

Bilateral Filtering, and Non-local Means Denoising

Erkut Erdem

Acknowledgement: The slides are adapted from the course “A Gentle Introduction to Bilateral Filtering and its Applications” given by Sylvain Paris, Pierre Kornprobst, Jack Tumblin, and Frédo Durand (http://people.csail.mit.edu/sparis/bf_course/)

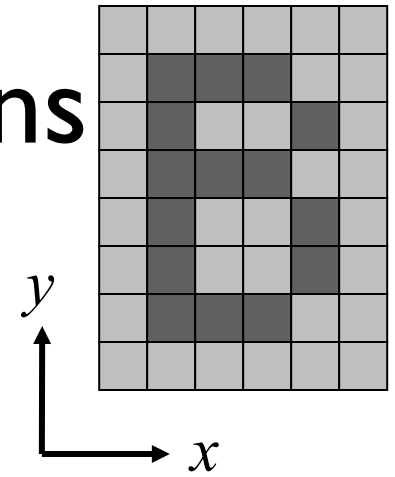
Review - Smoothing and Edge Detection

- While eliminating noise via smoothing, we also lose some of the (important) image details.
 - Fine details
 - Image edges
 - etc.
- What can we do to preserve such details?
 - Use edge information during denoising!
 - This requires a definition for image edges.

Chicken-and-egg dilemma!

- Edge preserving image smoothing

Notation and Definitions

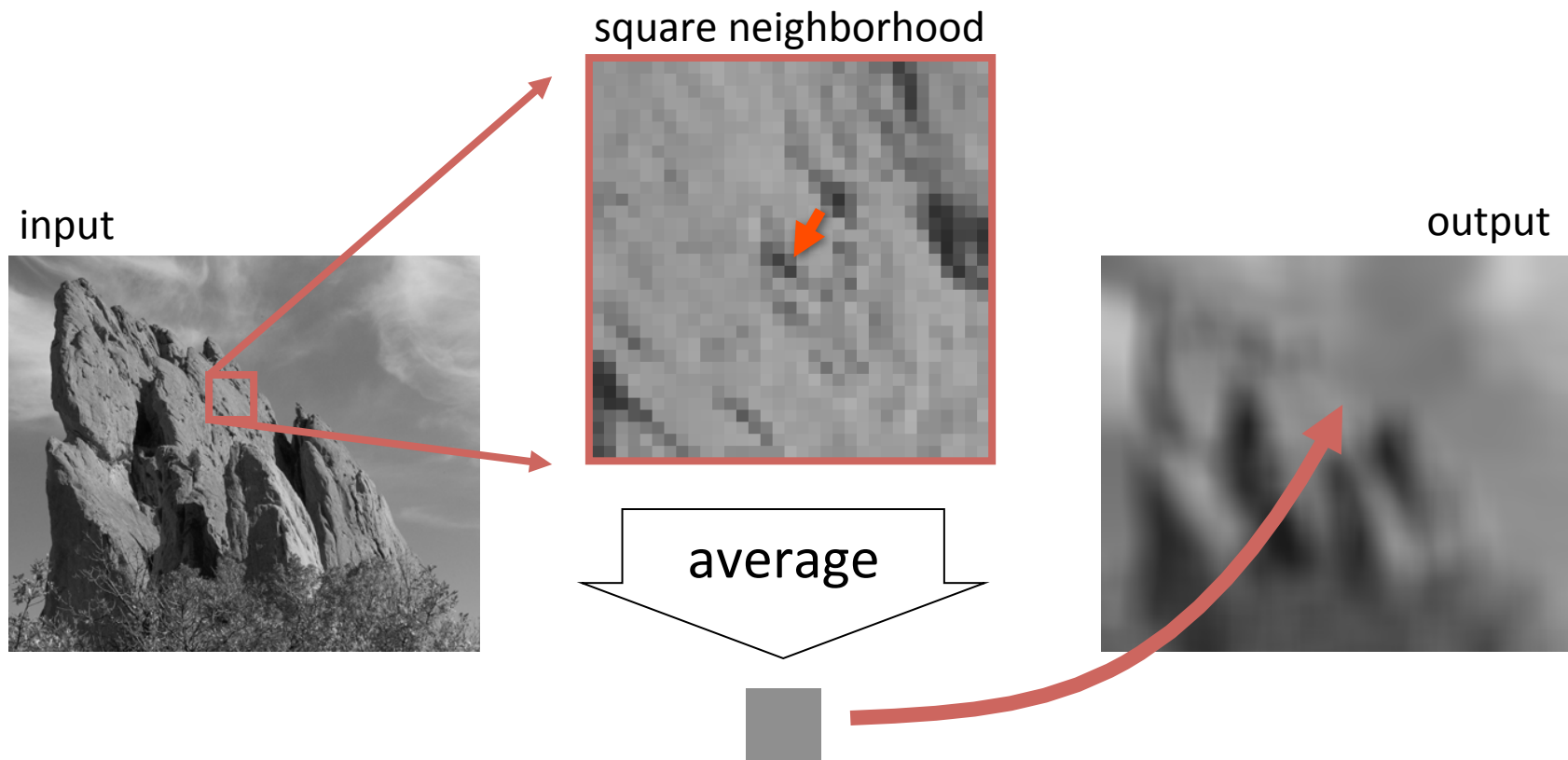


- Image = 2D array of pixels
- Pixel = intensity (scalar) or color (3D vector)
- $I_{\mathbf{p}}$ = value of image I at position: $\mathbf{p} = (p_x, p_y)$
- $F [I]$ = output of filter F applied to image I

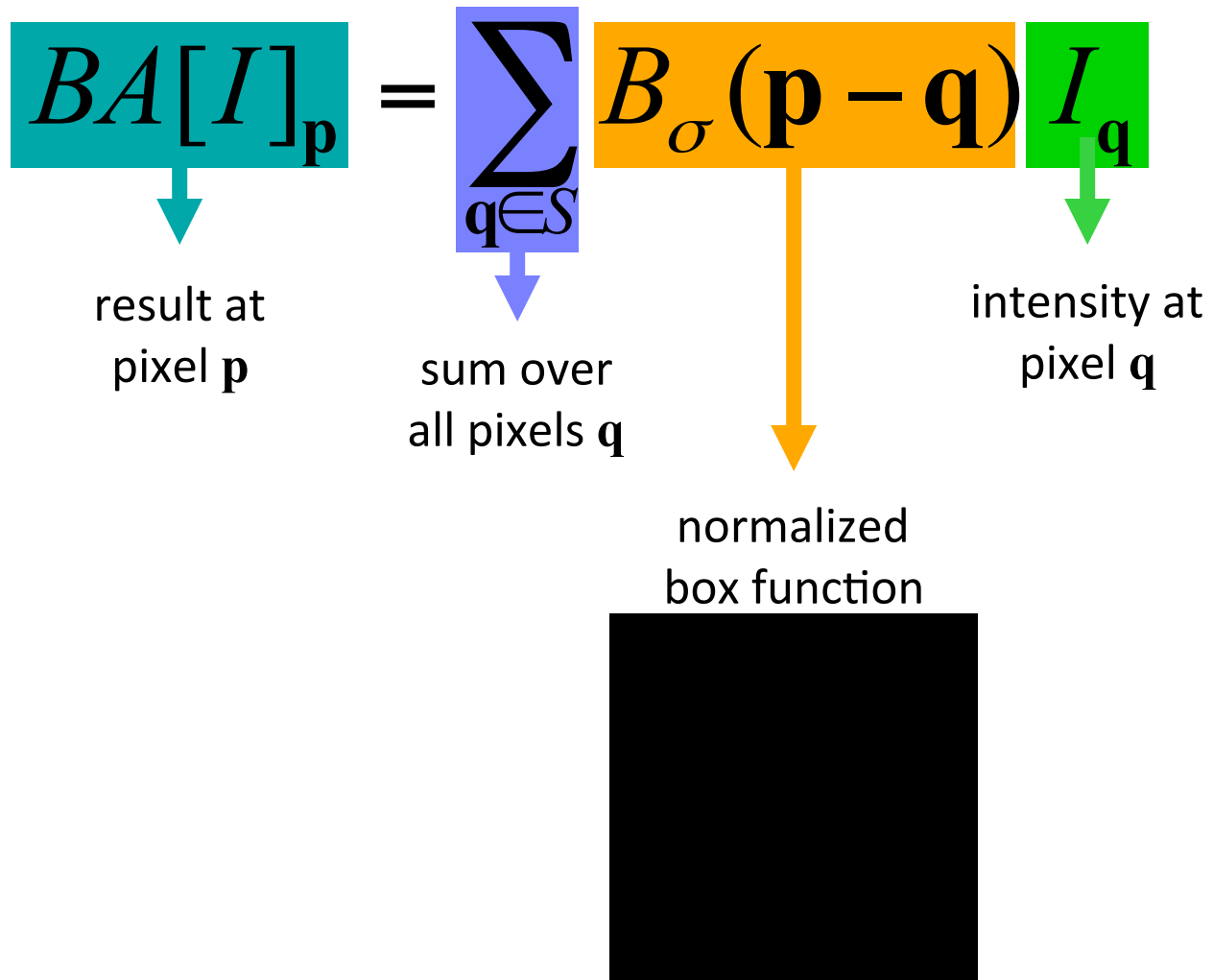
Strategy for Smoothing Images

- Images are not smooth because adjacent pixels are different.
- Smoothing = making adjacent pixels look more similar.
- Smoothing strategy
pixel \rightarrow average of its neighbors

Box Average



Equation of Box Average



Square Box Generates Defects

- Axis-aligned streaks
- Blocky results

input



output

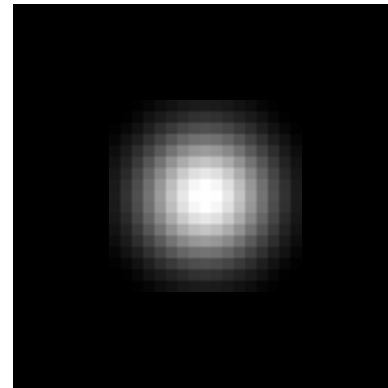


Strategy to Solve these Problems

- Use an isotropic (*i.e.* circular) window.
- Use a window with a smooth falloff.

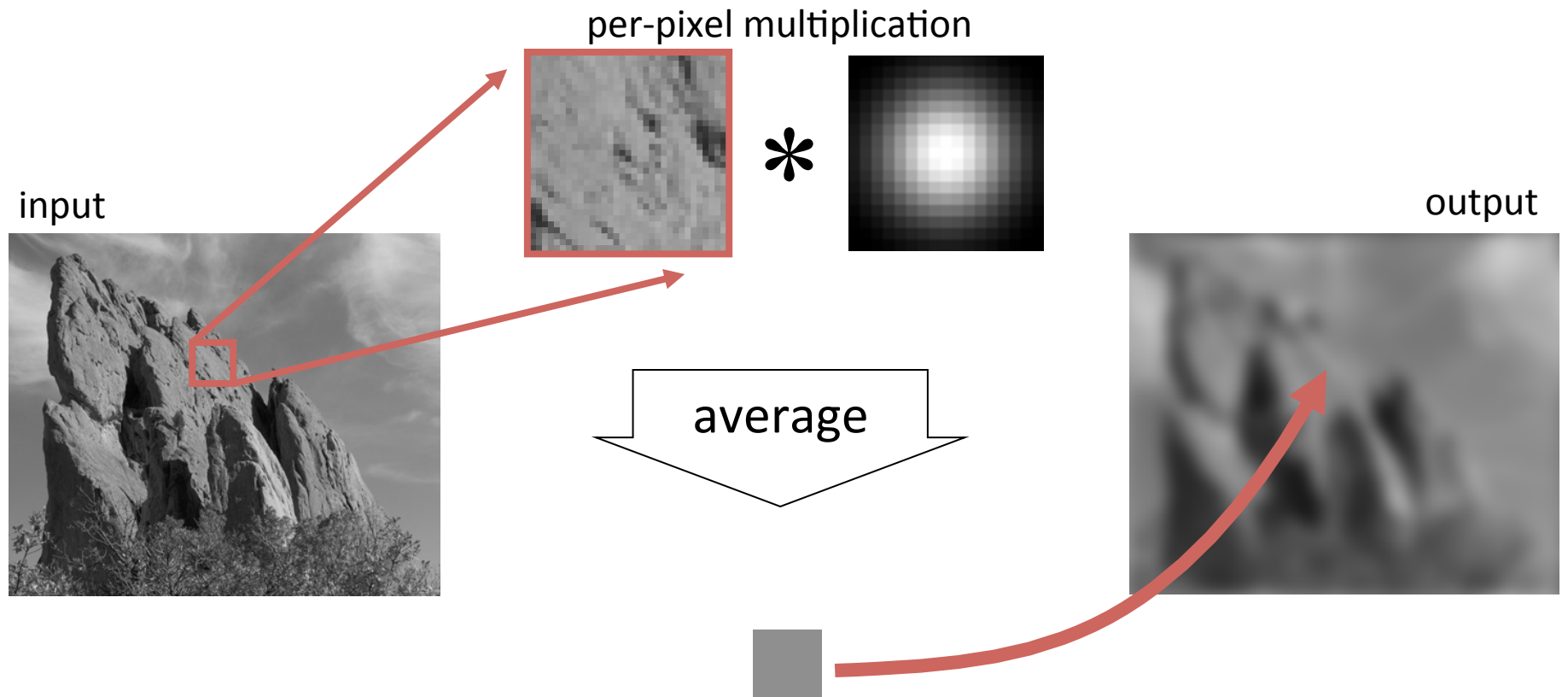


box window



Gaussian window

Gaussian Blur





input

box average

Gaussian blur



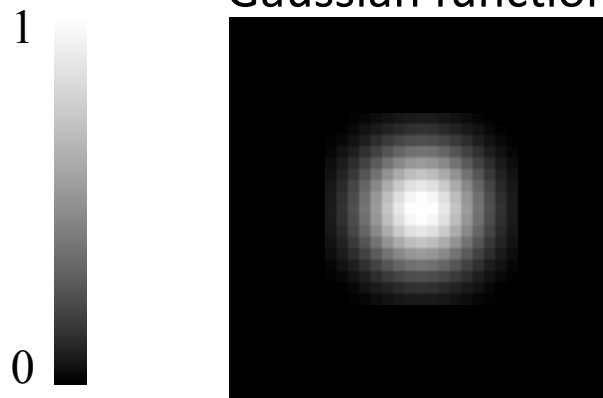
Equation of Gaussian Blur

Same idea: **weighted average of pixels.**

$$GB[I]_p = \sum_{q \in S} G_\sigma(\|\mathbf{p} - \mathbf{q}\|) I_q$$



normalized
Gaussian function



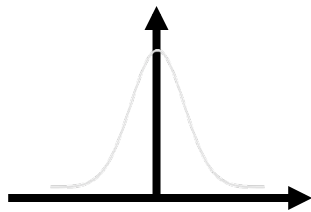
Spatial Parameter



input

$$GB[I]_p = \sum_{q \in S} G_{\sigma}(\|\mathbf{p} - \mathbf{q}\|) I_q$$

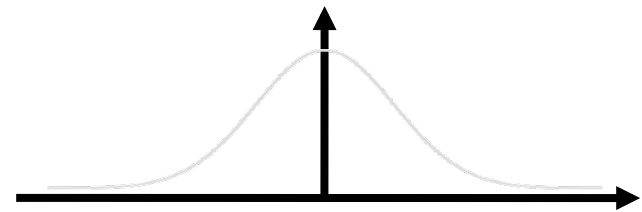
size of the window



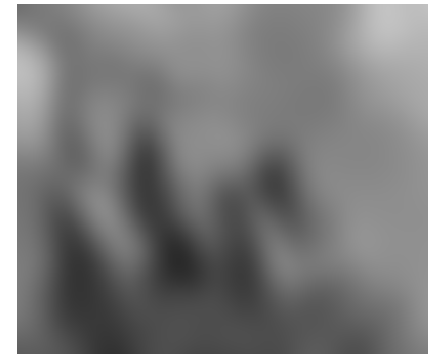
small σ



limited smoothing



large σ



strong smoothing

How to set σ

- Depends on the application.
- Common strategy: proportional to image size
 - e.g. 2% of the image diagonal
 - property: independent of image resolution

Properties of Gaussian Blur

- Weights independent of spatial location
 - linear convolution
 - well-known operation
 - efficient computation (recursive algorithm, FFT...)

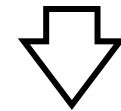
Properties of Gaussian Blur

- Does smooth images
- But smooths too much:
edges are blurred.
 - Only spatial distance matters
 - No edge term

$$GB[I]_p = \sum_{q \in S} G_{\sigma}(\| \mathbf{p} - \mathbf{q} \|) I_q$$

space

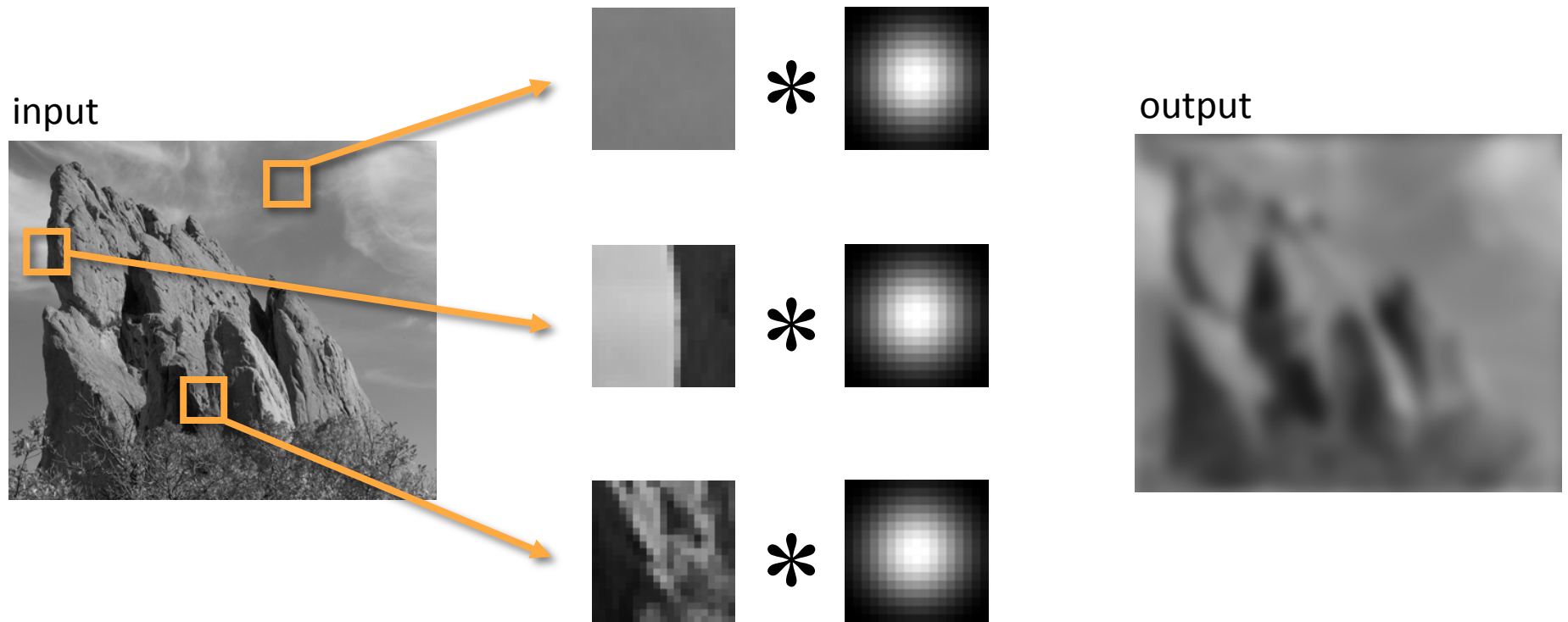
input



output



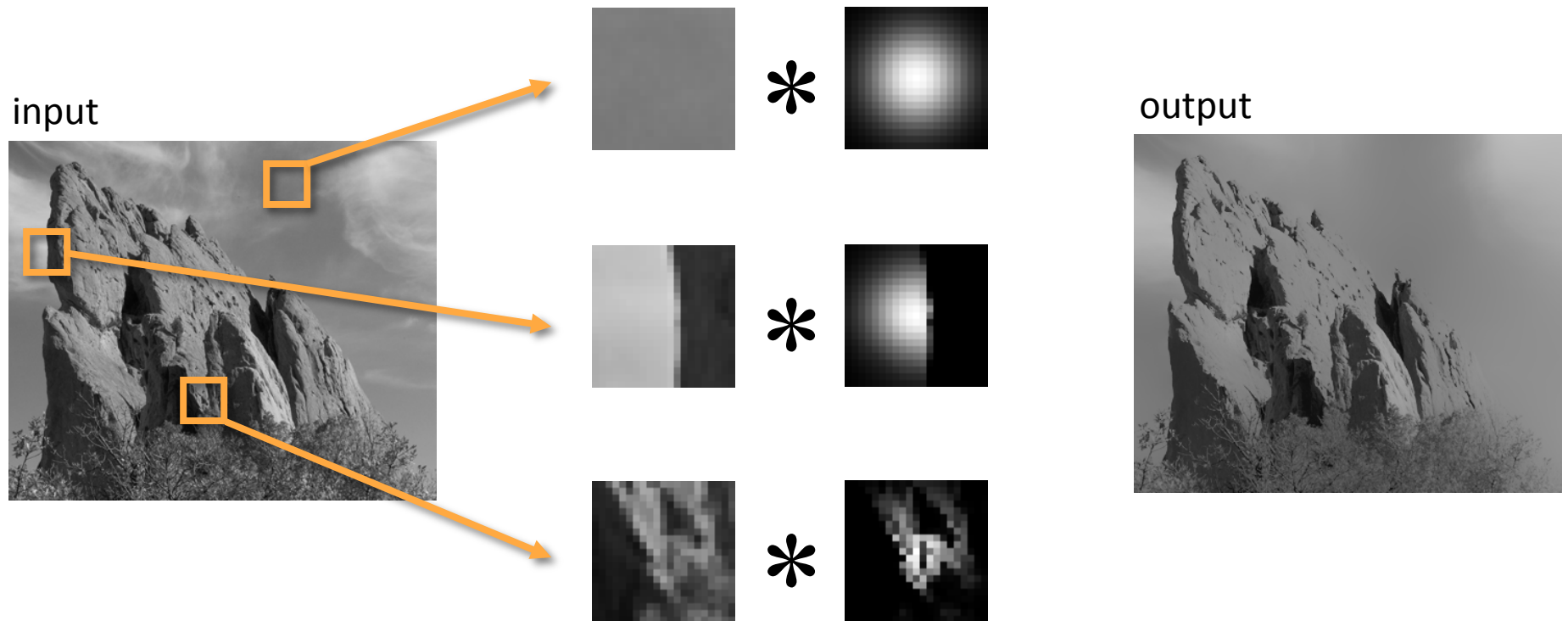
Blur Comes from Averaging across Edges



Same Gaussian kernel everywhere.

Bilateral Filter [Aurich 95, Smith 97, Tomasi 98]

No Averaging across Edges



The kernel shape depends on the image content.

Bilateral Filter Definition: an Additional Edge Term

Same idea: **weighted average of pixels.**

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in \mathcal{S}} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|) G_{\sigma_r}(|I_p - I_q|) I_q$$

new
not new
new

normalization factor

space weight

range weight

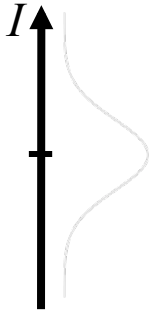
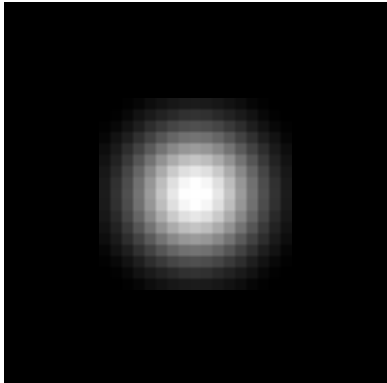
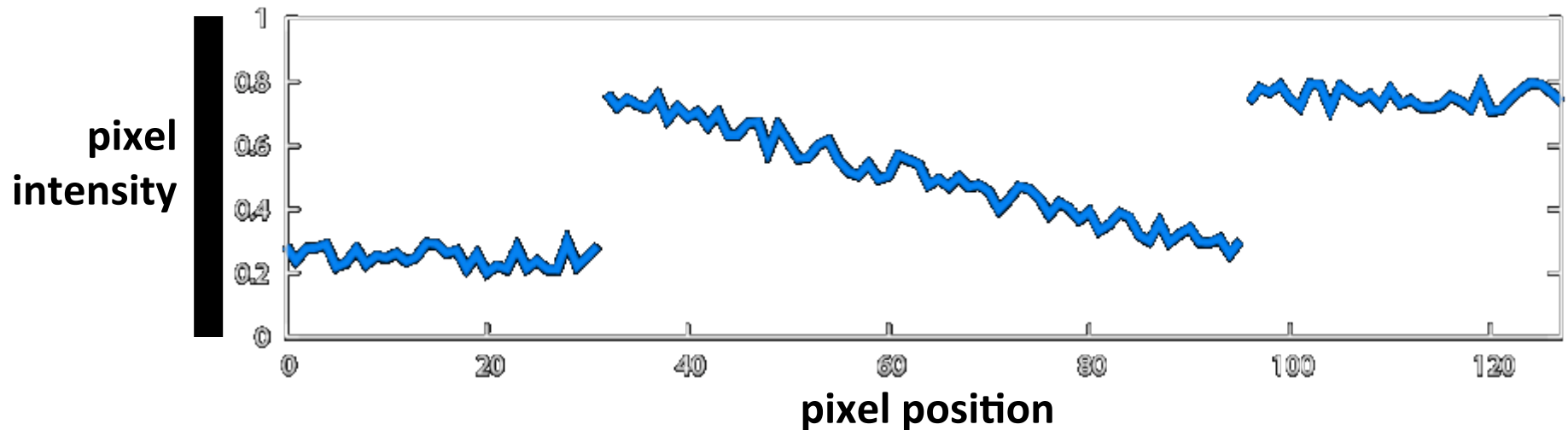


Illustration a 1D Image

- 1D image = line of pixels

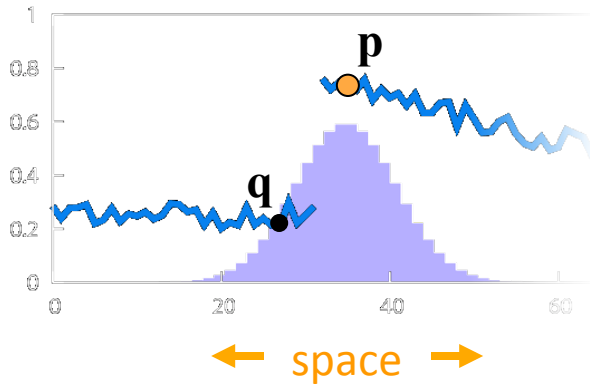


- Better visualized as a plot



Gaussian Blur and Bilateral Filter

Gaussian blur

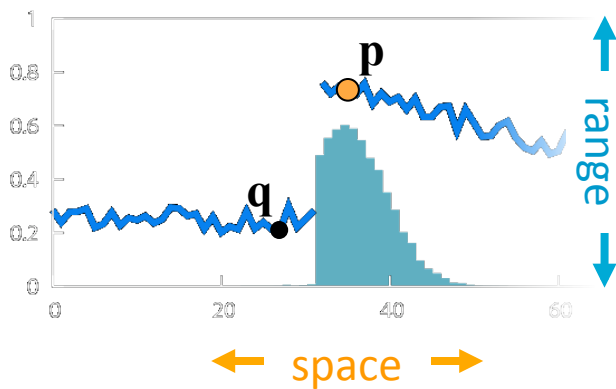


$$GB[I]_p = \sum_{q \in S} G_{\sigma}(\| \mathbf{p} - \mathbf{q} \|) I_q$$

space

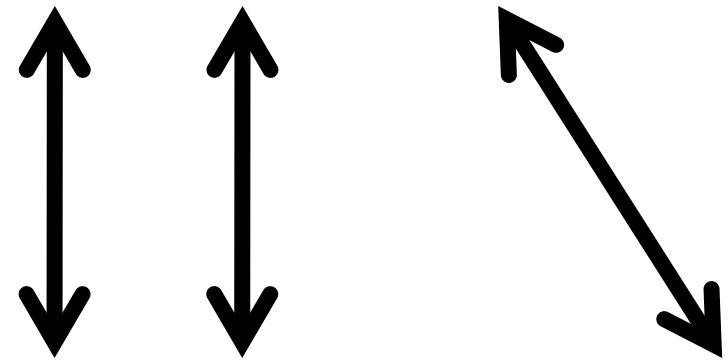
Bilateral filter

[Aurich 95, Smith 97, Tomasi 98]



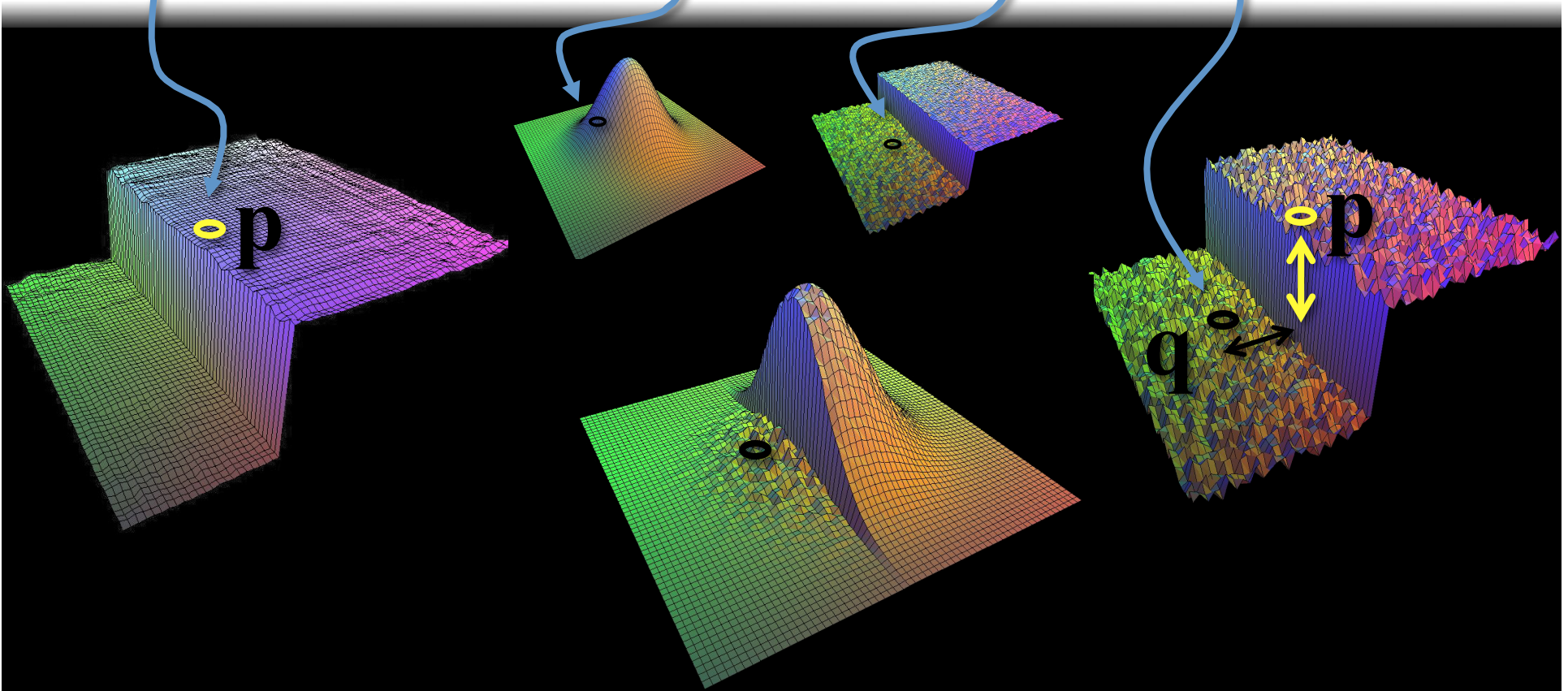
$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\| \mathbf{p} - \mathbf{q} \|) G_{\sigma_r}(|I_p - I_q|) I_q$$

normalization space range




Bilateral Filter on a Height Field

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} \underbrace{G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|)}_{\text{spatial}} \underbrace{G_{\sigma_r}(\|I_p - I_q\|)}_{\text{range}} I_q$$



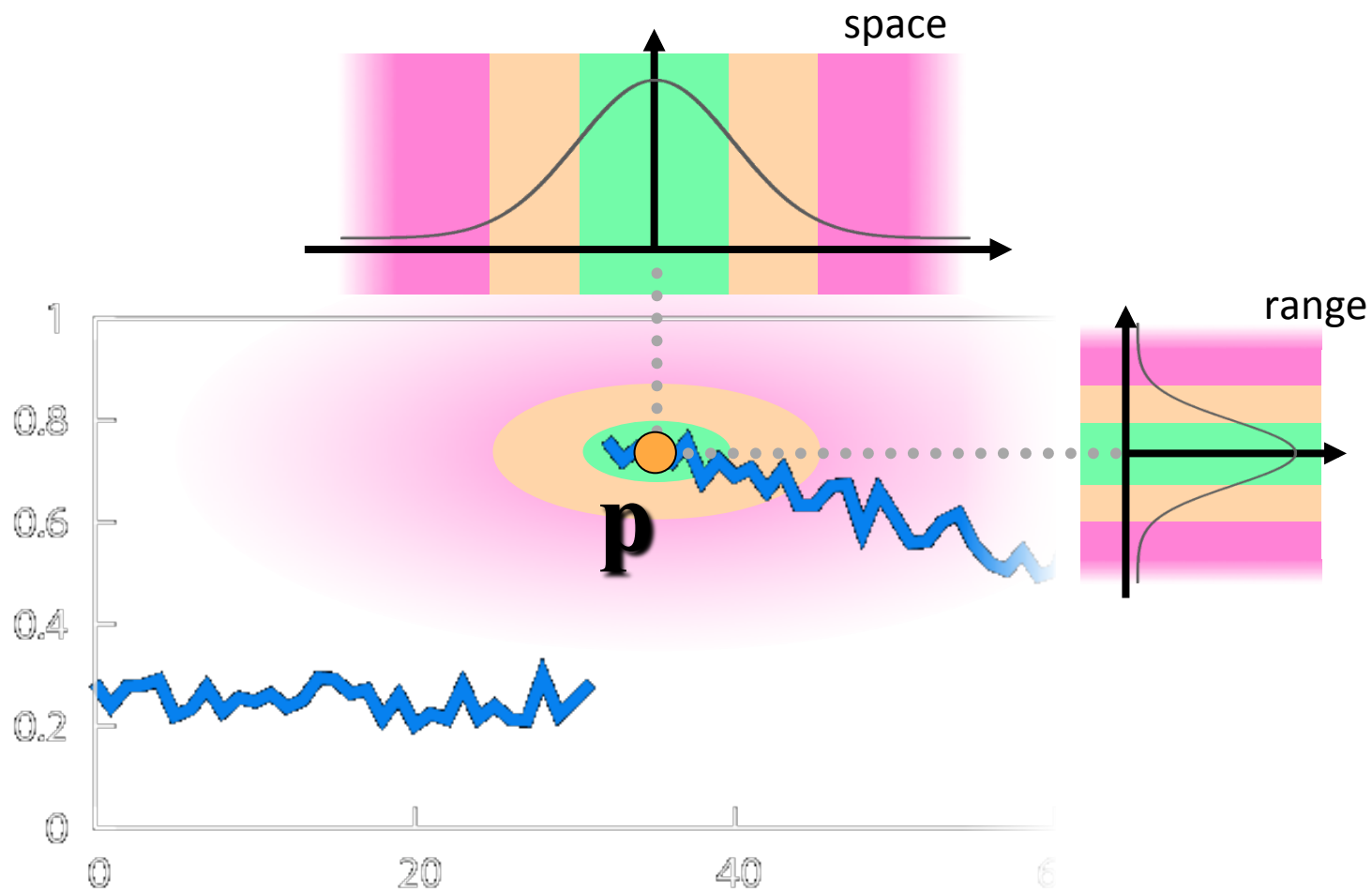
Space and Range Parameters

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|) G_{\sigma_r}(|I_{\mathbf{p}} - I_{\mathbf{q}}|) I_{\mathbf{q}}$$


- space σ_s : spatial extent of the kernel, size of the considered neighborhood.
- range σ_r : “minimum” amplitude of an edge

Influence of Pixels

Only pixels close in space and in range are considered.



Exploring the Parameter Space



input

$$\sigma_s = 2$$



$$\sigma_r = 0.1$$



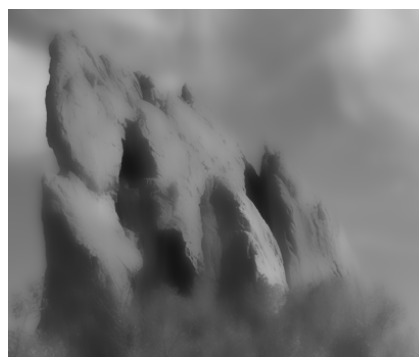
$$\sigma_r = 0.25$$



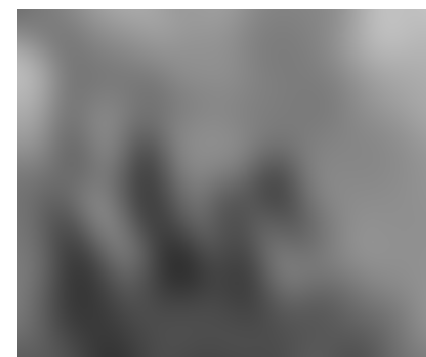
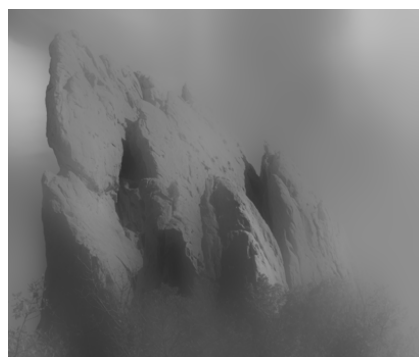
$$\sigma_r = \infty$$

(Gaussian blur)

$$\sigma_s = 6$$



$$\sigma_s = 18$$



Varying the Range Parameter



input

$\sigma_s = 2$

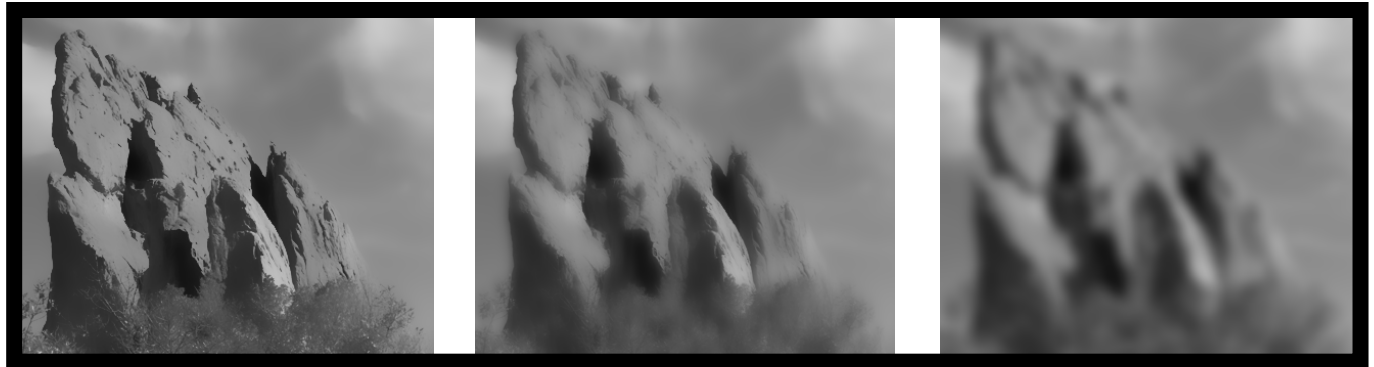
$\sigma_r = 0.1$

$\sigma_r = 0.25$

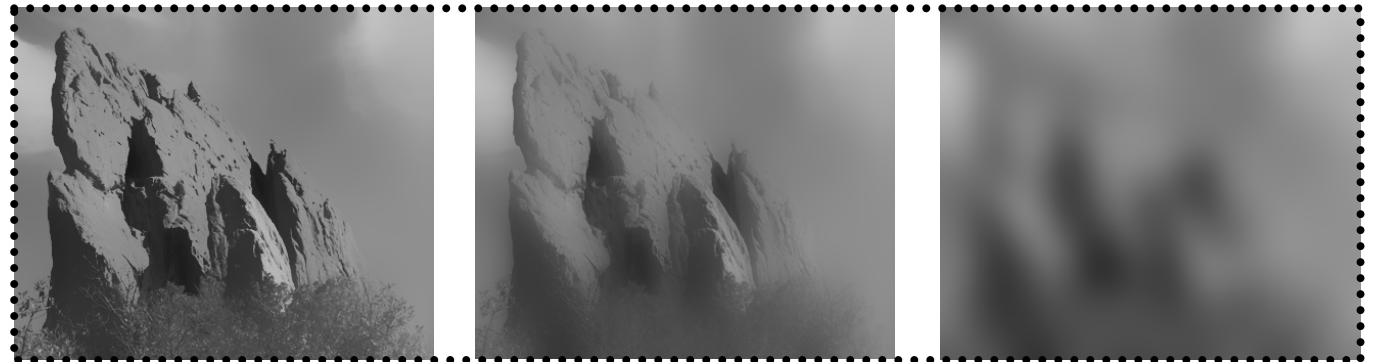
$\sigma_r = \infty$
(Gaussian blur)



$\sigma_s = 6$



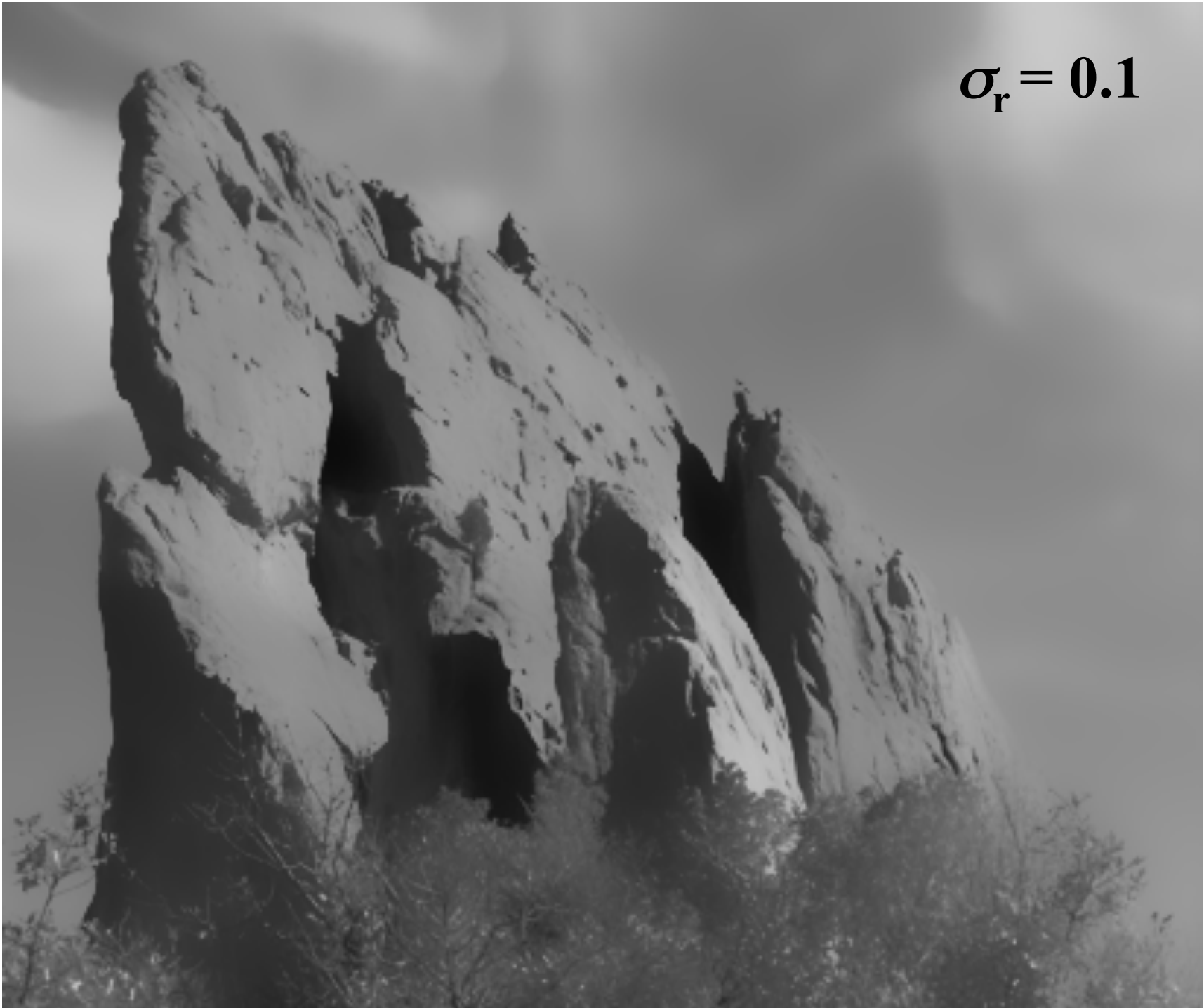
$\sigma_s = 18$



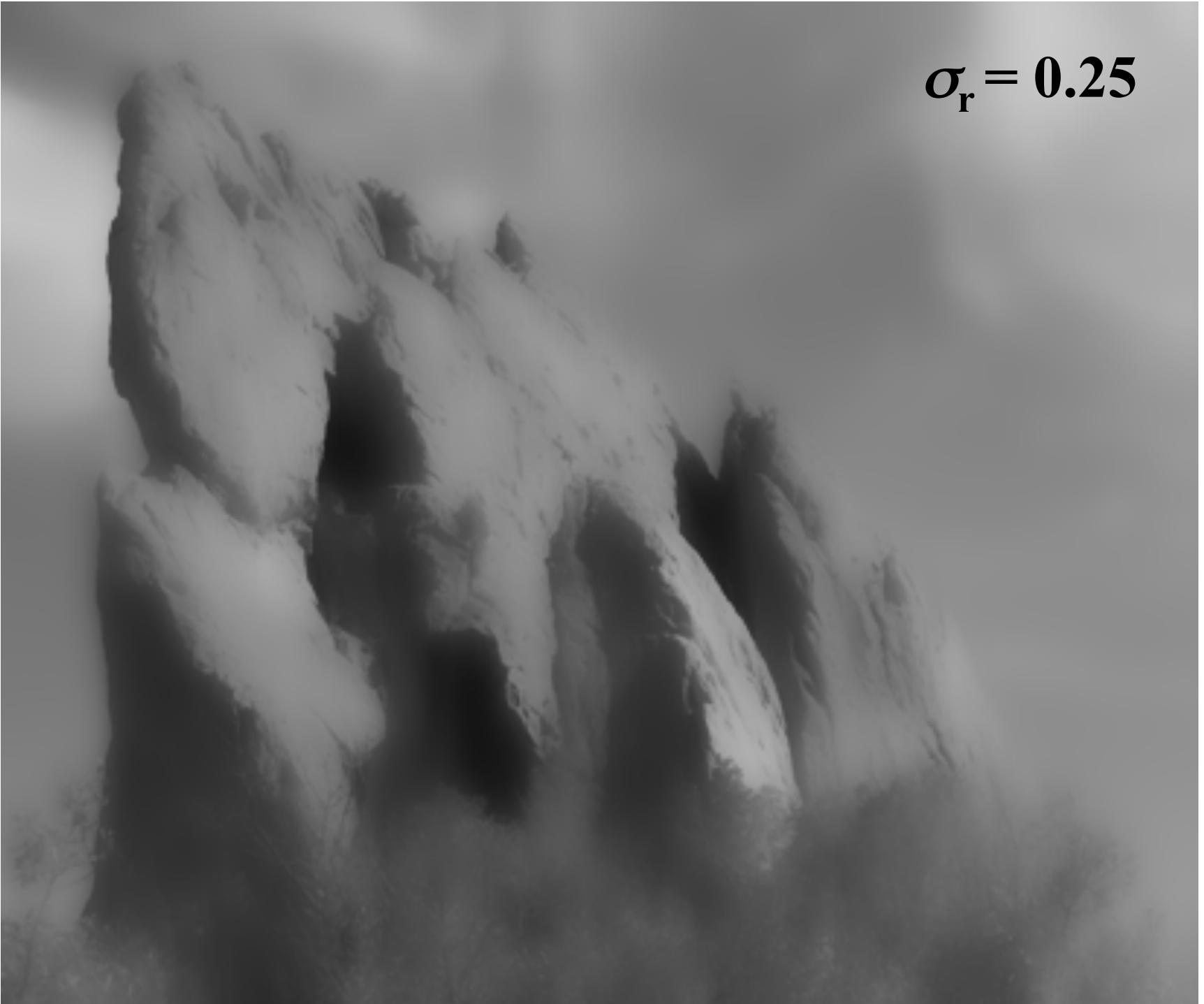


input

$$\sigma_r = 0.1$$



$$\sigma_r = 0.25$$



$\sigma_r = \infty$
(Gaussian blur)



Varying the Space Parameter



input

$\sigma_s = 2$



$\sigma_r = 0.1$

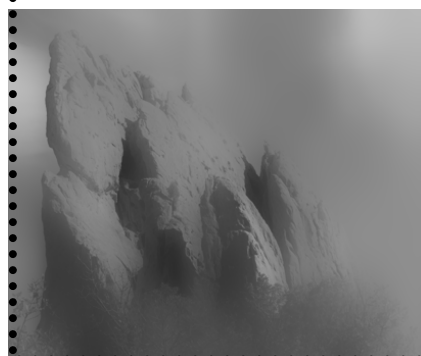
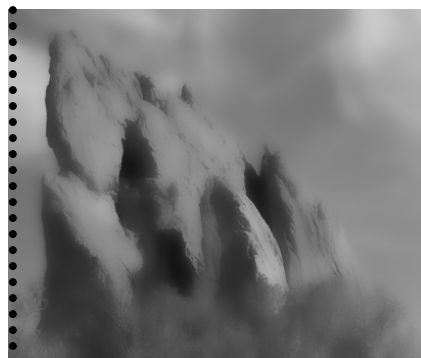
$\sigma_s = 6$



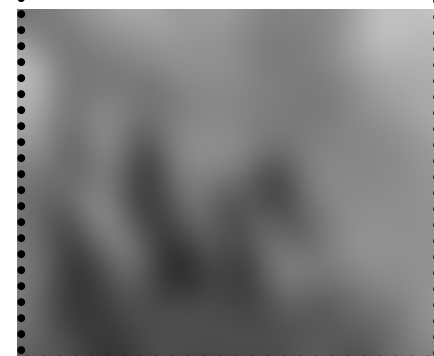
$\sigma_s = 18$



$\sigma_r = 0.25$



$\sigma_r = \infty$
(Gaussian blur)





input

$$\sigma_s = 2$$



$$\sigma_s = 6$$



$$\sigma_s = 18$$



How to Set the Parameters

Depends on the application. For instance:

- space parameter: proportional to image size
 - e.g., 2% of image diagonal
- range parameter: proportional to edge amplitude
 - e.g., mean or median of image gradients
- independent of resolution and exposure

Bilateral Filter Crosses Thin Lines

- Bilateral filter averages across features thinner than $\sim 2\sigma_s$
- Desirable for smoothing: more pixels = more robust
- Different from diffusion that stops at thin lines



Iterating the Bilateral Filter

$$I_{(n+1)} = BF [I_{(n)}]$$

- Generate more piecewise-flat images
- Often not needed in computational photo.



input

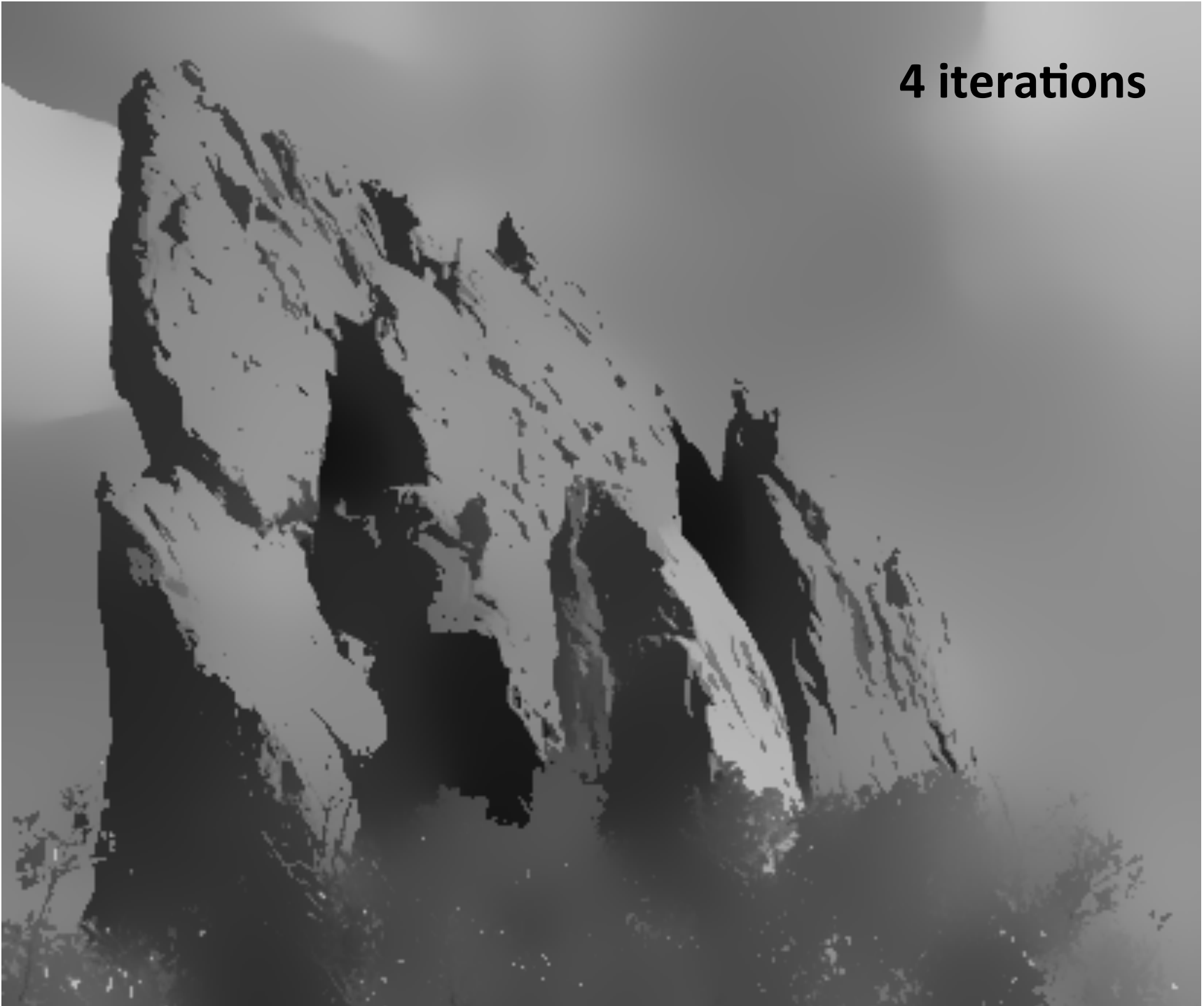
1 iteration



2 iterations



4 iterations



Bilateral Filtering Color Images

For gray-level images

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|) G_{\sigma_r}(\underbrace{\|I_p - I_q\|}_{\text{intensity difference}}) \underbrace{I_q}_{\text{scalar}}$$

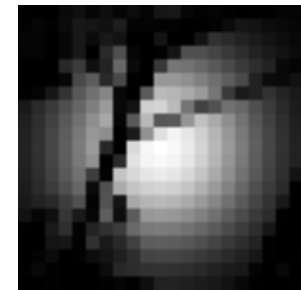
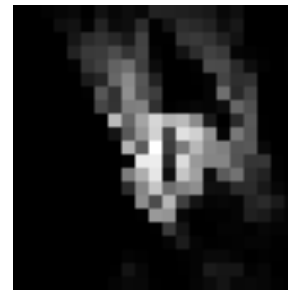
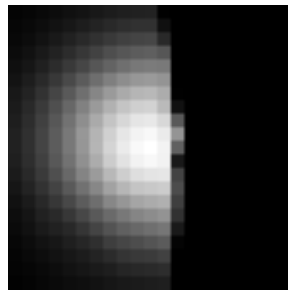
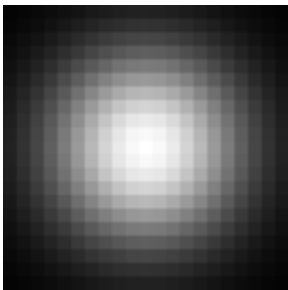
For color images

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|\mathbf{p} - \mathbf{q}\|) G_{\sigma_r}(\underbrace{\|\mathbf{C}_p - \mathbf{C}_q\|}_{\text{color difference}}) \underbrace{\mathbf{C}_q}_{\substack{\text{3D vector} \\ \text{(RGB, Lab)}}$$



Hard to Compute

- Nonlinear $BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(|I_p - I_q|) I_q$
- Complex, spatially varying kernels
 - Cannot be precomputed, no FFT...



- Brute-force implementation is slow $> 10\text{min}$

Additional Reading: *S. Paris and F. Durand, A Fast Approximation of the Bilateral Filter using a Signal Processing Approach, In Proc. ECCV, 2006*

Basic denoising

Noisy input



Bilateral filter 7x7 window



Basic denoising

Bilateral filter



Median 3x3



Basic denoising

Bilateral filter



Median 5x5



Basic denoising

Bilateral filter



Bilateral filter – lower sigma



Basic denoising

Bilateral filter

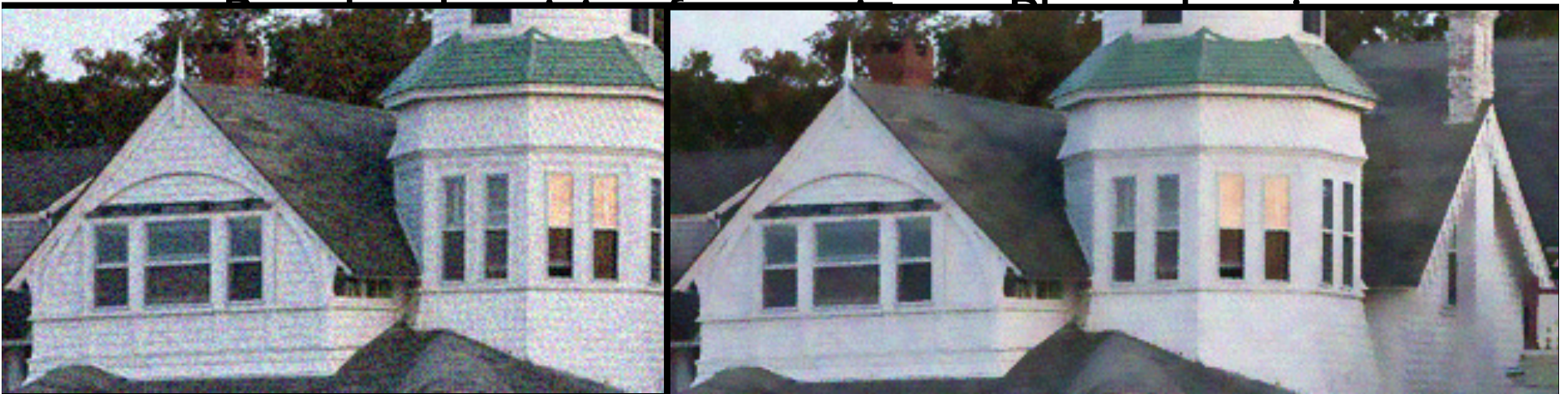


Bilateral filter – higher sigma

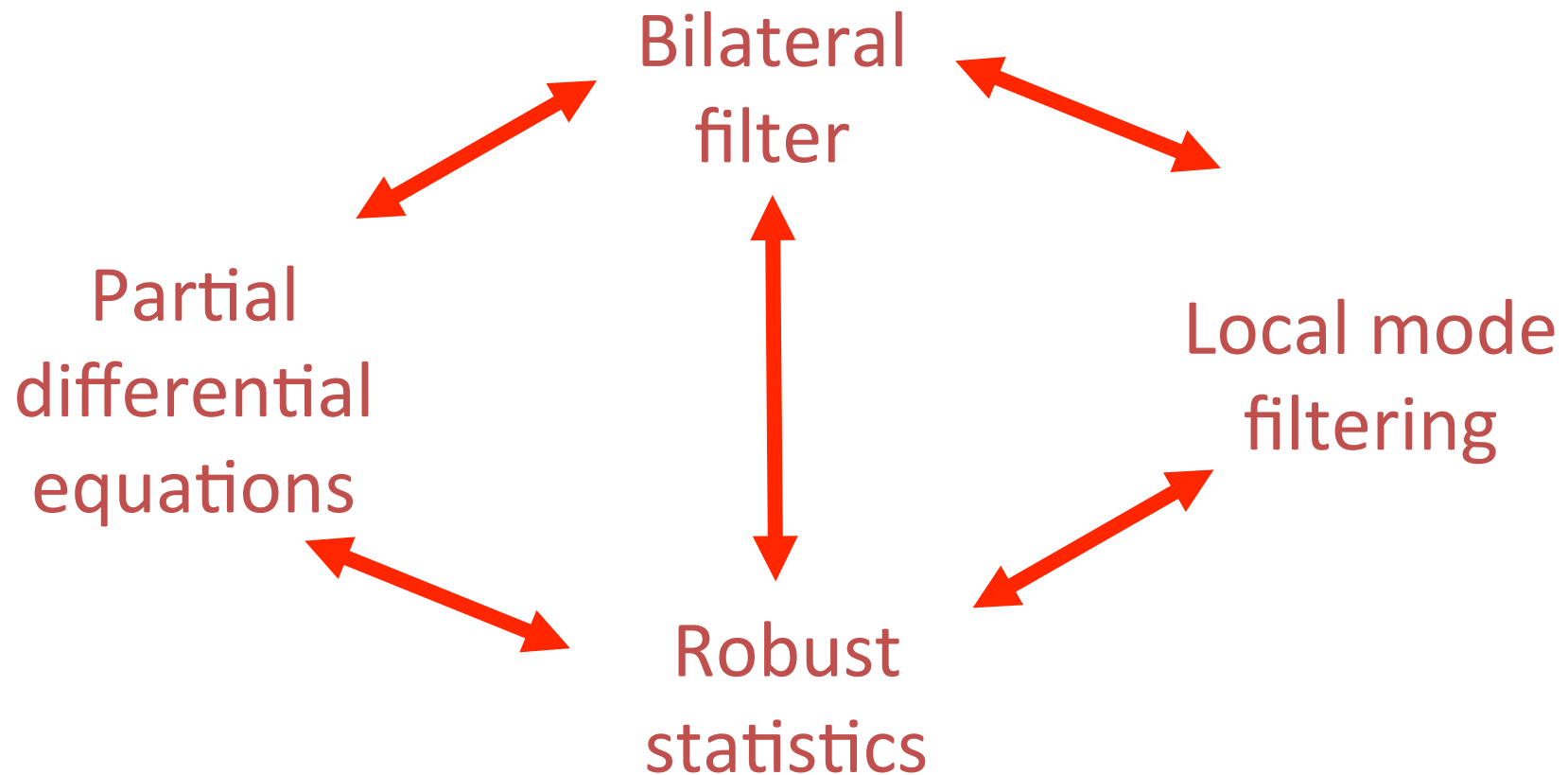


Denoising

- Small spatial sigma (e.g. 7x7 window)
- Adapt range sigma to noise level
- Maybe not best denoising method, but best simplicity/quality tradeoff
 - No need for acceleration (small kernel)



Goal: Understand how does bilateral filter relates with other methods



Additional Reading: *Generalised Nonlocal Image Smoothing*, L. Pizarro, P. Mrazek, S. Didas, S. Grewenig and J. Weickert, IJCV, 2010

New Idea:
NL-Means Filter (Buades 2005)

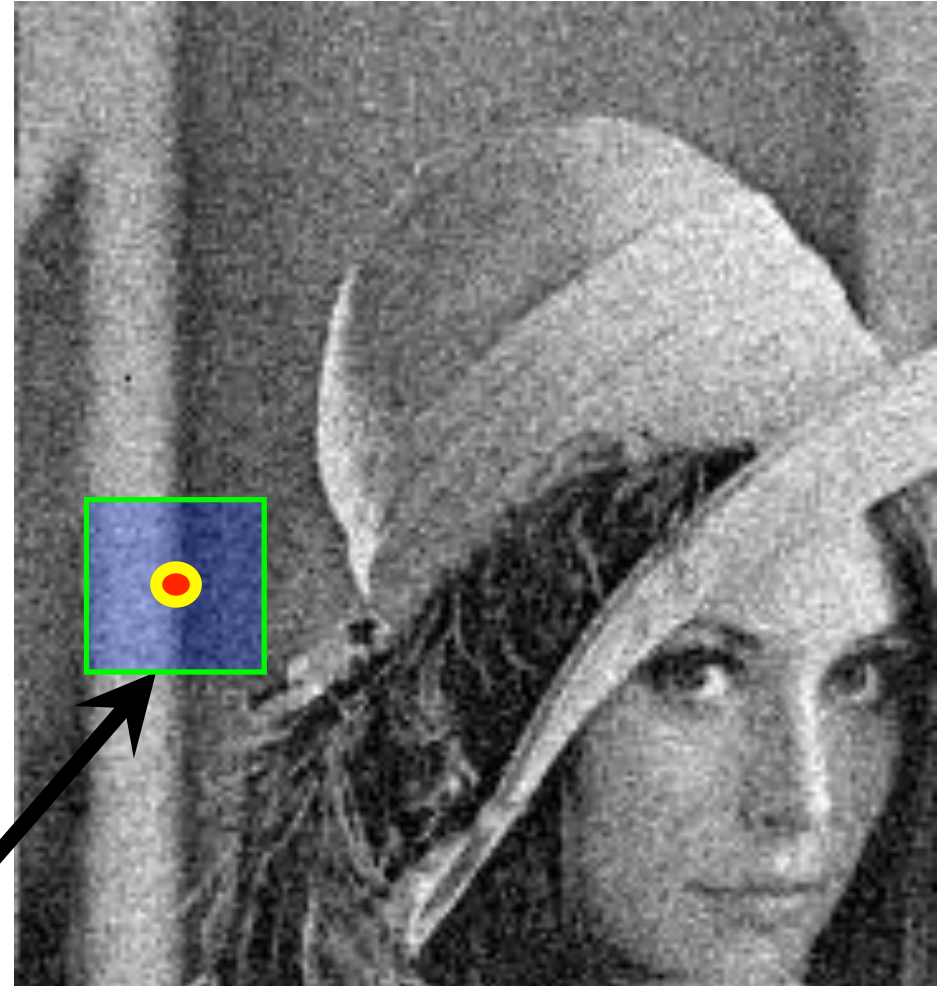
- Same goals: ‘Smooth within Similar Regions’
- **KEY INSIGHT**: Generalize, extend ‘Similarity’
 - **Bilateral**:
Averages neighbors with **similar intensities**;
 - **NL-Means**:
Averages neighbors with **similar neighborhoods!**

NL-Means Method: Buades (2005)

- For each and every pixel **p**:



NL-Means Method: Buades (2005)

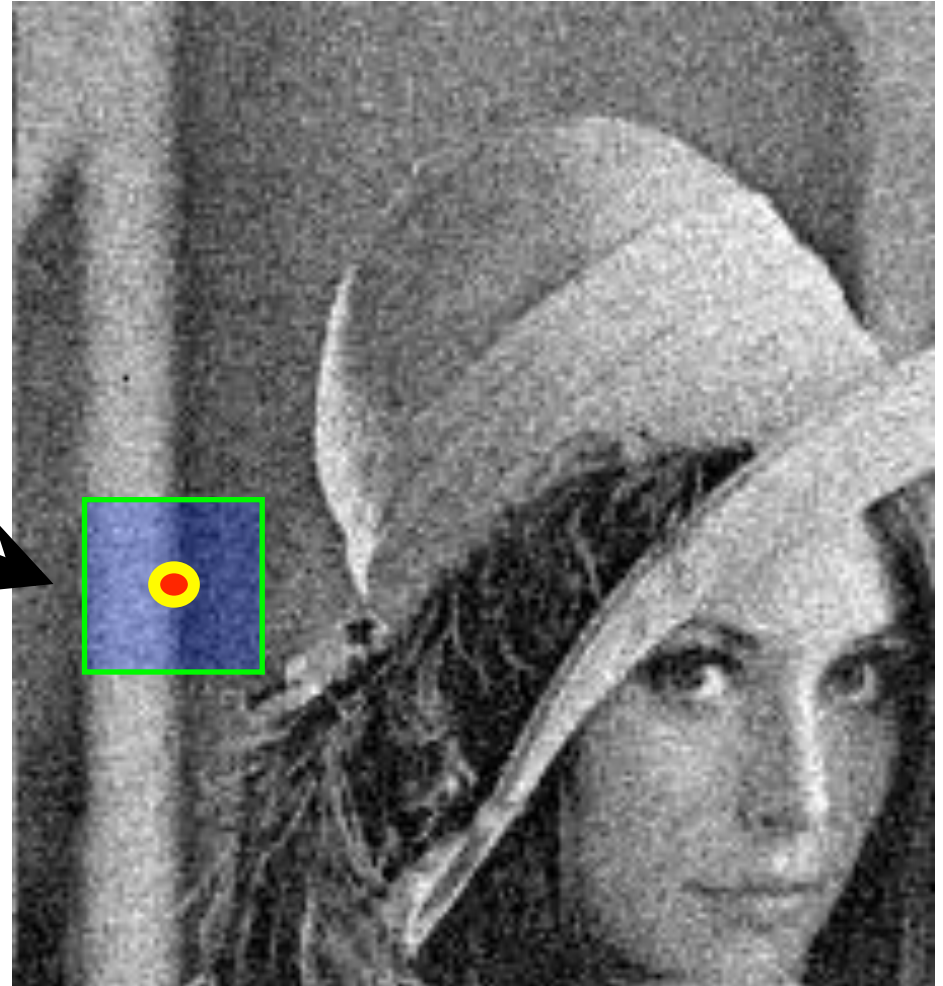
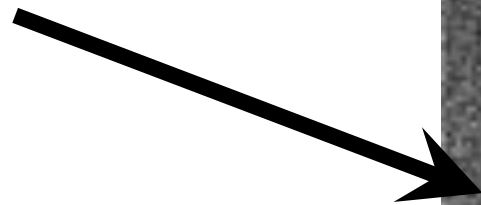


- For each and every pixel **p**:

- Define a small, simple fixed size neighborhood;

NL-Means Method: Buades (2005)

$$\mathbf{V}_p = \begin{bmatrix} 0.74 \\ 0.32 \\ 0.41 \\ 0.55 \\ \dots \\ \dots \\ \dots \end{bmatrix}$$



- For each and every pixel \mathbf{p} :

- Define a small, simple fixed size neighborhood;
- Define vector \mathbf{V}_p : a list of neighboring pixel values.

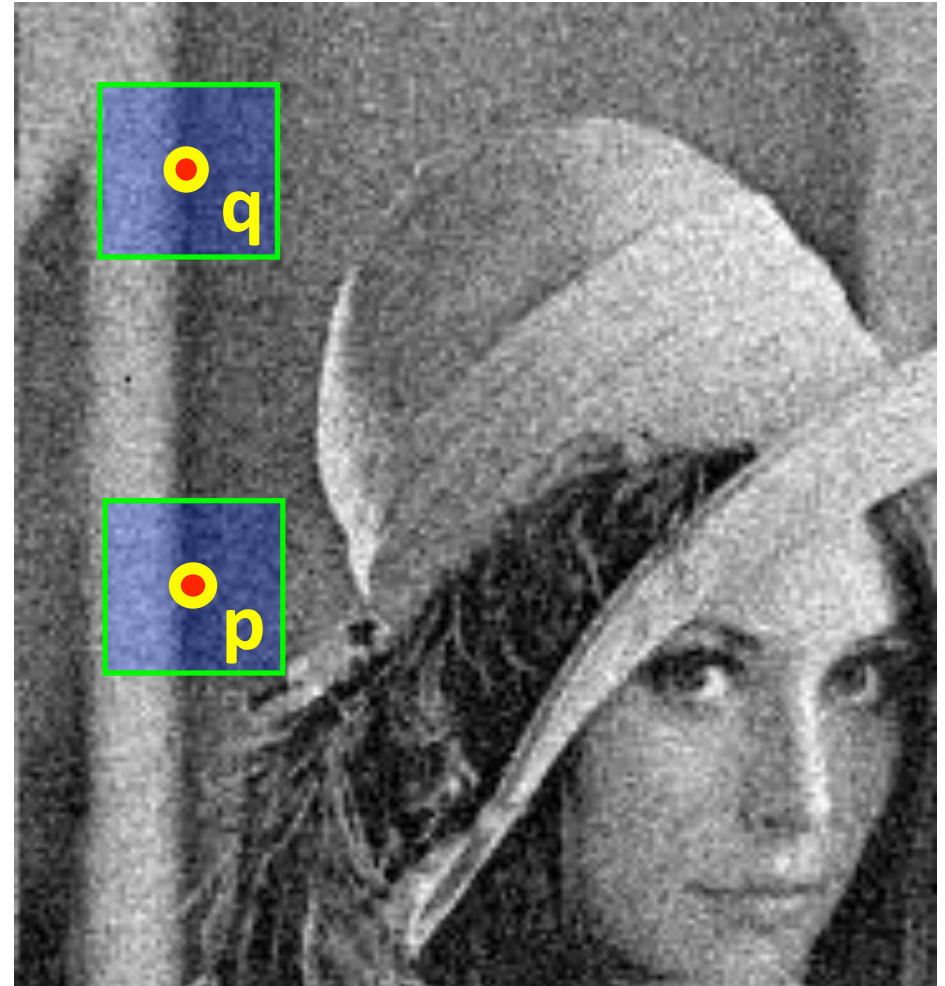
NL-Means Method: Buades (2005)

'Similar' pixels **p**, **q**

→ **SMALL**

vector distance;

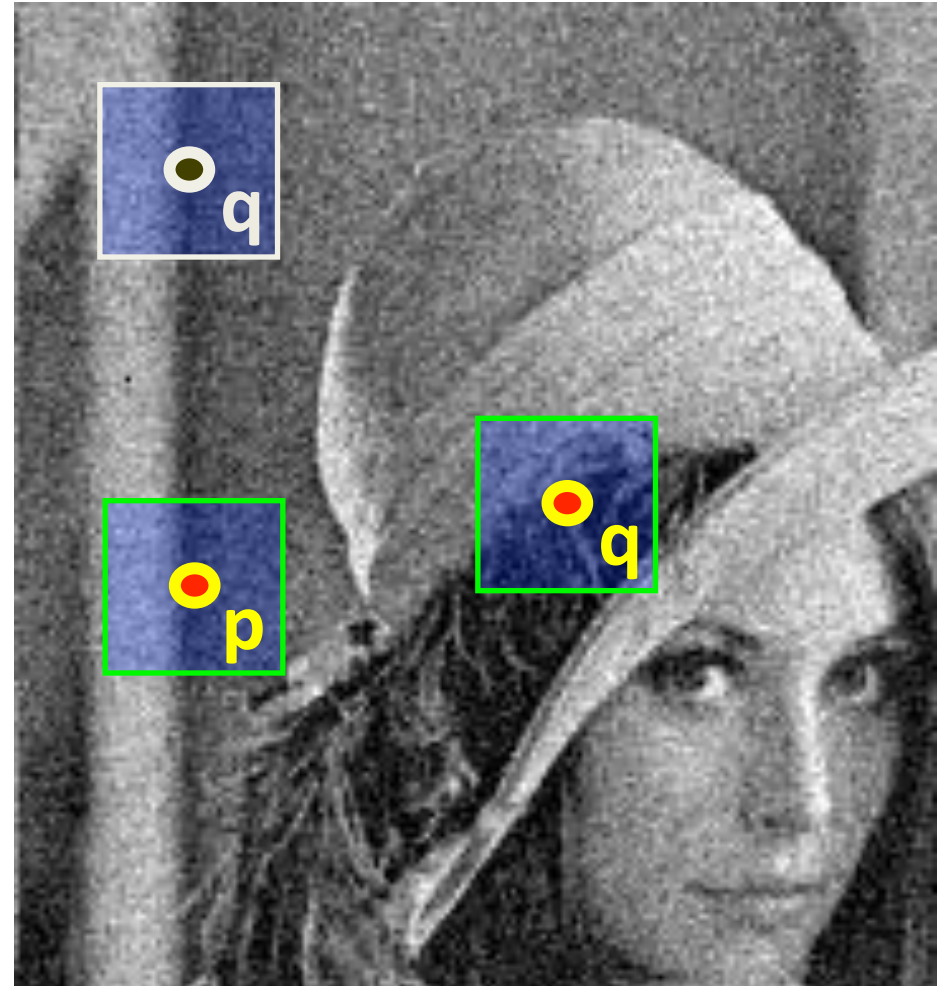
$$\| \mathbf{V}_p - \mathbf{V}_q \|^2$$



NL-Means Method: Buades (2005)

'Dissimilar' pixels **p, q**
→ **LARGE**
vector distance;

$$\| \mathbf{V}_p - \mathbf{V}_q \|^2$$

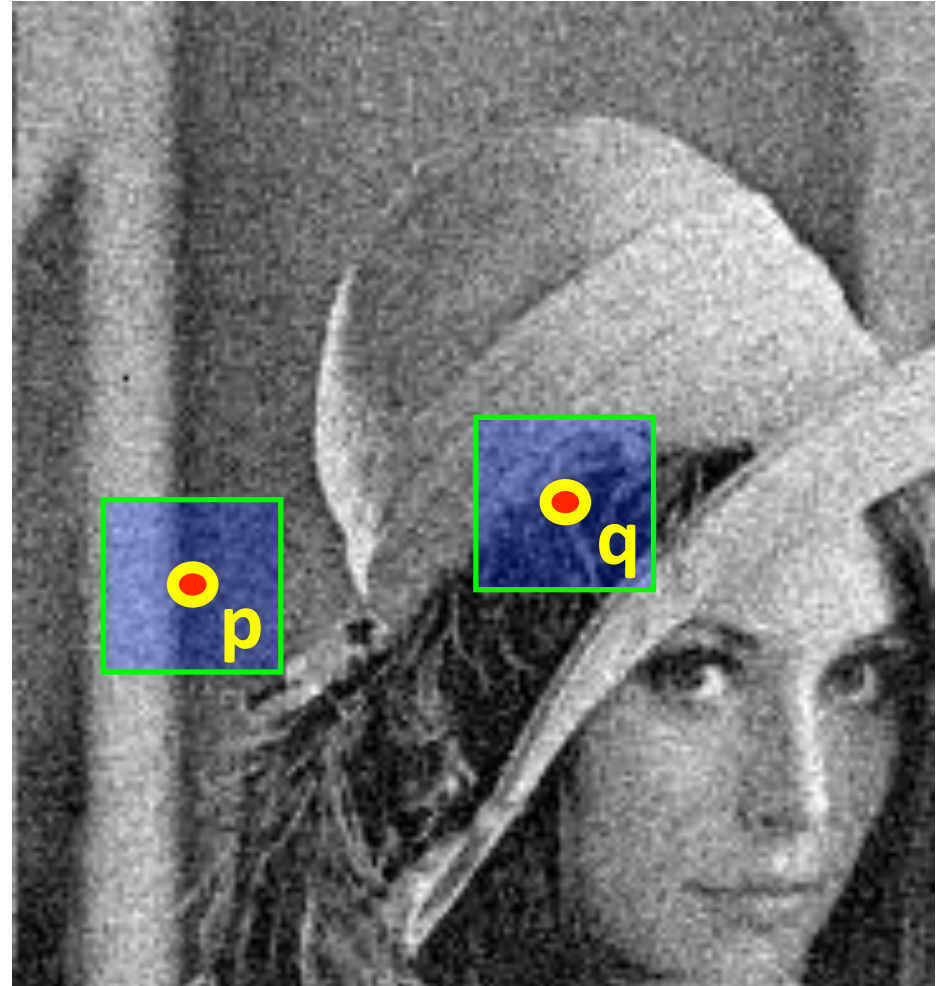


NL-Means Method: Buades (2005)

'Dissimilar' pixels **p, q**
→ **LARGE**
vector distance;

$$\|V_p - V_q\|^2$$

Filter with this!



NL-Means Method: Buades (2005)

p, q neighbors define
a vector distance;

$$\| \vec{V}_p - \vec{V}_q \|^2$$

Filter with this:
No spatial term!



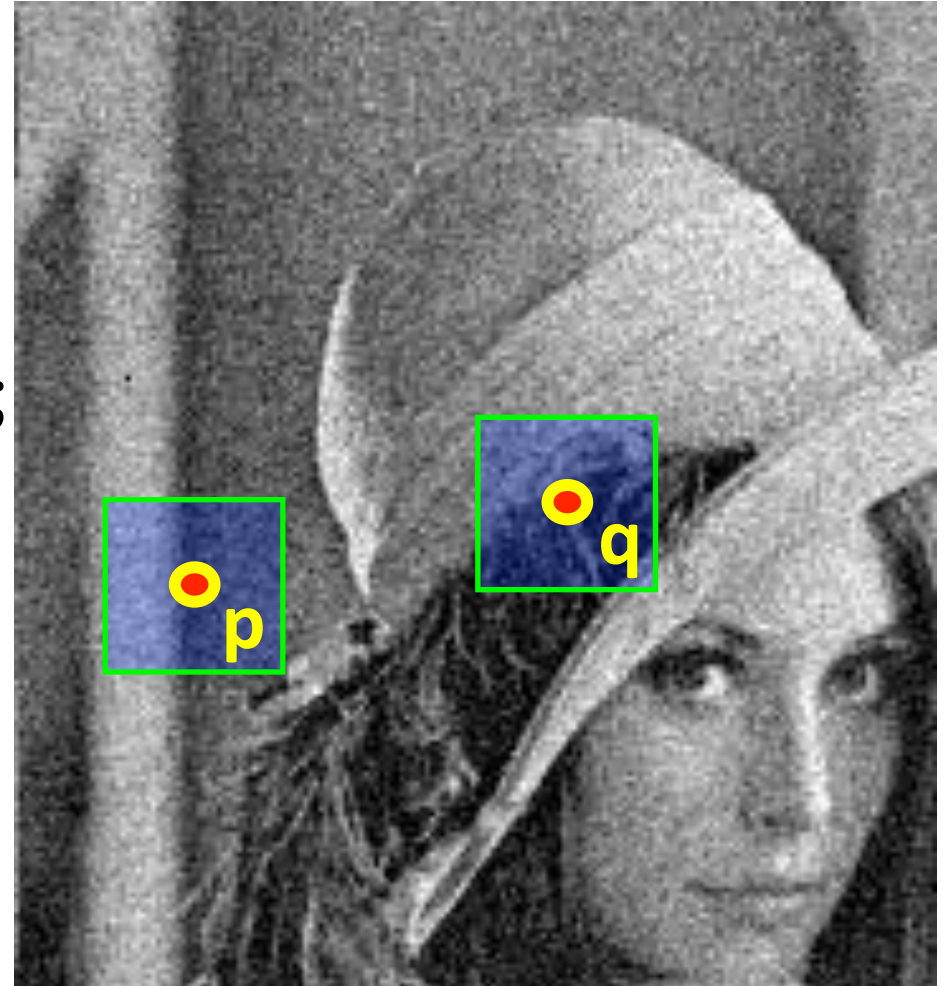
$$NLMF[I]_p = \frac{1}{W_p} \sum_{q \in S} \cancel{G_{\sigma_s}(\| \mathbf{p} - \mathbf{q} \|)} G_{\sigma_r}(\| \vec{V}_p - \vec{V}_q \|^2) I_q$$

NL-Means Method: Buades (2005)

pixels **p, q** neighbors
Set a vector distance;

$$\| \mathbf{V}_p - \mathbf{V}_q \|^2$$

**Vector Distance to p sets
weight for each pixel q**



$$NLMF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_r} \left(\| \vec{V}_p - \vec{V}_q \|^2 \right) I_q$$

NL-Means Filter (Buades 2005)

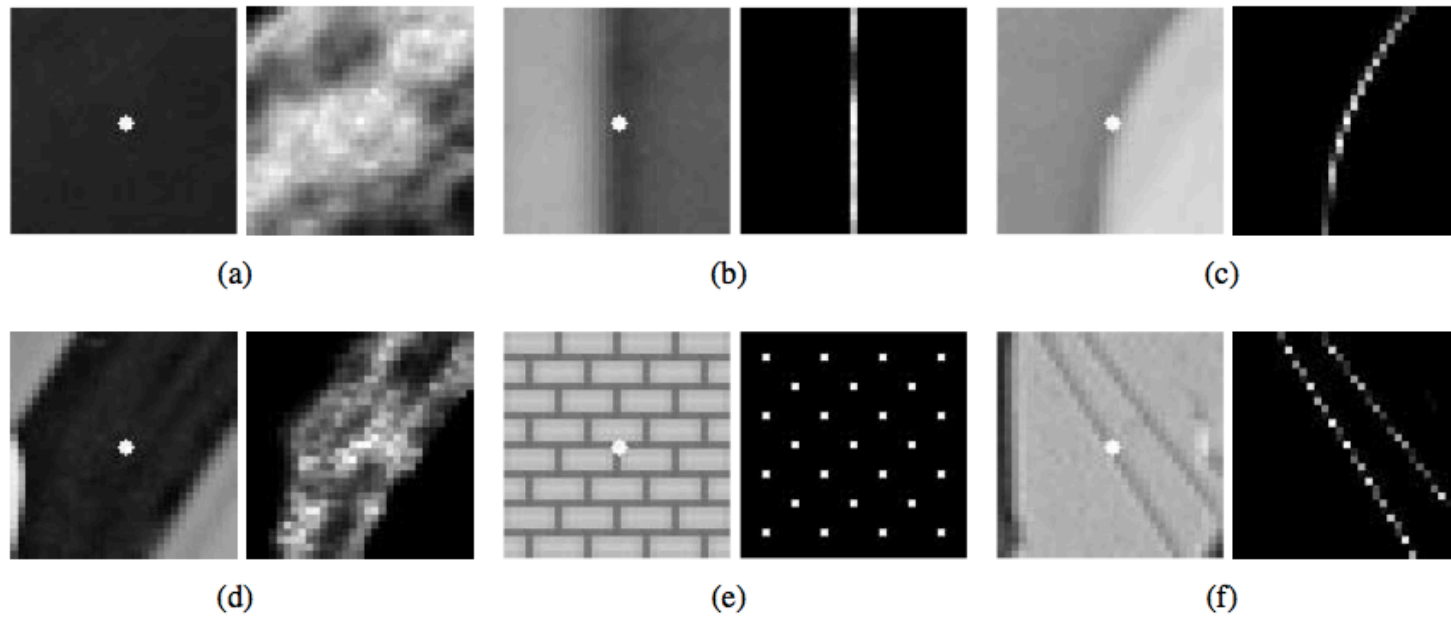


Figure 2. Display of the NL-means weight distribution used to estimate the central pixel of every image. The weights go from 1(white) to zero(black).

NL-Means Filter (Buades 2005)



FIG. 9. *NL-means denoising experiment with a natural image. Left: Noisy image with standard deviation 20. Right: Restored image.*

NL-Means Filter (Buades 2005)

- Noisy source image:



NL-Means Filter (Buades 2005)

- Gaussian Filter

Low noise,
Low detail



NL-Means Filter (Buades 2005)

- Anisotropic Diffusion

(Note
‘stairsteps’ :
~ piecewise
constant)



NL-Means Filter (Buades 2005)

- Bilateral Filter

(better, but similar
'stairsteps' :



NL-Means Filter (Buades 2005)

- NL-Means:

Sharp,
Low noise,
Few artifacts.



NL-Means Filter (Buades 2005)

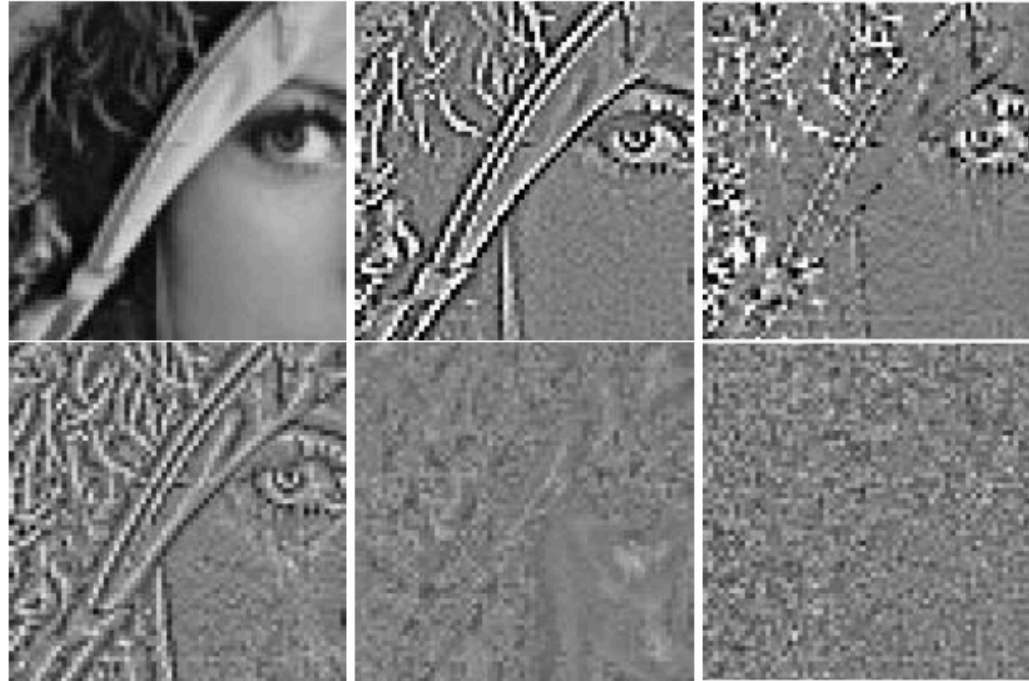


Figure 4. Method noise experience on a natural image. Displaying of the image difference $u - D_h(u)$. From left to right and from top to bottom: original image, Gauss filtering, anisotropic filtering, Total variation minimization, Neighborhood filtering and NL-means algorithm. The visual experiments corroborate the formulas of section 2.

NL-Means Filter (Buades 2005)

original

noisy, standard deviation 15

denoised



http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

NL-Means Filter (Buades 2005)



original

http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

NL-Means Filter (Buades 2005)



noisy

http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

NL-Means Filter (Buades 2005)



denoised

http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

NL-Means Filter (Buades 2005)



original

http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

NL-Means Filter (Buades 2005)



noisy

http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

NL-Means Filter (Buades 2005)



denoised

http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/