CMP717
Image Processing

Nonlinear filtering, Active Contours, Variational Segmentation Models

Erkut Erdem
Hacettepe University
Computer Vision Lab (HUCVL)

Review - Linear Diffusion

- Let $f(x)$ denote a grayscale (noisy) input image and $u(x, t)$ be initialized with $u(x,0) = u^0(x) = f(x)$.
- The linear diffusion process can be defined by the equation:

$$\frac{\partial u}{\partial t} = \nabla \cdot (\nabla u) = \nabla^2 u$$

where $\nabla \cdot$ denotes the divergence operator. Thus,

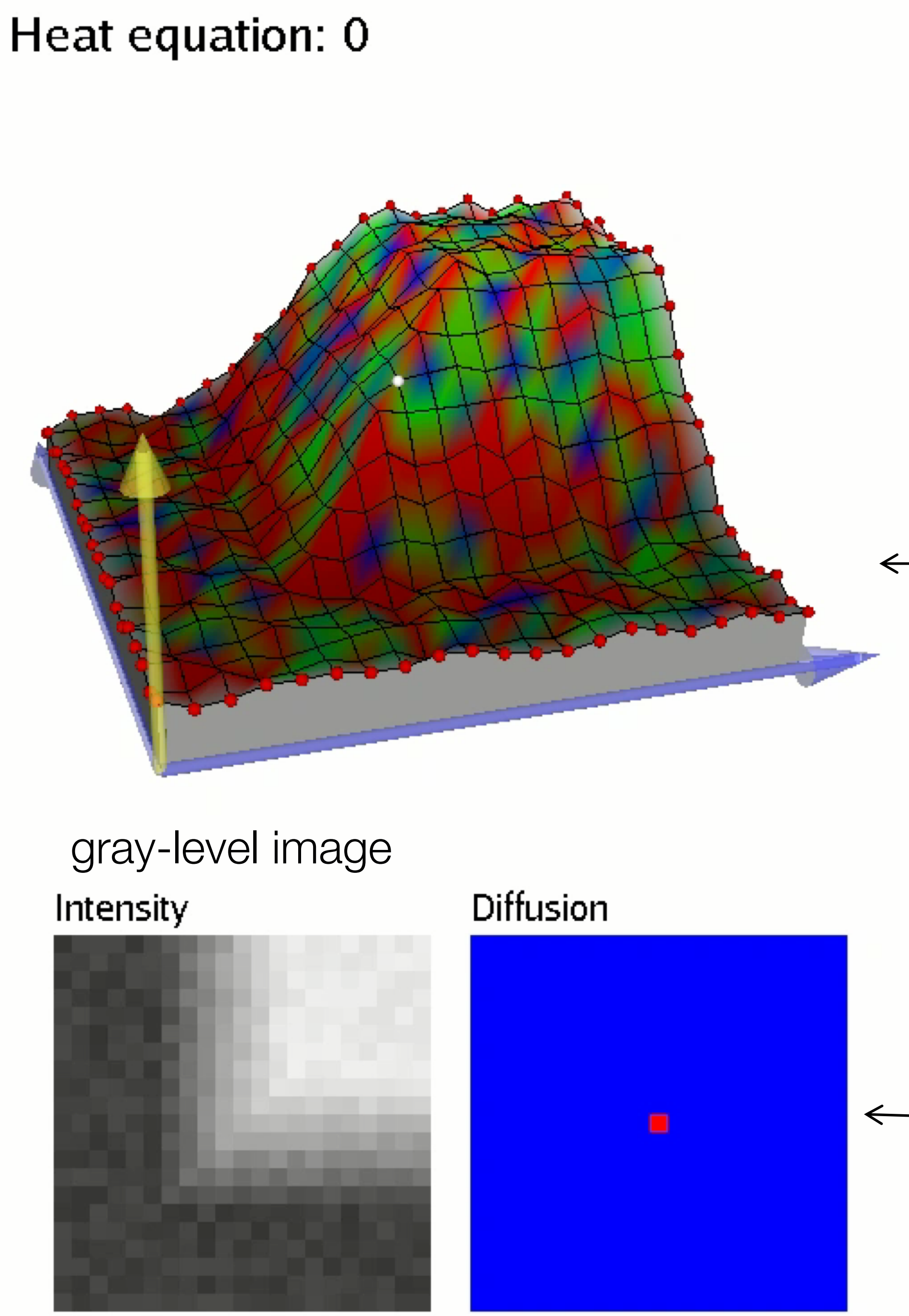
$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

Review - Linear Diffusion (cont'd.

$$\frac{\partial u}{\partial t} = \nabla \cdot (\nabla u) = \nabla^2$$

- Evolving images become more and more simplified

- Diffusion process removes the image structures at finer scales.



red: active areas
blue: inactive area

influence of the central pixel on the other pixels (red: high, blue: low)

Credit: S. Paris

Review - Linear Diffusion and Gaussian Filtering

- Solution of the linear diffusion can be explicitly estimated as:

$$u(x,T) = \left(G_{\sqrt{2T}} * f\right)(x)$$

with

$$G_\sigma(x) = \frac{1}{2\pi\sigma^2} exp\left(-\frac{|x|^2}{2\sigma^2}\right)$$

- Solution of the linear diffusion equation is equivalent to a proper convolution of the input image with the Gaussian kernel $G_\sigma(x)$ with standard deviation $\sigma = \sqrt{2T}$
- The higher the value of *T*, the higher the value of $\boldsymbol{\sigma}$, and the more smooth the image becomes.

Review - Numerical Implementation

- Original model:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

- Space discrete version:

$$\frac{du_{i,j}}{dt} = u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{i,j}$$

- Space-time discrete version:

$$\frac{u_{i,j}^{k+1} - u_{i,j}^{k}}{\Delta t} = u_{i+1,j}^{k} + u_{i-1,j}^{k} + u_{i,j+1}^{k} + u_{i,j-1}^{k} - 4u_{i,j}^{k}$$

homogeneous Neumann boundary condition along the image boundary

$\Delta t \leq 0.25$ is required for numerical stability

Variational interpretation of heat diffusion

- Cost functional:

$$\begin{aligned} E[u] &= \iint_\Omega \|\nabla u\|^2 dxdy \\ &= \iint_\Omega \left(u_x^2 + u_y^2\right) dxdy \end{aligned}$$

- Euler-Lagrange:

$$\begin{aligned} \frac{\delta E}{\delta u} &= \frac{\partial E}{\partial u} - \frac{\partial}{\partial x}\left(\frac{\partial E}{\partial u_x}\right) - \frac{\partial}{\partial y}\left(\frac{\partial E}{\partial u_y}\right) \\ &= -2\frac{\partial u_x}{\partial x} - 2\frac{\partial u_y}{\partial y} \\ &= -2(u_{xx} + u_{yy}) \end{aligned}$$

- Heat diffusion: modifies temperature to decrease E quickly

Slide credit: I. Kokkinos

Today

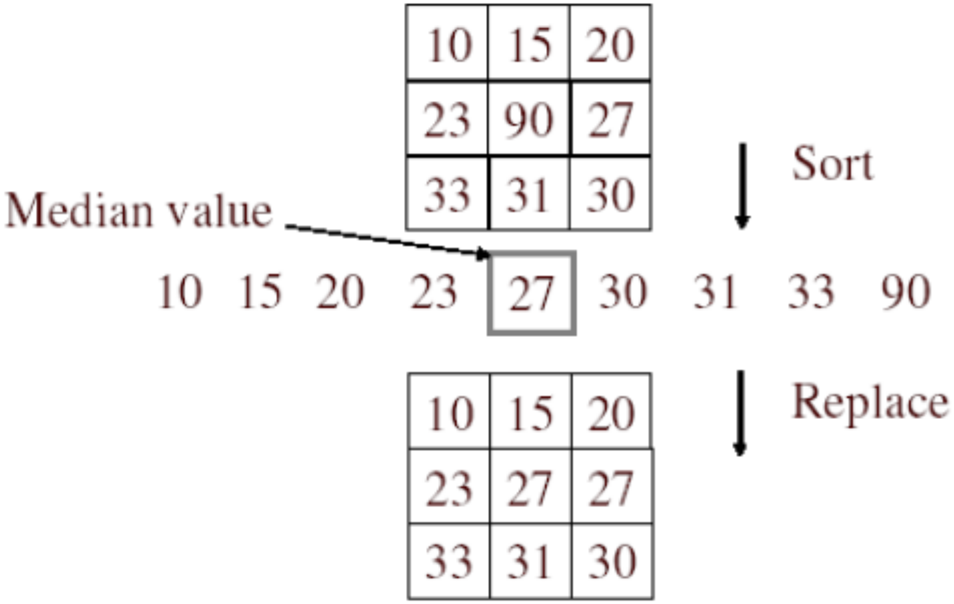
- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

Median filters

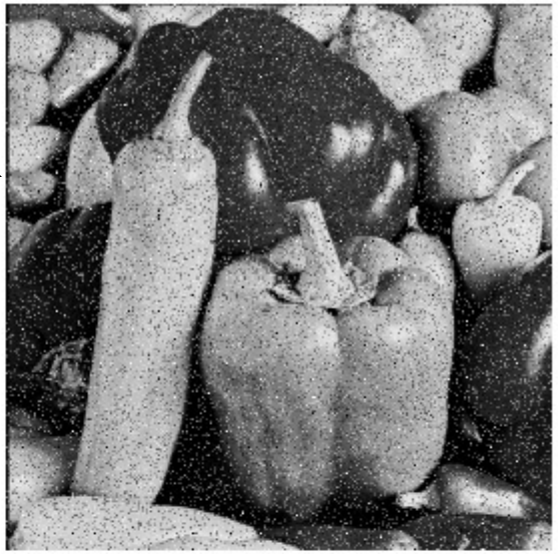
- A Median Filter operates over a window by selecting the median intensity in the window.
- What advantage does a median filter have over a mean filter?
- Is a median filter a kind of convolution?
- No new pixel values introduced
- Removes spikes:
  good for impulse, salt & pepper noise



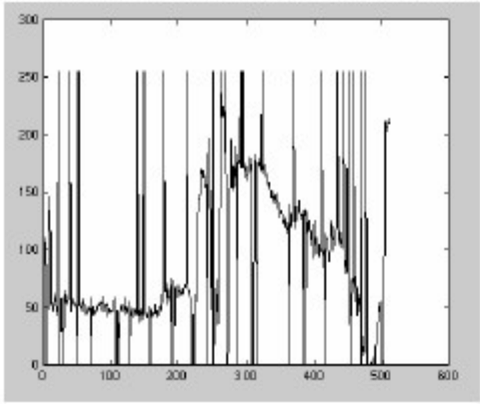
adapted from: S. Seitz, K. Grauman

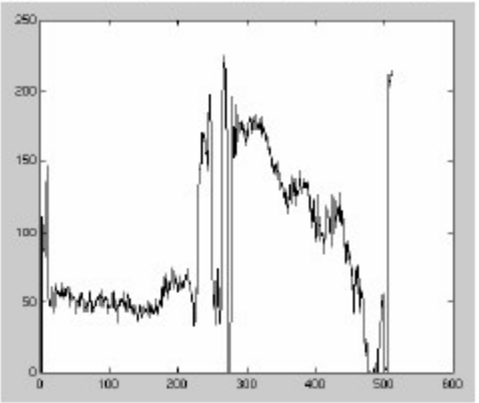
Median filters

Salt and pepper noise



Median filtered





- Robustness to outliers
- Median filter is edge preserving

Plots of a row of the image

Matlab: `output im = medfilt2(im, [h w]);`

Slide credit: M. Hebert

Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

Perona-Malik Type Nonlinear Diffusion

- earliest nonlinear diffusion model for image smoothing
- called *anisotropic diffusion* by Perona and Malik.
- a scalar-valued diffusivity

Original noisy image

Perona-Malik Diffusion

Perona-Malik Type Nonlinear Diffusion

- The Perona-Malik equation is:

$$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u|)\nabla u)$$

  with homogeneous Neumann boundary conditions and the initial condition $u0(x) = f(x)$, f denoting the input image.

- Constant diffusion coefficient of linear equation is replaced with a smooth non-increasing diffusivity function g satisfying
  - $g(0) = 1$,
  - $g(s) \geq 0$,
  - $\lim_{s \to \infty} g(s) = 0$
- Diffusivities become variable in both space and time (image dependent).

Perona-Malik Type Nonlinear Diffusion

- The Perona-Malik equation: $$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u|)\nabla u)$$

- Two different choices for the diffusivity function:

(1) $$g(s) \quad = \quad \frac{1}{1 + s^2/\lambda^2}$$

(2) $$g(s) \quad = \quad e^{-\frac{s^2}{\lambda^2}}$$

- λ corresponds to a contrast parameter.
- What is the effect of the parameter λ?

1D Analysis of Perona-Malik Diffusion

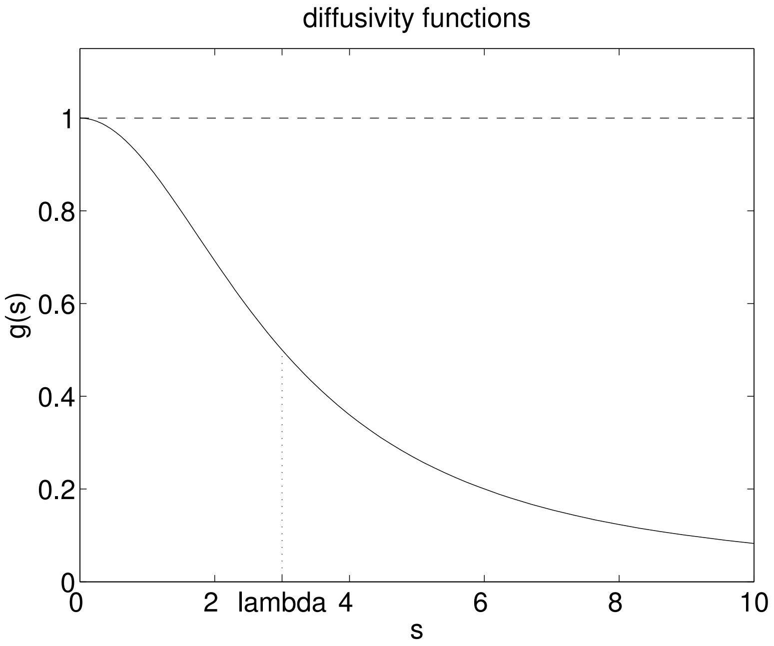
- 1D version to demonstrate the role of the contrast parameter
- For 1D case, the Perona-Malik equation is as follows:

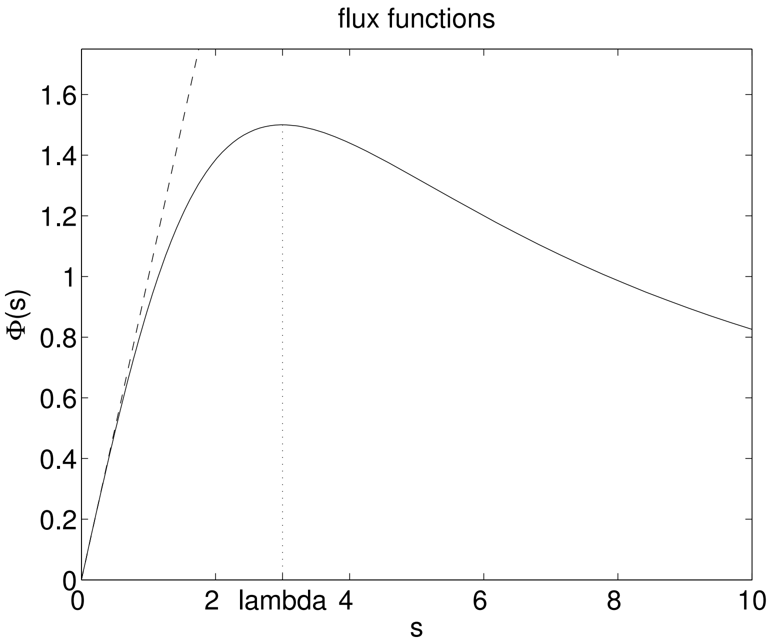
$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x} \underbrace{\left(g(|u_x|)u_x\right)}_{\Phi(u_x)} = \Phi'(u_x)u_{xx}$$

with $g(|u_x|) = \frac{1}{1+|u_x|^2/\lambda^2}$ or $g(|u_x|) = e^{-\frac{|u_x|^2}{\lambda^2}}$

1D Analysis of Perona-Malik Diffusion

- Diffusivities and the corresponding flux functions for the linear diffusion (*plotted in dashed line*) and the Perona-Malik type nonlinear diffusion (*plotted in solid line*).

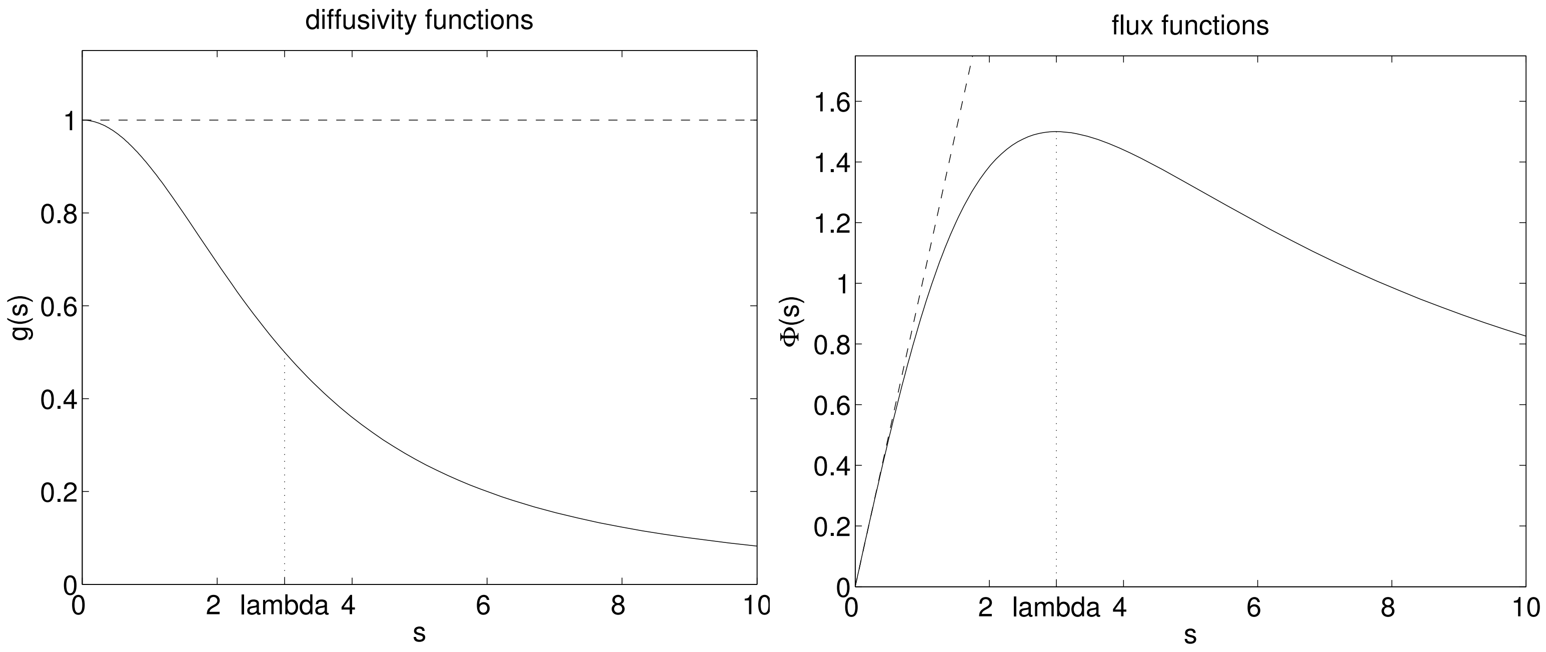




$$g(s) = \frac{1}{1+s^2/\lambda^2}$$

$$\lambda = 3$$

1D Analysis of Perona-Malik Diffusion

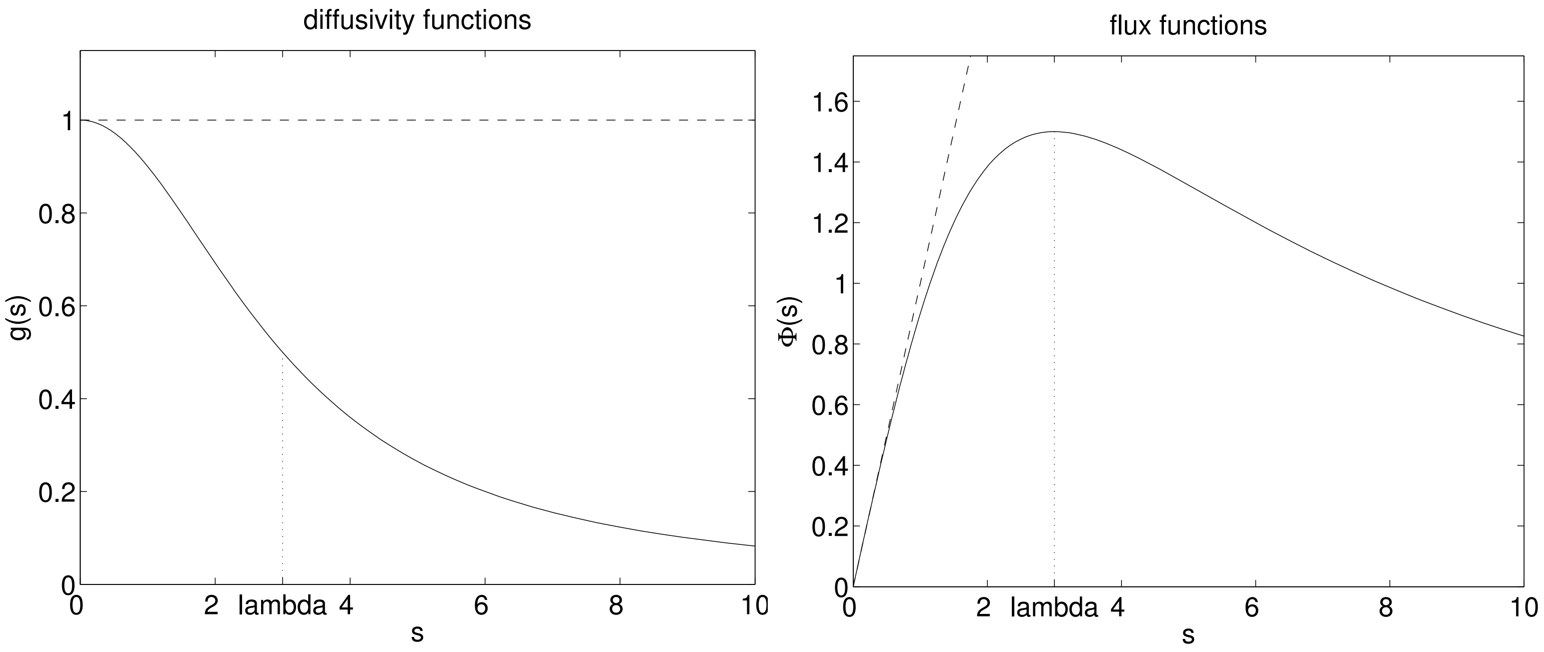


$$g(s) = \frac{1}{1+s^2/\lambda^2}$$

$$\lambda = 3$$

- For linear diffusion the diffusivity is constant ($g(s) = 1$), which results in a linearly increasing flux function.
- For linear diffusion all points, including the discontinuities, are smoothed equally.

1D Analysis of Perona-Malik Diffusion

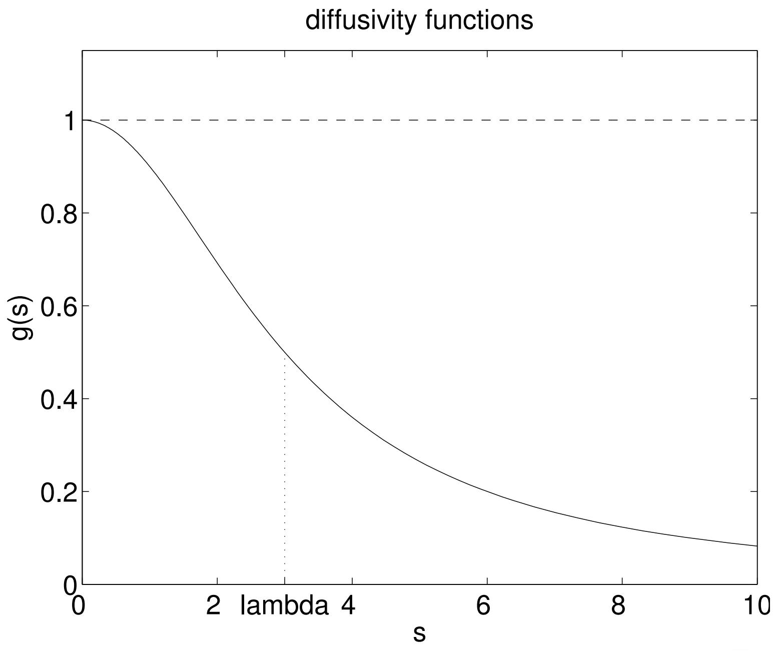


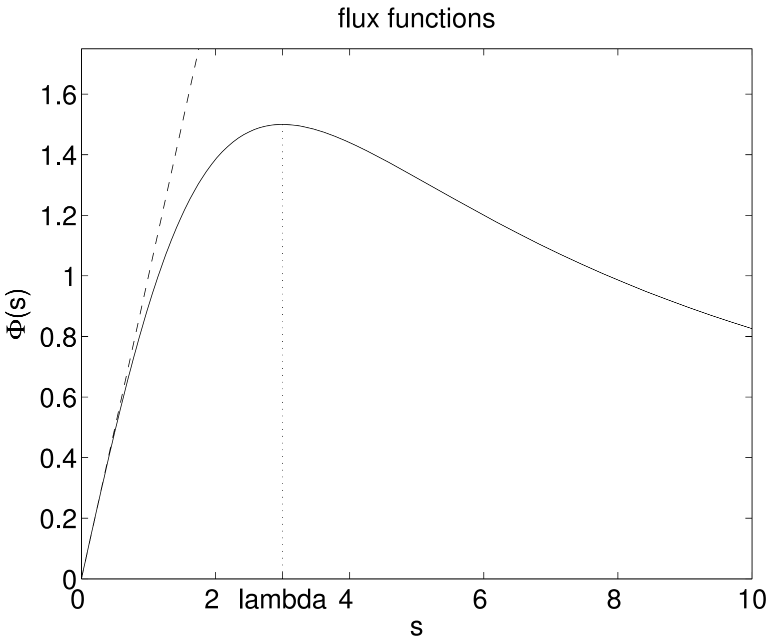
$g(s) = \frac{1}{1+s^2/\lambda^2}$

$\lambda = 3$

- Diffusivity is variable and decreases as $|u_x|$ increases.
- Decay in diffusivity is particularly rapid after the contrast parameter λ.
- Two different behaviors in the diffusion process

1D Analysis of Perona-Malik Diffusion





$g(s) = \frac{1}{1+s^2/\lambda^2}$

$\lambda = 3$

$$\frac{\partial u}{\partial t} = \Phi'(u_x)u_{xx}$$

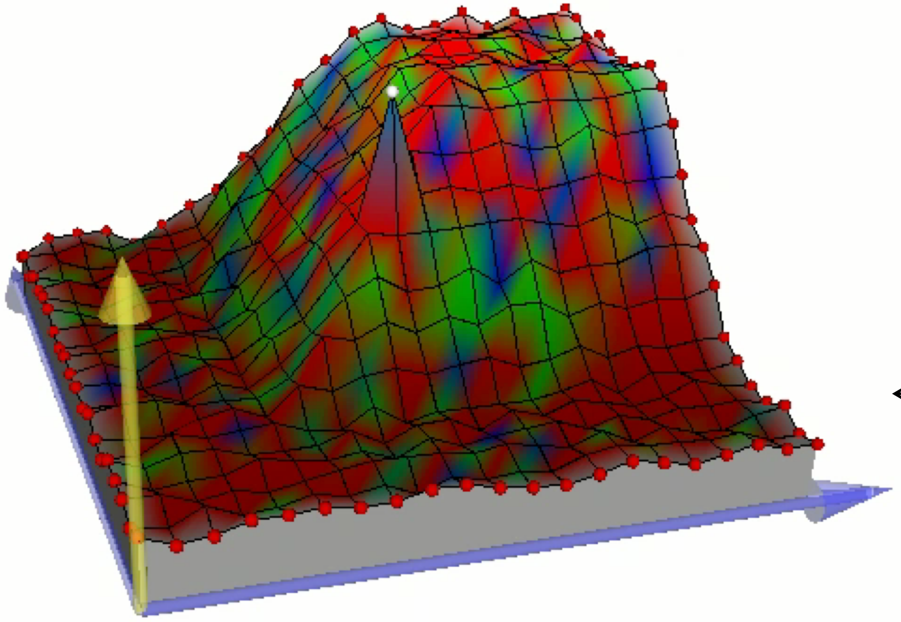
- For the points where $|u_x| < \lambda$, $\Phi'(u_x) > 0$we have lost in the material.
- For the points where $|u_x| > \lambda$, on the contrary, $\Phi'(u_x) < 0$ which generates an enhancement in the material.

Perona-Malik Type Nonlinear Diffusion

- In 2D case, diffusivities are reduced at the image locations where $|\nabla u|^2$ is large ($|\nabla u|^2$: a measure of edge likelihood)
- Amount of smoothing is low along image edges.
- Contrast parameter λ specifies a measure that determines which edge points are to be preserved or blurred during the diffusion process.
- Even edges can be sharpened due to the local backward diffusion behavior as discussed for the 1D case.
- Since the backward diffusion is a well-known ill-posed process, this may cause an instability, the so-called *staircasing effect.*

Perona Malik: 0

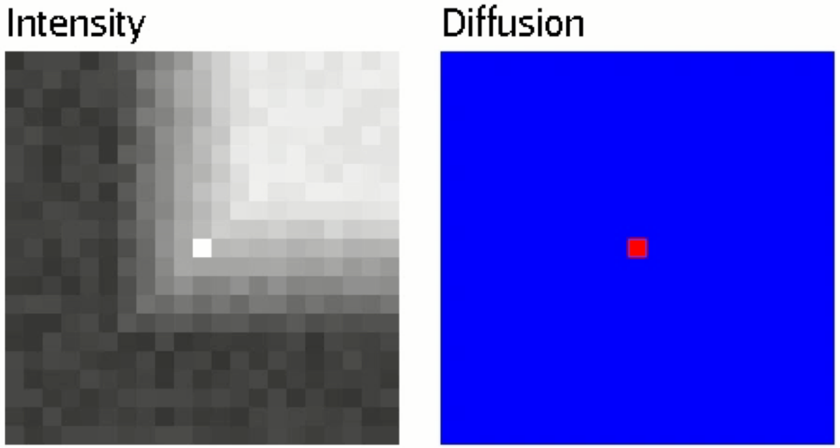
Perona-Malik (cont'd.)



$$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u|)\nabla u)$$

red: active areas
blue: inactive area

gray-level image



influence of the central pixel
on the other pixels
(red: high, blue: low)

Credit: S. Paris

Staircasing Effect

- Due to backward diffusion, a piece-wise smooth region in the original image evolves into many unintuitive piecewise constant regions.

Original noisy image

Perona-Malik Diffusion

- Solution: Use pre-filtered (regularized) gradients in diffusivity computations

Regularized Perona-Malik Model

- Replacing the diffusivities $g(|\nabla u|)$ with the regularized ones $g(|\nabla u_\sigma|)$ leads to the following equation:

$$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u_\sigma|)\nabla u)$$

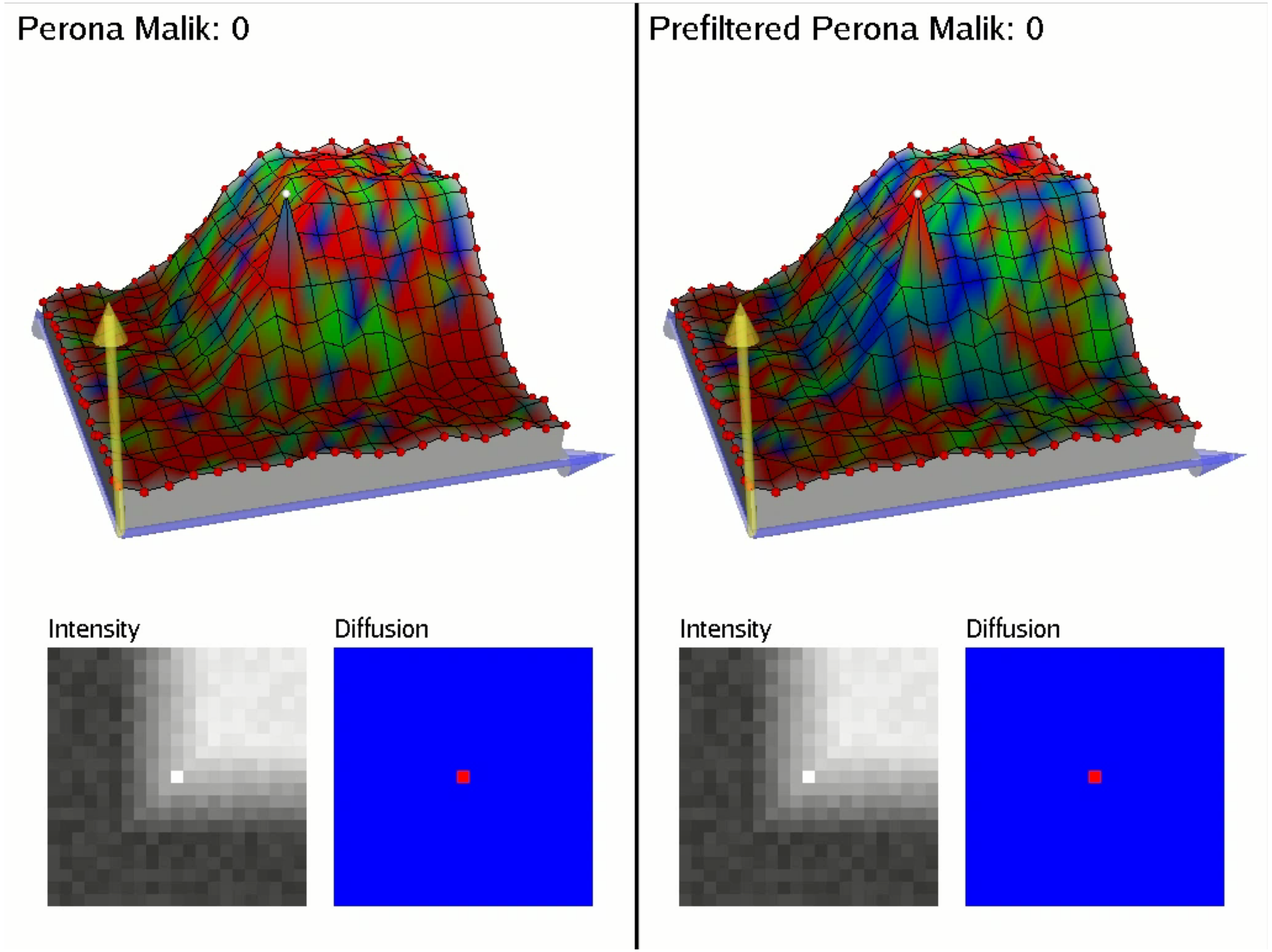
where $u_\sigma = G_\sigma * u$ represents a Gaussian-smoothed version of the image.

Original noisy image

Perona-Malik Diffusion

Regularized Perona-Malik Diffusion

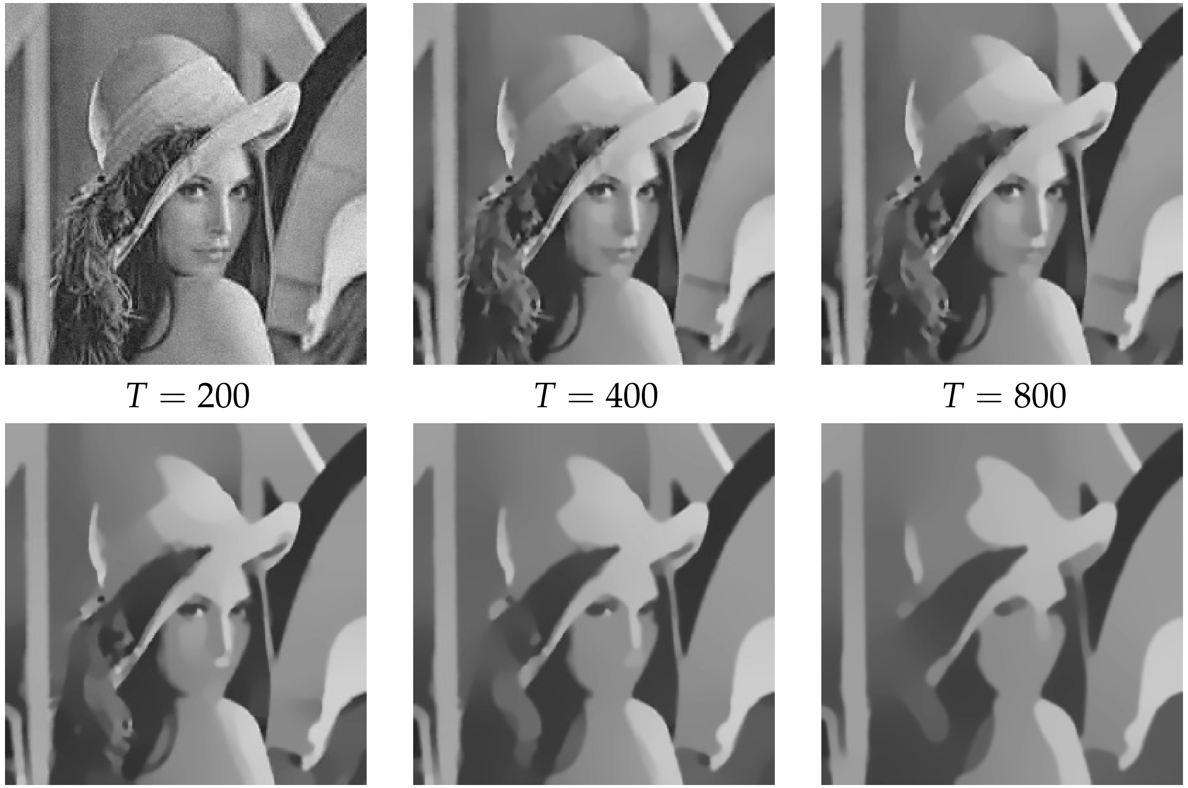
Regularized Perona-Malik (cont'd.)

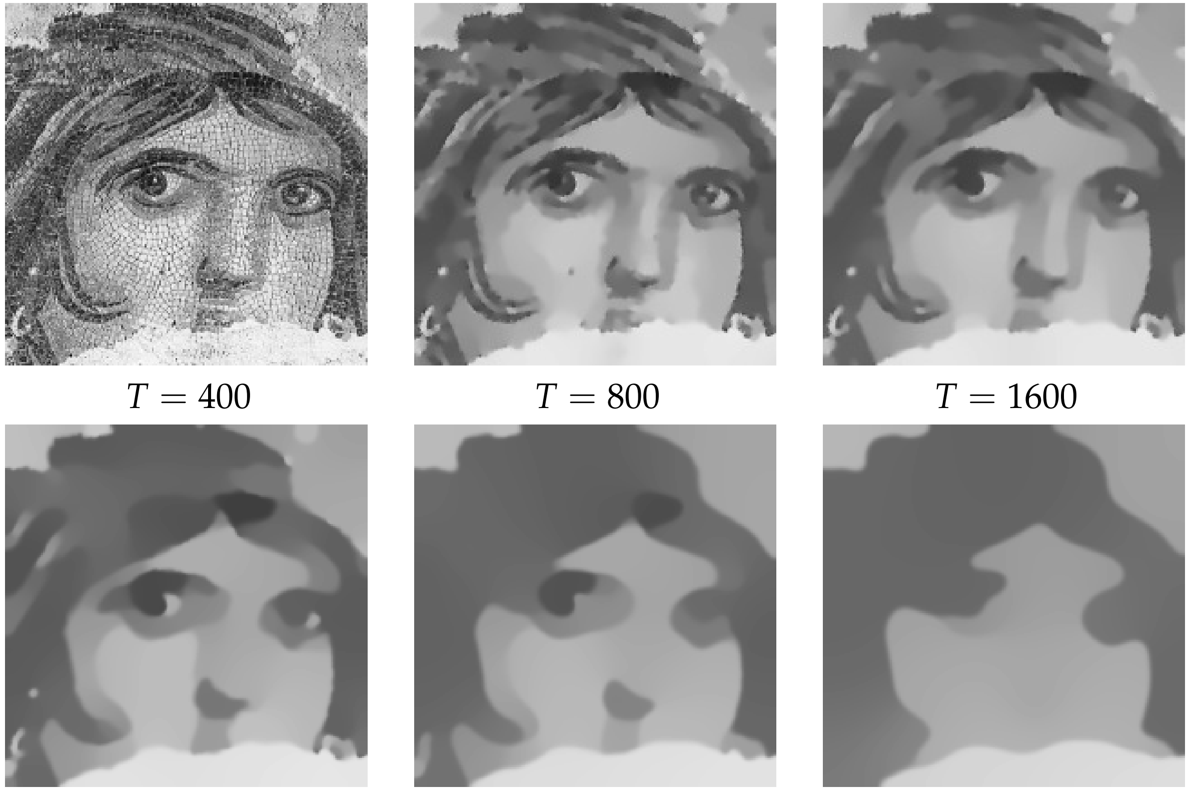


Credit: S. Paris

Regularized Perona-Malik Model

- Smoothing process diminishes noise while retaining or enhancing edges





$(\lambda = 1, \sigma = 1)$

Numerical Implementation

$$|\nabla u_{i,j}| = \sqrt{\left(\frac{du_{i,j}}{dx}\right)^2 + \left(\frac{du_{i,j}}{dy}\right)^2}$$

$$\approx \sqrt{\left(\frac{u_{i+1,j} - u_{i-1,j}}{2}\right)^2 + \left(\frac{u_{i,j+1} - u_{i,j-1}}{2}\right)^2}$$

- Original model:

$$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u|)\nabla u)$$

- Space discrete version:

$$\begin{aligned}
\frac{\partial u}{\partial t} &= \frac{\partial}{\partial x}\left(g(|\nabla u|)u_x\right) + \frac{\partial}{\partial y}\left(g(|\nabla u|)u_y\right) \\
\frac{du_{i,j}}{dt} &= g_{i+\frac{1}{2},j} \cdot \left(u_{i+1,j} - u_{i,j}\right) - g_{i-\frac{1}{2},j} \cdot \left(u_{i,j} - u_{i-1,j}\right) \\
&+ g_{i,j+\frac{1}{2}} \cdot \left(u_{i,j+1} - u_{i,j}\right) - g_{i,j-\frac{1}{2}} \cdot \left(u_{i,j} - u_{i,j-1}\right)
\end{aligned}$$

Numerical Implementation

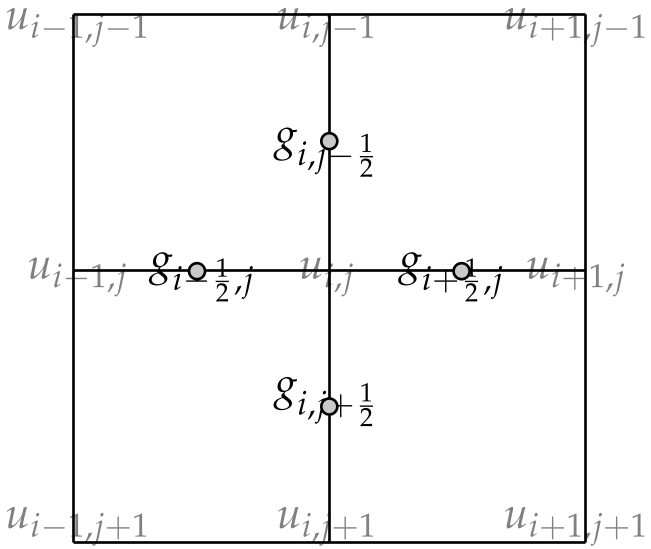
- Space discrete version:

$$\begin{aligned} \frac{du_{i,j}}{dt} &= g_{i+\frac{1}{2},j} \cdot (u_{i+1,j} - u_{i,j}) - g_{i-\frac{1}{2},j} \cdot (u_{i,j} - u_{i-1,j}) \\ &+ g_{i,j+\frac{1}{2}} \cdot (u_{i,j+1} - u_{i,j}) - g_{i,j-\frac{1}{2}} \cdot (u_{i,j} - u_{i,j-1}) \end{aligned}$$

- This discretization scheme requires the diffusivities to be estimated at mid-pixel points.
- computed by taking averages of the diffusivities over neighboring pixels:

$$g_{i\pm\frac{1}{2},j} = \frac{g_{i\pm1,j} + g_{i,j}}{2}$$

$$g_{i,j\pm\frac{1}{2}} = \frac{g_{i,j\pm1} + g_{i,j}}{2}$$



Numerical Implementation

- Space discrete version:

$$
\begin{aligned}
\frac{du_{i,j}}{dt} &= g_{i+\frac{1}{2},j} \cdot \left(u_{i+1,j} - u_{i,j}\right) - g_{i-\frac{1}{2},j} \cdot \left(u_{i,j} - u_{i-1,j}\right) \\
&+ g_{i,j+\frac{1}{2}} \cdot \left(u_{i,j+1} - u_{i,j}\right) - g_{i,j-\frac{1}{2}} \cdot \left(u_{i,j} - u_{i,j-1}\right)
\end{aligned}
$$

- Space-time discrete version:

$$
\begin{aligned}
\frac{u_{i,j}^{k+1} - u_{i,j}^{k}}{\Delta t} &= g_{i+\frac{1}{2},j}^{k} \cdot u_{i+1,j}^{k} + g_{i-\frac{1}{2},j}^{k} \cdot u_{i-1,j}^{k} + g_{i,j+\frac{1}{2}}^{k} \cdot u_{i,j+1}^{k} + g_{i,j-\frac{1}{2}}^{k} \cdot u_{i,j-1}^{k} \\
&- \left(g_{i+\frac{1}{2},j}^{k} + g_{i-\frac{1}{2},j}^{k} + g_{i,j+\frac{1}{2}}^{k} + g_{i,j-\frac{1}{2}}^{k}\right) \cdot u_{i,j}^{k}
\end{aligned}
$$

homogeneous Neumann boundary condition along the image boundary

$\Delta t \leq 0.25$ is required for numerical stability

Extension to vectorial images

- Extension of nonlinear diffusion to vectorial images:

$$\boldsymbol{u} = (u_1, u_2, \ldots, u_N)$$

$$\frac{\partial u}{\partial t} = \mathrm{div}\left(g(\|\nabla u\|)\nabla u\right)$$

generalization

$$\boxed{\frac{\partial u_i}{\partial t} = \mathrm{div}\left(g(\|\nabla \boldsymbol{u}\|)\nabla u_i\right),\ i = 1, .., N}$$

where: $\|\nabla \boldsymbol{u}\| = \sqrt{\sum_{i=1}^{N} \|\nabla u_i\|^2}$

Slide credit: I. Kokkinos

Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

Total Variation (TV) Regularization

- Rudin et al. (1992): image restoration as minimization of the total variation (TV) of a given image.
- The Total Variation (TV) regularization model is generally defined as:

$$E_{TV}(u) = \int_{\Omega} \left(\frac{1}{2}(u - f)^2 + \alpha |\nabla u| \right) dx$$

  - $\Omega \subset R^2$ is connected, bounded, open subset representing the image domain,
  - f is an image defined on Ω,
  - u is the smooth approximation of f,
  - $\alpha > 0$ is a scalar.

Total Variation (TV) Regularization

- The Total Variation (TV) regularization model:

$$E_{TV}(u) = \int_{\Omega} \left(\frac{1}{2}(u-f)^2 + \alpha|\nabla u| \right) dx$$

- The gradient descent equation for Equation (10) is defined by:

$$\frac{\partial u}{\partial t} = \nabla \cdot \left(\frac{\nabla u}{|\nabla u|} \right) - \frac{1}{\alpha}(u-f); \quad \left.\frac{\partial u}{\partial n}\right|_{\partial\Omega} = 0$$

- The value of α specifies the relative importance of the fidelity term.
- It can be interpreted as a scale parameter that determines the level of smoothing.

Sample TV Restoration results

$$E_{TV}(u) = \int_{\Omega} \left(\frac{1}{2}(u - f)^2 + \alpha |\nabla u| \right) dx$$

$\alpha = 50$

$\alpha = 100$

$\alpha = 200$

- The value of α specifies the relative importance of the fidelity term and thus the level of smoothing.

TV Regularization

- Observed image *f* was assumed to be degraded by additive Gaussian noise with zero mean and known variance σ^2.
- To restore a given image, solve the following constrained optimization problem:

$$\min_u \int_\Omega |\nabla u| dx$$

subject to

$$\int_\Omega (u - f)^2 dx = \sigma^2$$

- $\frac{1}{\alpha}$ can be considered as a Lagrange multiplier.

TV Regularization and TV Flow

- TV regularization can be associated with a nonlinear diffusion filter, the so-called *TV flow*.
- Ignoring the fidelity term in the TV regularization model leads to the PDE:

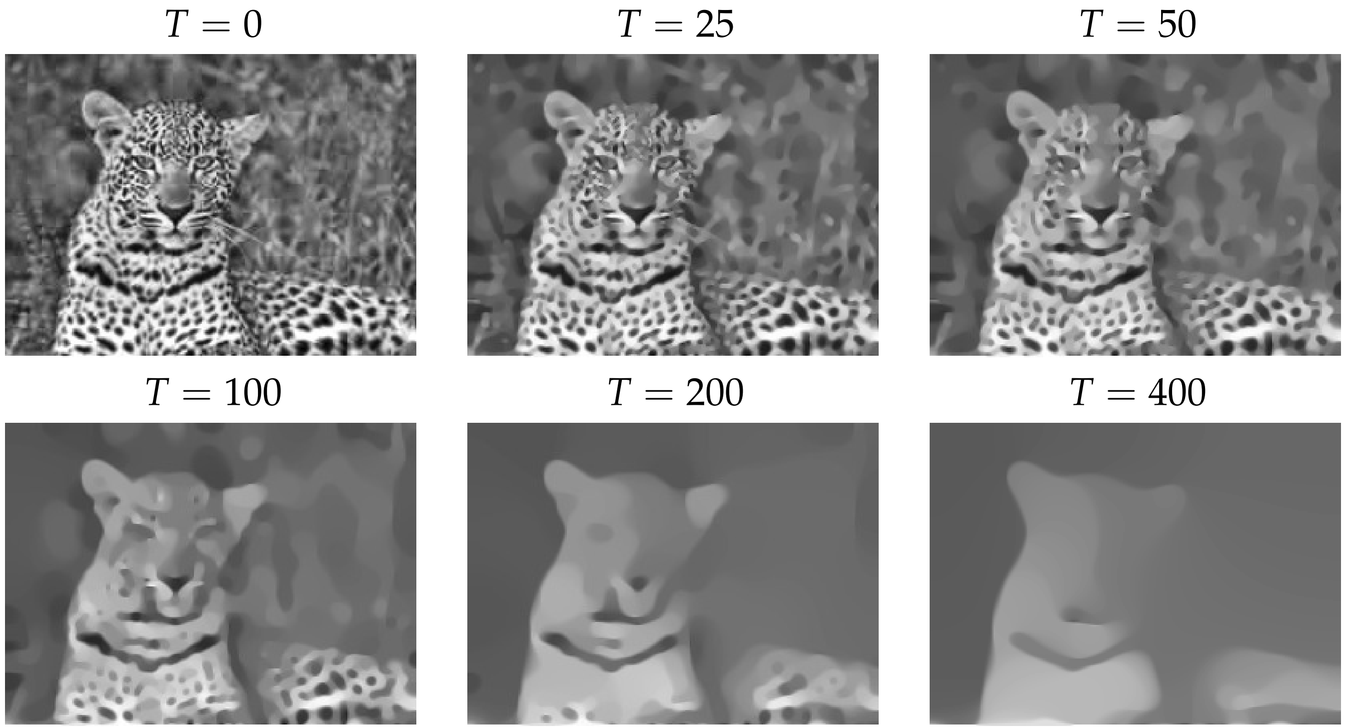
$$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u|)\nabla u)$$

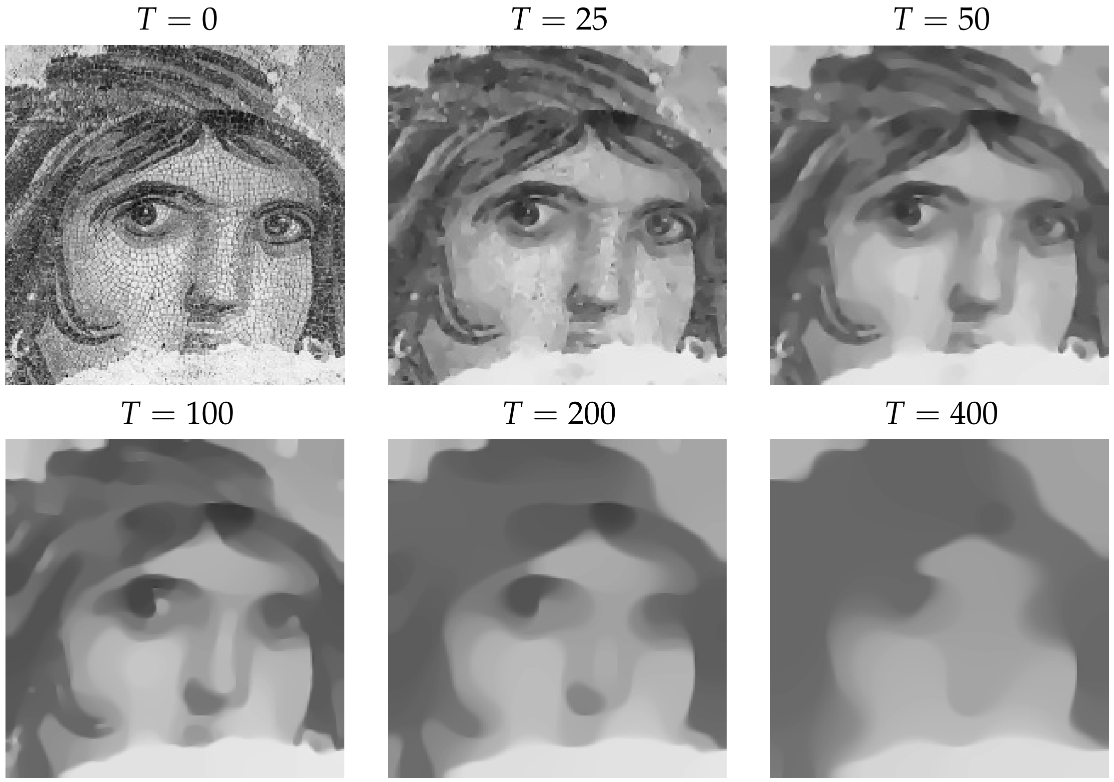
with $u^0 = f$ and the diffusivity function $g(|\nabla u|) = \frac{1}{|\nabla u|}$

- Notice that this diffusivity function has no additional contrast parameter as compared with the Perona-Malik diffusivities.

Sample TV Flow results

- Corresponding smoothing process yields segmentation-like, piecewise constant images.





Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

Mumford-Shah (MS) Segmentation Model

- Mumford & Shah, Comm. Pure Appl. Math., 1989
- Segmentation is formalized as a functional minimization:
  Given an image ***f***, compute a piecewise smooth image ***u*** and
  an edge set Γ

$$E_{MS}(u,\Gamma) = \beta \int_{\Omega} (u-f)^2 dx + \alpha \int_{\Omega\setminus\Gamma} |\nabla u|^2 dx + length(\Gamma)$$

  - $\Omega \subset R^2$ is connected, bounded, open subset representing the image domain,
  - f is an image defined on Ω,
  - $\Gamma \subset \Omega$ is the edge set segmenting Ω,
  - u is the piecewise smooth approximation of f ,
  - $\alpha, \beta > 0$ are the scale space parameters.

Mumford-Shah (MS) Segmentation Model

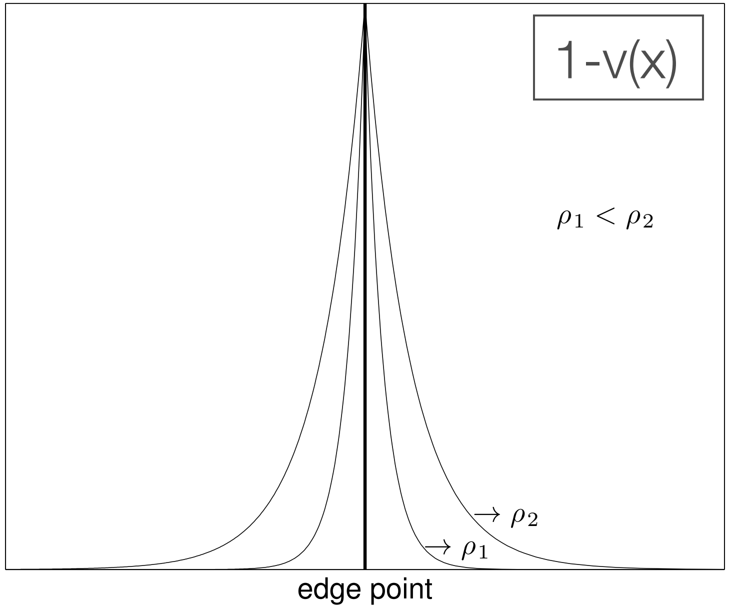
$$E_{MS}(u,\Gamma) = \underbrace{\beta \int_{\Omega} (u-f)^2 dx}_{\text{data fidelity term}} + \underbrace{\alpha \int_{\Omega \setminus \Gamma} |\nabla u|^2 dx + length(\Gamma)}_{\text{regularization or smoothness term}}$$

- Smoothing and edge detection processes work jointly to partition an image into segments.
- Unknown edge set Γ of a lower dimension makes the minimization of the MS model very difficult.
- In literature several approaches for approximating the MS model are suggested.

Ambrosio-Tortorelli (AT) Approximation

$$E_{AT}(u,v) = \int_{\Omega} \left(\beta(u-f)^2 + \alpha(v^2|\nabla u|^2) + \underbrace{\frac{1}{2}\left(\rho|\nabla v|^2 + \frac{(1-v)^2}{\rho}\right)}_{\text{length}(\Gamma)} \right) dx$$

- Unknown edge set Γ is replaced with a continuous function v(x)
  - v ≈ 0 along image edges
  - v grows rapidly towards 1 away from edges
- The function *v* can be interpreted as a blurred version of the edge set.
- The parameter ρ specifies the level of blurring.



Ambrosio-Tortorelli (AT) Approximation

- Solve the following system of coupled PDEs for piecewise smooth image *u* and the edge strength function *v*:

$$
\begin{aligned}
\frac{\partial u}{\partial t} &= \nabla \cdot (v^2 \nabla u) - \frac{\beta}{\alpha}(u - f); \quad \left.\frac{\partial u}{\partial n}\right|_{\partial\Omega} = 0 \\
\frac{\partial v}{\partial t} &= \nabla^2 v - \frac{2\alpha |\nabla u|^2 v}{\rho} - \frac{(v-1)}{\rho^2}; \quad \left.\frac{\partial v}{\partial n}\right|_{\partial\Omega} = 0
\end{aligned}
$$

Ambrosio-Tortorelli (AT) Approximation

f: raw image

u: smooth image

v: edge strength function

Ambrosio-Tortorelli (AT) Approximation

- Keeping v fixed, PDE for the process u minimizes the following convex quadratic functional:

$$\int_{\Omega} \left(\alpha v^2 |\nabla u|^2 + \beta (u - f)^2 \right) dx$$

- Data fidelity term provides a bias that forces u to be close to the original image f .
- In the regularization term, the edge strength function v specifies the boundary points and guides the smoothing accordingly.
- Since $v \approx 0$ along the boundaries, no smoothing is carried out at the boundary points, thus the edges are preserved.

Ambrosio-Tortorelli (AT) Approximation: v process

- Keeping *u* fixed, PDE for the process *v* minimizes the following convex quadratic functional:

$$\frac{\rho}{2}\int_{\Omega}\left(|\nabla v|^2 + \frac{1+2\alpha\rho|\nabla u|^2}{\rho^2}\left(v - \frac{1}{1+2\alpha\rho|\nabla u|^2}\right)^2\right)dx$$

- The function *v* is nothing but a smoothing of $\frac{1}{1+2\alpha\rho|\nabla u|^2}$
- The smoothness term forces some spatial organization by requiring the edges to be smooth.
- Ignoring the smoothness term and letting ρ go to 0, we have

$$v \approx \frac{1}{1+2\alpha\rho|\nabla u|^2}$$

Relating with the Perona-Malik Diffusion

- Replacing *v* with $1/(1+2\alpha\rho|\nabla u|^2)$, PDE for the process *u* can be interpreted as a biased Perona-Malik type nonlinear diffusion:

$$\frac{\partial u}{\partial t} = \nabla \cdot (g(|\nabla u|)\nabla u) - \frac{\beta}{\alpha}(u - f)$$

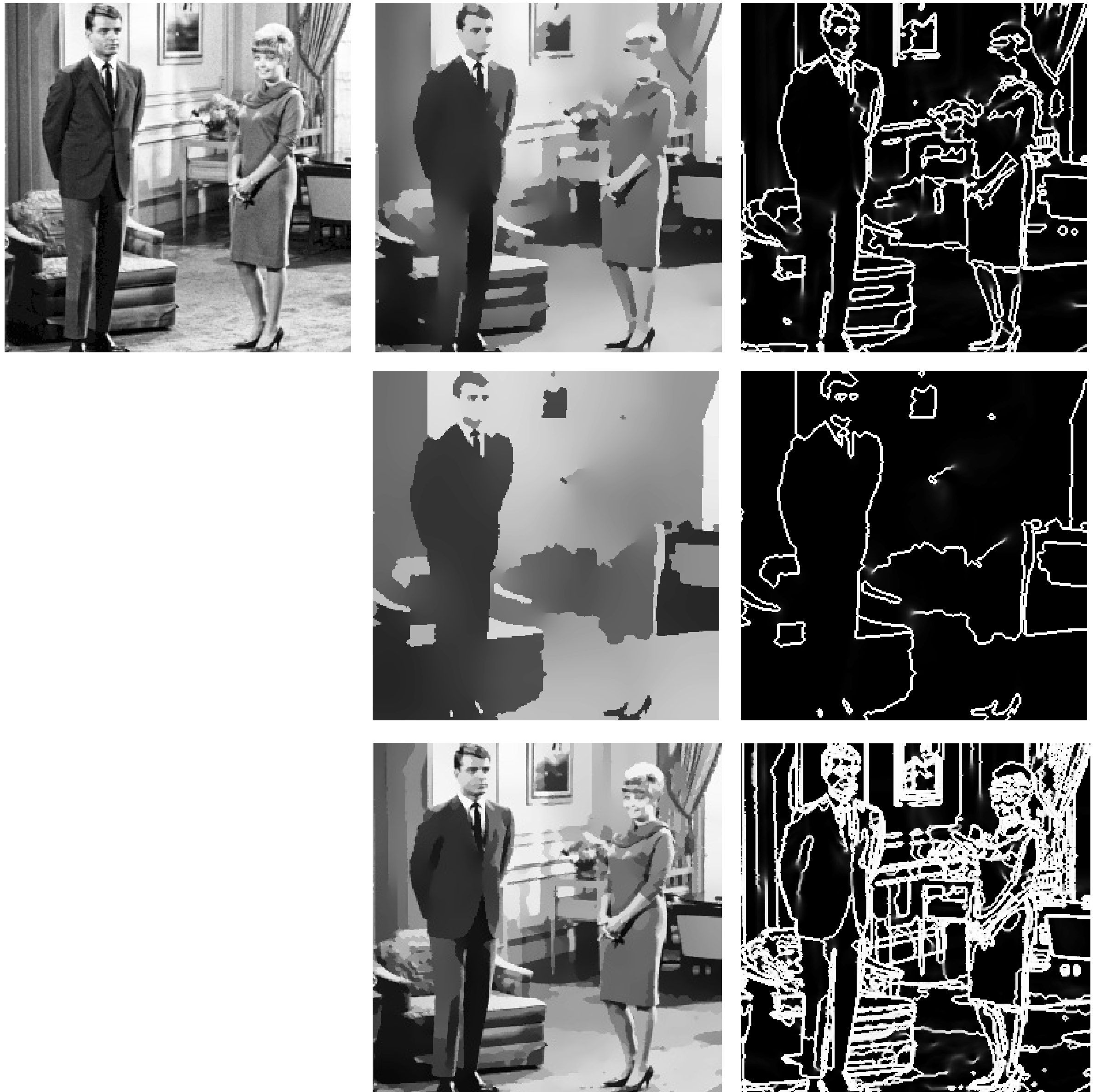
with

$$g(|\nabla u|) = \left(\frac{1}{1+|\nabla u|^2/\lambda^2}\right)^2$$

$$\lambda^2 = 1/\left(2\alpha\rho\right)$$

- $\sqrt{1/\left(2\alpha\rho\right)}$ as a contrast parameter
- Relative importance of the regularization term (scale) depends on the ratio between α and β.

Sample Results of the AT model

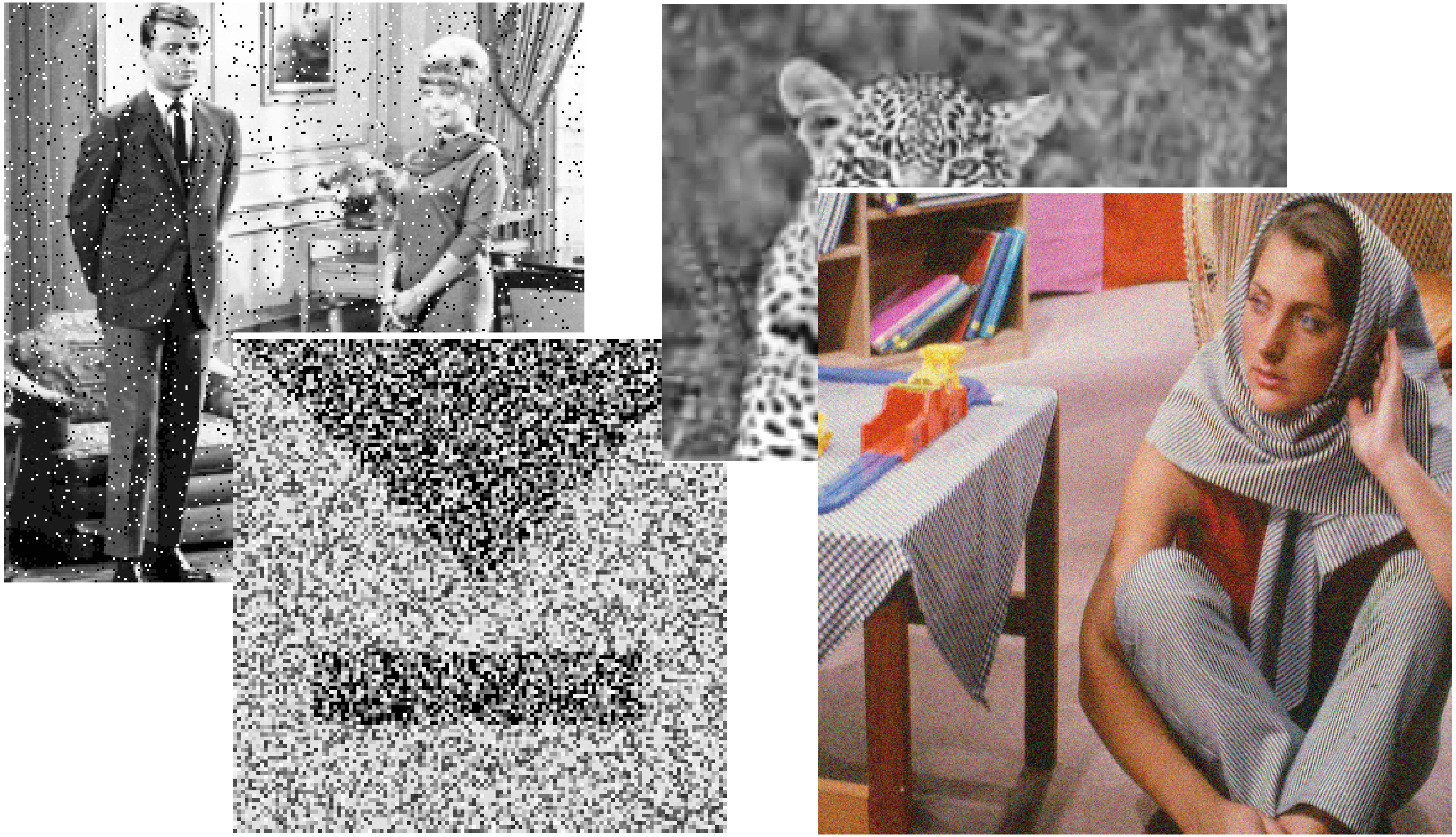


$\alpha = 1, \beta = 0.01, \rho = 0.01$

$\alpha = 1, \beta = 0.001, \rho = 0.01$

$\alpha = 4, \beta = 0.04, \rho = 0.01$

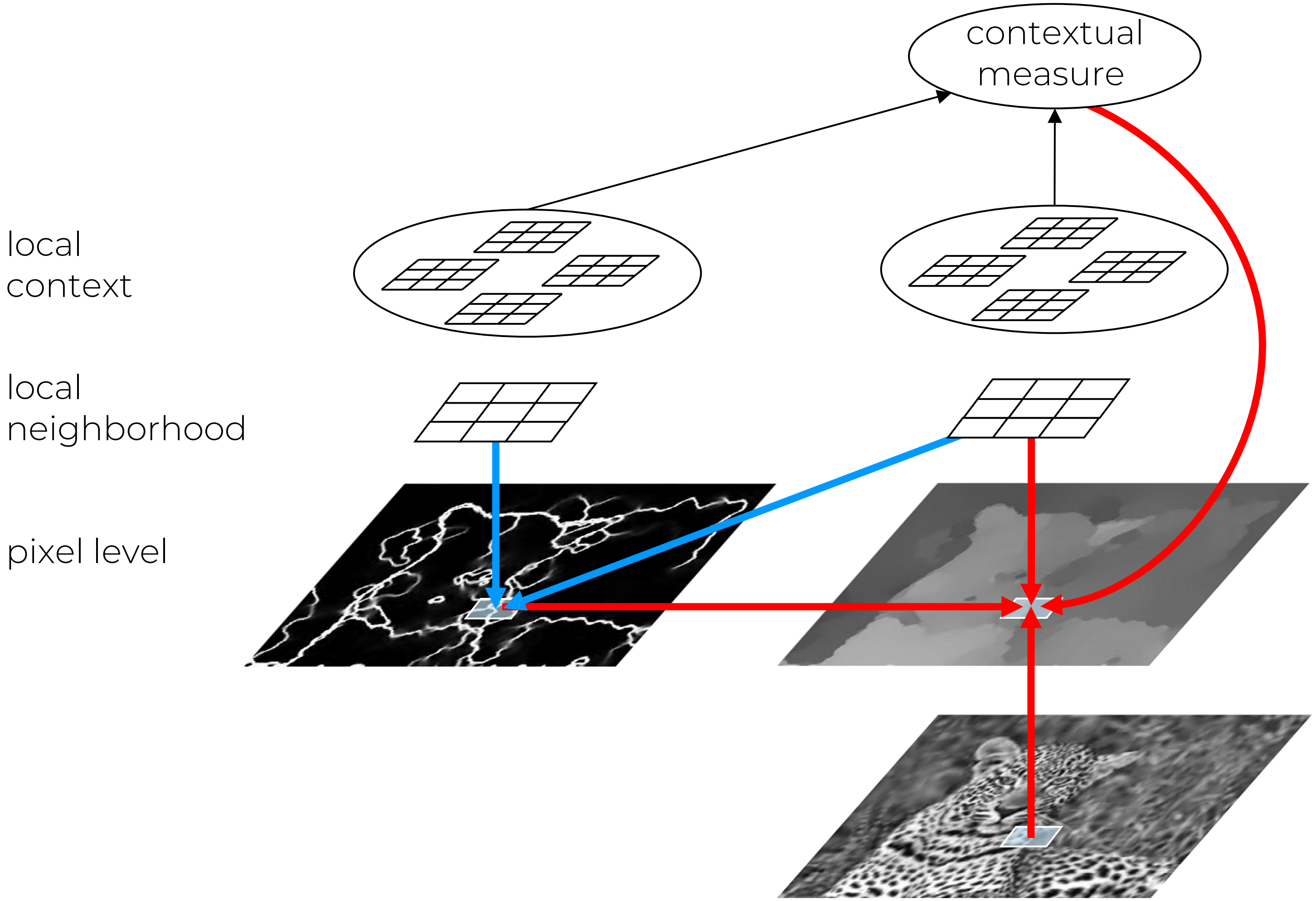
Challenging Cases for Ambrosio-Tortorelli Approximation



Context-Guided Image Smoothing

- E. Erdem and S. Tari,
  “Mumford-Shah Regularizer with Contextual Feedback”,
  Journal of Mathematical Imaging and Vision,
  Vol. 33, No.1, pp. 67-84, January 2009
- Contextual knowledge extracted from local image regions guides the regularization process.

Context-Guided Image Smoothing



Context-Guided Image Smoothing

- 2 coupled processes (u and v modules)

$$\frac{\partial v}{\partial t} = \nabla^2 v - \frac{2\alpha|\nabla u|^2 v}{\rho} - \frac{(v-1)}{\rho^2}; \qquad \left.\frac{\partial v}{\partial n}\right|_{\partial\Omega} = 0$$

$$\frac{\partial u}{\partial t} = \nabla \cdot ((cv)^2 \nabla u) - \frac{\beta}{\alpha}(u-f); \qquad \left.\frac{\partial u}{\partial n}\right|_{\partial\Omega} = 0$$

$$cv = \phi v + (1-\phi)V$$

$\phi \in [0,1]$ $\qquad V \in \{0,1\}$

The Roles of ϕ and V

1. Eliminating an accidentally occurring event
   - e.g., a high gradient due to noise
   - V=1, φ is low for accidental occurrences

$$(cv)_i^2 = (\phi_i v_i + (1 - \phi_i)\, 1)^2$$

2. Preventing an accidental elimination of a feature of interest
   - e.g., encourage edge formation
   - V=0, φ is low for meaningful occurrences

$$(cv)_i^2 = (\phi_i v_i + (1 - \phi_i)\, 0)^2$$

Experimental Results

- Suggested contextual measures:
  1. Directional consistency of edges
     - shapes have smooth boundaries
  2. Edge Continuity
     - gap filling
  3. Texture Edges
     - boundary between different textured regions
  4. Local Scale
     - Resolution varies throughout the image

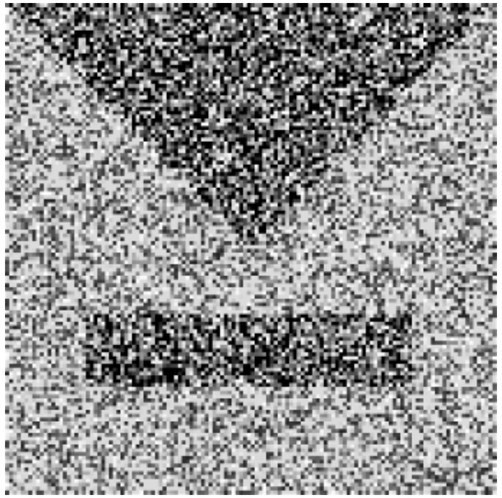
Directional Consistency

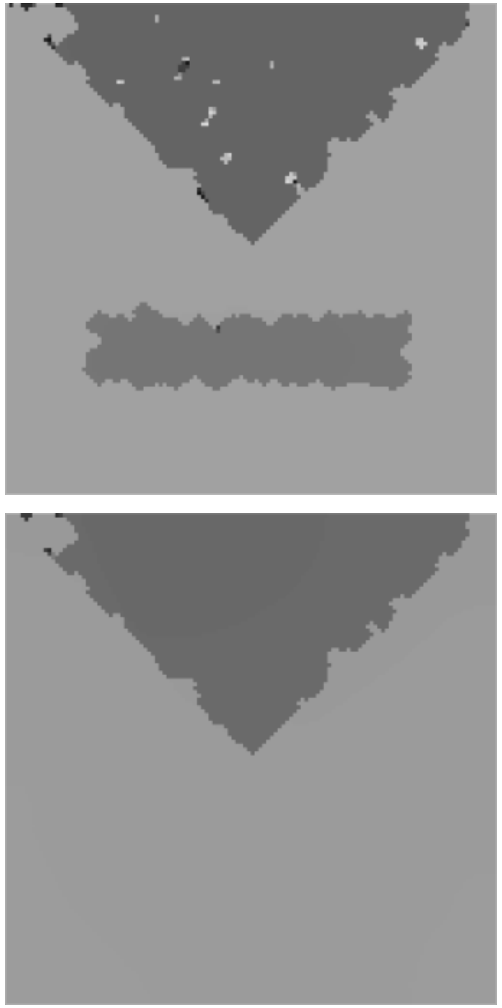
Approximate MS




Context guided filtering result

Directional Consistency

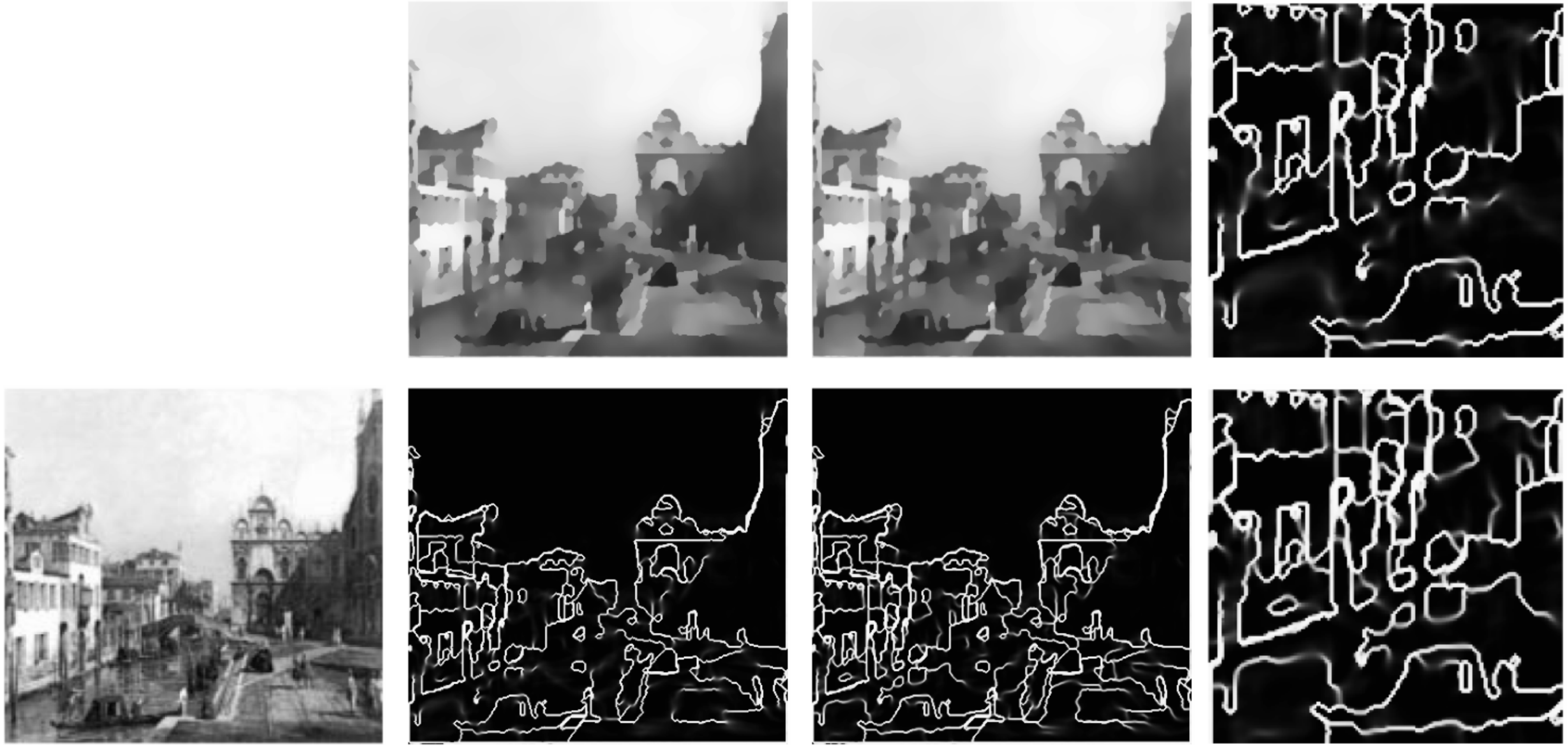




Approximate MS

Context guided filtering result

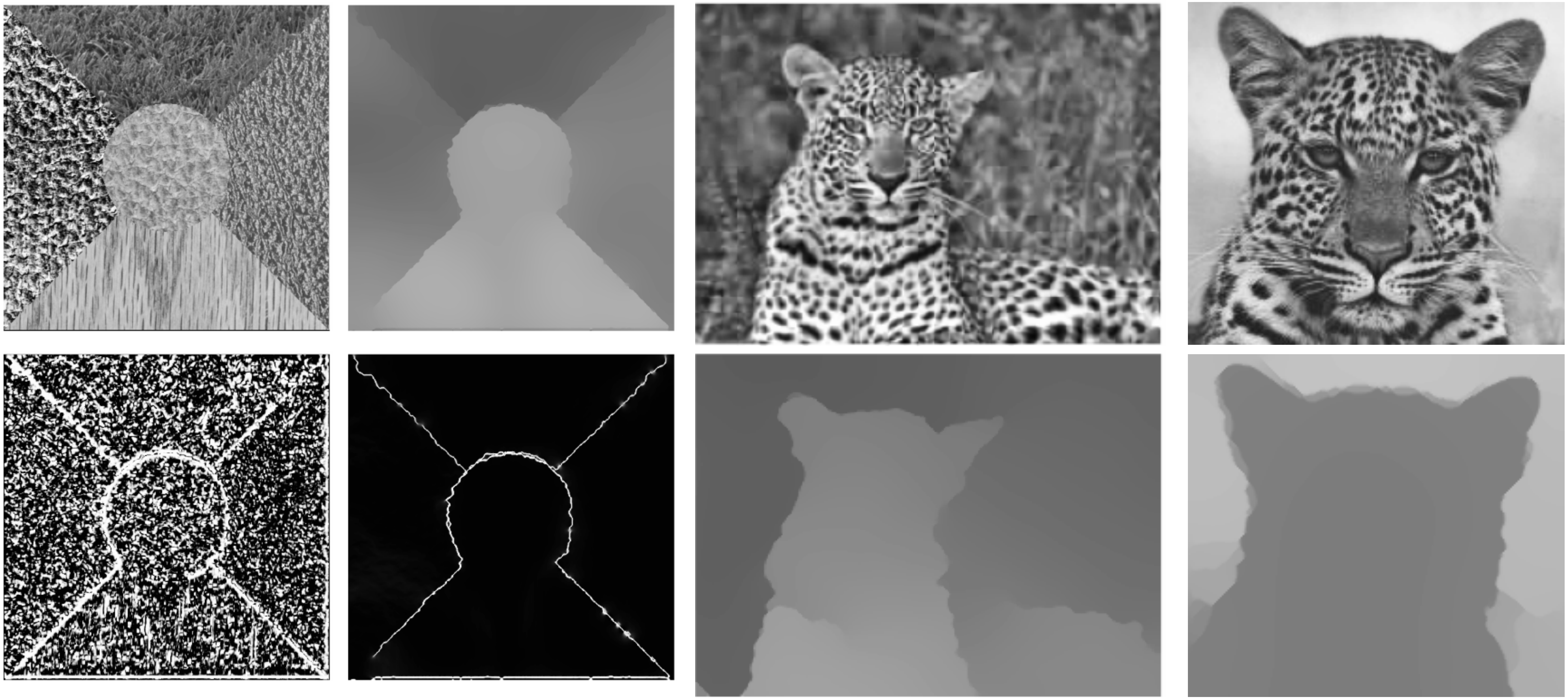
Edge Continuity



Approximate MS

Context guided
filtering result

Coalition of Directional Consistency and Texture Edges



Φ^{te}

Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

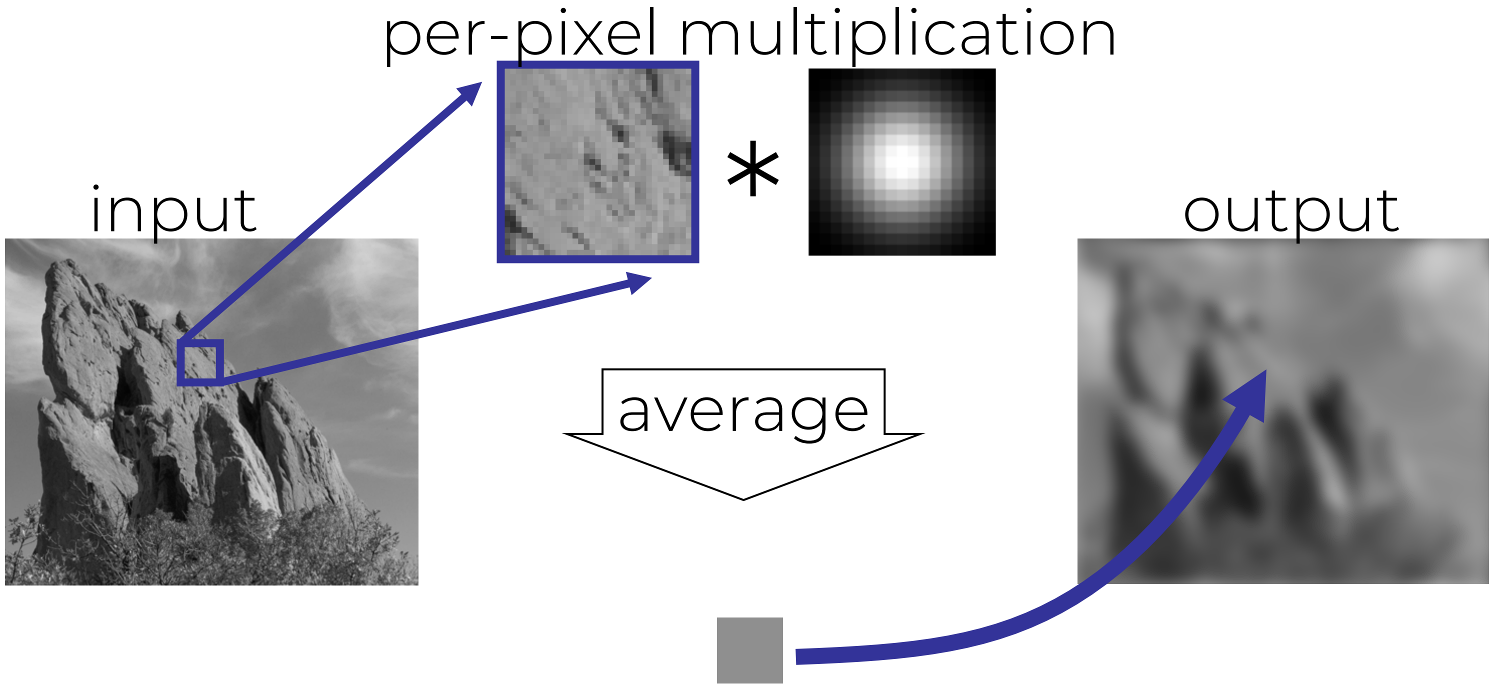
Slides are adapted from the ones prepared by S. Paris, P. Kornprobst, J. Tumblin, and F. Durand.

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

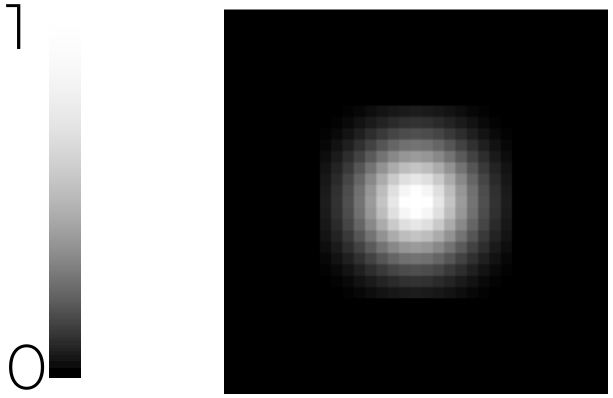
Gaussian Blur

Idea: **weighted average of pixels.**



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

normalized
Gaussian function

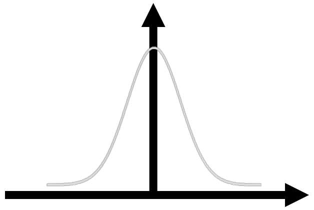


Spatial Parameter

input

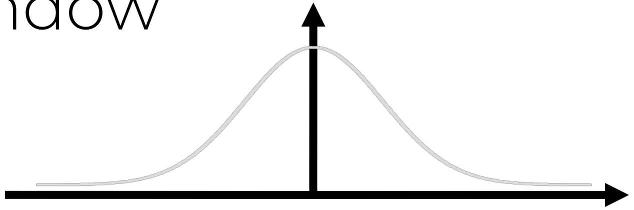
$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} G_{\sigma}(\|\mathbf{p}-\mathbf{q}\|)\, I_{\mathbf{q}}$$

size of the window



small $\boldsymbol{\sigma}$

limited smoothing



large $\boldsymbol{\sigma}$

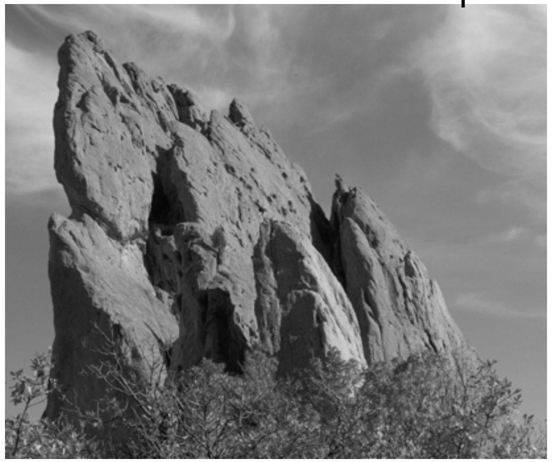
strong smoothing

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

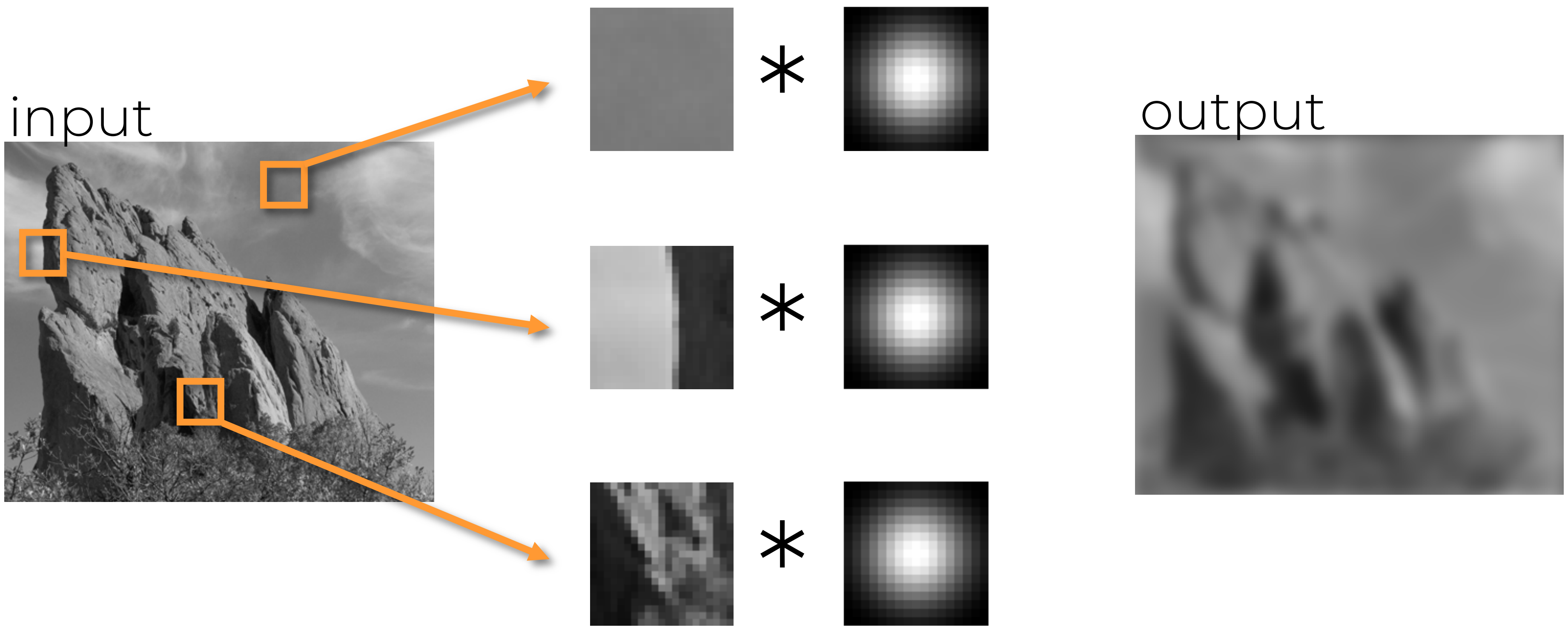
$$GB[I]_{\mathbf{p}} = \sum_{\mathbf{q}\in S} \underbrace{G_\sigma(\|\mathbf{p}-\mathbf{q}\|)}_{\text{space}} I_{\mathbf{q}}$$

input



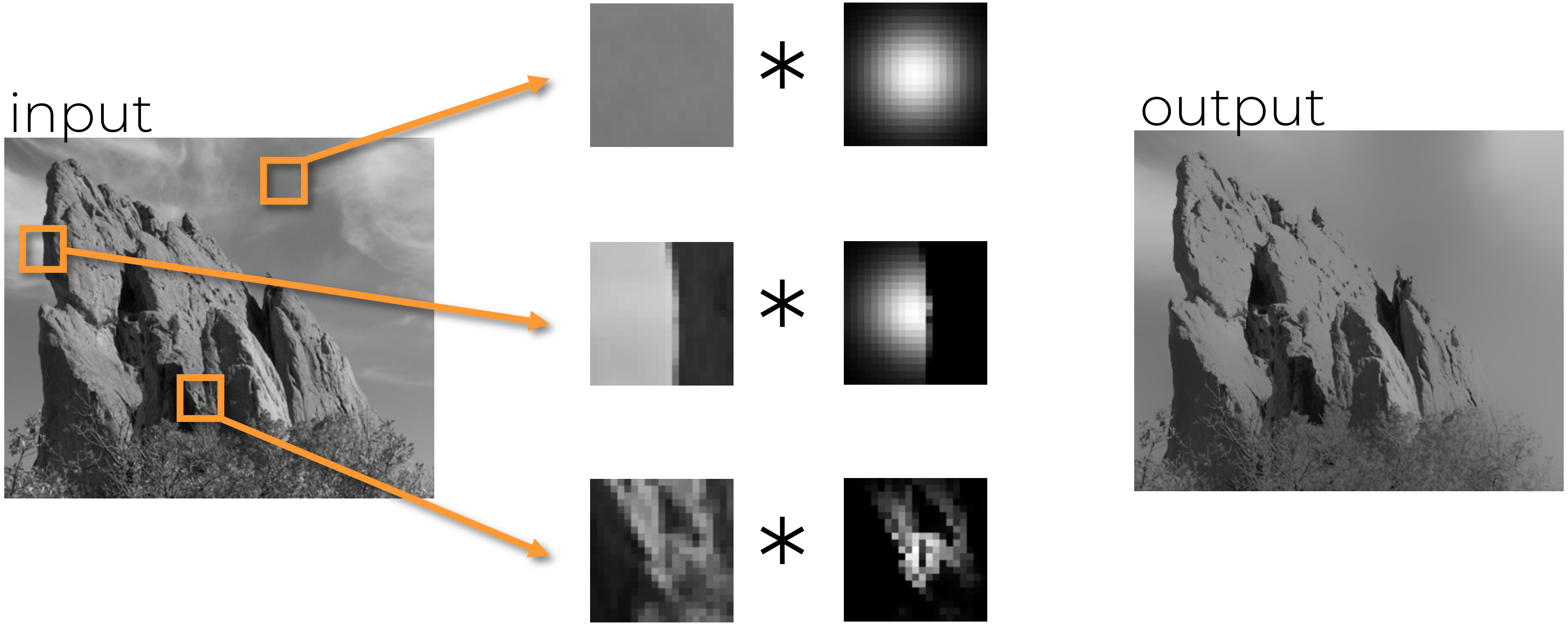
output

Blur Comes from Averaging across Edges



Same Gaussian kernel everywhere.

Bilateral Filter: No Averaging across Edges



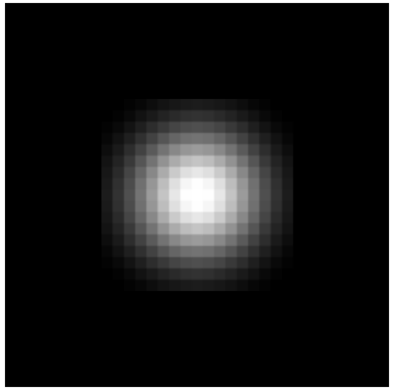
The kernel shape depends on the image content.

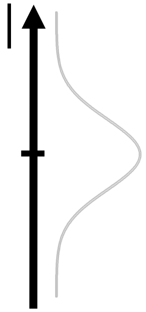
[Aurich 95, Smith 97, Tor

Bilateral Filter: An Additional Edge Term

Same idea: weighted average of pixels.

$$BF[I]_{\mathbf{p}} = \underbrace{\overset{\text{new}}{\boxed{\frac{1}{W_{\mathbf{p}}}}}}_{\text{normalization factor}} \sum_{\mathbf{q}\in S} \underbrace{\overset{\text{not new}}{\boxed{G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)}}}_{\text{space weight}} \underbrace{\overset{\text{new}}{\boxed{G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)}}}_{\text{range weight}} I_{\mathbf{q}}$$





Space and Range Parameters

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

- space σ_{s} : spatial extent of the kernel, size of the considered neighborhood.

- range σ_{r} : “minimum” amplitude of an edge

Exploring the Parameter Space

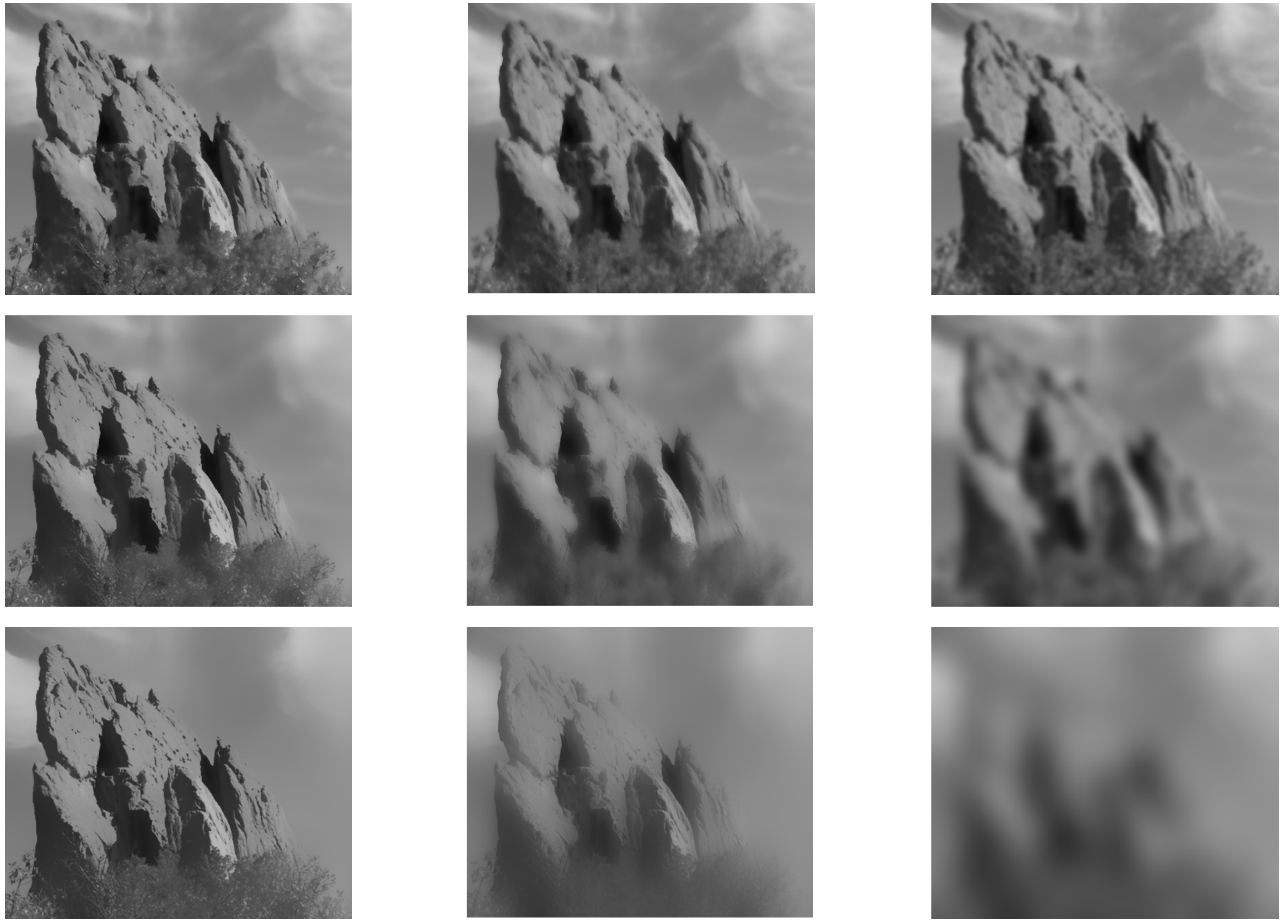

input

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

$\sigma_s = 2$

$\sigma_s = 6$

$\sigma_s = 18$



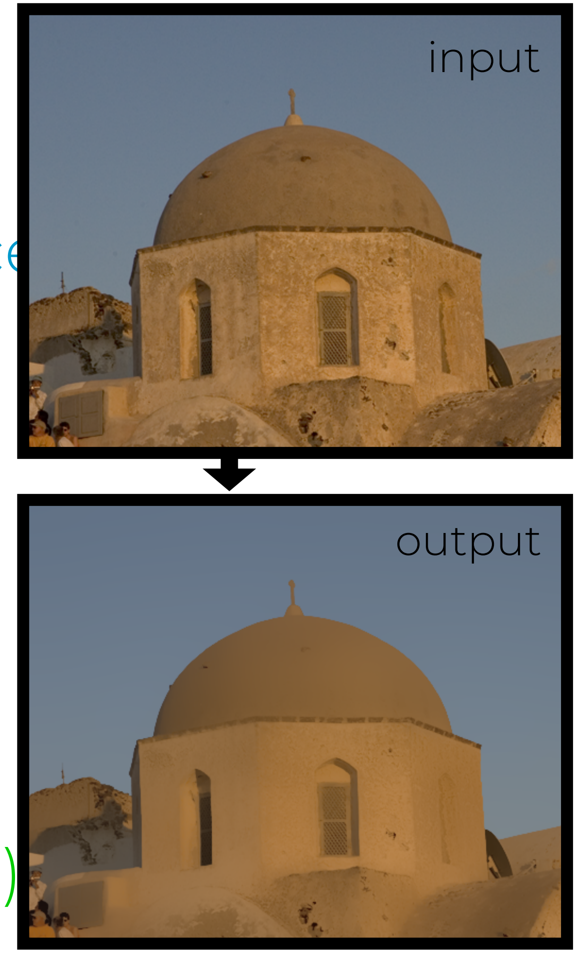
Bilateral Filtering Color Images

For gray-level images

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\| \mathbf{p} - \mathbf{q} \|) \, G_{\sigma_r}\Big(\overbrace{|I_{\mathbf{p}} - I_{\mathbf{q}}|}^{\text{intensity difference}}\Big) \underbrace{I_{\mathbf{q}}}_{\text{scalar}}$$

For color images

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



Hard to Compute

- Nonlinear

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

- Complex, spatially varying kernels
  - Cannot be precomputed, no FFT...

- Brute-force implementation is slow > 10min

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

Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

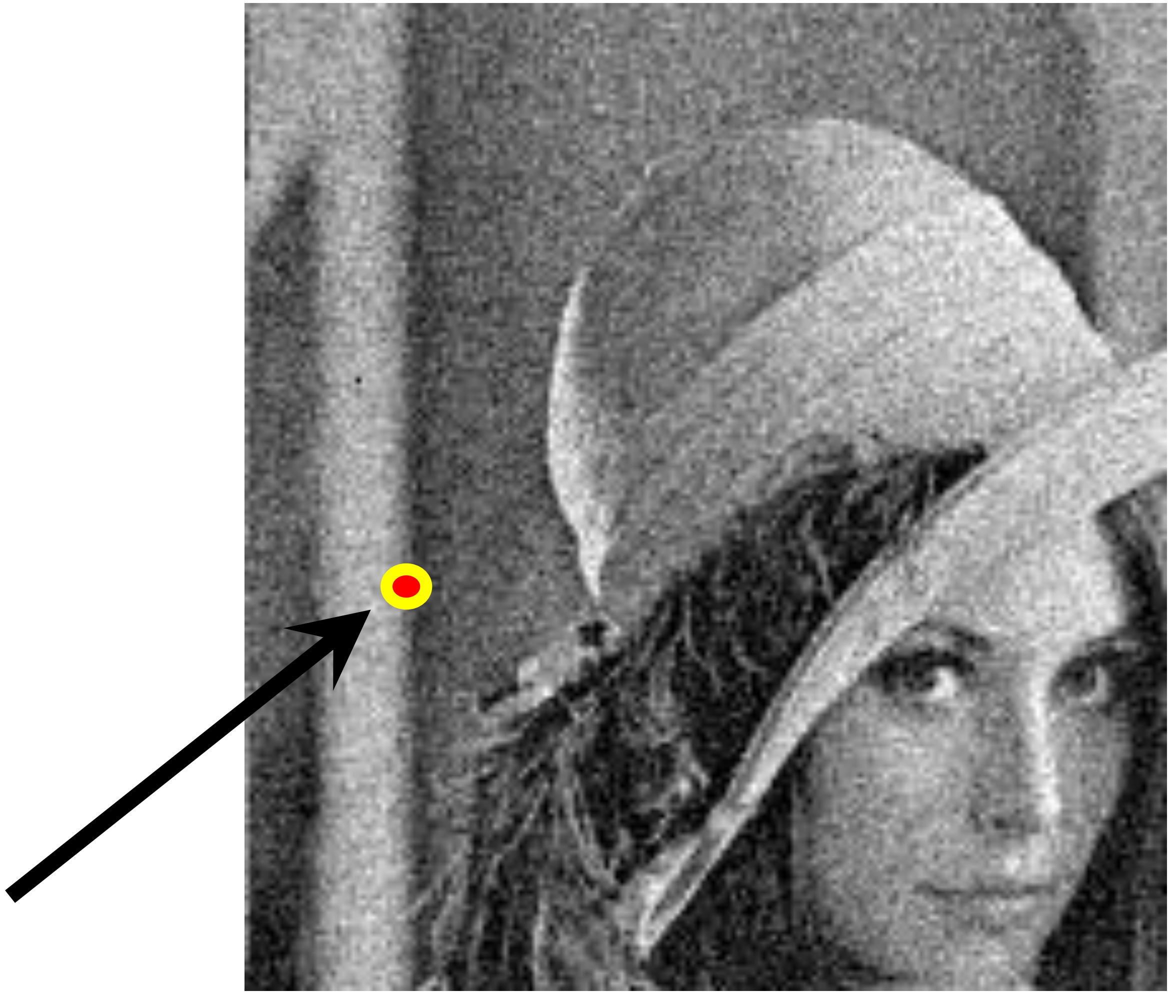
Slides are adapted from the ones prepared by S. Paris, P. Kornprobst, J. Tumblin, and F. Durand.

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

- For each and every pixel p:

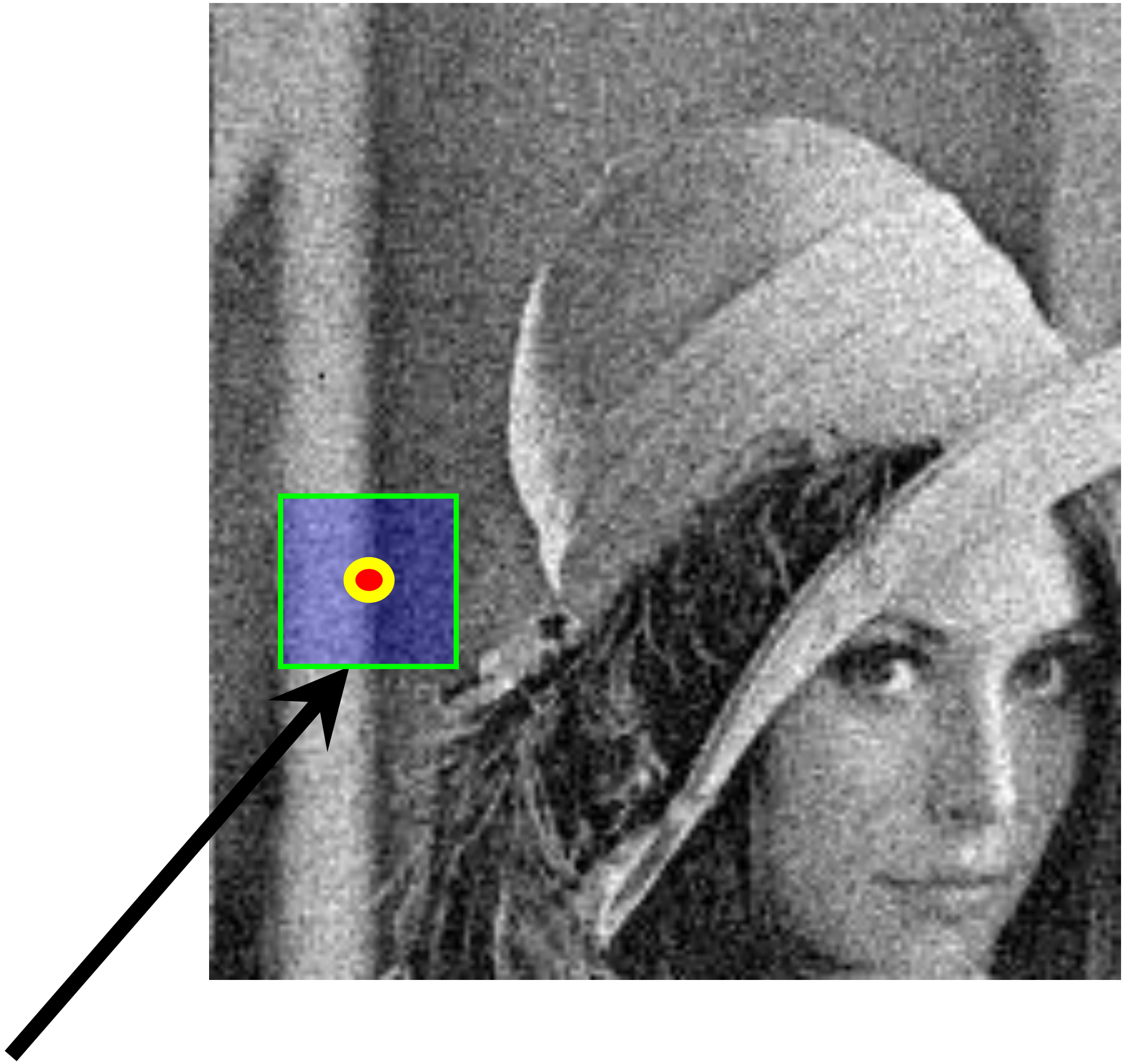


NL-Means Method

- For each and

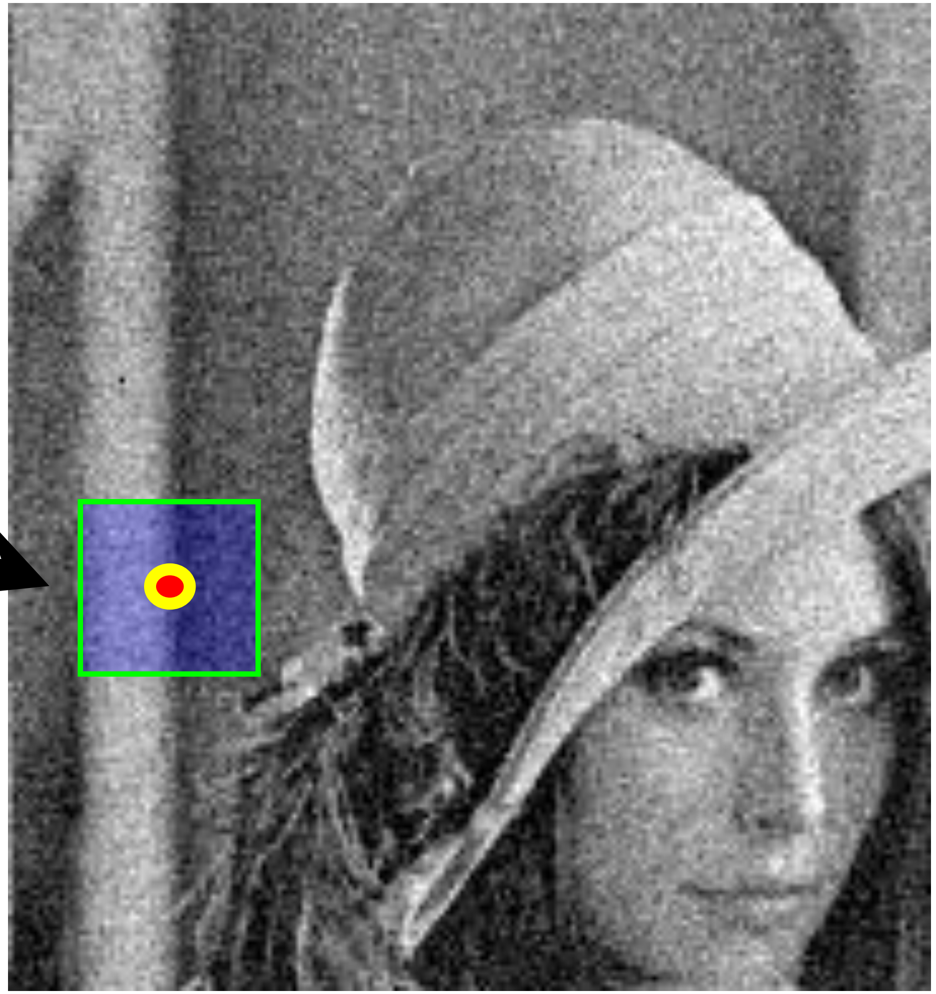
  every pixel p:

  – Define a small, simple fixed size neighborhood;



NL-Means Method

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



- 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;

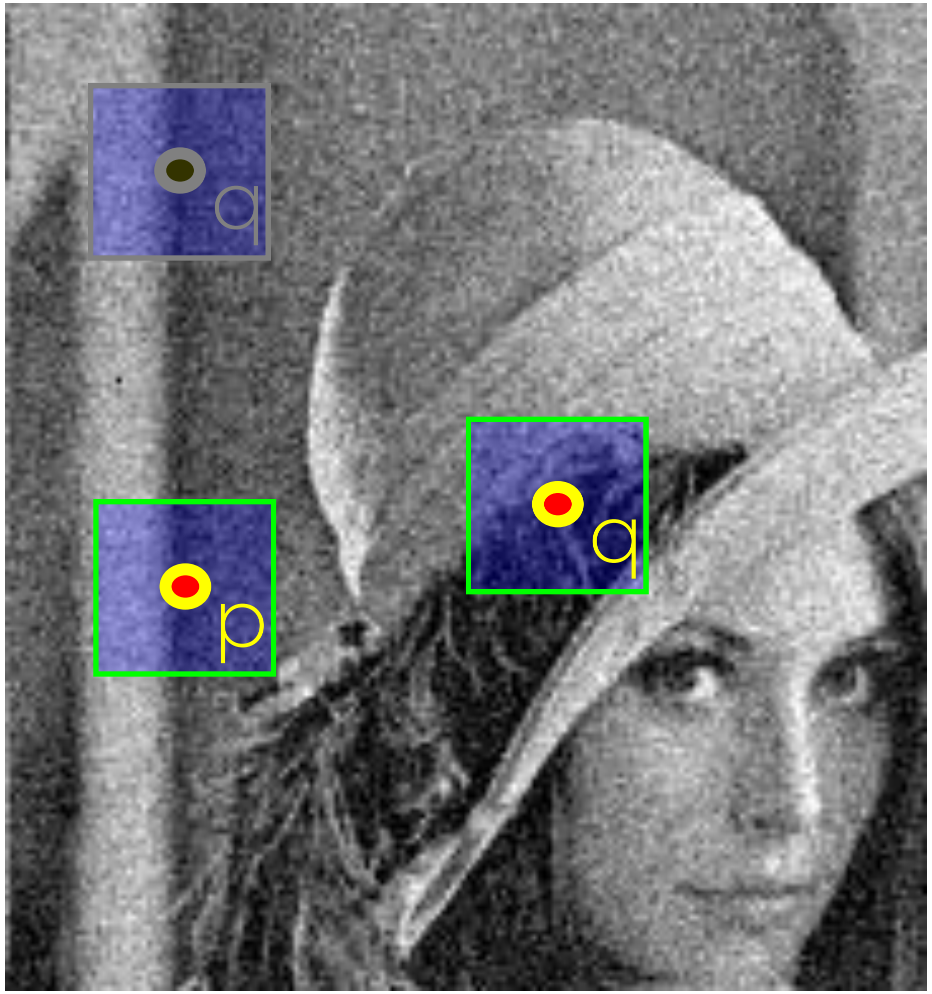
$$|| V_p - V_q ||^2$$

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;

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

Filter with this!

NL-Means Method

p, q neighbors define

a vector distance;

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

Filter with this:

No spatial term!

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

NL-Means Method

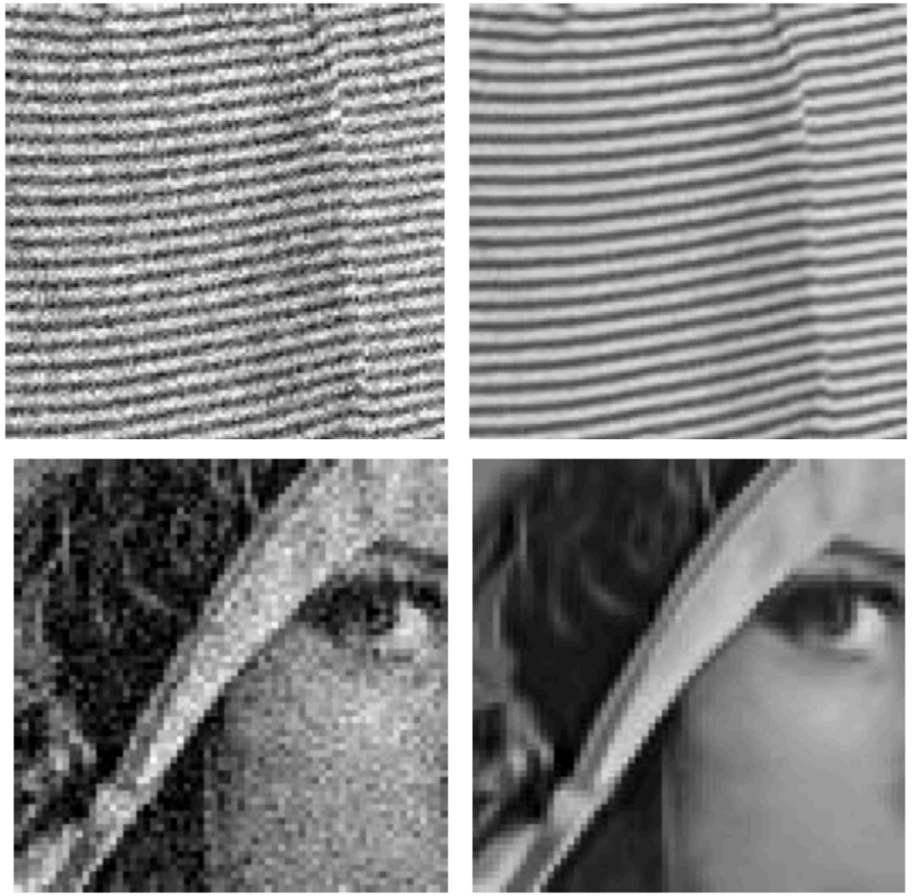
pixels p, q neighbors
Set a vector distance;

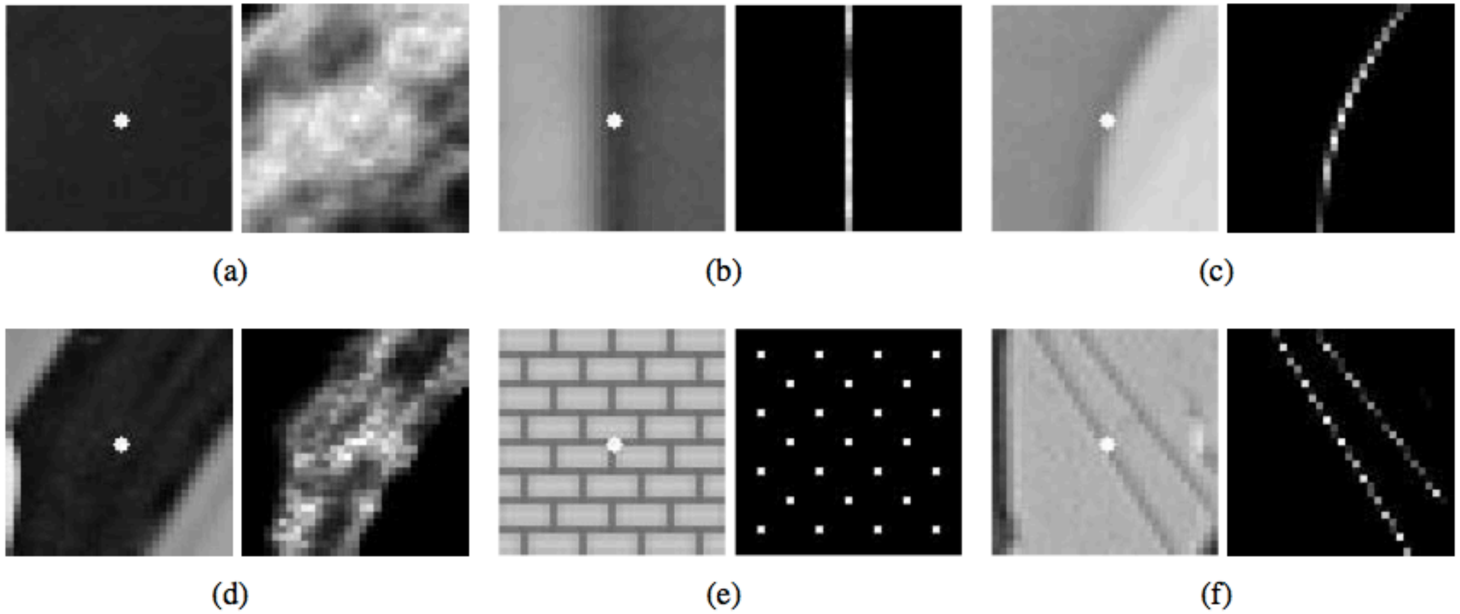
$$\| V_p - V_q \|^2$$

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

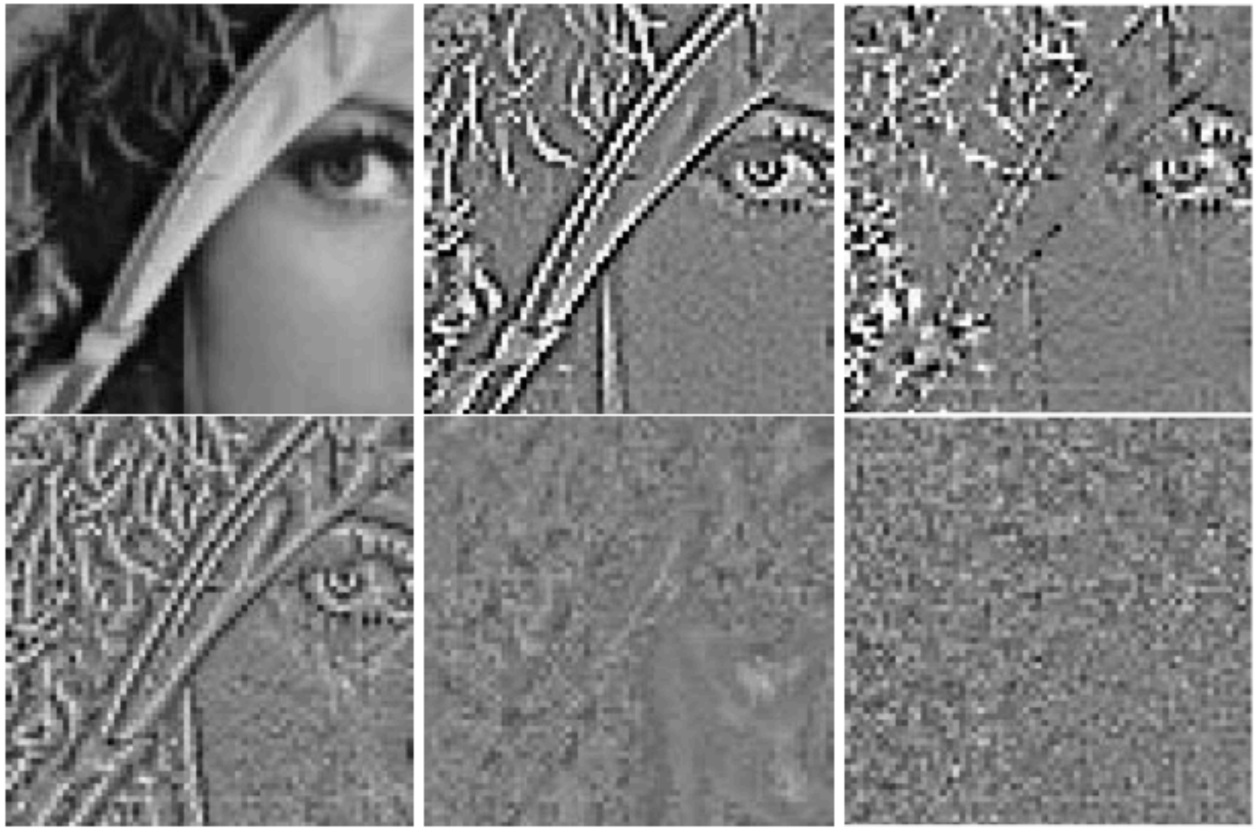


Figure 4. Method noise experience on a natural image. Displaying of the image difference $u - D_h(u)$. From left to right and from top to bottom: original image, Gauss filtering, anisotropic filtering, Total variation minimization, Neighborhood filtering and NL-means algorithm. The visual experiments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

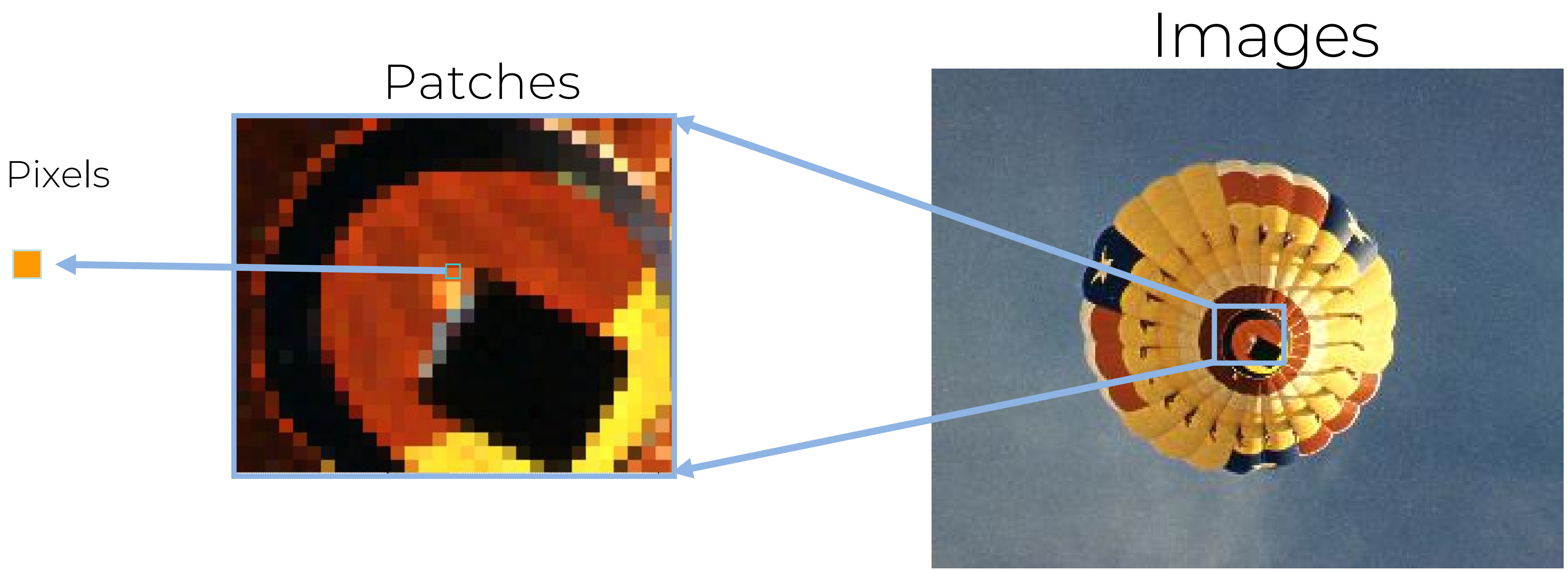
NL-Means Method

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

Today

- Median filter
- Perona-Malik Type Nonlinear Diffusion
- Total Variation (TV) Regularization
- Mumford-Shah Model
- Bilateral filtering
- Non-local means denoising
- Image smoothing via region covariance (RegCov smoothing)

From pixels to patches and to images

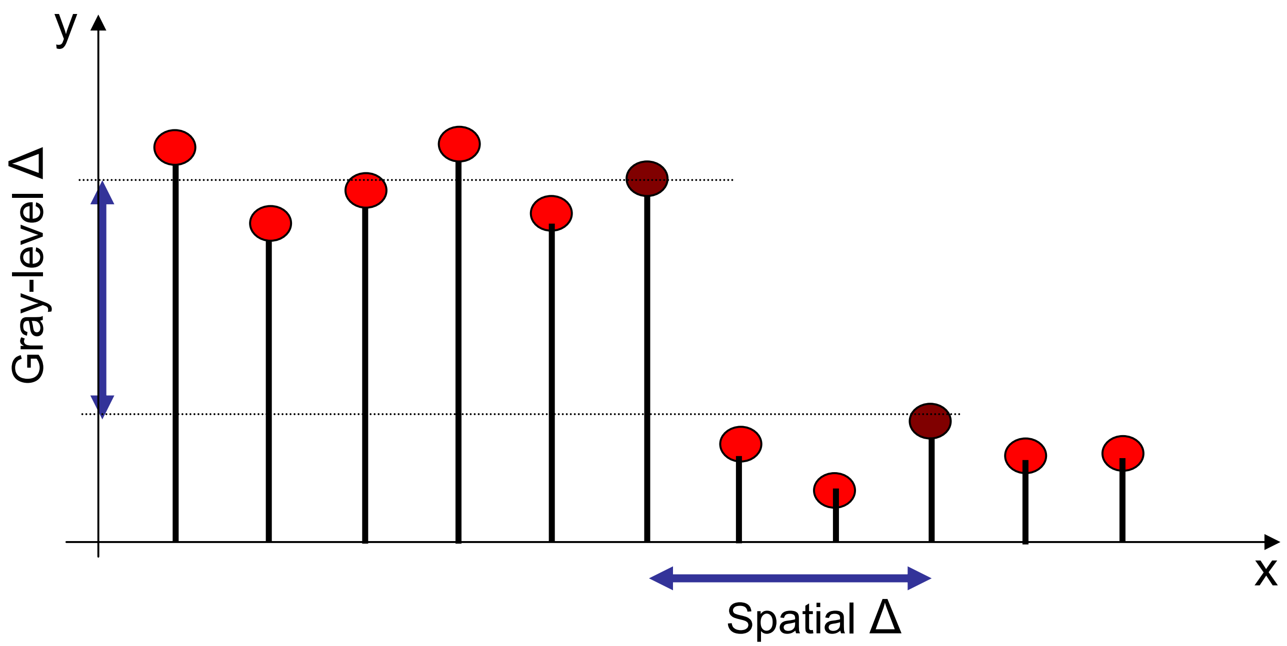


Similarities can be defined at different scales..

Slides: P. Milanfar.

Pixelwise similarity metrics

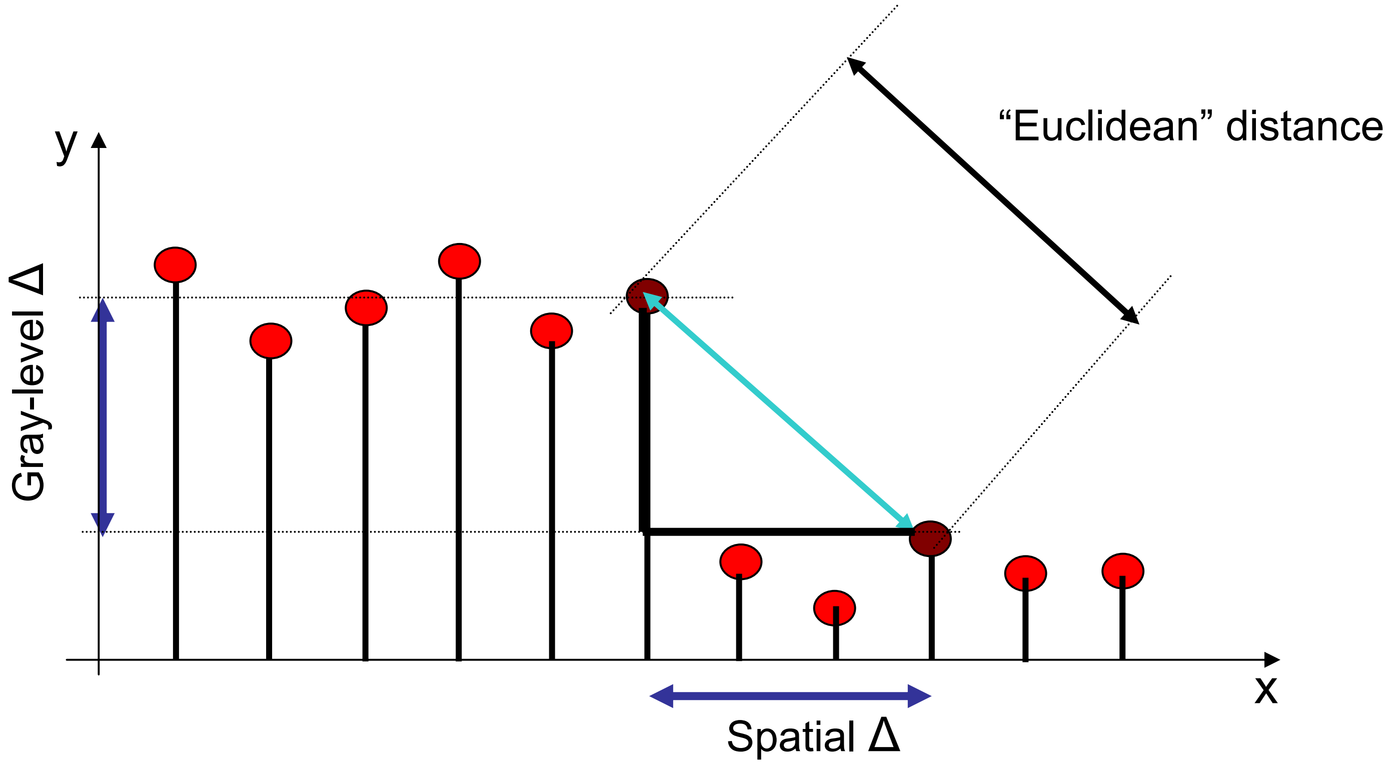
- To measure the similarity of two pixels, we can consider
  - Spatial distance
  - Gray-level distance



Slides: P. Milanfar.

Euclidean metrics

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



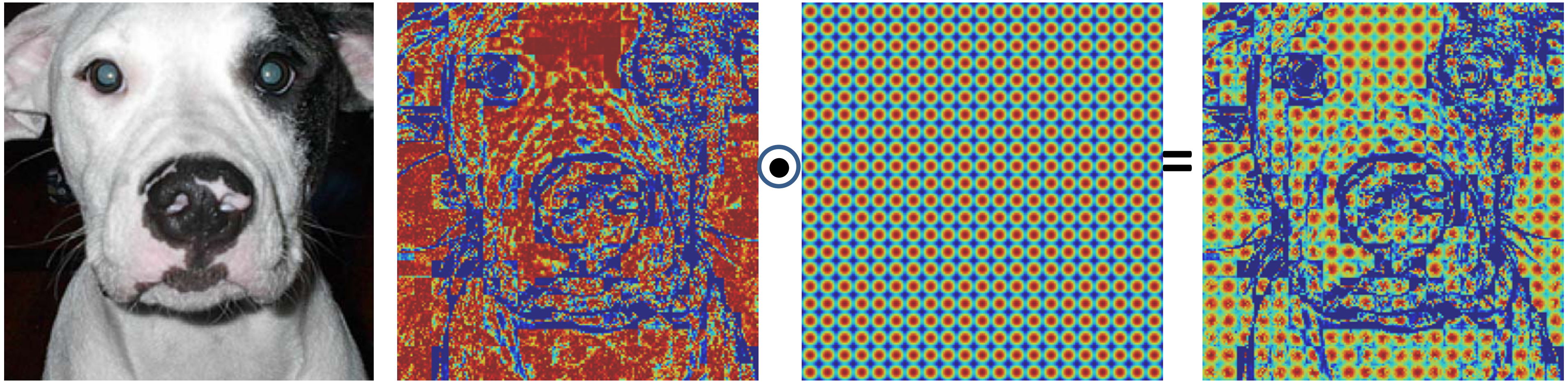
Slides: P. Milanfar.

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



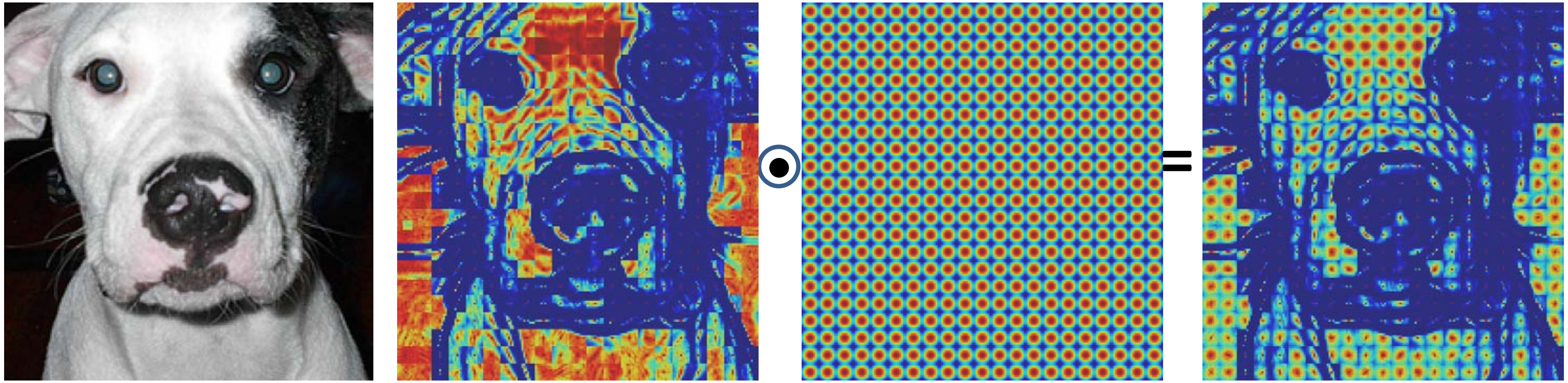
Slides: P. Milanfar.

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

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

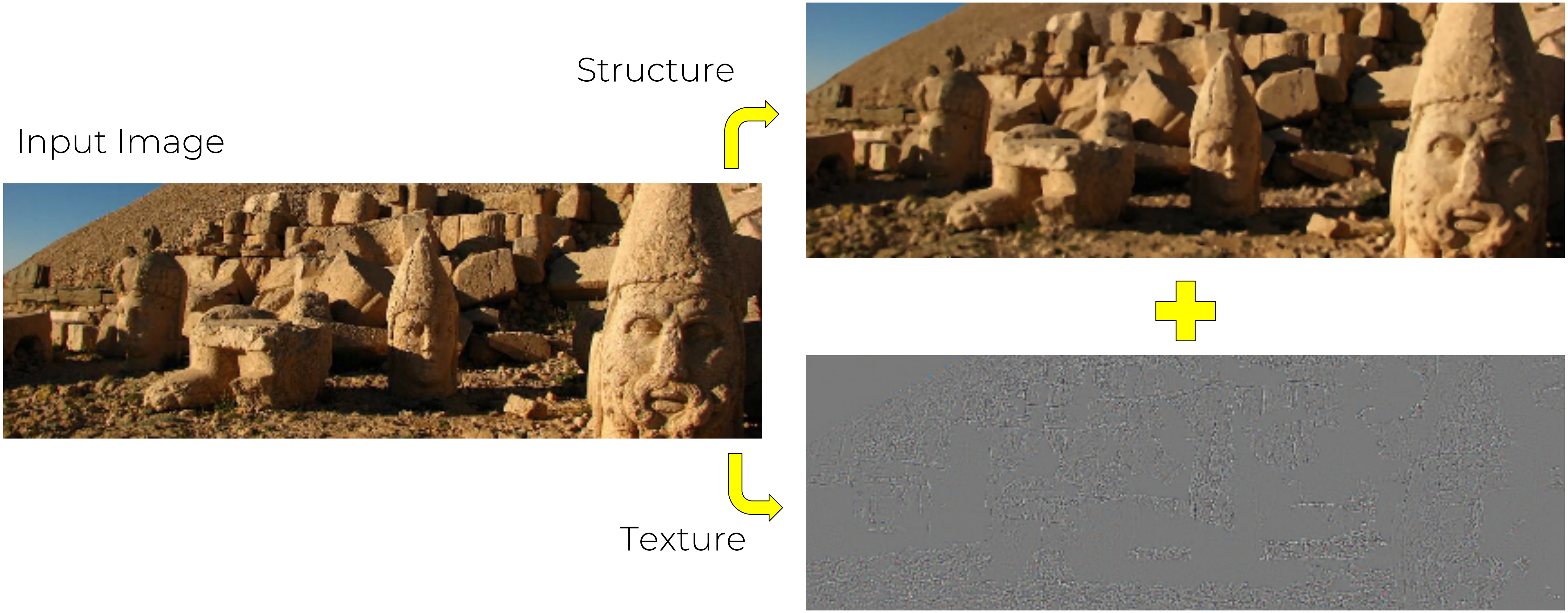
Structure-Texture Decomposition

- Decomposing an image into structure and texture components

Texture Component

Structure-Texture Decomposition

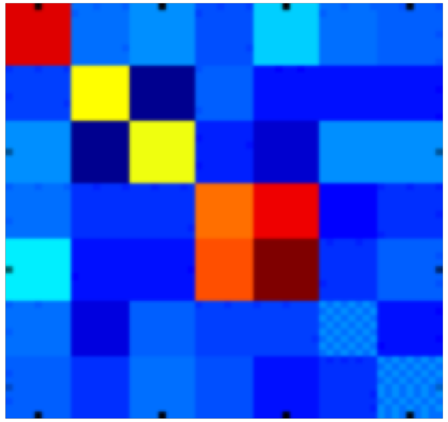
- Decomposing an image into structure and texture components



Structure-Texture Decomposition

$$\mathrm{F}(\mathrm{x},\mathrm{y}) = \phi(\mathrm{I},\mathrm{x},\mathrm{y})$$

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

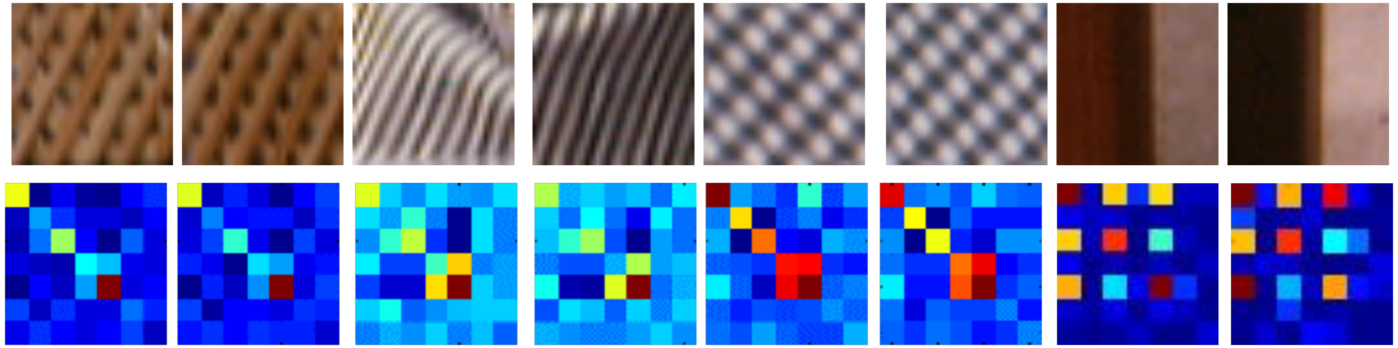


$$\mathbf{C}_R = \frac{1}{n-1} \sum_{i=0}^{n} (\mathbf{z}_k - \mu)(\mathbf{z}_k - \mu)^T$$

Tuzel et al., ECCV 2006

Structure-Texture Decomposition

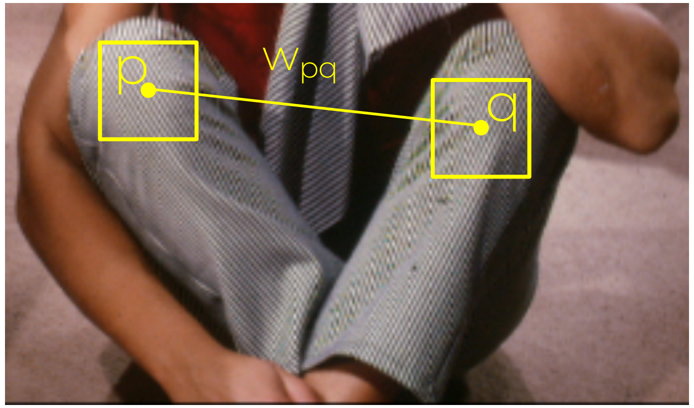
- Region covariances capture local structure and texture information.
- Similar regions have similar statistics.



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)

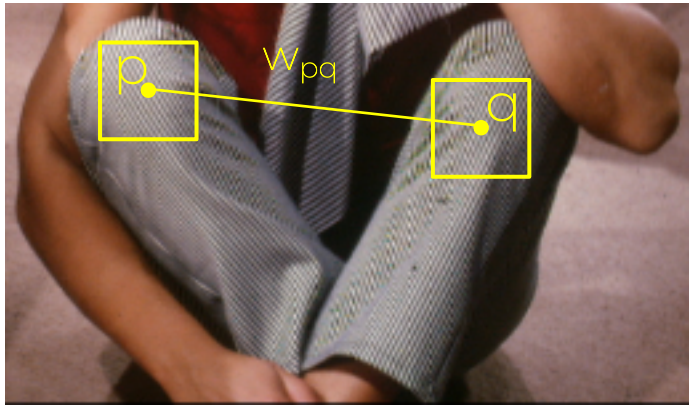
RegCov Smoothing – Model 1

- 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

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



Final representation

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

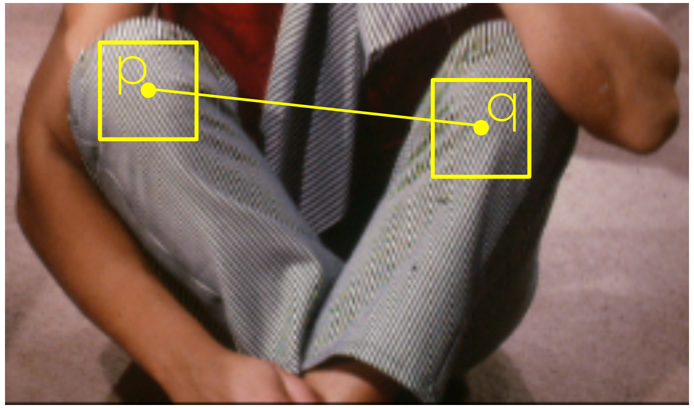
Resulting kernel function

$$w_{\mathbf{pq}} \propto \exp\left(-\frac{\|\Psi(\mathbf{C}_\mathbf{p}) - \Psi(\mathbf{C}_\mathbf{q})\|^2}{2\sigma^2}\right)$$

RegCov Smoothing – Model 2

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

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

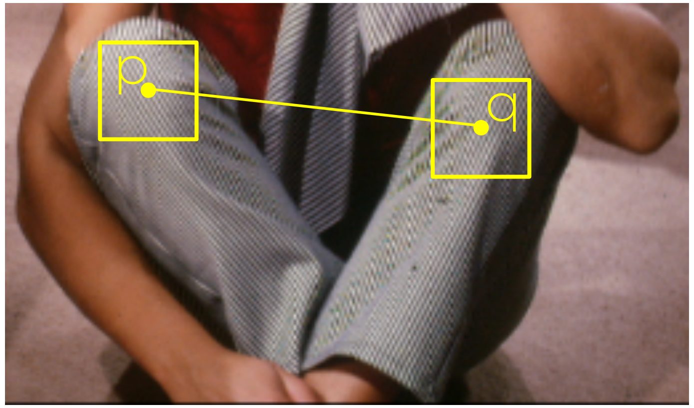


$$\mathbf{C} = \mathbf{C}_{\mathbf{p}} + \mathbf{C}_{\mathbf{q}}$$

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



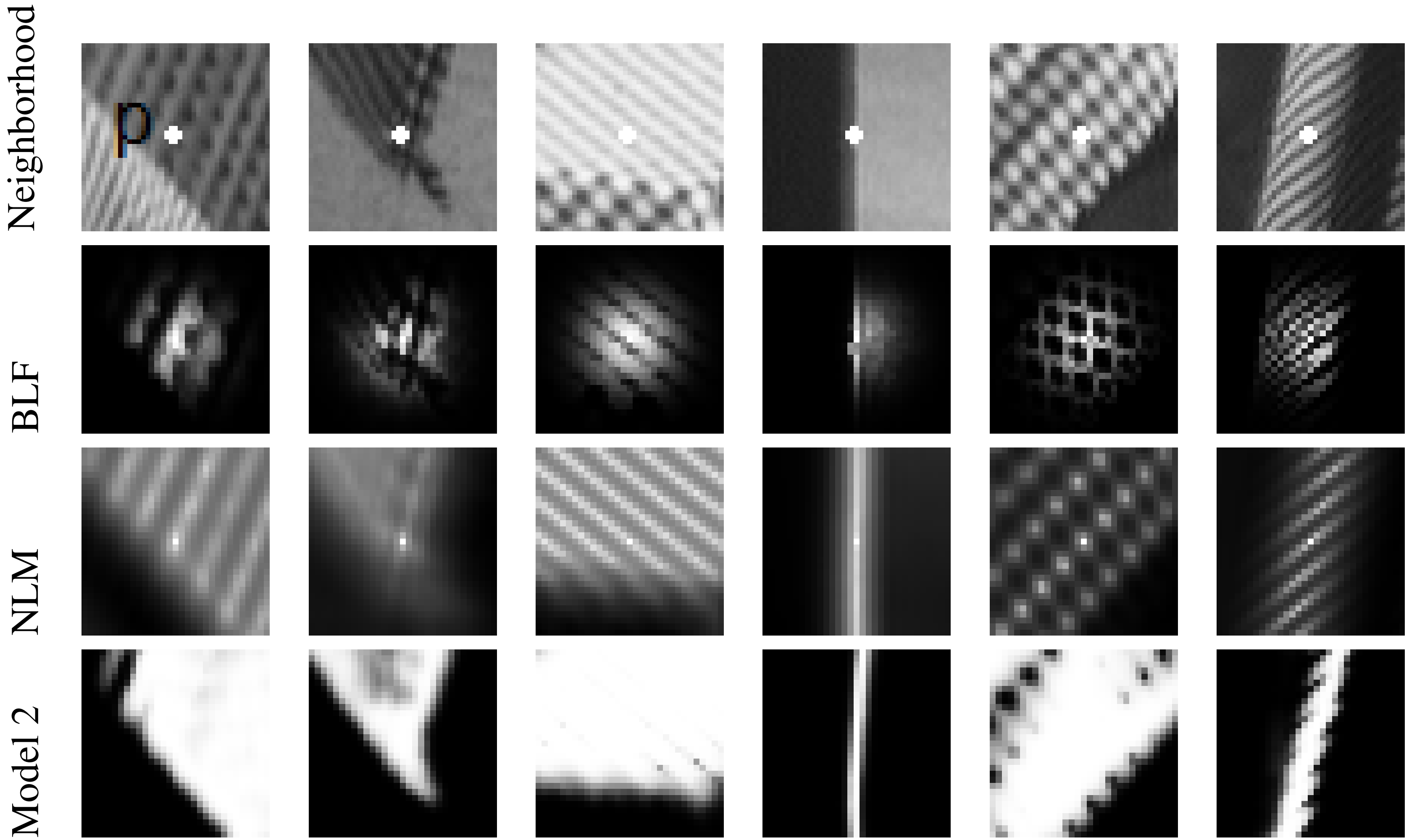
$$d_{KL}(\mathbf{p},\mathbf{q}) = \frac{1}{2}\left(tr(\mathbf{C_q}^{-1}\mathbf{C_p}) + (\mu_p - \mu_q)^T\mathbf{C_q}^{-1}(\mu_p - \mu_q) - k - ln\left(\frac{\det \mathbf{C_p}}{\det \mathbf{C_q}}\right)\right)$$

Resulting kernel

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

resulted from a discussion with Rahul Narain (Berkeley University)

RegCov Smoothing – Smoothing Kernels



Results

Input

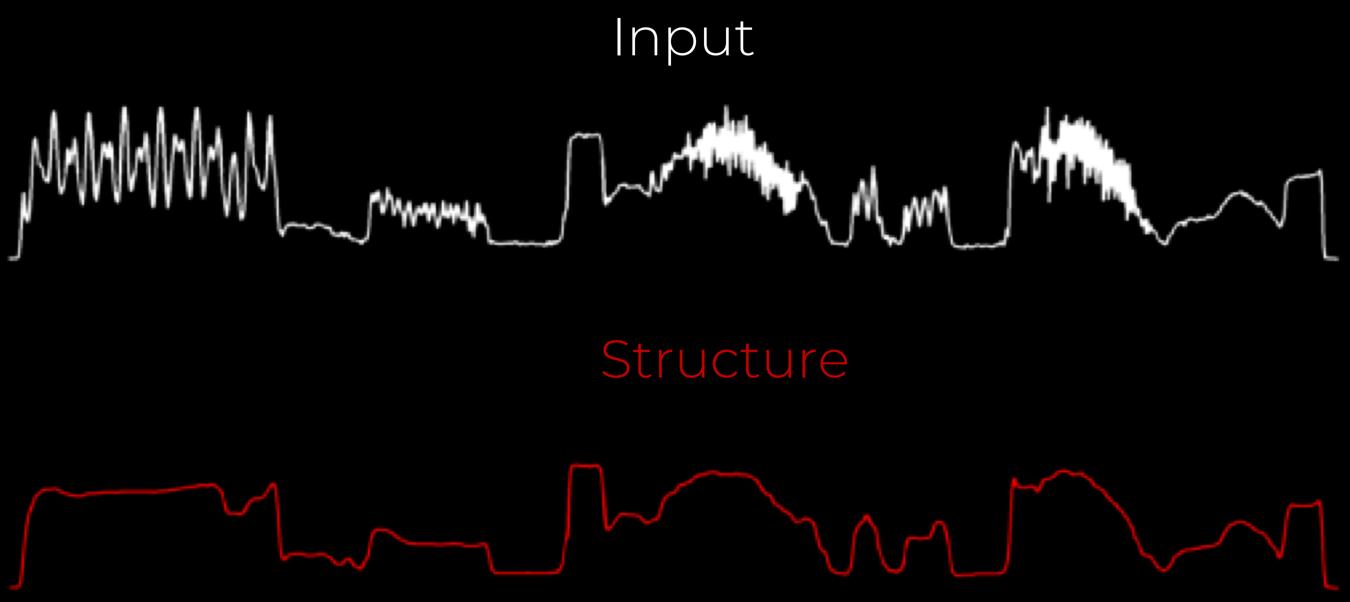
Results

Model1

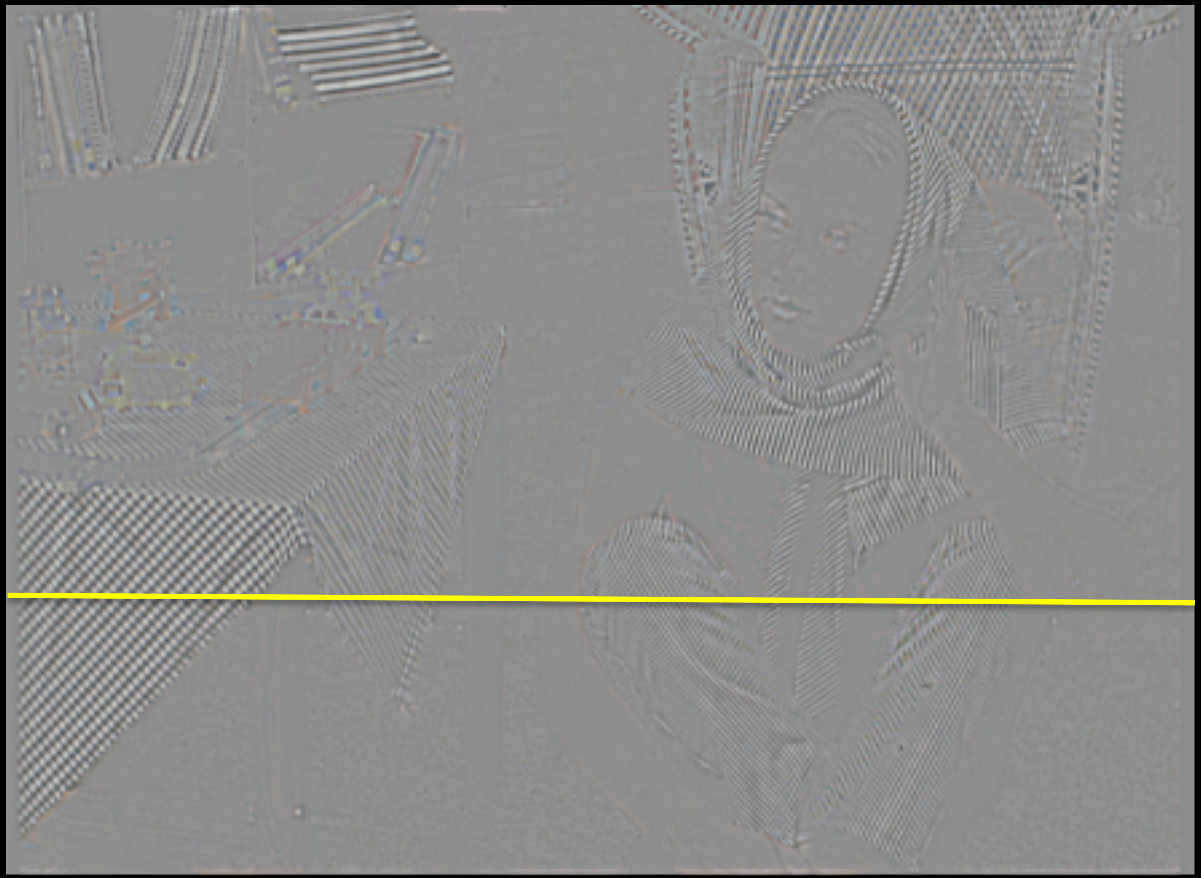
Model2

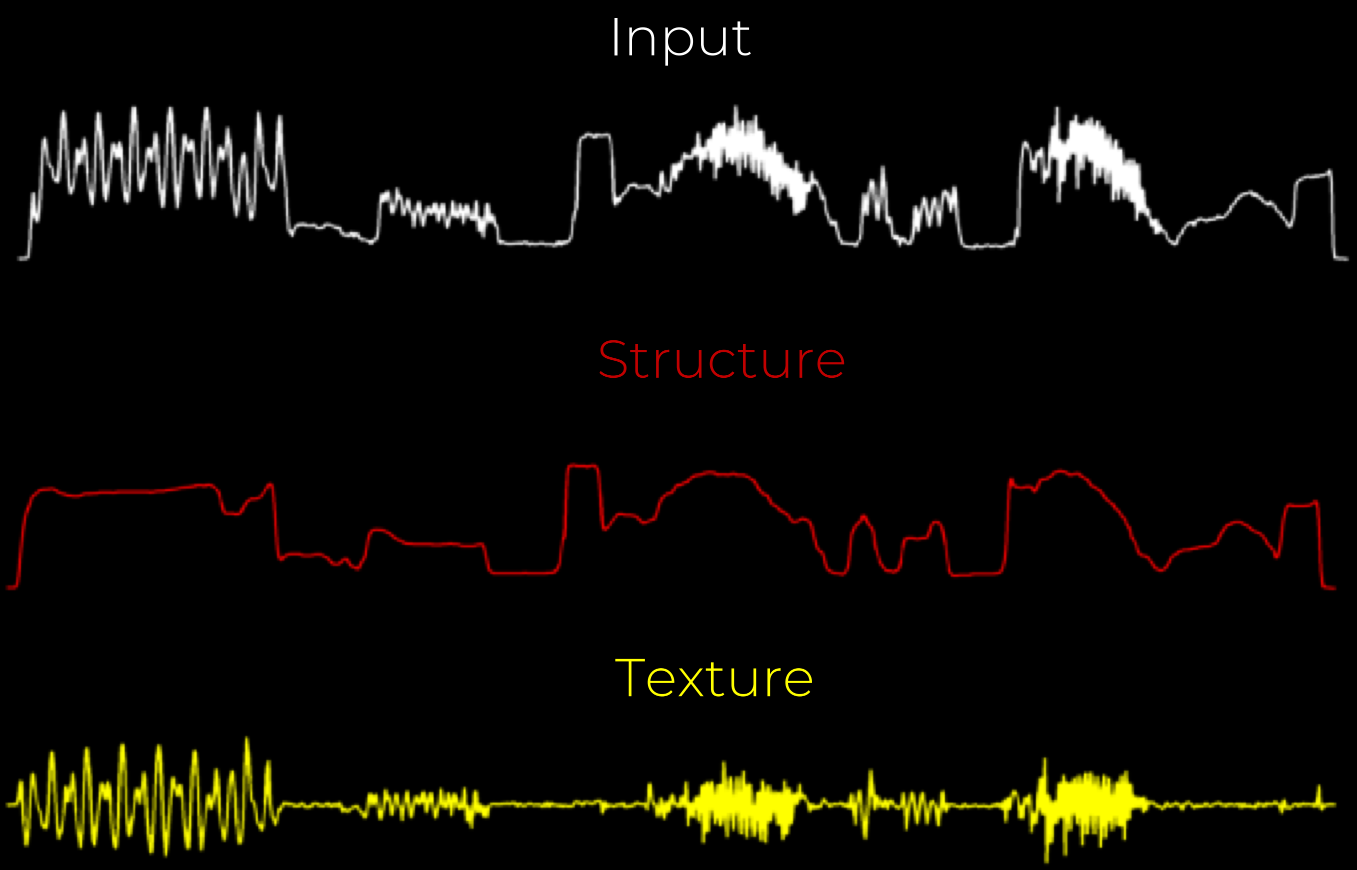
Model3

Input



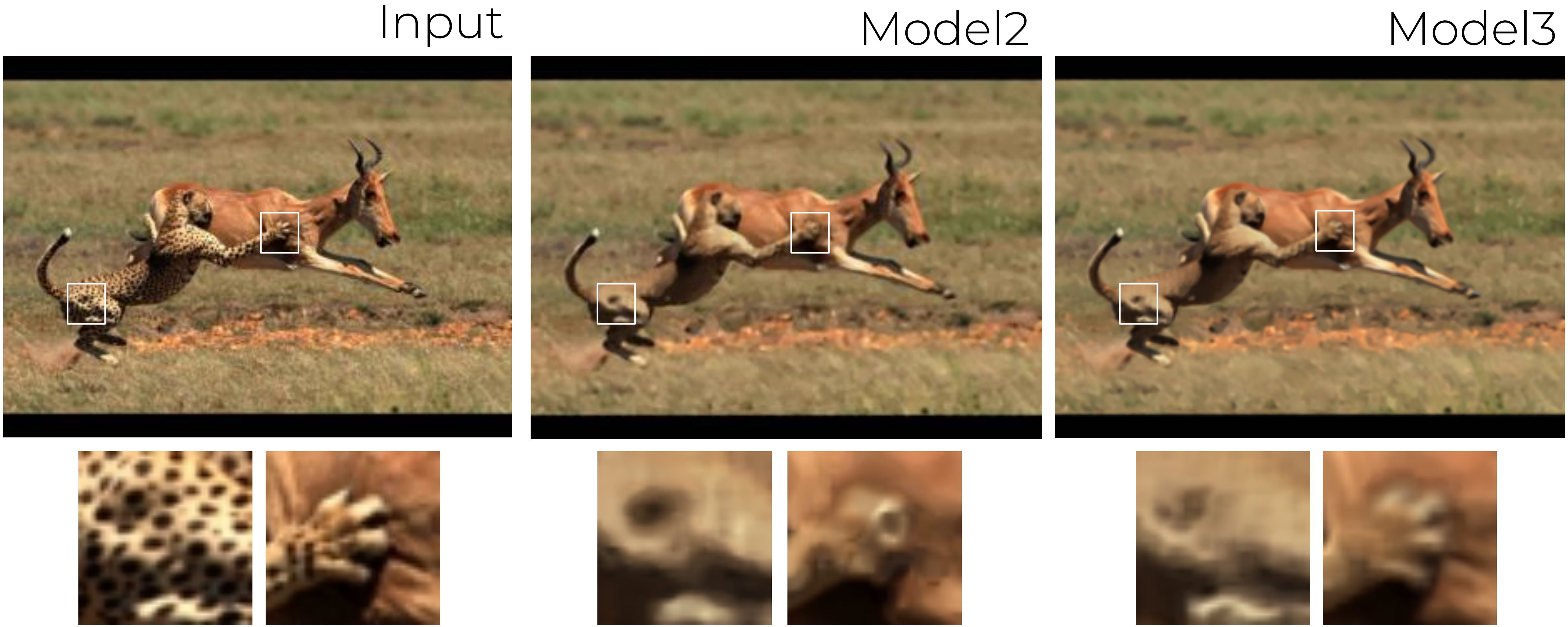
Model2 Structure





Model2 Texture

Results



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

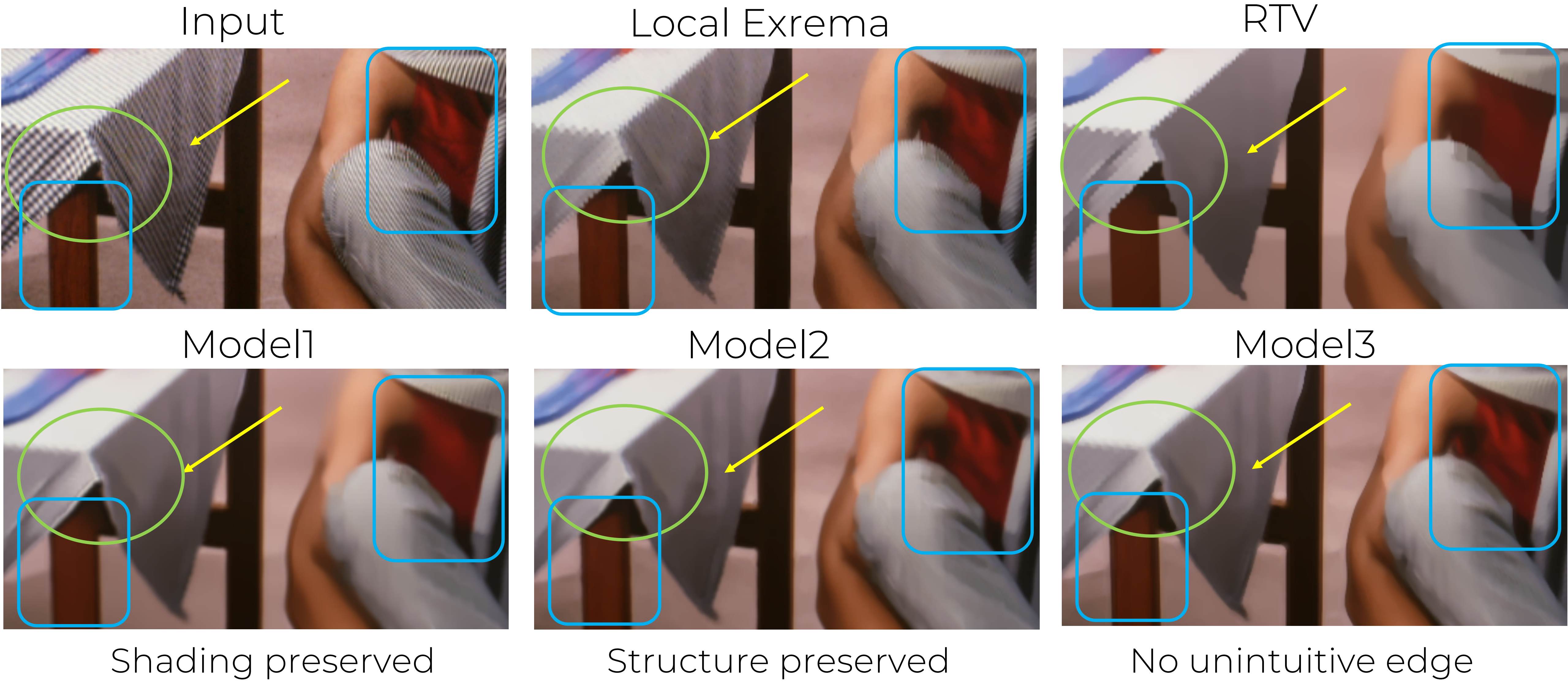
Experimental evaluation

Model 2

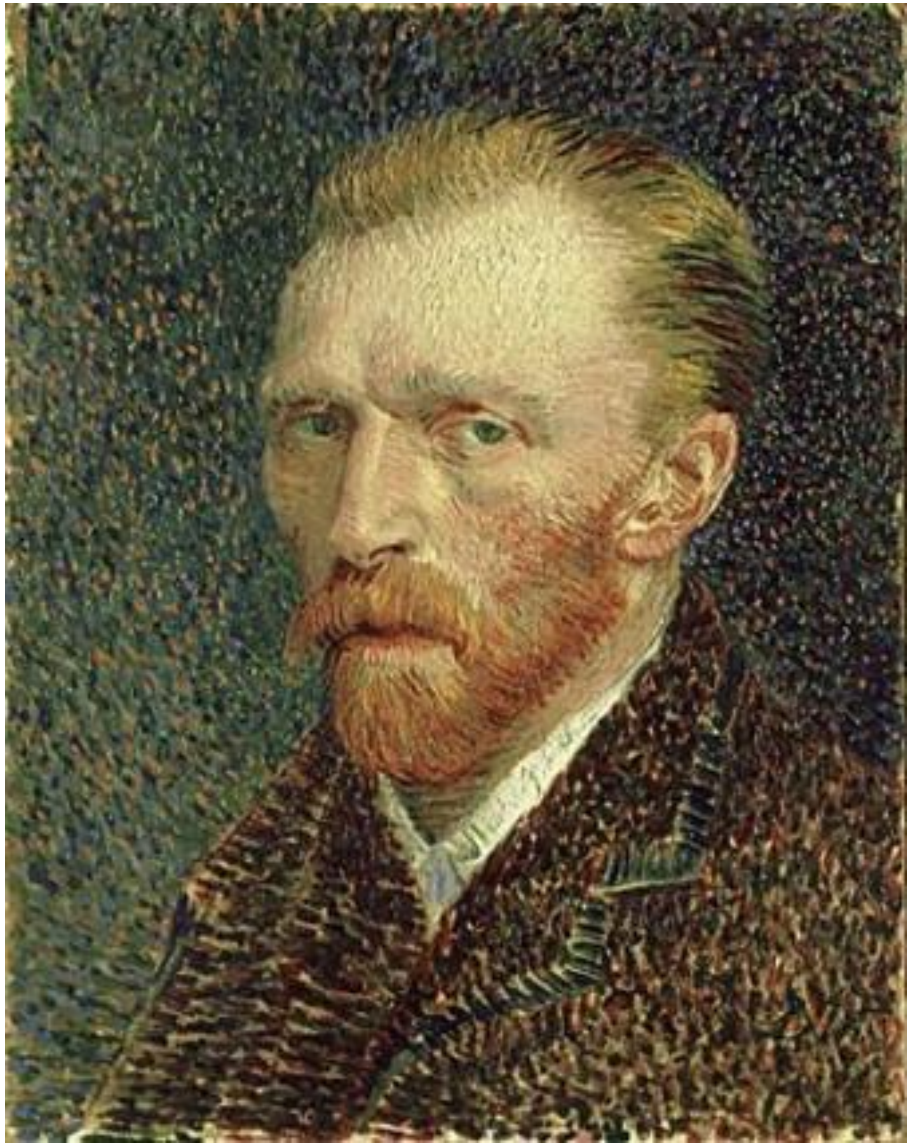
Experimental evaluation

Model 3

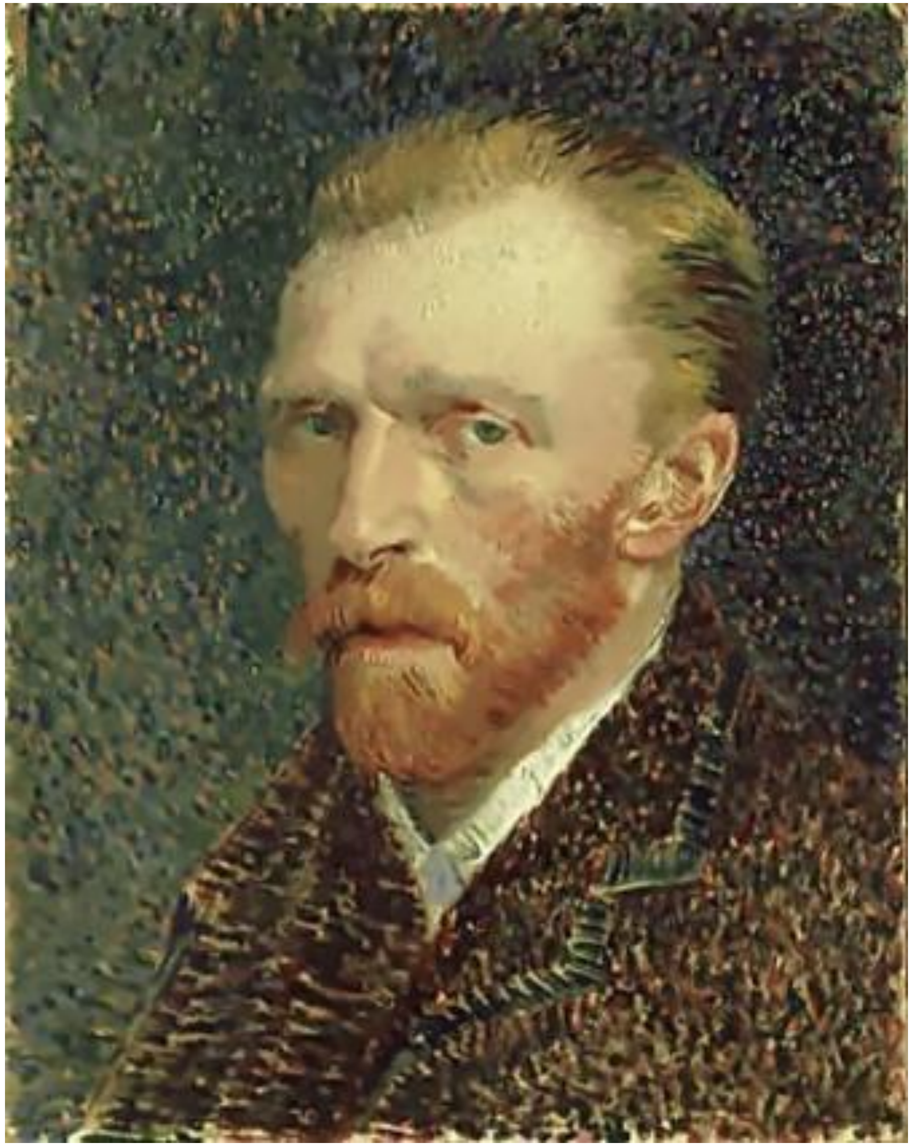
Experimental evaluation



Multiscale decomposition



Multiscale decomposition



$S_1(k = 5)$

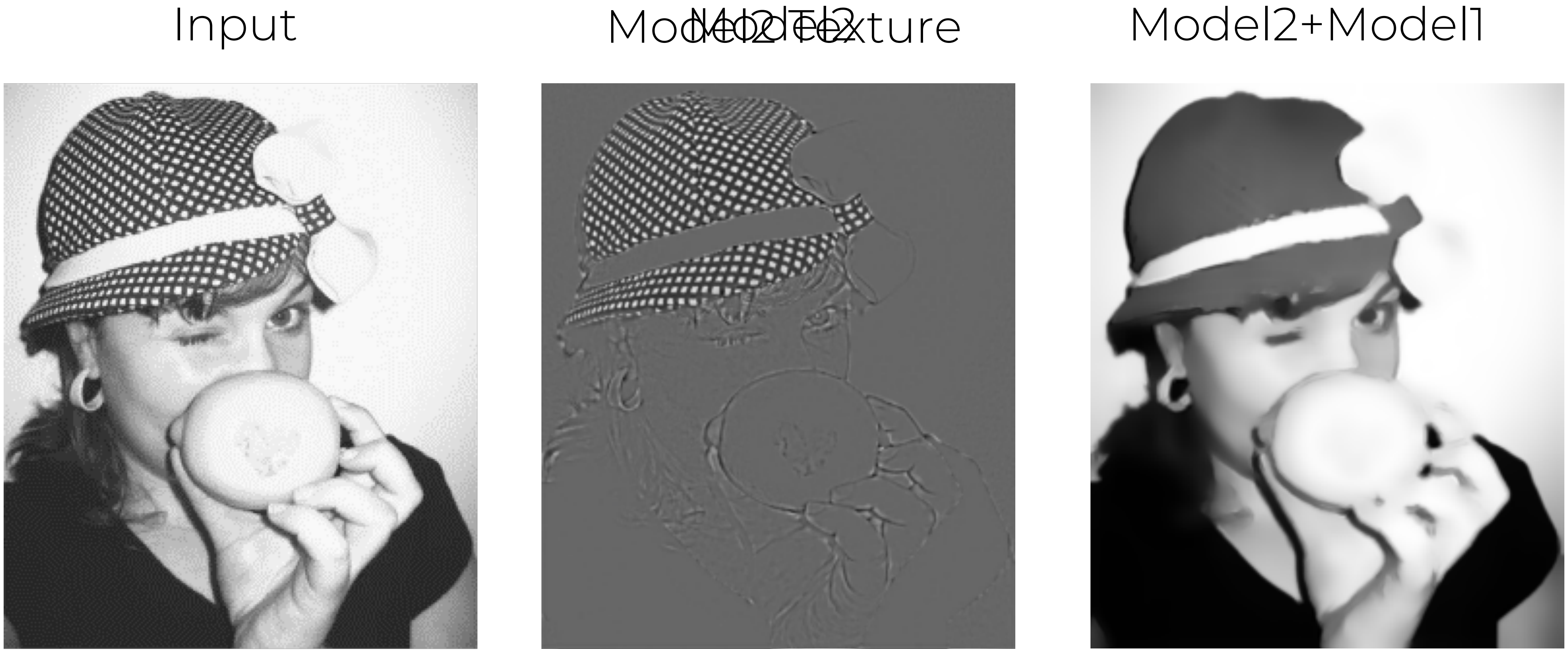
Multiscale decomposition

$S_2(k = 7)$

Multiscale decomposition

$S_3(k = 9)$

Challenging cases



Edge detection

Edge detection

Edge detection

Canny edges of original image

Canny edges of smoothed image

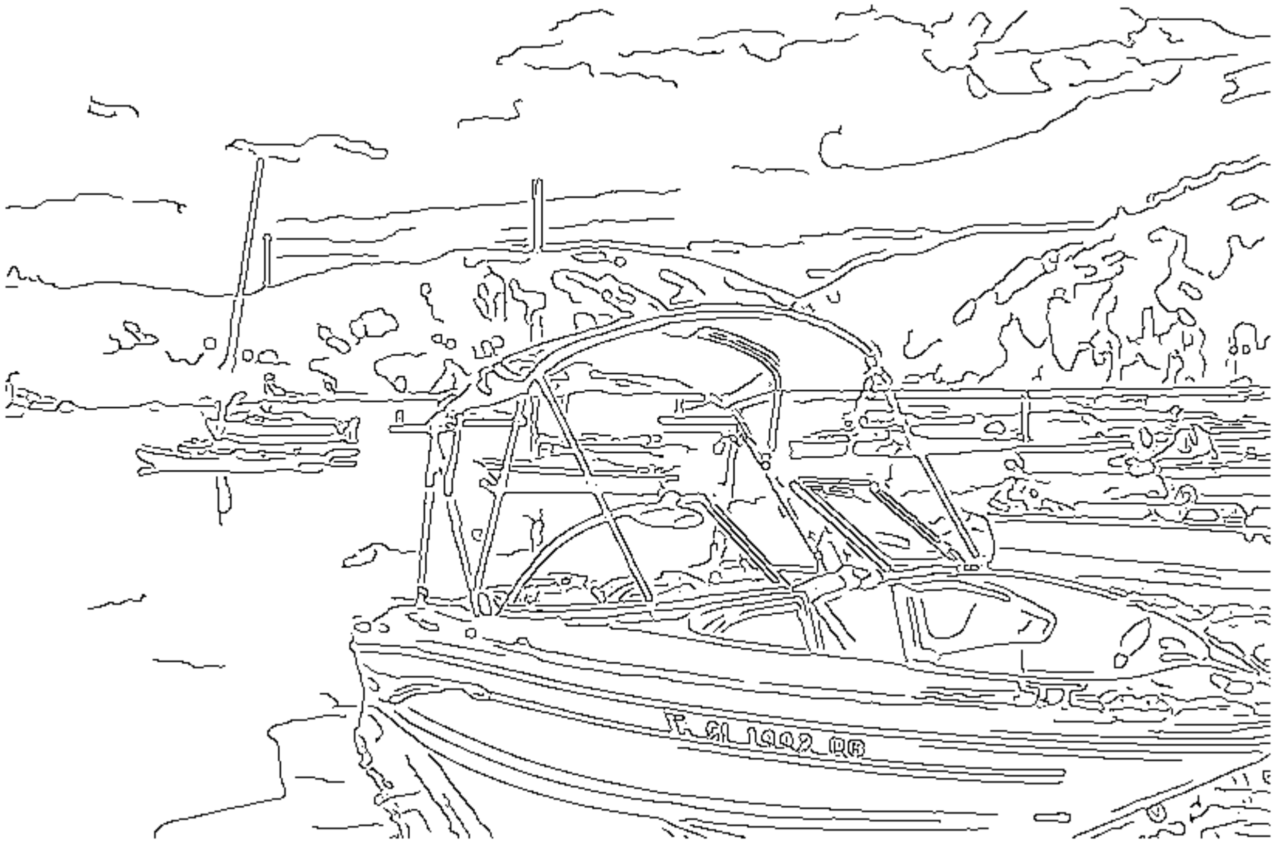


Image abstraction

Image abstraction

Detail boosting

Image composition