CMP717
Image Processing

Modern Image Smoothing

Erkut Erdem
Hacettepe University
Computer Vision Lab (HUCVL)

Today

- Bilateral filtering
- Non-local means denoising
- LARK filter
- Image smoothing via region covariance (RegCov smoothing)
- Rolling guidance filter

Today

- Bilateral filtering
- Non-local means denoising
- LARK filter
- Image smoothing via region covariance (RegCov smoothing)
- Rolling guidance filter

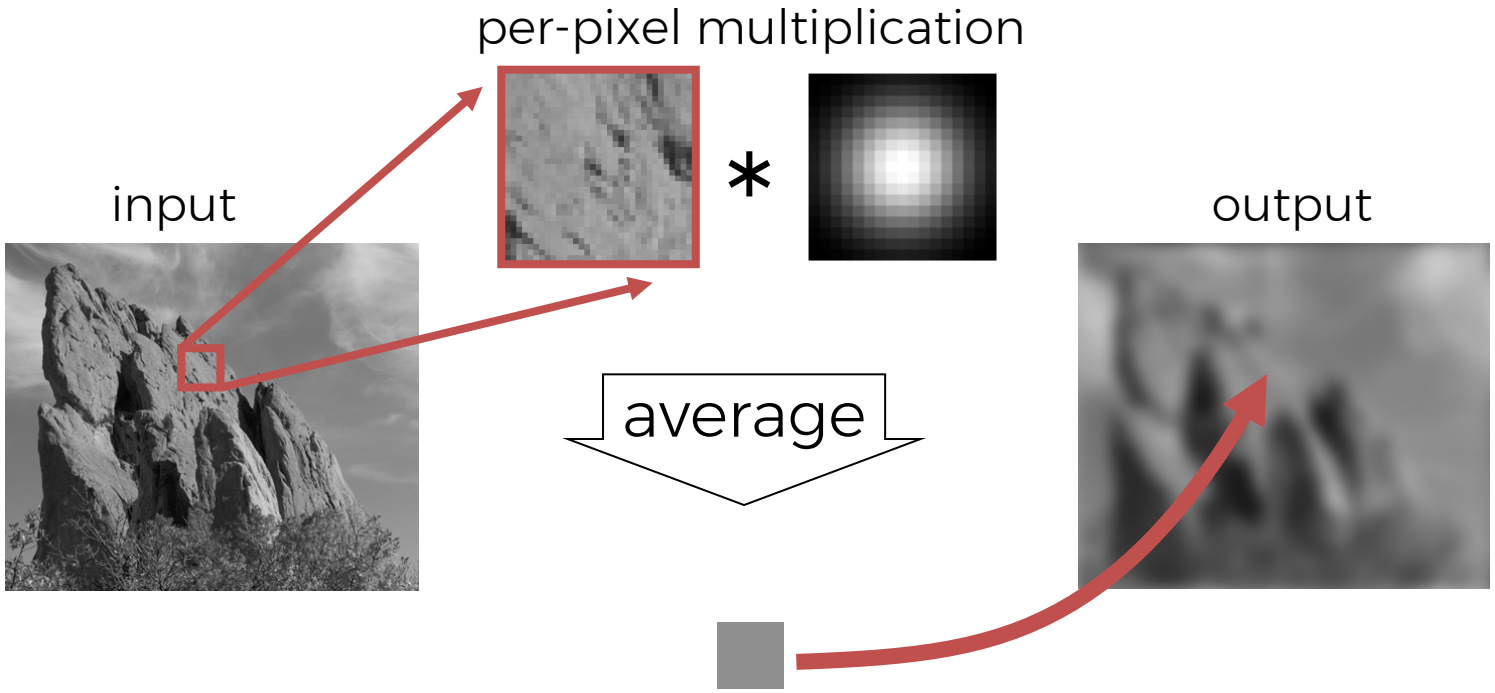
Slides are adapted from the ones prepared by S. Paris, P. Kornprobst, J. Tumblin, and F. Durand.

Strategy for Smoothing Images

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

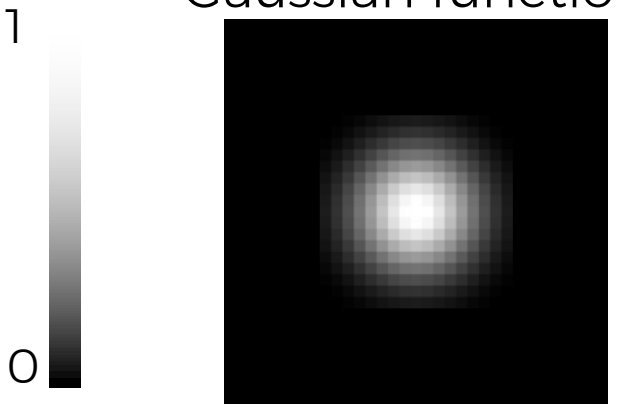
Gaussian Blur

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

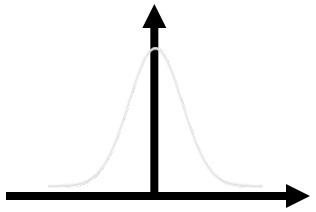


Spatial Parameter

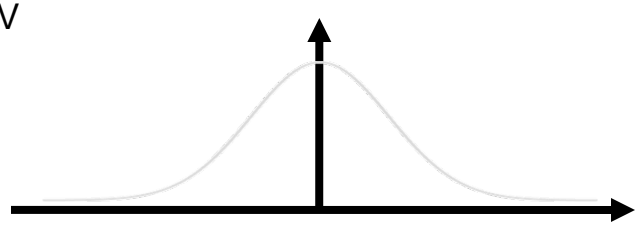
input

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

size of the window



small σ



large σ

limited smoothing

strong smoothing

Properties of Gaussian Blur

- Weights independent of spatial location
  - linear convolution
  - well-known operation
  - efficient computation (recursive algorithm, FFT…)
- Does smooth images
- But smoothes too much:
  edges are blurred.
  - Only spatial distance matters
  - No edge term

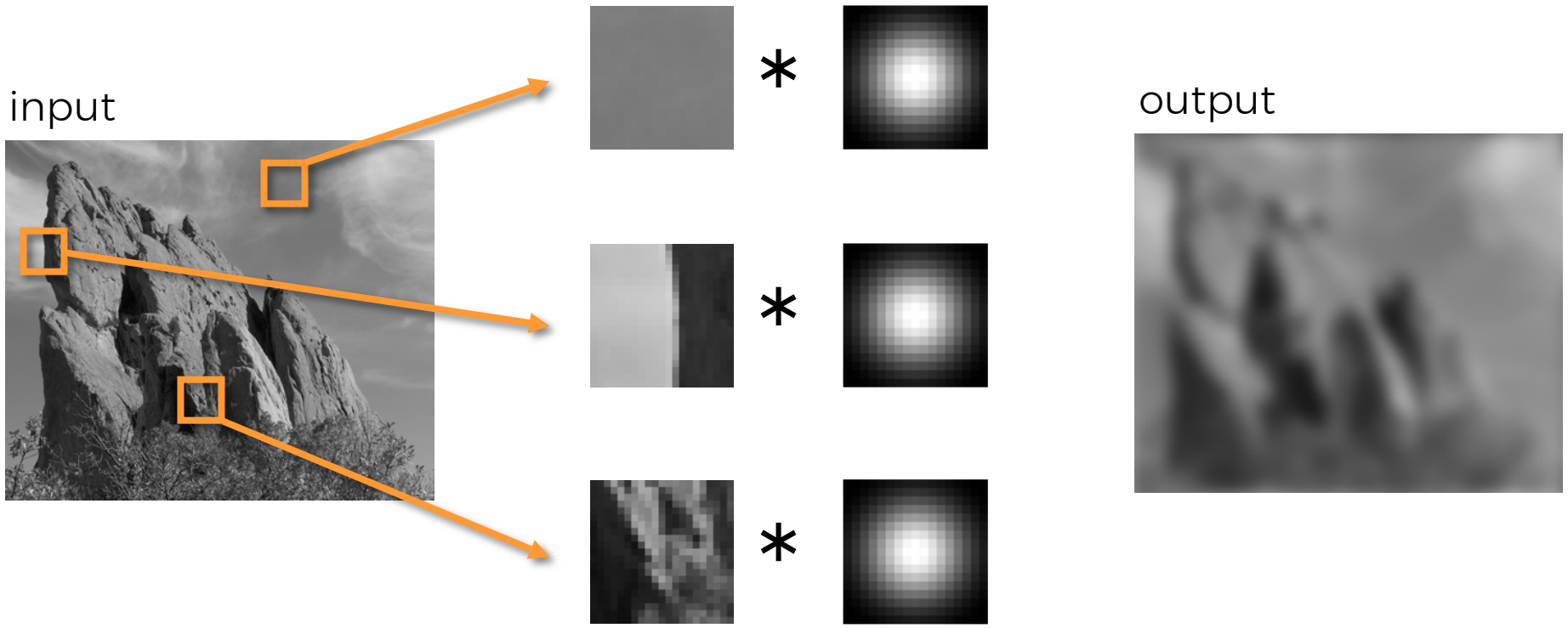
$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q} \in S} G_{\sigma}(\|\mathbf{p}-\mathbf{q}\|) I_{\mathbf{q}}$$

space

input

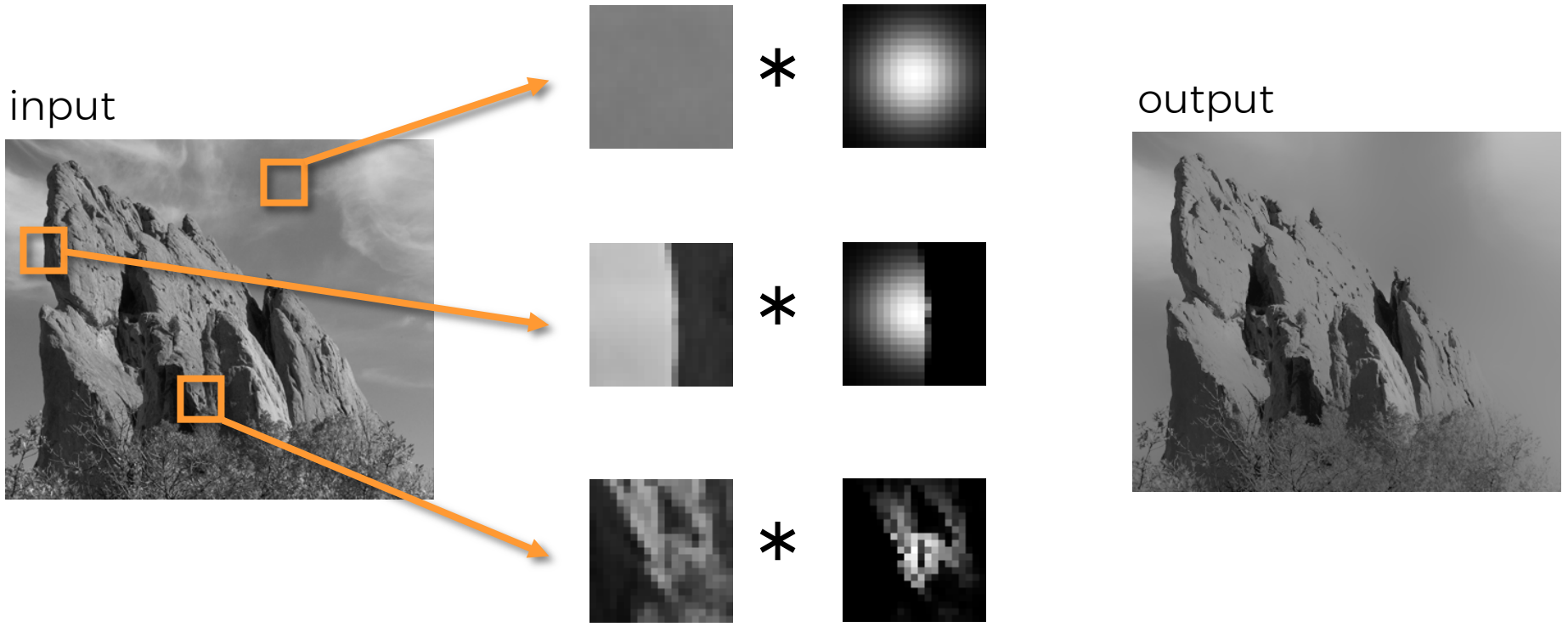
output

Blur Comes from Averaging across Edges



Same Gaussian kernel everywhere.

Bilateral Filter: No Averaging across Edges



The kernel shape depends on the image content.

[Aurich 95, Smith 97, Tomasi 98]

Bilateral Filter: An Additional Edge Term

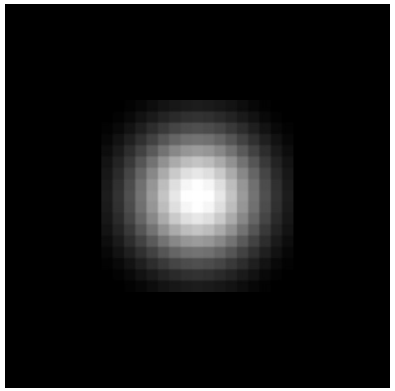
Same idea: weighted average of pixels.

new not new new

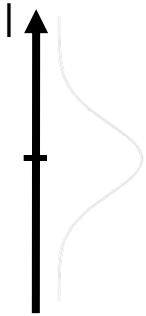
$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)\, I_{\mathbf{q}}$$

normalization factor

space weight



range weight



Space and Range Parameters

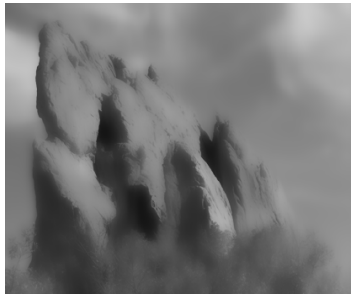
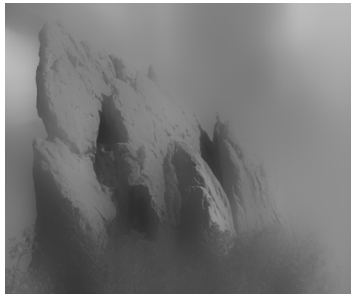
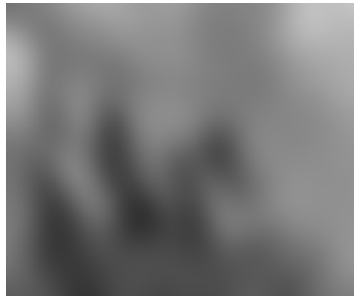
$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}\left(\| \mathbf{p} - \mathbf{q} \|\right) G_{\sigma_r}\left(| I_{\mathbf{p}} - I_{\mathbf{q}} |\right) I_{\mathbf{q}}$$

- space σ_s : spatial extent of the kernel, size of the considered neighborhood.

- range σ_r : “minimum” amplitude of an edge

Exploring the Parameter Space

input

	$\sigma_r = 0.1$	$\sigma_r = 0.25$	$\sigma_r = \infty$ (Gaussian blur)
$\sigma_s = 2$			
$\sigma_s = 6$			
$\sigma_s = 18$			

Bilateral Filtering Color Images

For gray-level images

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

intensity difference

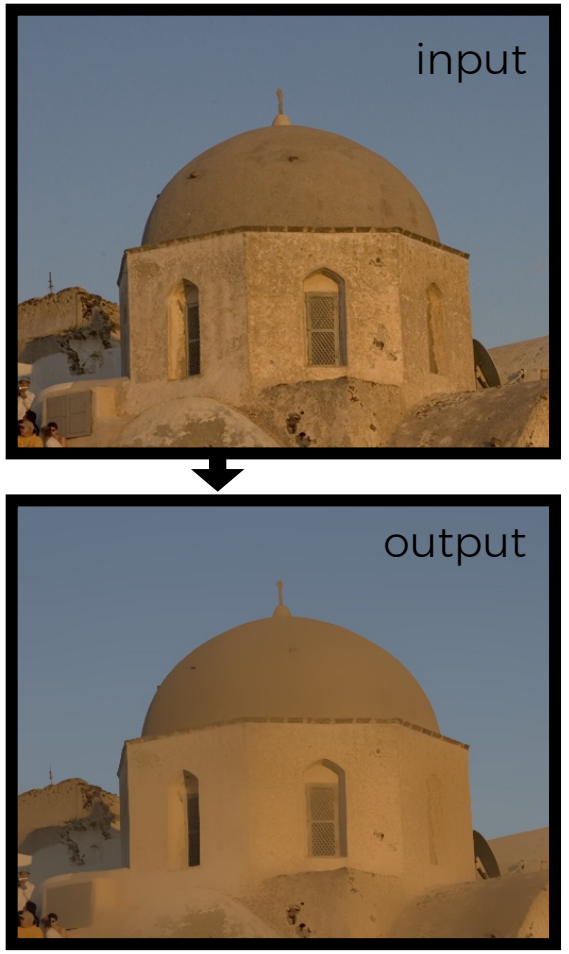
scalar

For color images

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

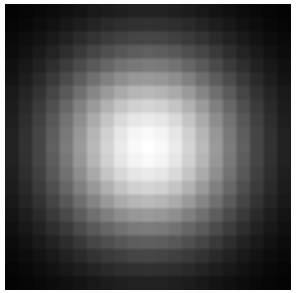
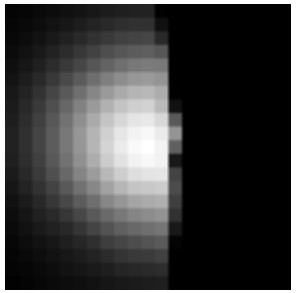
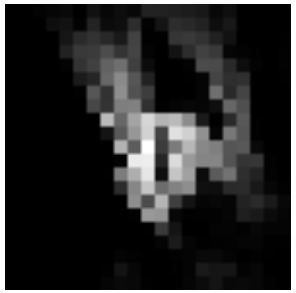
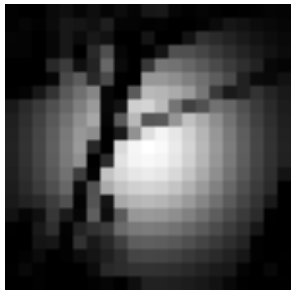
color difference

3D vector
(RGB, Lab)



Hard to Compute

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

- Brute-force implementation is slow > 10min

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

Today

- Bilateral filtering
- Non-local means denoising
- LARK filter
- Image smoothing via region covariance (RegCov smoothing)
- Rolling guidance filter

Slides are adapted from the ones prepared by S. Paris, P. Kornprobst, J. Tumblin, and F. Durand.

New Idea: NL-Means Filter (Buades 2005)

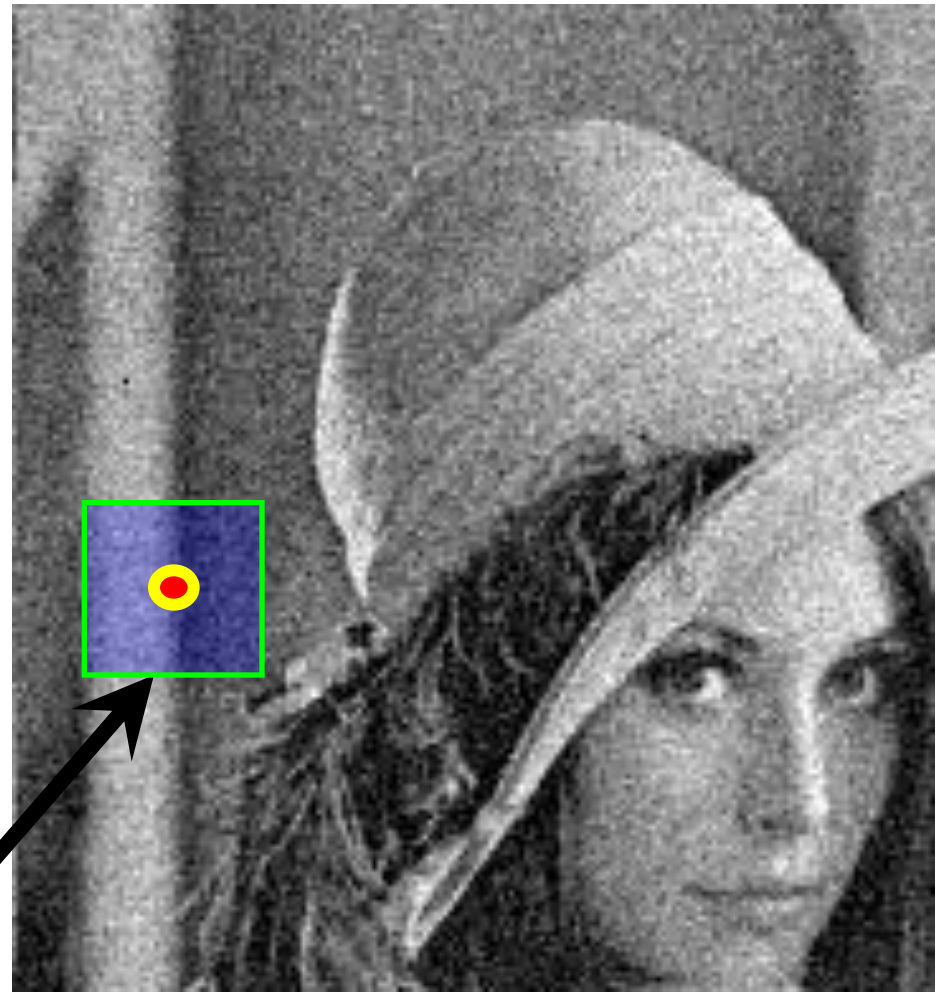
- Same goals: ‘Smooth within Similar Regions’

- KEY INSIGHT: Generalize, extend ‘Similarity’
  - Bilateral:
    Averages neighbors with similar intensities;
  - NL-Means:
    Averages neighbors with similar neighborhoods!

NL-Means Method

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

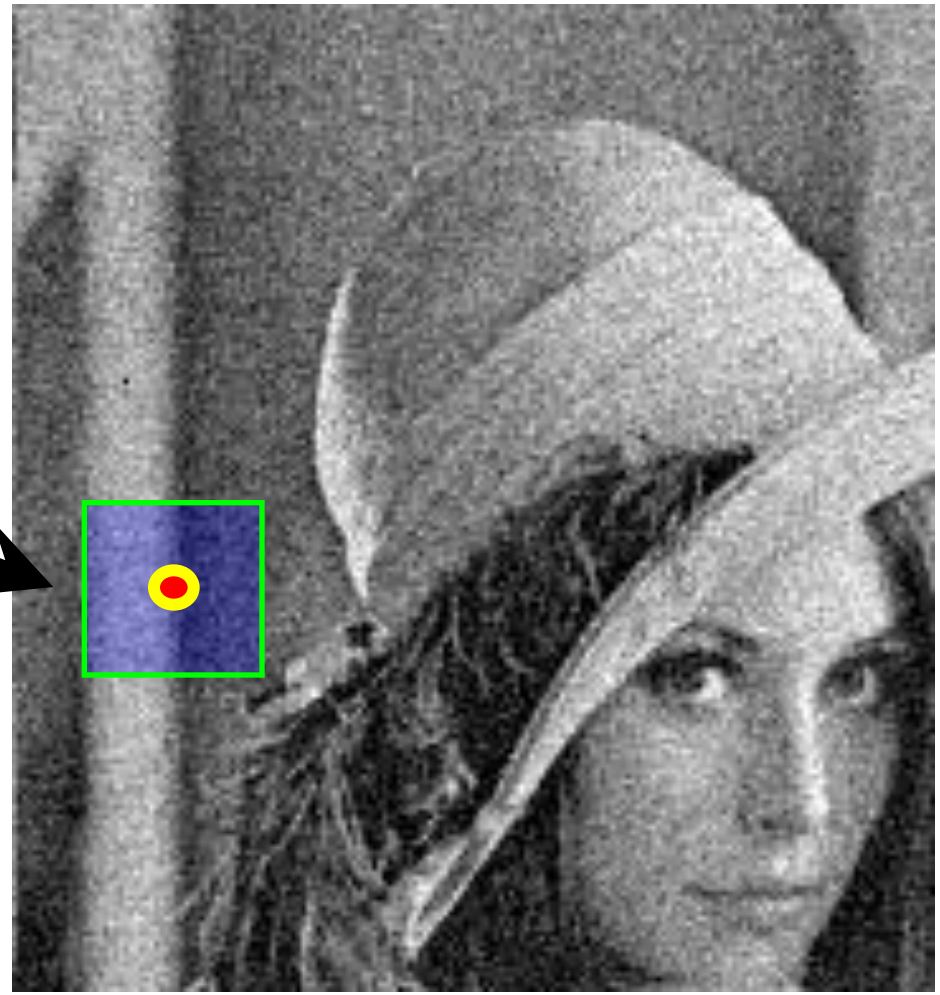
NL-Means Method



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

NL-Means Method

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



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

NL-Means Method

‘Similar’ pixels **p, q**

→ **SMALL**

vector distance;

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

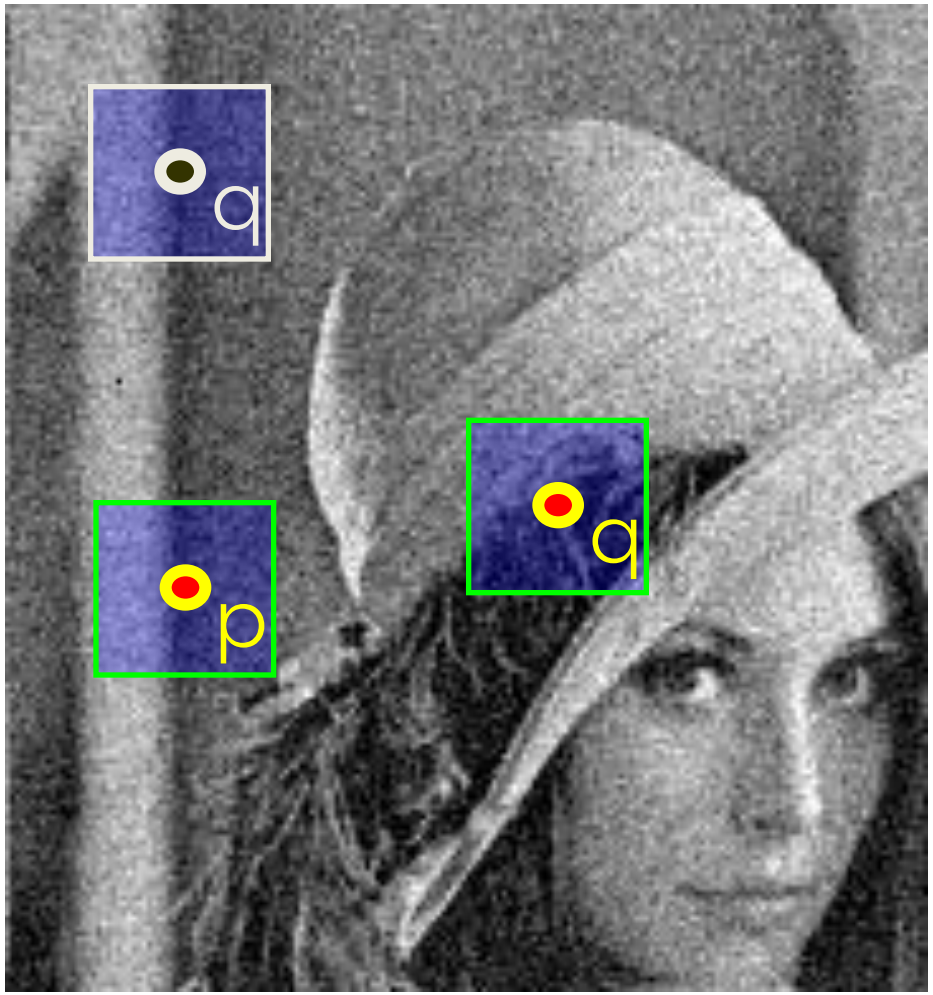
NL-Means Method

‘Dissimilar’ pixels **p, q**

→ **LARGE**

vector distance;

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



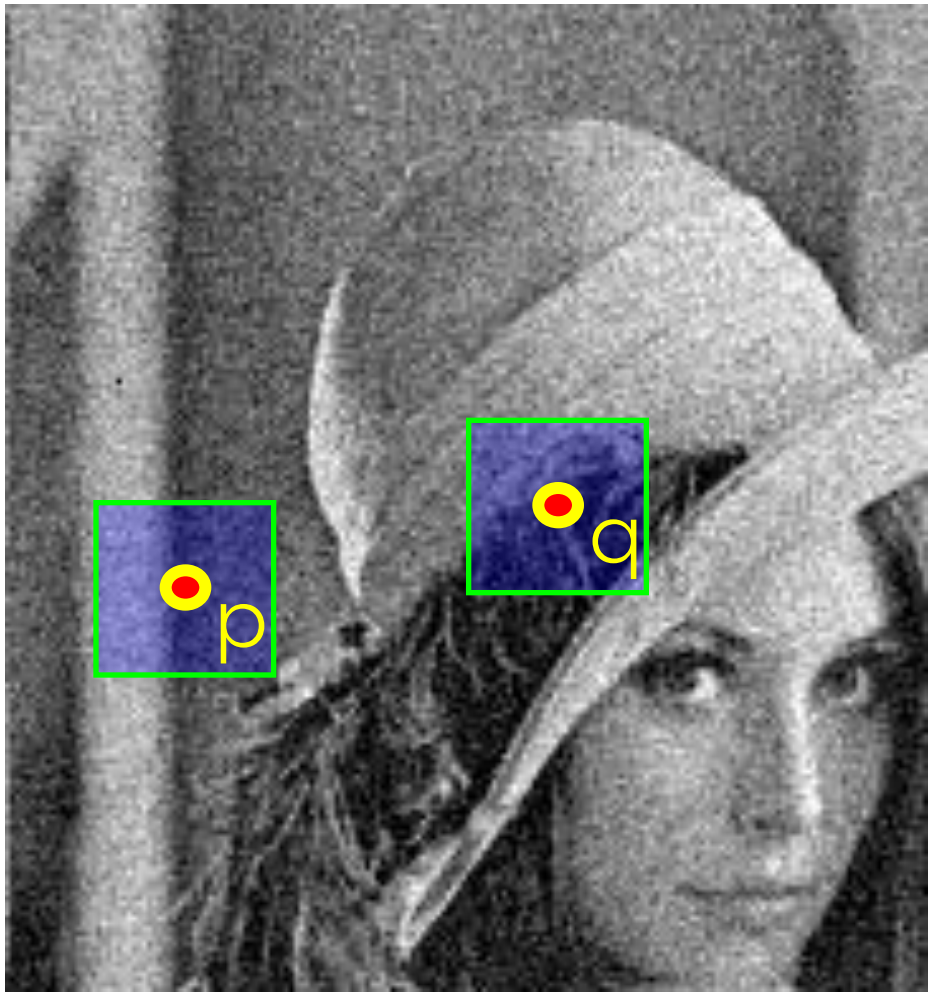
NL-Means Method

‘Dissimilar’ pixels **p, q**

→ **LARGE**
vector distance;

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

Filter with this!



NL-Means Method

p, q neighbors define a vector distance;

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

Filter with this:

No spatial term!

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

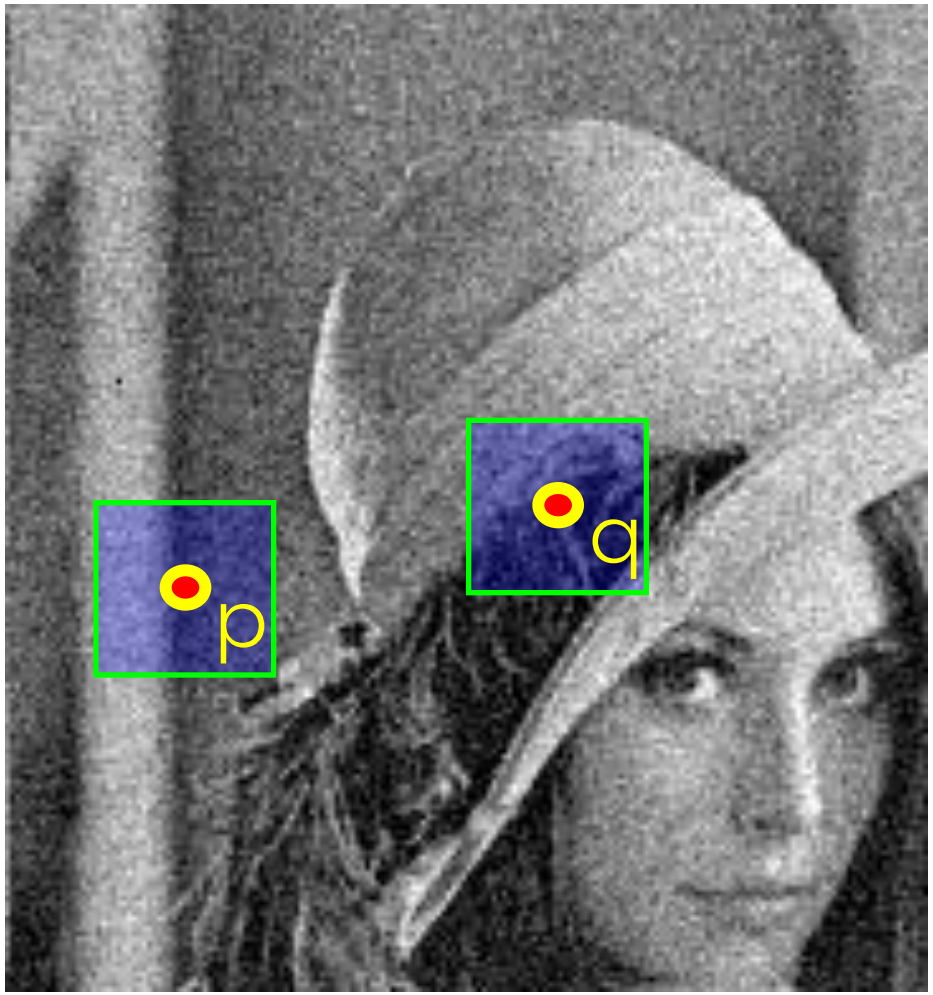
NL-Means Method

pixels **p, q** neighbors

Set a vector distance;

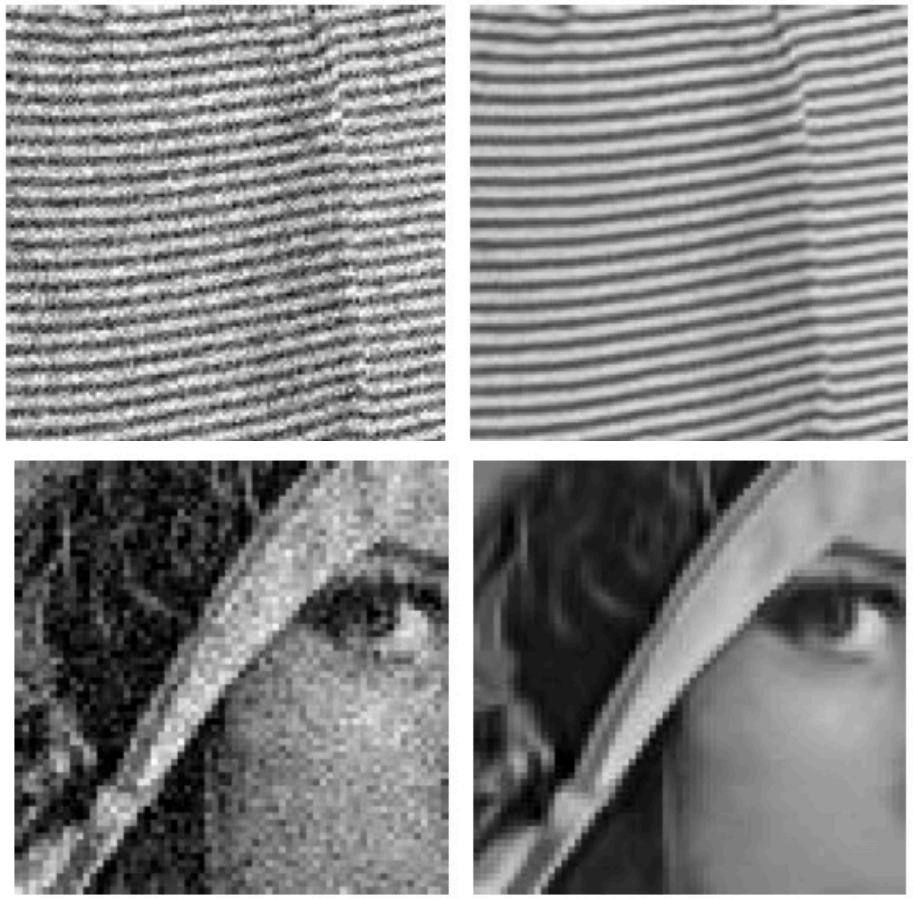
$$\| V_p - V_q \|^2$$

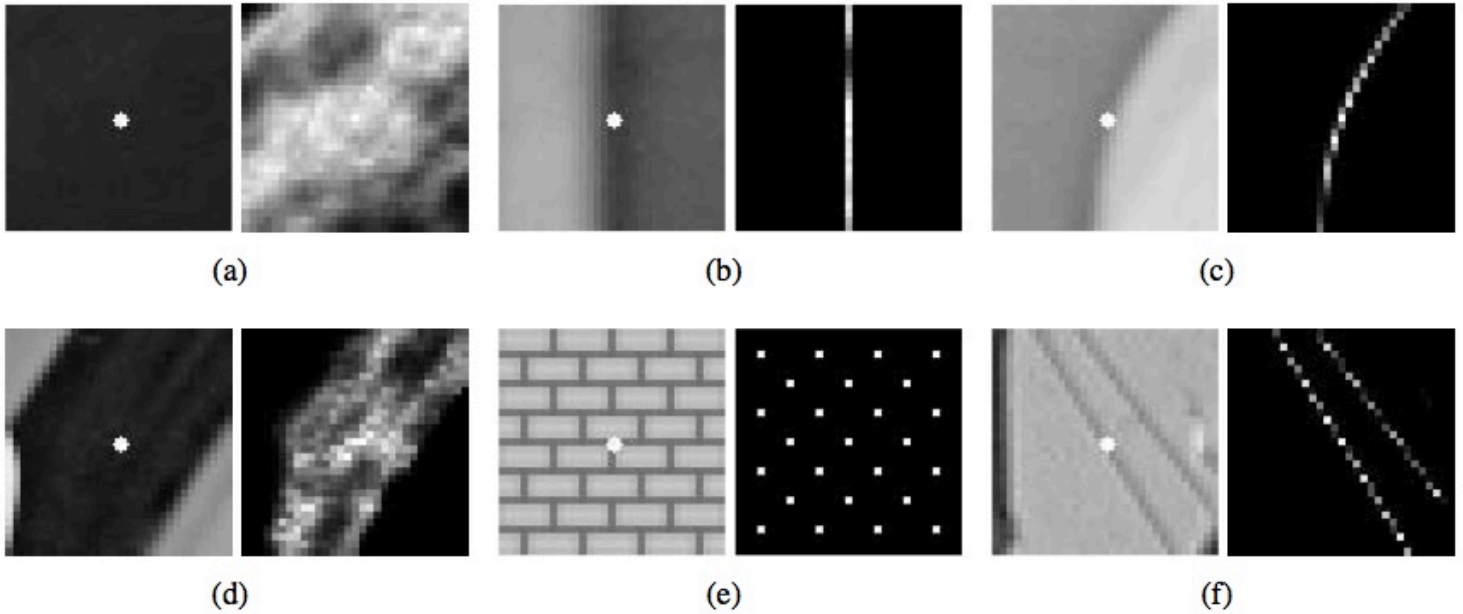
Vector Distance to p sets weight for each pixel q



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

NL-Means Method





NL-Means Method

- Noisy source image:

NL-Means Method

- Gaussian Filter

Low noise,
Low detail

NL-Means Method

- Anisotropic Diffusion

Note ‘stairsteps’:
~ piecewise constant

NL-Means Method

- Bilateral Filter

Better, but similar ‘stairsteps’:

NL-Means Method

- NL-Means:

Sharp,
Low noise,
Few artifacts.

NL-Means Method

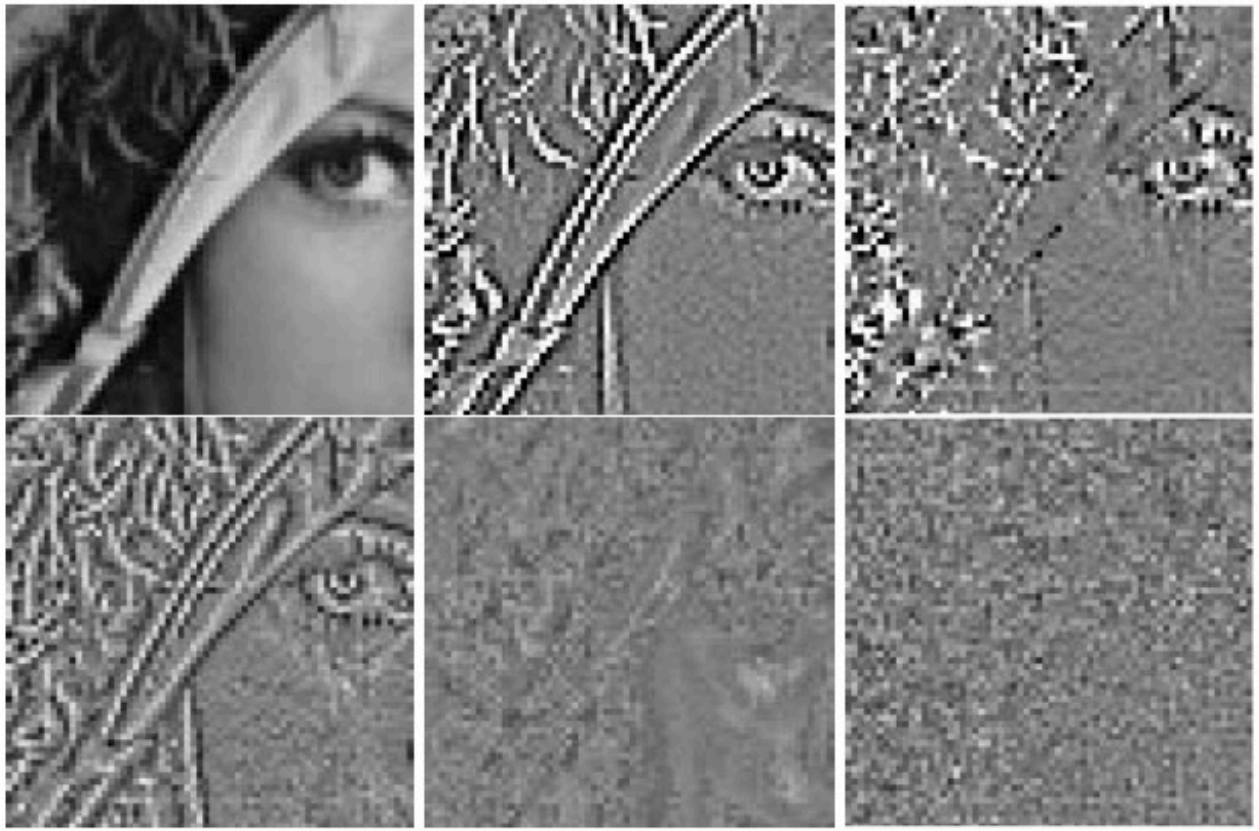


Figure 4. Method noise experience on a natural image. Displaying of the image difference $u - D_h(u)$. From left to right and from top to bottom: original image, Gauss filtering, anisotropic filtering, Total variation minimization, Neighborhood filtering and NL-means algorithm. The visual experiments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

NL-Means Method

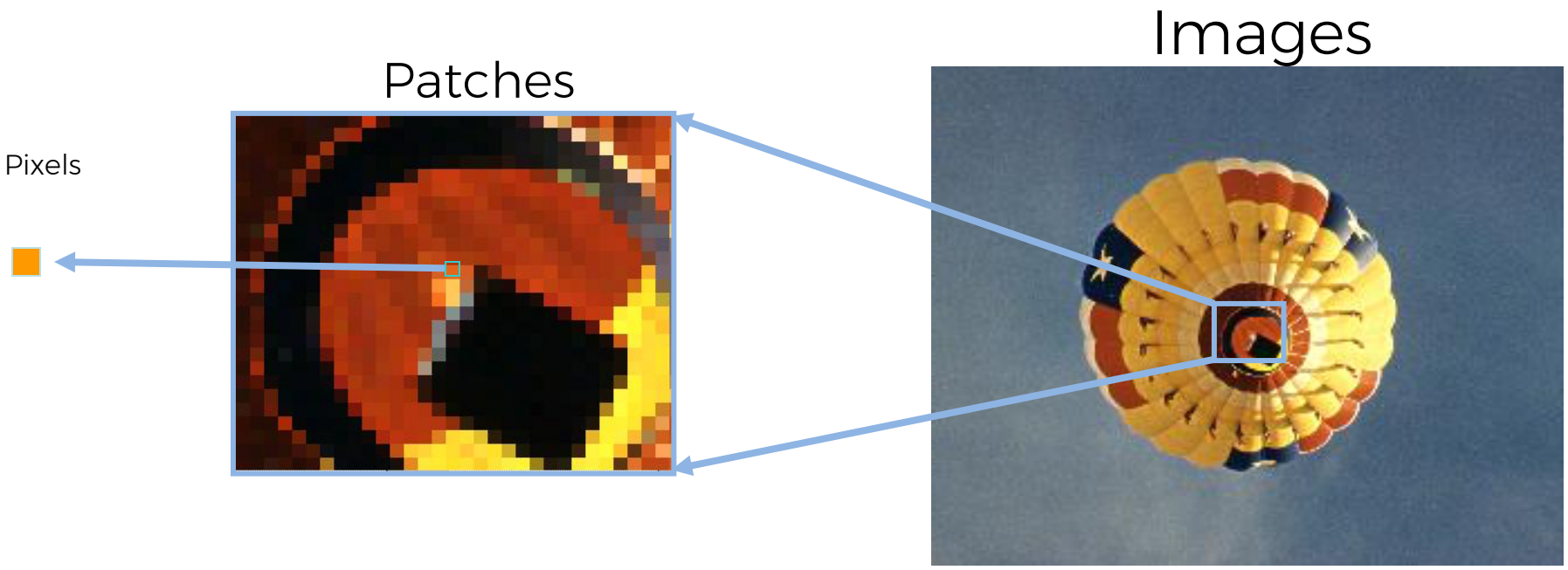
http://www.ipol.im/pub/algo/bcm_non_local_means_denoising/

Today

- Bilateral filtering
- Non-local means denoising
- LARK filter
- Image smoothing via region covariance (RegCov smoothing)
- Rolling guidance filter

Slides are adapted from the ones prepared by P. Milanfar.

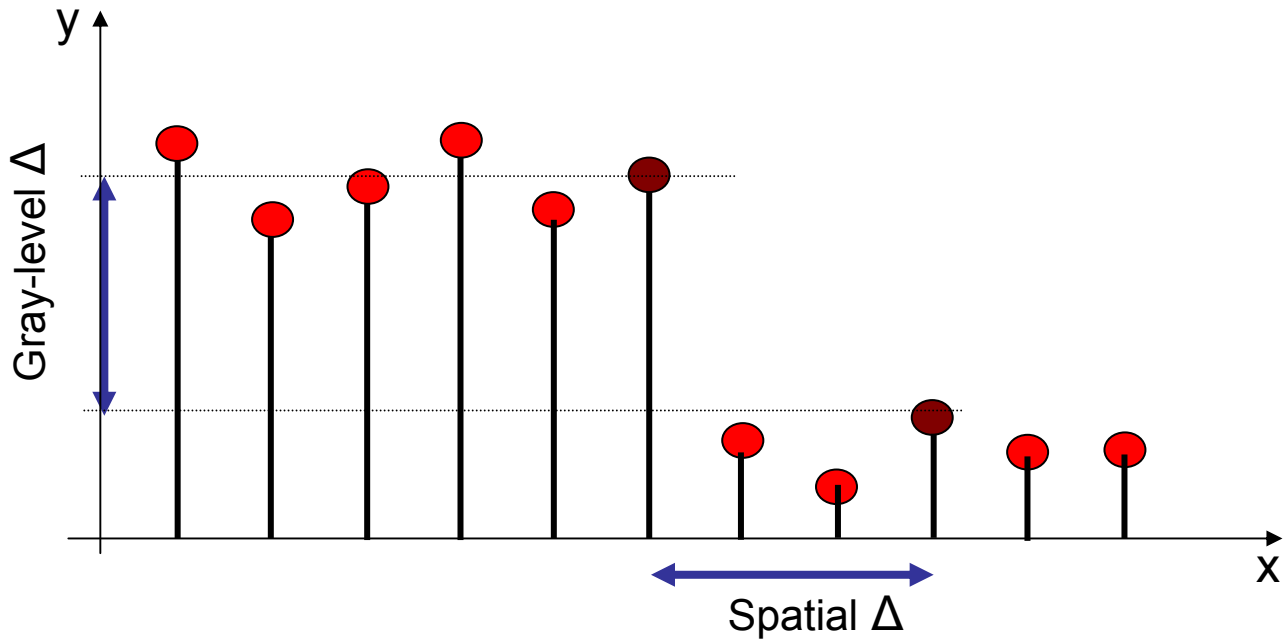
From pixels to patches and to images



Similarities can be defined at different scales..

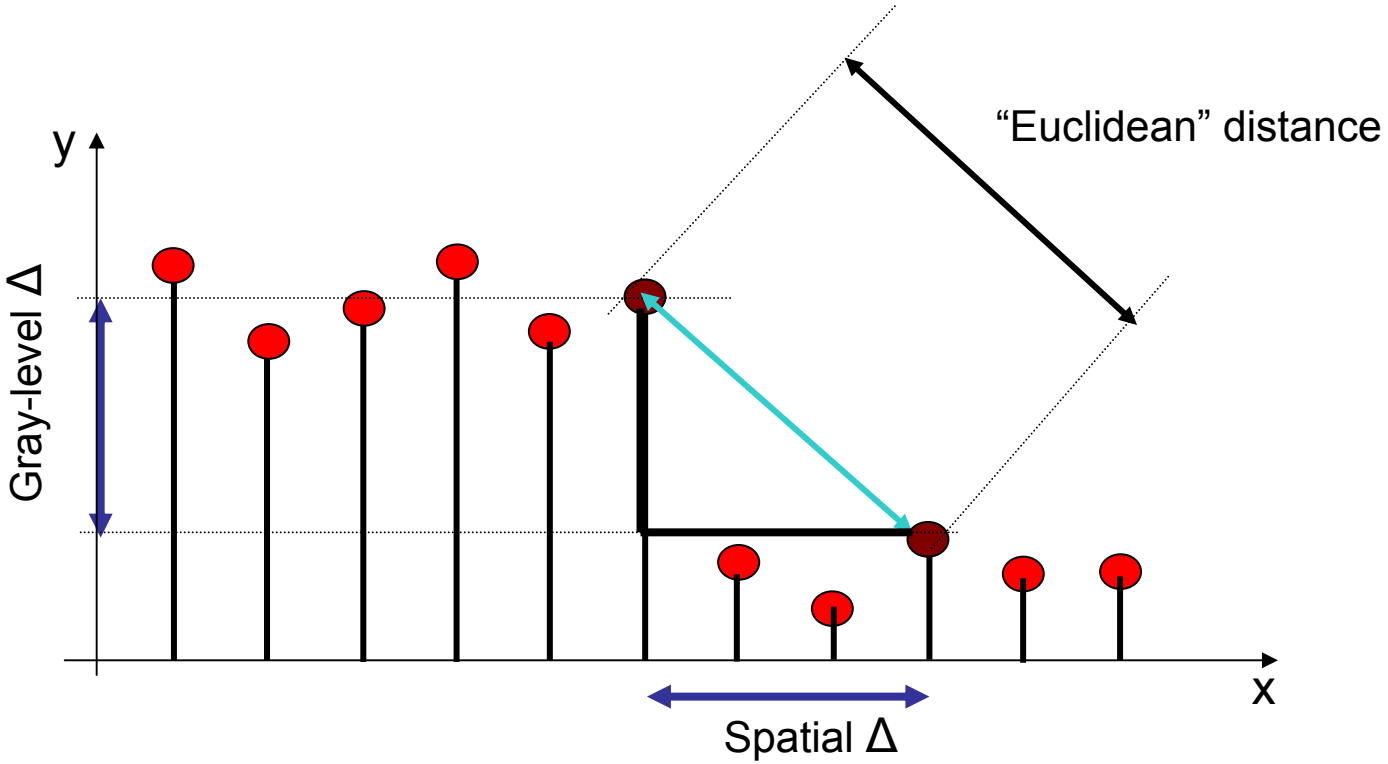
Pixelwise similarity metrics

- To measure the similarity of two pixels, we can consider
  - Spatial distance
  - Gray-level distance



Euclidean metrics

- Natural ways to incorporate the two Δs:
  - Bilateral Kernel [Tomasi, Manduchi, '98] (pixelwise)
  - Non-Local Means Kernel [Buades, et al. '05] (patchwise)



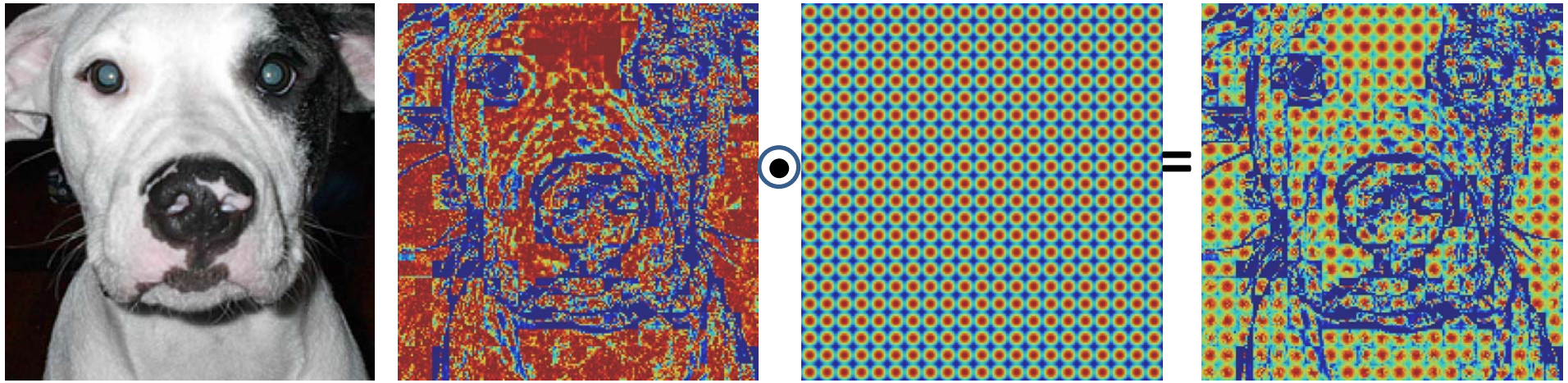
Bilateral Kernel (BL) [Tomasi et al. ‘98]

Pixels

$$K(\mathbf{x}_l, \mathbf{x}, y_l, y) = \exp\left\{-\frac{\|y_l - y\|^2}{h_r^2} - \frac{\|\mathbf{x}_l - \mathbf{x}\|^2}{h_d^2}\right\}$$

Pixel similarity

Spatial similarity



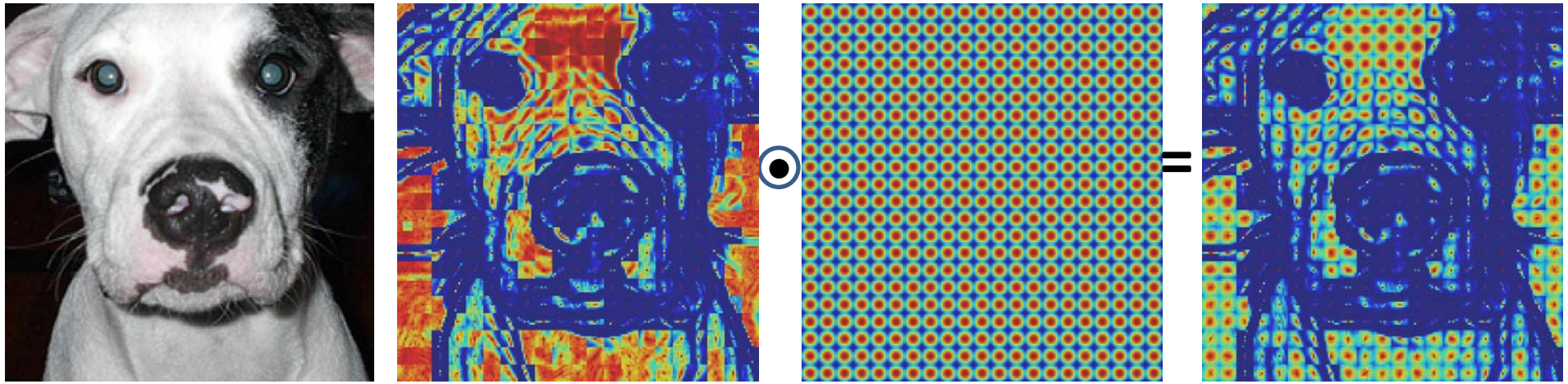
Non-local Means (NLM) [Buades et al. '05]

Patches

$$K(\mathbf{x}_l, \mathbf{x}, \mathbf{y}_l, \mathbf{y}) = \exp\left\{-\frac{\|\mathbf{y}_l - \mathbf{y}\|^2}{h_r^2} - \frac{\|\mathbf{x}_l - \mathbf{x}\|^2}{h_d^2}\right\}$$

Patch similarity

Spatial similarity



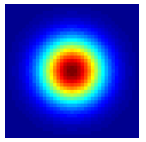
Smoothing effect

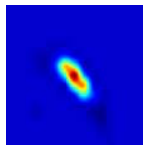
Defining Data-Adaptive Kernels

- *Classic* Kernel: Locally Linear Filter:

$$\hat{z}(\mathbf{x}) = \hat{\beta}_0 = \sum_i W(\mathbf{x}_i, \mathbf{x}, N)\, y_i$$

Uses distance $x\text{-}x_i$





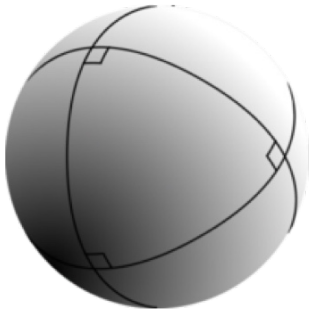
- *Data- Adaptive* Kernel:
  Locally Non-Linear Filter:

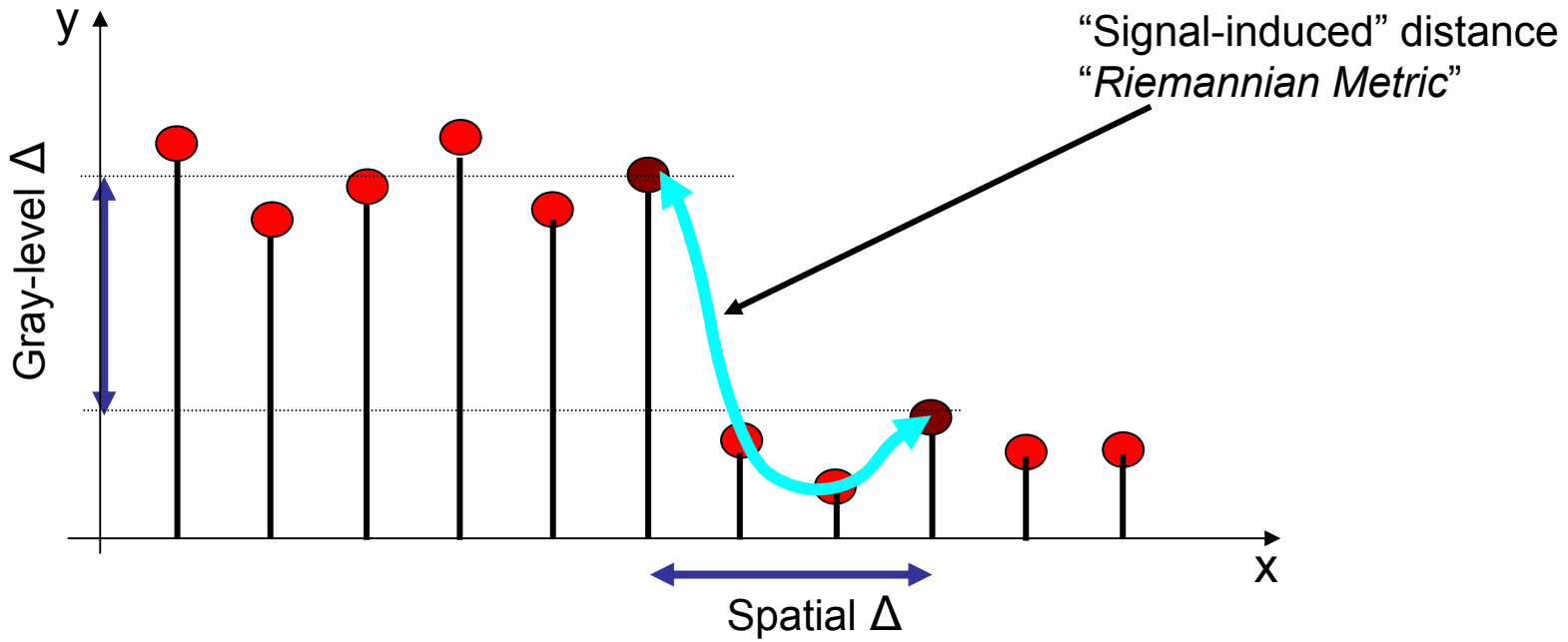
$$\hat{z}(\mathbf{x}) = \hat{\beta}_0 = \sum_i W(\mathbf{x}_i, \mathbf{x}, y_i, y, N)\, y_i$$

Uses $x\text{-}x_i$ and $y\text{-}y_i$

Beyond Euclidean metrics

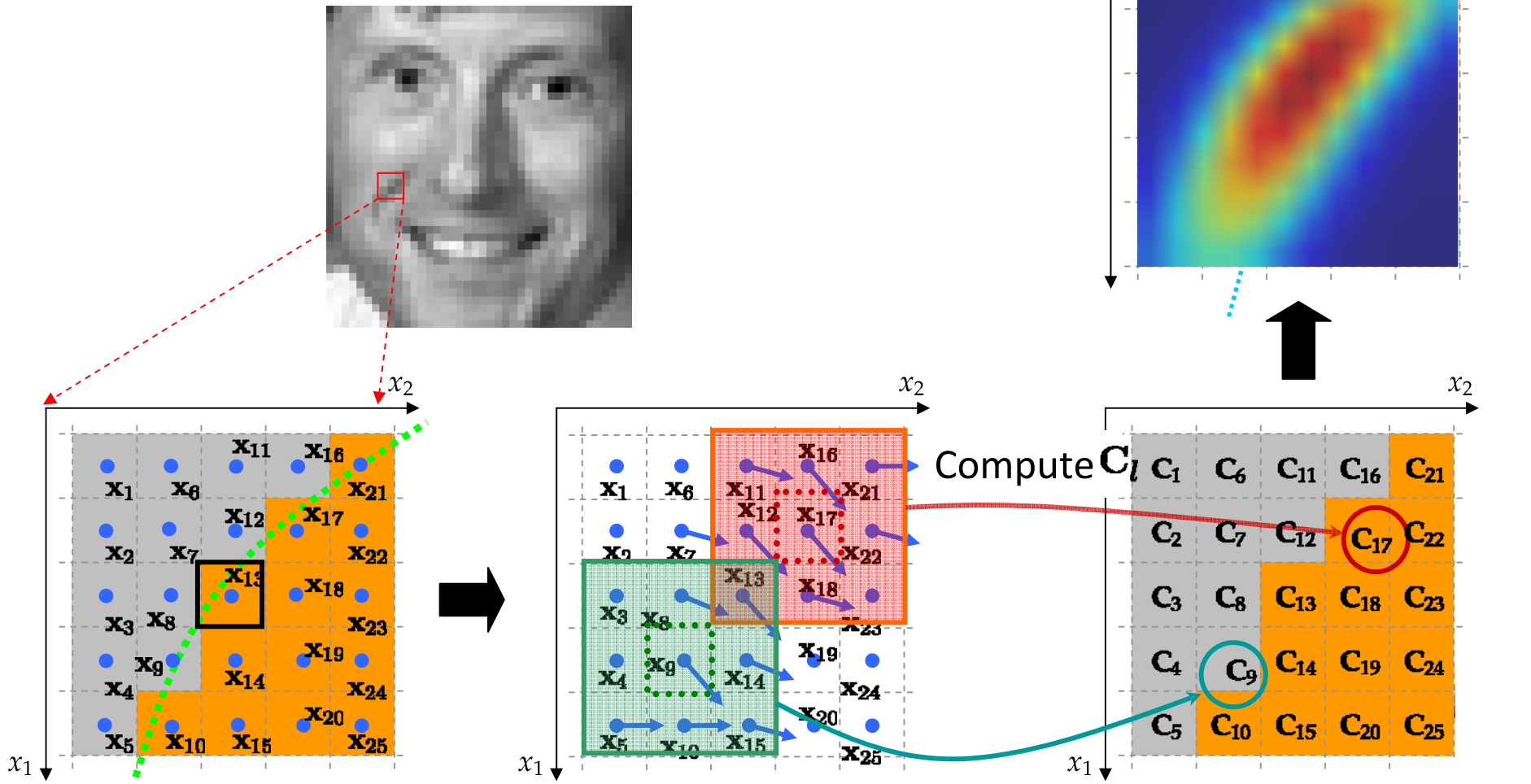
- Better similarity measures
- More effective ways to combine the two Δs:
  - LARK Kernel [Takeda, et al. '07]
  - Beltrami Kernel [Sochen, et al.'98]



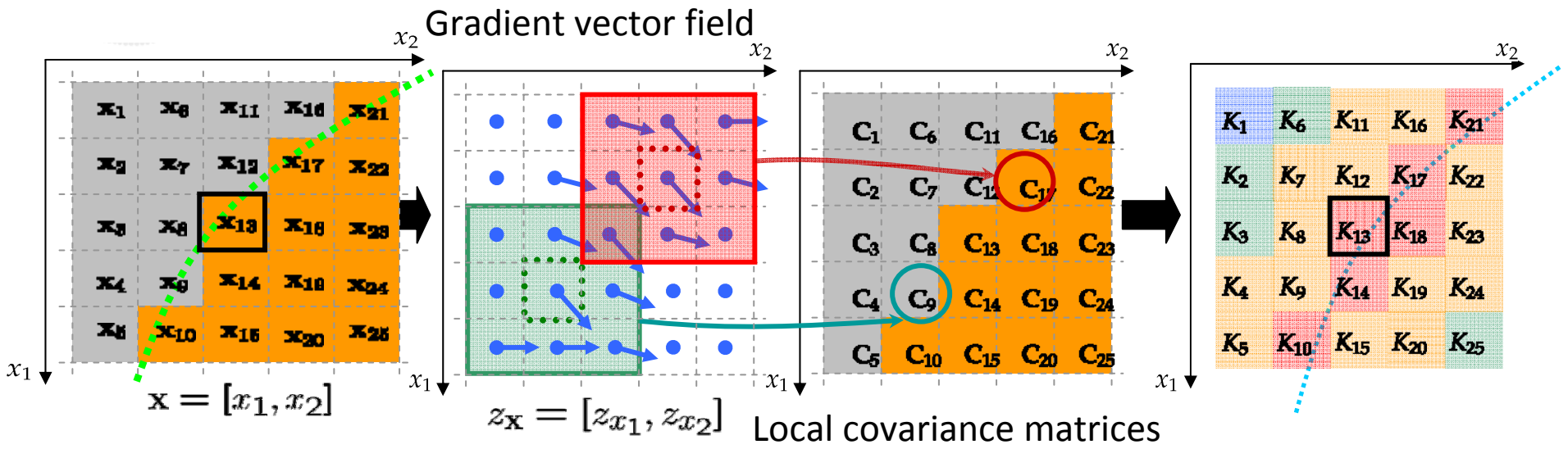


LARK Kernels

$$K(\mathbf{C}_l, \mathbf{x}_l, \mathbf{x}) = \exp\left\{-(\mathbf{x}_l - \mathbf{x})'\mathbf{C}_l(\mathbf{x}_l - \mathbf{x})\right\}$$



LARK Kernels



Locally Adaptive Regression Kernel: LARK

$$K(\mathbf{C}_l, \mathbf{x}_l, \mathbf{x}) = \exp\left\{-(\mathbf{x}_l - \mathbf{x})' \mathbf{C}_l (\mathbf{x}_l - \mathbf{x})\right\}$$

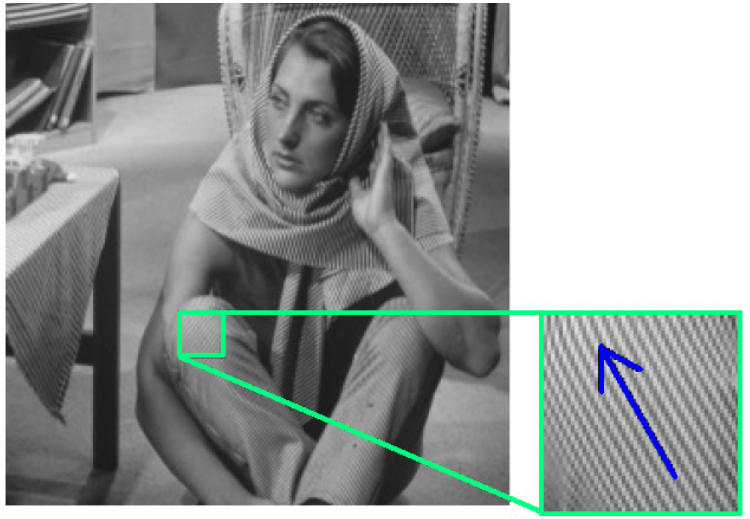
"Structure tensor"

$$\mathbf{C}_l = \sum_{k \in \Omega_l} \begin{bmatrix} z_{x_1}^2(\mathbf{x}_k) & z_{x_1}(\mathbf{x}_k) z_{x_2}(\mathbf{x}_k) \\ z_{x_1}(\mathbf{x}_k) z_{x_2}(\mathbf{x}_k) & z_{x_2}^2(\mathbf{x}_k) \end{bmatrix}$$

Gradient Covariance Matrix and Local Geometry

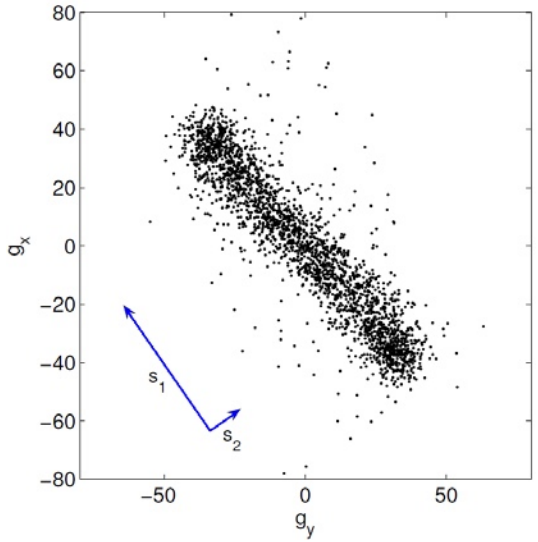
Gradient matrix over a local patch:

$$\mathbf{C}_l = \sum_{k \in \Omega_l} \begin{bmatrix} z_{x_1}^2(\mathbf{x}_k) & z_{x_1}(\mathbf{x}_k) z_{x_2}(\mathbf{x}_k) \\ z_{x_1}(\mathbf{x}_k) z_{x_2}(\mathbf{x}_k) & z_{x_2}^2(\mathbf{x}_k) \end{bmatrix}$$



$$\mathbf{C}_l = \mathbf{G}^{\mathbf{T}}\mathbf{G}$$

$$\mathbf{G} = \mathbf{U}\mathbf{S}\mathbf{V}^T = \mathbf{U} \begin{bmatrix} s_1 & 0 \\ 0 & s_2 \end{bmatrix} \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 \end{bmatrix}^T$$

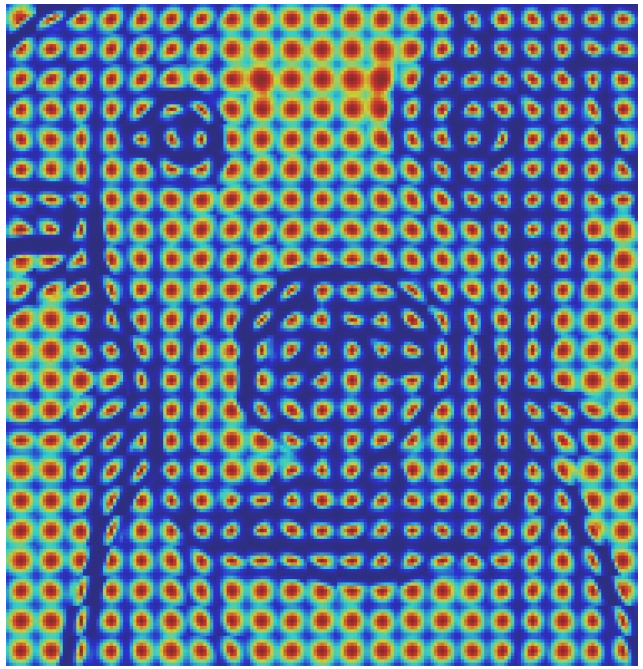


Capturing locally dominant orientations

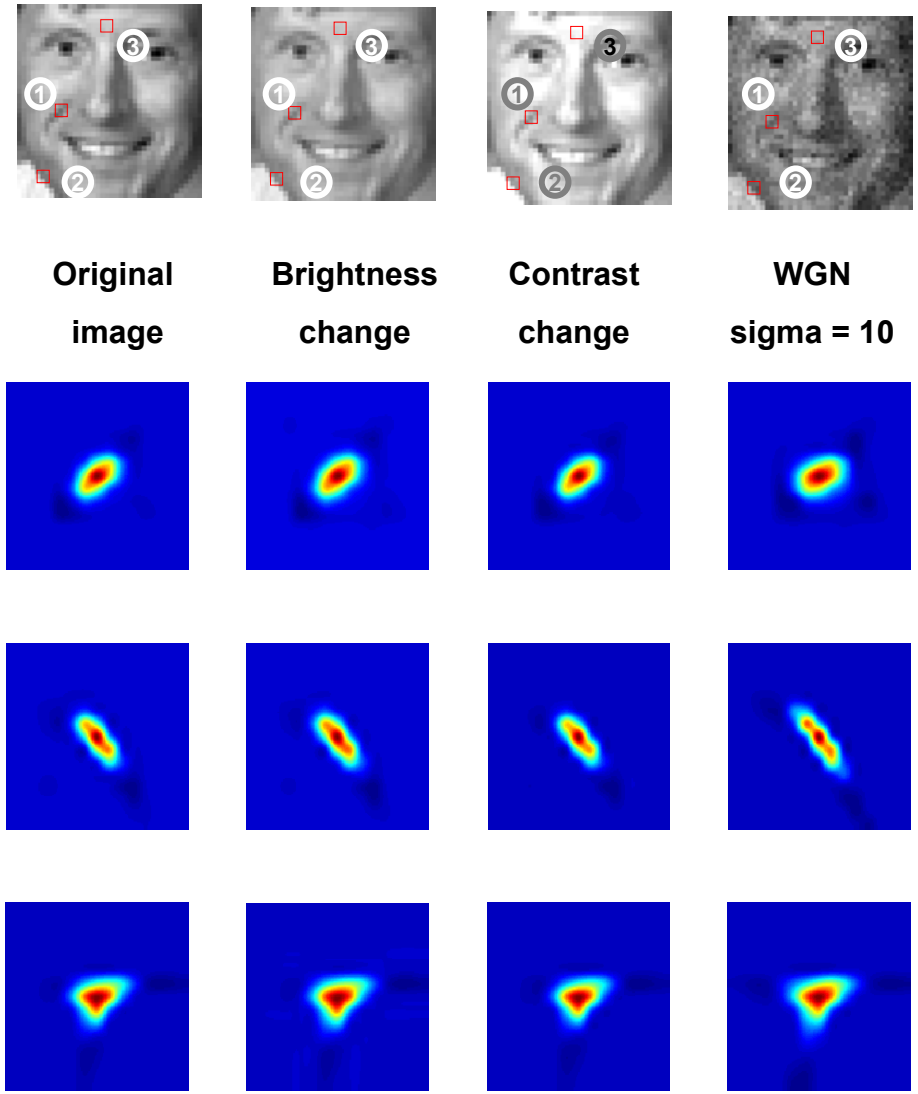
(Dense) LARK Kernels as Visual Descriptors [Seo and Milanfar '10]

$$K(\mathbf{C}_l, \mathbf{x}_l, \mathbf{x}) = \exp\left\{-(\mathbf{x}_l - \mathbf{x})'\mathbf{C}_l(\mathbf{x}_l - \mathbf{x})\right\}$$

Measure the similarity of pixels using the metric implied by the local structure of the image

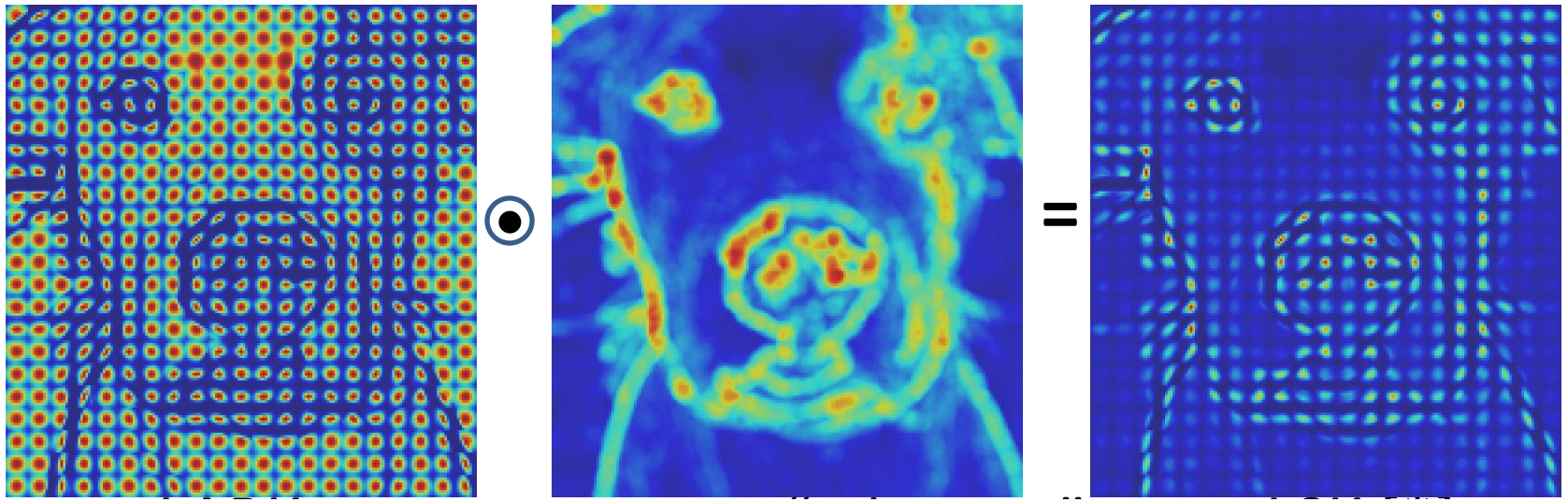


Robustness of LARK Descriptors



A Variant Better-suited for Restoration [Takeda et al. ’07]

$$K(\mathbf{C}_l, \mathbf{x}_l, \mathbf{x}) = \sqrt{\det \mathbf{C}_l} \exp\left\{-(\mathbf{x}_l - \mathbf{x})' \mathbf{C}_l (\mathbf{x}_l - \mathbf{x})\right\}$$



LARK Edge strength LSK

A Variant Better-suited for Restoration [Takeda et al. ’07]

Noisy image

LARK

Today

- Bilateral filtering
- Non-local means denoising
- LARK filter
- Image smoothing via region covariance (RegCov smoothing)
- Rolling guidance filter

Context-guided filtering

- Contextual knowledge extracted from local image regions guides the regularization process.
  - E. Erdem, A. Sancar-Yilmaz, and S. Tari, “Mumford-Shah Regularizer with Spatial Coherence”, In SSVM 2007
  - E. Erdem and S. Tari, “Mumford-Shah Regularizer with Contextual Feedback”, JMIV 2009

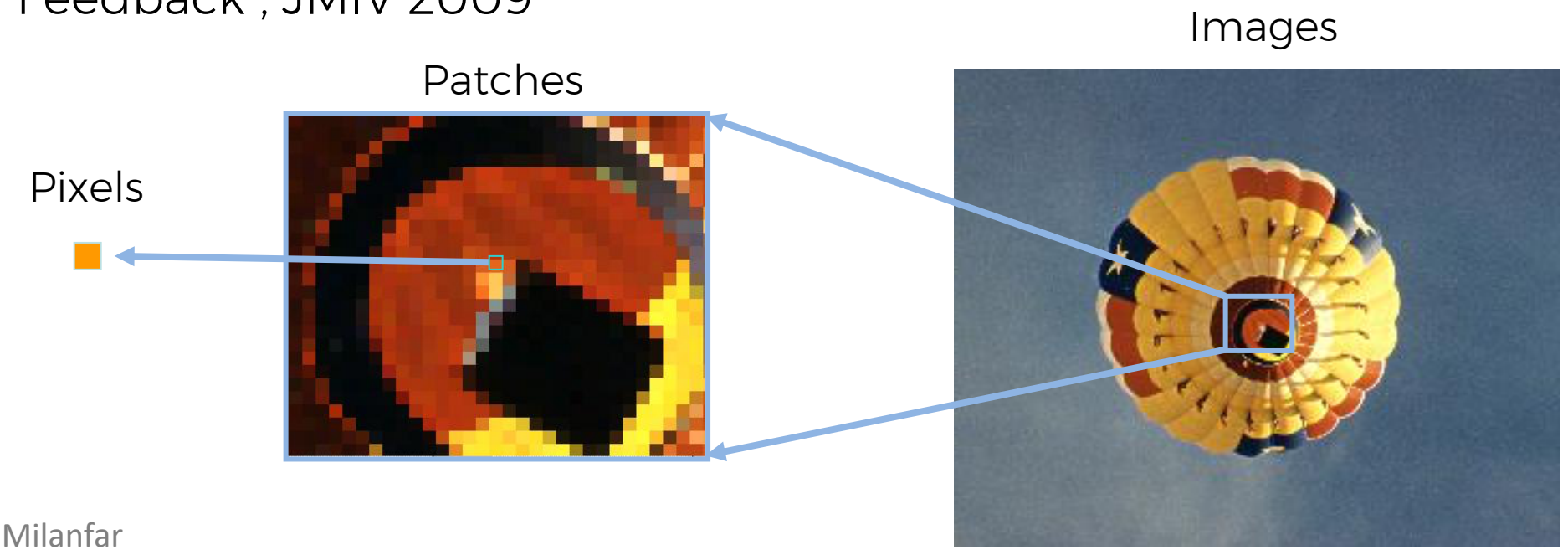


Image Credit: P. Milanfar

Structure-Texture Decomposition

- Decomposing an image into structure and texture components

Input Image

Structure-Texture Decomposition

- Decomposing an image into structure and texture components

Structure Component

Structure-Texture Decomposition

- Decomposing an image into structure and texture components

Texture Component

Structure-Texture Decomposition

- Decomposing an image into structure and texture components

Input Image

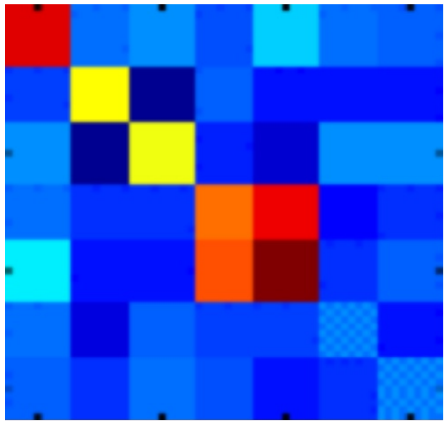
Structure

Texture

Structure-Texture Decomposition

$$F(x,y) = \phi(I,x,y)$$

$$F(x,y) = \left[I(x,y) \quad \left|\frac{\partial I}{\partial x}\right| \quad \left|\frac{\partial I}{\partial y}\right| \quad \left|\frac{\partial^2 I}{\partial x^2}\right| \quad \left|\frac{\partial^2 I}{\partial y^2}\right| \quad x \quad y \right]^T$$

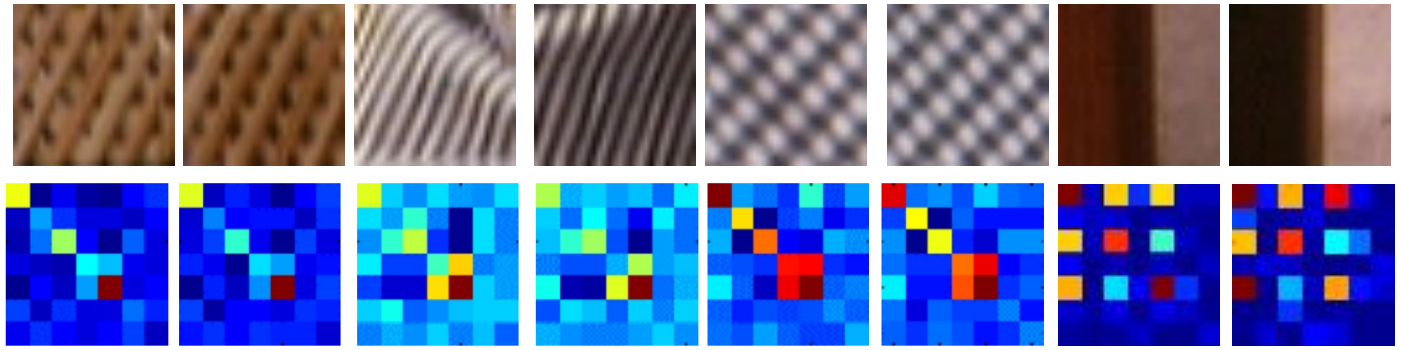


$$\mathbf{C}_R = \frac{1}{n-1}\sum_{i=0}^{n}(\mathbf{z}_k - \mu)(\mathbf{z}_k - \mu)^T$$

Tuzel et al., ECCV 2006

Structure-Texture Decomposition

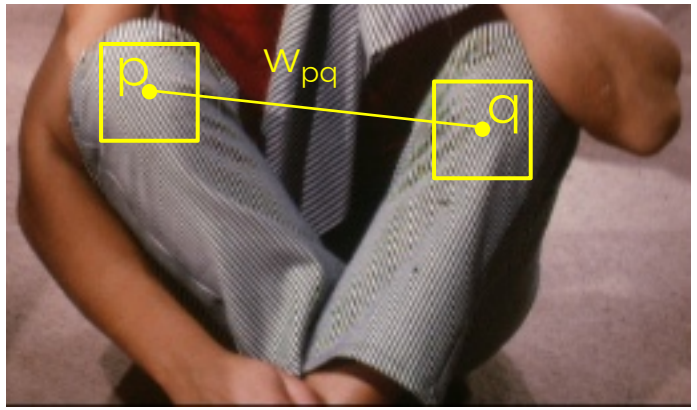
- Region covariances capture local structure and texture information.
- Similar regions have similar statistics.



RegCov Smoothing - Formulation

$$I = S + T$$

$$S(\mathbf{p}) = \frac{1}{Z_{\mathbf{p}}} \sum_{\mathbf{q} \in N(\mathbf{p},r)} w_{\mathbf{pq}} I(\mathbf{q})$$



- Structure-texture decomposition via smoothing
- Smoothing as weighted averaging
- Different kernels (w_{pq}) result in different types of filters.
- Three novel patch-based kernels for structure texture decomposition.

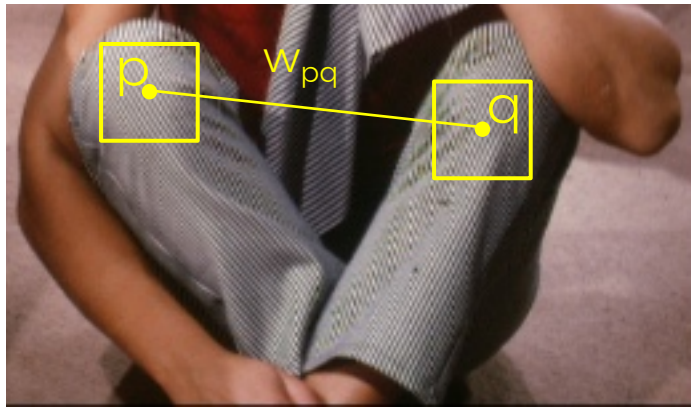
– L. Karacan, A. Erdem, E. Erdem, “Structure Preserving Image Smoothing via Region Covariances”, ACM TOG 2013 (SIGGRAPH Asia 2013)

RegCov Smoothing – Model 1

- Depends on sigma-points representation of covariance matrices (Hong et al.,CVPR'09)

$\mathbf{C} = \mathbf{L}\mathbf{L}^T$ Cholesky Decomposition

$\mathcal{S} = \{\mathbf{s}_i\}$ Sigma Points $\quad \mathbf{s}_i = \begin{cases} \alpha\sqrt{d}\mathbf{L}_i & \text{if } 1 \leq i \leq d \\ -\alpha\sqrt{d}\mathbf{L}_i & \text{if } d+1 \leq i \leq 2d \end{cases}$



Final representation

$$\Psi(\mathbf{C}) = (\mu, \mathbf{s}_1, \ldots, \mathbf{s}_d, \mathbf{s}_{d+1}, \ldots, \mathbf{s}_{2d})^T$$

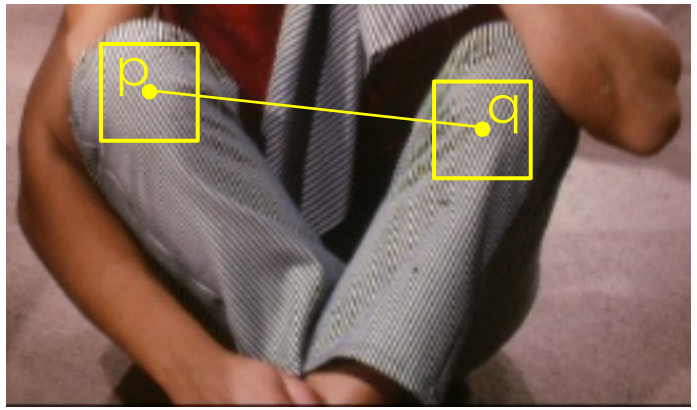
Resulting kernel function

$$w_{\mathbf{pq}} \propto \exp\left(-\frac{\|\Psi(\mathbf{C_p}) - \Psi(\mathbf{C_q})\|^2}{2\sigma^2}\right)$$

RegCov Smoothing – Model 2

- An alternative way is to use statistical similarity measures.
- A Mahalanobis-like distance measure to compare to image patches.

$$d(\mathbf{p},\mathbf{q}) = \sqrt{(\mu_\mathbf{p} - \mu_\mathbf{q})\mathbf{C}^{-1}(\mu_\mathbf{p} - \mu_\mathbf{q})^T}$$

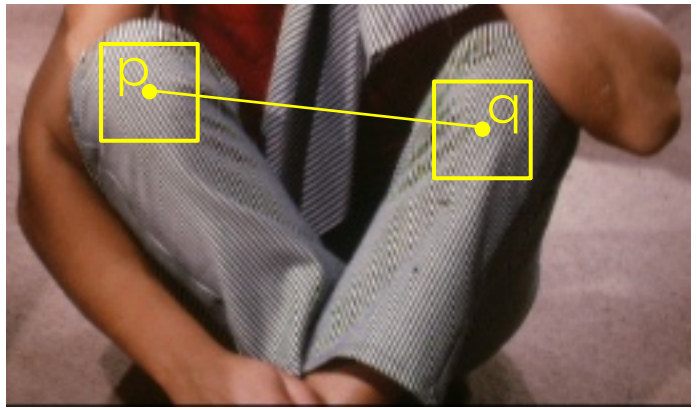


$$\mathbf{C} = \mathbf{C_p} + \mathbf{C_q}$$

Resulting kernel $w_\mathbf{pq} \propto \exp\left(-\frac{d(\mathbf{p},\mathbf{q})^2}{2\sigma^2}\right)$

RegCov Smoothing – Model 3

- We use Kullback-Leibler(KL)-Divergence measure from probability theory.
- A KL-Divergence form is used to calculate statistical distance between two multivariate normal distribution

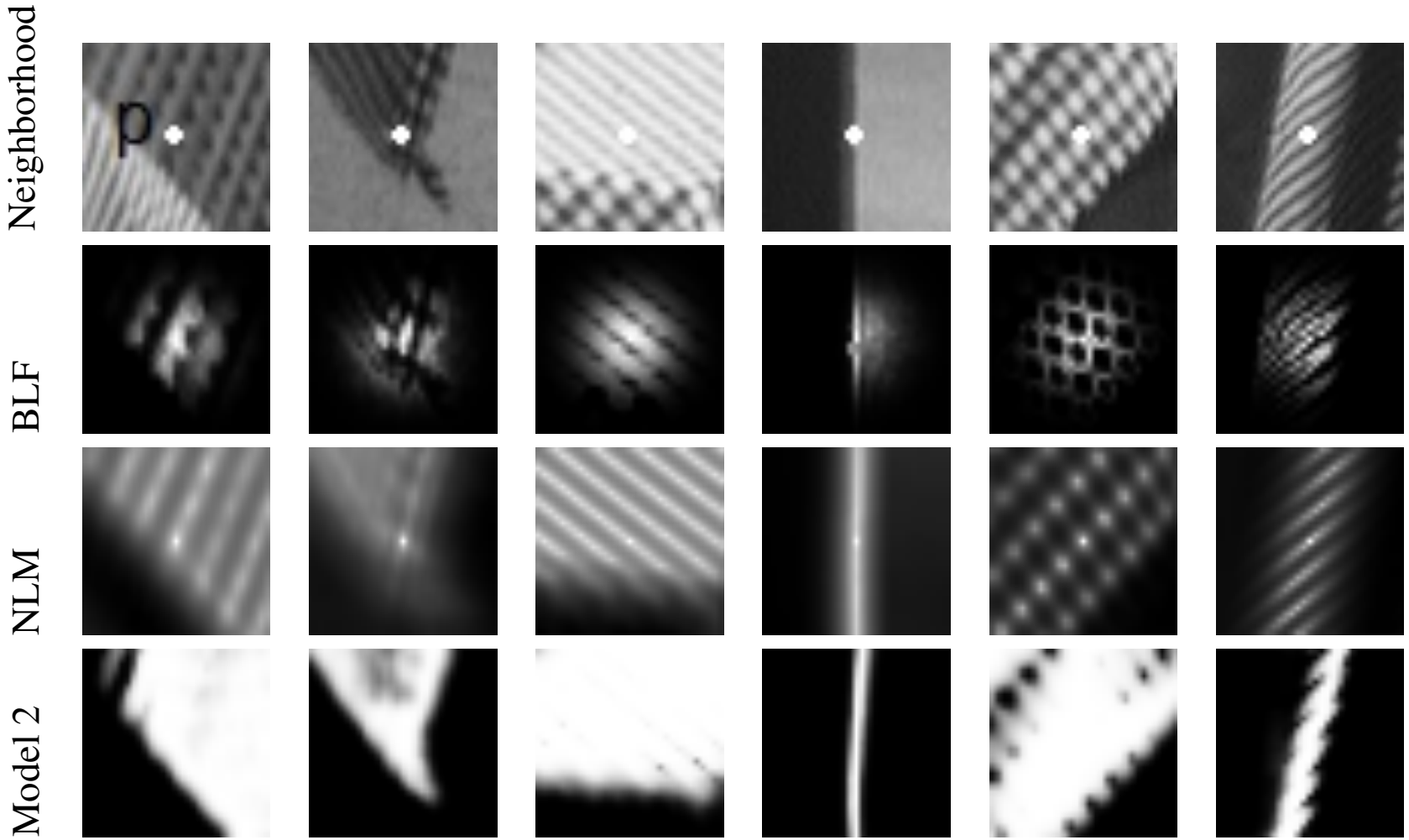


$$d_{KL}(\mathbf{p},\mathbf{q}) = \frac{1}{2}\left(tr(\mathbf{C_q}^{-1}\mathbf{C_p}) + (\mu_p - \mu_q)^T \mathbf{C_q}^{-1}(\mu_p - \mu_q) - k - ln\left(\frac{\det \mathbf{C_p}}{\det \mathbf{C_q}}\right)\right)$$

Resulting kernel $\quad w_{pq} \propto \frac{d_{KL}(\mathbf{p},\mathbf{q})}{2\sigma^2}$

resulted from a discussion with Rahul Narain (Berkeley University)

RegCov Smoothing – Smoothing Kernels



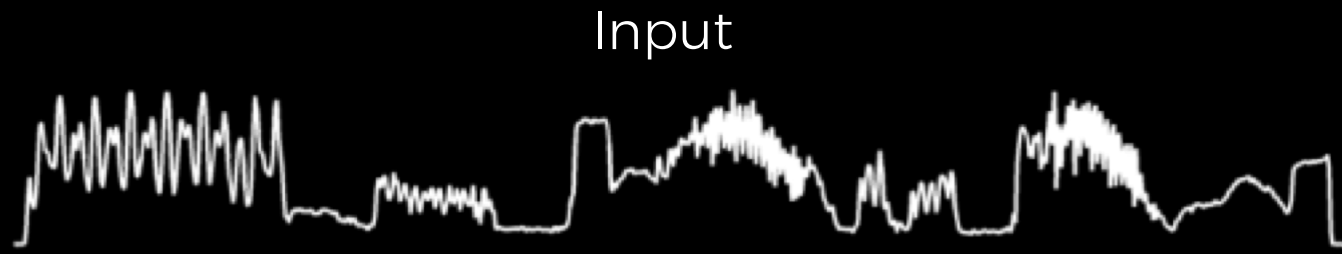
Results

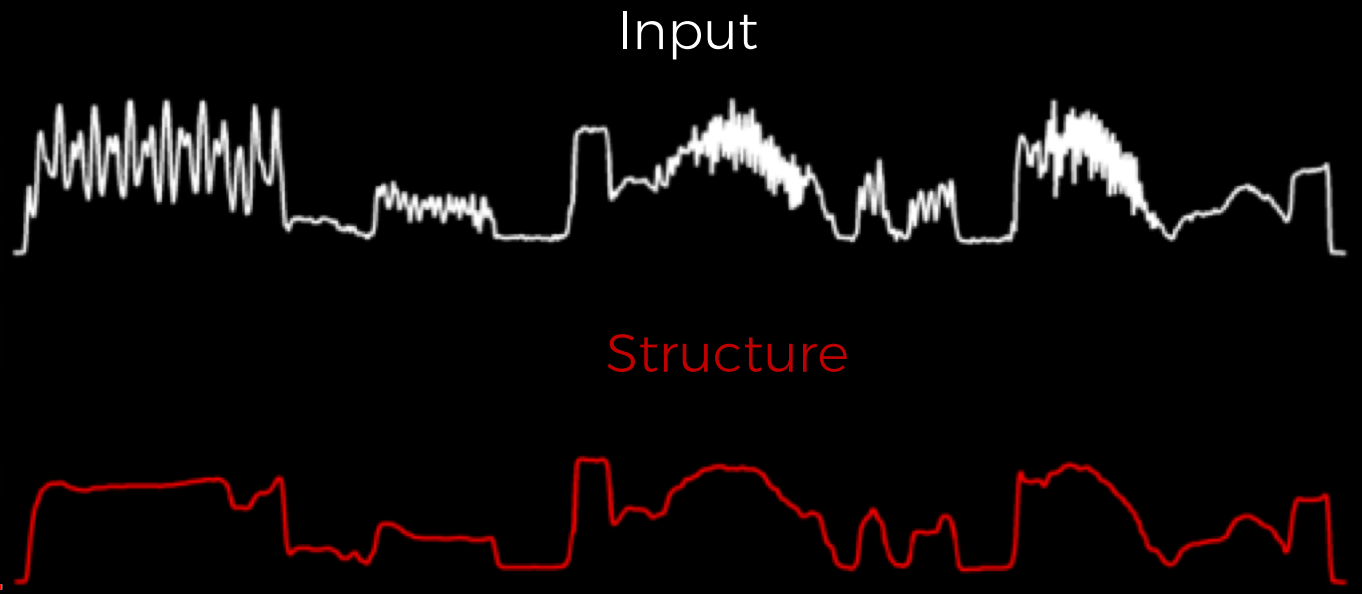
Input

Results

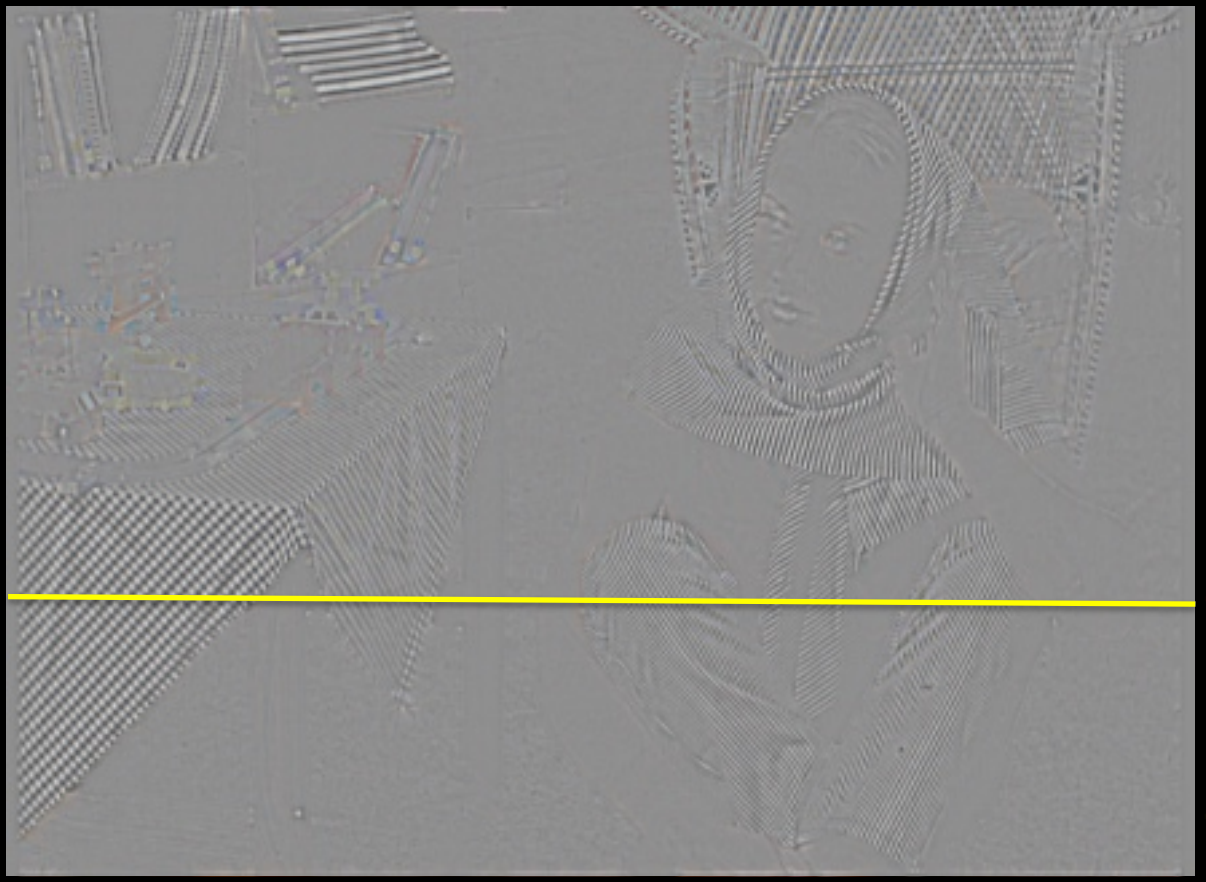
Model1 Model2 Model3

Input



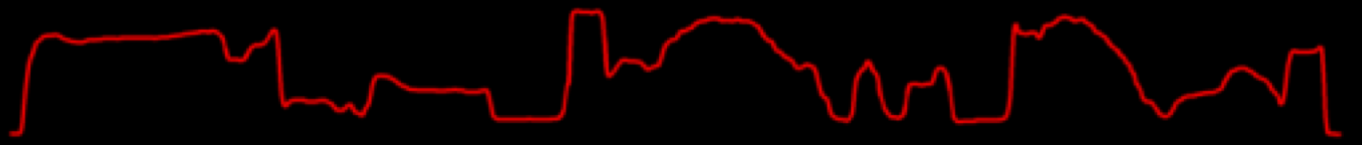


Model2 Structure



Input

Structure

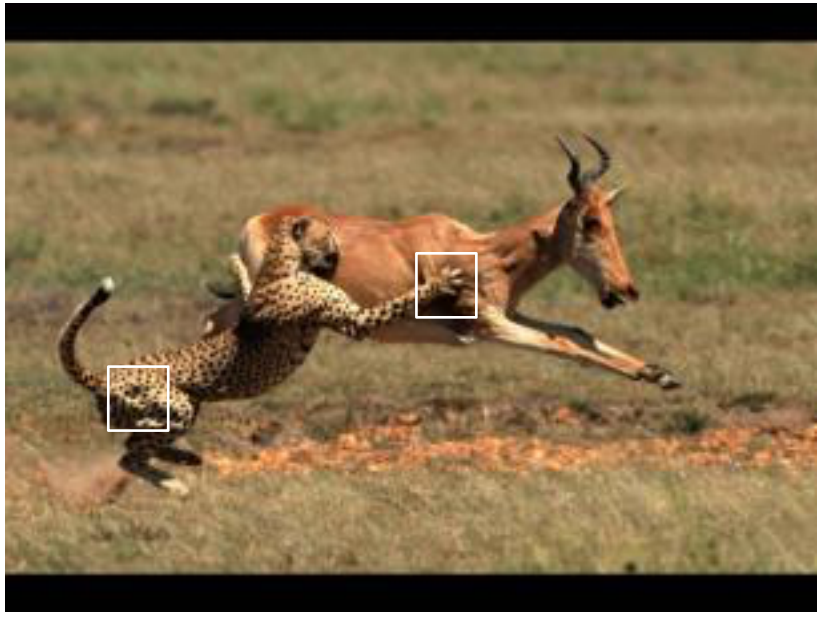


Texture

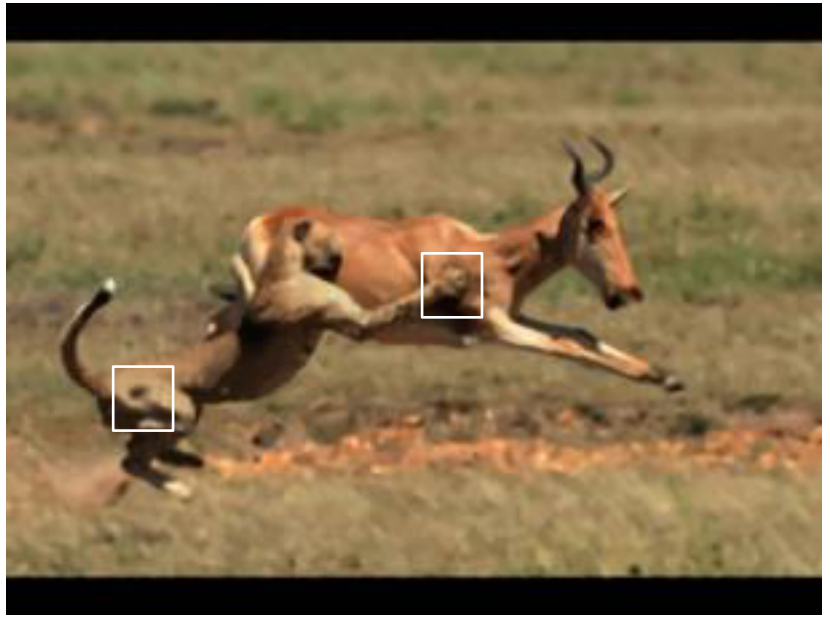
Model2 Texture

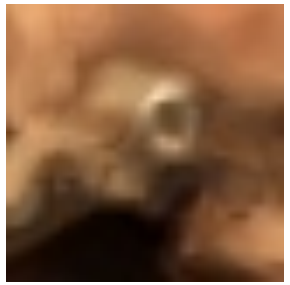
Results

Input

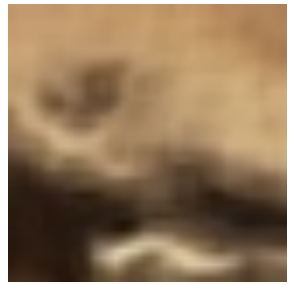


Model2





Model3



Experimental evaluation

Input

Experimental evaluation

TV

Rudin et al. 1992

Experimental evaluation

Bilateral Filter

Experimental evaluation

Envelope Extraction

Subr et al. 2009

Experimental evaluation

RTV

Xu et al. 2012

Experimental evaluation

Model 1

Experimental evaluation

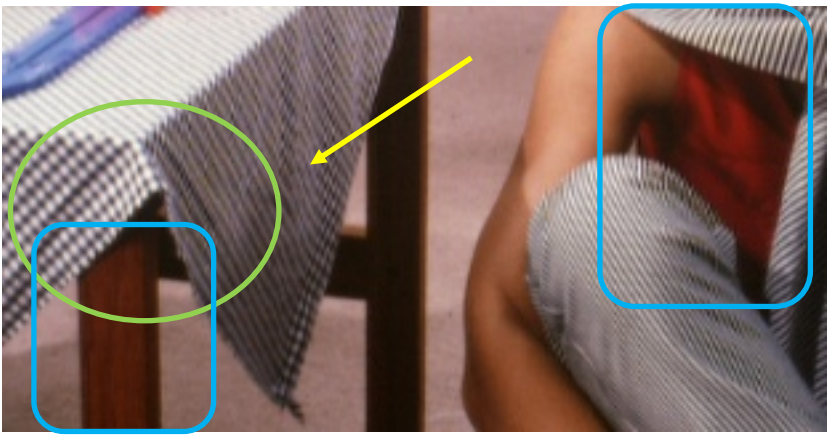
Model 2

Experimental evaluation

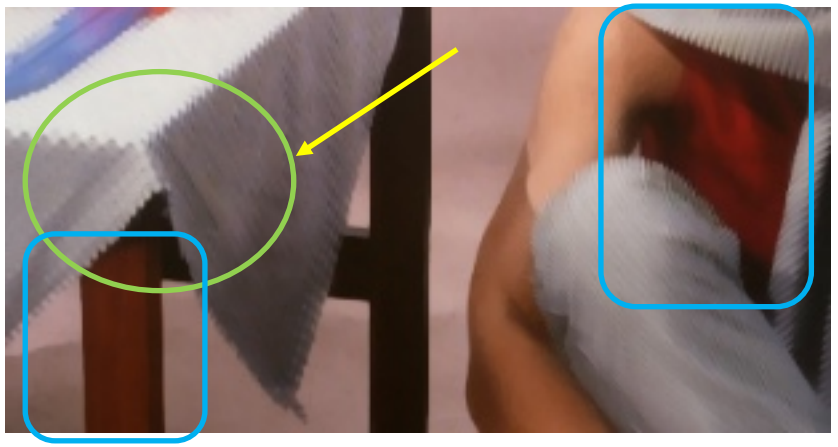
Model 3

Experimental evaluation

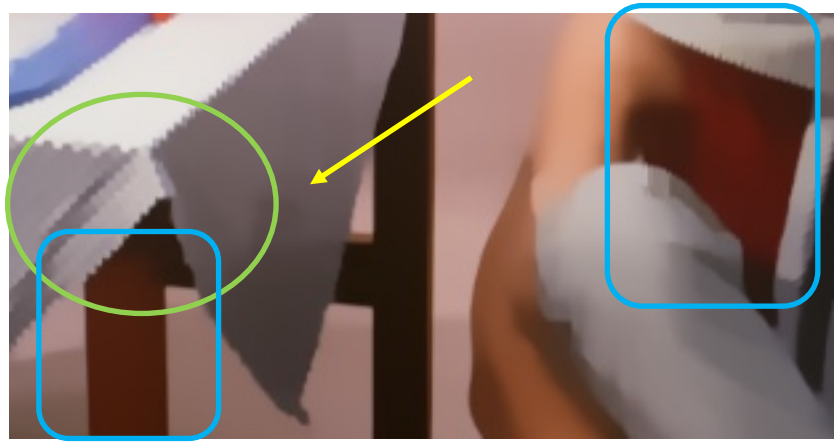
Input



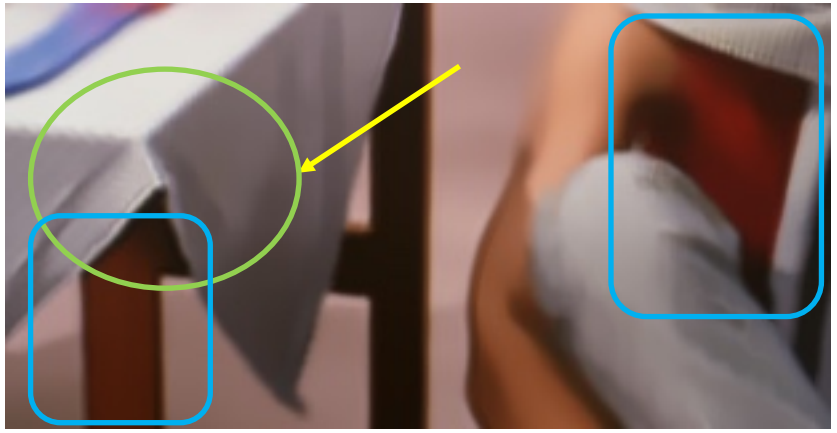
Local Exrema



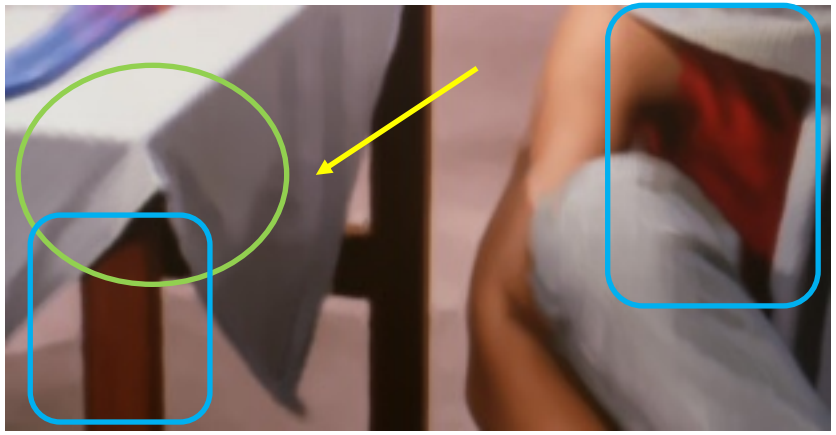
RTV



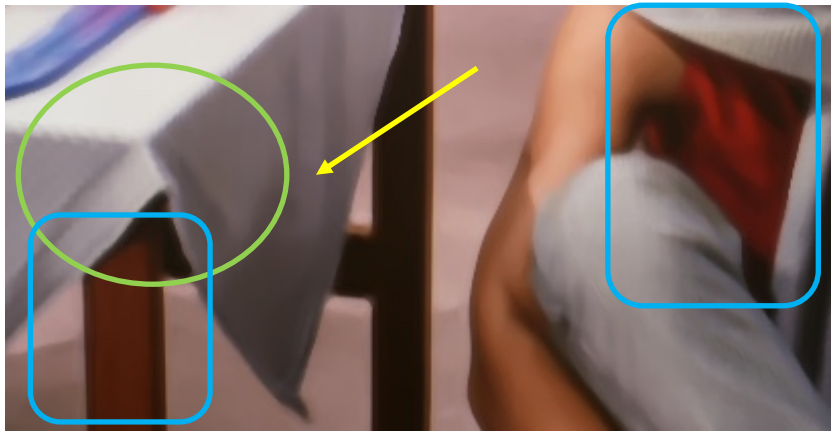
Model1



Model2



Model3

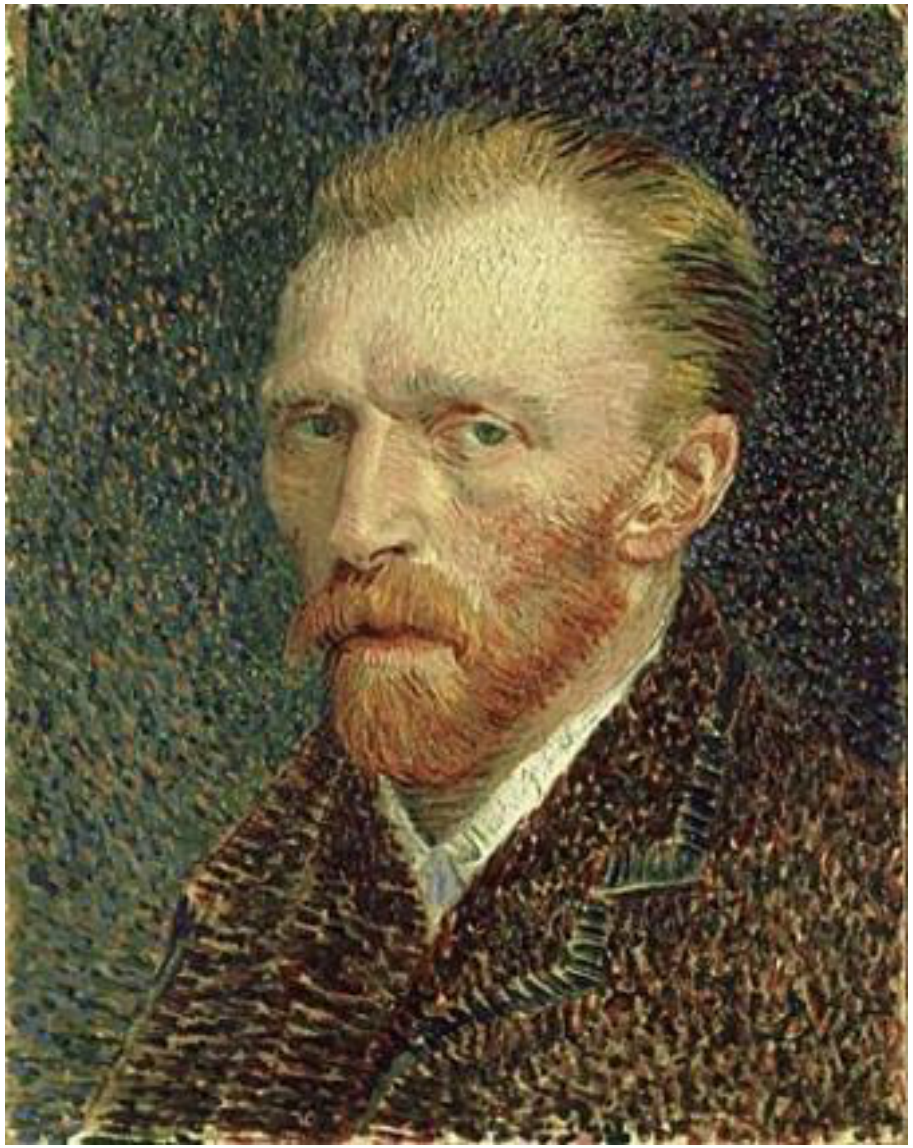


Shading preserved

Structure preserved

No unintuitive edge

Multiscale decomposition



Multiscale decomposition

$S_1(k = 5)$

Multiscale decomposition

$S_2(k = 7)$

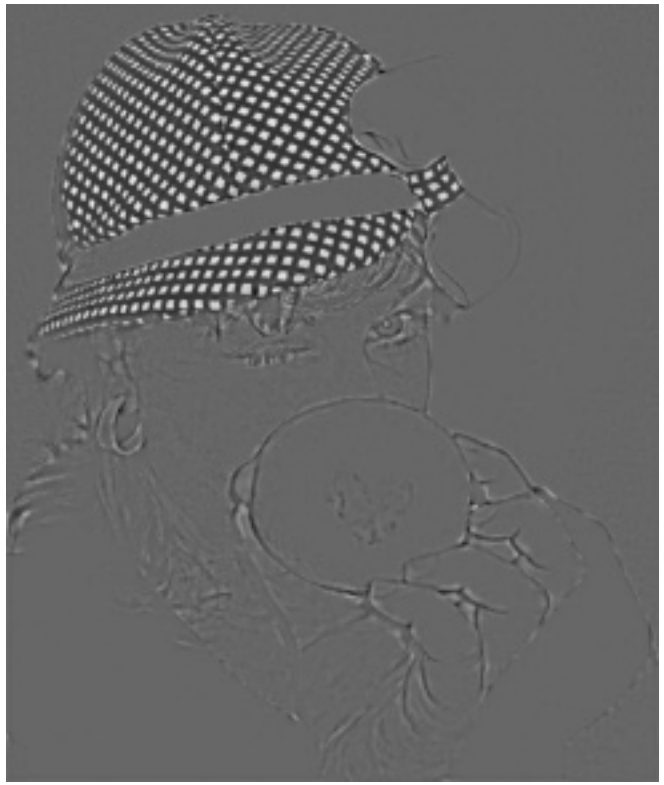
Multiscale decomposition

$S_3(k = 9)$

Challenging cases

Input

Model2Texture



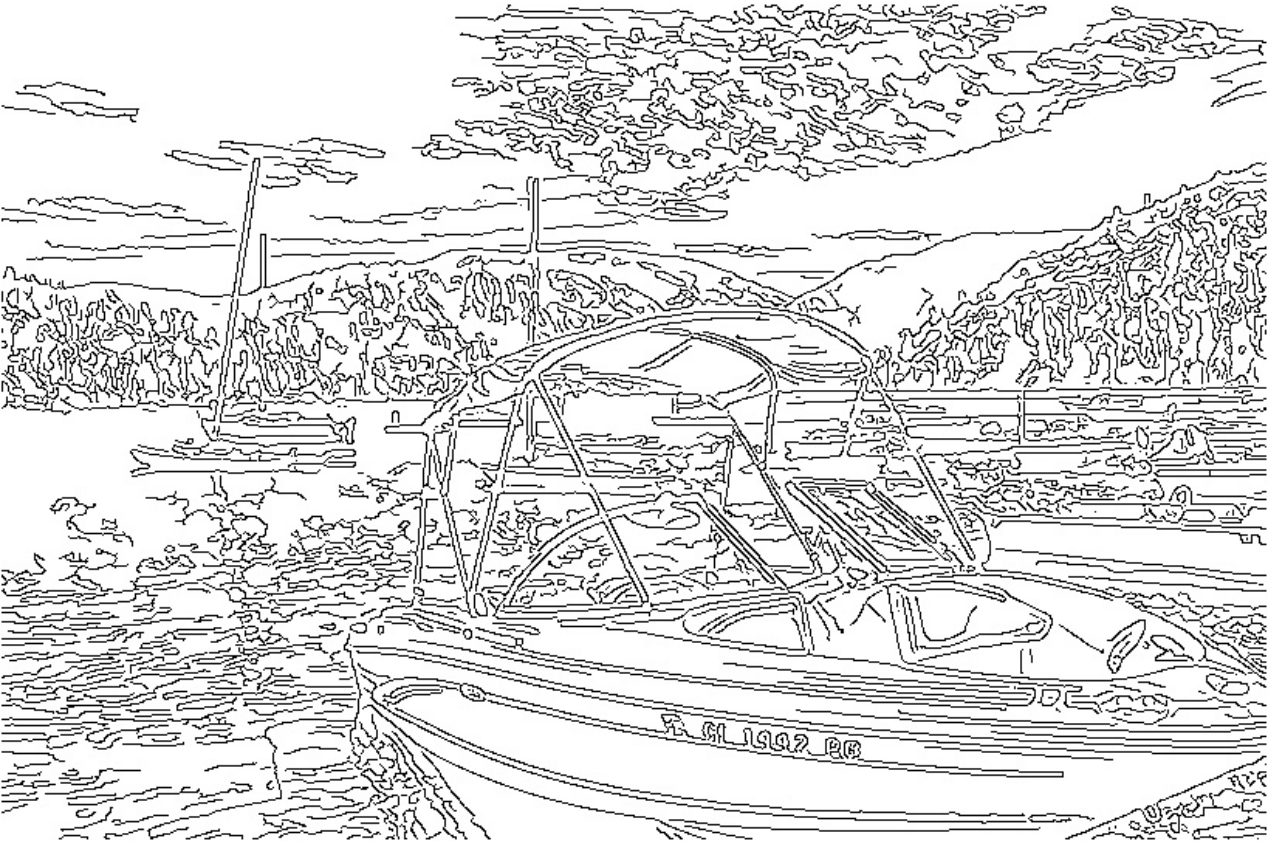
Model2+Model1

Edge detection

Edge detection

Edge detection

Canny edges of original image



Canny edges of smoothed image

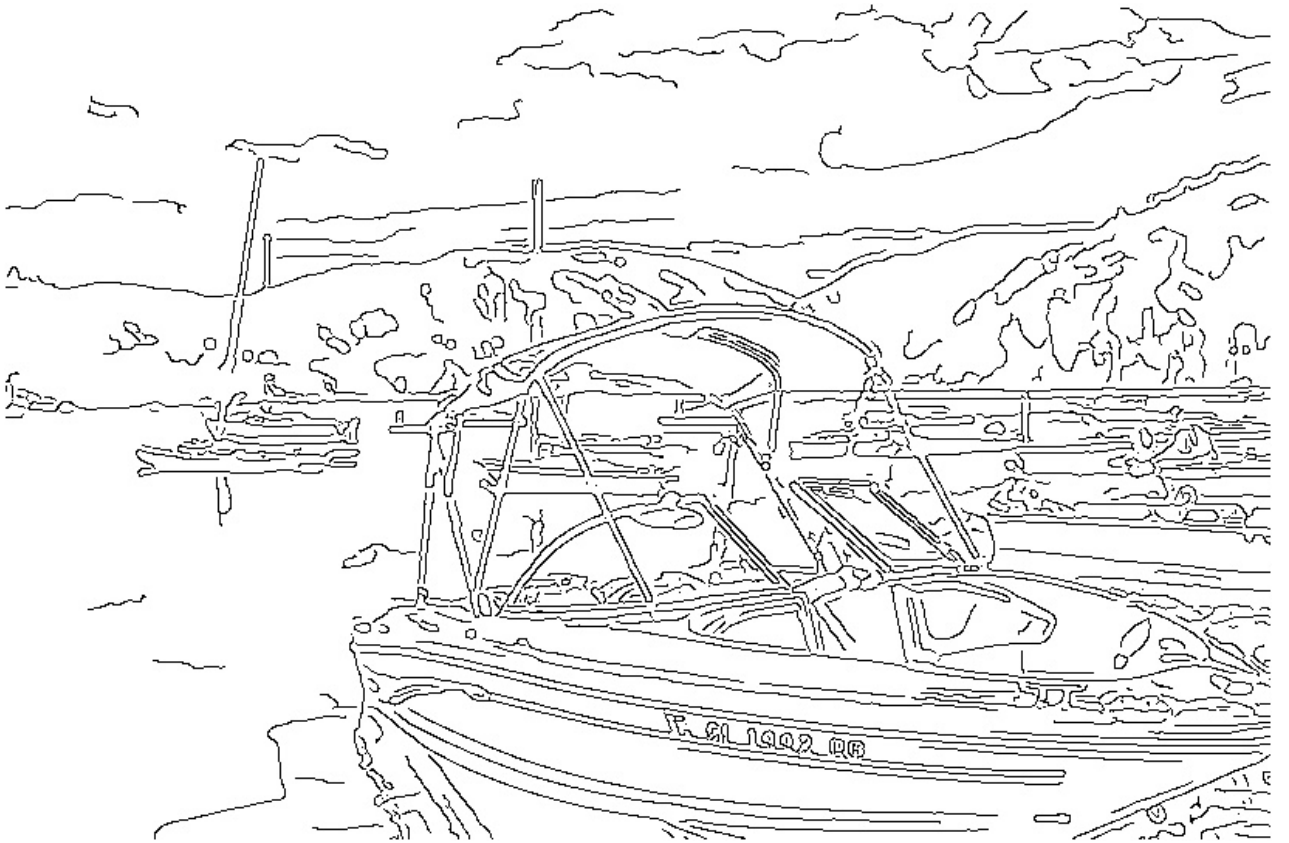


Image abstraction

Image abstraction

Detail boosting

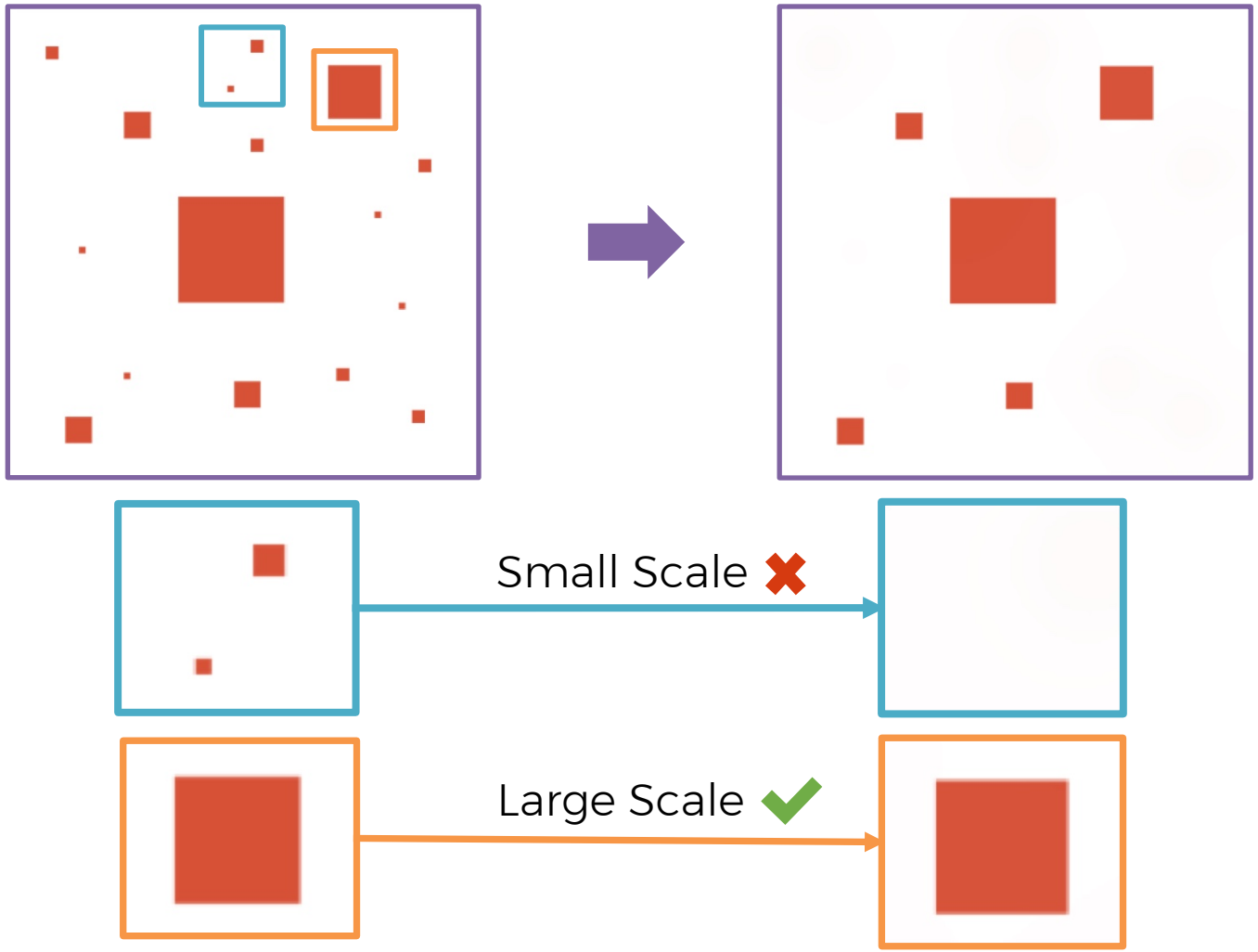
Image composition

Today

- Bilateral filtering
- Non-local means denoising
- LARK filter
- Image smoothing via region covariance (RegCov smoothing)
- Rolling guidance filter

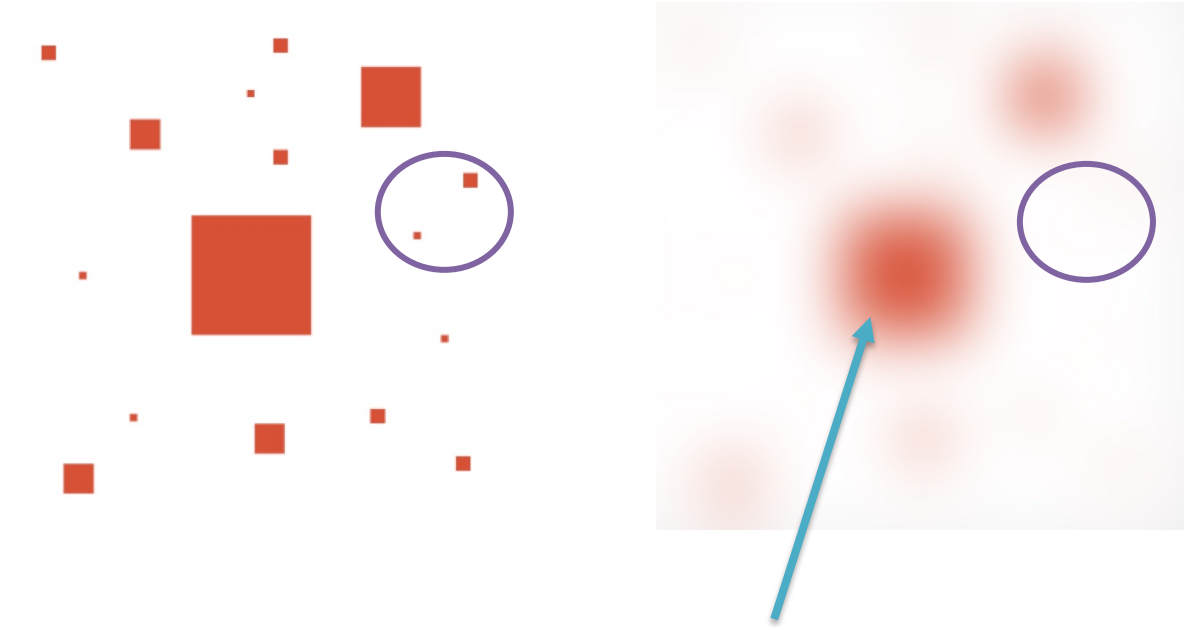
Slides are adapted from the ones prepared by J. Jia.

Scale-Aware Filtering



Main Idea

- Scale Space Theory [Lindeberg, 1994]:
  - An object of size t, will be largely smoothed away with Gaussian filter of variance t^2.



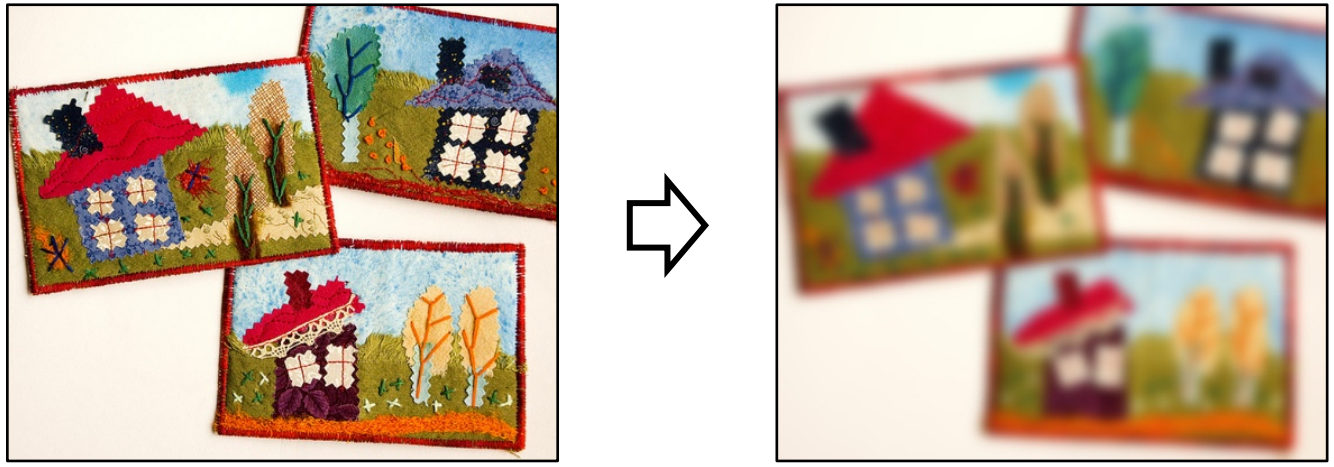
RGF: A scale-aware Filter

Step 1
Small
Structures
Removal

Step 2
Edge Recovery

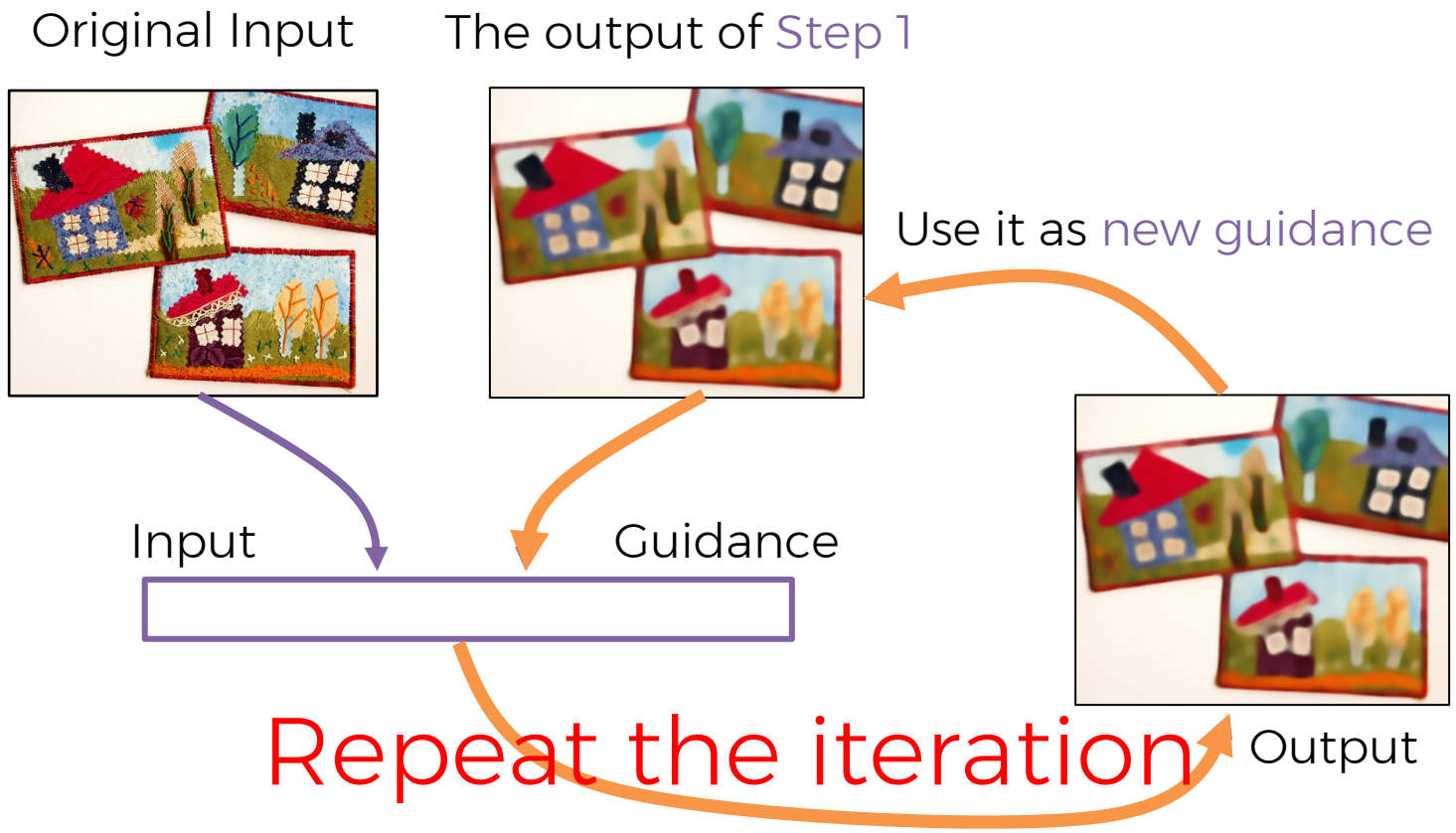
Step 1: Small Structures Removal

Gaussian Filter

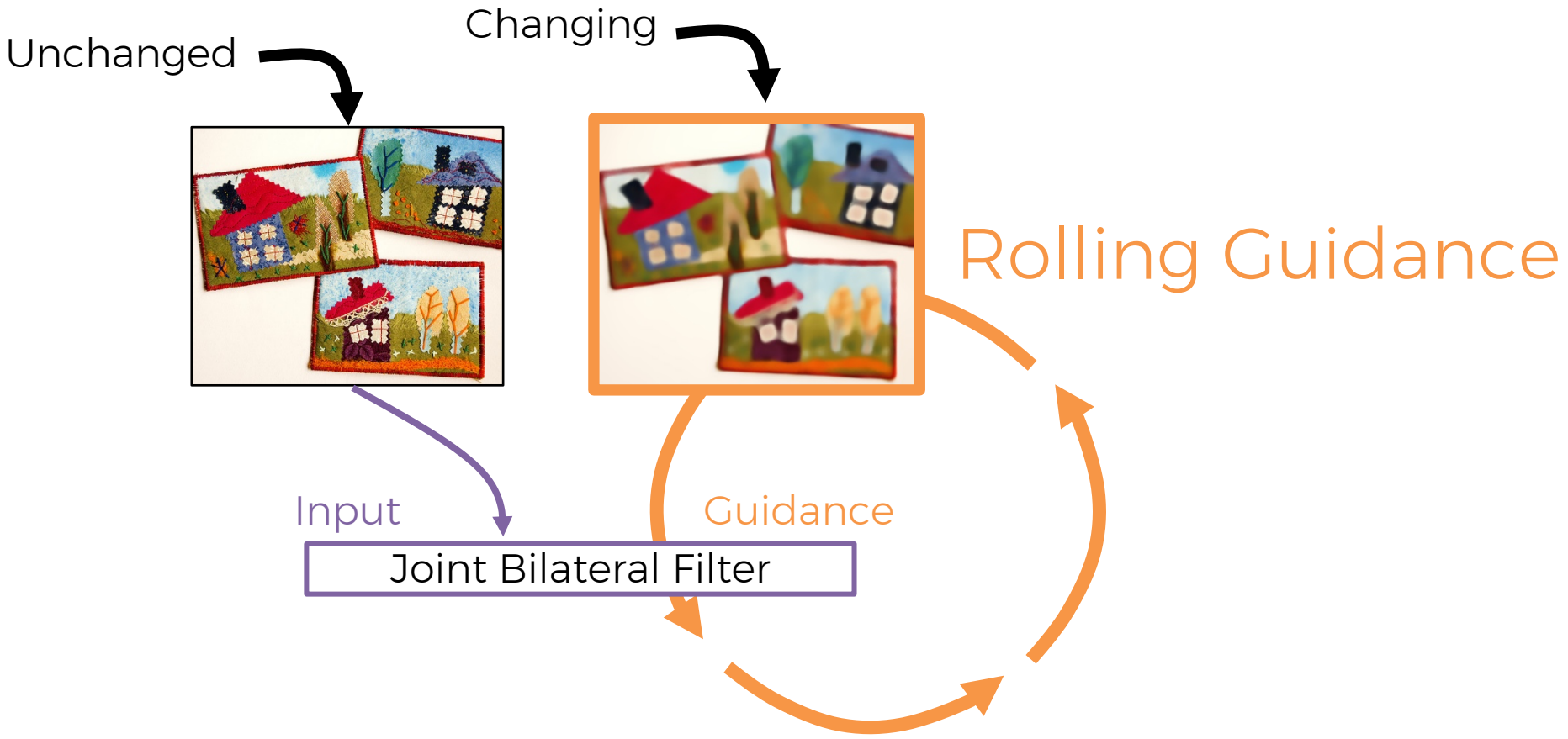


Step 2: Edge Recovery

- A rolling guidance



Rolling Guidance



Rolling Guidance

Guidance for the 1st iteration

Rolling Guidance

Guidance for the 2nd iteration

Rolling Guidance

Guidance for the 3rd iteration

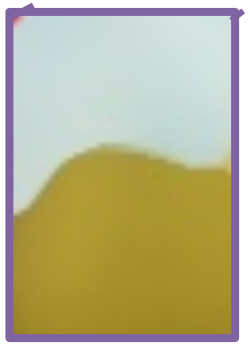
Rolling Guidance

Guidance for the 5th iteration

Rolling Guidance

Input

Output



Small structures are removed.
Large structure are NOT blurred.

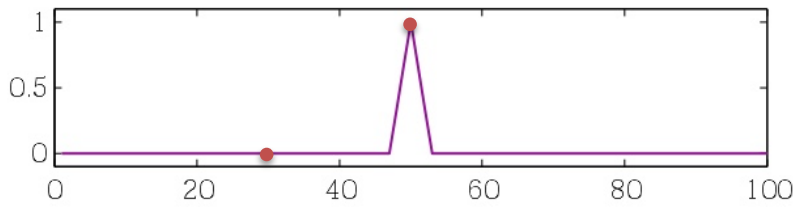
Implementation

Rolling Guidance Filter (RGF) has only 1 line of code

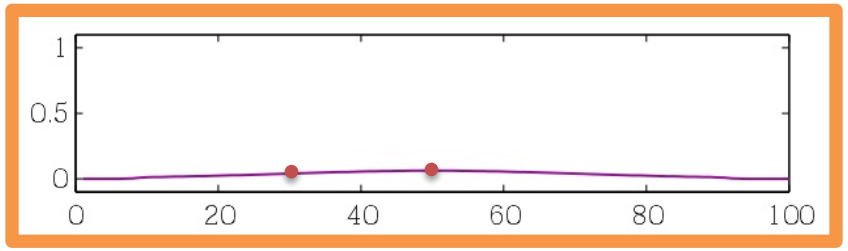
```
Mat rollingGuidanceFilter(Mat im, float scale, int iter){
    Mat res = im.mul(0);
    while(iter--) res = bilateralFilter(im,res,scale,SIGMA_R);
    return res;
}
```

Small Structure

Input



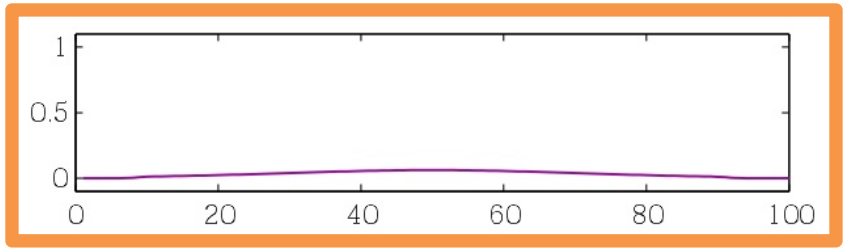
Guidance (output of step 1)



$$J^{t+1}(p) = \frac{1}{K_p} \sum_{q \in N(p)} \exp\left(- \frac{\|p-q\|^2}{2\sigma_s^2} \right)$$

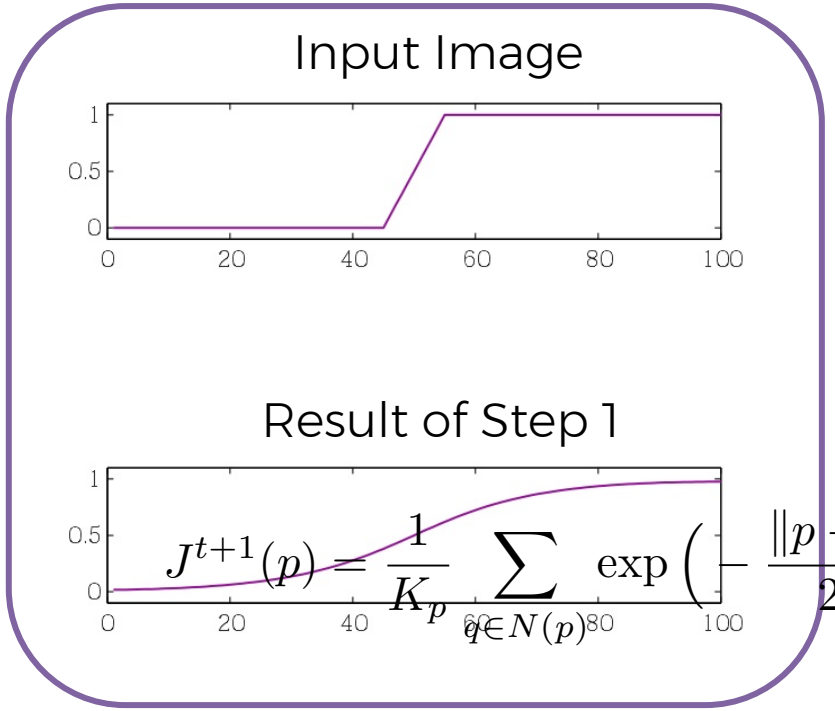
It becomes a Gaussian filter

Joint Bilateral Filter



Same

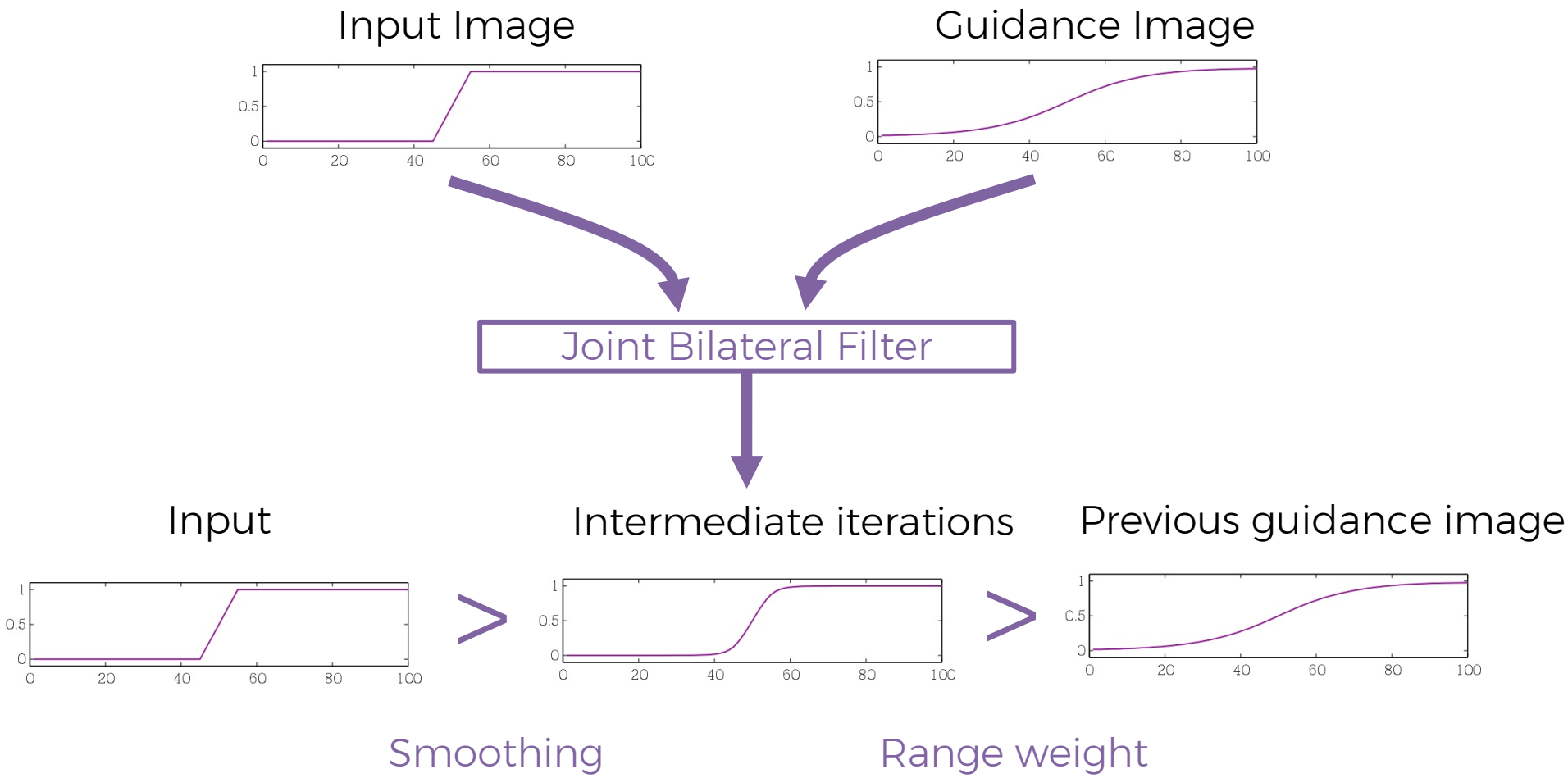
Large Structure



$$J^{t+1}(p) = \frac{1}{K_p} \sum_{q \in N(p)} \exp\left(- \frac{\|p-q\|^2}{2\sigma_s^2} - \frac{\|J^t(p) - J^t(q)\|^2}{2\sigma_r^2} \right) I(q)$$

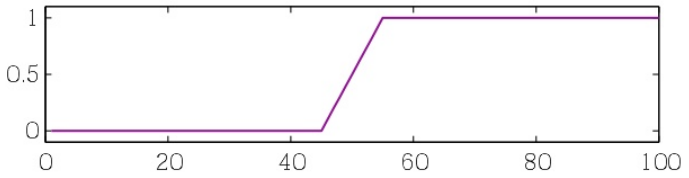
Due to this range weight
It generates sharper results than Gaussian!

Processing

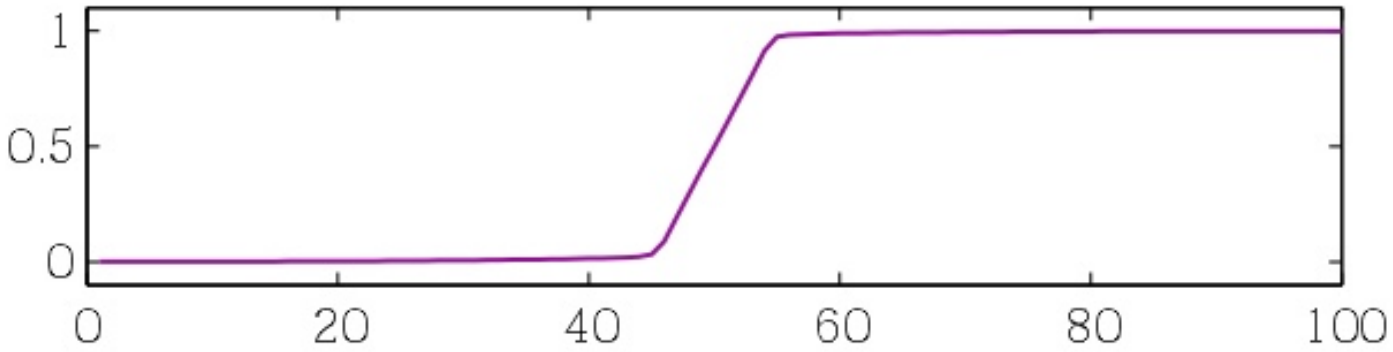

Input Image
Guidance Image
Joint Bilateral Filter
Input
Intermediate iterations
Previous guidance image
Smoothing
Range weight

Processing

Input Image



Guidance Image



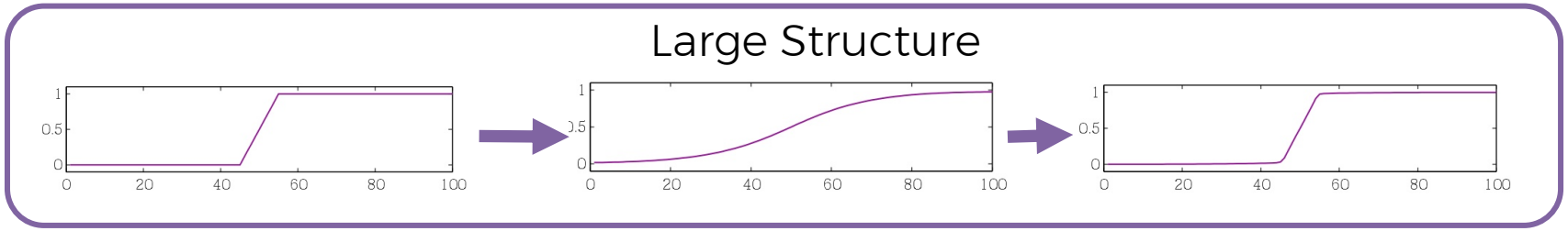
3rd Iteration

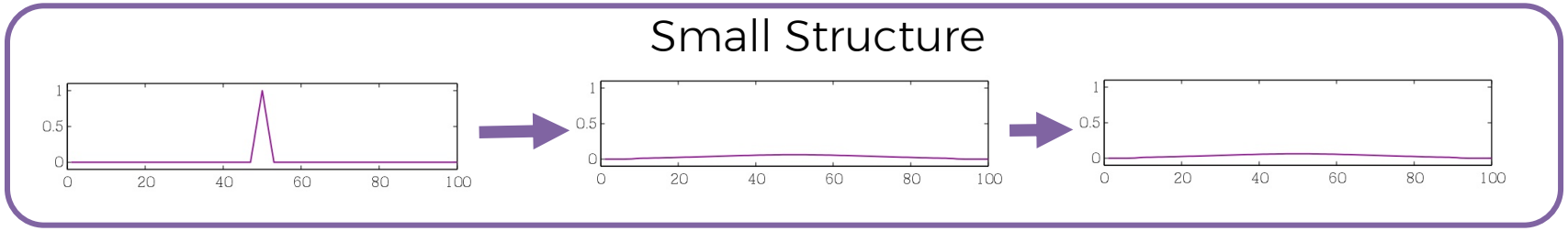
Large Structure

Take-home message

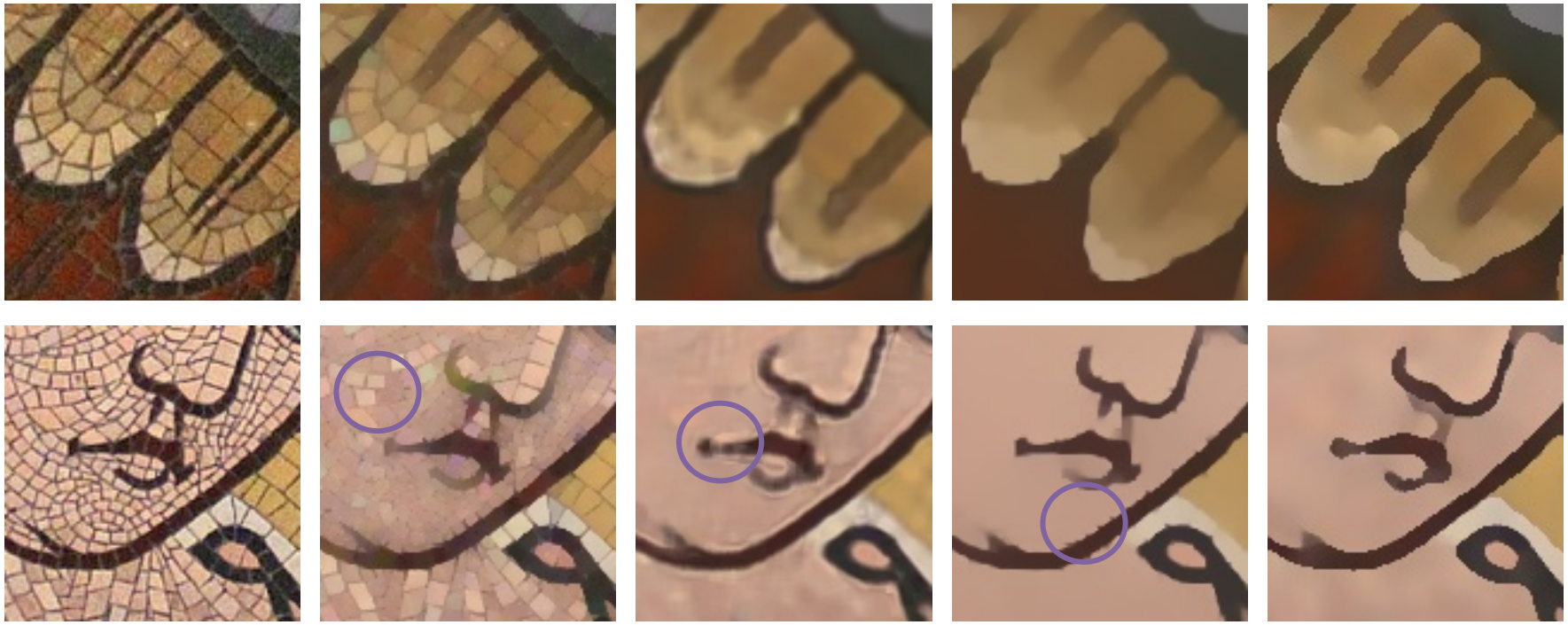
Rolling guidance recovers an edge as long as it still exists in the blurred image after Gaussian smoothing.

Rolling Guidance Filter



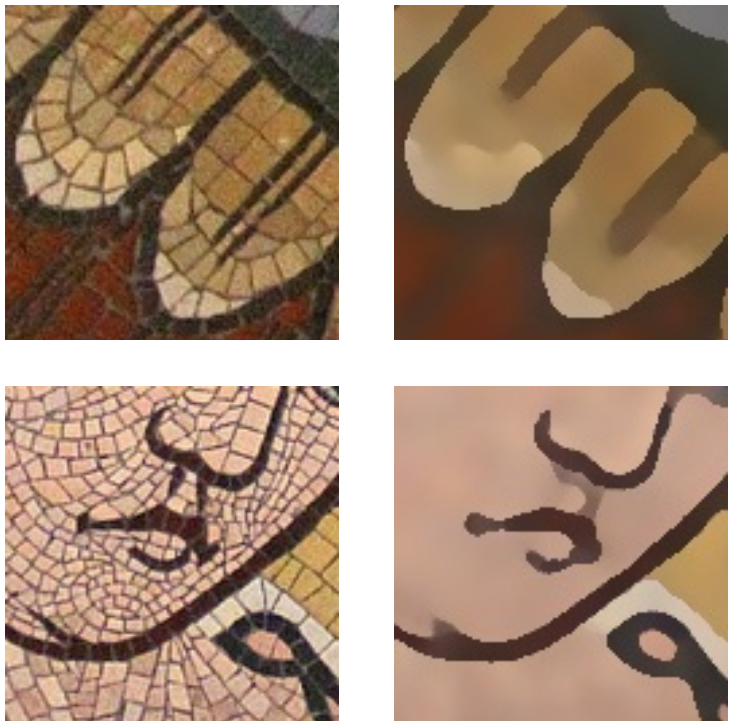


Result Comparison



Input [Subr et al.] [Karacan et al.] [Xu et al.] RGF

Performance Comparison



Input

RGF
2013]

For 4 Megapixel Image

2

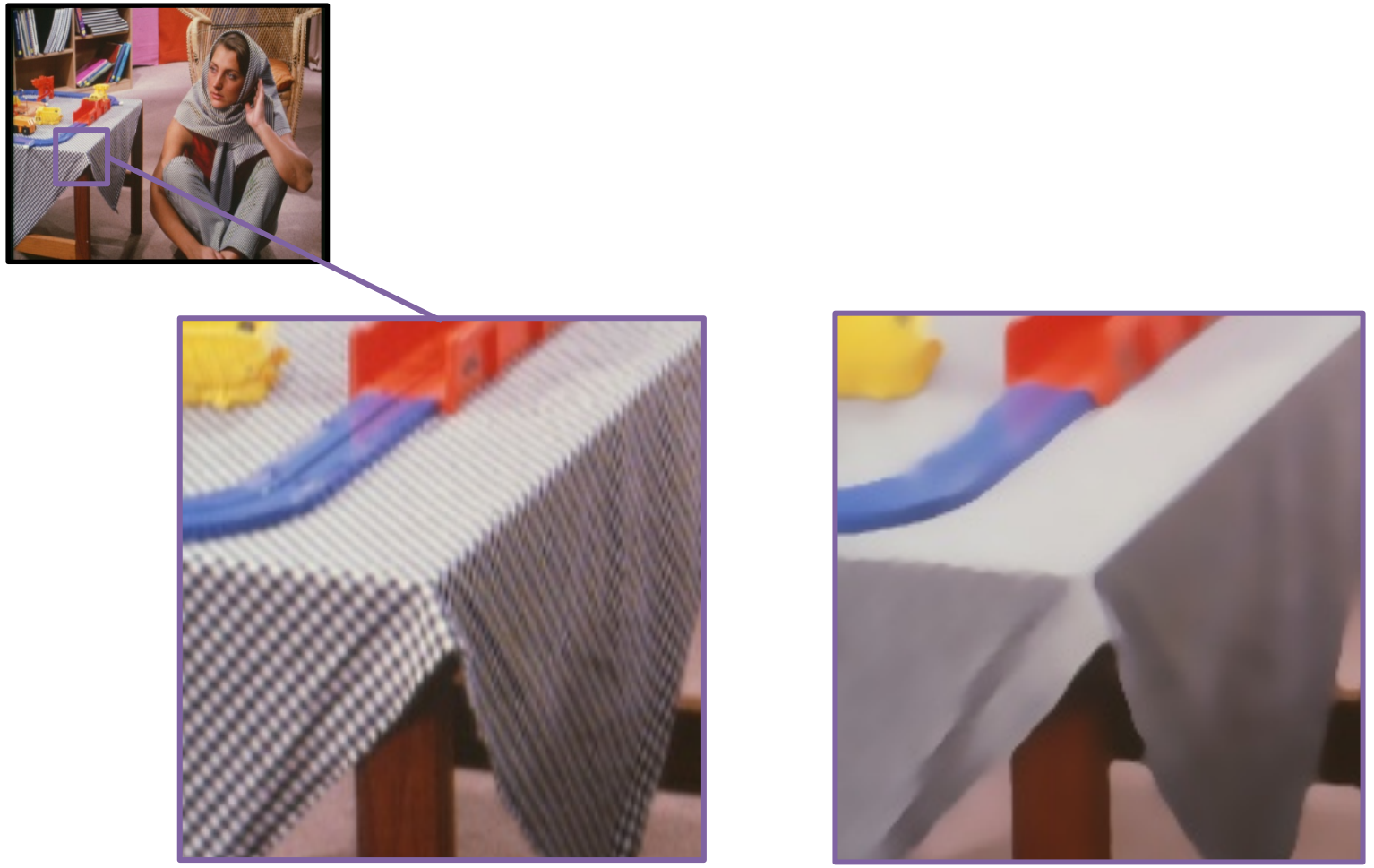
seconds

Performance Comparison

Algorithms	Time (seconds/Megapixel)
Local Extrema [Subr et al., 2009]	95
RTV [Xu et al., 2012]	14
Region Covariance [Karacan et al., 2013]	240
RGF	0.05(Real-time)

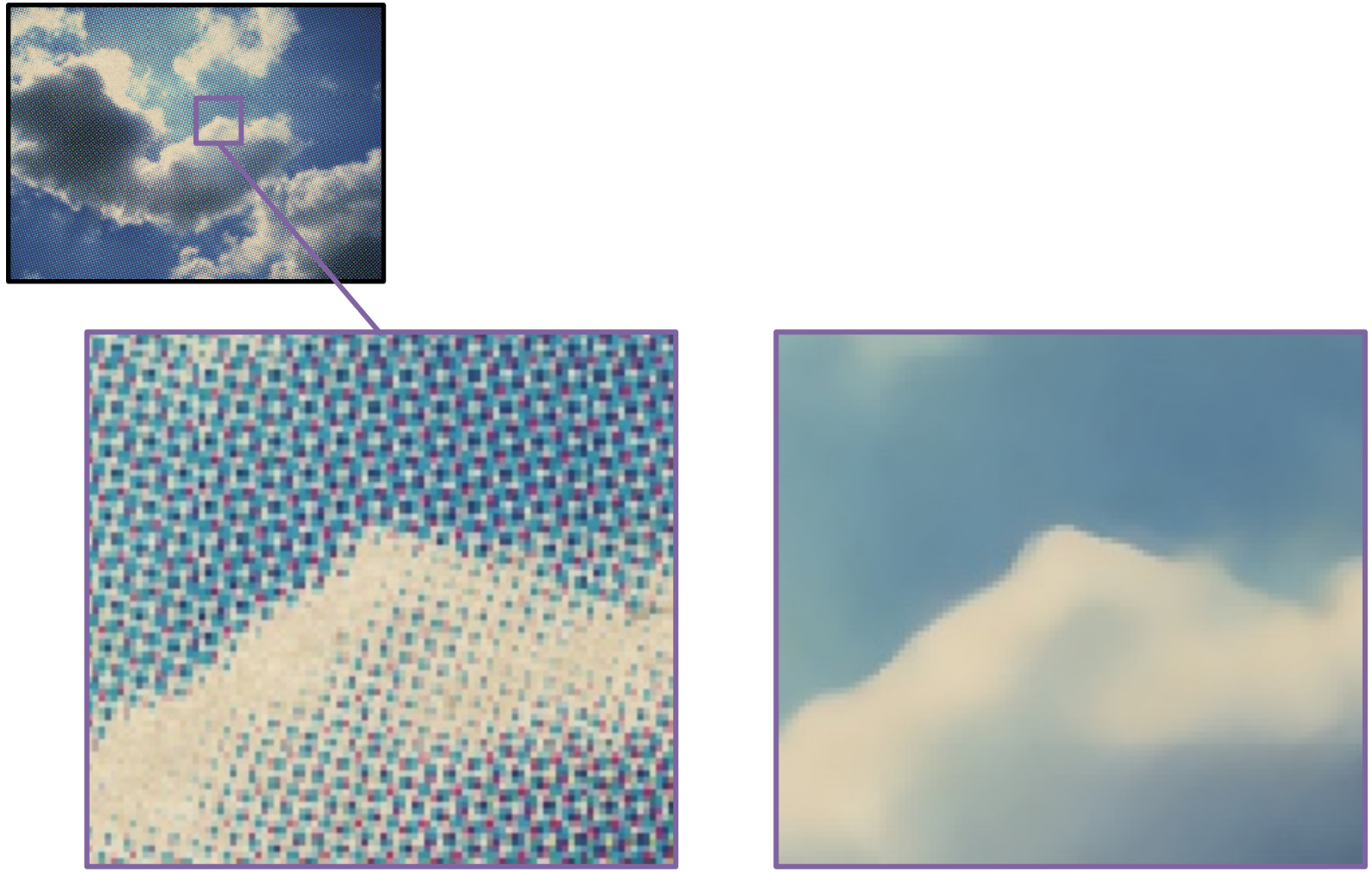
Texture Removal

Texture Removal



Halftone Image

Halftone Image

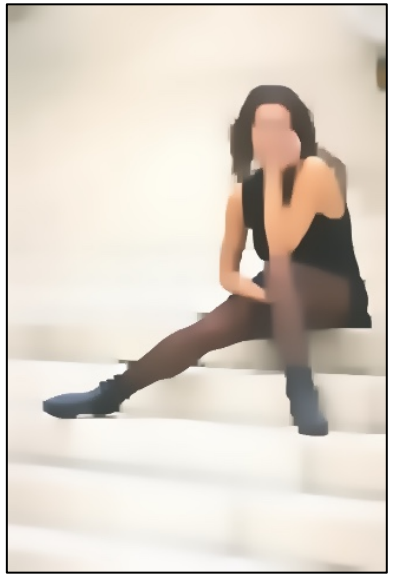


Boundary detection

Input

Boundary Detection

Filtered Input



Boundary Detection

Multi-Scale Filtering

= 30

determine the scale.

Limitations

- Sharp corners could be rounded
  - It is because sharp corner presents high frequency change.
  - In other words, sharp corners are small-scale structures.