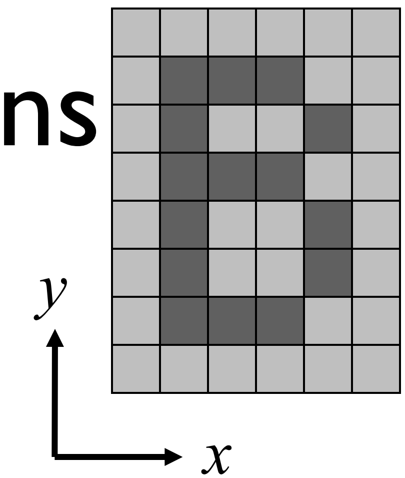
Bilateral Filtering, and Non-local Means Denoising

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/)

Notation and Definitions

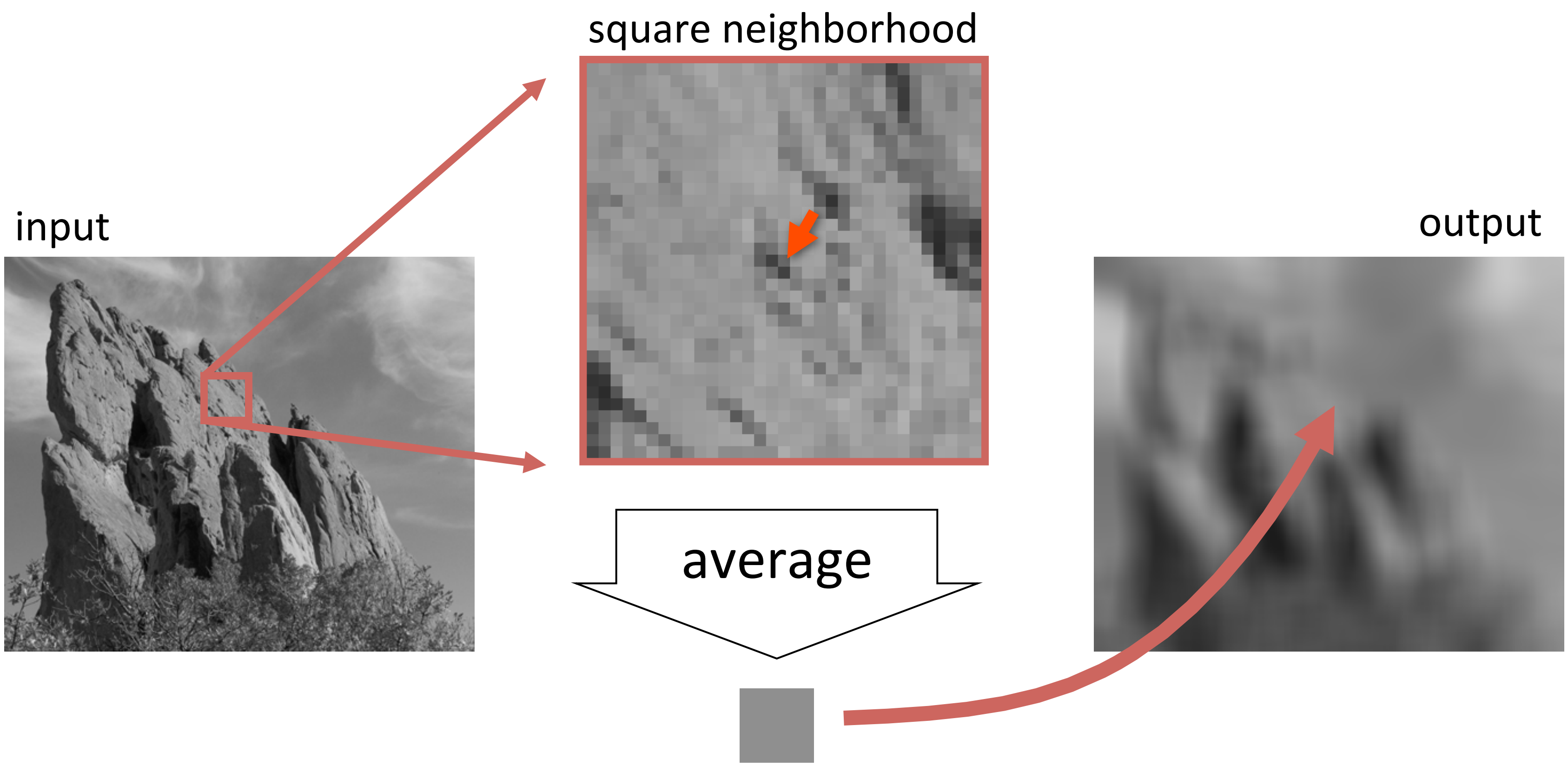


- Image = 2D array of pixels
- Pixel = intensity (scalar) or color (3D vector)
- $I_{\mathbf{p}}$ = value of image I at position: $\mathbf{p} = (p_x , p_y)$
- $F [I]$ = output of filter F applied to image I

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



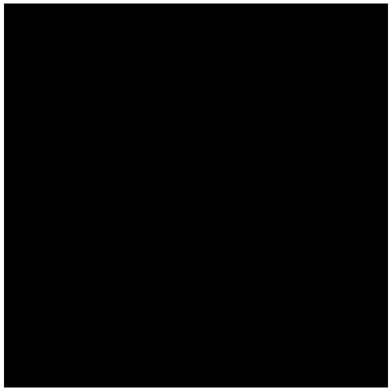
Equation of Box Average

$$BA[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} B_{\sigma}(\mathbf{p}-\mathbf{q})\, I_{\mathbf{q}}$$

result at pixel $\mathbf{p}$

sum over all pixels $\mathbf{q}$

normalized box function



intensity at pixel $\mathbf{q}$

Square Box Generates Defects

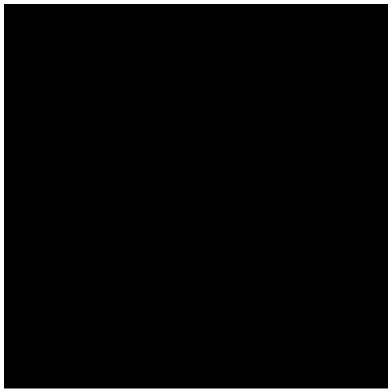
- Axis-aligned streaks
- Blocky results

output

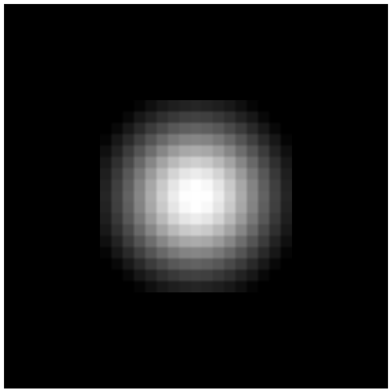
input

Strategy to Solve these Problems

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

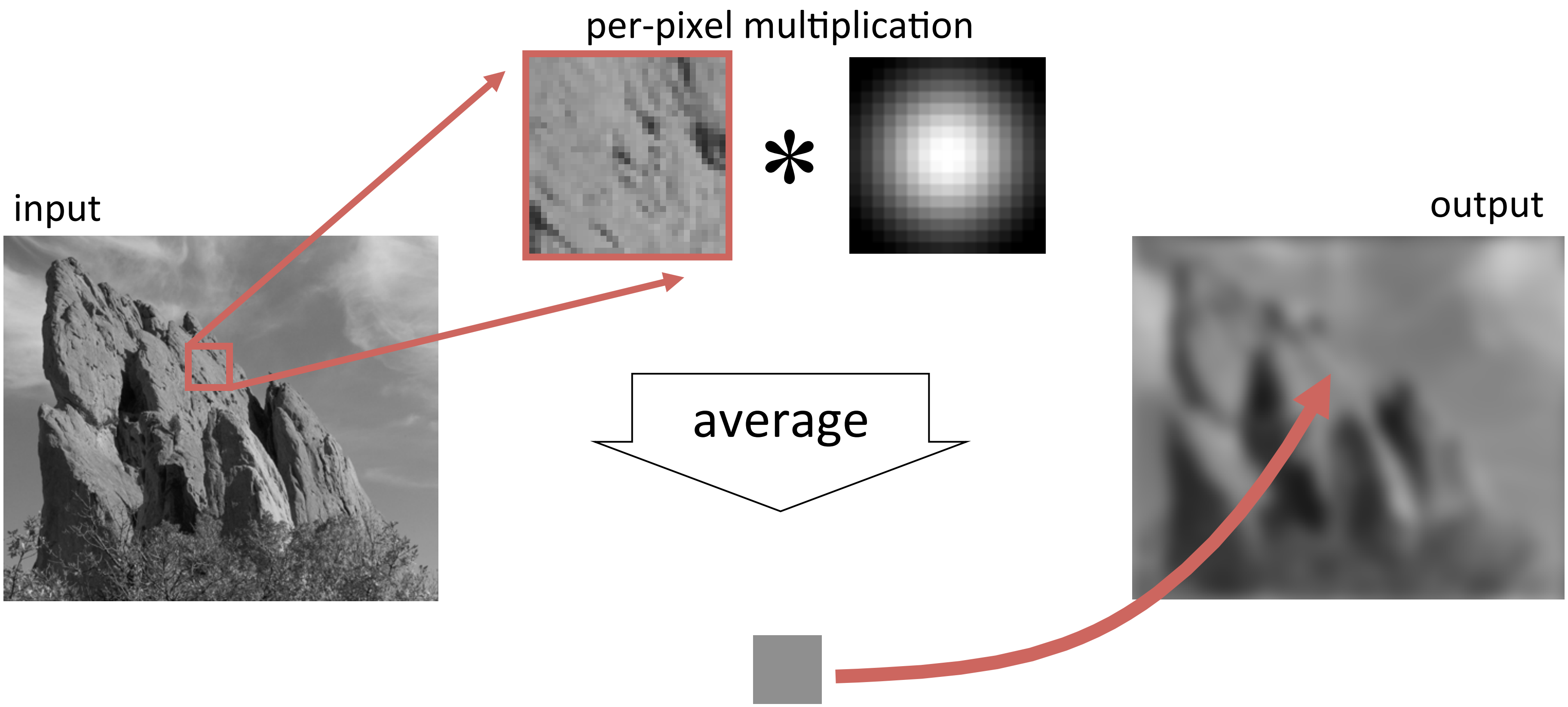


box window



Gaussian window

Gaussian Blur




input


box average

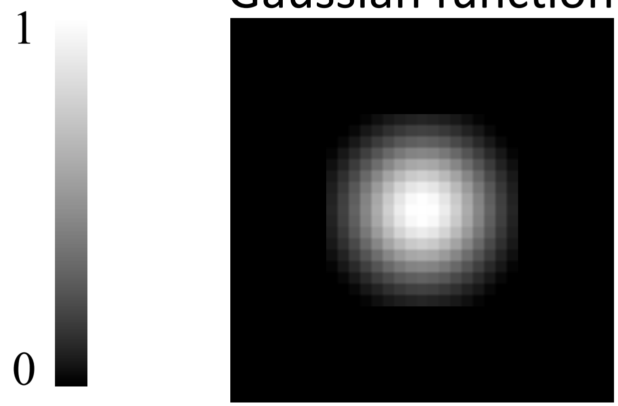

Gaussian blur

Equation of Gaussian Blur

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

$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} G_{\sigma}\left(\|\mathbf{p}-\mathbf{q}\|\right) I_{\mathbf{q}}$$

normalized
Gaussian function

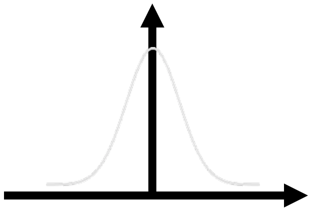


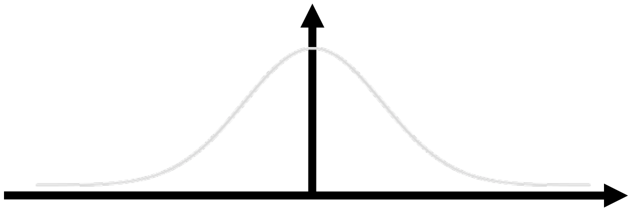
Spatial Parameter


input

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

size of the window


small σ


large σ


limited smoothing


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

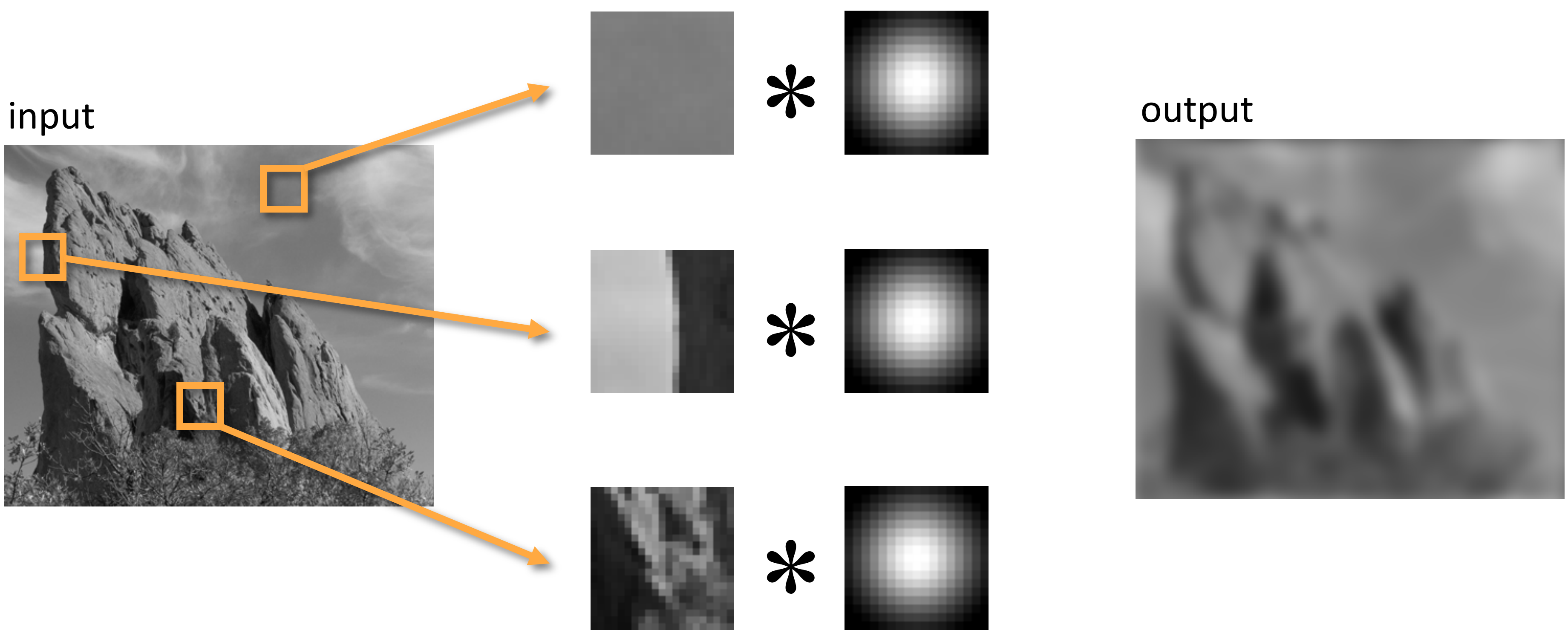
input

output

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

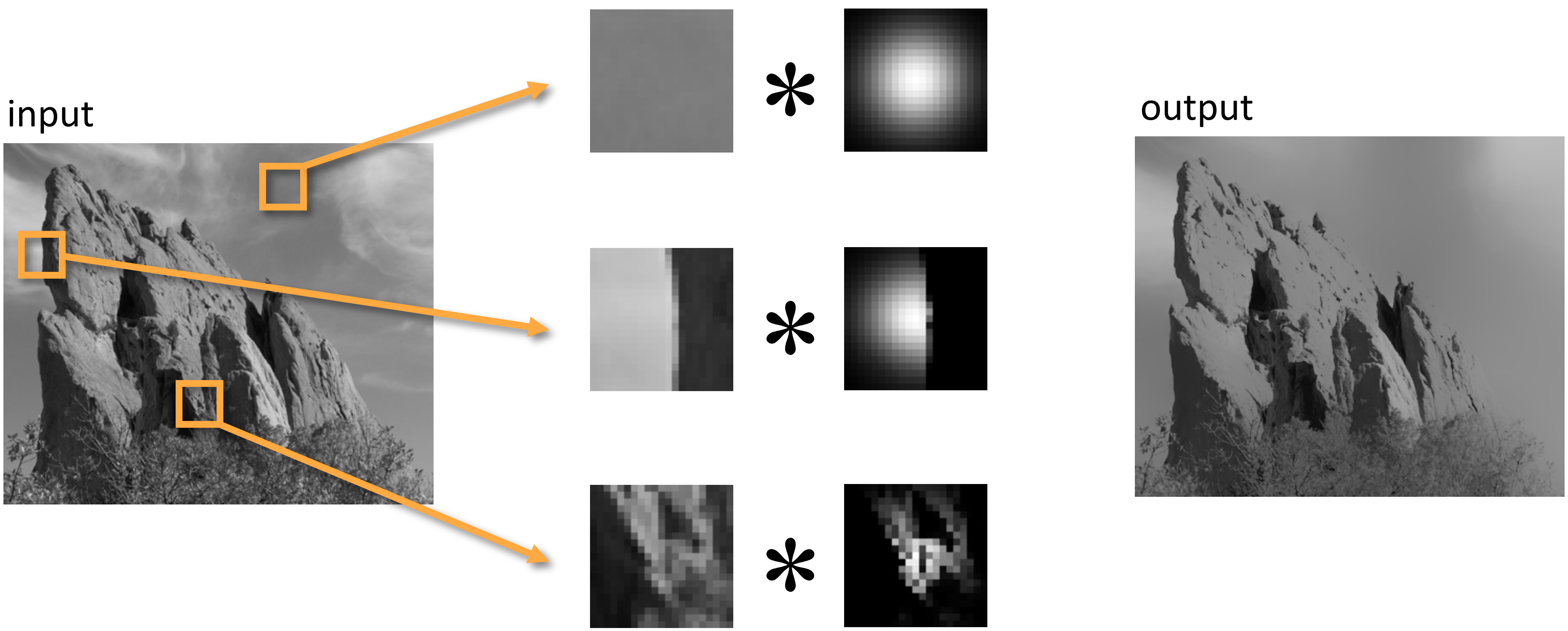
$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q} \in S} \underbrace{G_{\sigma}(\| \mathbf{p} - \mathbf{q} \|)}_{\text{space}} I_{\mathbf{q}}$$

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

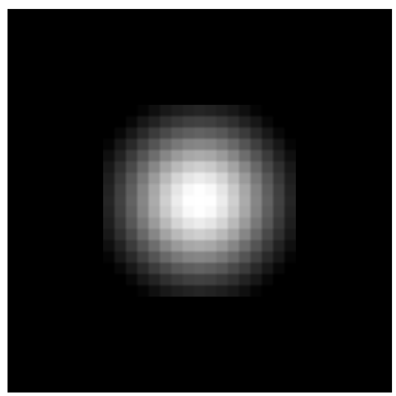
Same idea: **weighted average of pixels.**

new   not new   new

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q}\in S} G_{\sigma_{\mathrm{s}}}\left(\|\mathbf{p}-\mathbf{q}\|\right) G_{\sigma_{\mathrm{r}}}\left(|I_{\mathbf{p}} - I_{\mathbf{q}}|\right) I_{\mathbf{q}}$$

normalization factor

space weight



range weight

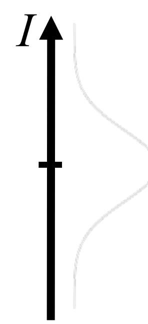
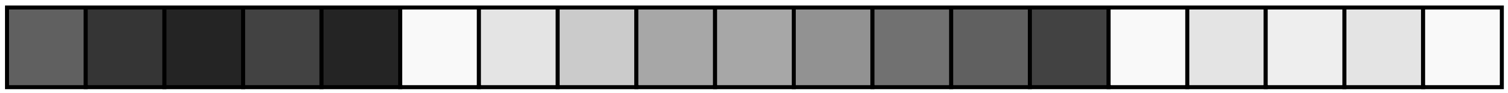
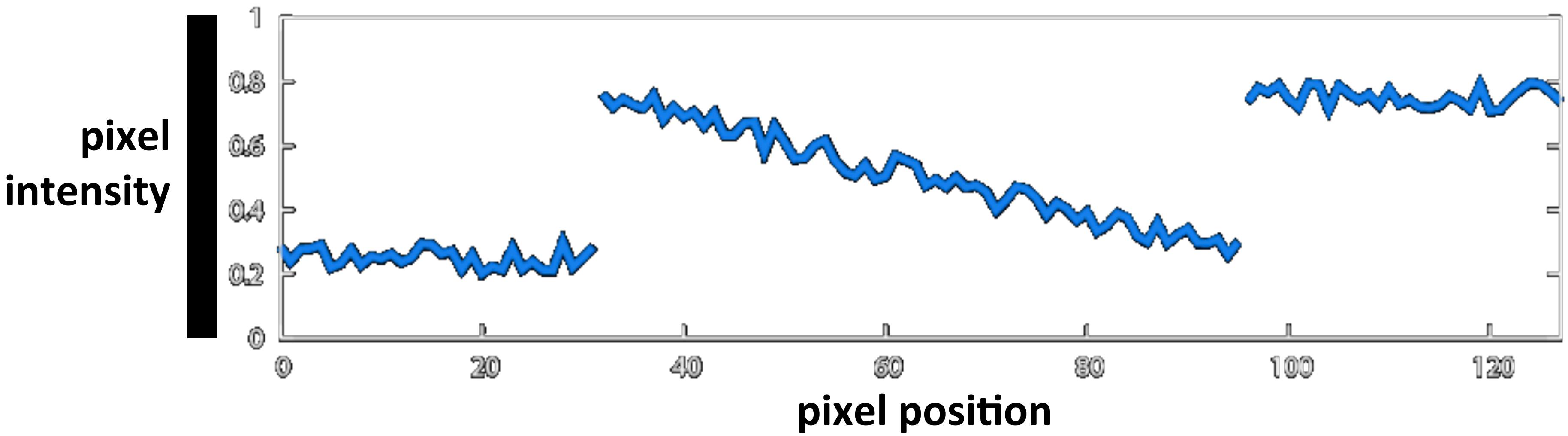


Illustration a 1D Image

- 1D image = line of pixels

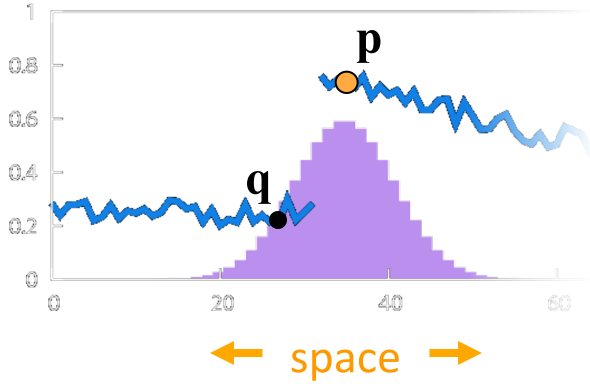


- Better visualized as a plot



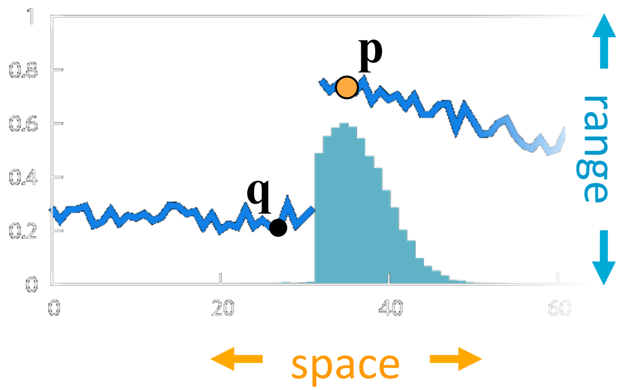
Gaussian Blur and Bilateral Filter

Gaussian blur



Bilateral filter

[Aurich 95, Smith 97, Tomasi 98]

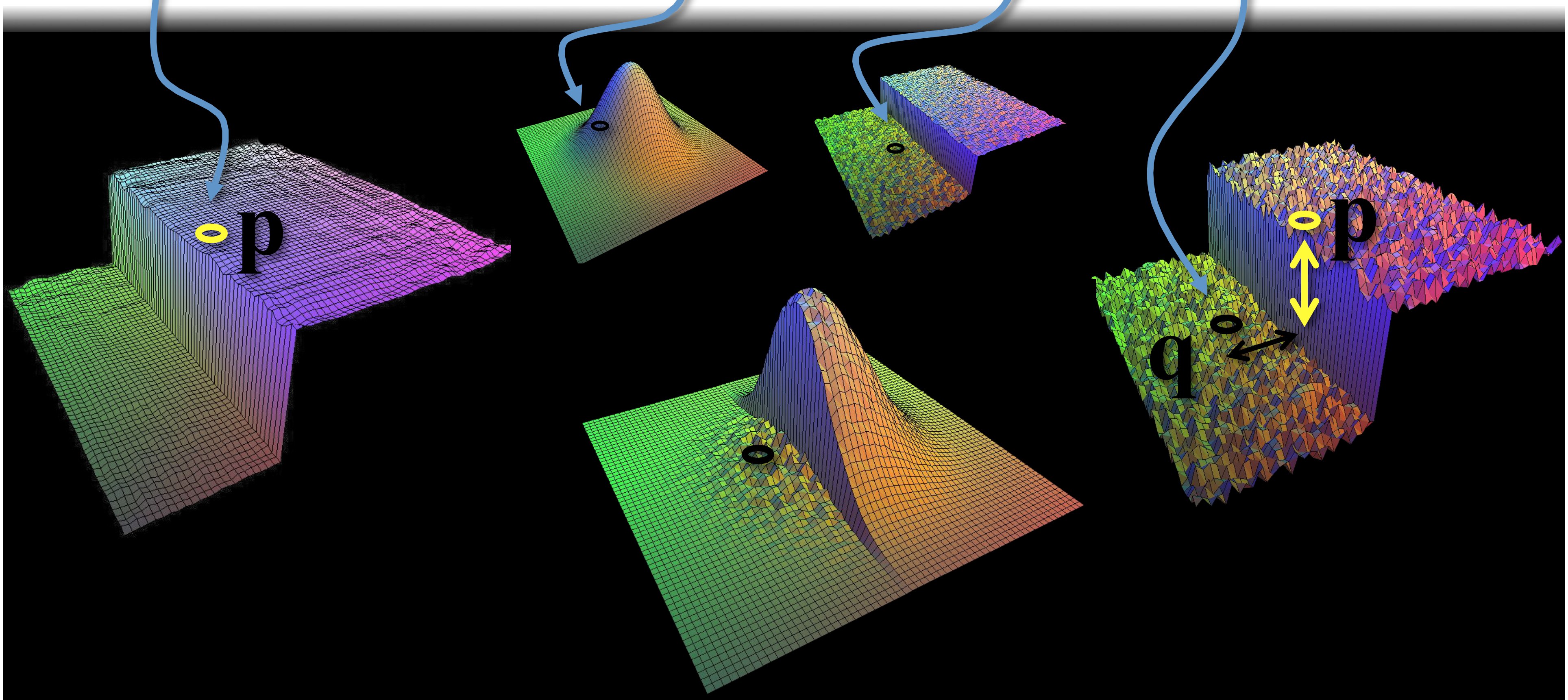


$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} \underbrace{G_{\sigma}(\|\mathbf{p}-\mathbf{q}\|)}_{\text{space}} I_{\mathbf{q}}$$

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

Bilateral Filter on a Height Field

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



reproduced from [Durand 02]

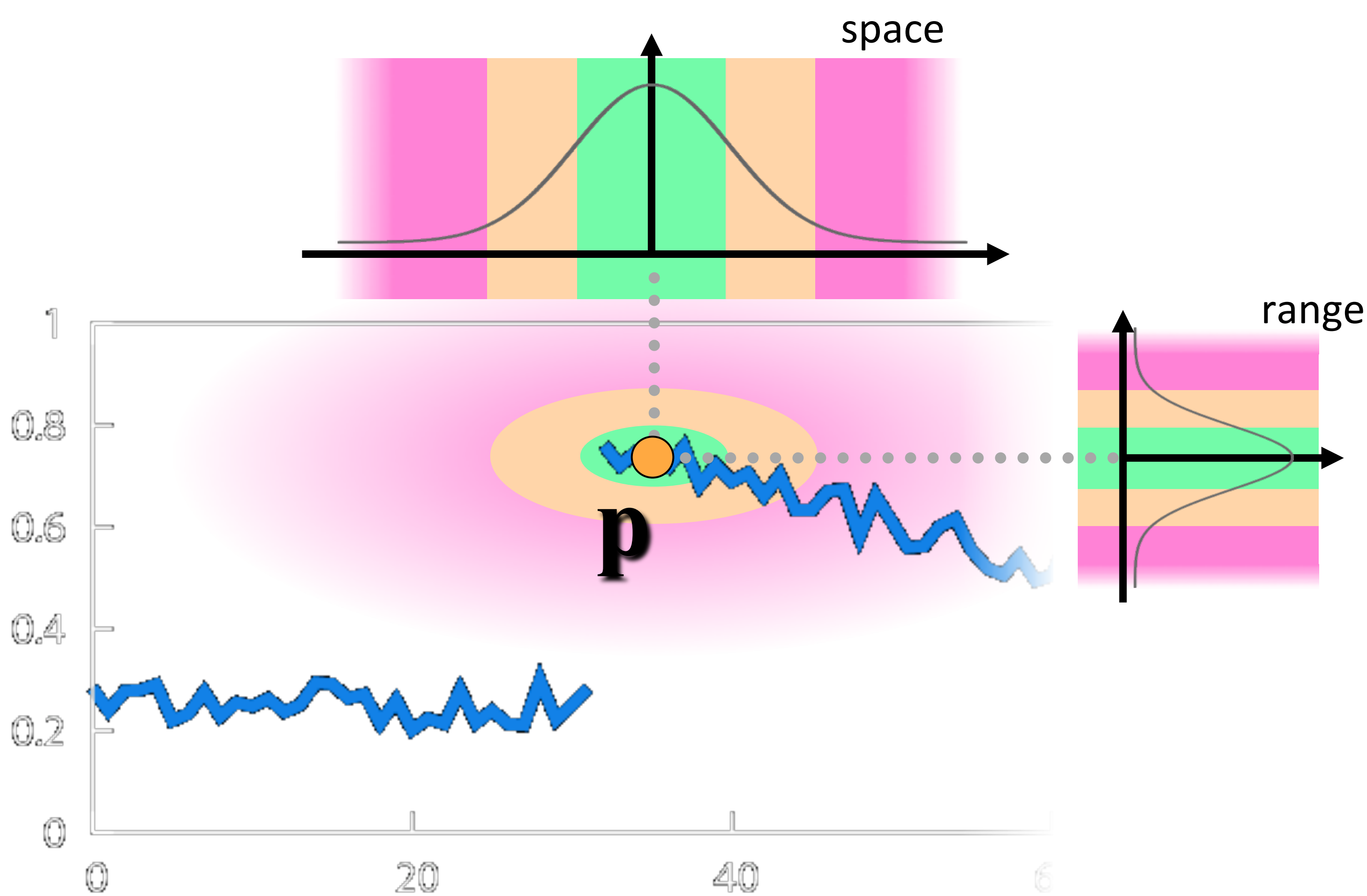
Space and Range Parameters

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_{\mathrm{s}}}\left(\| \mathbf{p} - \mathbf{q} \|\right) G_{\sigma_{\mathrm{r}}}\left(| I_{\mathbf{p}} - I_{\mathbf{q}} |\right) I_{\mathbf{q}}$$

- 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

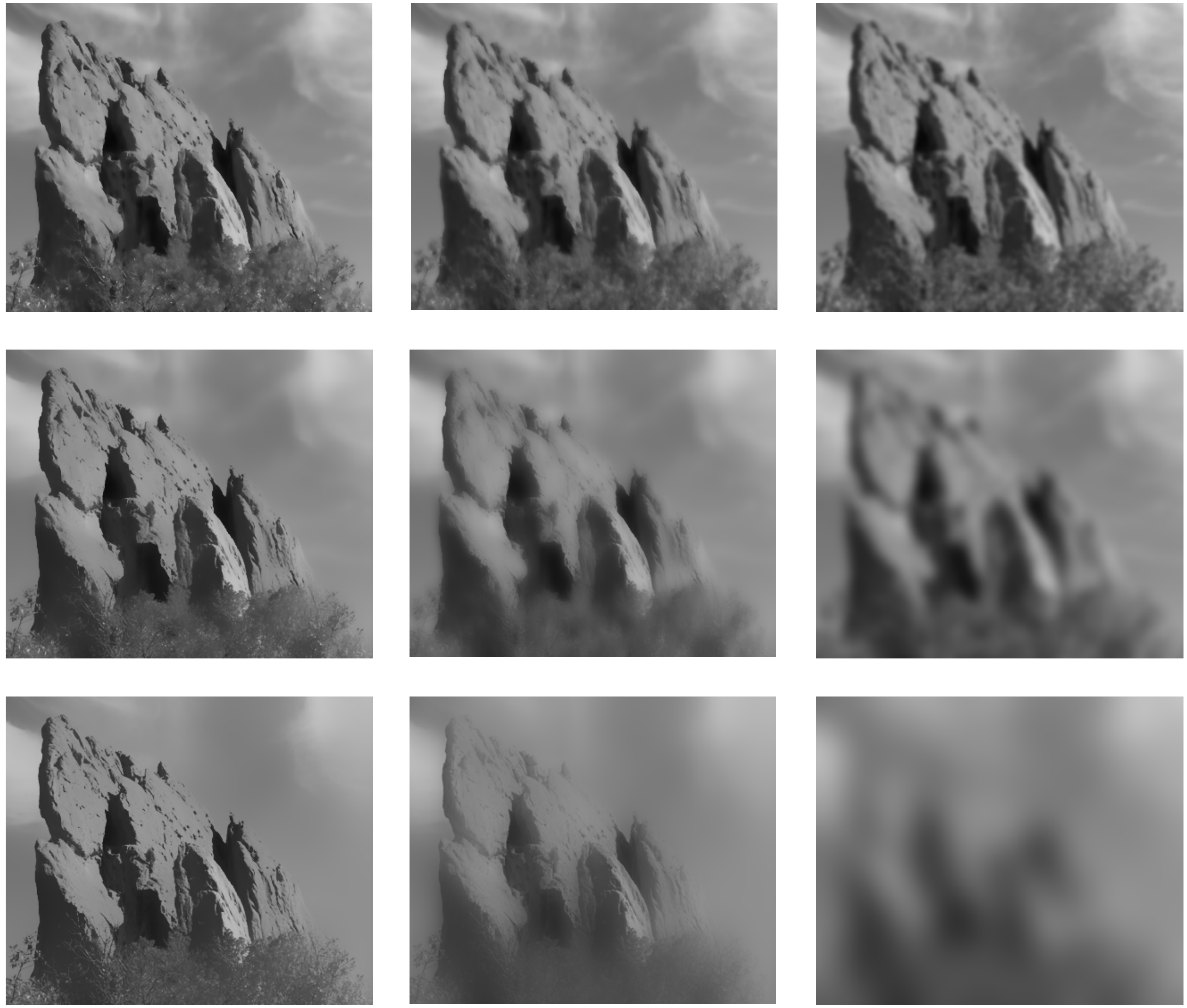
input

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

$\sigma_s = 2$

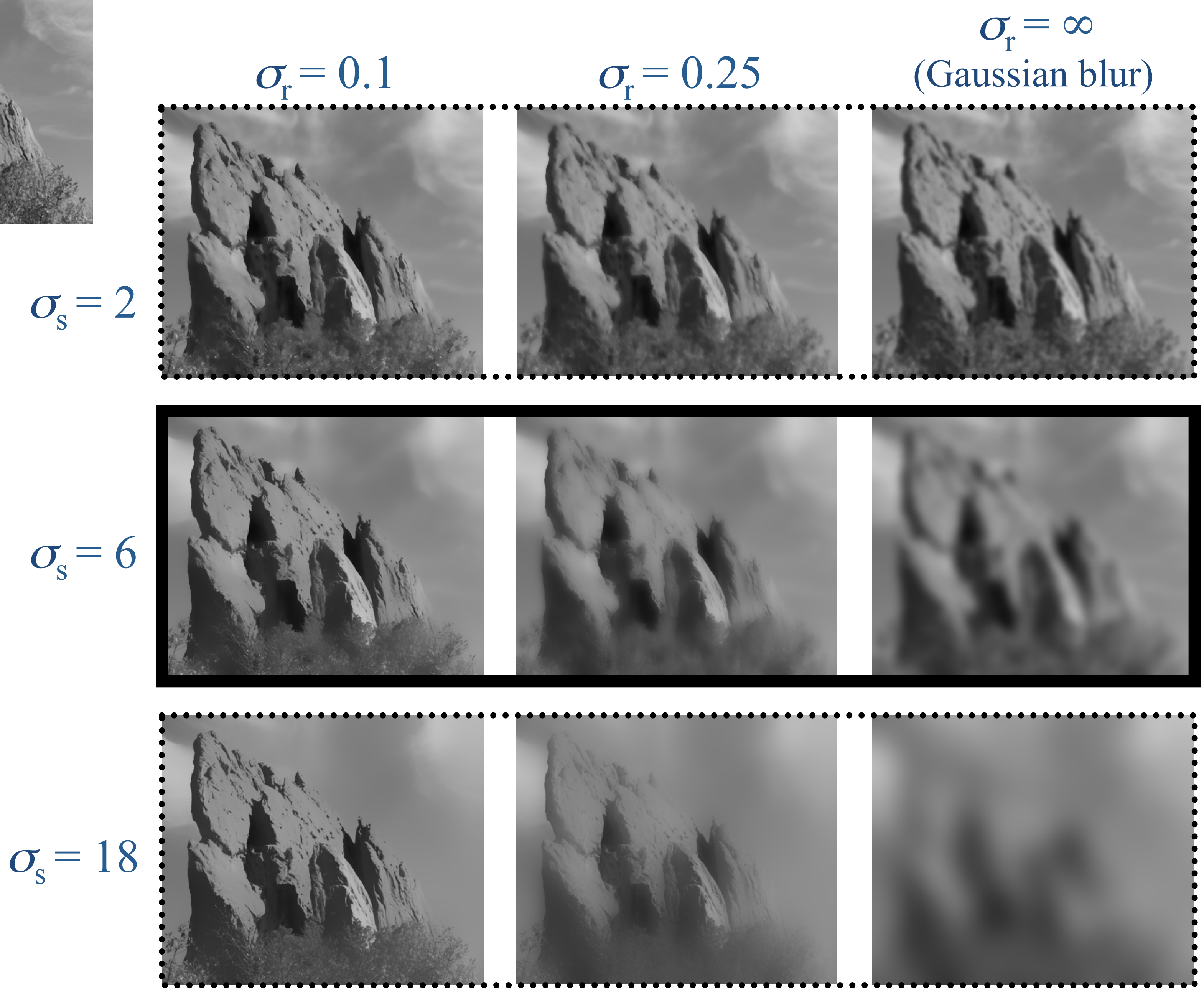
$\sigma_s = 6$

$\sigma_s = 18$



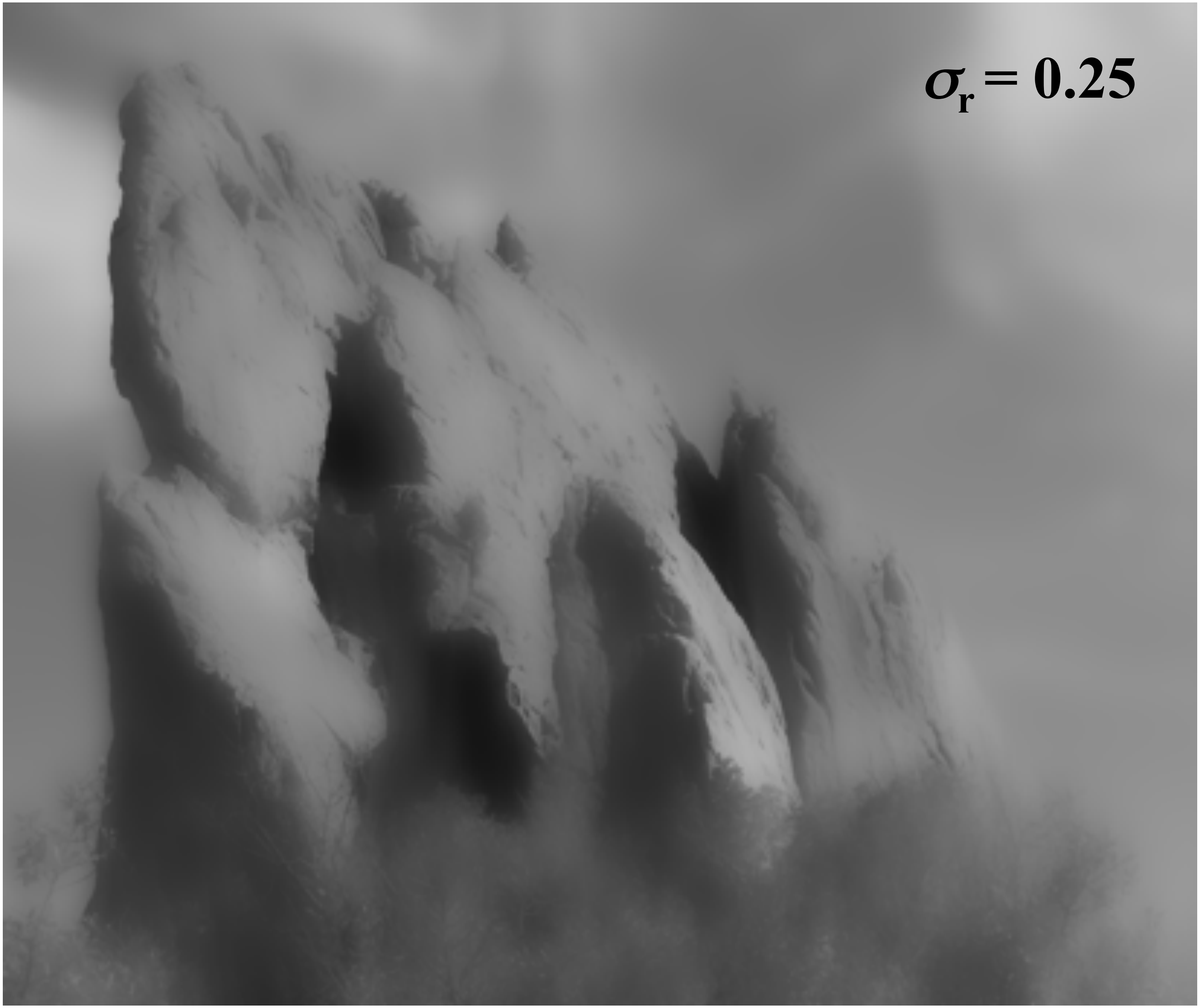
Varying the Range Parameter

input




input


$\sigma_r = 0.1$


$\sigma_r = 0.25$


$\sigma_r = \infty$
(Gaussian blur)

input

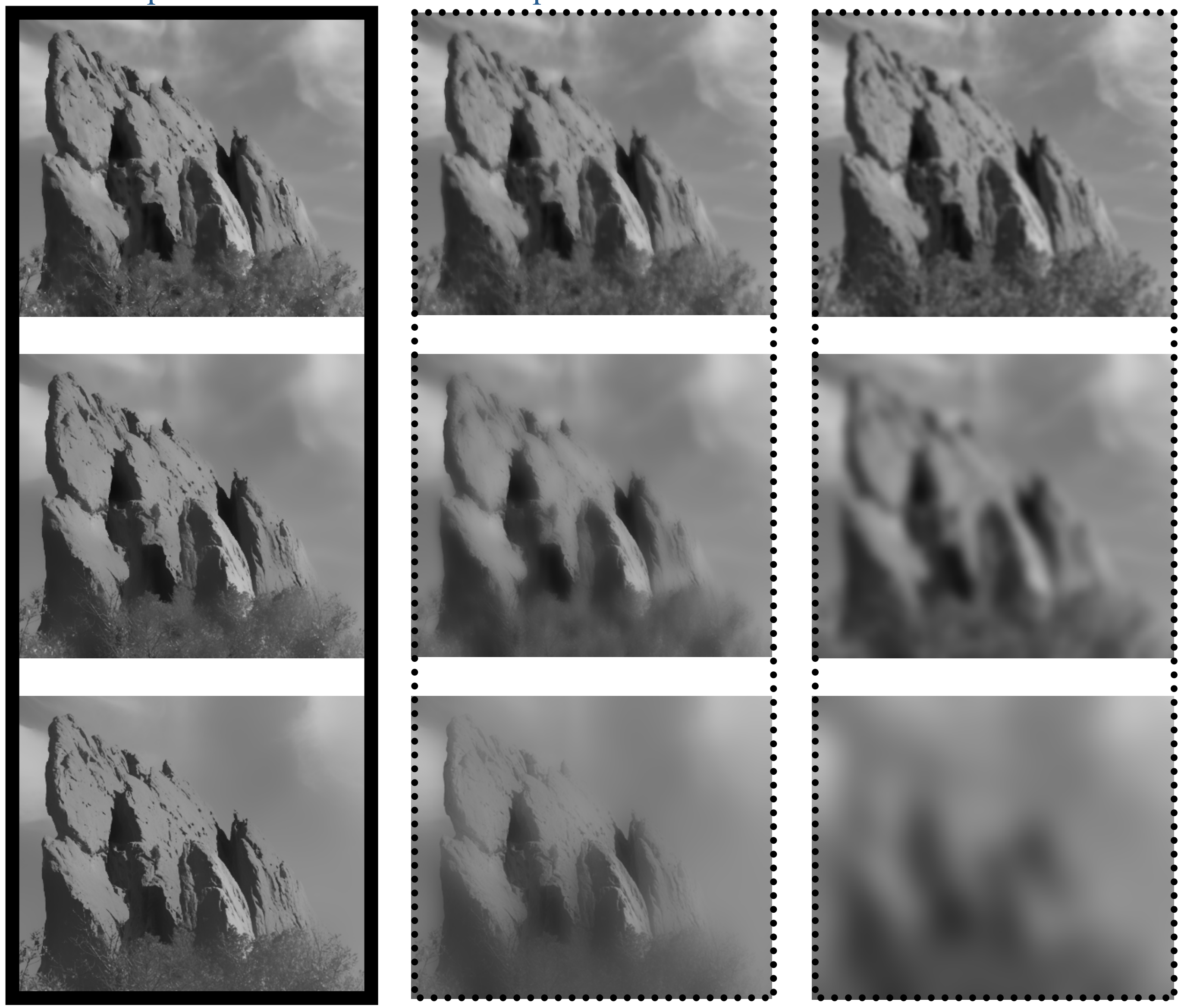
Varying the Space Parameter

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

$\sigma_s = 2$

$\sigma_s = 6$

$\sigma_s = 18$




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 ~$2\sigma_s$
- Desirable for smoothing: more pixels = more robust
- Different from diffusion that stops at thin lines

Iterating the Bilateral Filter

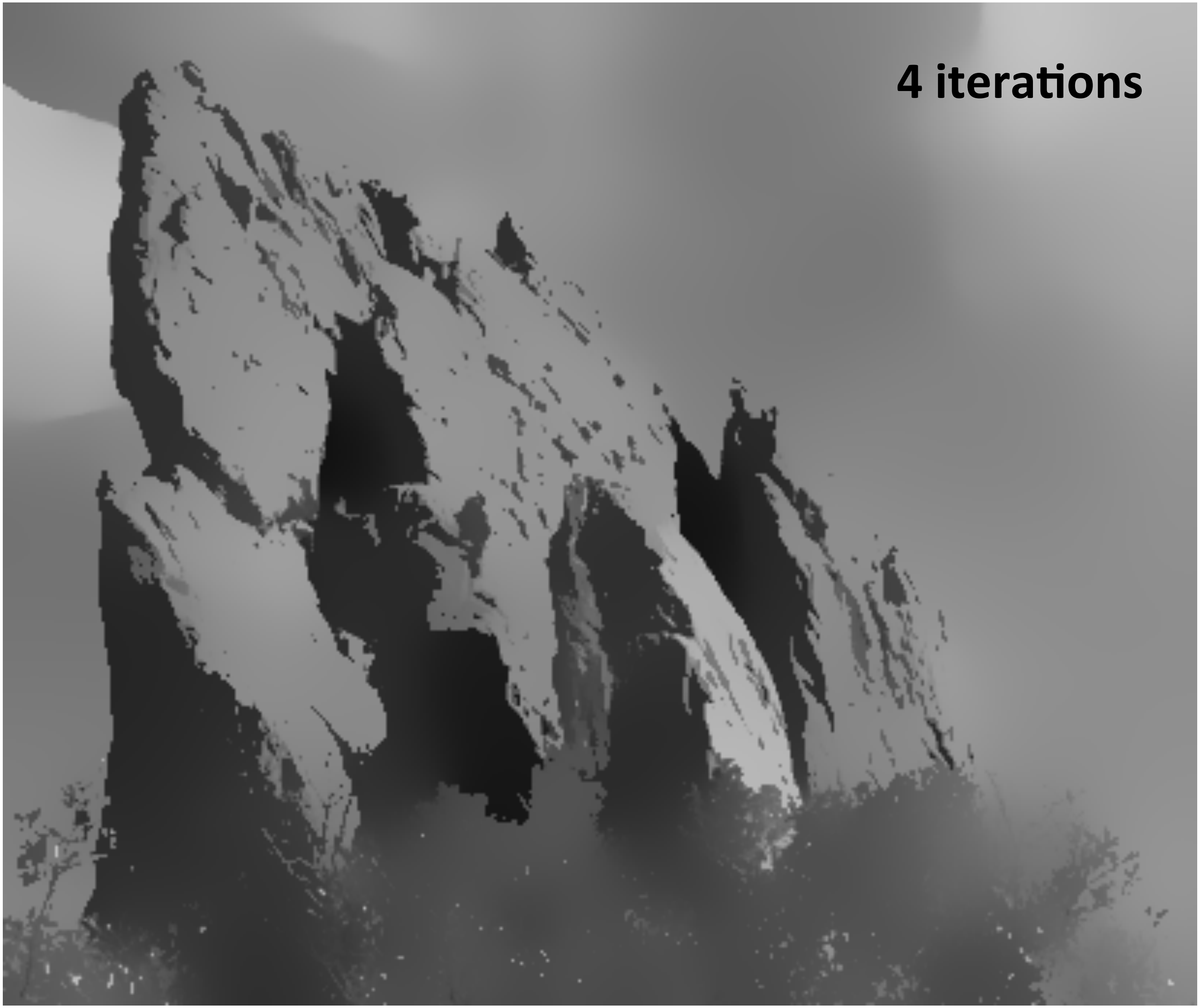
$$I_{(n+1)} = BF\left[I_{(n)}\right]$$

- 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

intensity difference

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

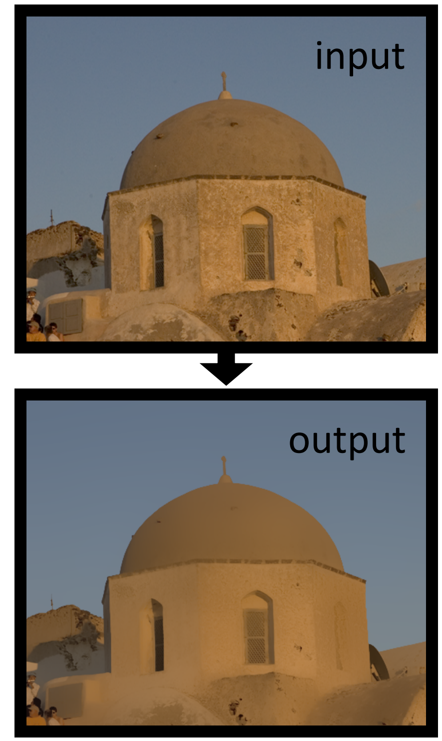
scalar

For color images

color difference

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

3D vector (RGB, Lab)



Hard to Compute

- Nonlinear $$BF[I]_{\mathbf{p}} = \boxed{\frac{1}{W_{\mathbf{p}}}} \sum_{\mathbf{q}\in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\,\boxed{G_{\sigma_r}\left(|I_{\mathbf{p}}-I_{\mathbf{q}}|\right)}\,I_{\mathbf{q}}$$
- Complex, spatially varying kernels
  - Cannot be precomputed, no FFT…

- Brute-force implementation is slow > 10min

<u>Additional Reading:</u> *Constant time O(1) Bilateral Filtering*, F. Porikli, Proc. IEEE CVPR, 1998

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

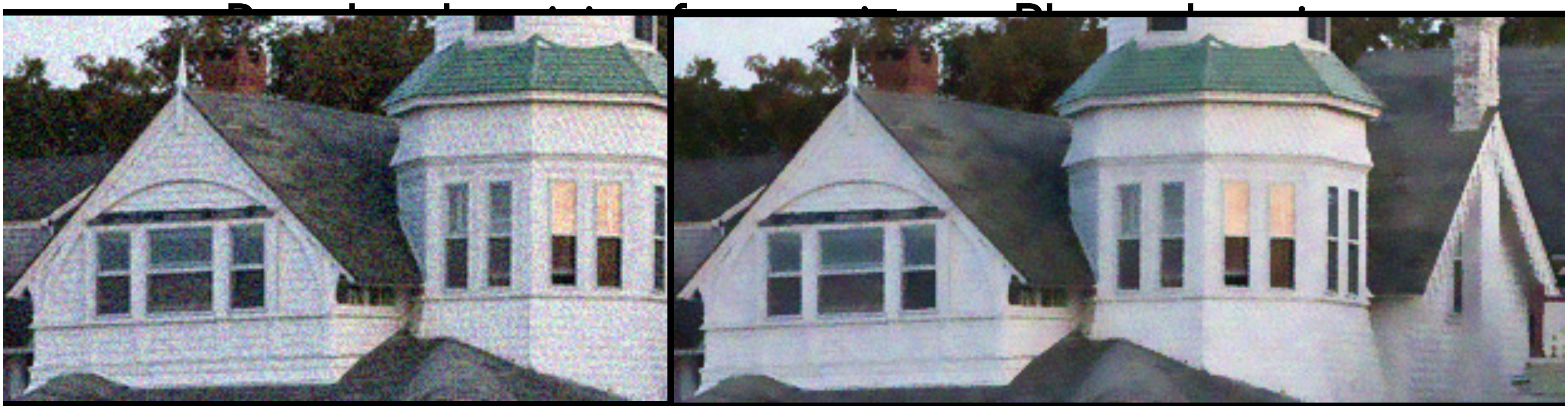
Basic denoising

Bilateral filter

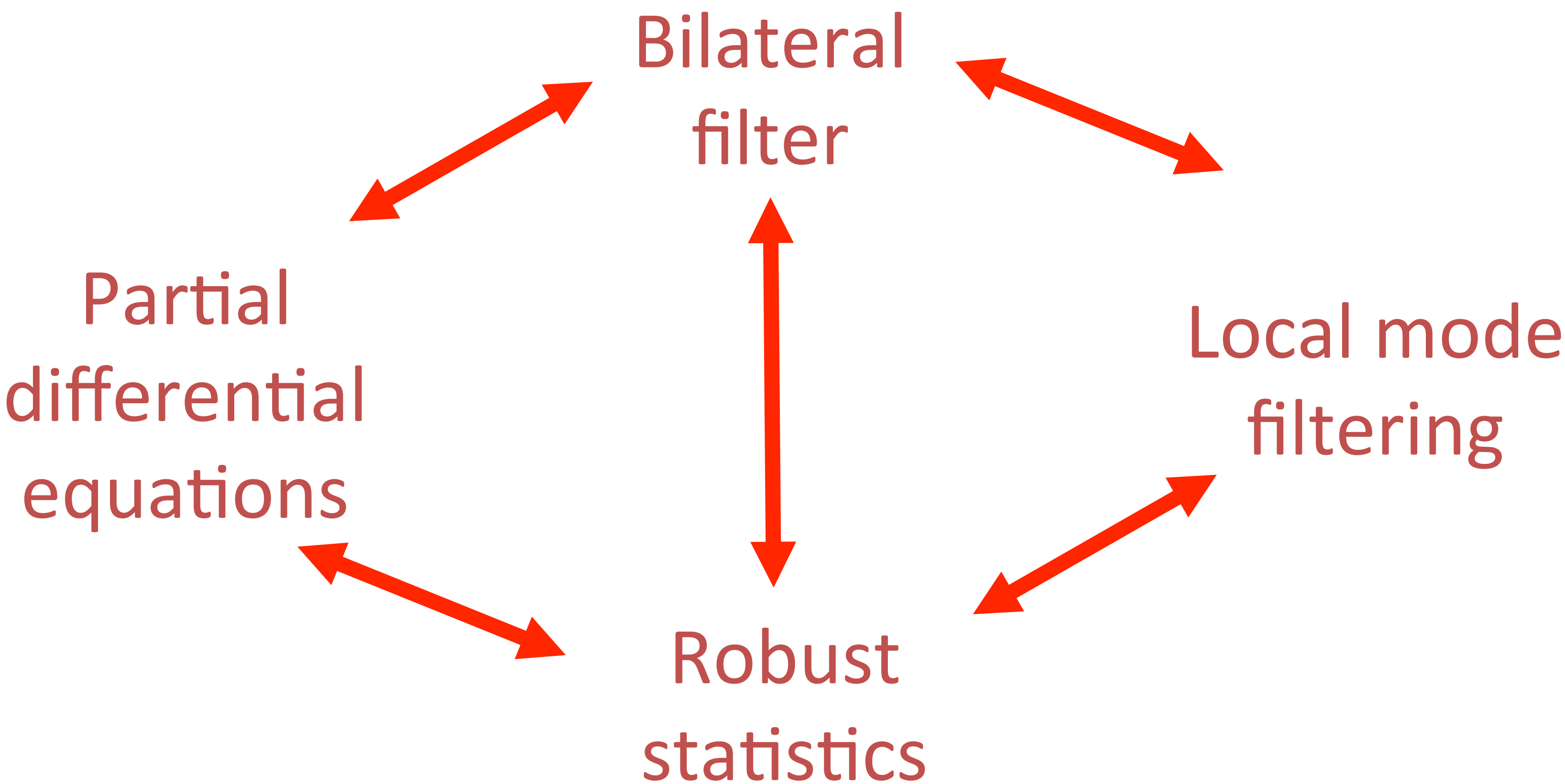
Bilateral filter – higher sigma

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



<u>Additional Reading:</u> *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'

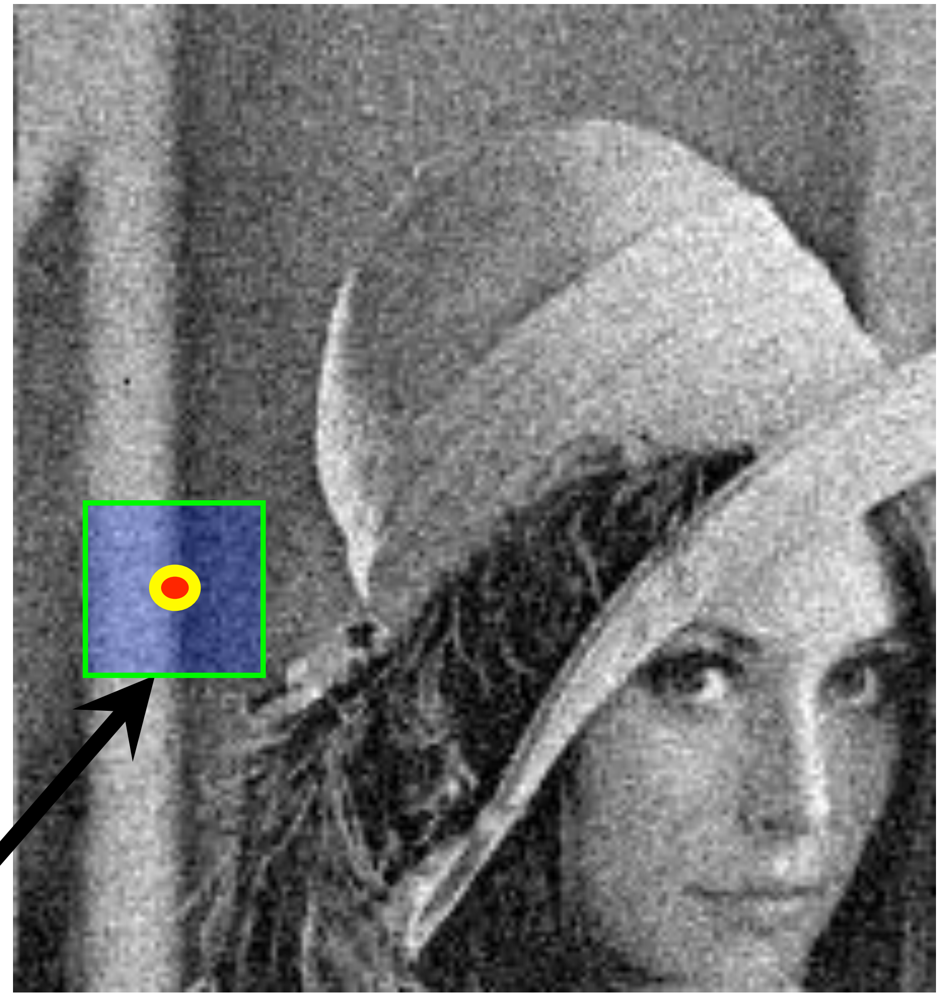
- **<u>KEY INSIGHT</u>**: Generalize, extend 'Similarity'
  - **Bilateral:**
    Averages neighbors with **<u>similar intensities</u>**;

  - **NL-Means:**
    Averages neighbors with **<u>similar neighborhoods!</u>**

NL-Means Method: Buades (2005)

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

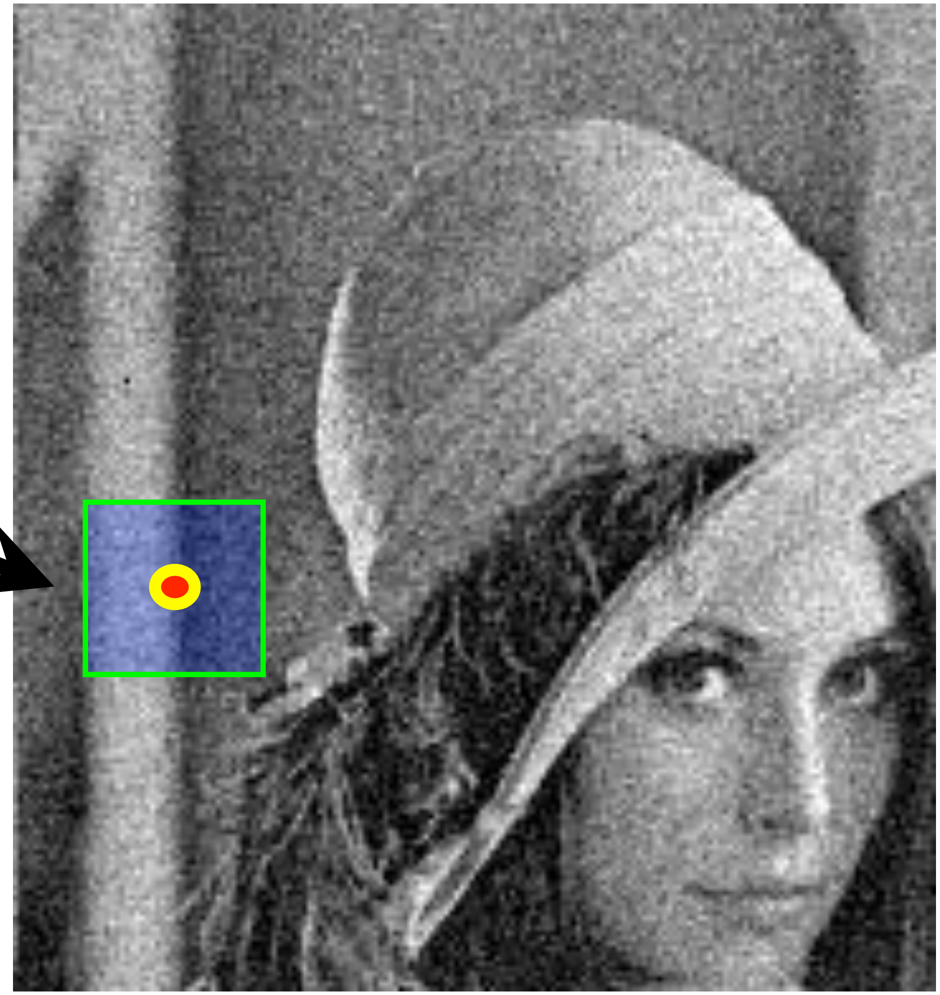
NL-Means Method: Buades (2005)



- For each and every pixel **p:**
  - Define a small, simple fixed size neighborhood;

NL-Means Method: Buades (2005)

$$\mathbf{V_p} = \begin{bmatrix} 0.74 \\ 0.32 \\ 0.41 \\ 0.55 \\ \ldots \\ \ldots \\ \ldots \\ \end{bmatrix}$$



- For each and every pixel **p**:
  - Define a small, simple fixed size neighborhood;
  - Define vector $\mathbf{V_p}$: a list of neighboring pixel values.

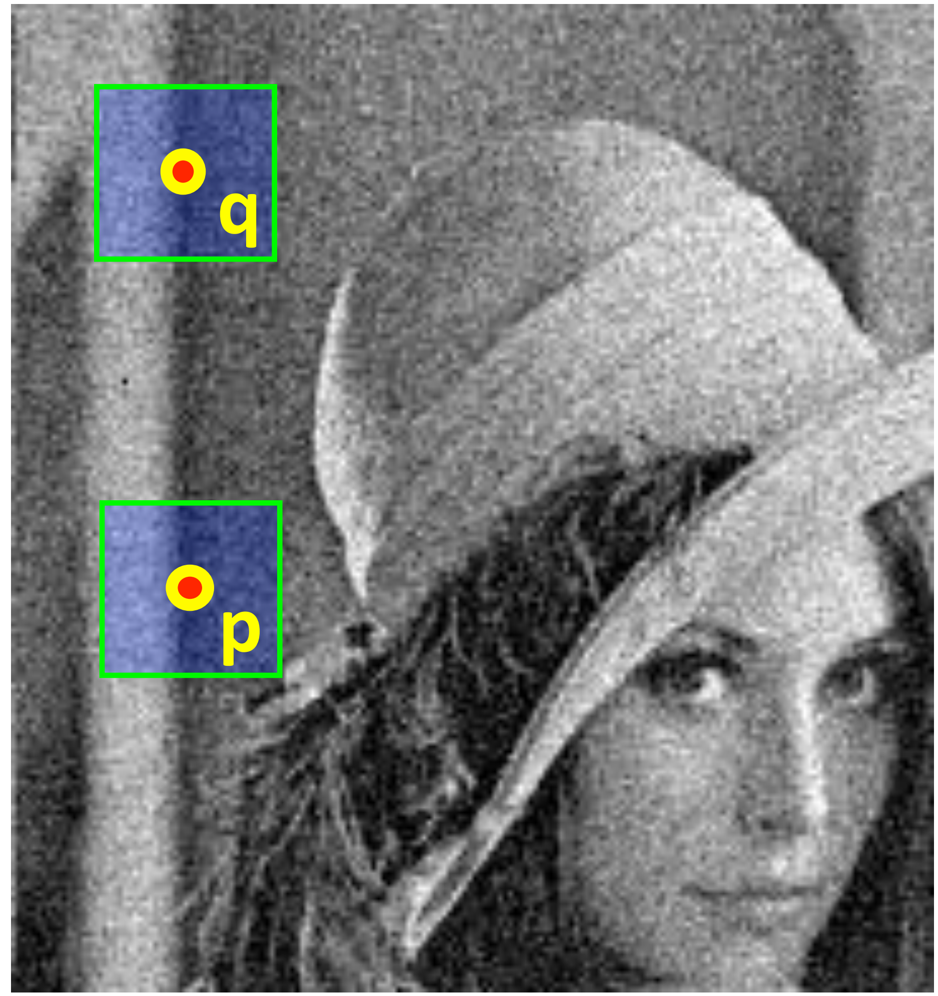
NL-Means Method: Buades (2005)

<u>'Similar'</u> pixels **p, q**

→ **SMALL**

vector distance;

$$\| \mathbf{V_p} - \mathbf{V_q} \|^2$$

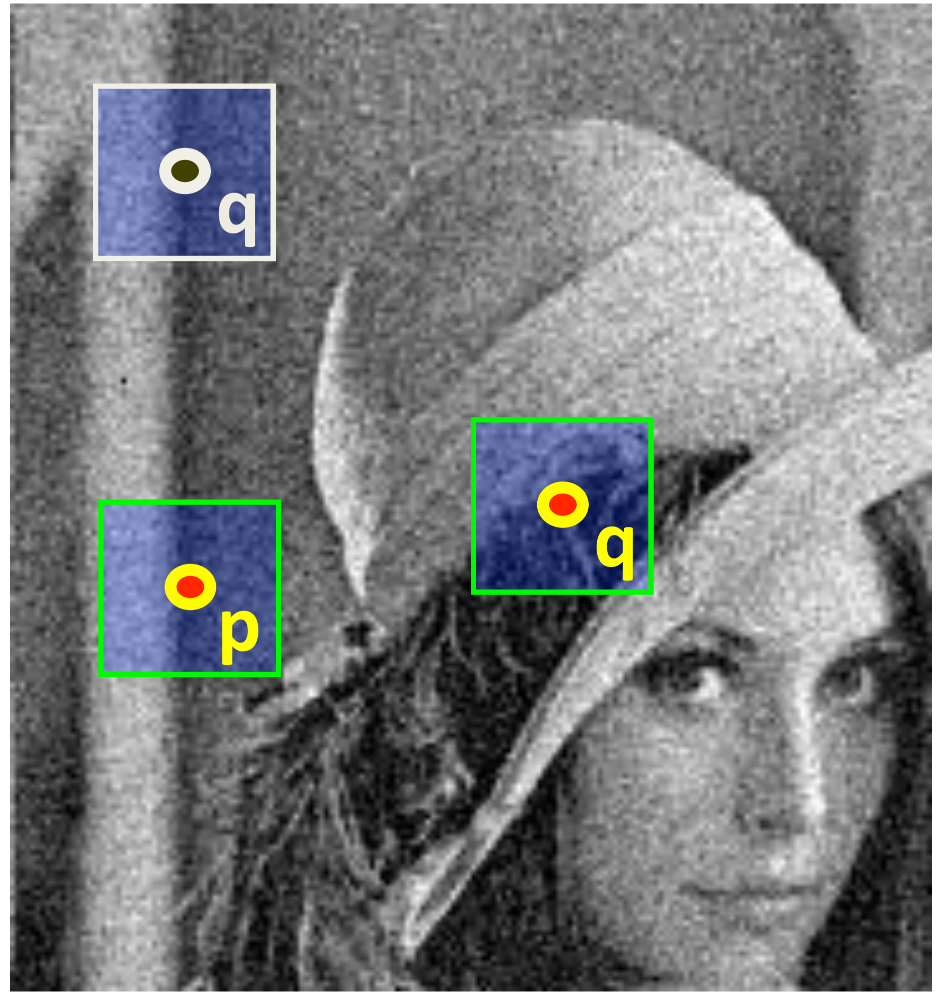


NL-Means Method: Buades (2005)

'Dissimilar' pixels **p, q**

→ **LARGE** vector distance;

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



NL-Means Method: Buades (2005)

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

$$\boxed{\| V_p - V_q \|^2}$$

Filter with this!

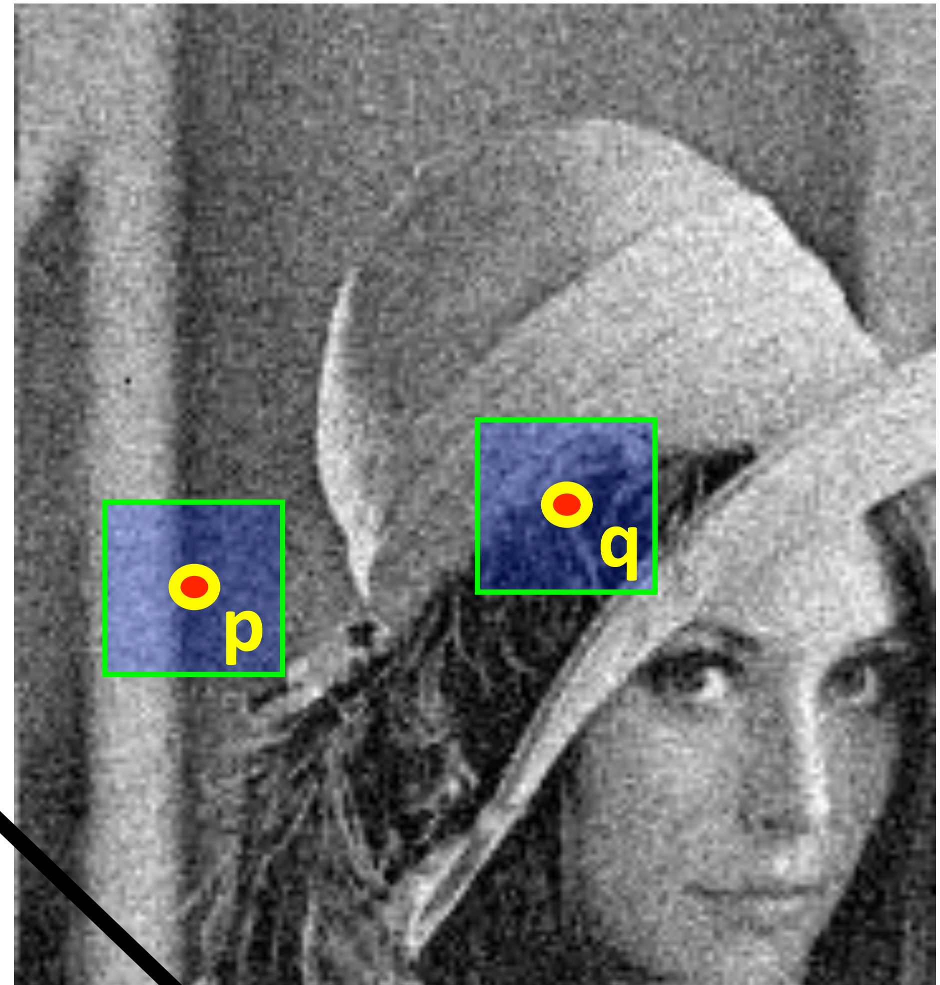
NL-Means Method: Buades (2005)

p, q neighbors define a vector distance;

$$\boxed{\| V_p - V_q \|^2}$$

Filter with this:

No spatial term!



$$NLMF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} \cancel{G_{\sigma_s}(\| \mathbf{p} - \mathbf{q} \|)} \, \boxed{G_{\sigma_r}\left(\boxed{\| \vec{V}_{\mathbf{p}} - \vec{V}_{\mathbf{q}} \|^2} \right)} I_{\mathbf{q}}$$

NL-Means Method: Buades (2005)

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

$$\boxed{\| V_p - V_q \|^2}$$

Vector Distance to p sets weight for each pixel q

$$NLMF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_r}\left(\| \vec{V}_{\mathbf{p}} - \vec{V}_{\mathbf{q}} \|^2 \right) I_{\mathbf{q}}$$

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.