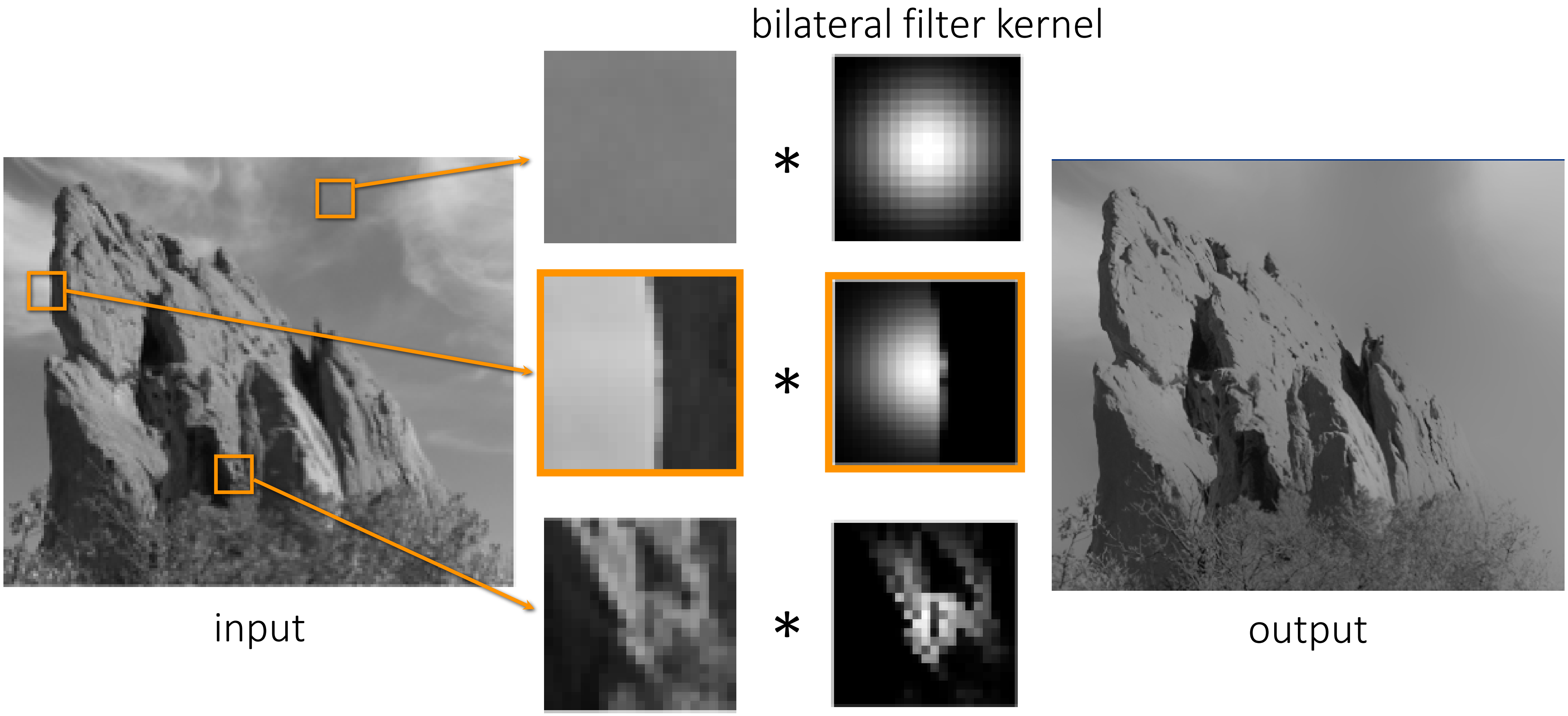
$\sigma_s = 2$

$\sigma_s = 6$

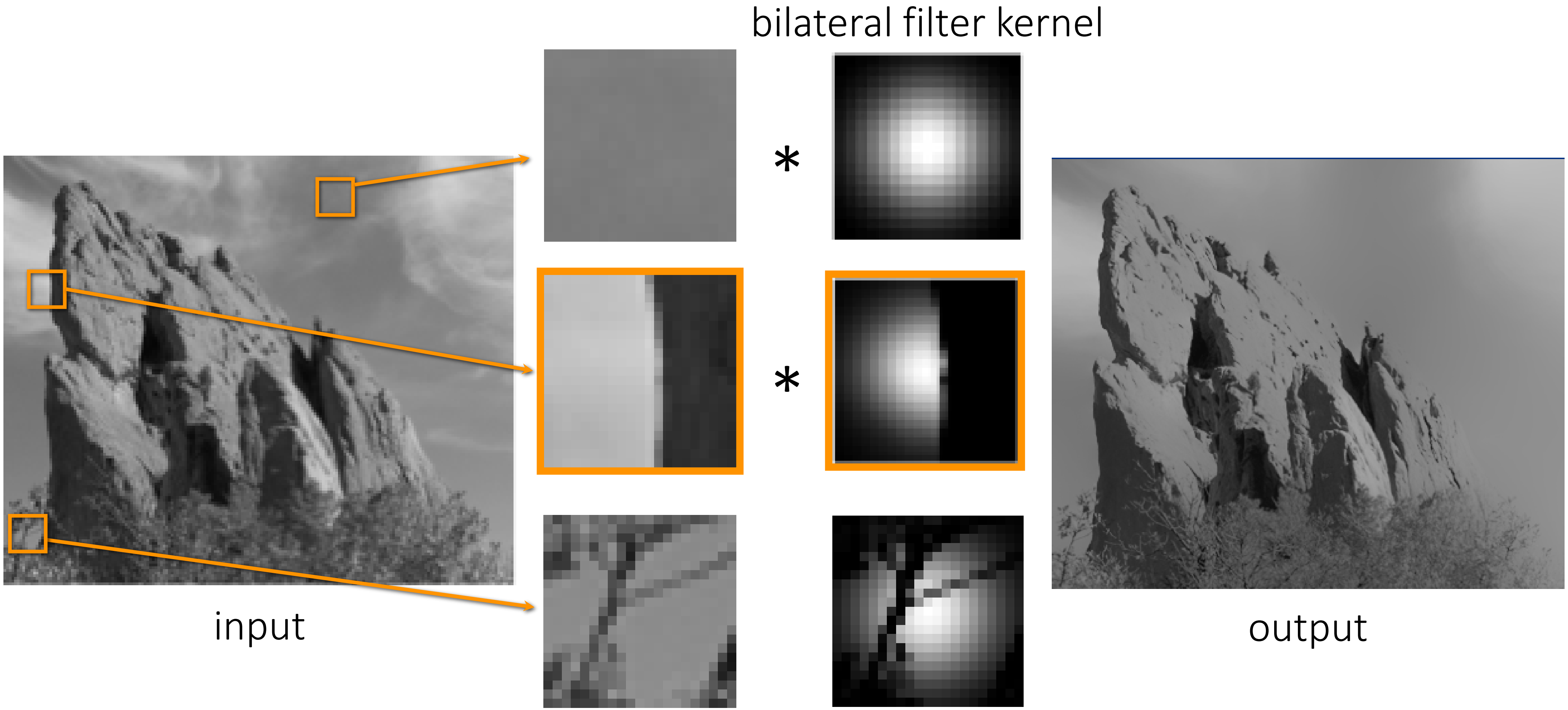
$\sigma_s = 18$



Does the bilateral filter respect all edges?



Does the bilateral filter respect all edges?



Bilateral filter crosses (and blurs) thin edges.

Denoising

noisy input

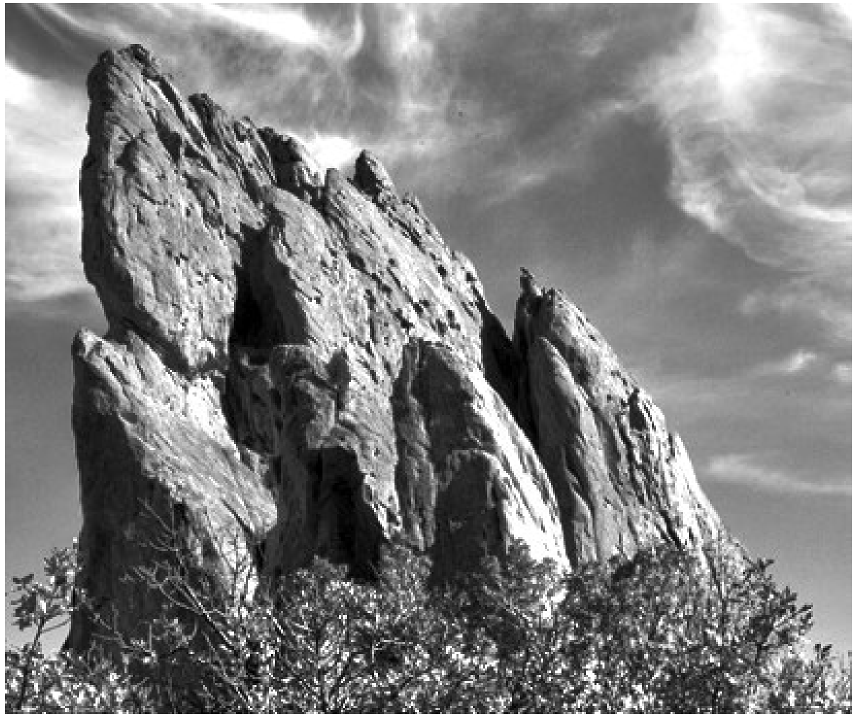
bilateral filtering

median filtering

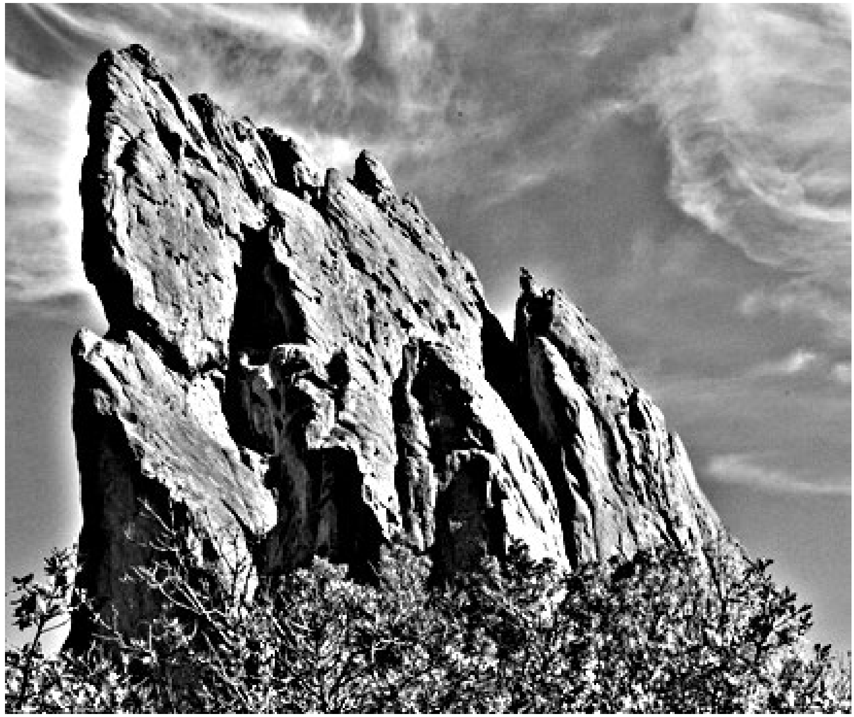
Contrast enhancement

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

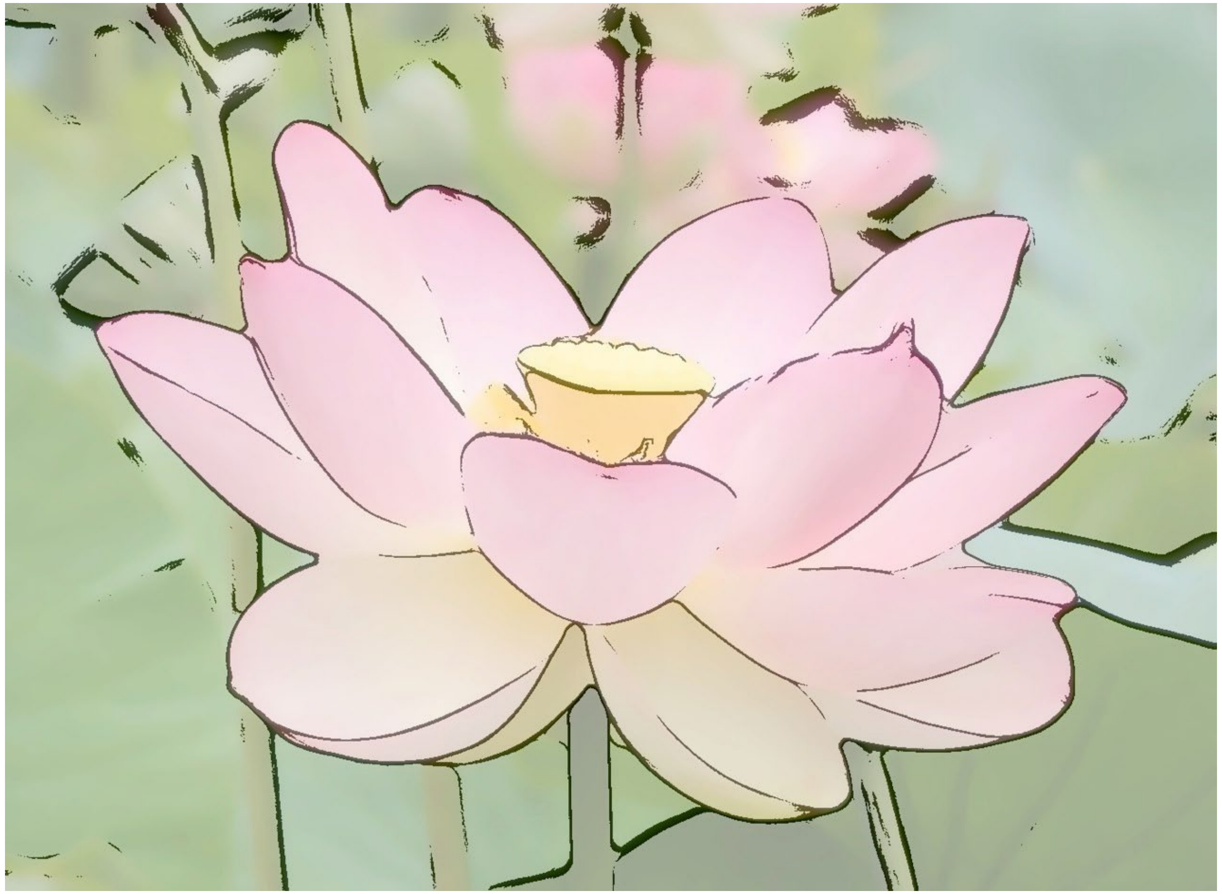
digital pore removal (aka bilateral filtering)

Cartoonization

input

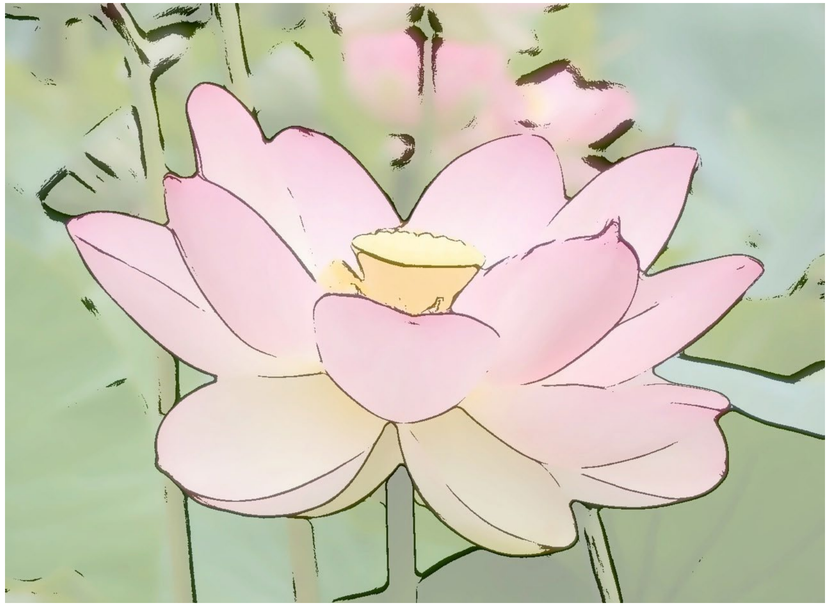
cartoon rendition

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

Local Laplacian Filters: Edge-aware Image Processing with a Laplacian Pyramid

Sylvain Paris
Adobe Systems, Inc.

Samuel W. Hasinoff
Toyota Technological Institute at Chicago and MIT CSAIL

Jan Kautz
University College London

(a) input HDR image tone-mapped with a simple gamma curve (details are compressed)

(b) our pyramid-based tone mapping, set to preserve details without increasing them

(c) our pyramid-based tone mapping, set to strongly enhance the contrast of details

Figure 1: *We demonstrate edge-aware image filters based on the direct manipulation of Laplacian pyramids. Our approach produces high-quality results, without degrading edges or introducing halos, even at extreme settings. Our approach builds upon standard image pyramids and enables a broad range of effects via simple point-wise nonlinearities (shown in corners). For an example image (a), we show results of tone mapping using our method, creating a natural rendition (b) and a more exaggerated look that enhances details as well (c). Laplacian pyramids have previously been considered unsuitable for such tasks, but our approach shows otherwise.*

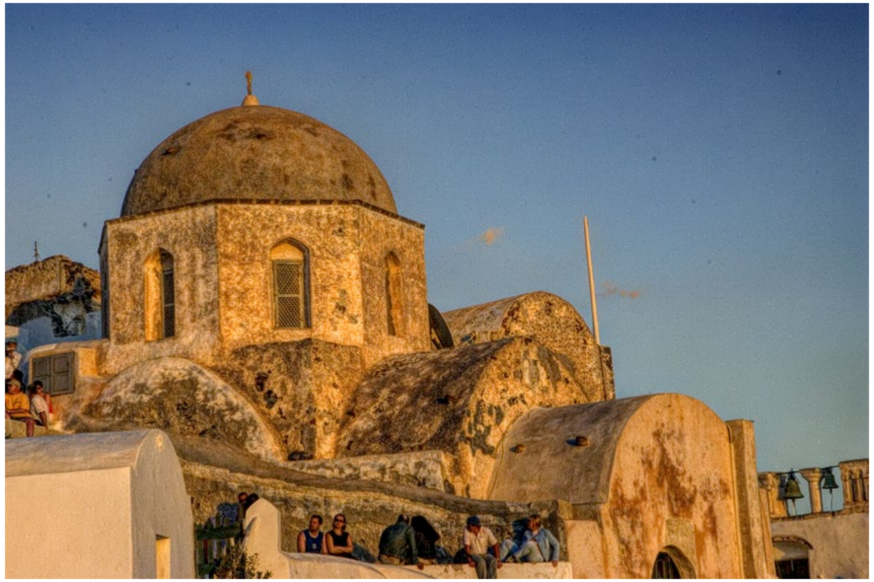
Modern edge-aware filtering: local Laplacian pyramids

Modern edge-aware filtering: local Laplacian pyramids

input

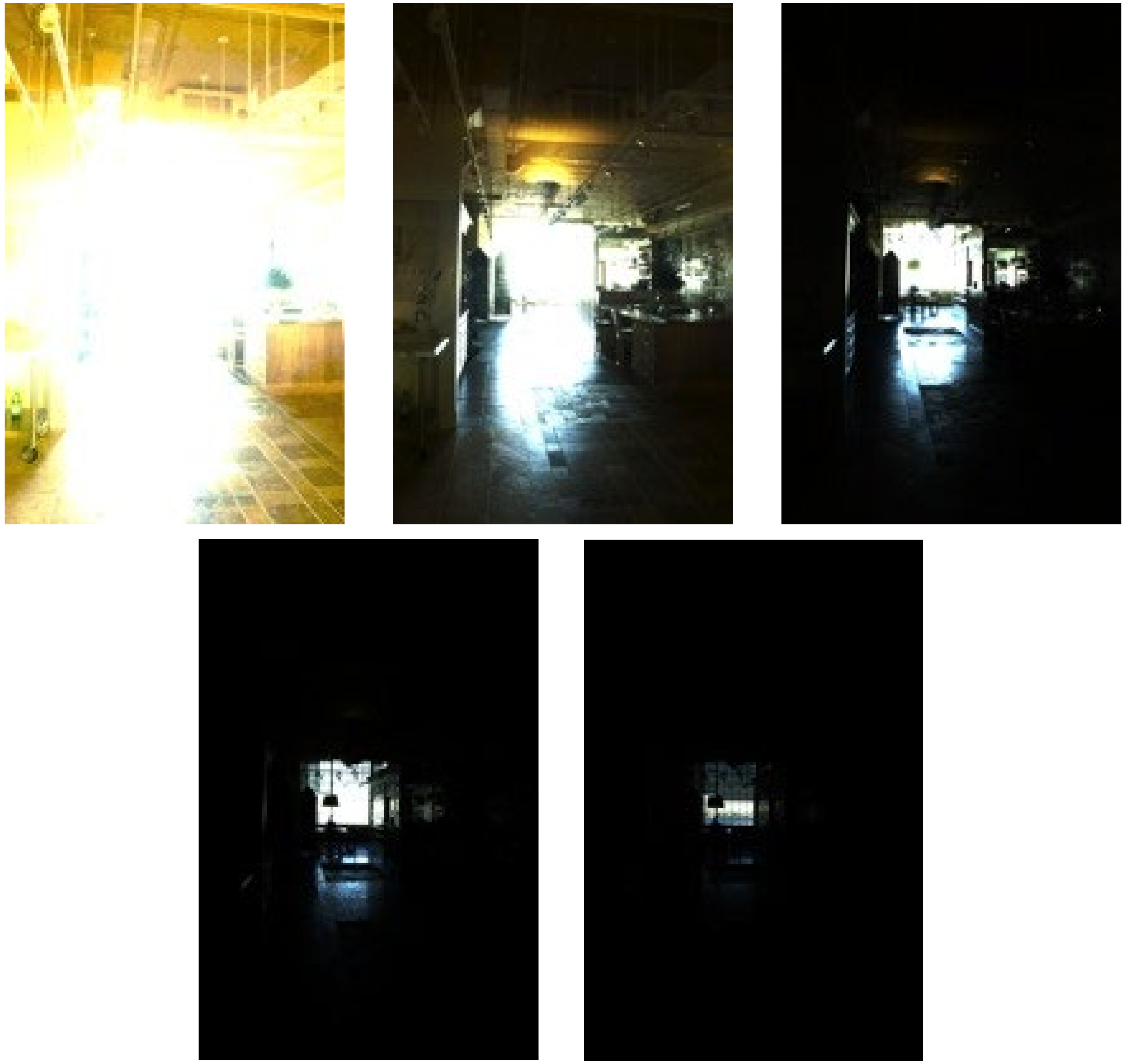
texture increase

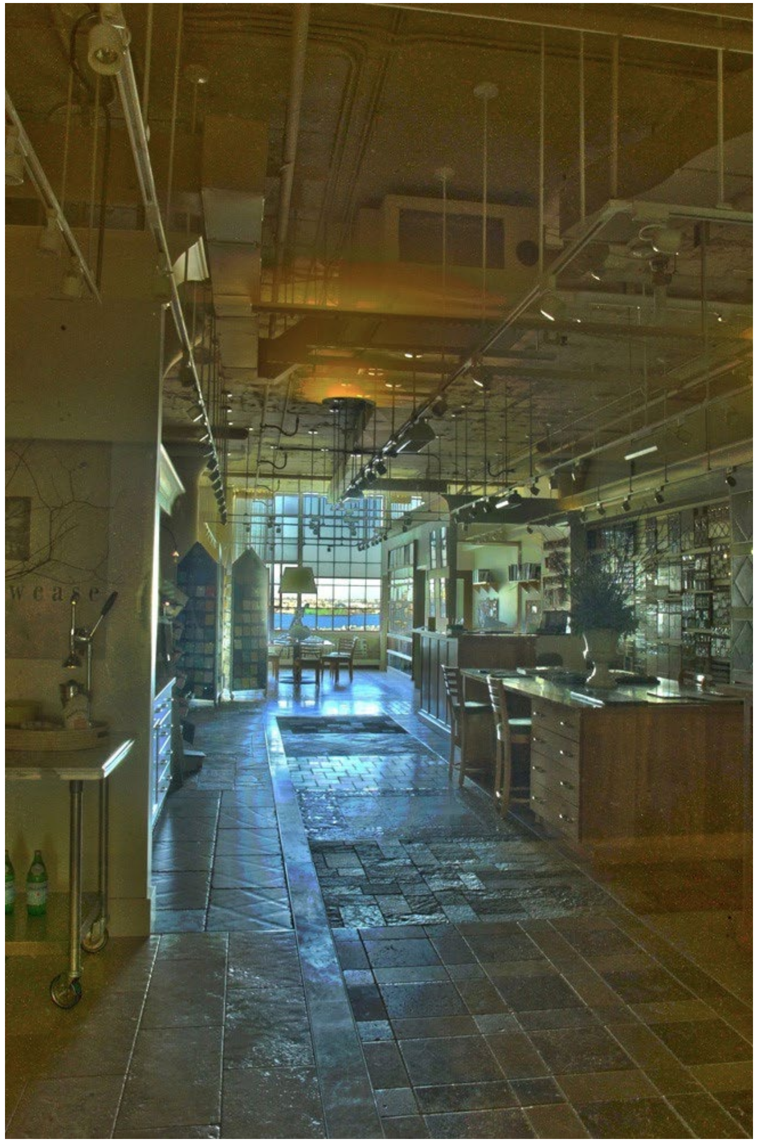
texture decrease



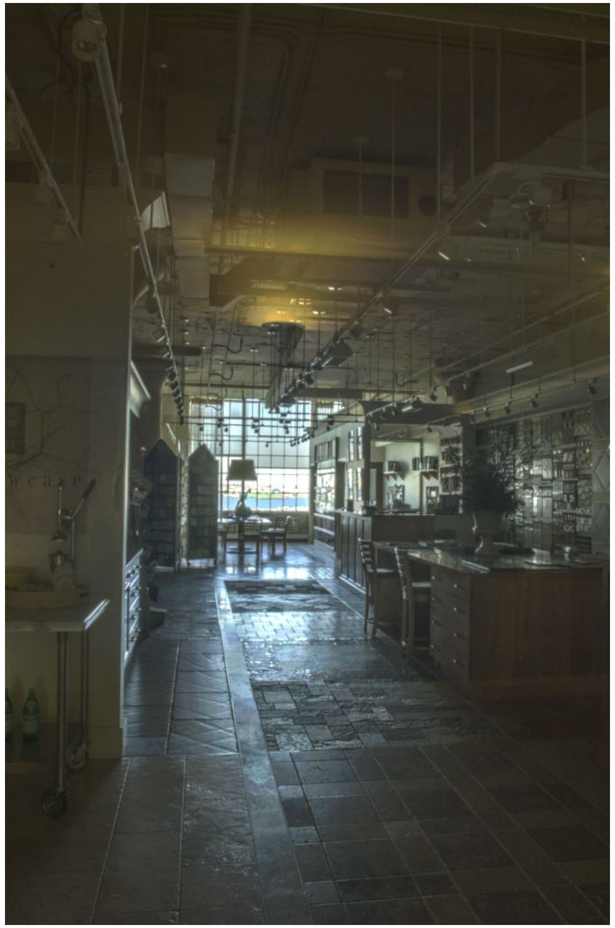
large texture increase

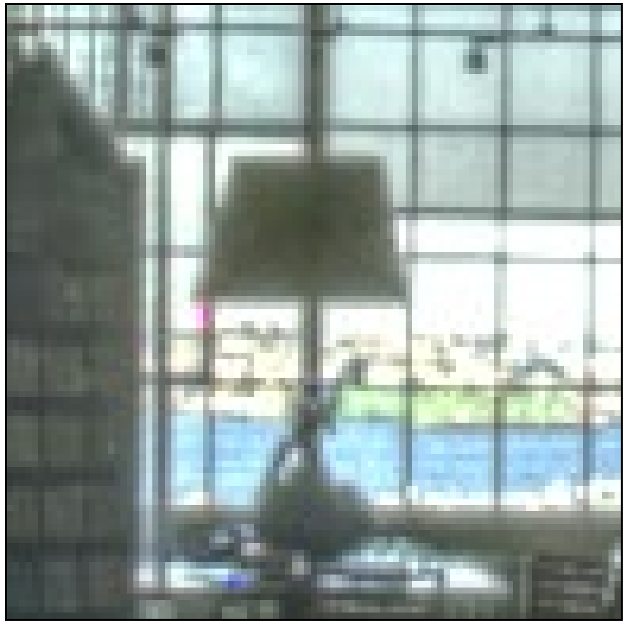
Tonemapping with edge-aware filtering

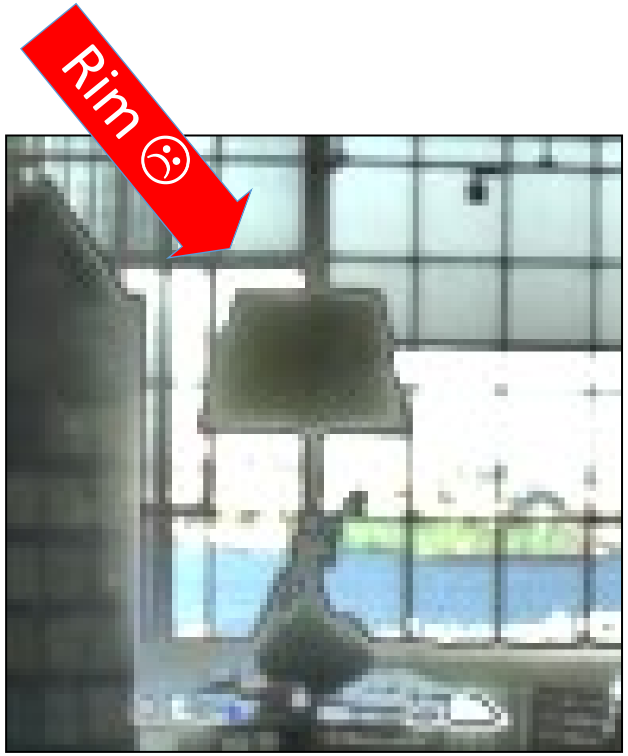


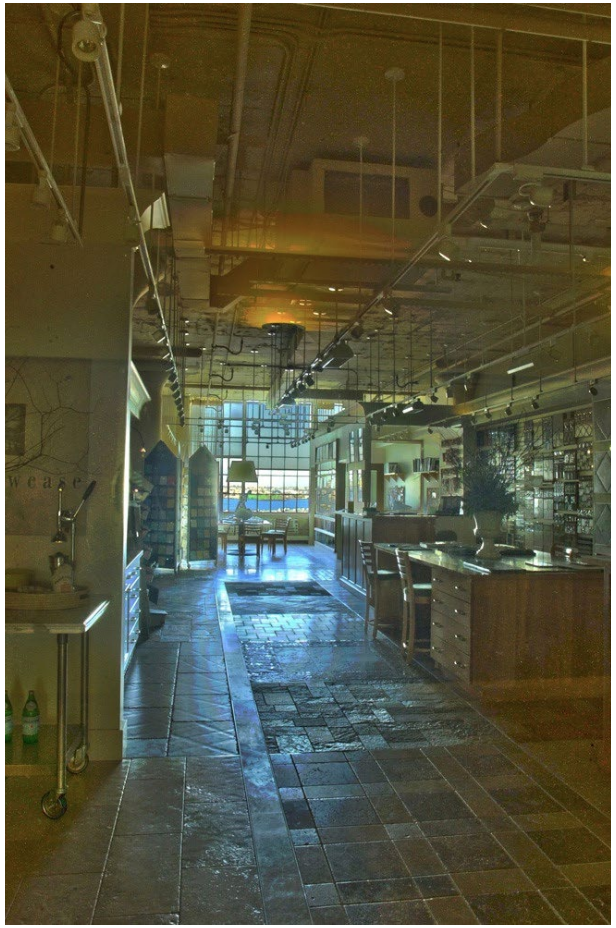


Tonemapping with edge-aware filtering









local Laplacian pyramids

bilateral filter

Modern edge-aware filtering: local Laplacian pyramids

Modern edge-aware filtering: domain transform

Domain Transform for Edge-Aware Image and Video Processing

Eduardo S. L. Gastal* Manuel M. Oliveira†
Instituto de Informática – UFRGS

(a) *Photograph* **(b)** *Edge-aware smoothing* **(c)** *Detail enhancement*

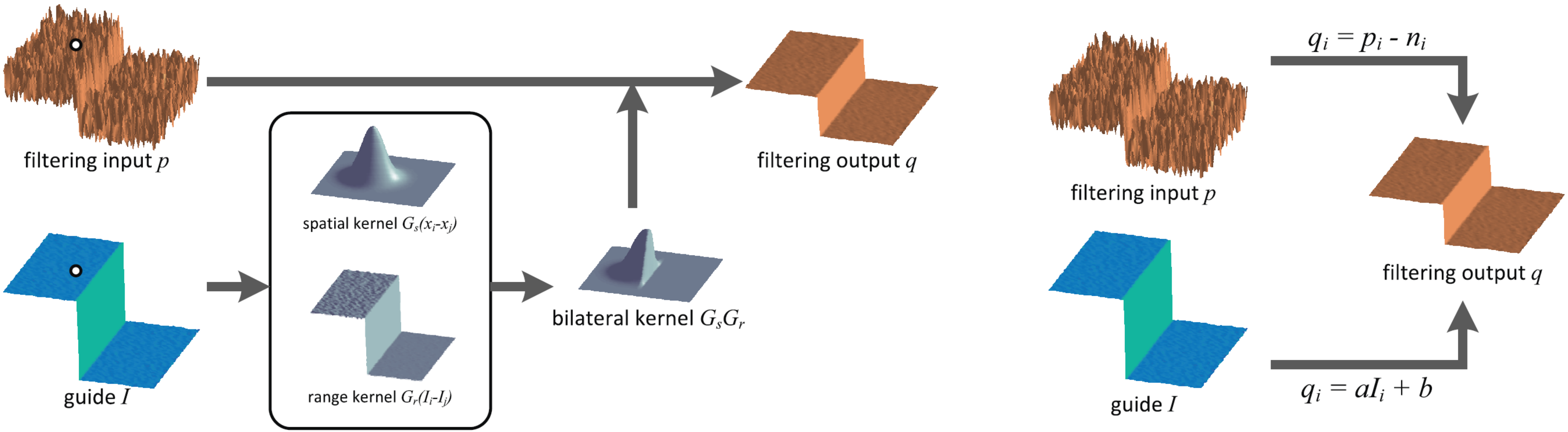
(d) *Stylization* **(e)** *Recoloring* **(f)** *Pencil drawing* **(g)** *Depth-of-field*

Lots of great examples at: https://www.inf.ufrgs.br/~eslgastal/DomainTransform/

Modern edge-aware filtering: guided filter

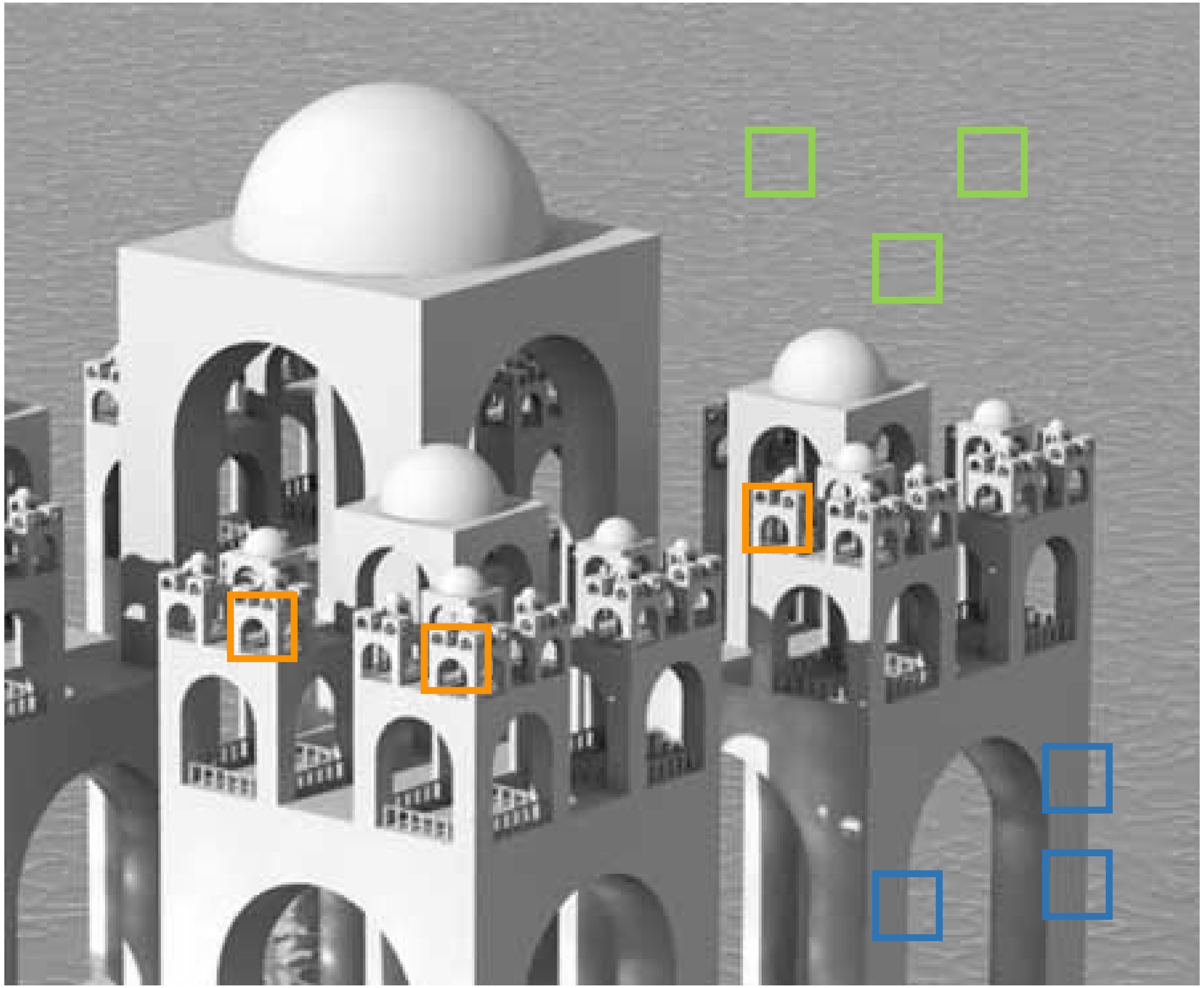
Guided Image Filtering

Kaiming He, *Member, IEEE*, Jian Sun, *Member, IEEE*, and Xiaoou Tang, *Fellow, IEEE*



Non-local means

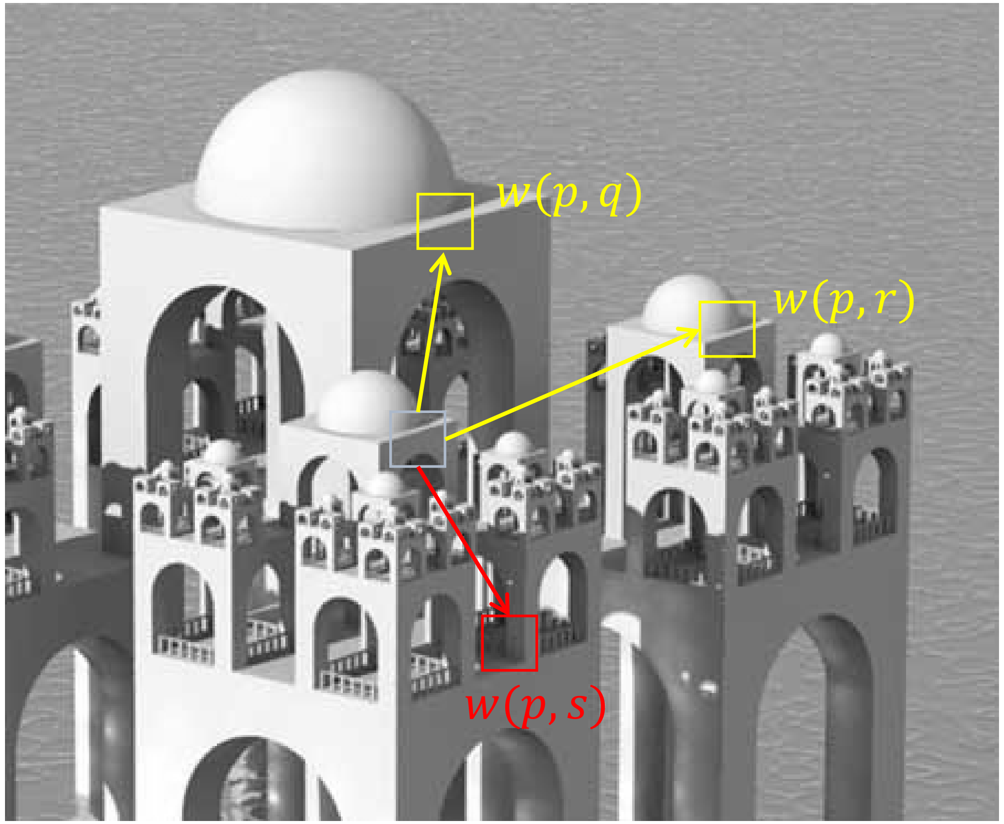
Redundancy in natural images



Non-local means

No need to stop at neighborhood. Instead search *everywhere* in the image.

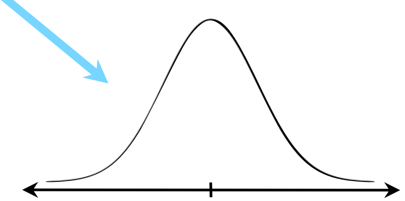
$$\hat{x}(i) = \frac{1}{C_i} \sum_j y(j) \underbrace{e^{-\frac{SSD\left(y(N_i)-y(N_j)\right)}{2\sigma^2}}}_{w(i,j)}$$



Non-local means vs bilateral filtering

Non-local means filtering

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

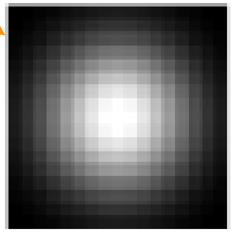


$x = f[m,n] - f[m+k, n+l]$

Intensity range weighting: favor *similar* pixels (patches in case of non-local means)

Bilateral filtering

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



Spatial weighting: favor *nearby* pixels

Everything put together

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

Non-local means

Smooths similar patches no matter how far away
Only depends on *intensity* distance

Denoising example

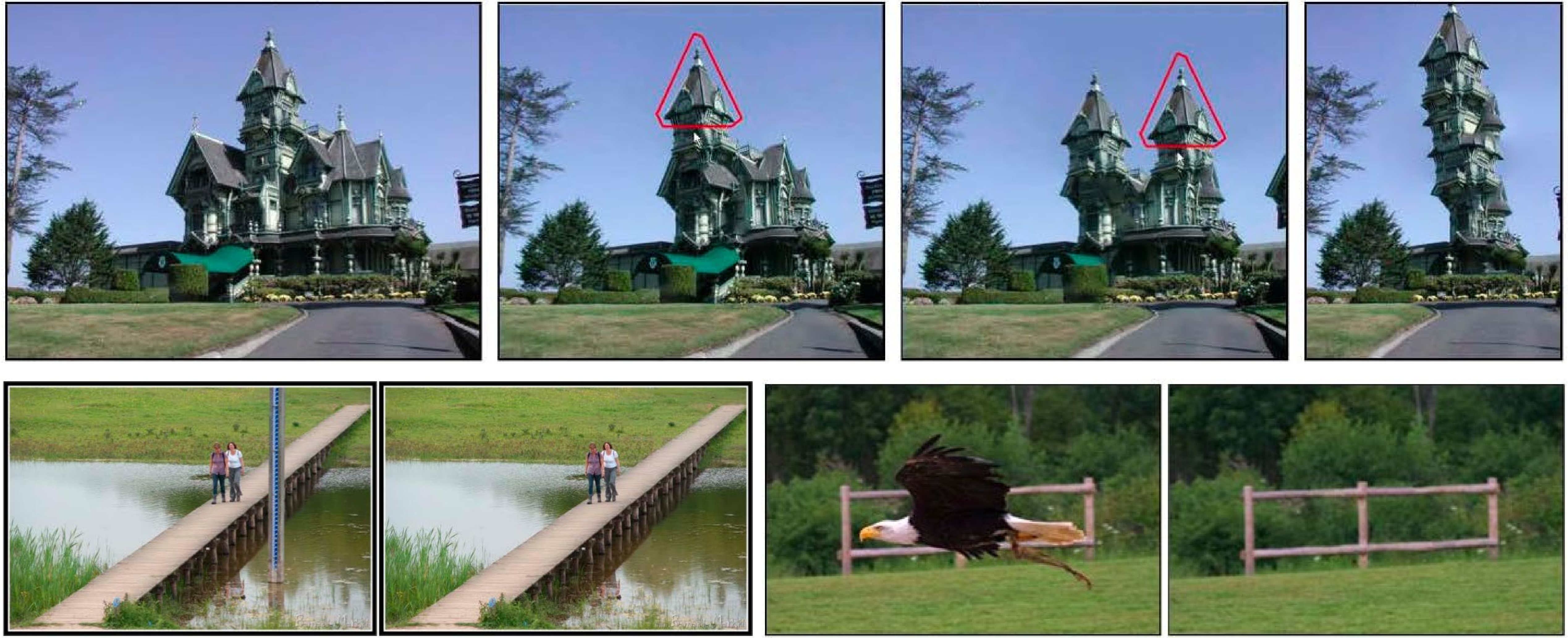
noisy input

Gaussian filtering

bilateral filtering

non-local means

Very general forms of “structural” filtering



We will see more in later lectures.

Is non-local means:

Linear?

Shift-invariant?

Is non-local means:

Linear?

- No.

Shift-invariant?

- No.

Non-local means is not a convolution, and is generally very very challenging to implement efficiently.

Efficient algorithms for non-local means are an active research area.

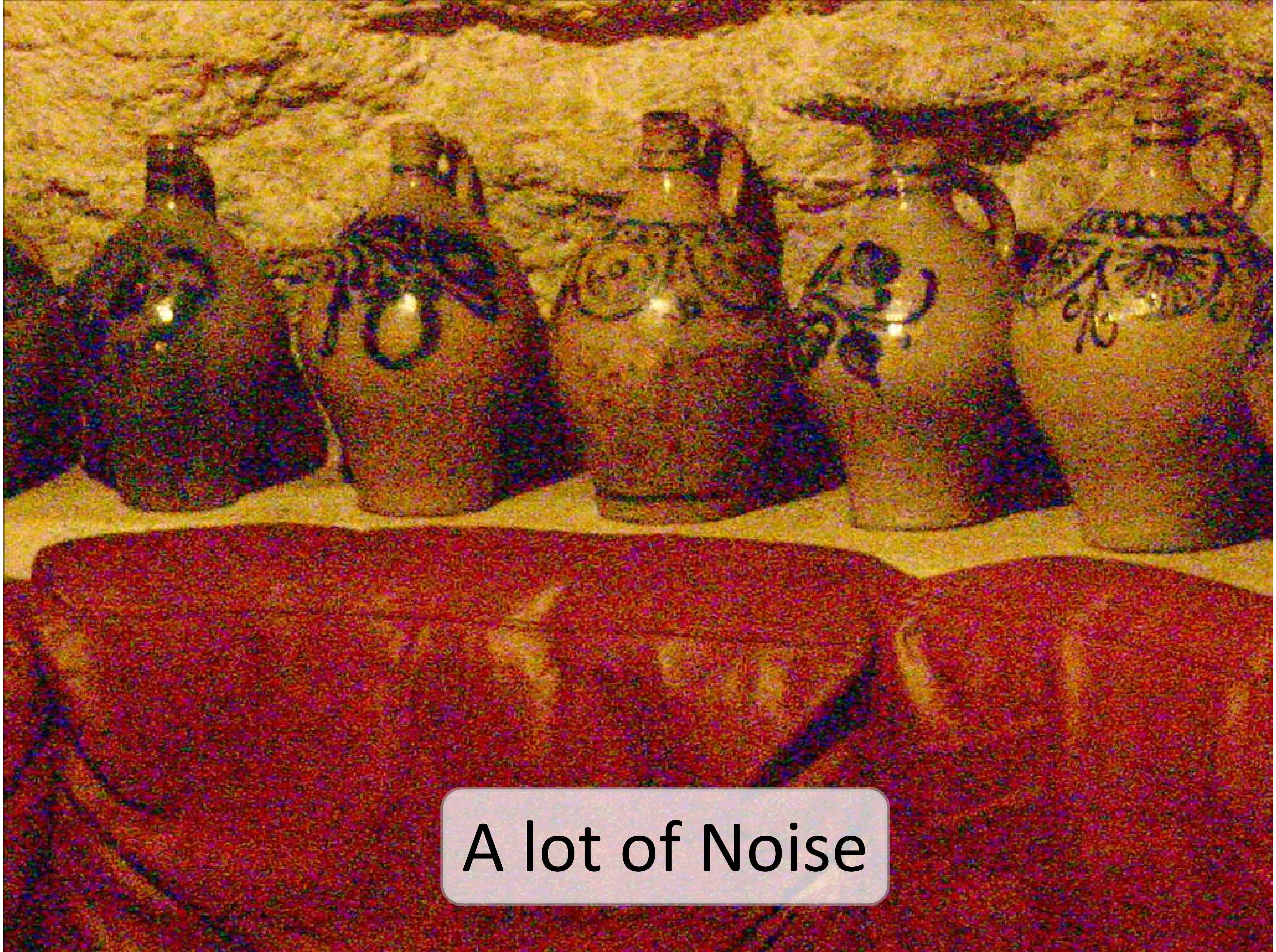
Flash/no-flash photography

Red Eye


Unflattering Lighting

Motion Blur


Noise



A lot of Noise


Ruined Ambiance

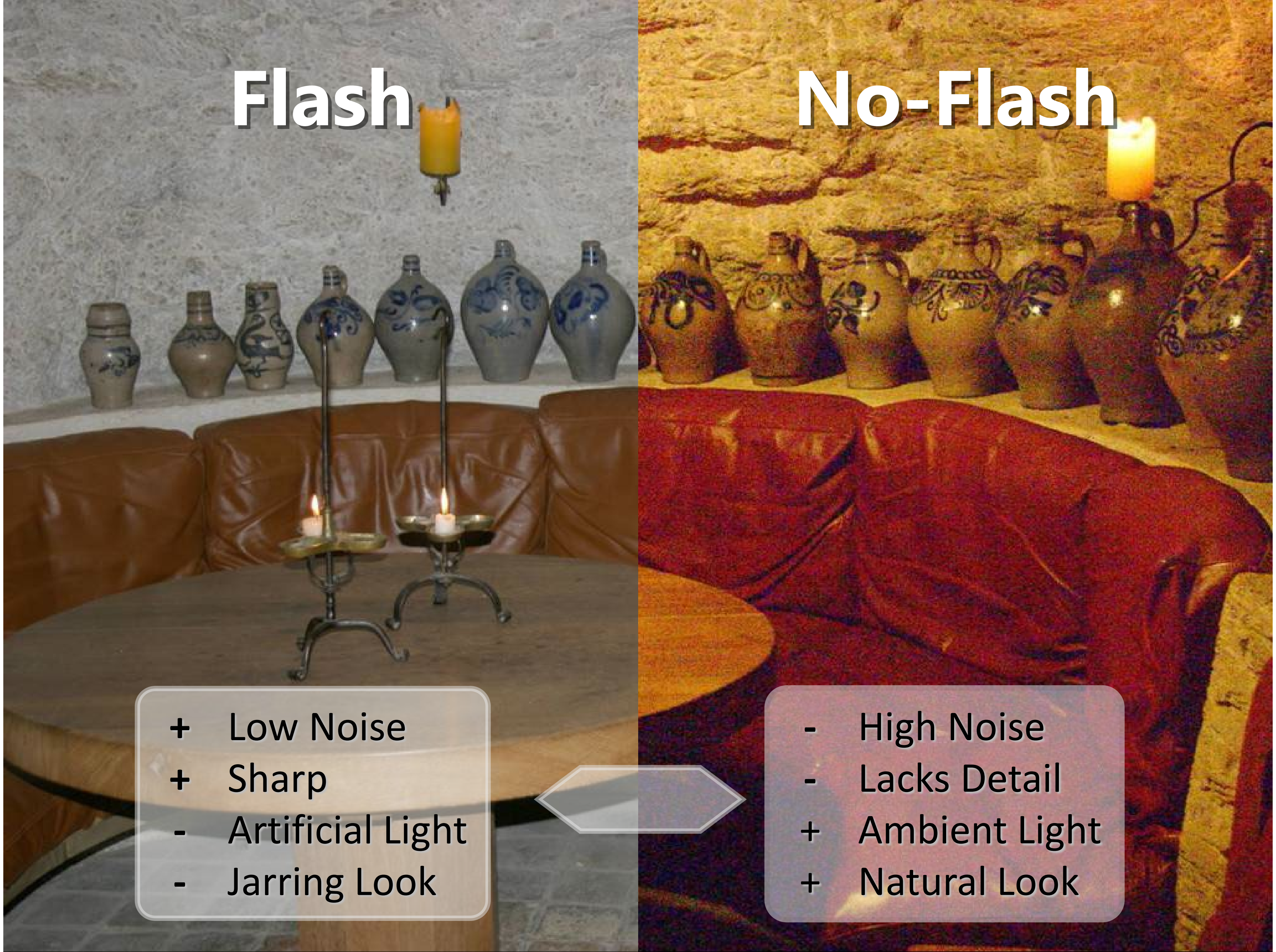

Flash
No-Flash
+ Low Noise
+ Sharp
- Artificial Light
- Jarring Look
- High Noise
- Lacks Detail
+ Ambient Light
+ Natural Look

Image acquisition

Image acquisition

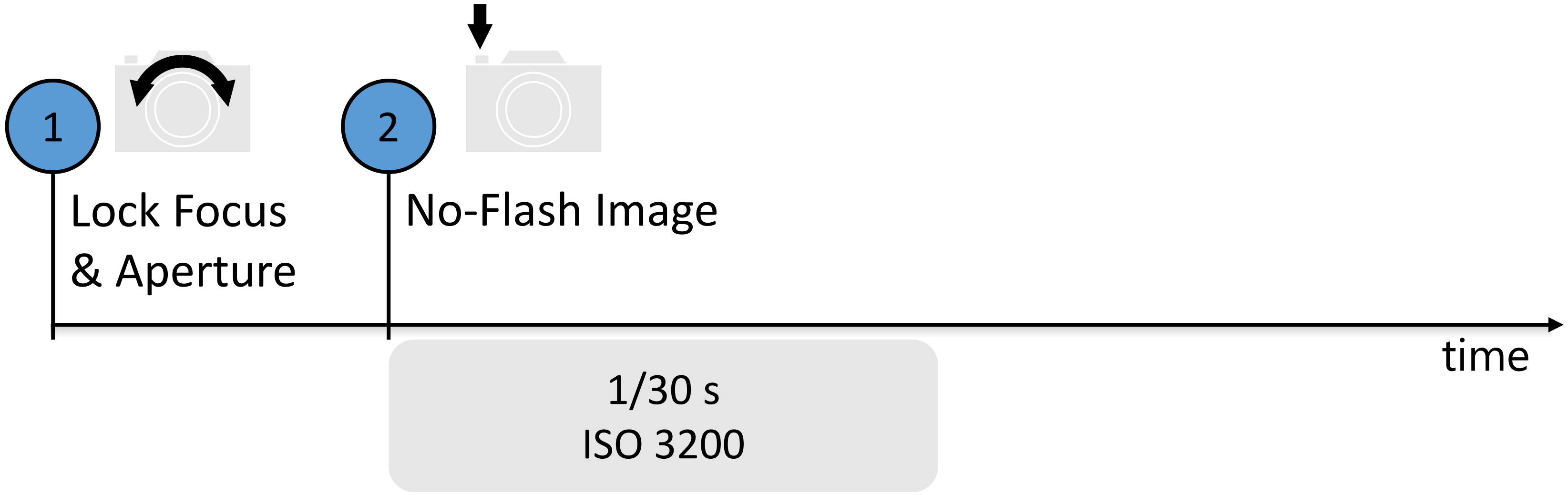
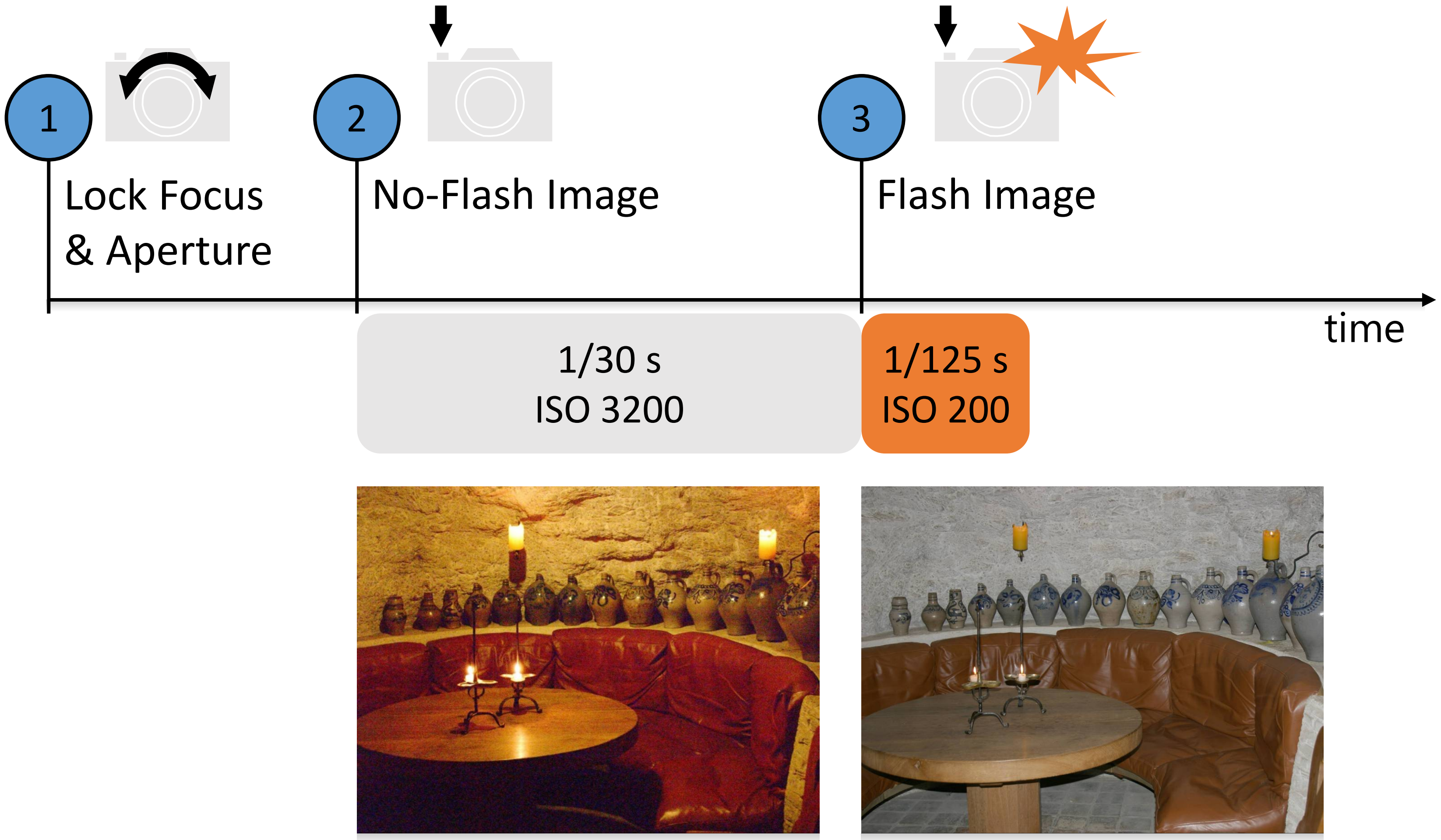


Image acquisition




Denoising Result


No-Flash


Denoising Result

Key idea

Denoise the no-flash image while maintaining the edge structure of the flash image

- How would you do this using the image editing techniques we’ve learned about?

Joint bilateral filtering

Denoising with bilateral filtering

noisy input

bilateral filtering

median filtering

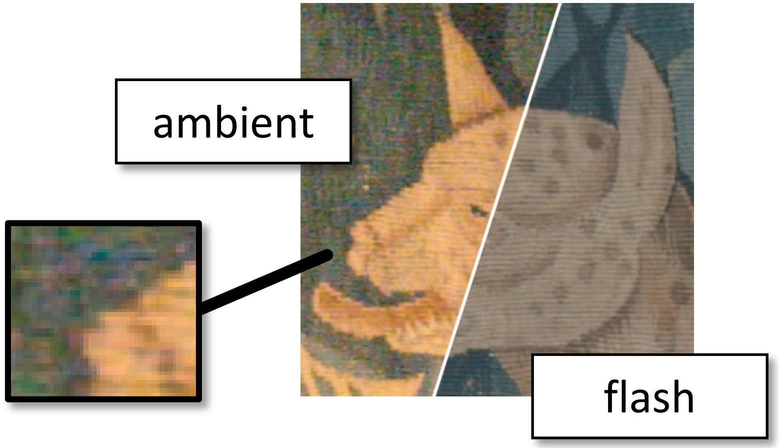
Denoising with bilateral filtering

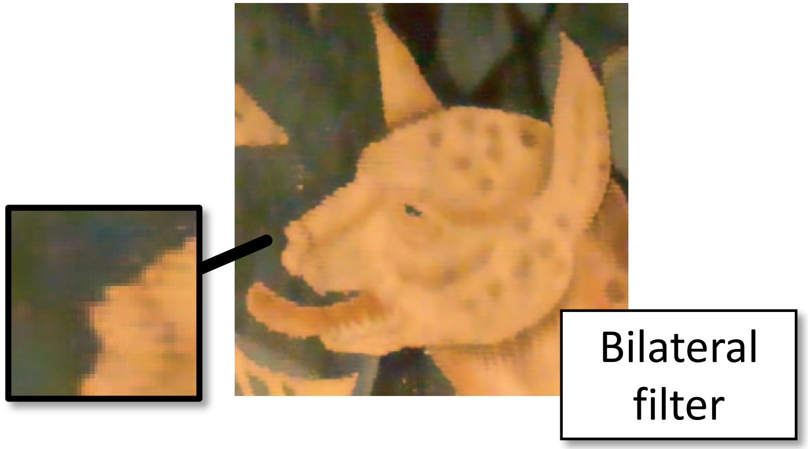
spatial kernel

$$A^{Base}_{p(col)} = \frac{1}{k(p(col))} \sum_{p' \in \Omega} \boxed{g_d(|p - p'|)} \boxed{g_r\left(A_{p(col)} - A_{p'(col)}\right)} A_{p'(col)}$$

intensity kernel

- However, results still have noise or blur (or both)





Denoising with joint bilateral filtering

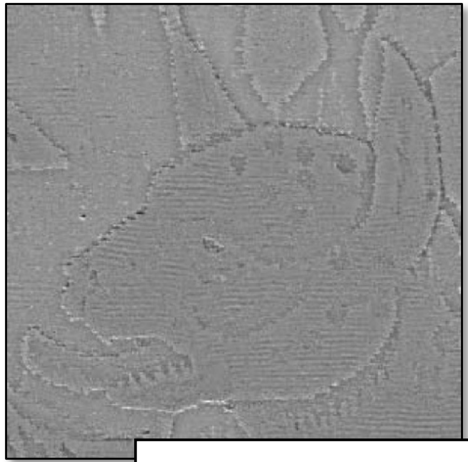
$$A^{NR}_{p(col)} = \frac{1}{k(p(col))} \sum_{p' \in \Omega} g_d(|p - p'|) \, g_r\left(\boldsymbol{F}_{p(col)} - \boldsymbol{F}_{p'(col)}\right) A_{p'(col)}$$

- In the flash image there are many more *details*
- Use the flash image F to find edges

Denoising with joint bilateral filtering

$$A^{NR}_{p(col)} = \frac{1}{k(p(col))} \sum_{p' \in \Omega} g_d(|p - p'|)\, g_r\left(\boldsymbol{F}_{\boldsymbol{p}(\boldsymbol{col})} - \boldsymbol{F}_{\boldsymbol{p}'(\boldsymbol{col})}\right) A_{p'(col)}$$

Bilateral filter



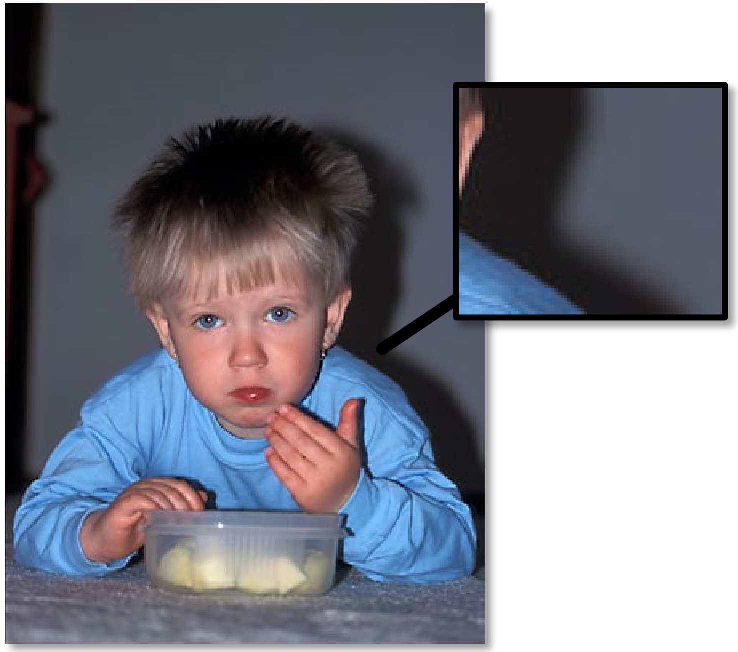
The difference

Joint Bilateral filter

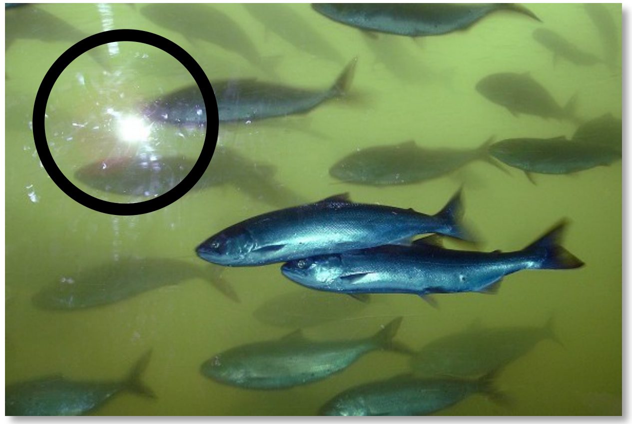
Not all edges in the flash image are real

Can you think of any types of edges that may exist in the flash image but not the ambient one?

Not all edges in the flash image are real



shadows



specularities

- May cause over- or under-blur in joint bilateral filter
- We need to eliminate their effect

Detecting shadows

- **Observation:** the pixels in the flash shadow should be similar to the ambient image.
- Not identical:
  1. Noise.
  2. Inter-reflected flash.
- Compute a *shadow mask.*
- Take pixel *p* if $F_{p(col)}^{Lin} - A_{p(col)}^{Lin} \leq \tau_{Shadow}$
- τ_{Shadow} is manually adjusted
- Mask is *smoothed* and *dilated*

Detecting specularities

- Take pixels where sensor input is close to maximum (very bright).
  - Over fixed threshold τ_{Spec}
- Create a *specularity mask.*
- Also smoothed.
- **M** – the combination of shadow and specularity masks:

Where $M_p=1$, we use A^{Base}. For other pixels we use A^{NR}.

Detail transfer

- Denoising cannot add details *missing* in the ambient image
- Exist in flash image because of high SNR
- We use a *quotient image*:

$$F_{p(col)}^{Detail} = \frac{F_{p(col)} + \varepsilon}{F_{p(col)}^{Base} + \varepsilon}$$

Reduces the effect of noise in F

Bilateral filtered

- Multiply with A^{NR} to add the details
- Masked in the same way

Why does this quotient image make sense for detail?

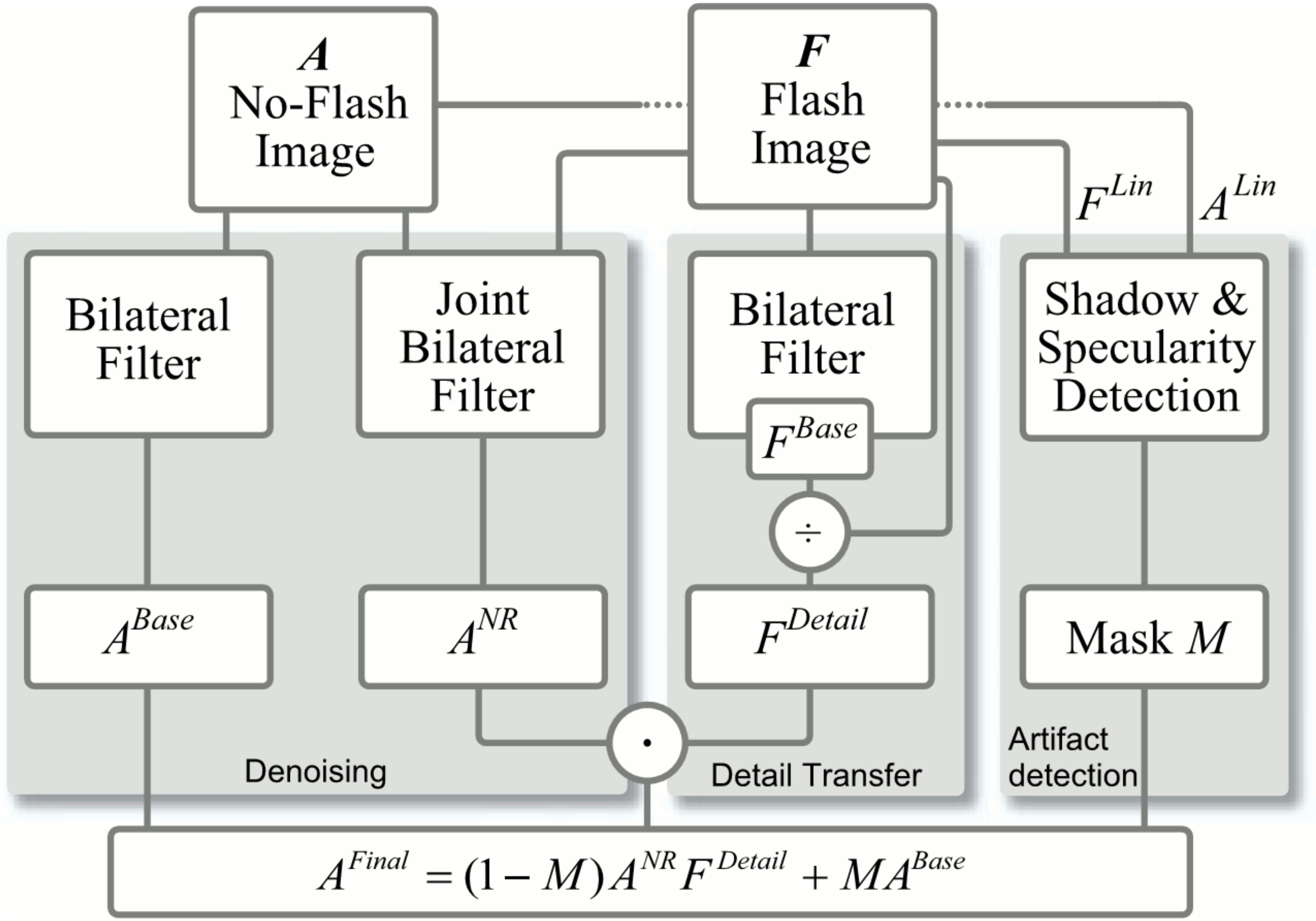
Detail transfer

- Denoising cannot add details *missing* in the ambient image
- Exist in flash image because of high SNR
- We use a *quotient image*:

$$F^{Detail}_{p(col)} = \frac{F_{p(col)} + \varepsilon}{F^{Base}_{p(col)} + \varepsilon}$$

Reduces the effect of noise in F

Full pipeline



Demonstration

ambient-only

joint bilateral and detail transfer


Flash


No-Flash


No-Flash


Result


Flash


No-Flash


No-Flash


Result


Flash


No-Flash


Flash


No-Flash

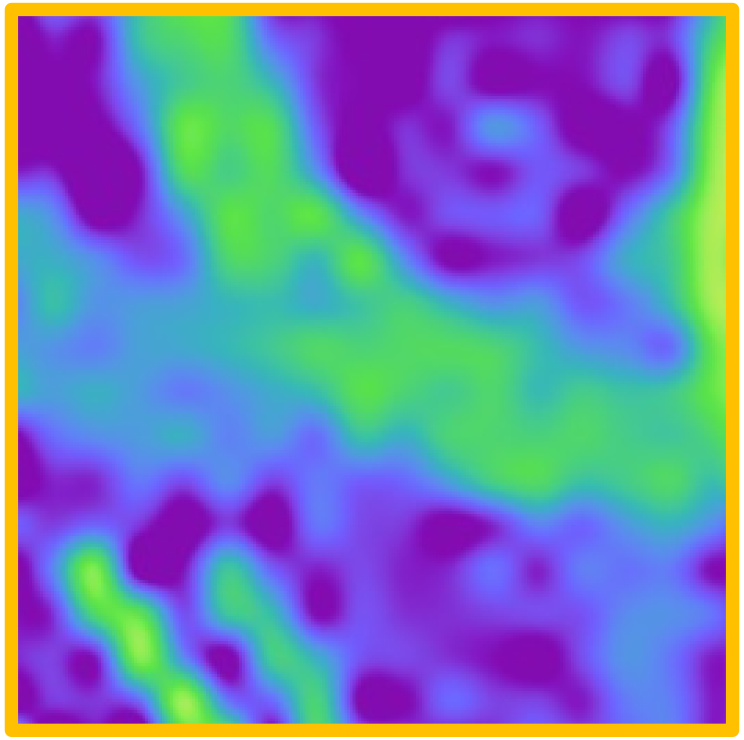

Result

Edge-aware depth denoising

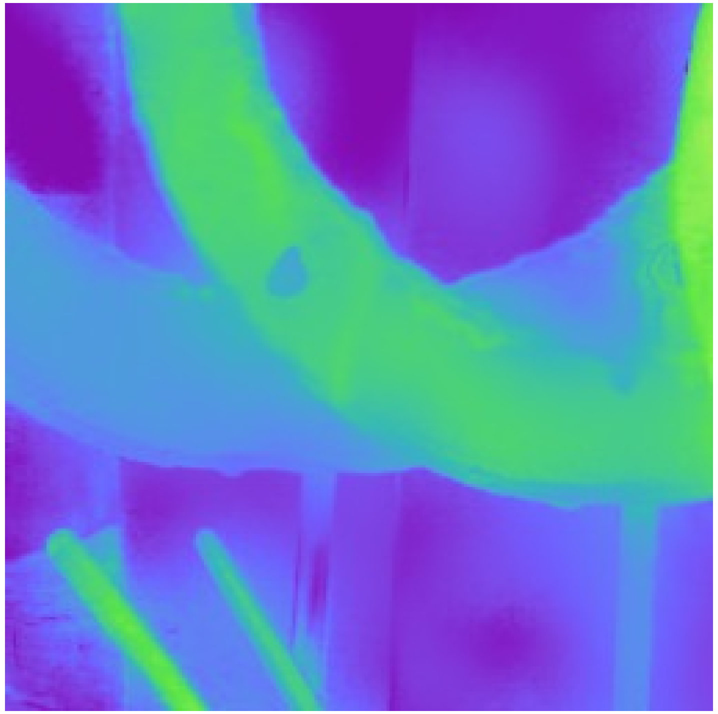
$$A_{p(col)} = \frac{1}{k(p(col))} \sum_{p' \in \Omega} g_d(|p - p'|)\, g_r\left(F_{p(col)} - F_{p'(col)}\right) A_{p'(col)}$$

Use joint bilateral filtering, with the input image as guide.

One of two input images



Depth from disparity

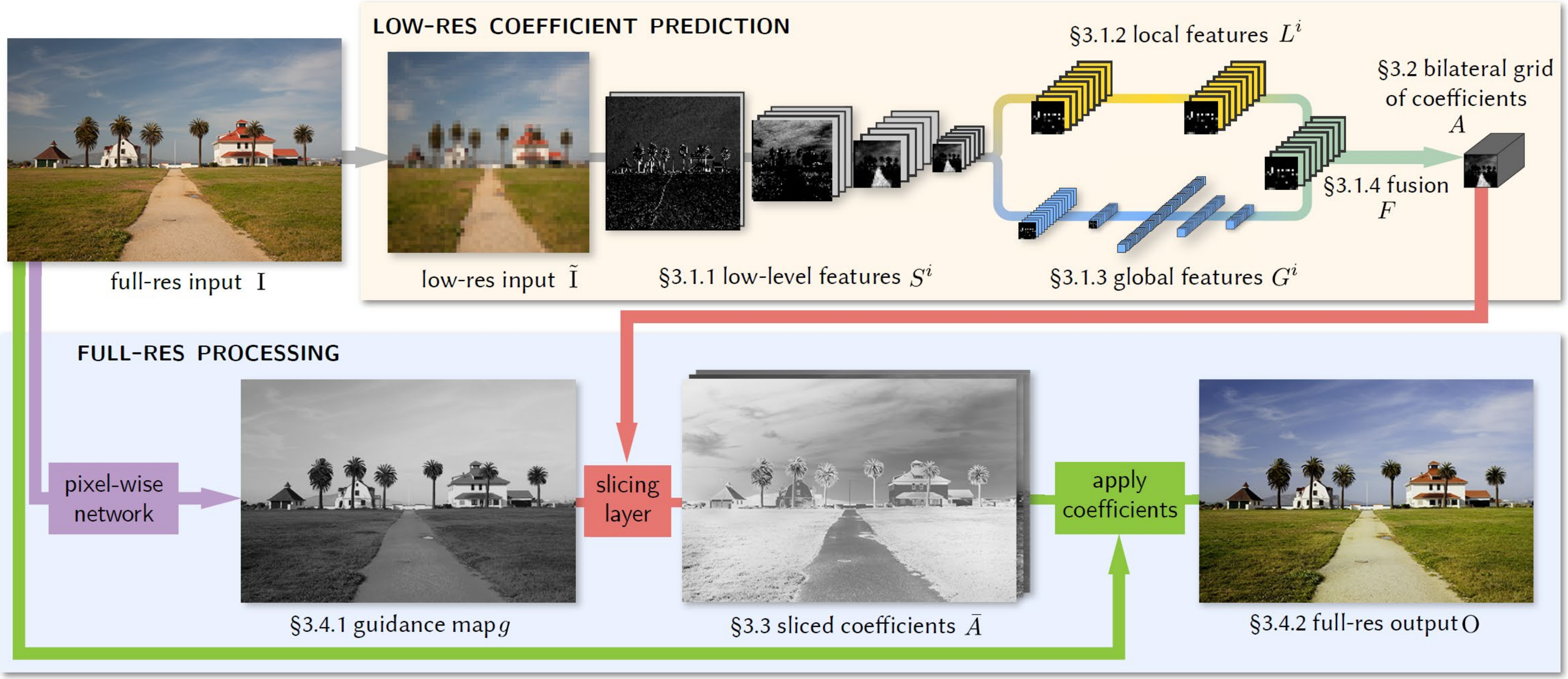


Guided filtering

Other applications of joint bilateral filtering

Deep Bilateral Learning for Real-Time Image Enhancement

MICHAËL GHARBI, MIT CSAIL
JIAWEN CHEN, Google Research
JONATHAN T. BARRON, Google Research
SAMUEL W. HASINOFF, Google Research
FRÉDO DURAND, MIT CSAIL / Inria, Université Côte d'Azur



References

Basic reading:

- 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., "Bilateral Filtering: Theory and Applications," Foundations and Trends® in Computer Graphics and Vision 2009.
  A comprehensive review of bilateral filtering.
- Paris et al., "A Gentle Introduction to the Bilateral Filter and Its Applications," SIGGRAPH 2007-08, CVPR 2008, https://people.csail.mit.edu/sparis/bf_course/
  Short course on the bilateral filter, including discussion of fast implementations.
- Petschnigg et al., "Digital photography with flash and no-flash image pairs," SIGGRAPH 2004.
- Eisemann and Durand, "Flash Photography Enhancement via Intrinsic Relighting," SIGGRAPH 2004.
  The first two papers exploring the idea of photography with flash and no-flash pairs using the joint bilateral filter.

Additional reading:

- Chen et al., "Real-time edge-aware image processing with the bilateral grid," SIGGRAPH 2007.
- Paris and Durand, "A Fast Approximation of the Bilateral Filter Using a Signal Processing Approach," IJCV 2009.
  Two papers on acceleration techniques for the bilateral filer.
- 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.
- Buades et al., "Nonlocal Image and Movie Denoising," IJCV 2008.
  The journal version of the original non-local means paper.
- Barnes et al., "PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing," SIGGRAPH 2009.
  A paper on a very efficient implementation of non-local means, including a few amazing applications focusing on creative manipulation of images.
- He et al., "Guided image filtering," PAMI 2013.
- Gastal and Oliveira, "Domain Transform for Edge-Aware Image and Video Processing," SIGGRAPH 2011.
  The papers introducing the two different types of edge-aware filtering we mentioned.
- Gharbi et al., "Deep Bilateral Learning for Real-Time Image Enhancement," SIGGRAPH 2017.
  Learning image transformations using bilateral filtering.
- Barron et al., "Fast bilateral-space stereo for synthetic defocus," CVPR 2015.
- Barron and Poole, "The fast bilateral solver," ECCV 2016.
  The above two papers show how to combine edge-aware filtering (and bilateral filtering in particular) with disparity matching for robust stereo. The first paper also shows how the resulting depth maps can be used to create synthetic defocus blur.