SIGGRAPH2008

A Gentle Introduction to Bilateral Filtering and its Applications

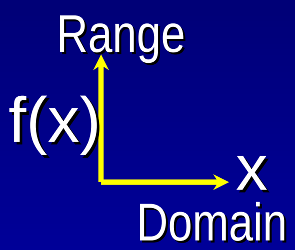
SIGGRAPH2008

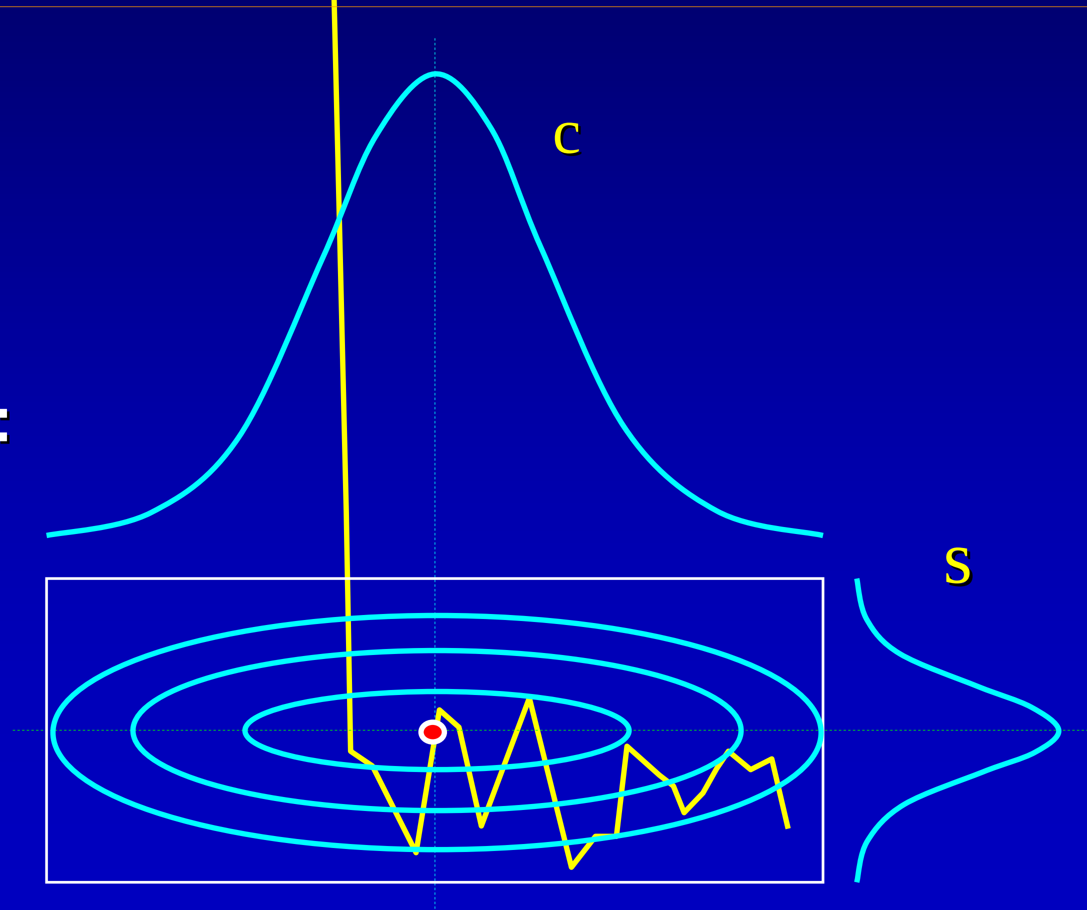
07/10: Novel Variants of the Bilateral Filter

Jack Tumblin – EECS, Northwestern University

Review: Bilateral Filter

A 2-D filter window: weights vary with intensity





2 Gaussian Weights:
product =
ellisoidal footprint

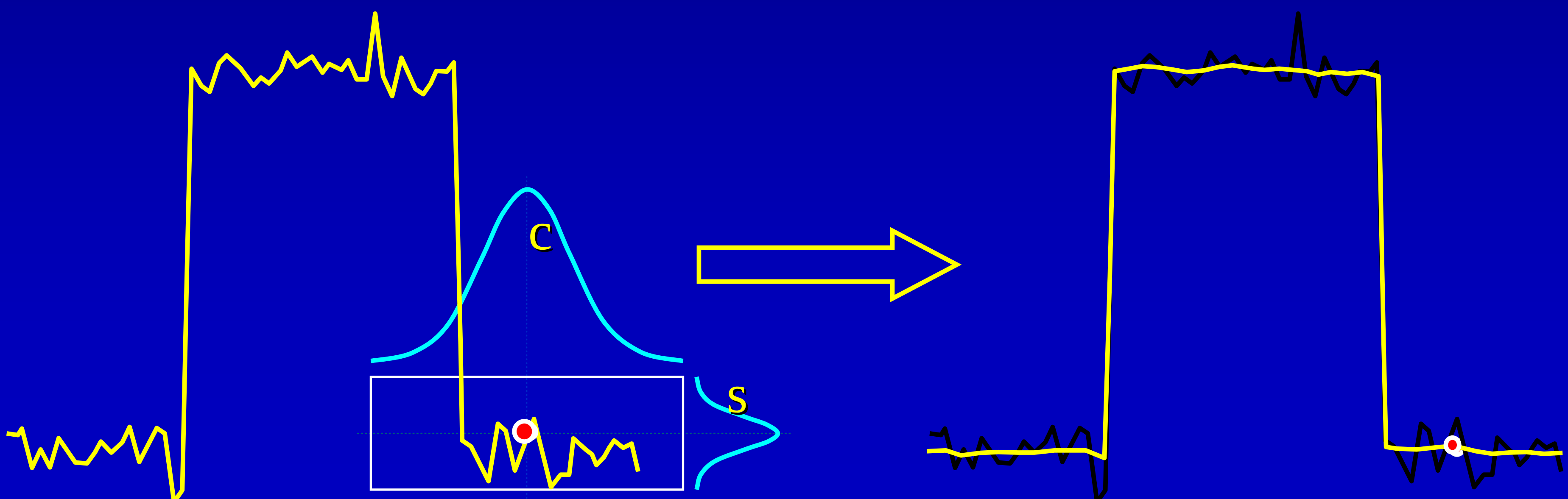
Normalize weights to always sum to 1.0

Review: Bilateral Strengths

Piecewise smooth result

- averages local small details, ignores outliers
- preserves steps, large-scale ramps, and curves,...

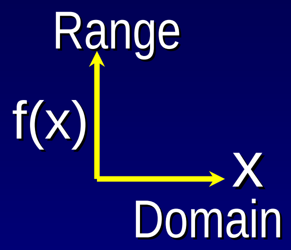
- Some equivalence to anisotropic diffusion, robust statistics [Black98,Elad02,Durand02]
- Simple & Fast (esp. w/ [Durand02], [Paris06], [Porikli08] other speedup methods)

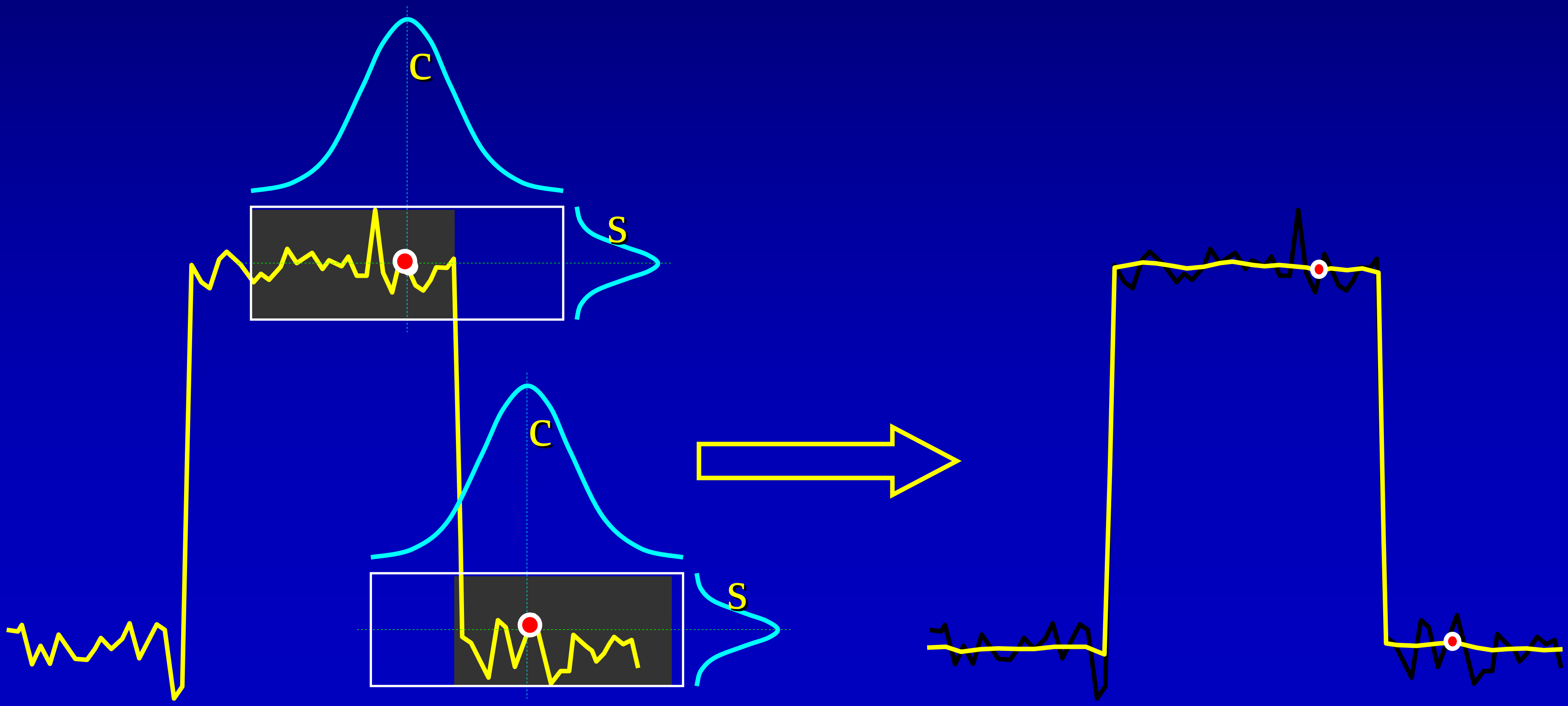


Review: Bilateral Filter

Why it works: graceful segmentation

- Smoothing for 'similar' parts *ONLY*
- Range Gaussian **s** acts as a 'filtered region' finder





Bilateral Filter Variants

- Before the 'Bilateral' name :
  - Yaroslavsky (1985): T.D.R.I.M.
  - Smith & Brady (1997): SUSAN

And now, a growing set of extended variants:

- 'Trilateral' Filter (Choudhury et al., EGSR 2003)
- Cross-Bilateral (Petschnigg04, Eisemann04)
- NL-Means (Buades05)
- Bilateral Retinex(Elad05), Joint-Bilateral Upsampling (Kopf07), many more exist…

And many more coming: application driven…

Who was first? Many Pioneers

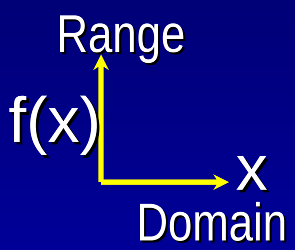
- Elegant, Simple, Broadly useful Idea

  → → 'Invented' several times

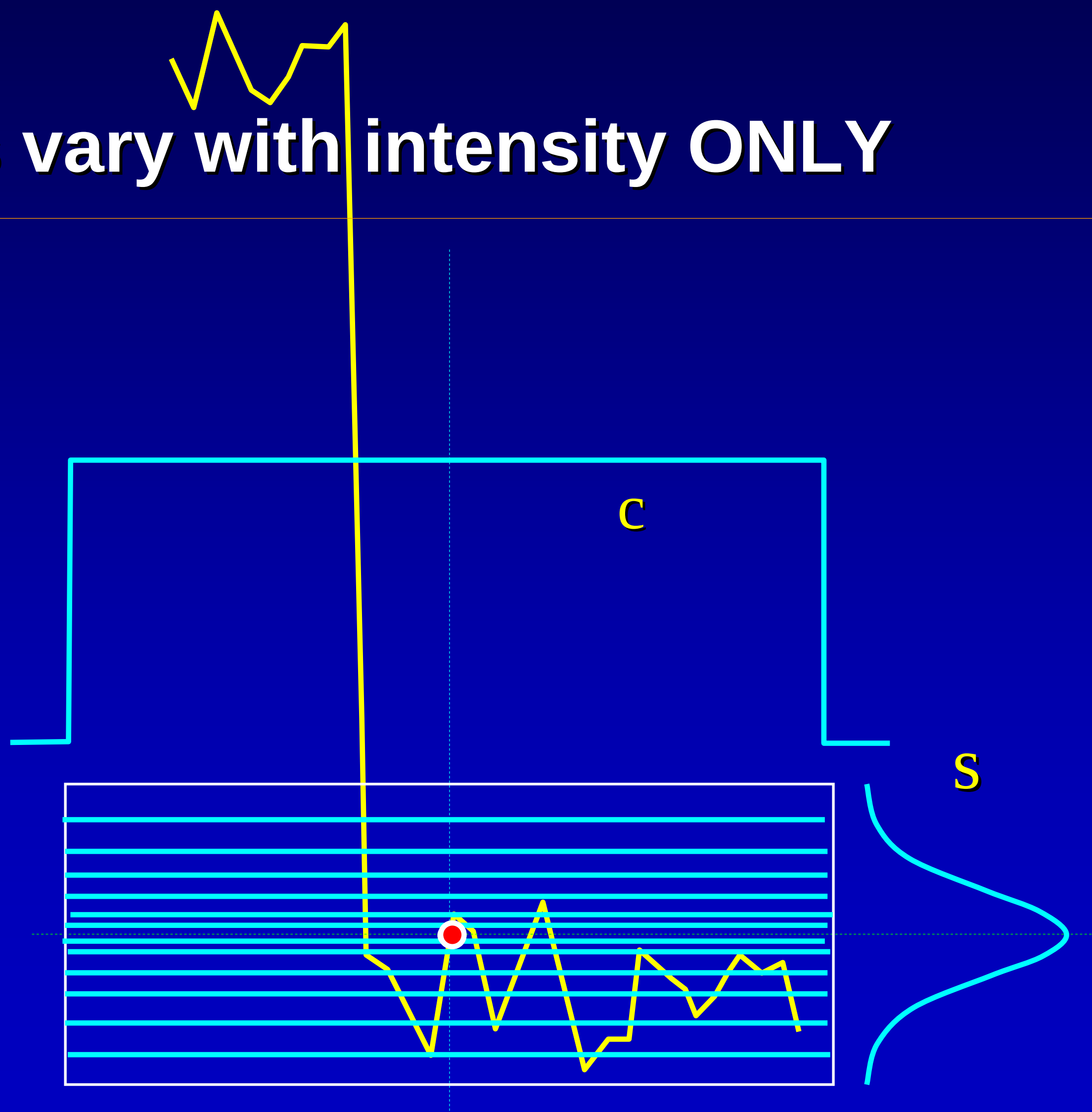
- Different Approaches, Increasing Clarity
  - Yaroslavsky(1985):
    'Transform Domain Image Restoration Methods'
  - Smith & Brady (1995): 'SUSAN'
    "Smallest Univalue
    Segment-Assimilating Nucleus"
  - Tomasi & Manduchi(1998): 'Bilateral Filter'

New Idea!
1985 Yaroslavsky:

A 2-D filter window:
weights vary with intensity ONLY



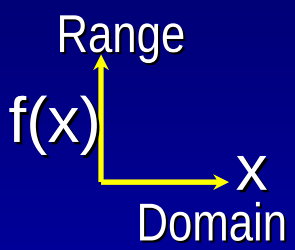
Square neighborhood,
Gaussian Weighted
'similarity'



Normalize weights to always sum to 1.0

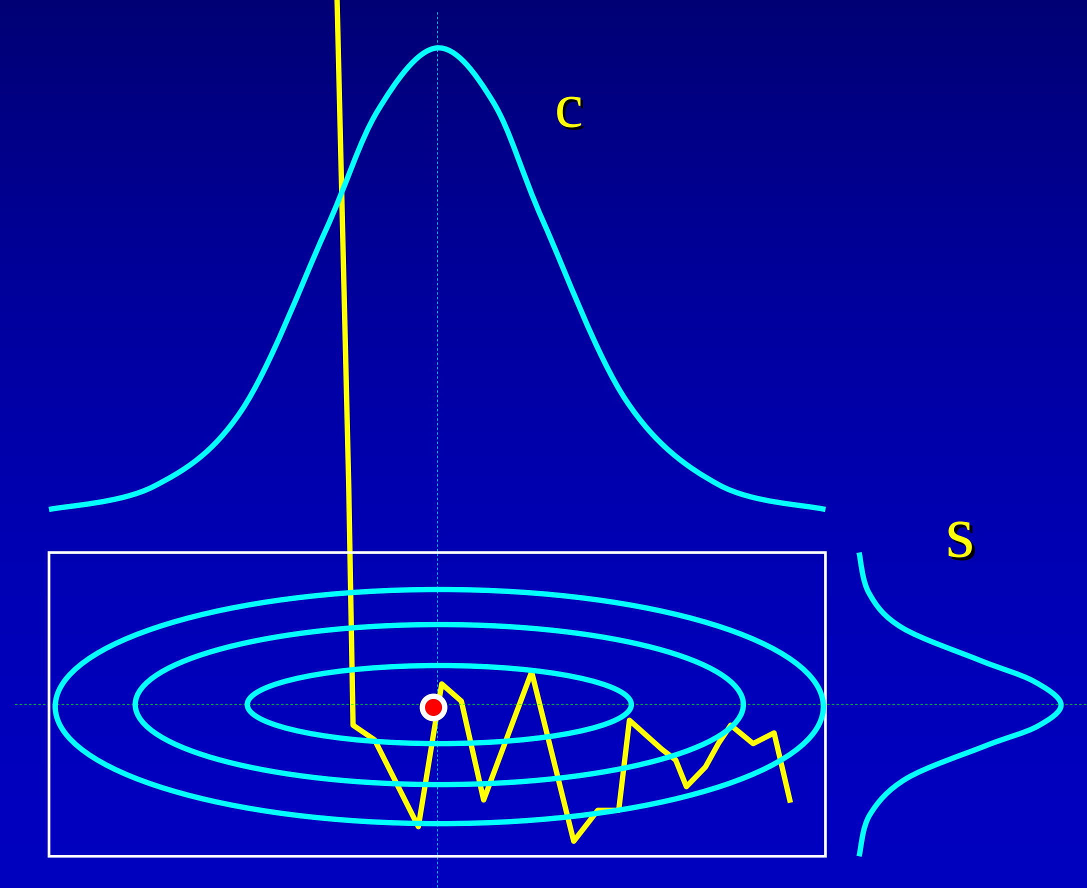
New Idea!
1995 Smith: 'SUSAN' Filter

A 2-D filter window: weights vary with intensity



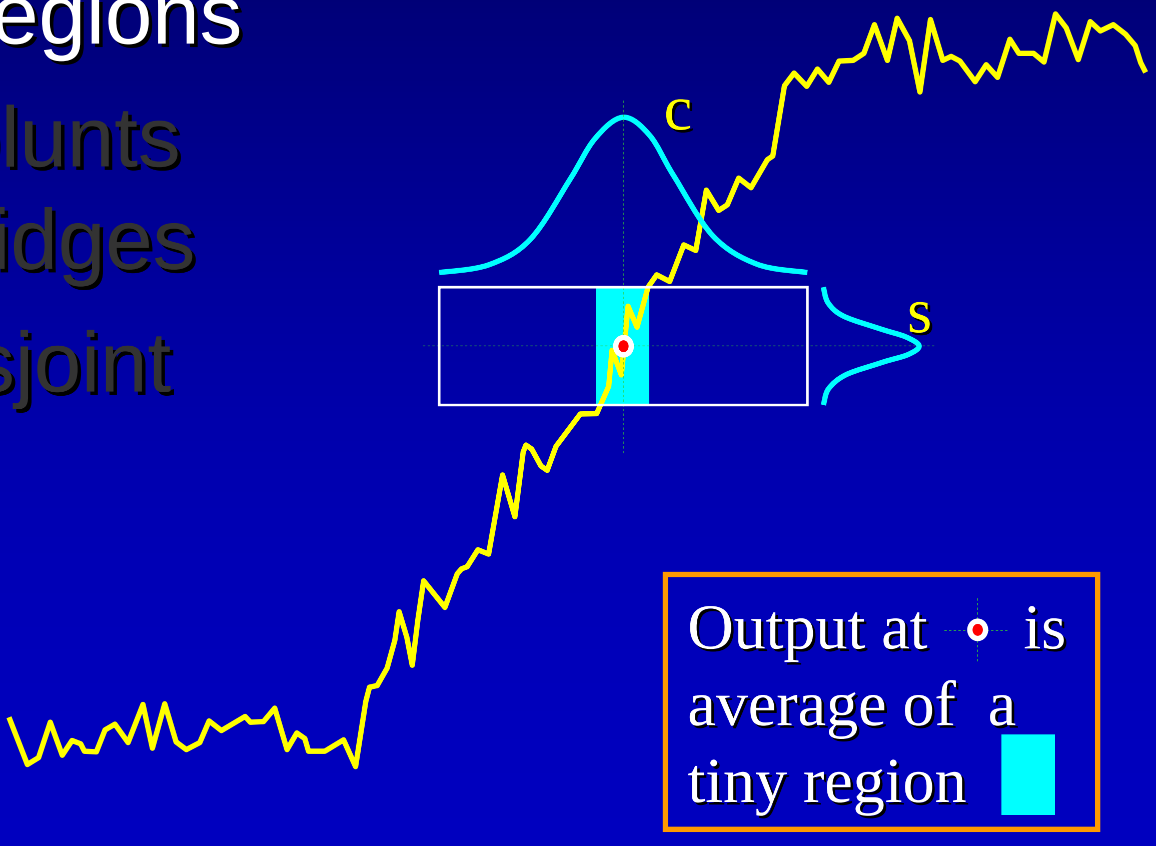
2 Gaussian Weights:
product =
ellisoidal footprint

Normalize weights to always sum to 1.0



Bilateral Filter: 3 Difficulties

- Poor Smoothing in High Gradient Regions
- Smoothes and blunts cliffs, valleys & ridges
- Can combine disjoint signal regions

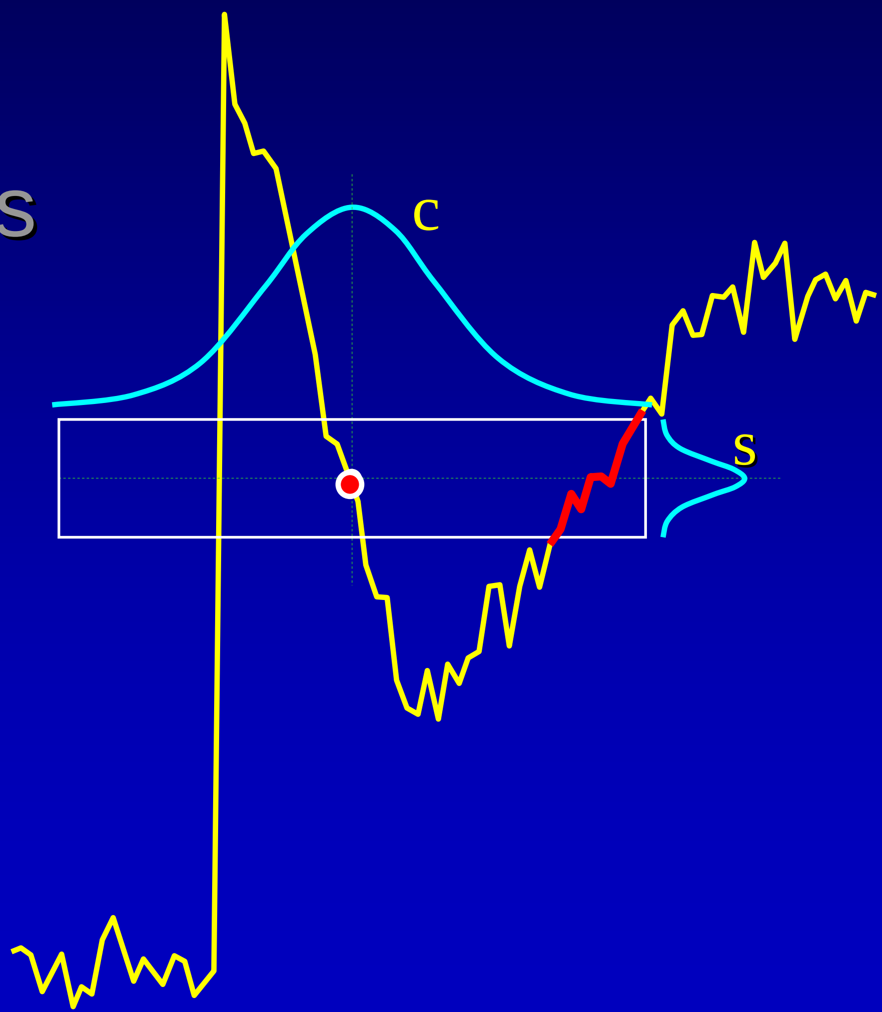


Bilateral Filter: 3 Difficulties

- Poor Smoothing in High Gradient Regions
- Smoothes and blunts cliffs, valleys & ridges
- Can combine disjoint signal regions

Bilateral Filter: 3 Difficulties

- Poor Smoothing in High Gradient Regions
- Smoothes and blunts cliffs, valleys & ridges
- Disjoint regions can blend together



‘Blunted Corners’ → Weak Halos

Bilateral :

What we get

‘Blunted Corners’ → Weak Halos

What we want

‘Trilateral’ result

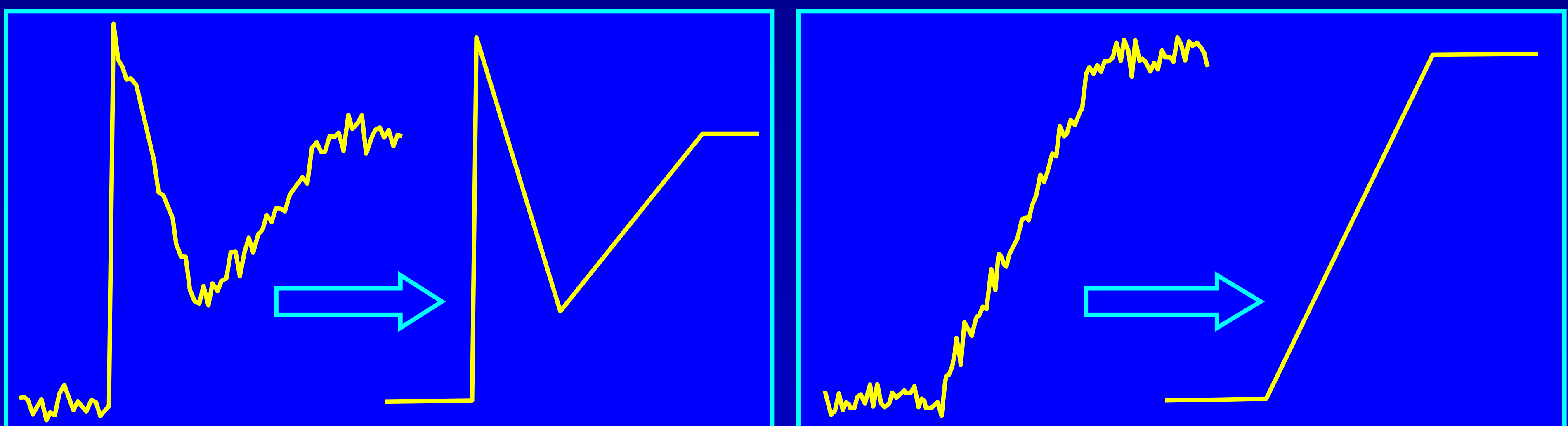
Try to fix this:
Trilateral Filter (Choudhury 2003)

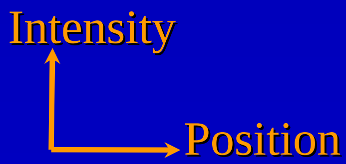
Goal:

Piecewise linear smoothing, not piecewise constant

Method:

‘Steer’ Bilateral Filter with smoothed gradients





EXAMPLE: remove noise from a piecewise linear scanline

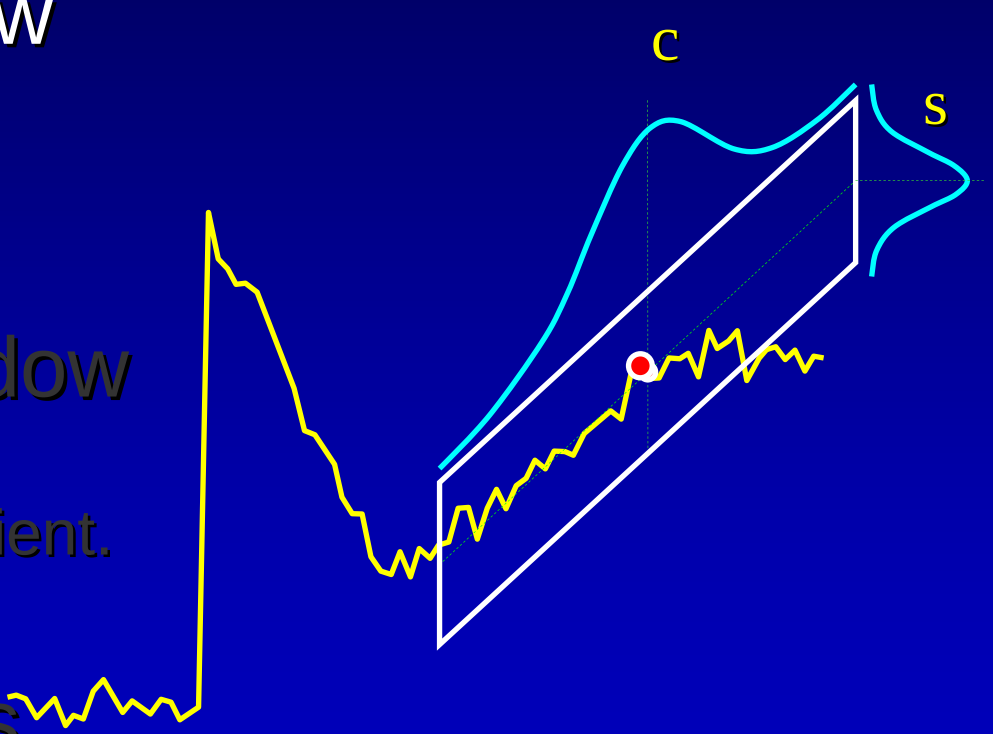
Outline: Bilateral→Trilateral Filter

Three Key Ideas:

- **Tilt** the filter window
  according to bilaterally-
  smoothed gradients

- **Limit** the filter window
  to connected regions
  of similar smoothed gradient.

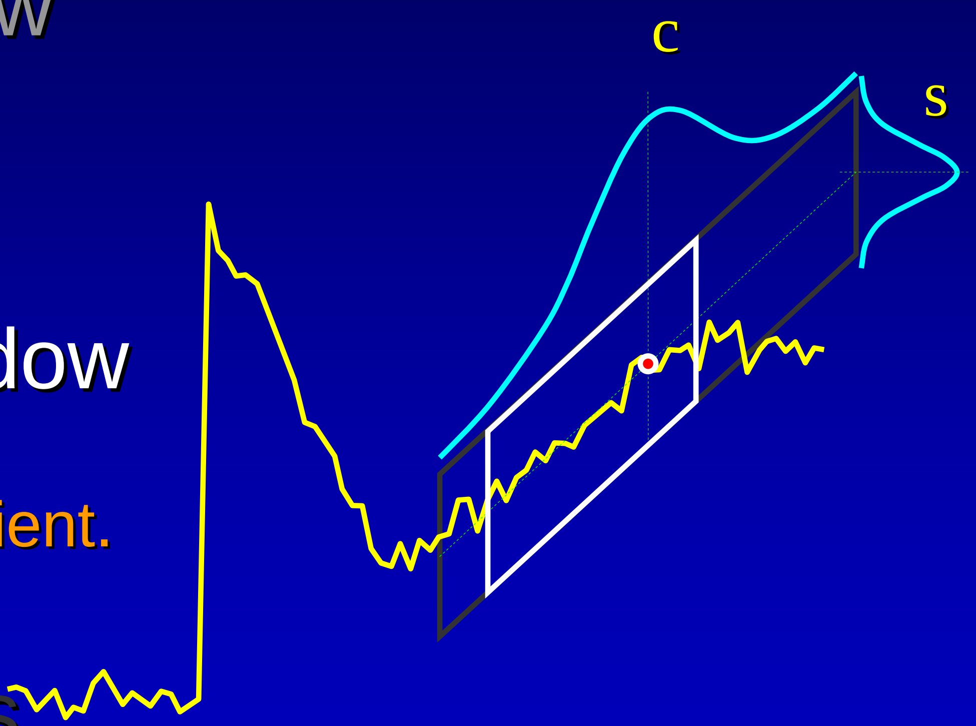
- **Adjust** Parameters
  from measurements
  of the windowed signal



Outline: Bilateral→Trilateral Filter

Key Ideas:

- **Tilt** the filter window
  according to bilaterally-smoothed gradients
- **Limit** the filter window
  to connected regions of similar smoothed gradient.
- **Adjust** Parameters
  from measurements of the windowed signal



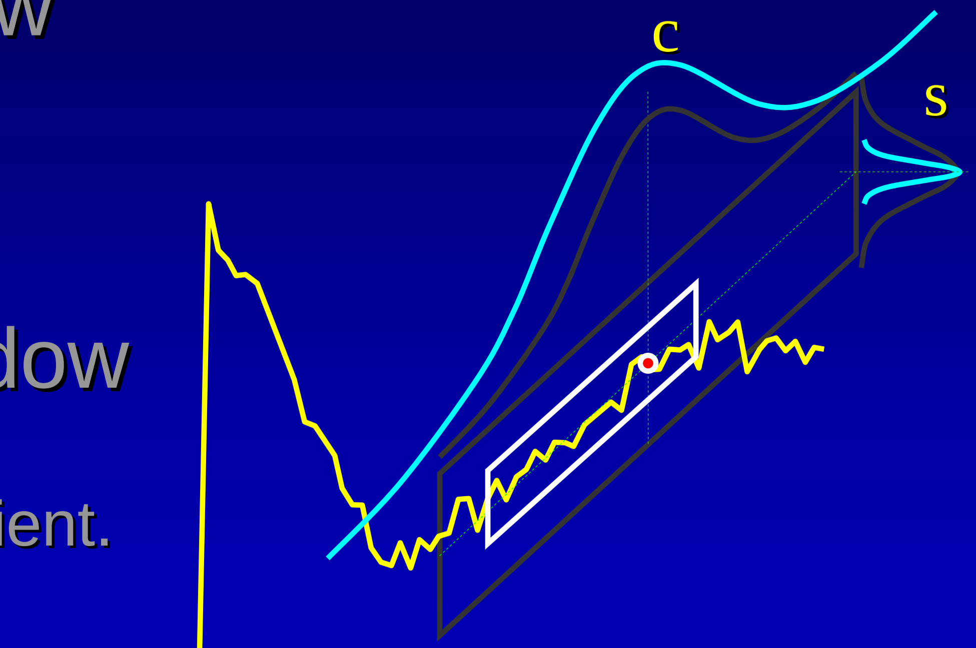
Outline: Bilateral→Trilateral Filter

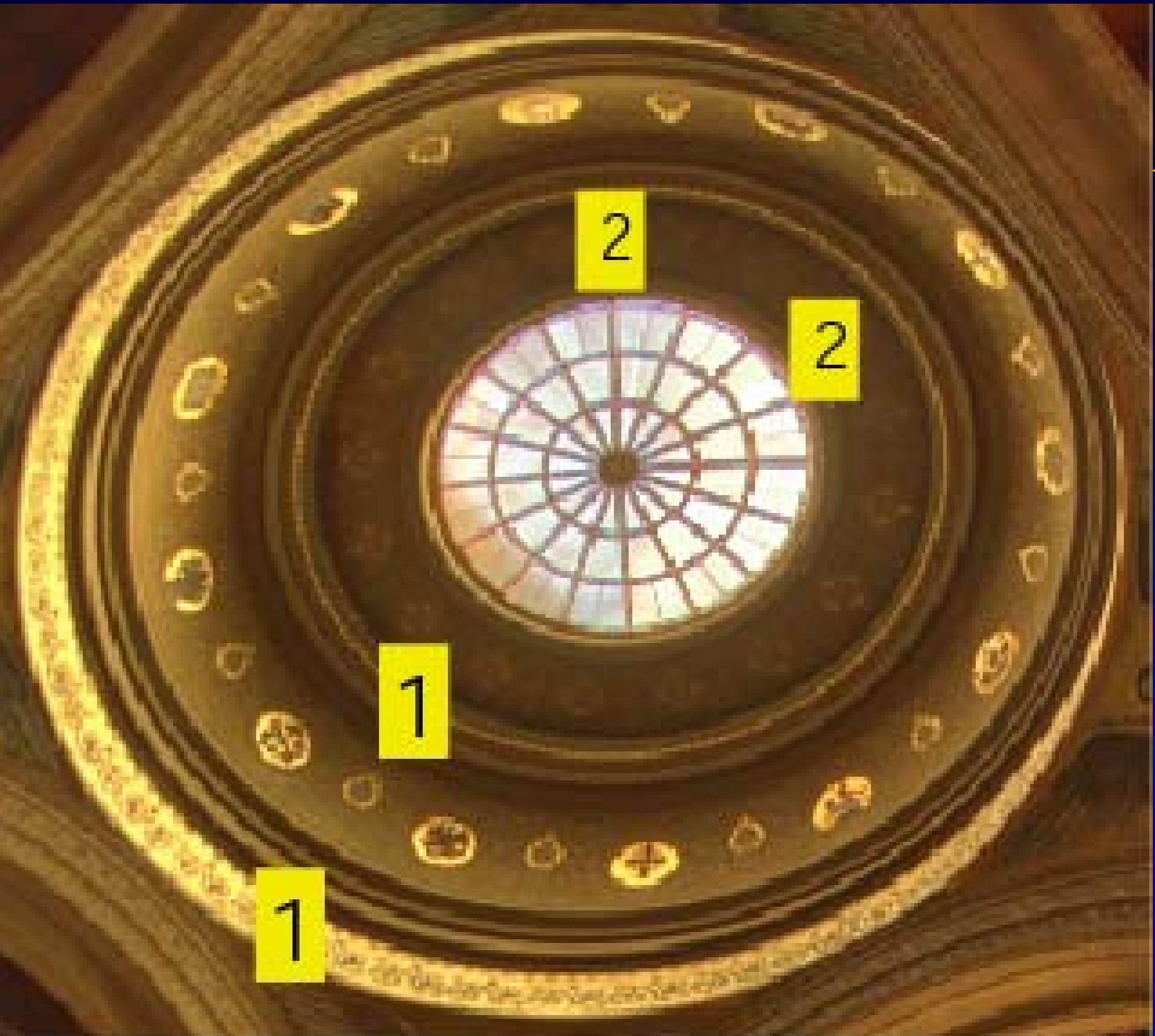
Key Ideas:

- **Tilt** the filter window
  according to bilaterally-
  smoothed gradients

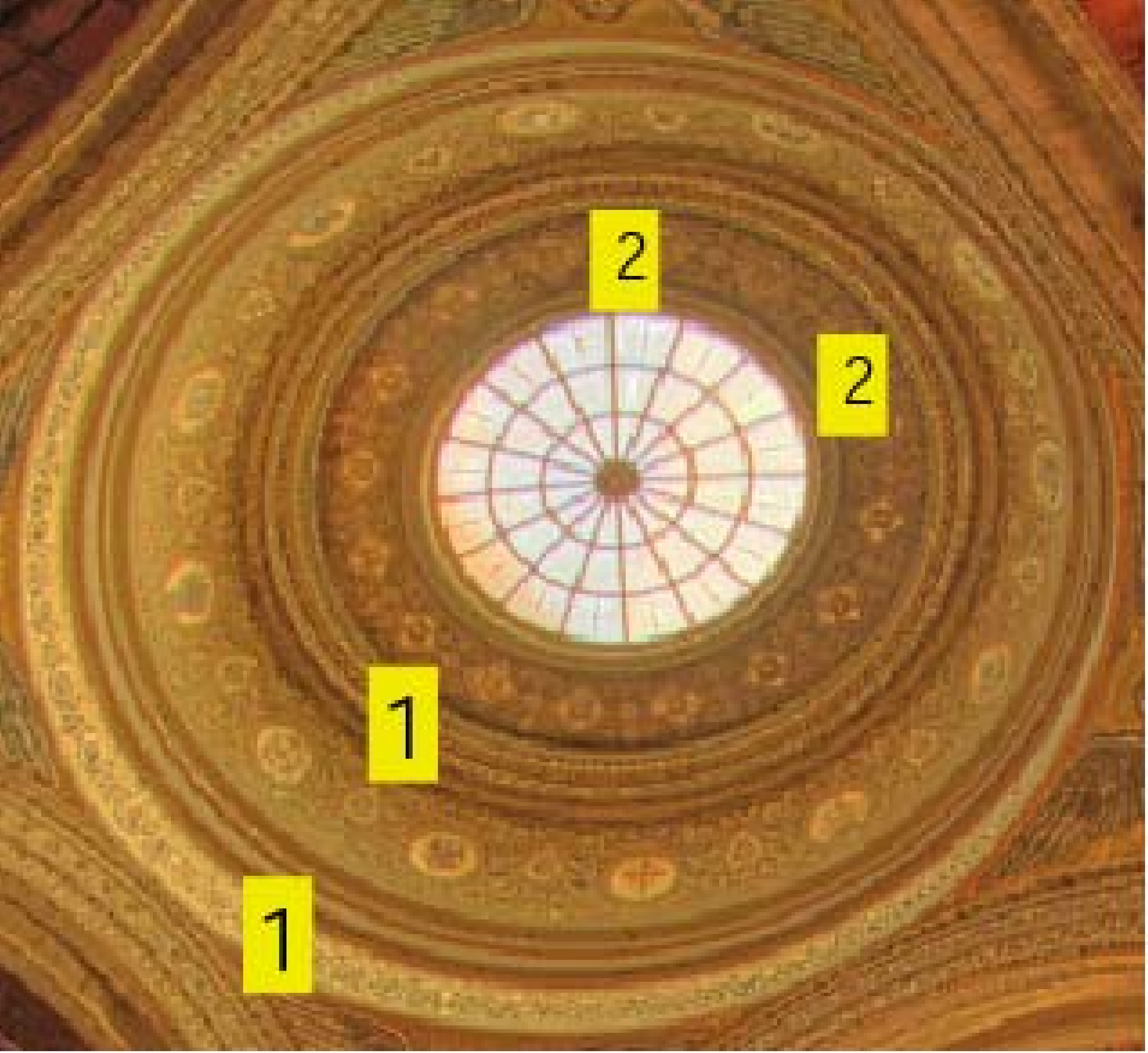
- **Limit** the filter window
  to connected regions
  of similar smoothed gradient.

- **Adjust** Parameters
  from measurements
  of the windowed signal

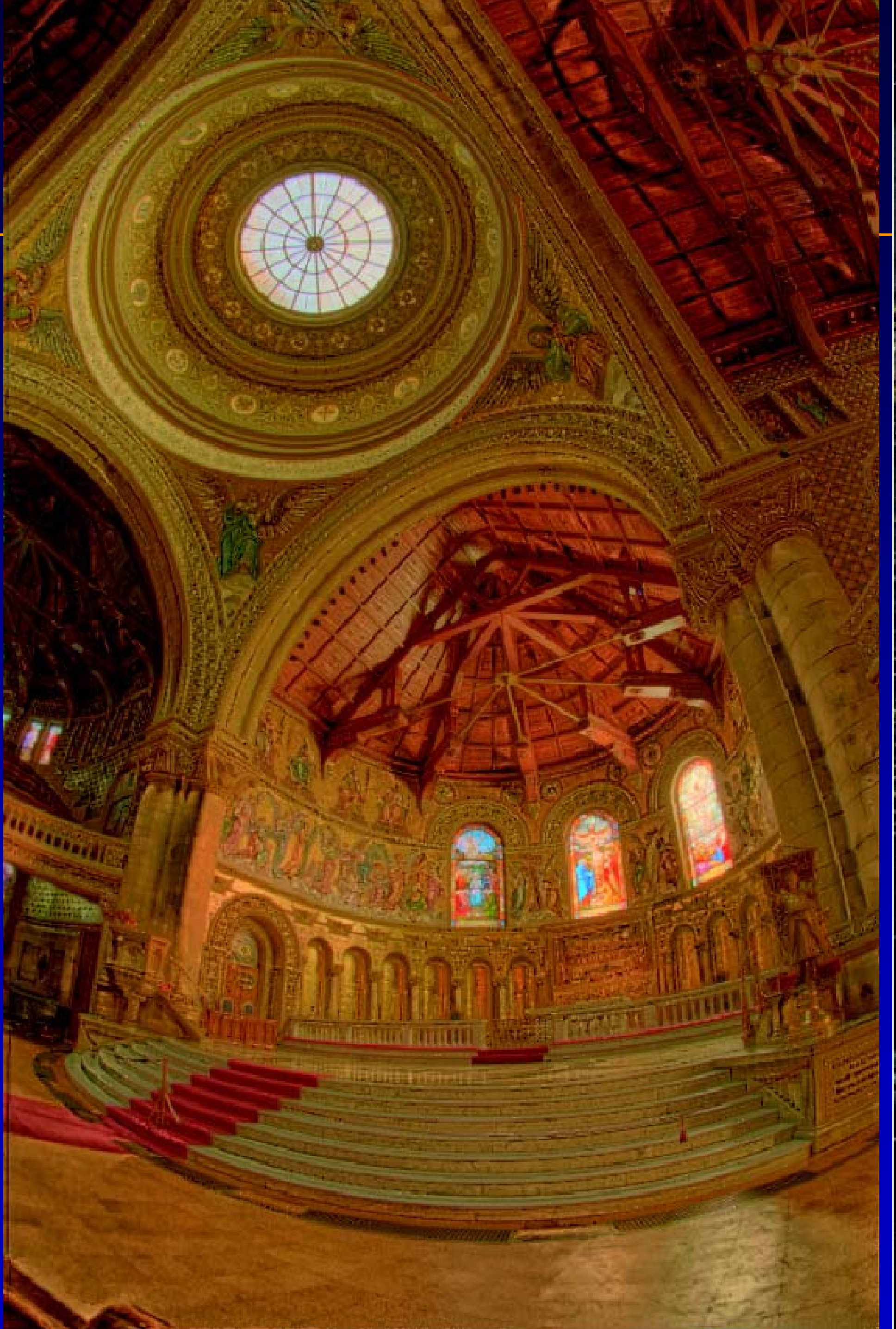




Bilateral

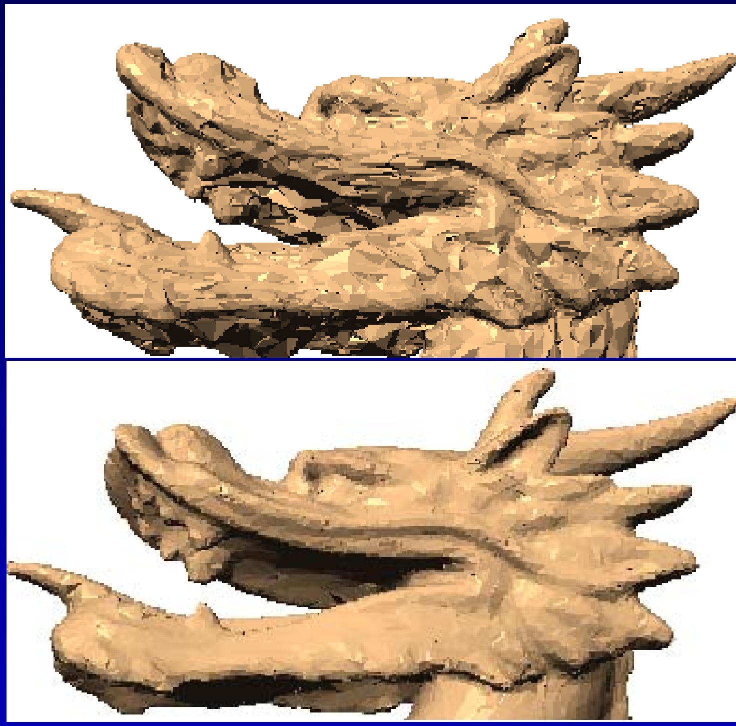


Trilateral



Trilateral Filter (Choudhury 2003)

- Strengths
  - Sharpens corners
  - Smoothes similar gradients
  - Automatic parameter setting
  - 3-D mesh de-noising, too!



- Weaknesses
  - S-L-O-W; very costly connected-region finder
  - Shares Bilateral's 'Lonely Outlier Pixel' artifacts
  - Noise Tolerance limits; disrupts 'tilt' estimates

NEW IDEA : ‘Joint’ or ‘Cross’ Bilateral’ Petschnigg(2004) and Eisemann(2004)

Bilateral → *two kinds* of weights

NEW : get them from *two kinds* of images.

- Smooth image **A** pixels locally, but
- Limit to “similar regions” found in image **B**

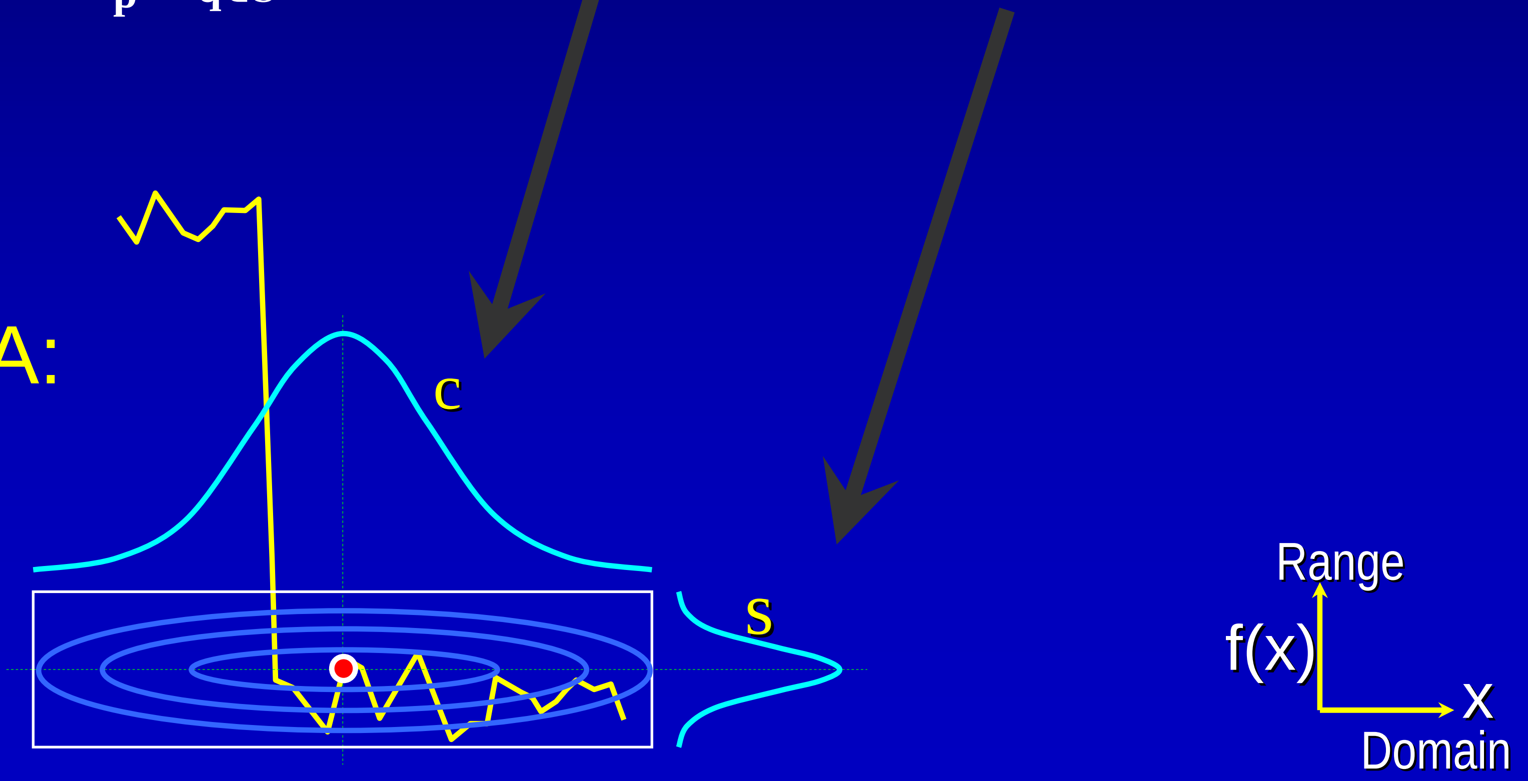
Why do this? To get **‘best of both images’**

Ordinary Bilateral Filter

Bilateral → ***two kinds*** of weights, one image A :

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

Image A:



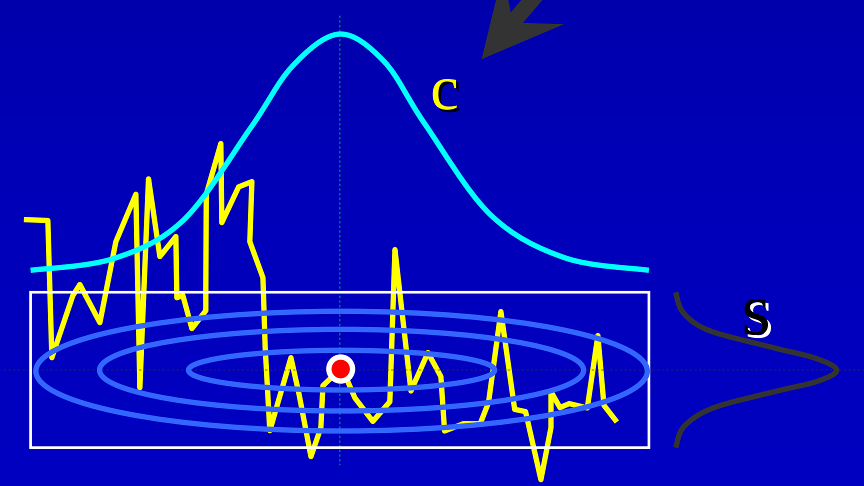
'Joint' or 'Cross' Bilateral Filter

NEW: *two kinds* of weights, *two* images

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

A: Noisy, dim
(ambient image)

B: Clean,strong
(Flash image)



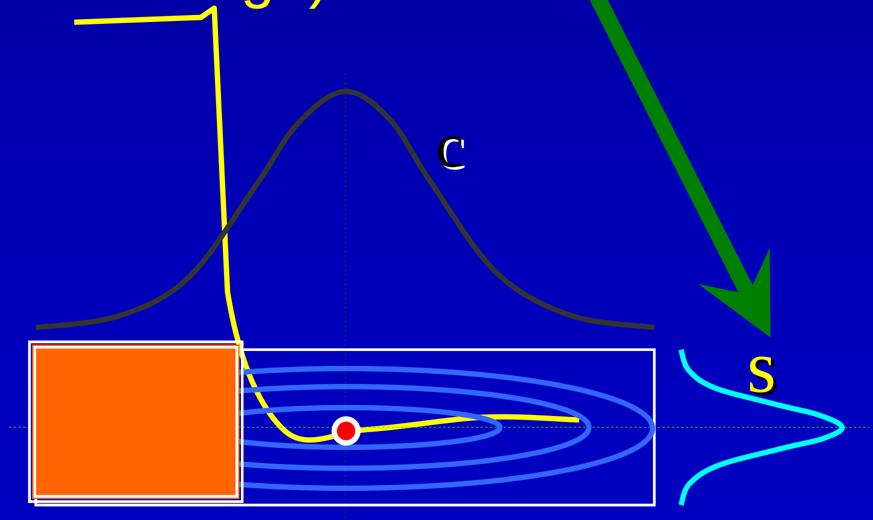


Image A: Warm, shadows, but too Noisy

(too dim for a good quick photo)

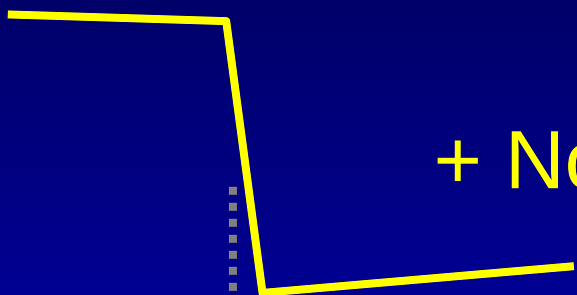
Image B: Cold, Shadow-free, Clean
(flash: simple light, ALMOST no shadows)

MERGE BEST OF BOTH: apply ‘Cross Bilateral’ or ‘Joint Bilateral’

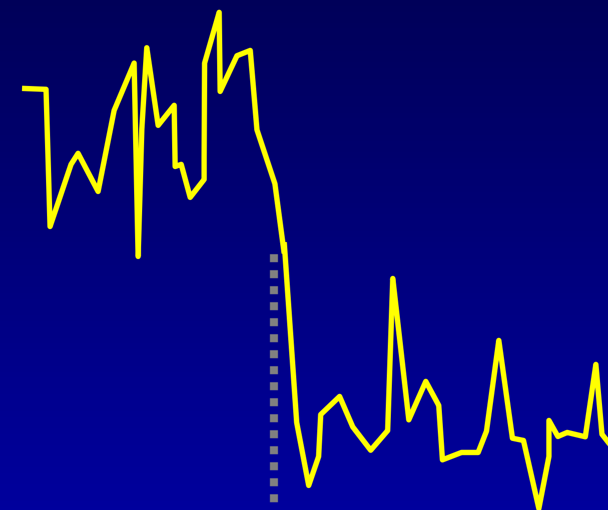
(it really is *much* better!)

Recovers Weak Signals Hidden by Noise

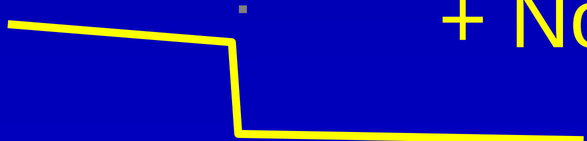
Noisy but Strong…



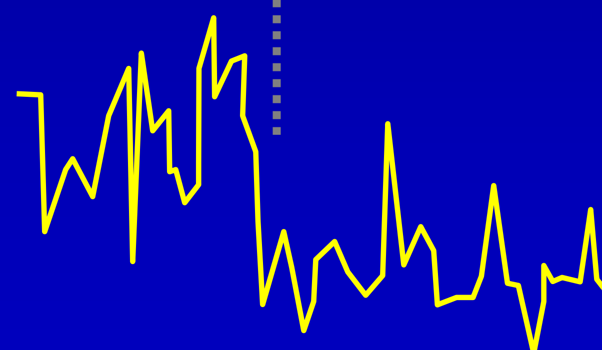
+ Noise =



Noisy and Weak…

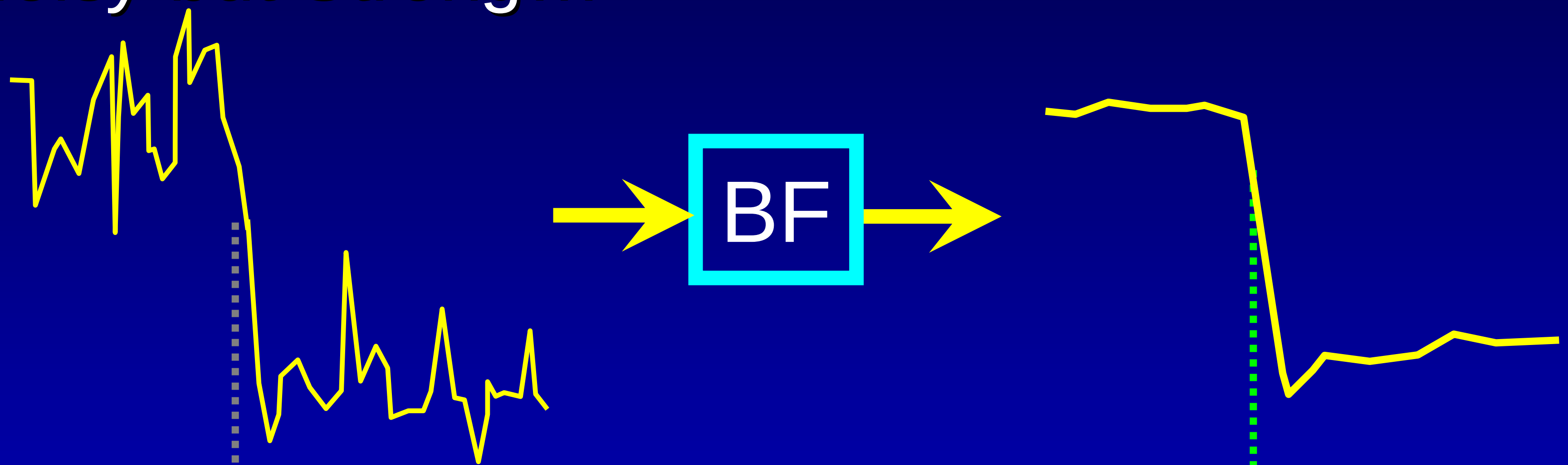


+ Noise =



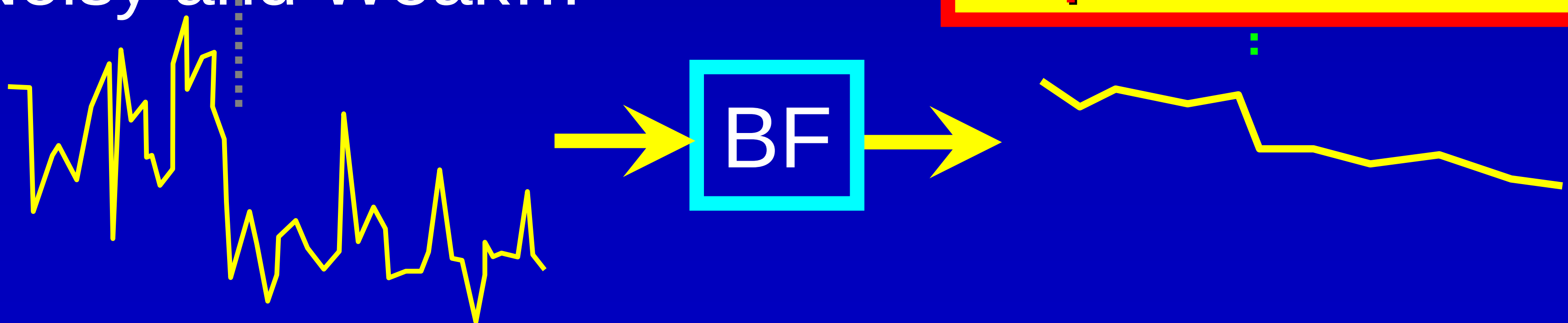
Ordinary Bilateral Filter?

Noisy but Strong…



Step feature GONE!!

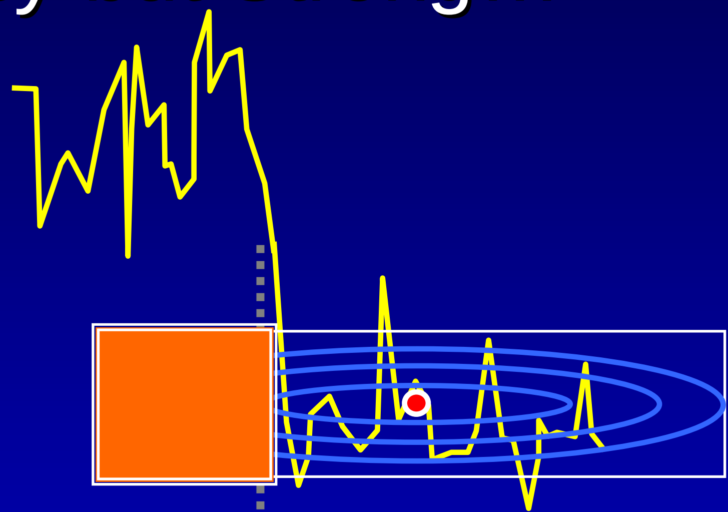
Noisy and Weak…

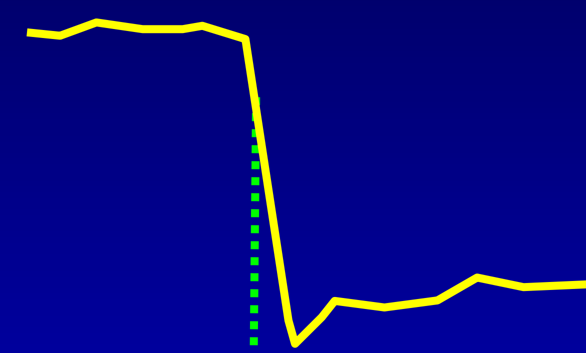


Ordinary Bilateral

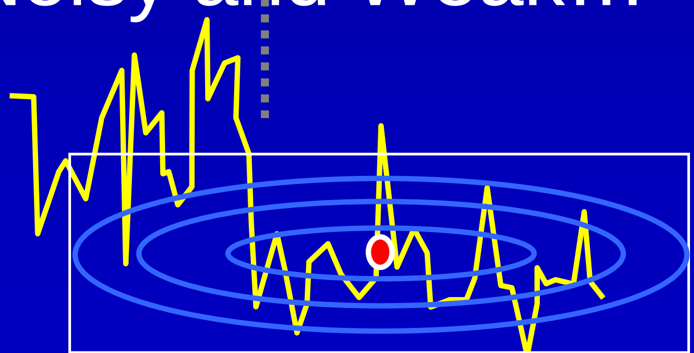
Noisy but Strong…

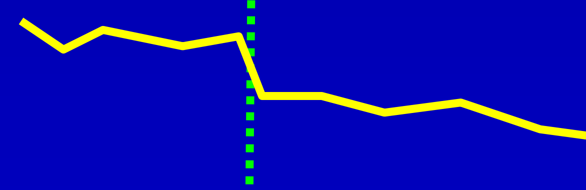
Range filter preserves signal





Noisy and Weak…



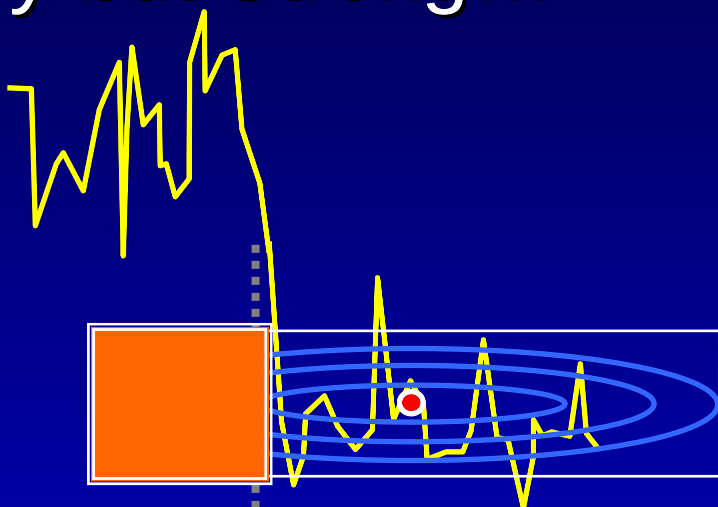


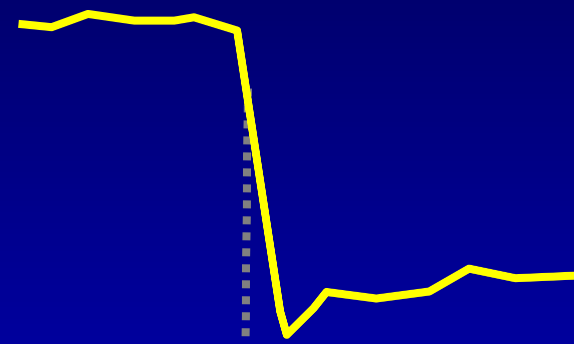
Signal too weak to reject

‘Cross’ or ‘Joint’ Bilateral Idea:

Noisy but Strong…

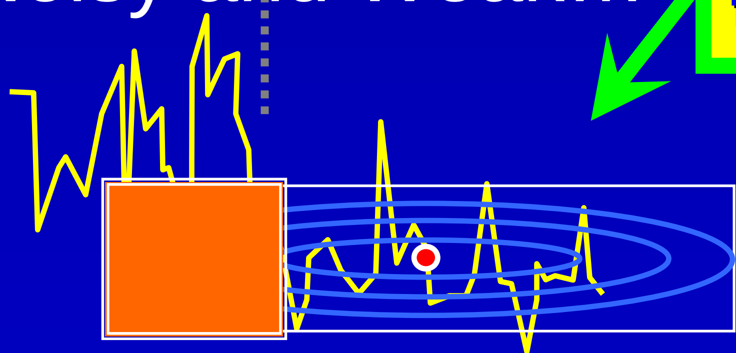
Range filter preserves signal

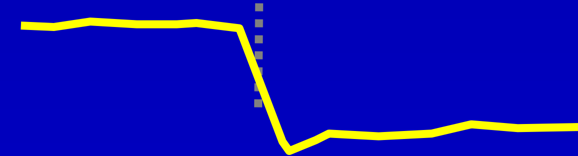




Noisy and Weak…

Use stronger signal’s range to set other’s filter weights…





‘Joint’ or ‘Cross’ Bilateral Filter Petschnigg(2004) and Eisemann(2004)

- CBF(A,B): smoothes image A only; (e.g. the ‘no flash’ image)
- Limits smoothing to stay within regions where Image B is ~uniform (e.g. flash)

- **Useful Residues.** To transfer details,
  - – CBF(A,B) to remove A’s noisy details
  - – CBF(B,A) to extract B’s clean details, and
  - – Add to CBF(A,B) → clean, detailed image!

New Idea:
NL-Means Filter (Buades 2005)

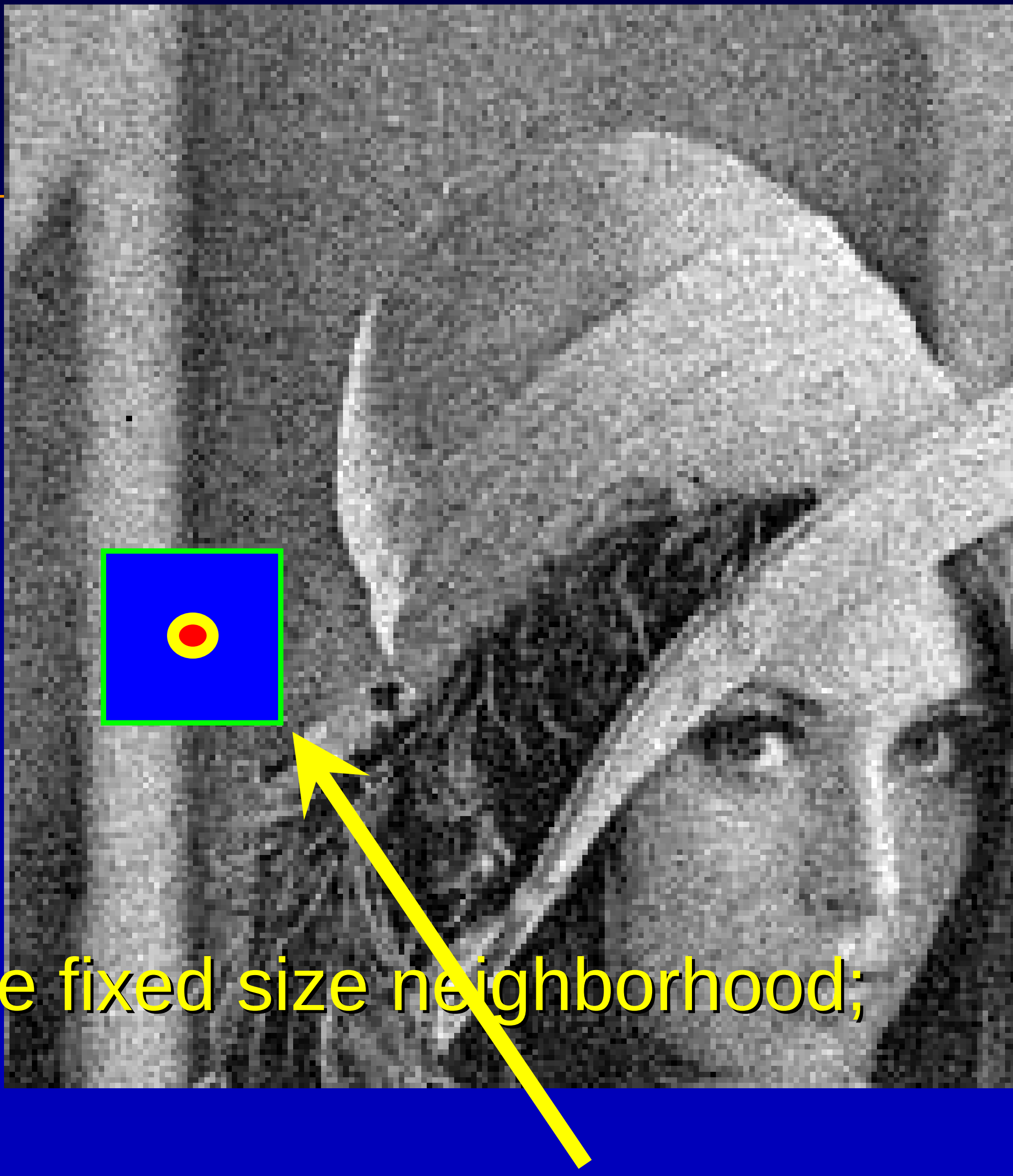
- Same goals: ‘Smooth within Similar Regions’

- **<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:**

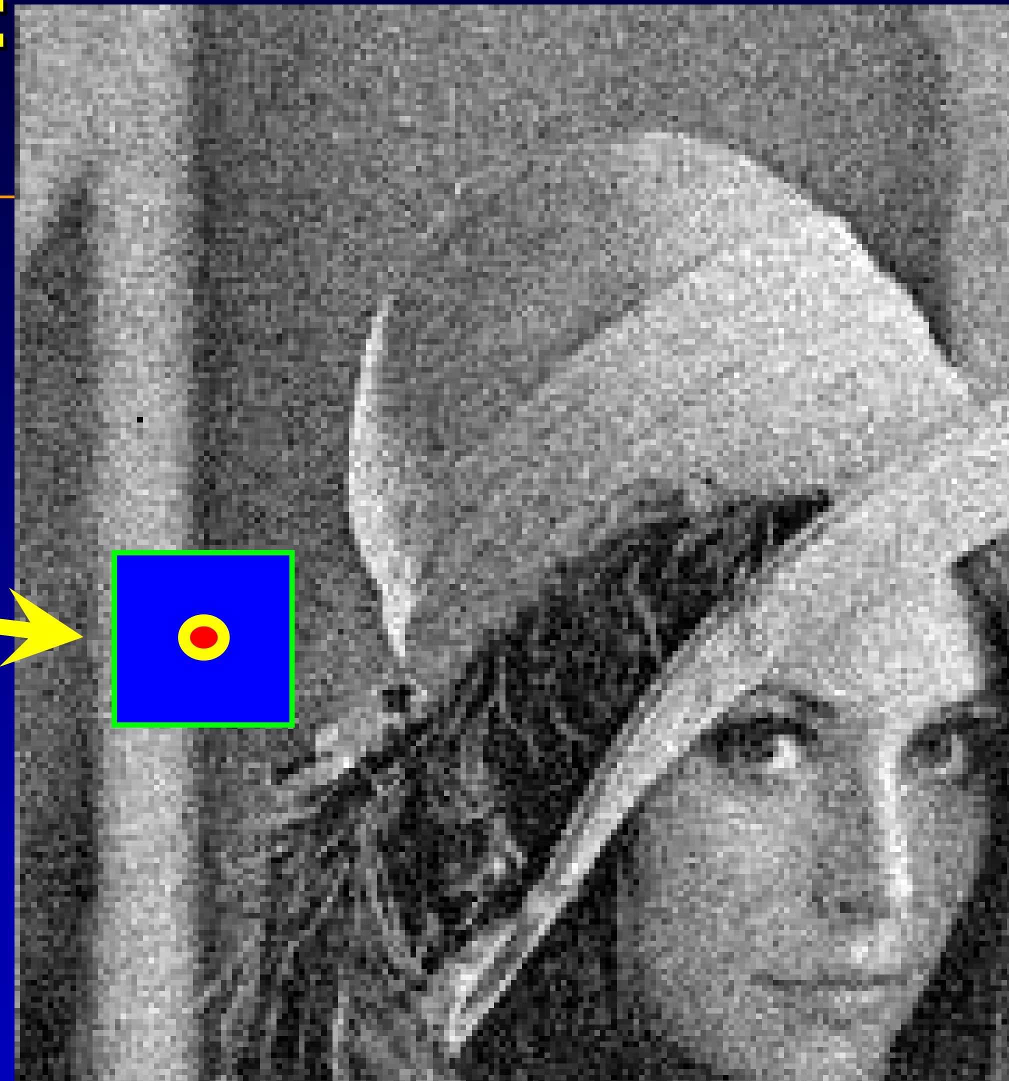
NL-Means Method: Buades (2005)



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

NL-Means Method: Buades (2005)

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



- 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)

'Similar' pixels **p, q**

→ **SMALL**

vector distance;

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

NL-Means Method: Buades (2005)

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

→ **LARGE** vector distance;

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

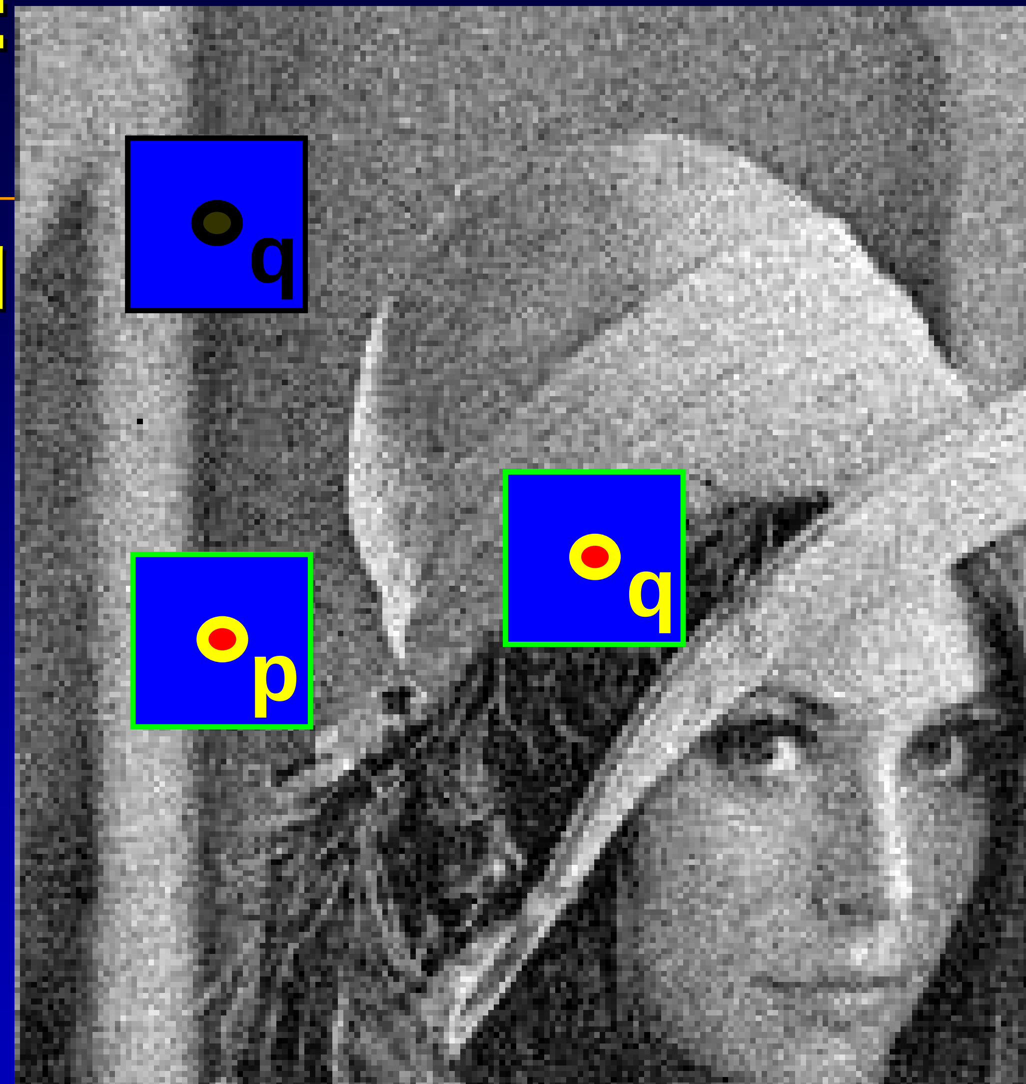
NL-Means Method: Buades (2005)

'Dissimilar' pixels **p, q**

→ **LARGE** vector distance;

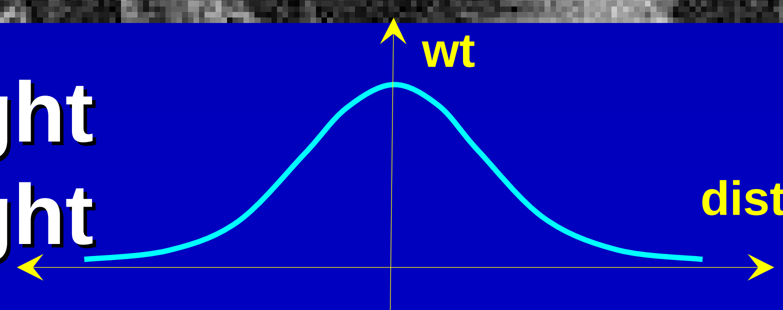
$$\| V_p - V_q \|^2$$

Filter with this!



tiny distance → big weight

big distance → tiny weight



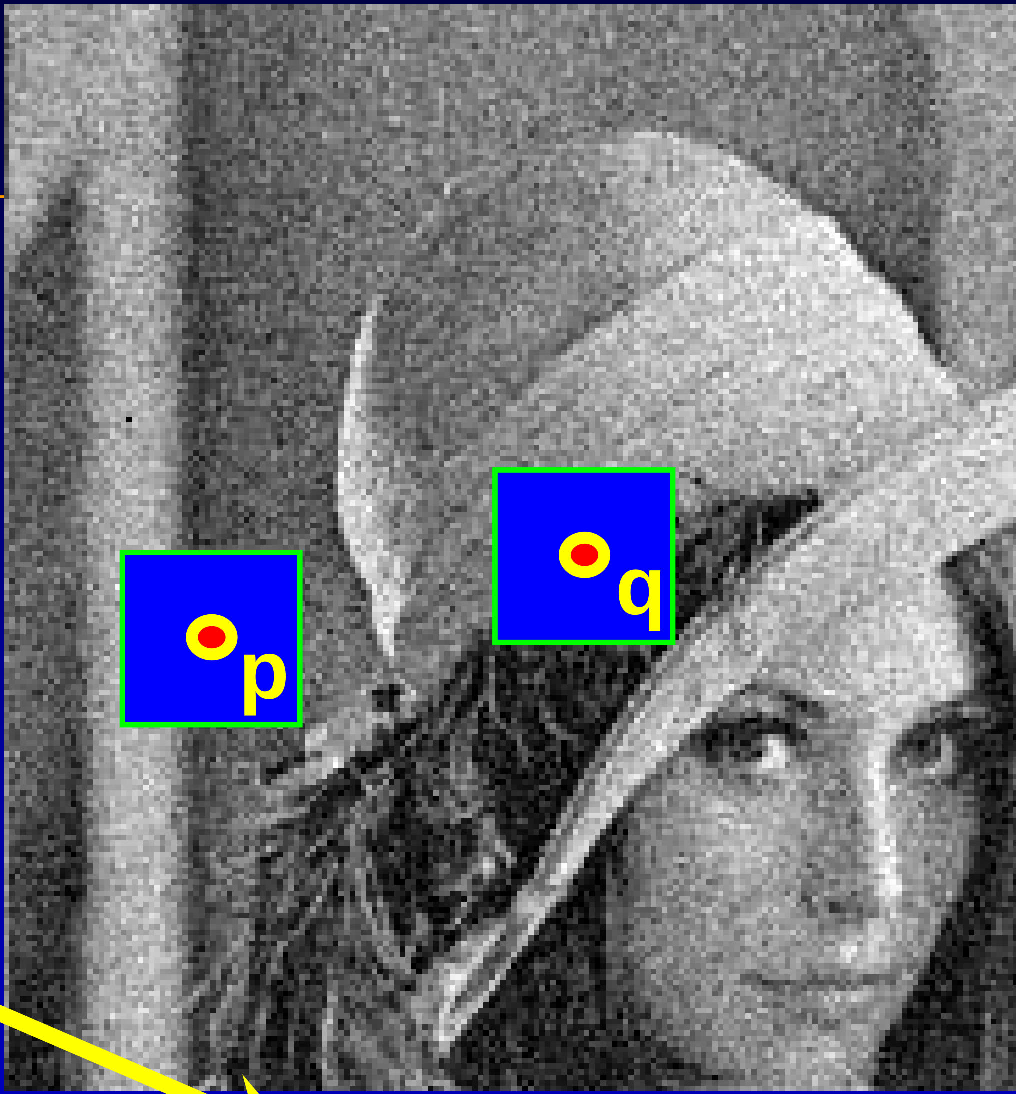
NL-Means Method: Buades (2005)

p, q neighbors define a vector distance;

$$\boxed{\| V_{\mathbf{p}} - V_{\mathbf{q}} \|^2}$$

Filter with this:

No spatial term!

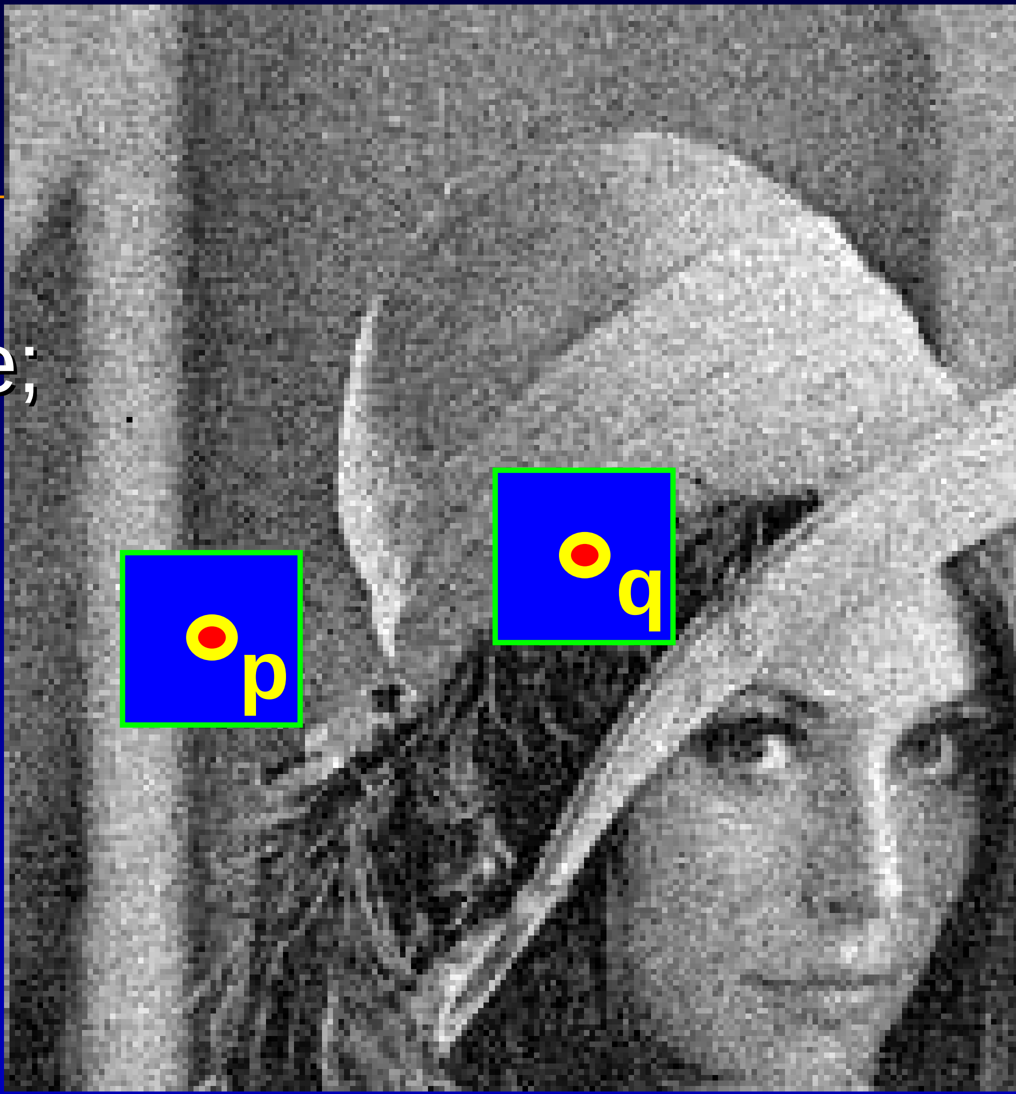


$$NLMF\,[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} \cancel{G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)}\, \boxed{G_{\sigma_r}\left(\| \vec{V}_{\mathbf{p}} - \vec{V}_{\mathbf{q}} \|^2\right)} I_{\mathbf{q}}$$

NL-Means Method: Buades (2005)

pixels **p, q** neighbors

Set a vector distance;



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

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.

Non-Local Similarity (You, 2008)

- Buades NL Means: vector similarity helps, but is only shift-invariant…
- You: expand to rotation & scale invariance; exploit SIFT for similarity finding…

Weighted Least-Squares Optimization

http://www.cs.huji.ac.il/~danix/epd/

- **Improved low-halo detail scales….**

SIGGRAPH2008 “Edge-Preserving Decompositions for Multi-Scale Tone and Detail Manipulation”
Z.Farbman, R. Fattal,ID. Lischinski R. Szeliski

Many More Possibilities: EXPERIMENT!

- Bilateral goals are *subjective*;

  --'Local smoothing within similar regions'
  --'Edge-preserving smoothing'
  --'Separate large structure & fine detail'
  --'Eliminate outliers'
  --'Filter within edges, not across them'

- It's simplicity *invites new & inventive answers.*

SIGGRAPH2008

Variants

- 15 Minutes
- <20 slides