Point Processing & Filtering

CS 4243 Computer Vision & Pattern Recognition

Angela Yao

Recap & Outline

Last Week

- What is computer vision
- Applications in computer vision
- Why computer vision is difficult
- Historical development of the field
- Digital image capture and representations
- RGB & HSV colour spaces

Today's Lecture

- Point processing operations: brightness, contrast, normalization, non-linear mapping
- Histogram stretching, equalization, segmentation
- Linear filtering, cross-correlation vs. convolution operations
- Denoising, sharpening, template matching
- Non-linear filtering, e.g. median filters

Point Processing

Brightness, contrast, normalization, gamma, logarithmic & exponential mapping

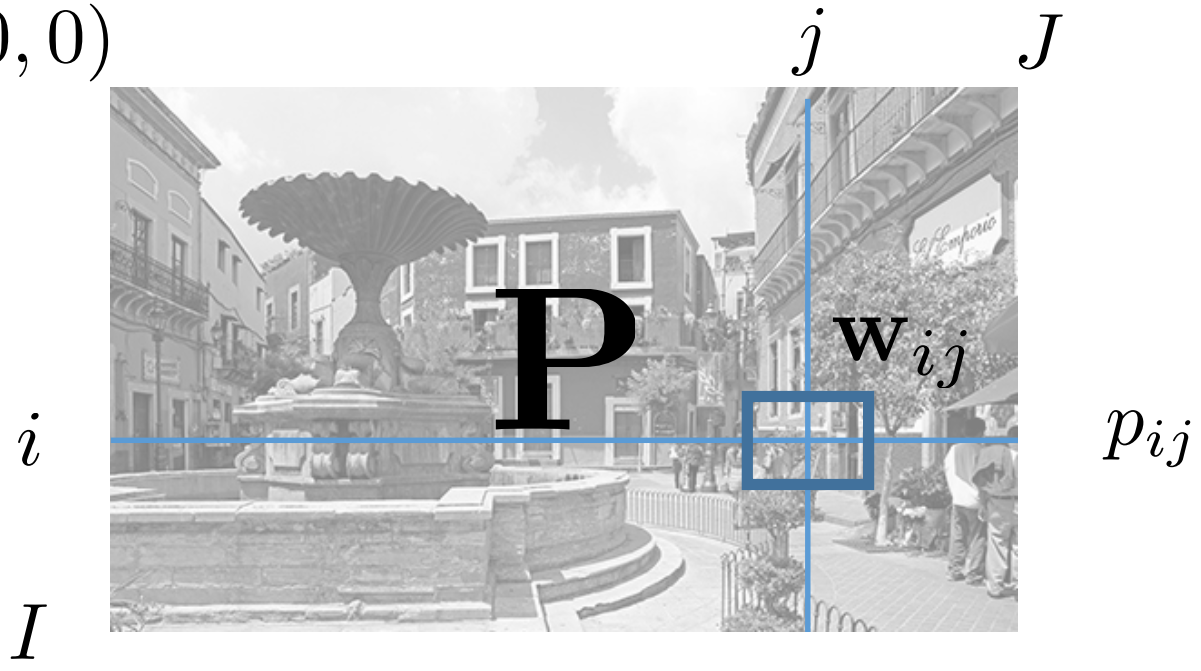
Image histograms, histogram stretching & equalization

Notation

$\mathbf{P}$: 2D array of pixel data

p_{ij} : element at i^{th} of I rows, j^{th} of J columns

$\mathbf{w}_{ij}$: window of data centered at i, j

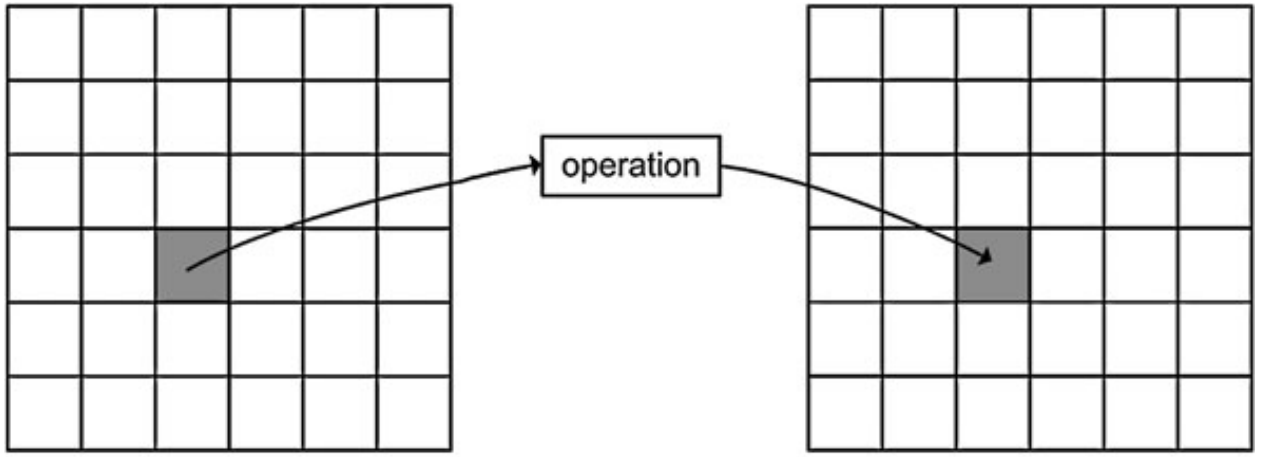


Transformations

Point processing

$$x_{ij} = f(p_{ij})$$

corresponding pixel from input image

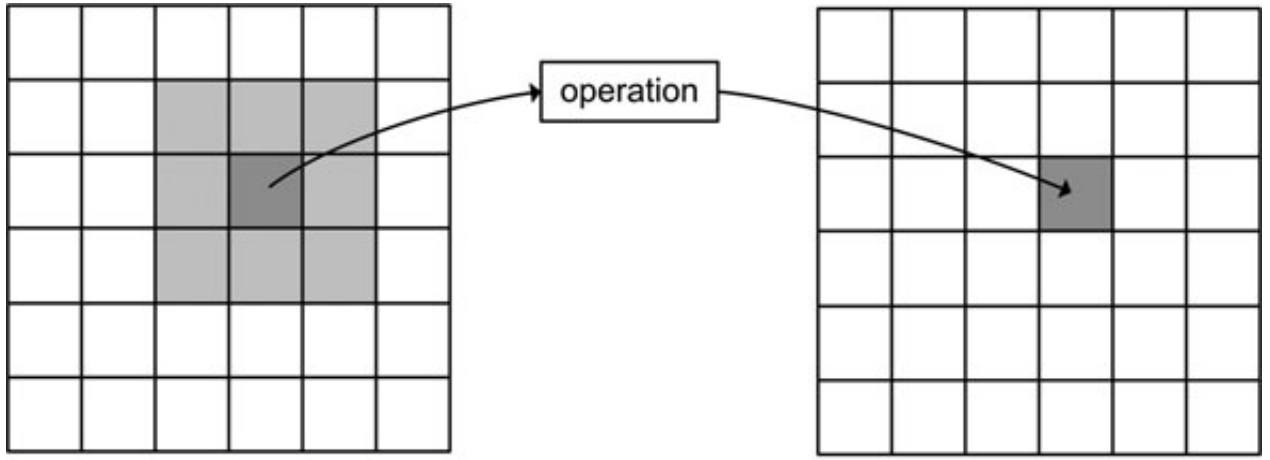


P: input image with elements p_{ij} **X**: output image with elements x_{ij}

Filtering

$$x_{ij} = f(\mathbf{w}_{ij})$$

local window of data from input



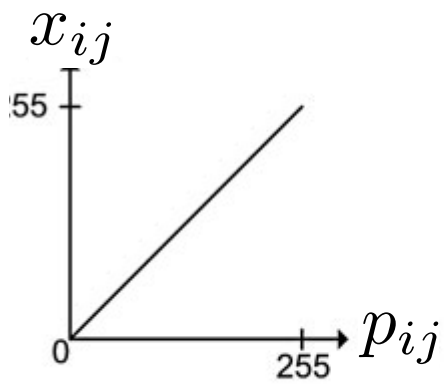
P: input image with elements p_{ij} **X**: output image with elements x_{ij}

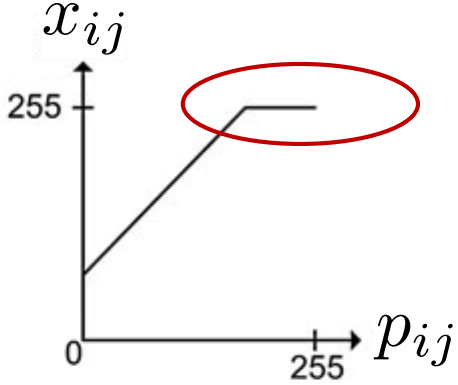
Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

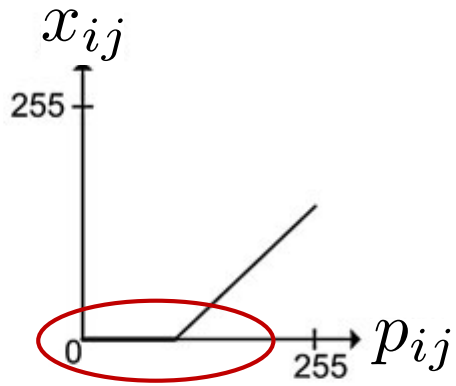
Adjusting Brightness

$$x_{ij} = p_{ij} + b$$

clipping behaviour!







original image

g(x,y), b = 0

no change

g(x,y), b = 75

increase brightness, b>0

g(x,y), b = -100

decrease brightness, b>0

Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Intensity Scaling

$$x_{ij} = a \cdot p_{ij}$$

a < 1

Decreased contrast

a = 1

Input image

a > 1

Increased contrast

This image on the left is not entirely representative. Scaling with a<1 decreases the contrast, but this example also has an increase in brightness which is why it looks so grey-ish (otherwise it should look more black). Please have a try with the demo python notebook to get a better idea.

Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Image Normalization (Whitening)

$$\text{mean} \quad \mu = \frac{\sum_{i=1}^{I} \sum_{j=1}^{J} p_{ij}}{IJ}$$

$$\text{variance} \quad \sigma^2 = \frac{\sum_{i=1}^{I} \sum_{j=1}^{J} (p_{ij} - \mu)^2}{IJ}$$

resulting image pixels are zero-mean, unit variance

$$x_{ij} = \frac{p_{ij} - \mu}{\sigma}$$

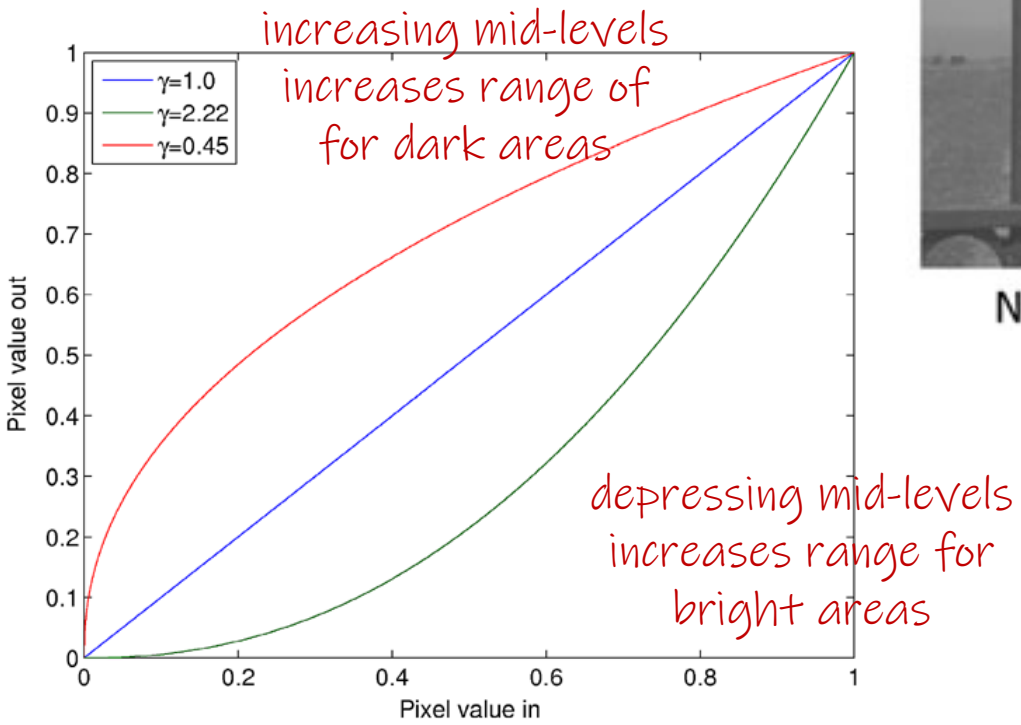
Removes contrast and constant additive luminance variations

Original

Whitened

Gamma Mapping

Mappings can also be non-linear, e.g. $x_{ij} = 255 \cdot \left(\frac{p_{ij}}{255}\right)^{\gamma}, \ \gamma > 0$



No gamma correction

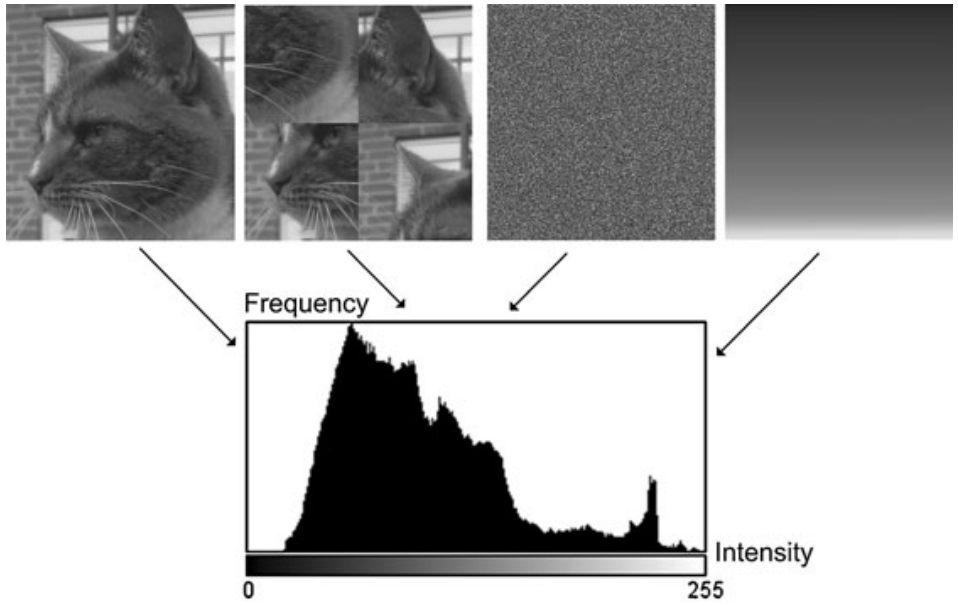
Gamma value: 2.22

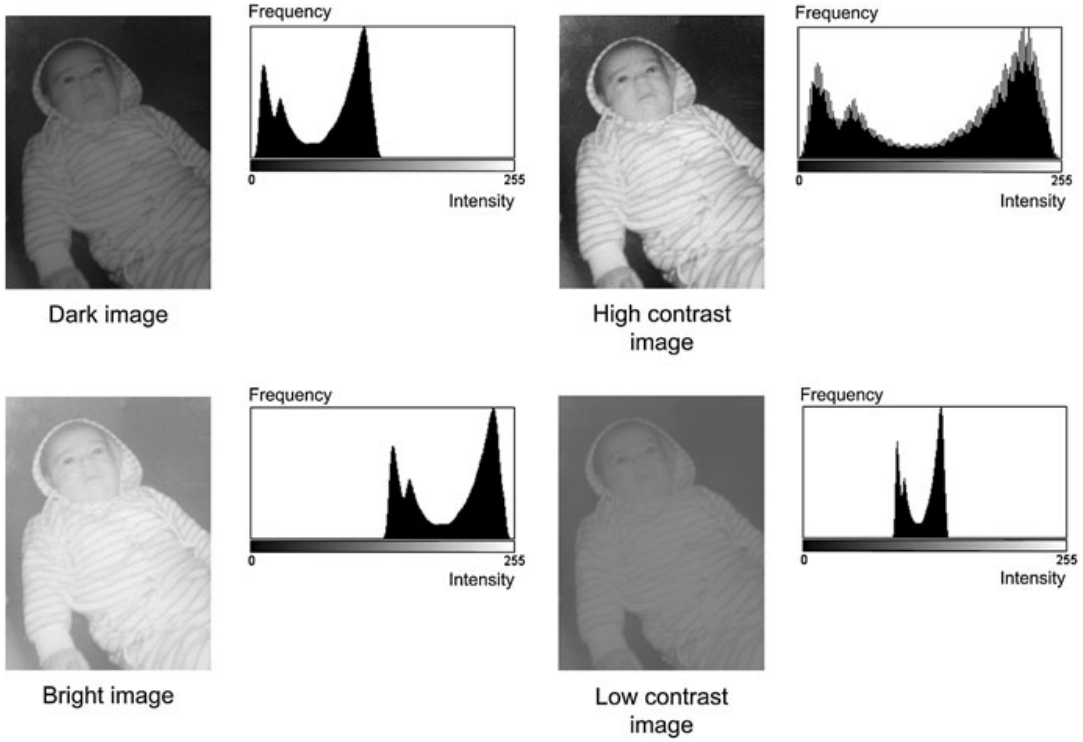
Gamma value: 0.45

Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Image Histogram

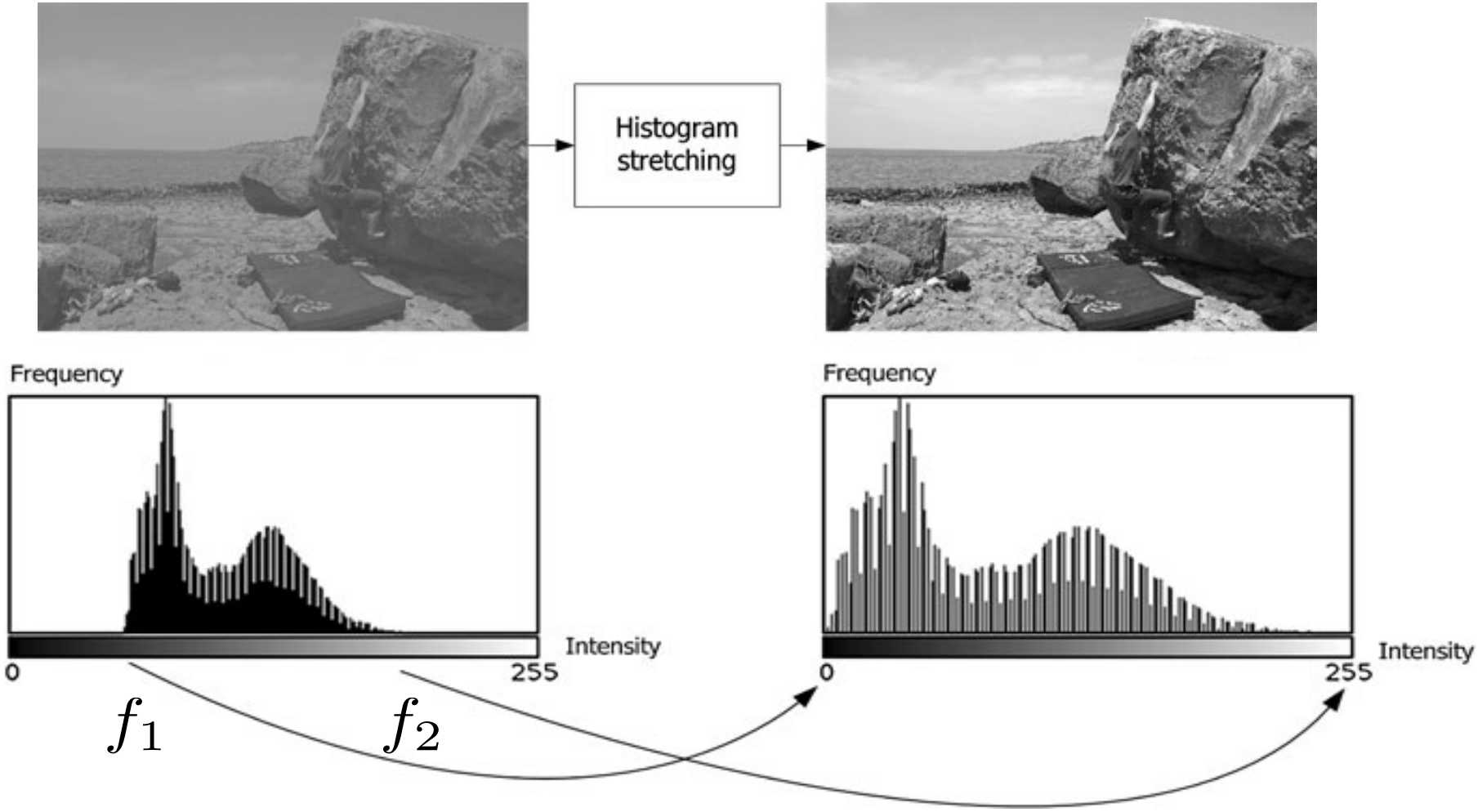
- Information is global (pixel locations don't matter)





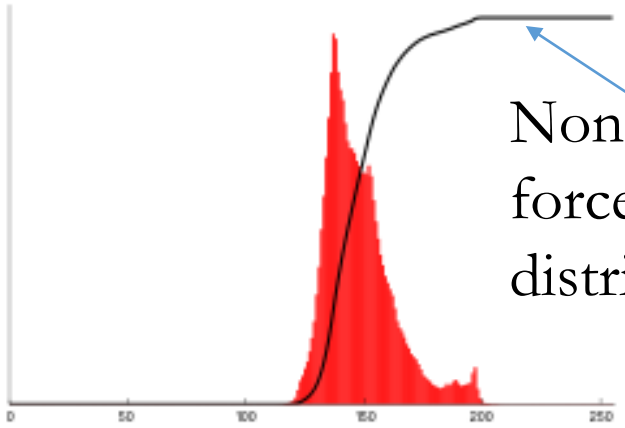
Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Histogram Stretching

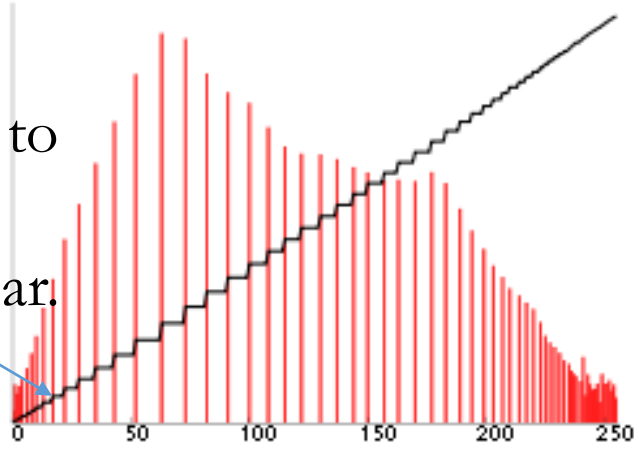


Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Histogram Equalization



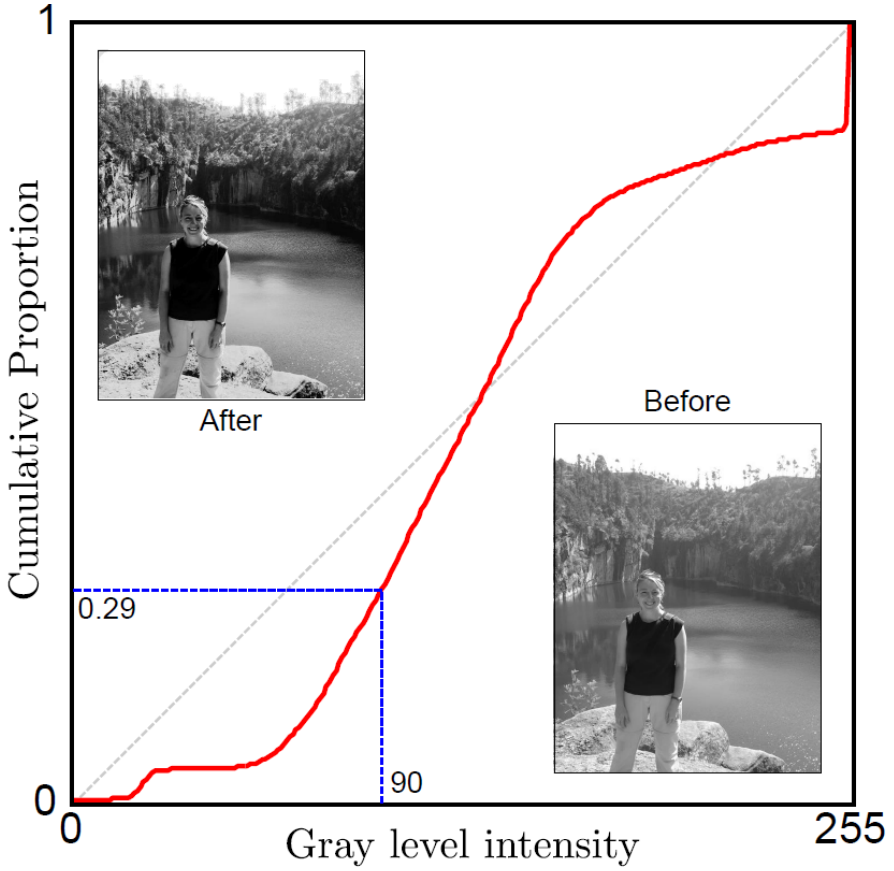
Non-linear operation to force the cumulative distribution to be linear.



https://en.wikipedia.org/wiki/Histogram_equalization

Histogram Equalization

Forces the cumulative distribution to be linear.



histogram

$$h_k = \sum_{i=1}^{I}\sum_{j=1}^{J} \delta[p_{ij} - k],$$

Cumulative sum of histogram ("CDF")

$$c_k = \frac{\sum_{l=1}^{k} h_l}{IJ}.$$

K is usually the maximum intensity, e.g. 255

$$x_{ij} = K c_{p_{ij}}.$$

Look up for each pixel intensity, the corresponding cumulative proportion value and remap e.g. 90 maps to 255*0.29 = 74

Equalization vs. Whitening

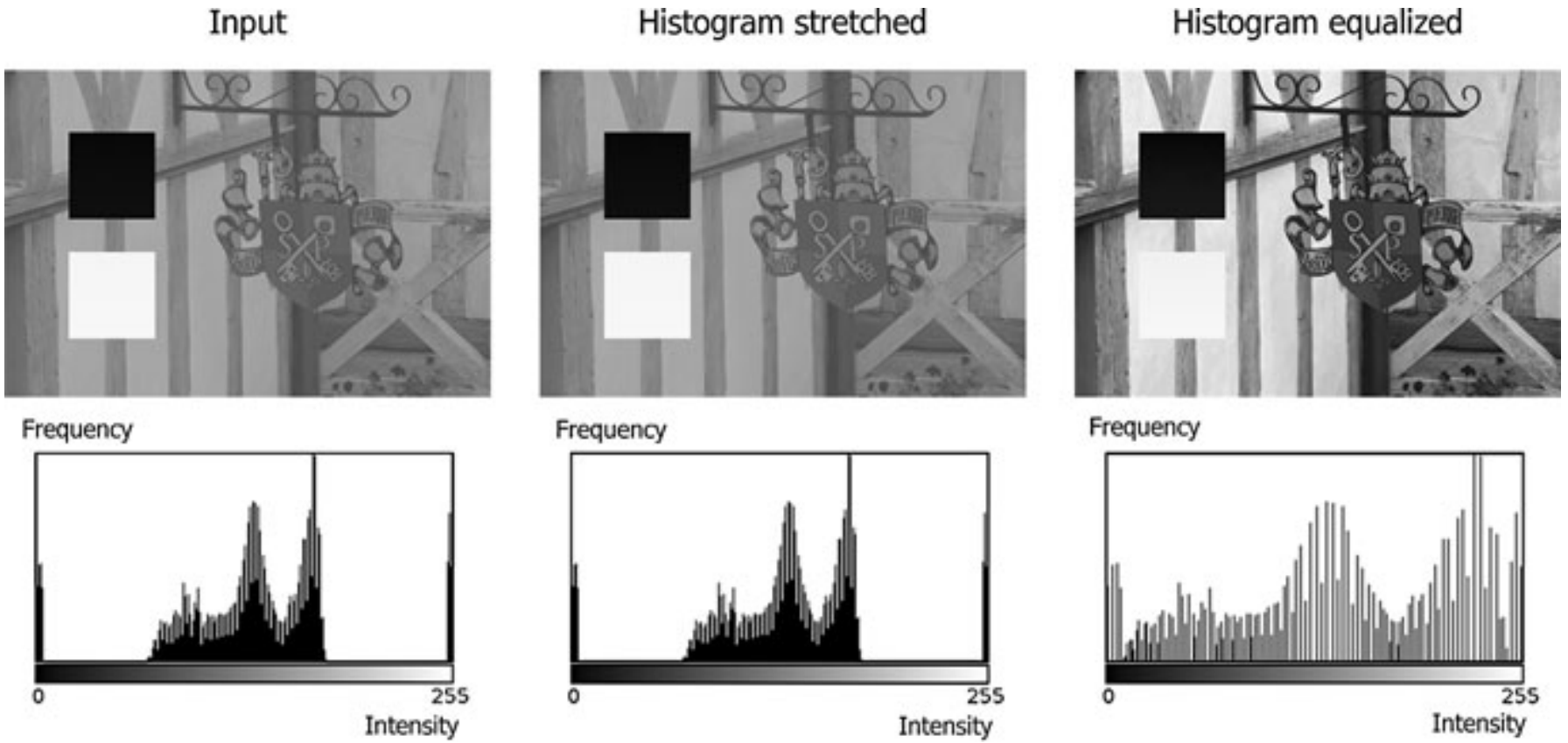
Equalization boosts the contrast more than whitened data – however, equalization is a more expensive operation.

Original

Whitened

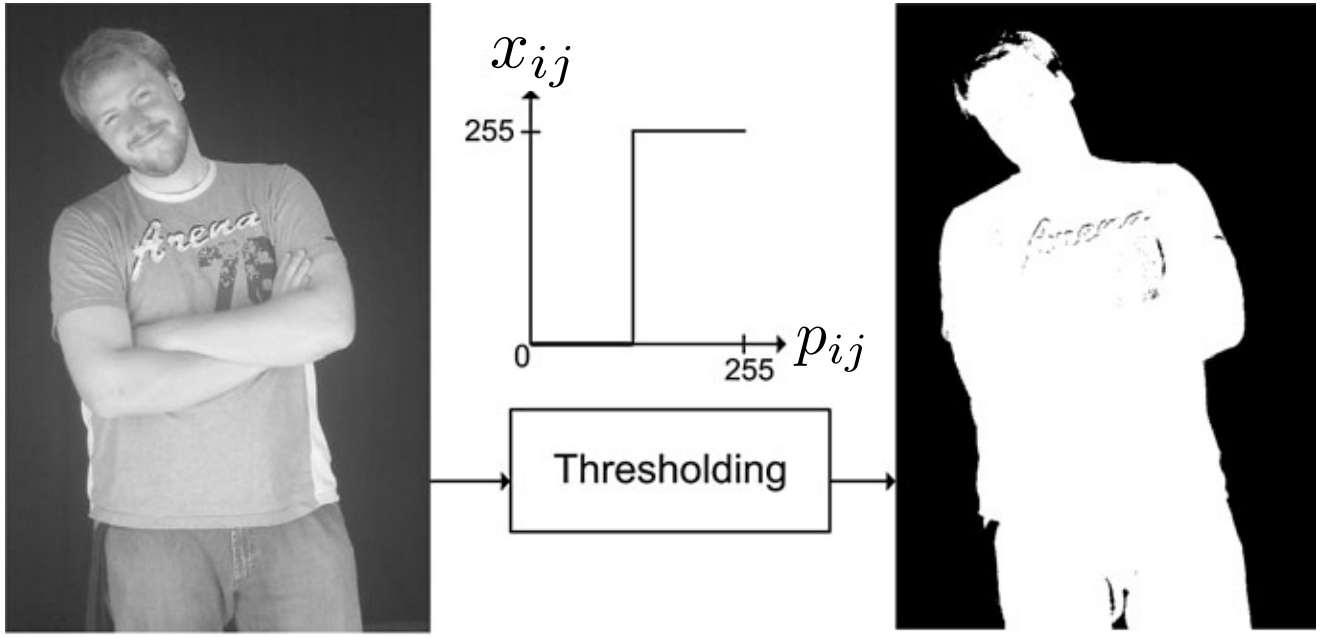
Histogram Equalized

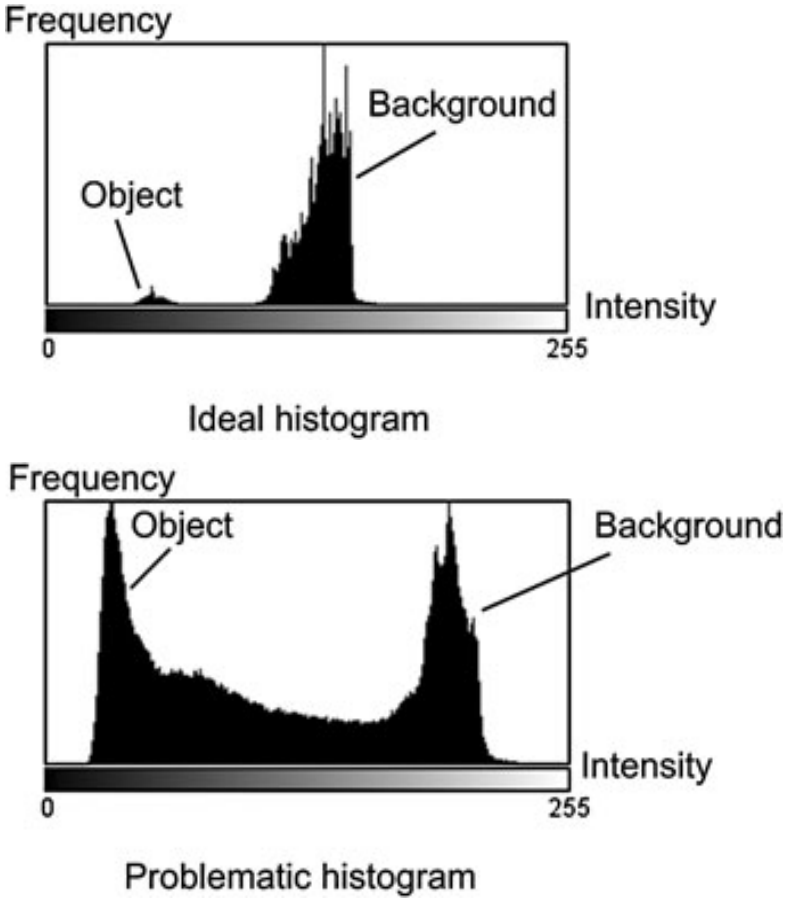
Equalization vs. Stretching



Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

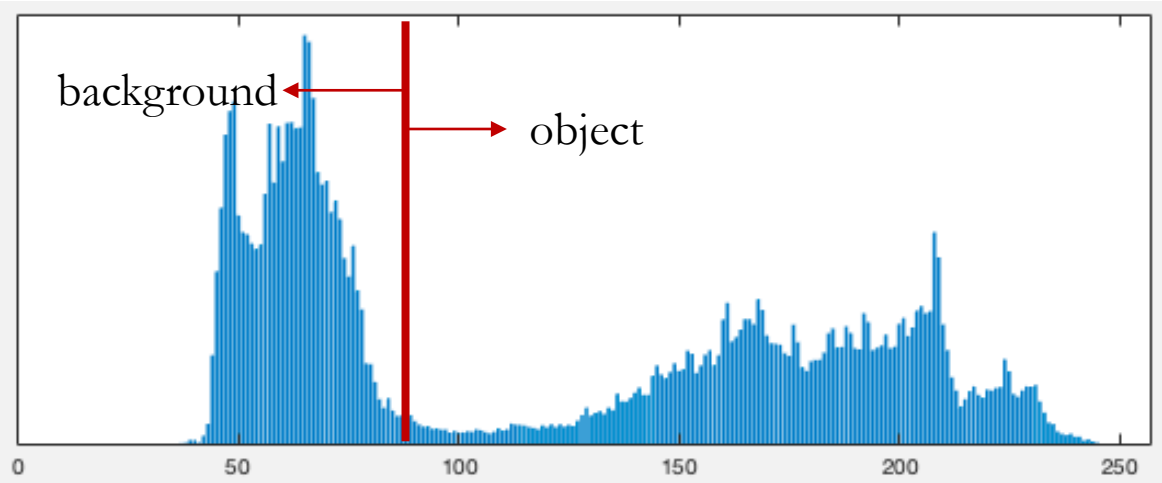
What are Histograms Good For?





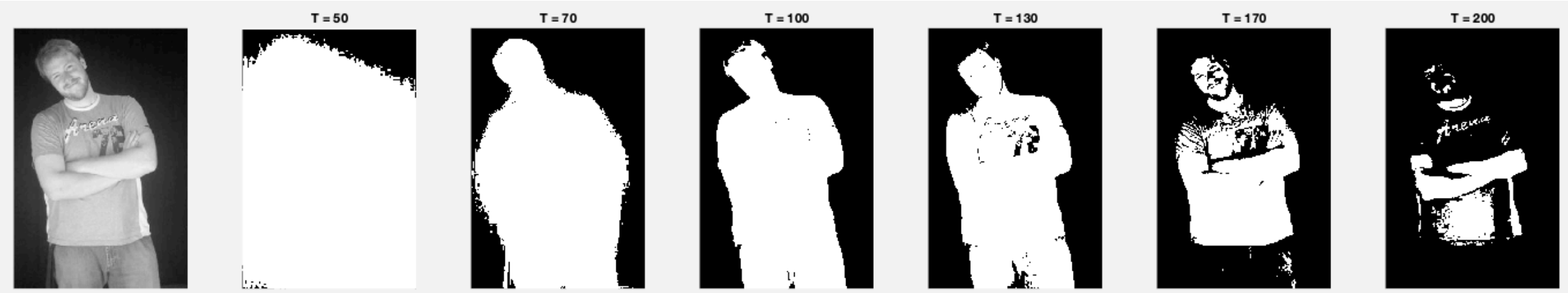
Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Histogram Thresholding for FG/BG Separation

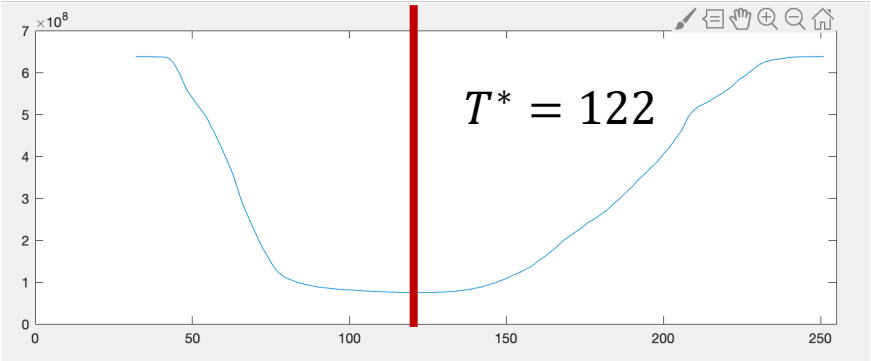


The threshold which separates the object from the background depends on the image. Can we choose the right threshold automatically?

If the image intensity histogram is bi-modal (two "humps"), then the threshold should sit somewhere in between the two modes.



Otsu's Method – Automated Thresholding



$$T^* = \underset{T}{\operatorname{argmin}}\ w_1(T) \cdot \sigma_1^2(T) + w_2(T) \cdot \sigma_2^2(T)$$

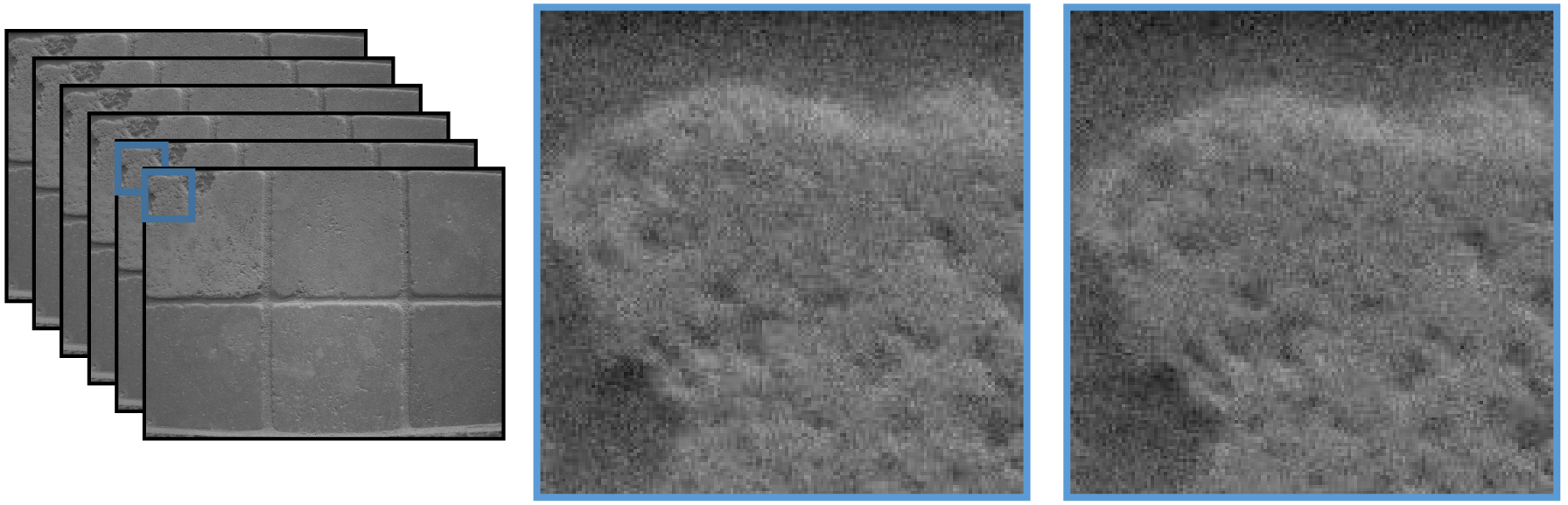
- optimal threshold T^* minimizes the sum of weighted variances of object and background.
- rationale:
  - correct threshold produces two narrow modes
  - incorrect threshold produces (at least one) wide mode
  - width of the mode is measured by the variance σ^2
  - weighting by number of pixels in each mode
- $\sigma_1^2(T), \sigma_2^2(T)$ - variance of pixels less than or equal to and greater than threshold respectively
- $w_1(T), w_2(T)$ - number of pixels less than or equal to and greater than threshold

Correlation Filters

Types of image noise; cross-correlation operation

Box & Gaussian smoothing; sharpening & template matching

Motivation: Noise Reduction



Even multiple images of the **same static scene** will not be identical.

Why not? Where does noise come from?

Slide Credit: K. Grauman

Common types of noise

Impulse & salt and pepper noise: random occurrences of white (and black) pixels, usually due to defective sensor elements

Gaussian noise: variations in intensity drawn from a Gaussian distribution, due to inherent noise of the sensor

$$p_{ij} = \hat{p}_{ij} + \eta \quad \eta \sim \mathcal{N}(\mu_n, \sigma_n)$$

observation — ideal image — noise

Original

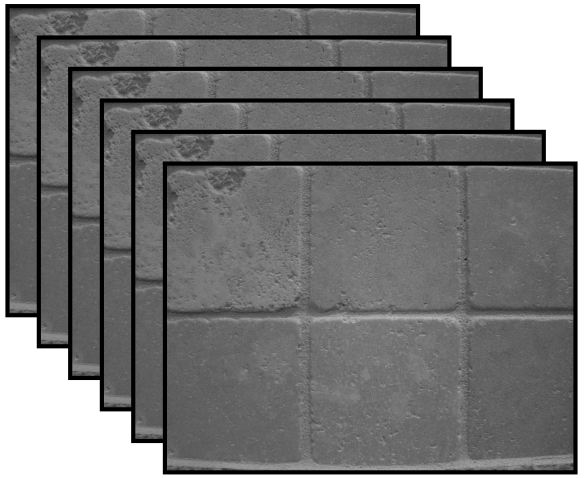
Impulse noise

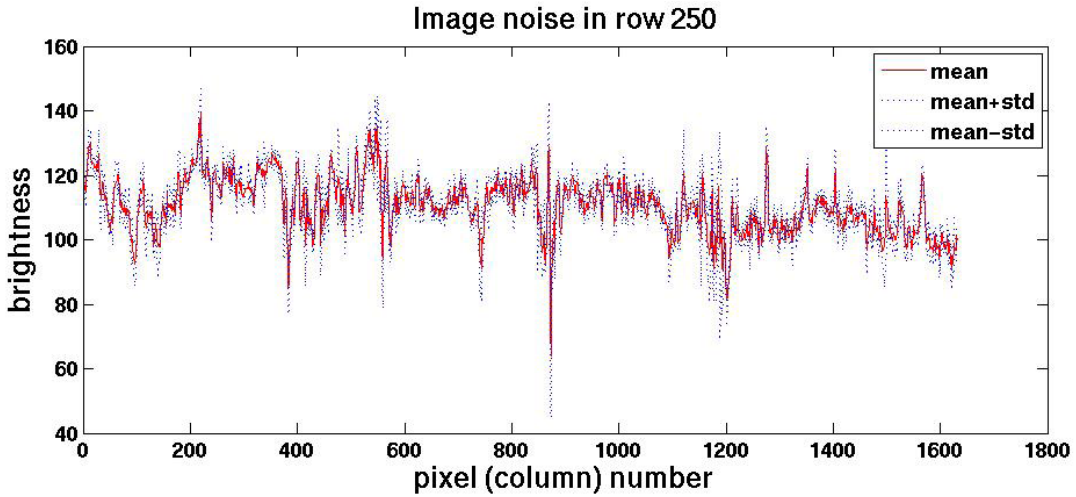
Salt and pepper noise

Gaussian noise

Adapted from S. Seitz

Motivation: Noise Reduction





How could we reduce the noise, give an estimate of the true intensities?

What if there's only one image?

First attempt: Replace each pixel with average of all the values in its neighborhood.

This assumes that

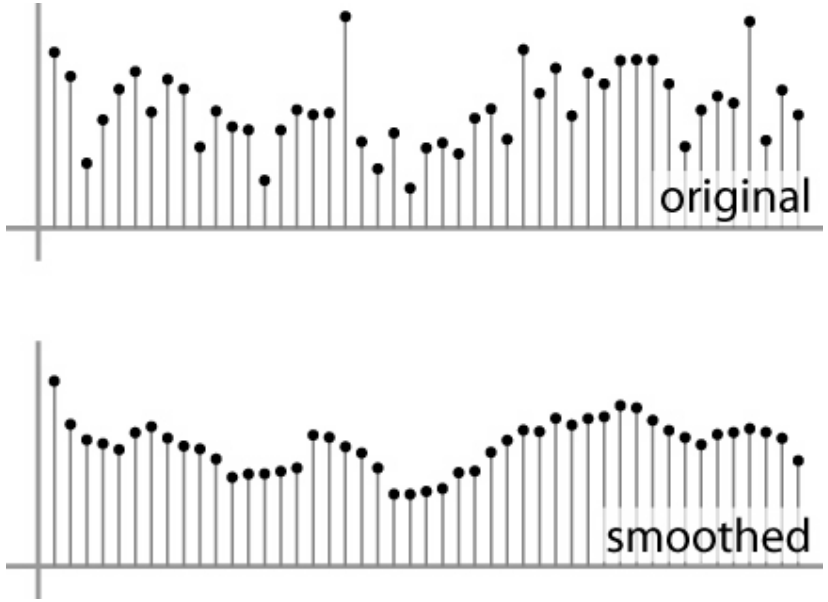
- (True) pixel values are similar to neighbours
- noise processes are independent from pixel to pixel.

Adapated from K. Grauman

First Solution Attempt

Replace each pixel with an average of all the values in its neighborhood

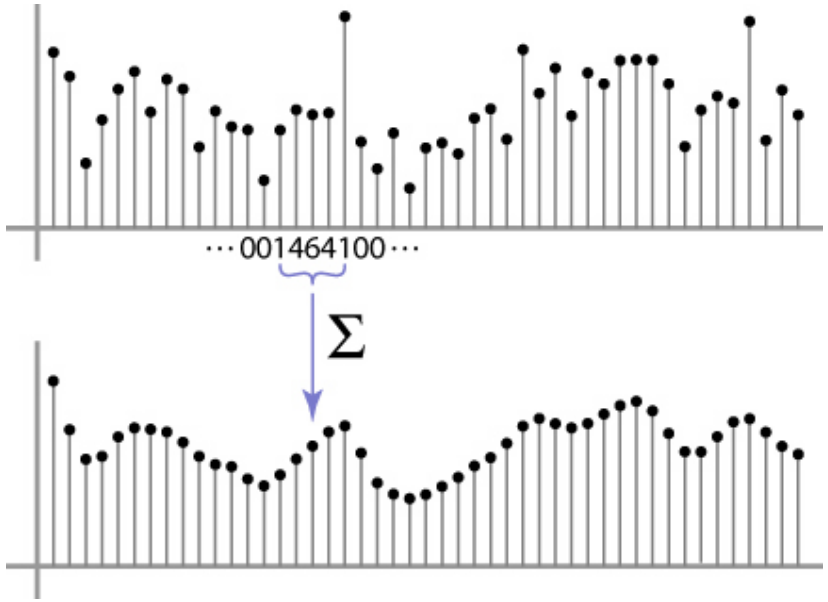
Moving average in 1D:



Slide Credit: S. Marschner

Weighted Moving Average

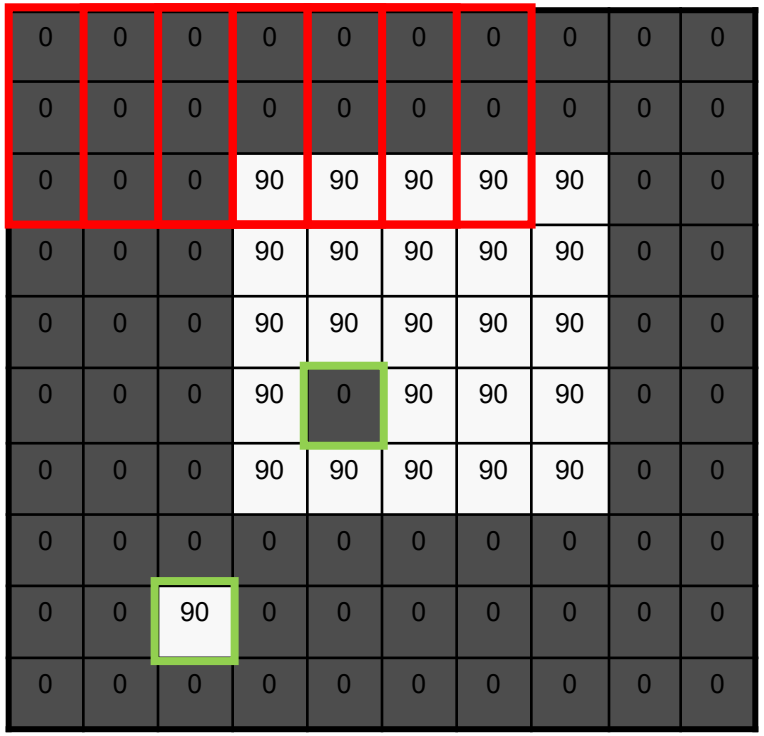
- We can weight the moving average
  - Uniform weights: [1, 1, 1, 1, 1] /5
  - Non-uniform weights [1, 4, 6, 4, 1] / 16



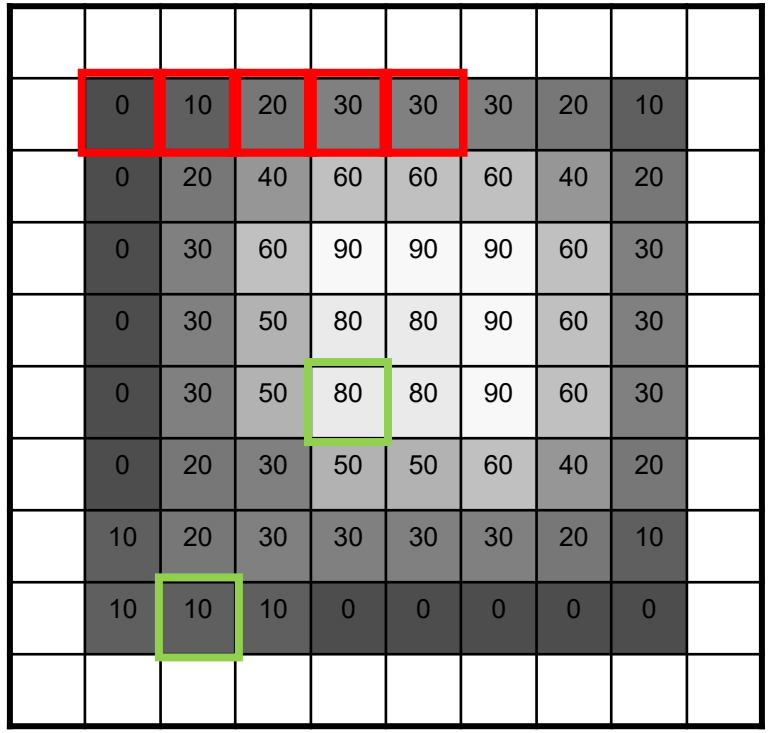
Slide Credit: S. Marschner

Moving Average in 2D

P



X



gives uniform weight to each pixel

$$x_{ij} = \frac{1}{2k+1}^2 \sum_{u=-k}^{k} \sum_{v=-k}^{k} p_{i+u,j+v}$$

sum all pixels in (2k+1)*(2k+1) neighborhood around p_{ij}

Slide Credit: S. Seitz

Correlation filtering

Generalize to allow different weights depending on neighboring pixel's relative position:

Can further generalize, no need to be square of (2k+1)x(2k+1)

$$x_{ij} = \sum_{u=-k}^{k} \sum_{v=-k}^{k} f_{uv} \cdot p_{i+u,j+v}$$

Weight depends on neighbour's relative position wrt p_{ij}

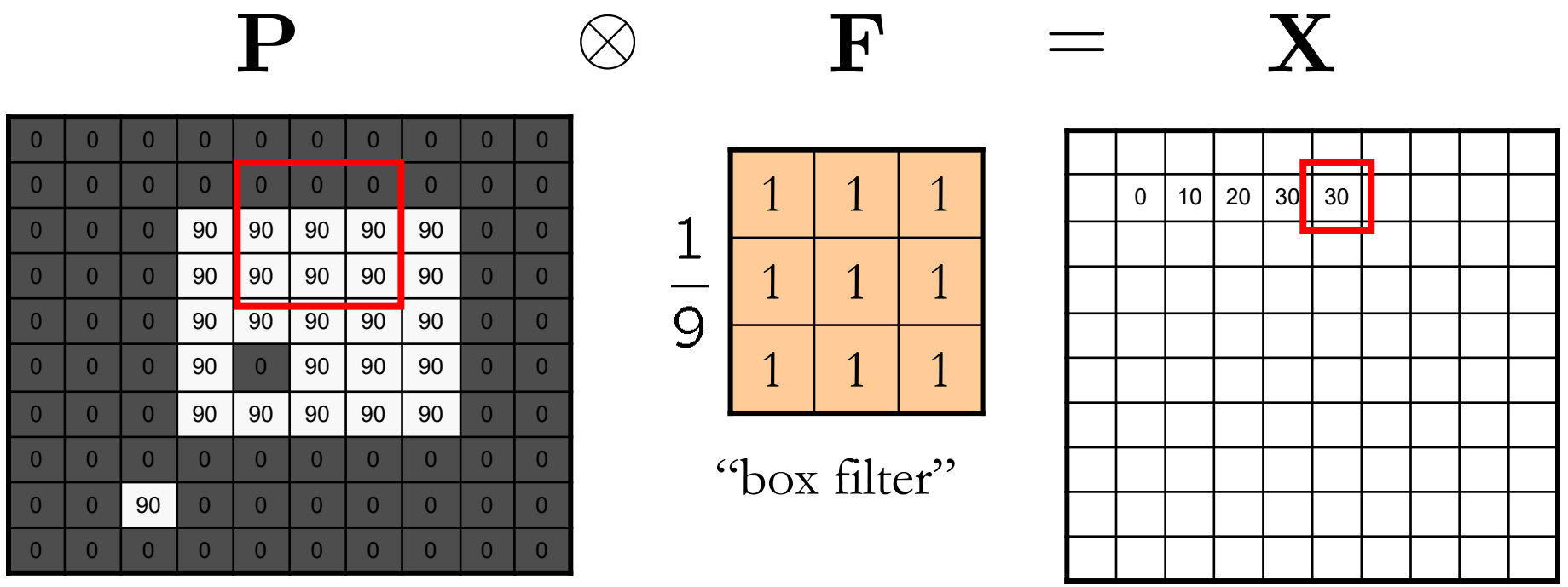
This is called cross-correlation, denoted $\mathbf{X} = \mathbf{P} \otimes \mathbf{F}$

Cross-correlation is a linear operation.

Filter Kernel (stores the weights).

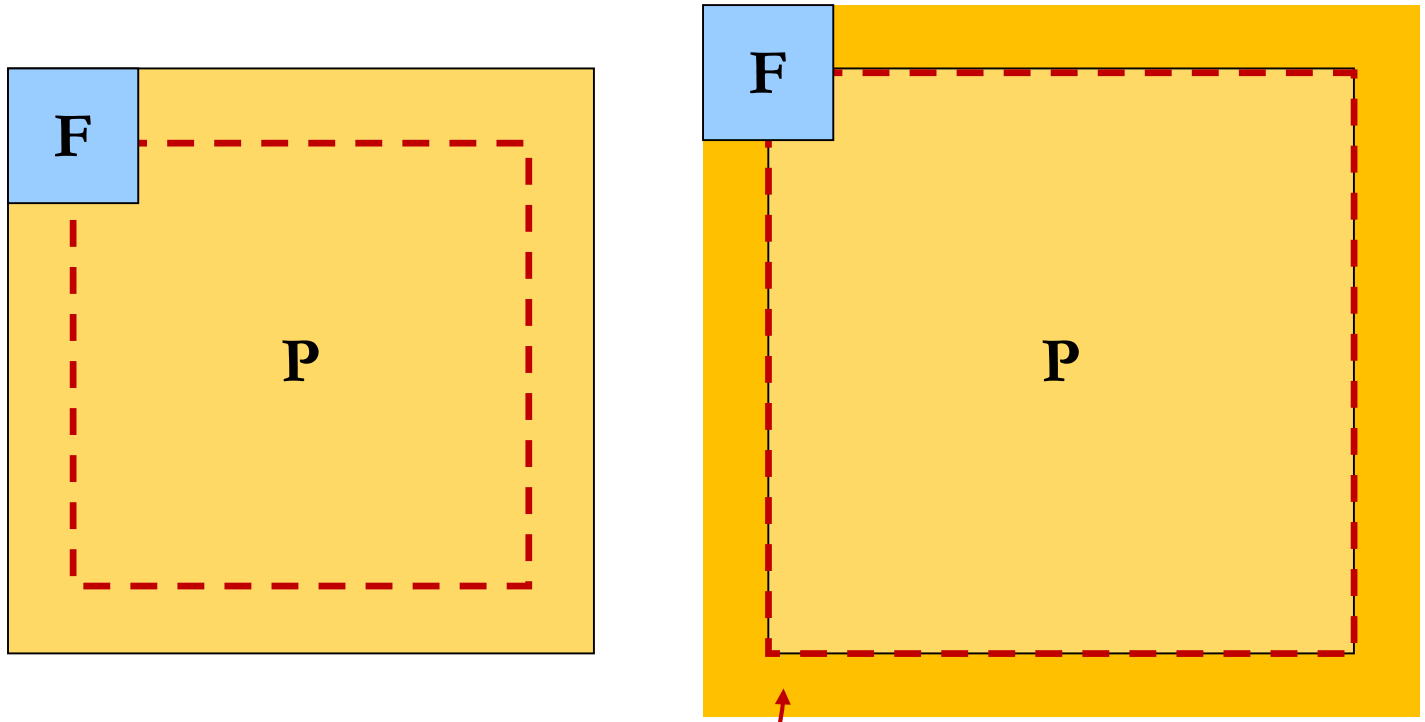
Averaging filter (Box filter)

Which values belong in the kernel *F* for the moving average example?



Slide Credit: K. Grauman

Output Size & Boundaries



- zero-padding
- wrap around
- copy edge
- reflect across edge

Adapted from K. Grauman

Gaussian filter

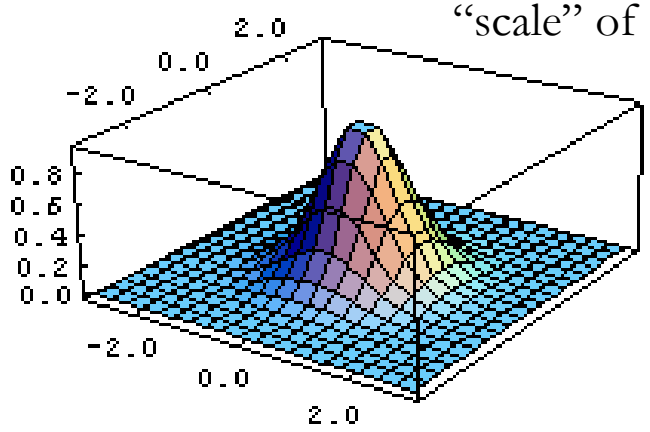
What if we want nearest neighboring pixels to have the most influence on the output?

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

$$f_{uv} = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$

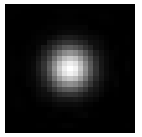
"scale" of Gaussian



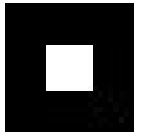
Both the box filter and the Gaussian filter has a smoothing effect.

Slide Credit: S. Seitz

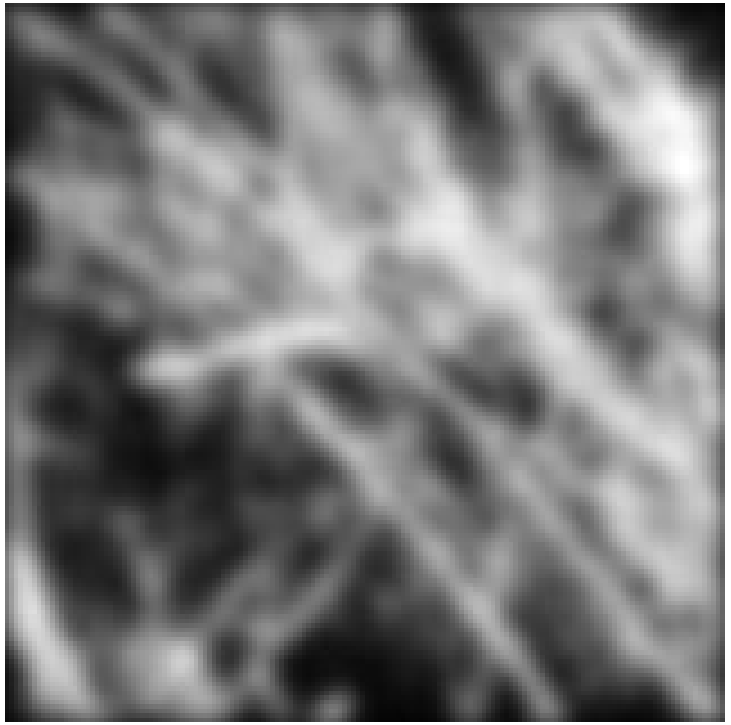
Smoothing with Gaussian vs. Box



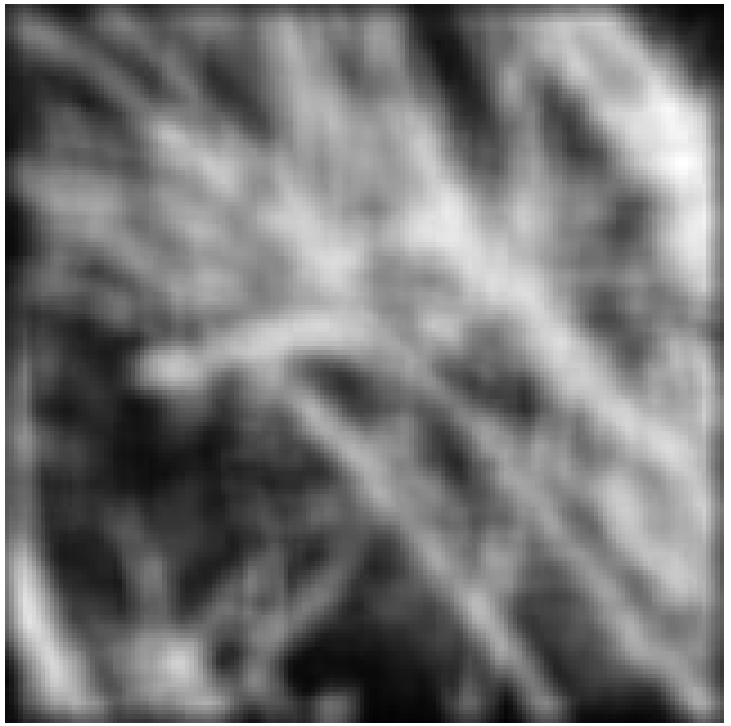
depicts filter kernels:
white = high value
black = low value



original



filtered
(Gaussian kernel)



filtered
(Box kernel)

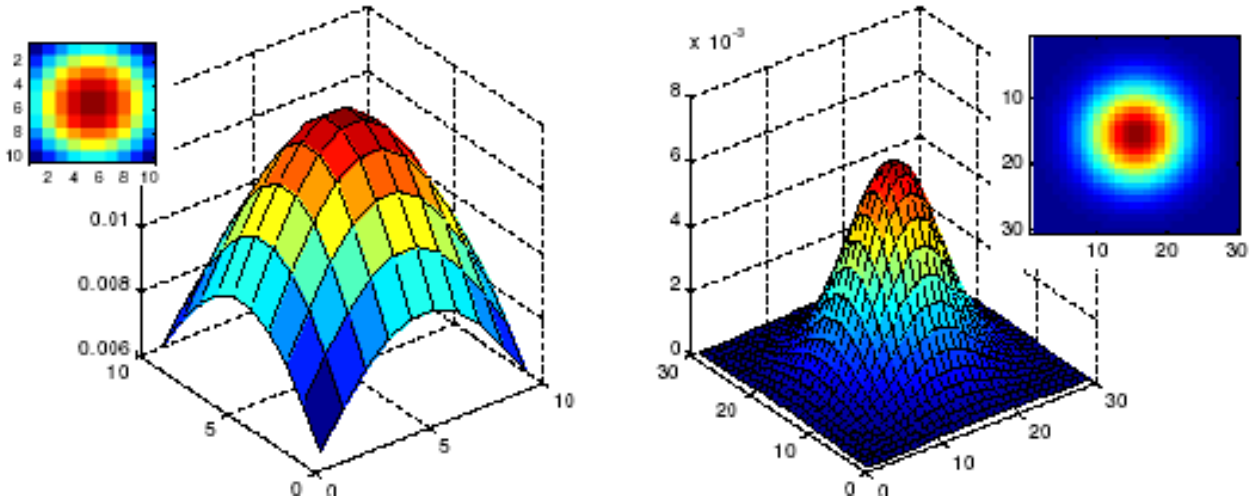
Slide Credit: K. Grauman

Gaussian filters

Which parameters matter?

Size of kernel?

Note, Gaussian function has infinite support, but discrete filters use finite kernels



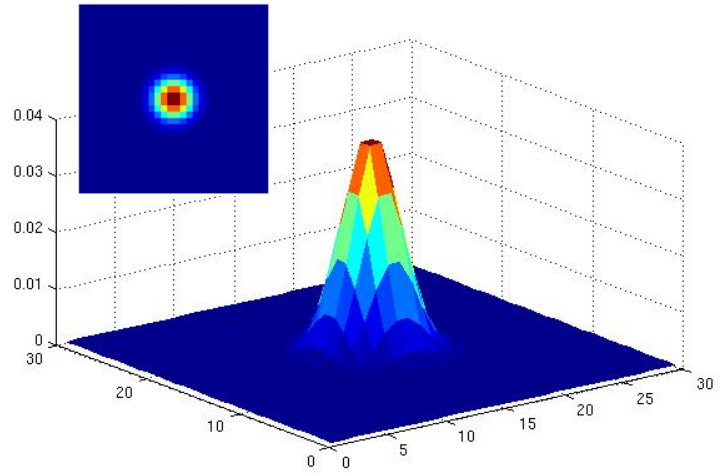
σ = 5 with 10 x 10 kernel σ = 5 with 30 x 30 kernel

Slide Credit: K. Grauman

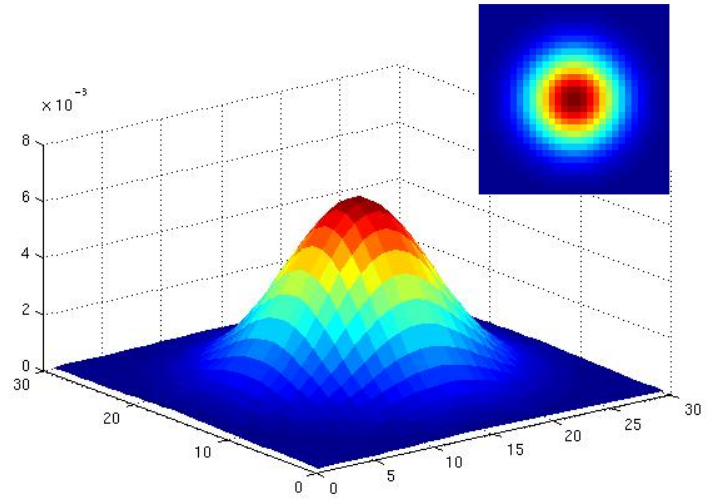
Gaussian filters

Which parameters matter?

Variance of Gaussian (σ): determines extent of smoothing



σ = 2 with 30 x 30 kernel

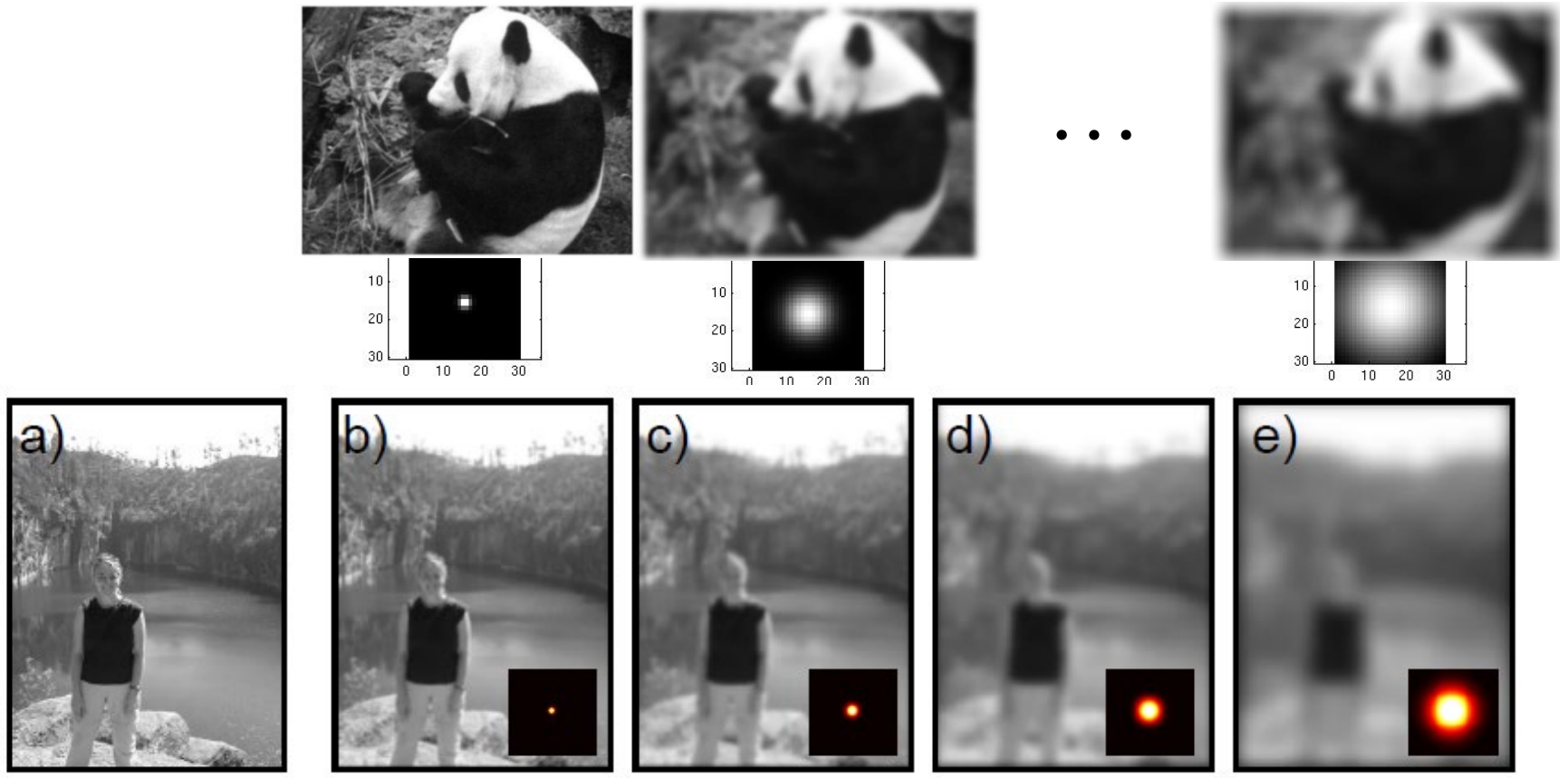


σ = 5 with 30 x 30 kernel

Slide Credit: K. Grauman

Smoothing with a Gaussian

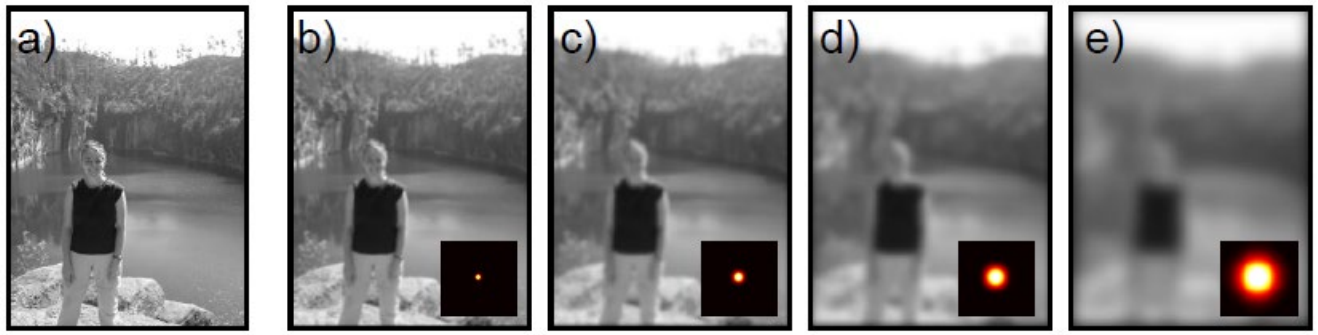
σ is the "scale" of the Gaussian kernel, and controls the amount of smoothing.



Slide Credit: K. Grauman

Gaussian Noise vs. Gaussian Smoothing

- Gaussians are used as a filter kernel applied for blurring or smoothing



- Gaussians are also used to model the noise processes.

$$p_{ij} = \hat{p}_{ij} + \eta \qquad \eta \sim \mathcal{N}(\mu_n, \sigma_n)$$

observation — ideal image — noise

In real life, the "ideal image" is not obtainable. We show (manually) corrupted versions to illustrate the impact of additive Gaussian noise.

- You can use Gaussian smoothing (or any other spatial smoothing kernel) to reduce the effects of Gaussian noise ☺

Practice with Correlation Filters

Original

$\otimes$

0	0	0
0	1	0
0	0	0

Filtered (no change)

Original

$\otimes$

0	0	0
0	0	1
0	0	0

Filtered (shifts to the left)

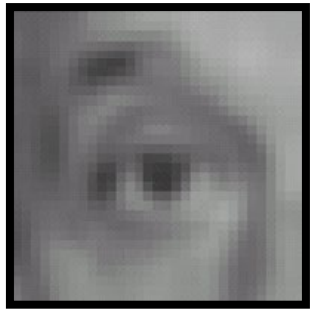
Source: D. Lowe

Practice with Correlation Filters

$$-\frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & -17 & 1\\ 1 & 1 & 1\end{bmatrix}$$

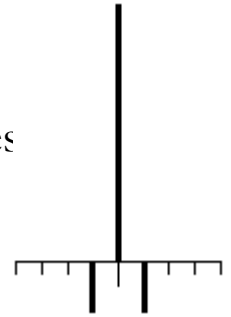
Original

$$\otimes\left(\begin{bmatrix}0 & 0 & 0\\ 0 & 2 & 0\\ 0 & 0 & 0\end{bmatrix} - \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}\right)$$



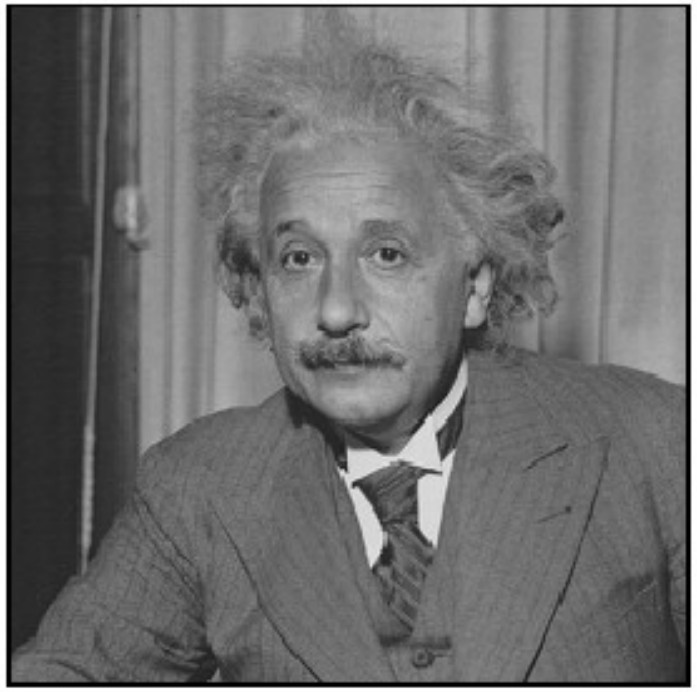
Blur (with a box filter)

Sharpening filter:
accentuates differences
with local average

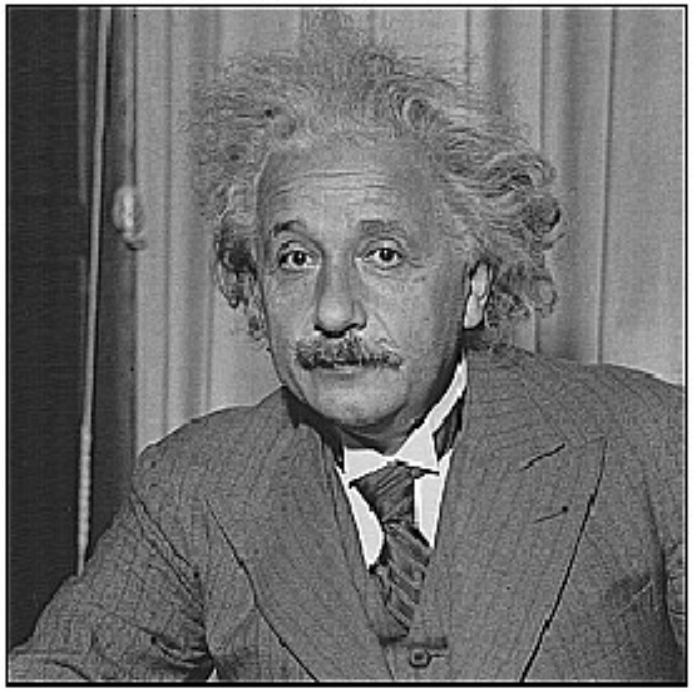


Source: D. Lowe

Understanding Sharpening



before



after

- do nothing for flat areas
- stresses intensity peaks and differences with respect to the surroundings

Adapted from K. Grauman

Understanding Sharpening

input

filter

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

output

sharpening

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Response if region is "flat"?

$$\begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Region has an edge?

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Region has noise?

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

No change

$$\frac{1}{9}\begin{bmatrix} 0 & -2 & 14 & 12 & 14 \\ 0 & -3 & 12 & 9 & 12 \\ 0 & -3 & 12 & 9 & 12 \\ 0 & -3 & 12 & 9 & 12 \\ 0 & -2 & 14 & 12 & 14 \end{bmatrix}$$

Edge gets emphasized

$$\frac{1}{9}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & -1 & -1 & 0 \\ 0 & -1 & 17 & -1 & 0 \\ 0 & -1 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

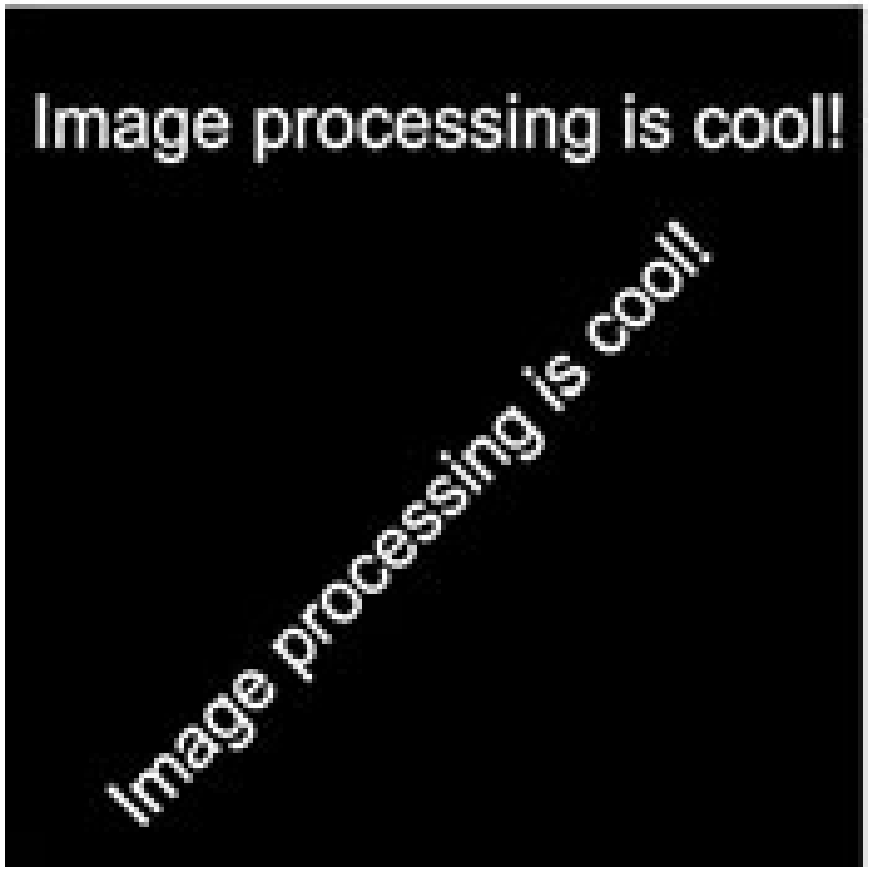
Noise also emphasized ☹

Template Matching

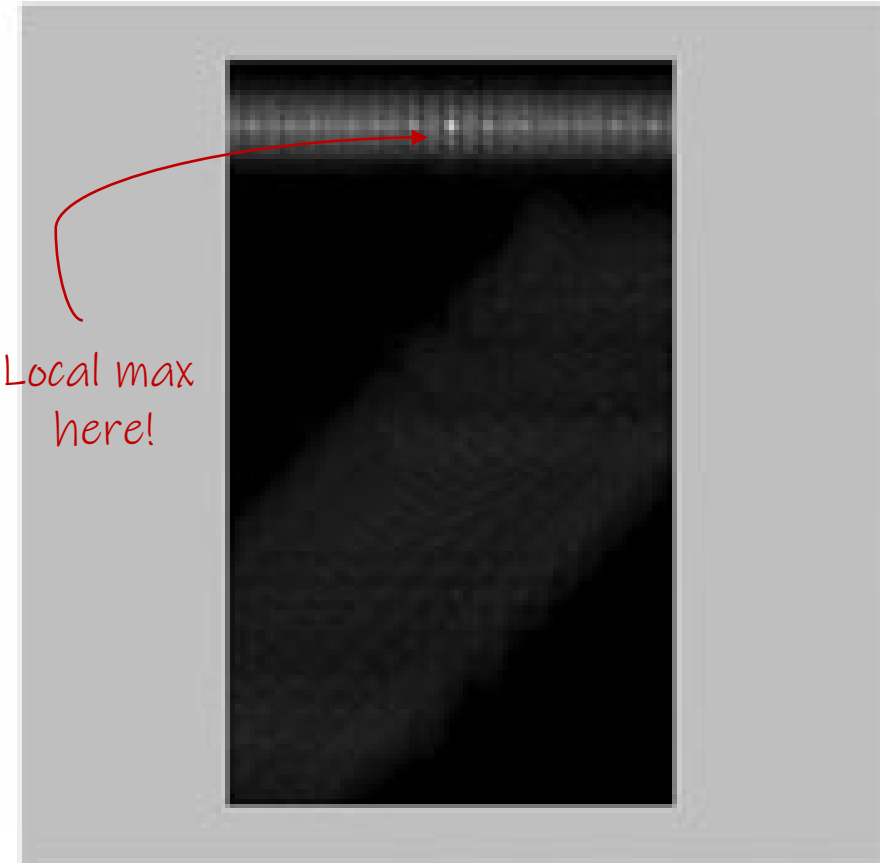
Input image

Matrix of numbers

"filter kernel" i.e. matrix of numbers

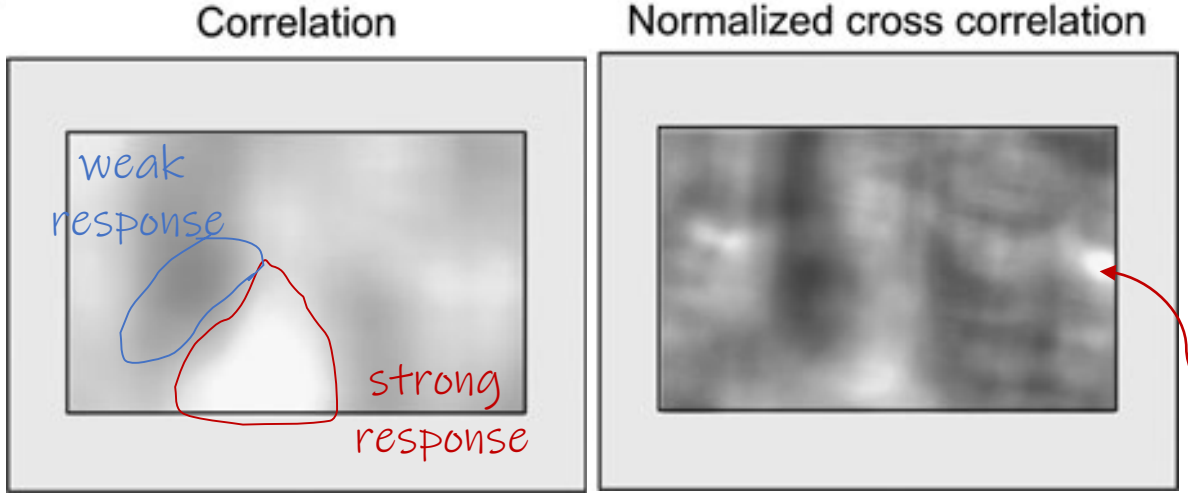


Correlation result



Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Normalized Cross-Correlation



[0-255]!

Scale to sum up to 1

$$x_{ij} = \sum_{u=-k}^{k} \sum_{v=-k}^{k} f_{uv} \cdot p_{i+u,j+v}$$

$$x_{ij} = \frac{1}{|\mathbf{F}||\mathbf{w}_{ij}|} \sum_{u=-k}^{k} \sum_{v=-k}^{k} f_{uv} \cdot p_{i+u,j+v}$$

Normalized version scales with respect to both the filter and the corresponding input window

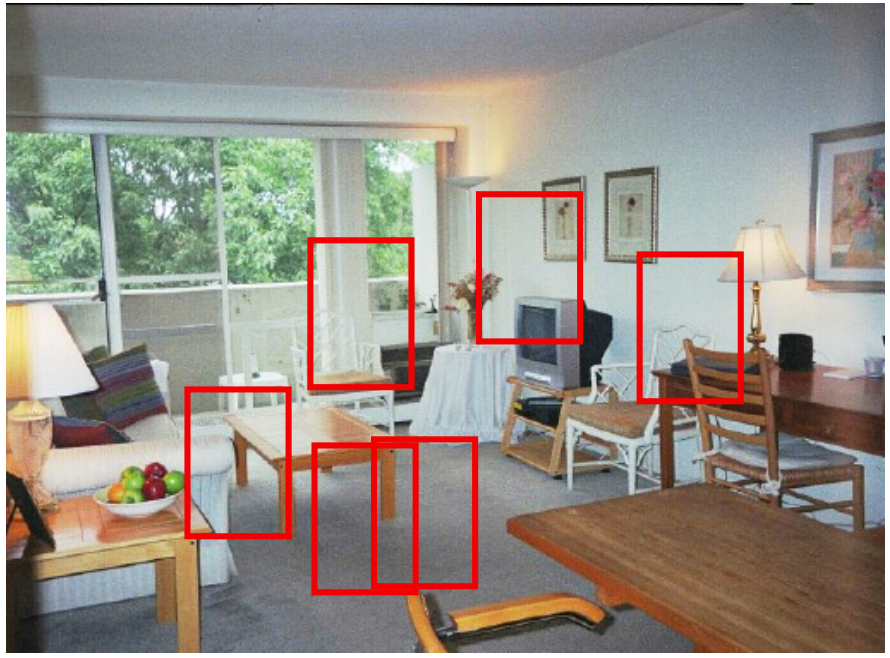
|•| indicates magnitude of kernel, i.e. square root of sum of squares of all elements in kernel.

Local max here!

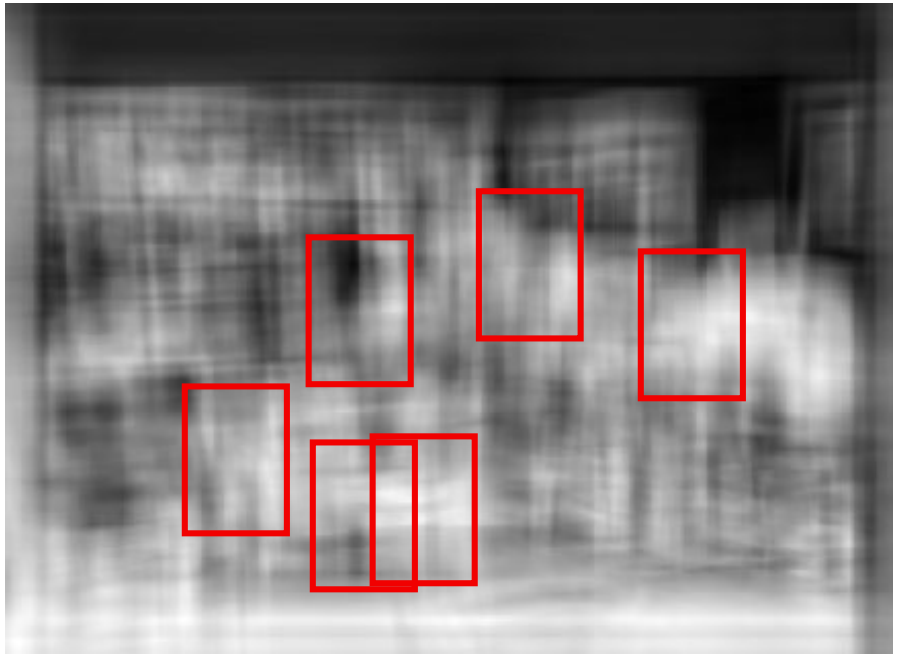
Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Template Matching for Detecting Objects?

Find the chair in this image



Simple template matching doesn't cut it...



A "popular method is that of template matching, by point to point correlation of a model pattern with the image pattern. These techniques are inadequate for three-dimensional scene analysis for many reasons, such as occlusion, changes in viewing angle, and articulation of parts." Nevatia & Binford, 1977.

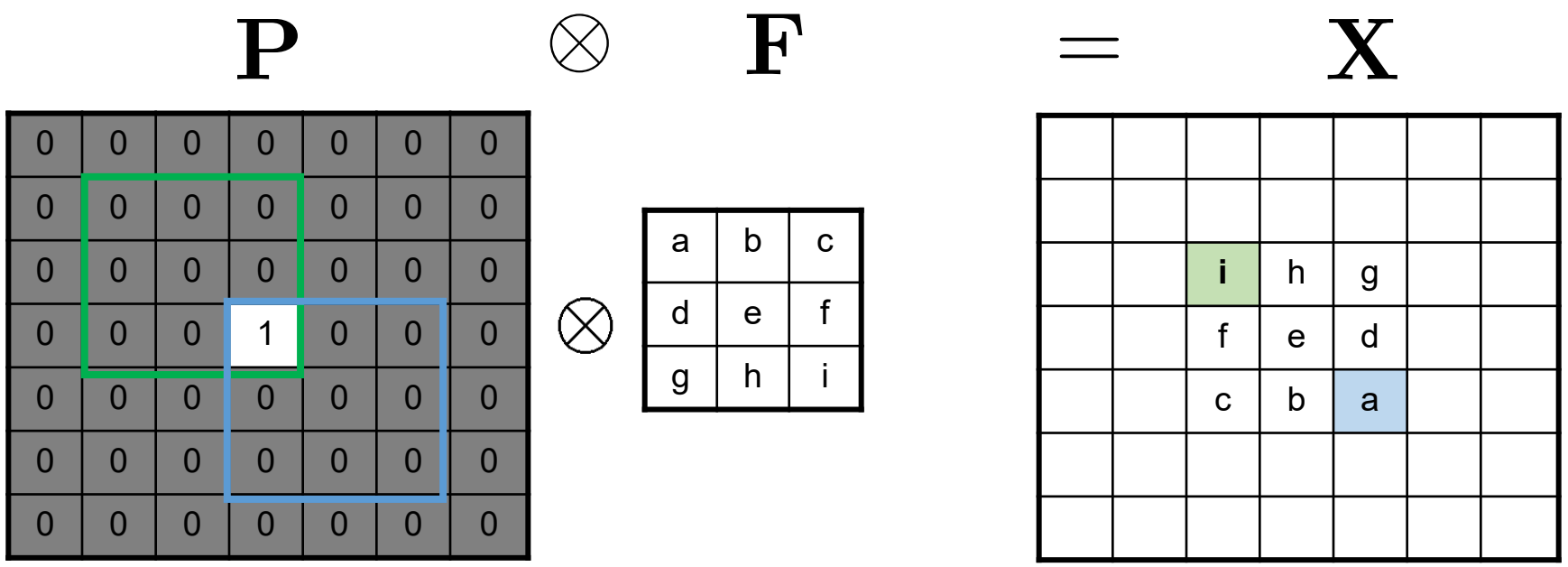
Convolution Filtering

Convolution operation;

Convolution vs. cross-correlation

Filtering an impulse signal

What is the result of filtering an impulse signal (image) with an arbitrary kernel?



Slide Credit: K. Grauman

Convolution

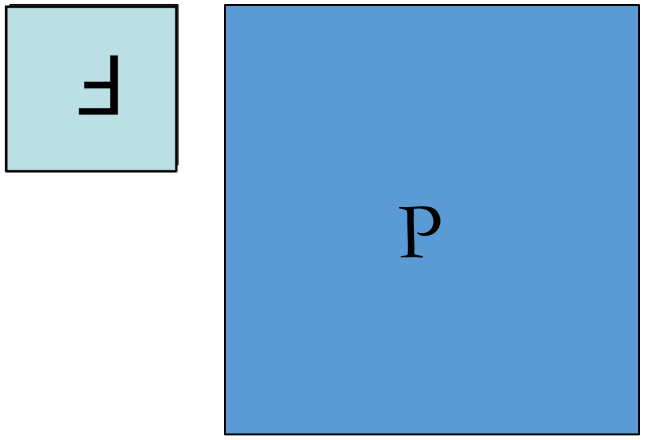
Flip the kernel in both dimensions, then apply cross-correlation

Subtraction of u,v indices!

$$x_{ij} = \sum_{u=-k}^{k} \sum_{v=-k}^{k} f_{uv} \cdot p_{i-u,j-v}$$

$$\mathbf{X} = \mathbf{F} * \mathbf{P}$$

convolution operator



Slide Credit: K. Grauman

Convolution vs. Cross-correlation

Convolution $x_{ij} = \sum_{u=-k}^{k} \sum_{v=-k}^{k} f_{uv} \cdot p_{i-u,j-v}$ $\mathbf{X} = \mathbf{P} * \mathbf{F}$

Cross-correlation $x_{ij} = \sum_{u=-k}^{k} \sum_{v=-k}^{k} f_{uv} \cdot p_{i+u,j+v}$ $\mathbf{X} = \mathbf{P} \otimes \mathbf{F}$

Properties of Convolution

Commutative:	$f * g = g * f$
Associative:	$(f * g) * h = f * (g * h)$
Distributes over addition:	$f * (g + h) = (f * g) + (f * h)$
Scalars factor out:	$\boldsymbol{\lambda} f * g = f * \boldsymbol{\lambda} g = \boldsymbol{\lambda}(f * g)$
Identity:	$f * e = f$, where $e = [\ldots, 0, 0, 1, 0, 0, \ldots]$.
Shift Invariant:	behaves the same everywhere, i.e. output value depends only on the pattern and not position of neighbourhood

Adapted from K. Grauman

Convolution vs. Correlation

- convolution is the "standard" we refer to when we talk about filtering unless otherwise explicitly stated
- for images, convolution and correlation are often used interchangeably since the two operations are equivalent if the filter kernels are symmetric
- symmetric: if "flipping" of kernel does not have any effect, e.g.
  - Gaussian
  - Box kernel

Non-Linear Filters

Median filtering, denoising impulse noise

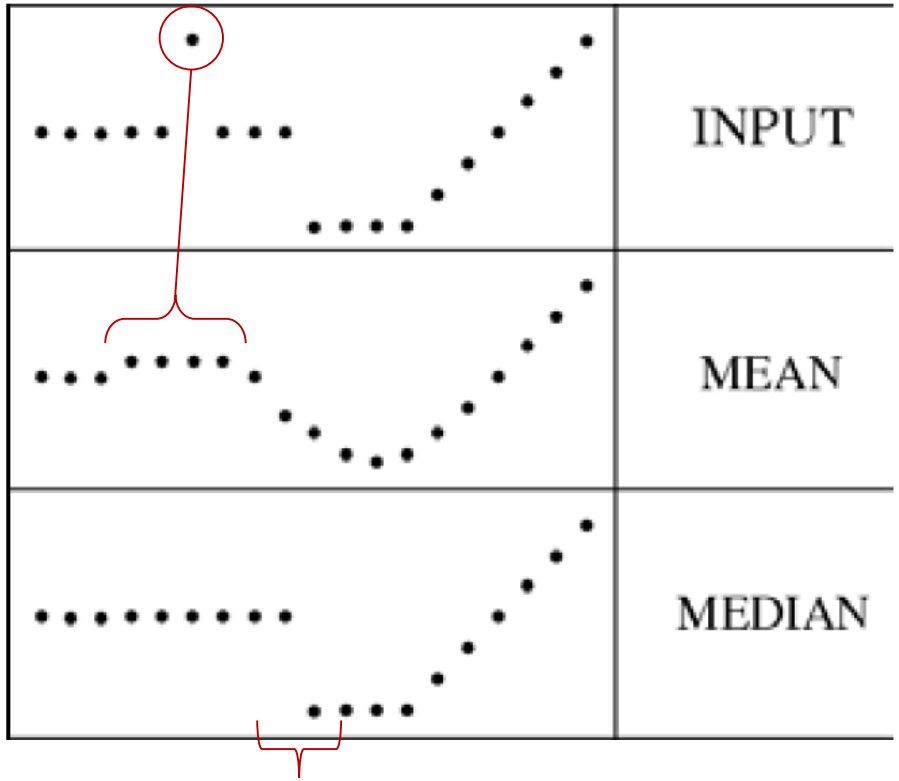
Non-Linear Filters

- Correlation and convolution are linear operations
- Non-linear filters perform non-linear operations, e.g.
  - Median
    - Median filtering is commonly used for removing impulse and salt & pepper noise
  - Min/Max
    - The max operation is used in max-pooling for deep neural networks.
  - Morphology
- We will take a closer look at the median filter

Denoising: Median Filter

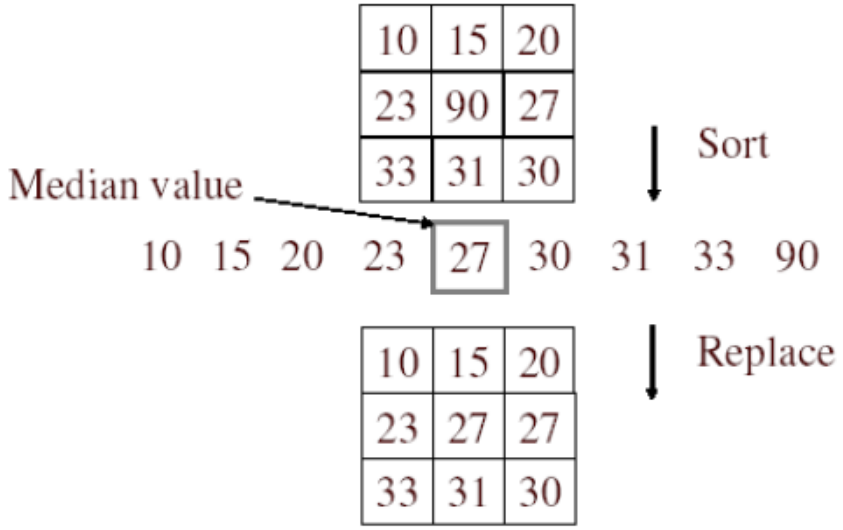
Salt and pepper noise

Impulse noise



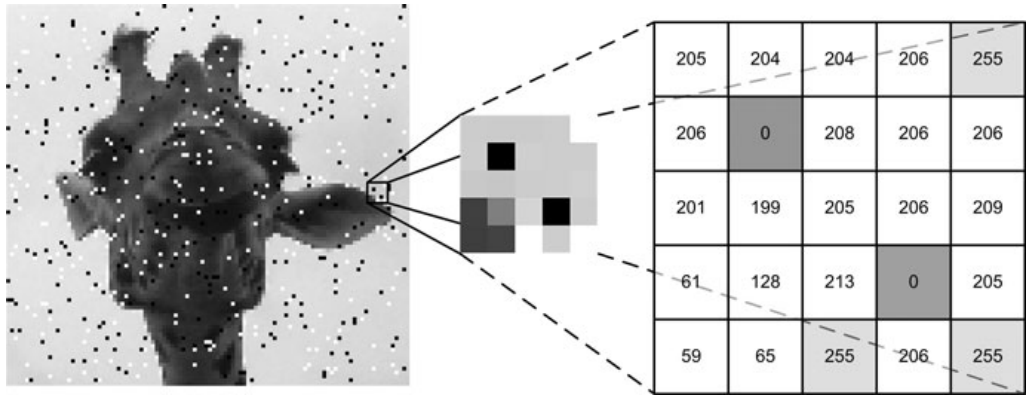
edge-preserving

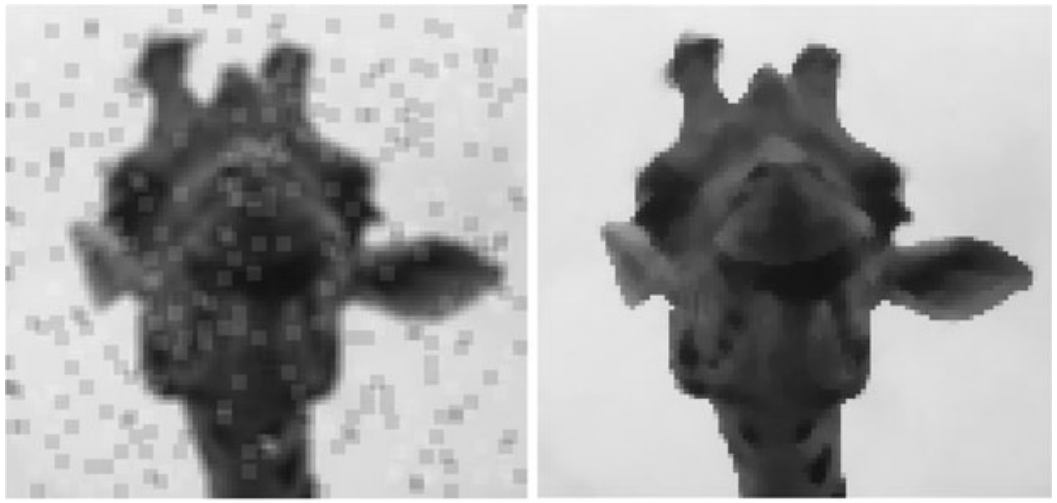
- No new pixel values introduced
- Removes spikes: good for impulse, salt & pepper noise



Slide Credit: K. Grauman

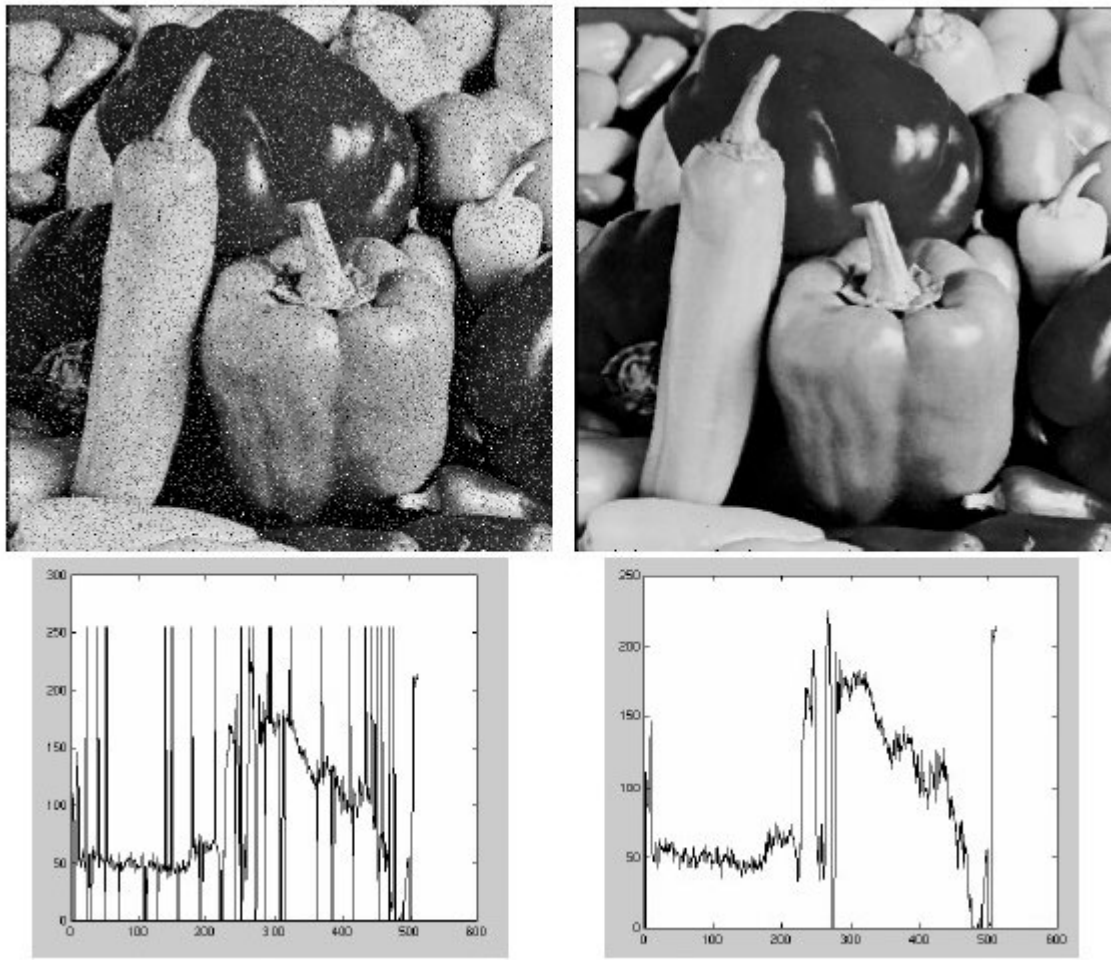
Median Filter Outputs





Mean filtered Median filtered

Salt and pepper noise Median filtered



Plots of a scanline.

Introduction to Video & Image Processing, Thomas B. Moeslund, Springer 2012

Summary

1. Point-wise Processing
   - Brightness, contrast, normalization, non-linear mappings
   - Histograms: stretching, equalization, segmentation
2. Linear Filtering
   - Achieves many effects: denoising, sharpening, moving, template matching
   - (Cross-)Correlation:
     - weighted summation of input window, weights stored in filter / kernel
     - normalized version for template matching
   - Convolution
     - Flips kernel before applying cross-correlation
3. Non-linear filters
   - Median filter for smoothing salt & pepper, impulse noise