CENTER FOR RESEARCH IN COMPUTER VISION

CAP5415
Computer Vision

Yogesh S Rawat

yogesh@ucf.edu

HEC-241

Image Filtering

Lecture 3

Outline

- Image as a function
- Extracting useful information from Images
  - Histogram
  - Edges
  - Smoothing/Removing noise
  - Convolution/Correlation
  - Image Derivatives/Gradient
  - Filtering (linear)

- *Read Szeliski, Chapter 3.*
- *Read Shah, Chapter 2.*
- *Read/Program CV with Python, Chapters 1 and 2.*

Image Filtering

Lecture 3

Digitization

Digitization

- Computers use discrete form of the images
- The process of transforming *continuous space* into *discrete space* is called digitization

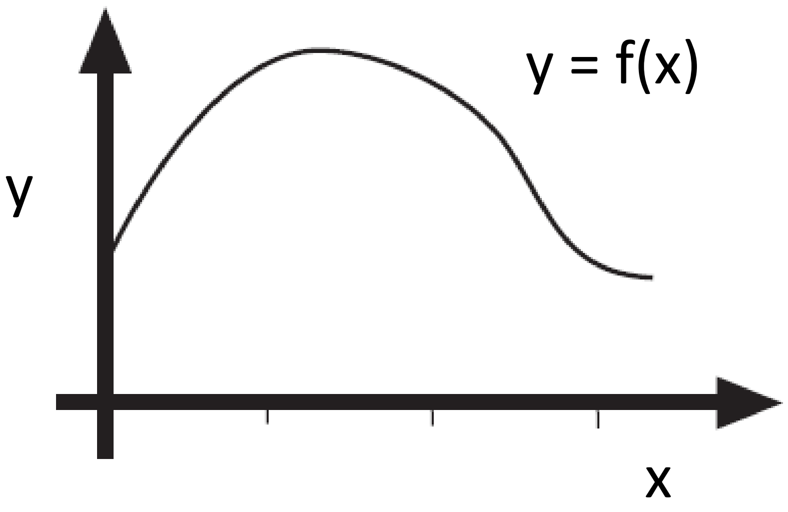
Image → Sampling+ Quantization → Digital Image

Digitization

- Function

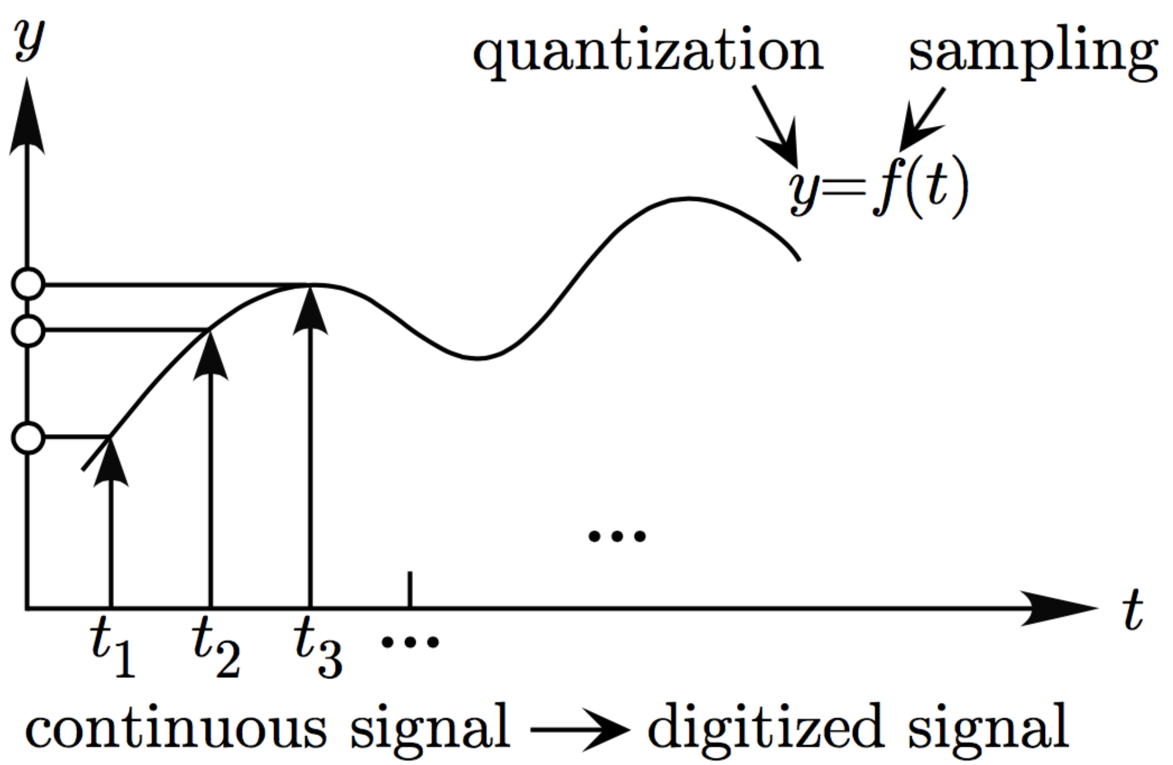
  $y = f(x)$

- Domain of a function
- Range of a function
- Sampling
  - Discretization of domain
- Quantization
  - Discretization of range

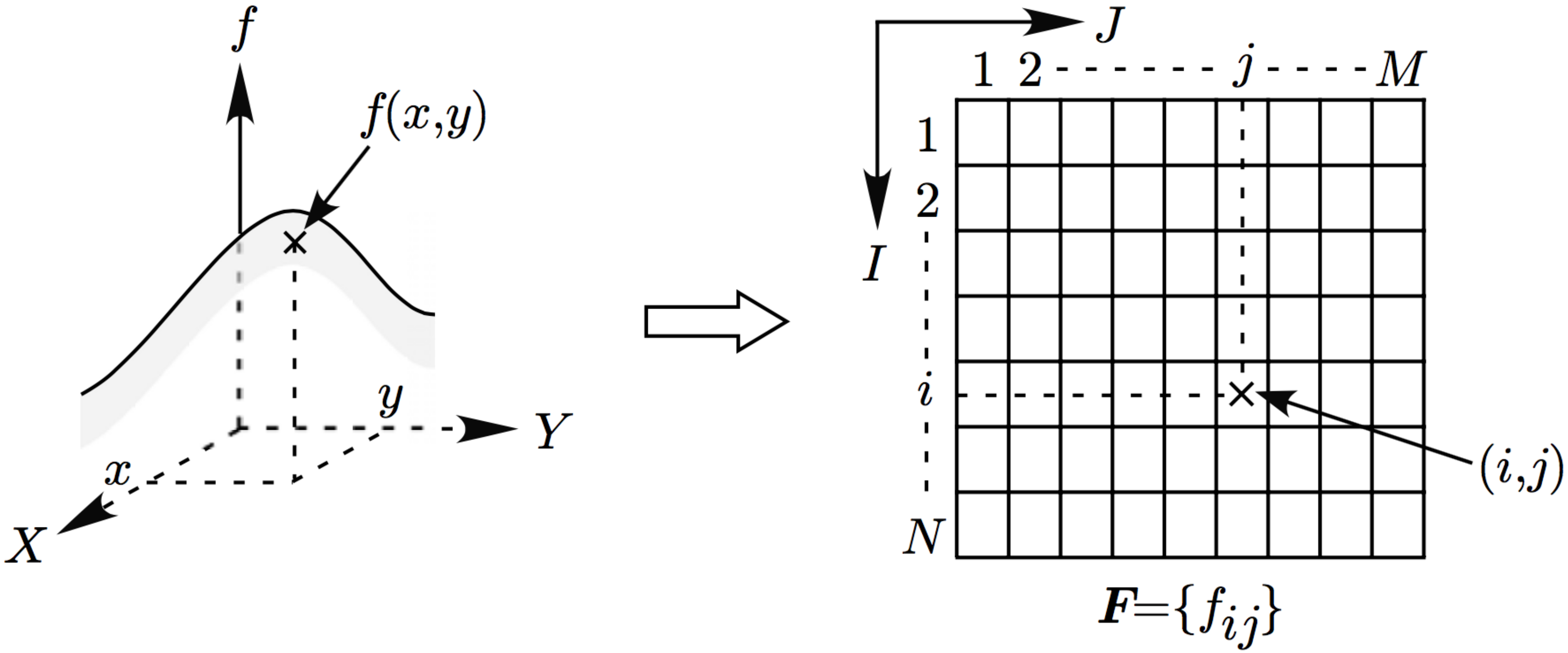


Digitization of 1D function

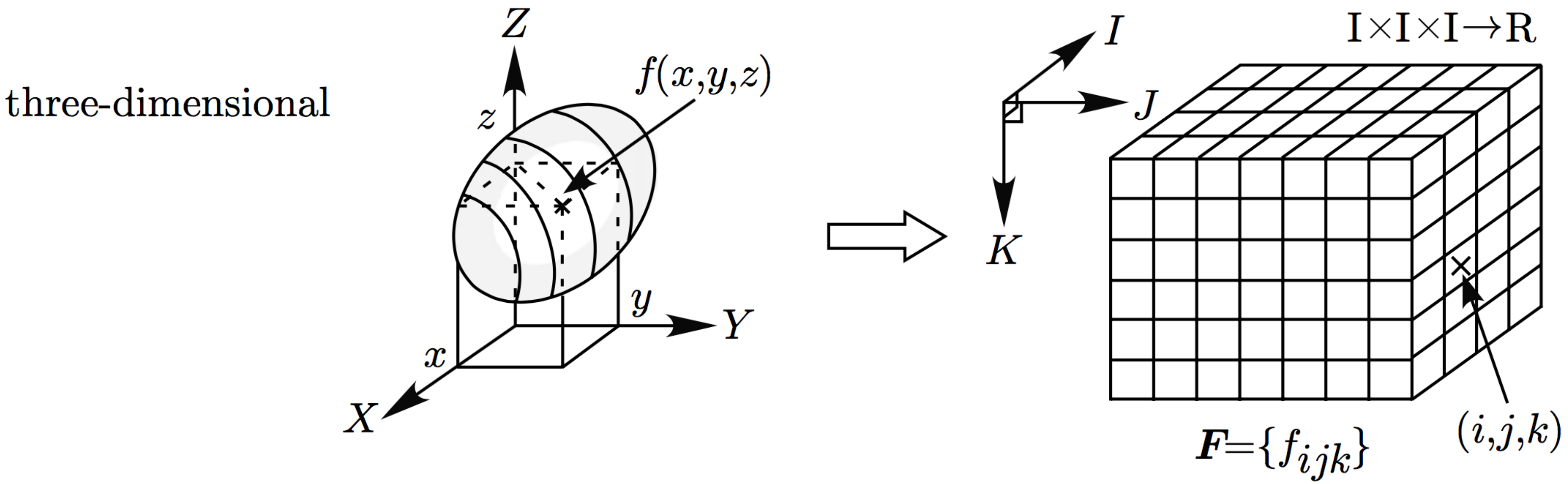
one-dimensional



Digitization of 2D function



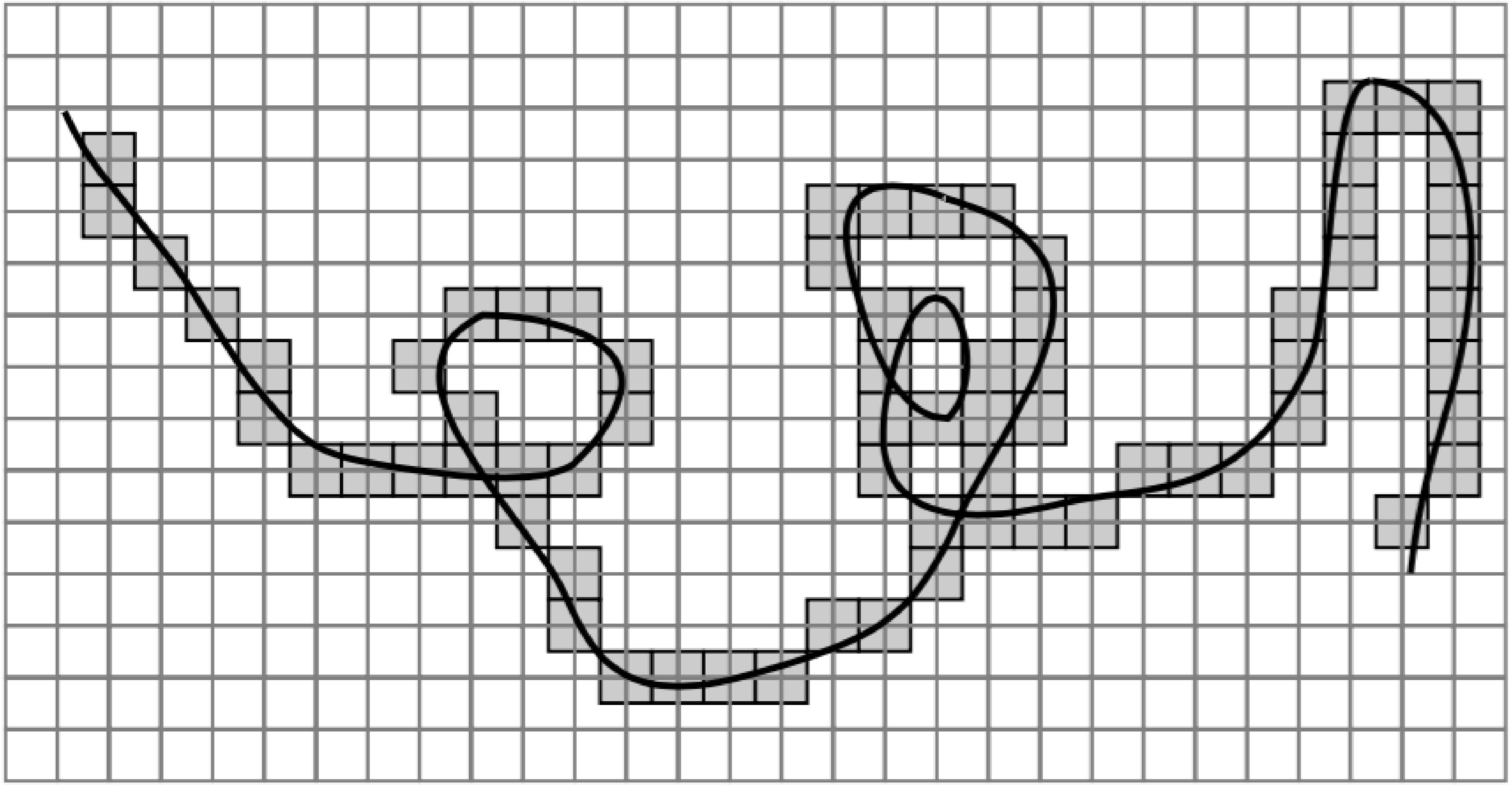
Digitization of 3D function



continuous image

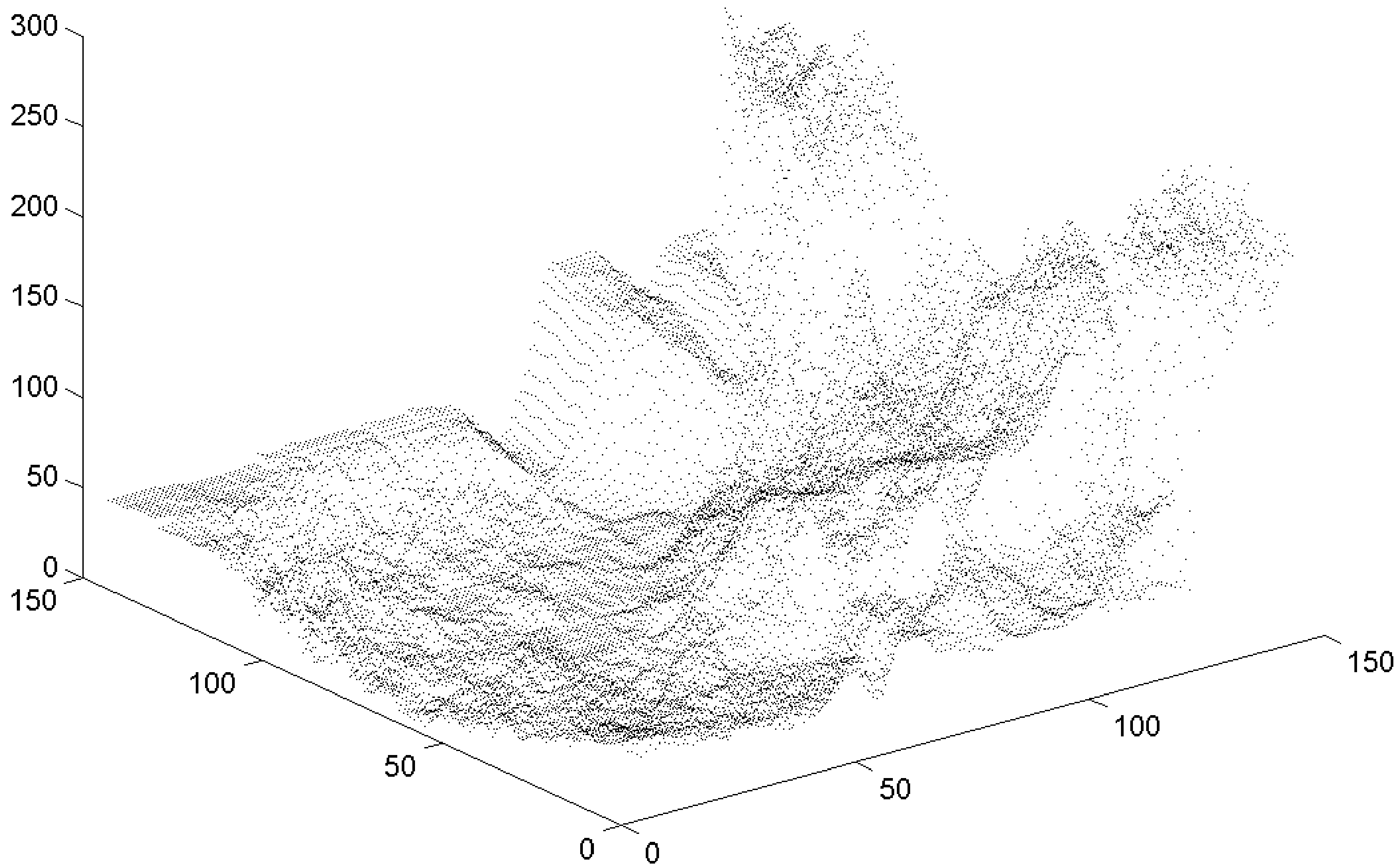
digitized image

Digitization of an arc



Gray scale digital image

Brightness
or intensity



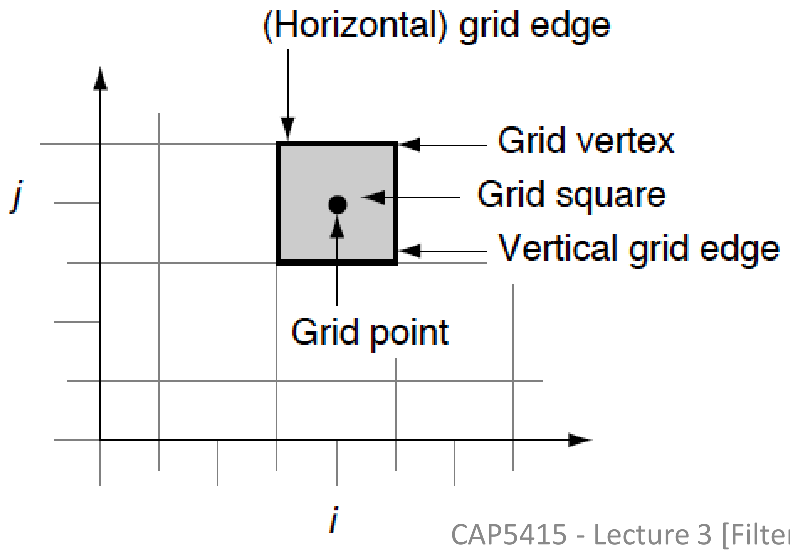
Danny Alexander

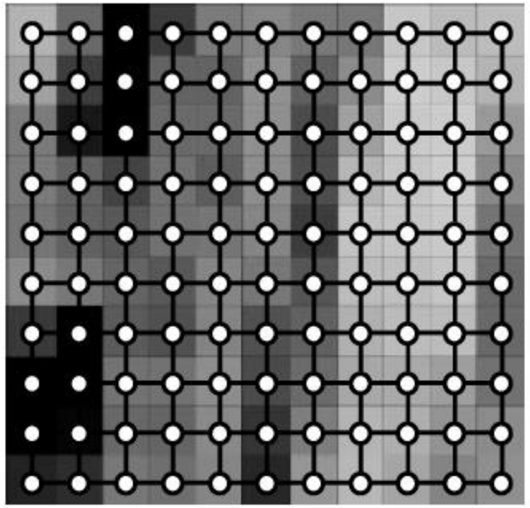
Definition

- An image *P* is a function defined on a (finite) rectangular subset *G* of a regular planar orthogonal array.
- *G* is called (2D) grid, and an element of *G* is called a pixel.
- *P* assigns a value of $P(p)$ to each $p \in G$

Definition

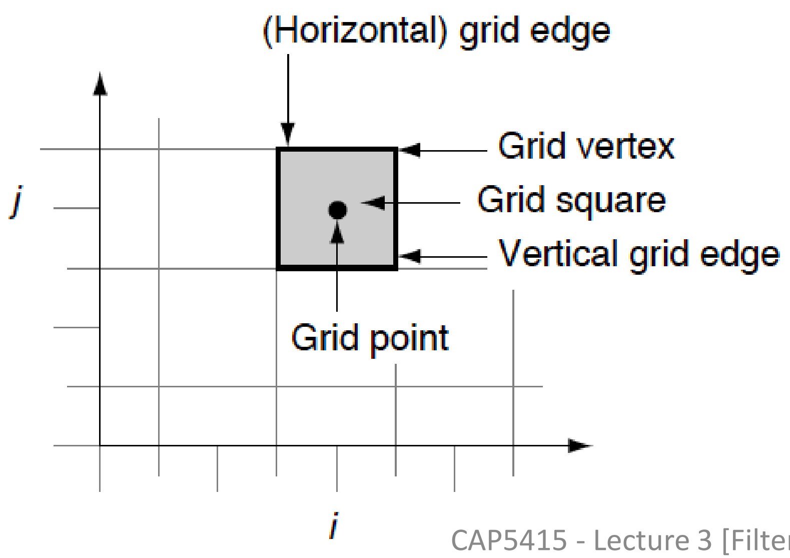
- An image *P* is a function defined on a (finite) rectangular subset *G* of a regular planar orthogonal array.
- *G* is called (2D) grid, and an element of *G* is called a pixel.
- *P* assigns a value of *P(p)* to each $p \in G$

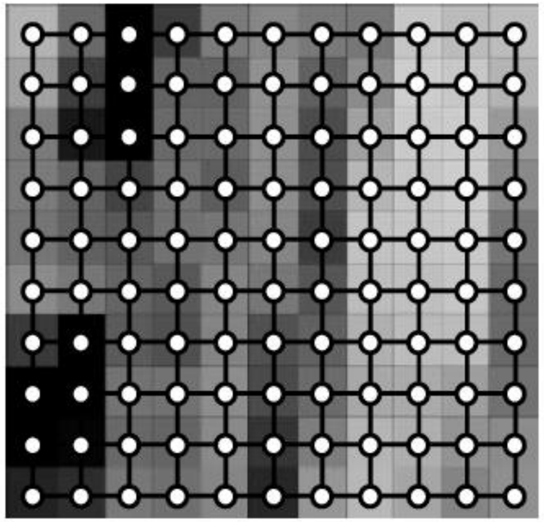




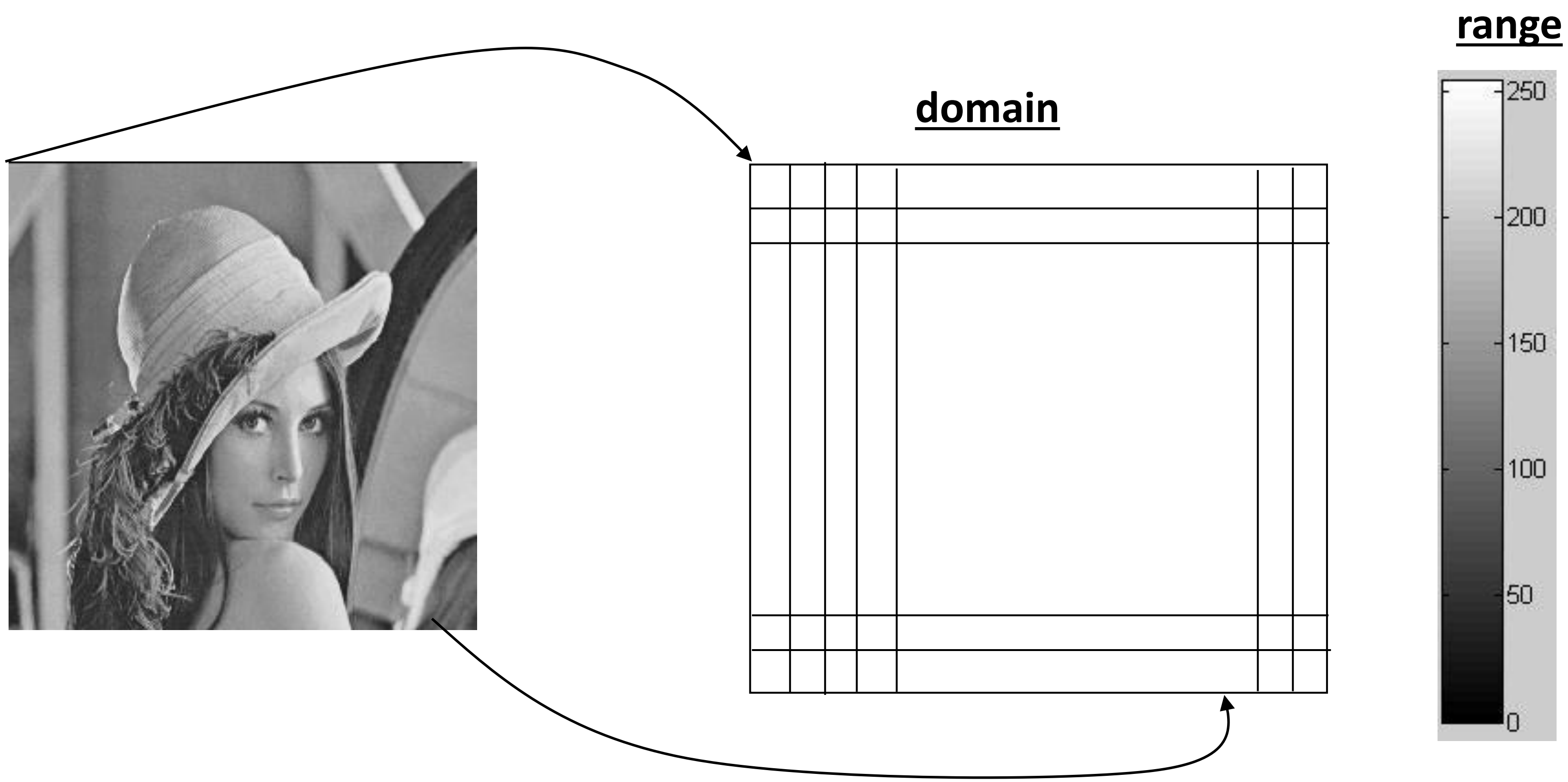
Definition

- Pictures are not only sampled
- They are also quantized
    - they may have only a finite number of possible values
    - i.e., 0 to 255, 0-1, …





Digitization



RGB Channels

Sampling

Quantization

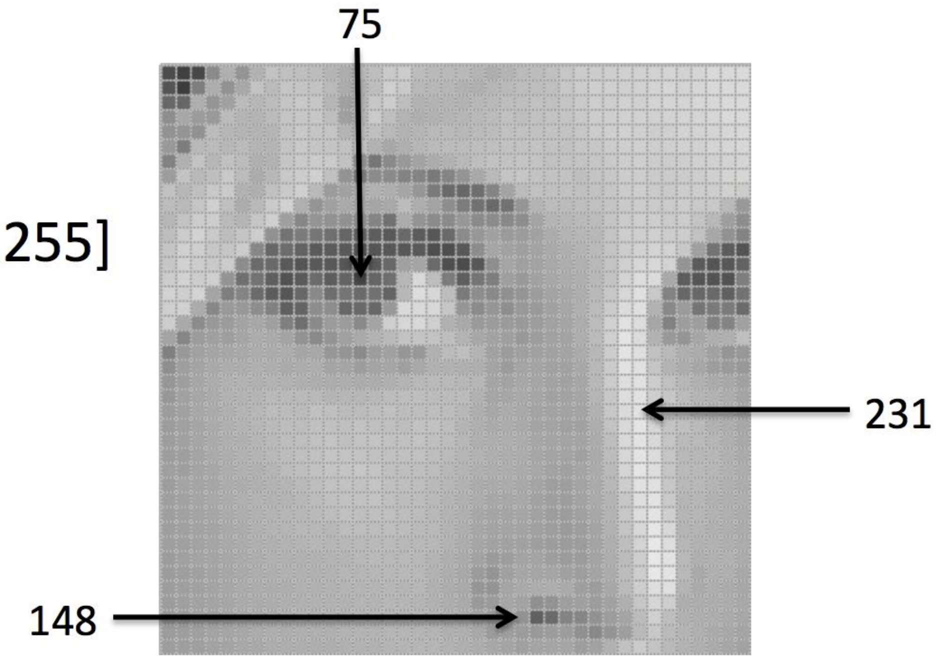
Original (256 colors) | 8 colors | 4 colors

Resolution

- Also, a display parameter
  - defined in dots per inch (DPI) or
  - measure of spatial pixel density
  - standard value for recent screen technologies is 72 dpi.
  - recent printer resolutions are in 300 dpi and/or 600 dpi.

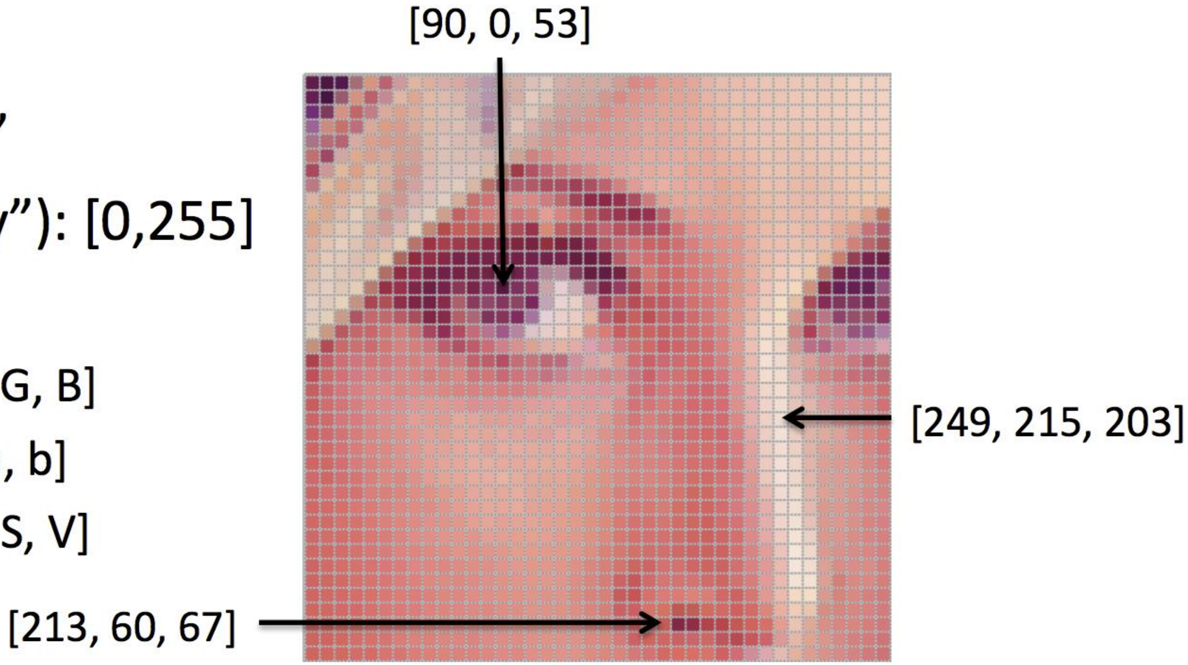
Gray scale image

- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale"
    (or "intensity"): [0,255]



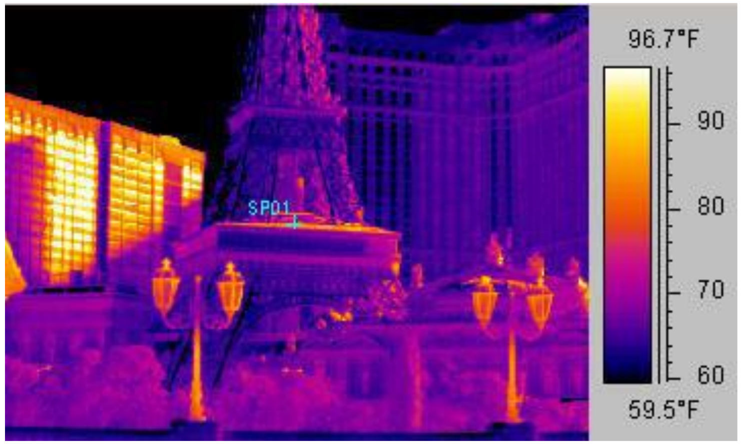
Color image

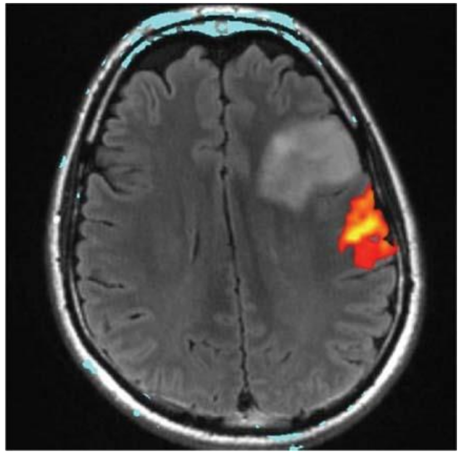
- An image contains discrete number of pixels
  - A simple example
  - Pixel value:
    - "grayscale"
    (or "intensity"): [0,255]
    - "color"
      - RGB: [R, G, B]
      - Lab: [L, a, b]
      - HSV: [H, S, V]



Source: F.F. Li

Image – other examples





Danny Alexander

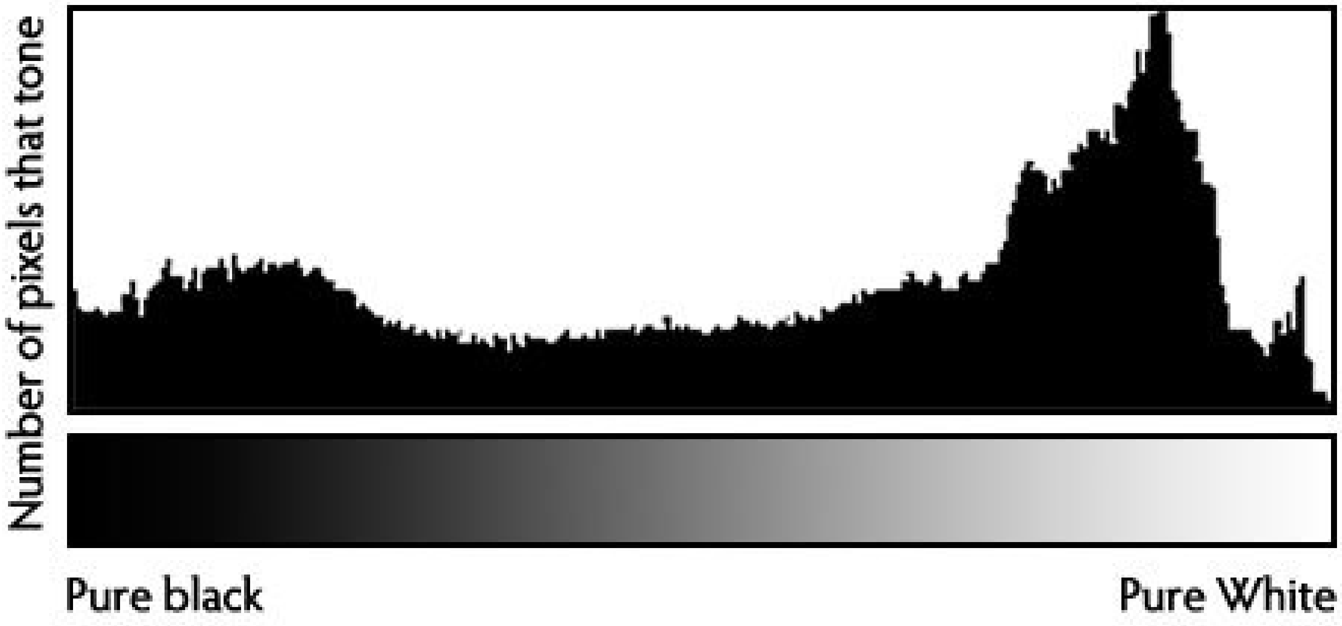
Questions?

Image Filtering

Lecture 3

Histogram

Image Histogram



UCF CENTER FOR RESEARCH IN COMPUTER VISION

Histogram Example

Use ImageJ and/or FIJI Credit: Klette 2012.

Questions?

CAP5415
Computer Vision

Yogesh S Rawat

yogesh@ucf.edu

HEC-241

Administrative

- Questions during lecture
  - Please wait for the question slide
  - Let us go over the topic once before asking questions
- Homework grading
  - Maximum two attempts
  - Best score will be used
  - If you have questions about homework, please visit office hours

Questions?

CAP5415
Computer Vision

Yogesh S Rawat

yogesh@ucf.edu

HEC-241

Image Filtering

Lecture 3

Noise

Intensity profiles for selected (two) rows

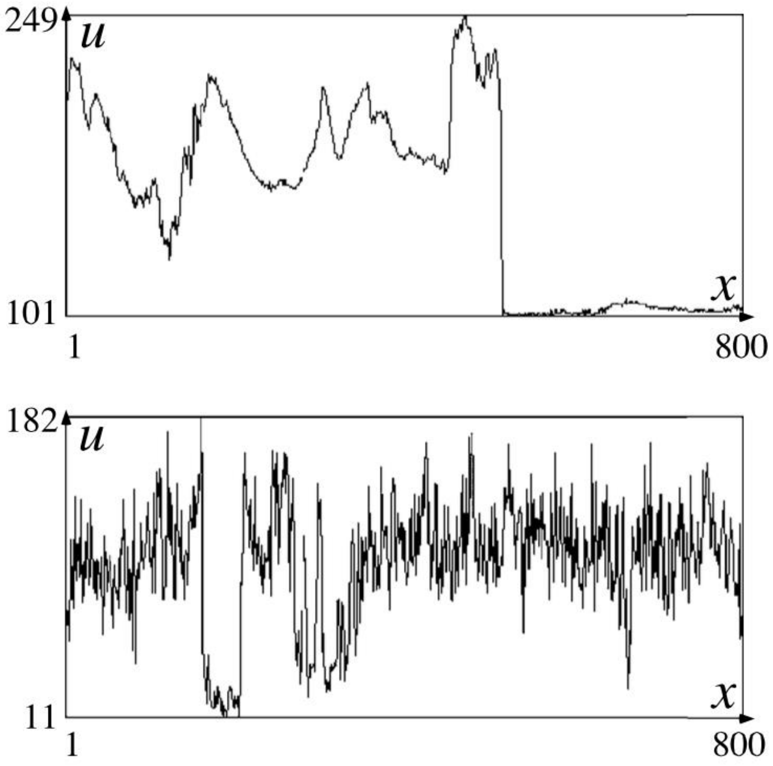


Image noise

- Light Variations
- Camera Electronics
- Surface Reflectance
- Lens

- Noise is random,
  - it occurs with some probability
- It has a distribution

Noise

- $I_{original}(x,y)$ – true pixel value at (x,y)
- n(x,y) - noise at (x,y)
- $I_{observed}(x,y) = I_{original}(x,y) + n(x,y)$ additive noise

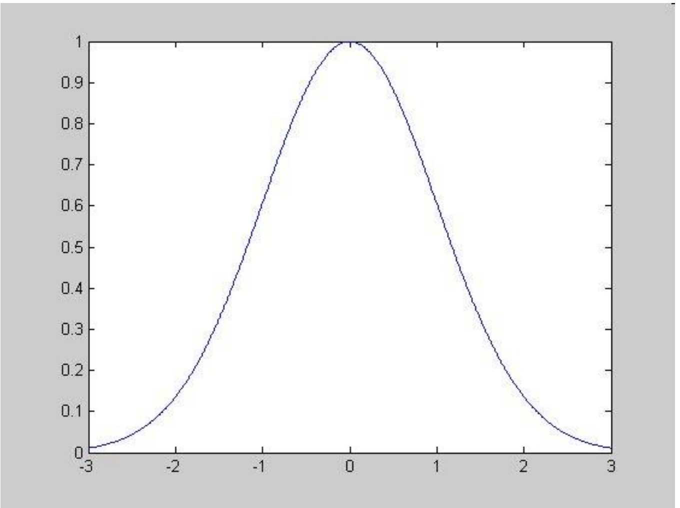
Noise

- $I_{original}(x,y)$ – true pixel value at (x,y)
- n(x,y) - noise at (x,y)
- $I_{observed}(x,y) = I_{original}(x,y) * n(x,y)$ multiplicative noise

Gaussian Noise

$$n(x,y) \approx g(n) = e^{\frac{-n^2}{2\sigma^2}}$$

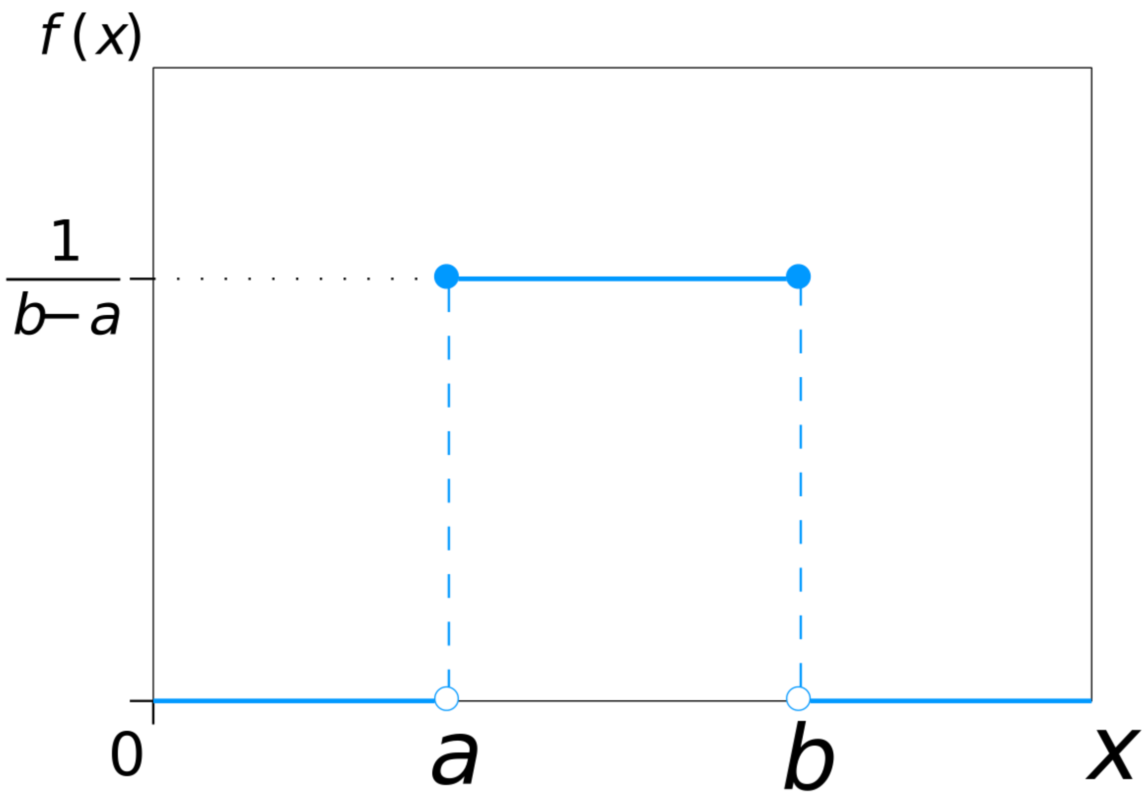
$g(n)$



n

Probability Distribution
n is a random variable

Uniform distribution



Salt and pepper noise

- Each pixel is randomly made black or white with a uniform probability distribution

➔

Salt-pepper

Questions?

UCF CENTER FOR RESEARCH IN COMPUTER VISION

Image Filtering

Lecture 3

Filtering

Image filtering

- Image filtering: compute function of local neighborhood at each position

h=output f=filter I=image

$$h[m,n]=\sum_{k,l} f[k,l]\, I[m+k,n+l]$$

2d coords=k,l 2d coords=m,n

$$[\quad] \qquad [\,] \qquad [\quad]$$

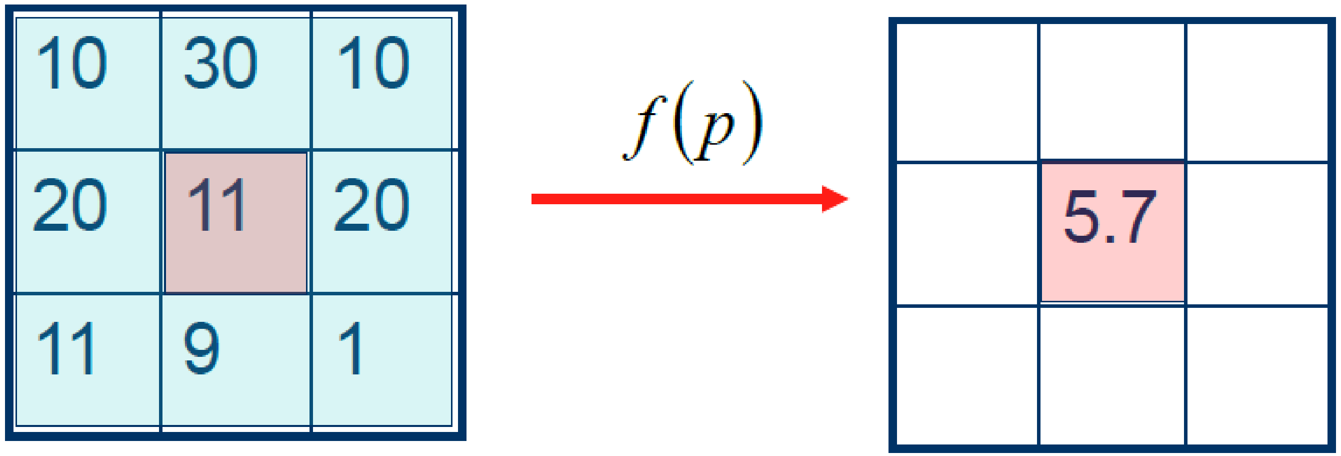
Image filtering

- Image filtering: compute function of local neighborhood at each position

- Enhance images
  - Denoise, resize, increase contrast, etc.
- Extract information from images
  - Texture, edges, distinctive points, etc.
- Detect patterns
  - Template matching

Filtering

- Modify pixels based on some function of neighborhood



UCF CENTER FOR RESEARCH IN COMPUTER VISION

Filtering

- Output is linear combination of the neighborhood pixels

$$\underset{\text{Image}}{\begin{bmatrix} 1 & 3 & 0 \\ 2 & 10 & 2 \\ 4 & 1 & 1 \end{bmatrix}} \otimes \underset{\text{Kernel}}{\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0.1 & -1 \\ 1 & 0 & -1 \end{bmatrix}} = \underset{\text{Filter Output}}{\begin{bmatrix} & & \\ & 5 & \\ & & \end{bmatrix}}$$

Correlation (linear relationship)

$$f \otimes h = \sum_k \sum_l f(k,l)h(k,l)$$

$$f = \text{Image}$$

$$h = \text{Kernel}$$

$$\overset{f}{\begin{array}{|c|c|c|}\hline f_1 & f_2 & f_3 \\ \hline f_4 & f_5 & f_6 \\ \hline f_7 & f_8 & f_9 \\ \hline\end{array}} \otimes \overset{h}{\begin{array}{|c|c|c|}\hline h_1 & h_2 & h_3 \\ \hline h_4 & h_5 & h_6 \\ \hline h_7 & h_8 & h_9 \\ \hline\end{array}} \longrightarrow$$

$$\begin{aligned} f \otimes h = {} & f_1h_1 + f_2h_2 + f_3h_3 \\ & + f_4h_4 + f_5h_5 + f_6h_6 \\ & + f_7h_7 + f_8h_8 + f_9h_9 \end{aligned}$$

Convolution

$$f * h = \sum_k \sum_l f(k,l)\,h(-k,-l)$$

f = Image

h = Kernel

h

h_1	h_2	h_3
h_4	h_5	h_6
h_7	h_8	h_9

$X - flip$

h_7	h_8	h_9
h_4	h_5	h_6
h_1	h_2	h_3

$Y - flip$

f

f_1	f_2	f_3
f_4	f_5	f_6
f_7	f_8	f_9

$*$

h_9	h_8	h_7
h_6	h_5	h_4
h_3	h_2	h_1

$$\begin{aligned} f * h = {} & f_1 h_9 + f_2 h_8 + f_3 h_7 \\ & + f_4 h_6 + f_5 h_5 + f_6 h_4 \\ & + f_7 h_3 + f_8 h_2 + f_9 h_1 \end{aligned}$$

Convolution

- Convolution is associative

$$F * (G * I) = (F * G) * I$$

Correlation and Convolution

- Convolution is a filtering operation
  - expresses the amount of overlap of one function as it is shifted over another function

- Correlation compares the similarity of two sets of data
  - relatedness of the signals!

Averages

- Mean

$$I = \frac{I_1 + I_2 + \ldots I_n}{n} = \frac{\sum_{i=1}^{n} I_i}{n}$$

- Weighted mean

$$I = \frac{w_1 I_1 + w_2 I_2 + \ldots + w_n I_n}{n} = \frac{\sum_{i=1}^{n} w_i I_i}{n}$$

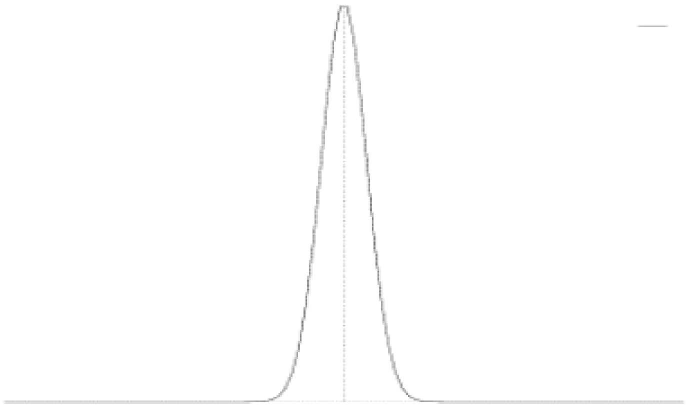
Questions?

Image Filtering

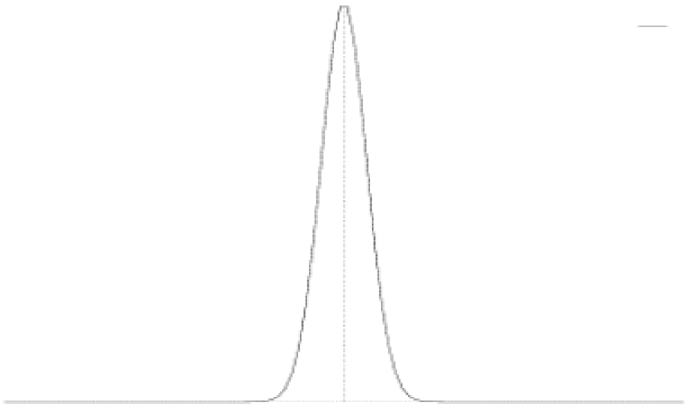
Lecture 3

Filtering Examples

Gaussian filter

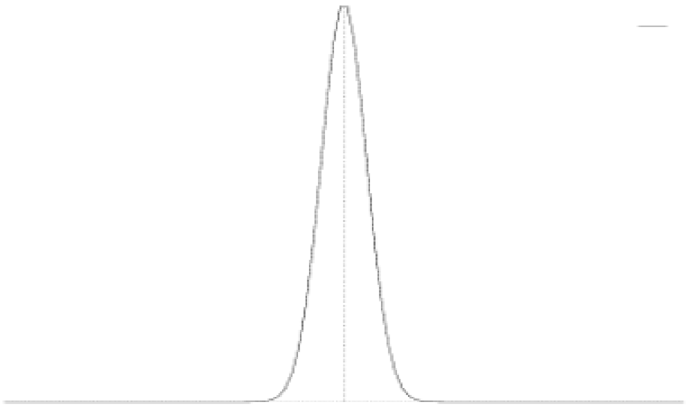


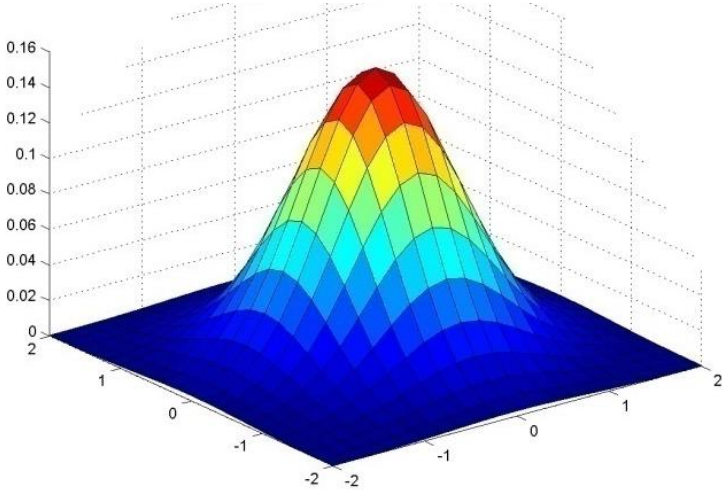
Gaussian filter



$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$

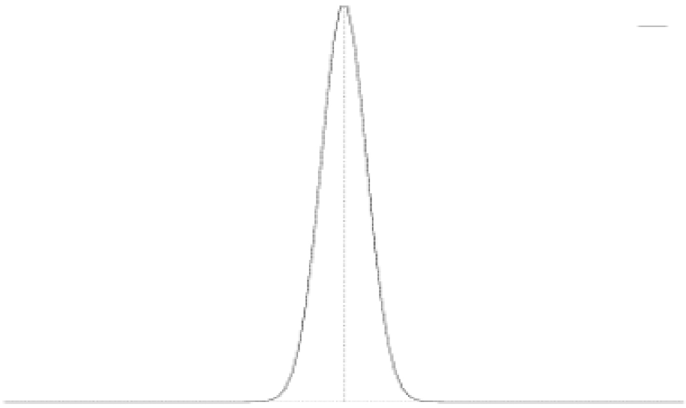
Gaussian filter

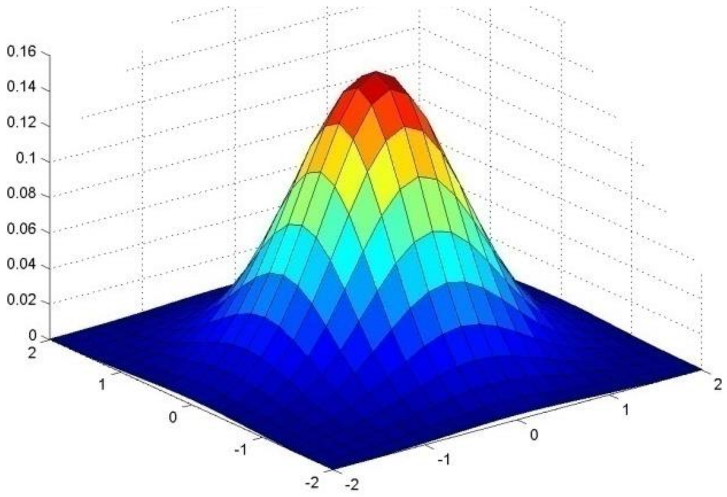




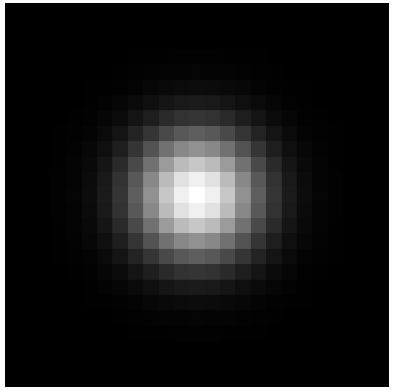
$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$

Gaussian filter

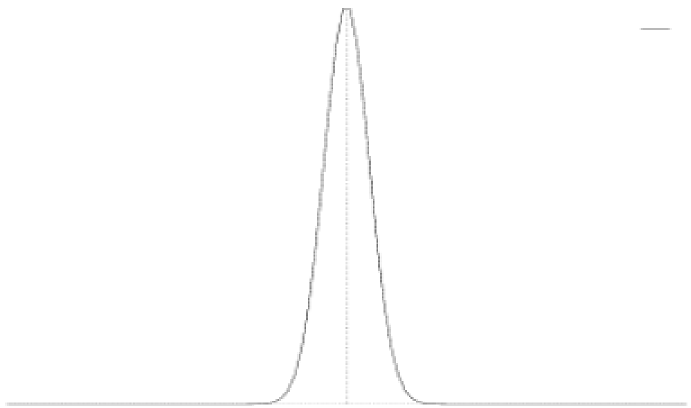


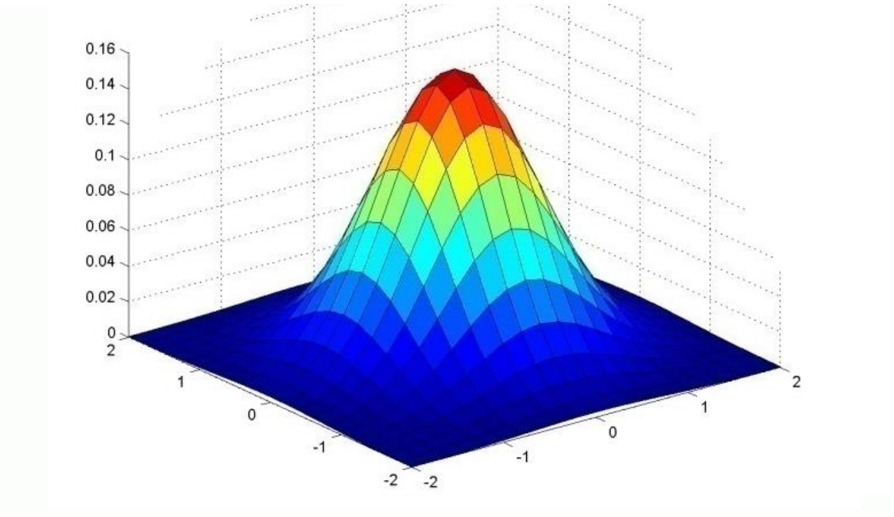


$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$



Gaussian filter

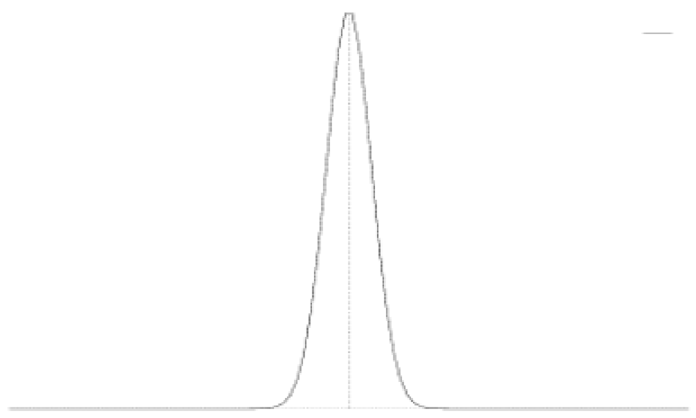


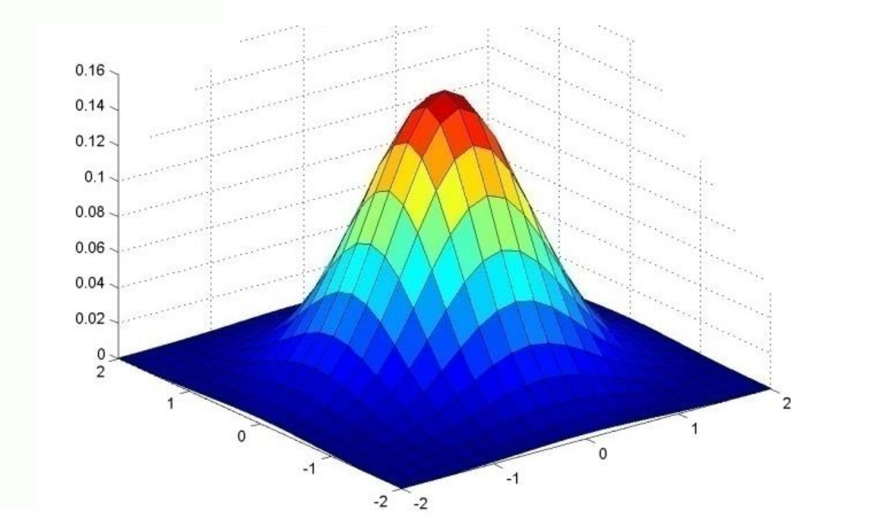


$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$

$$g(x,y) = e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

Gaussian filter





$$g(x) = e^{\frac{-x^2}{2\sigma^2}}$$

$$g(x,y) = e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

$$g(x) = \begin{bmatrix} .011 & .13 & .6 & 1 & .6 & .13 & .011 \end{bmatrix}$$

Gaussian filter - properties

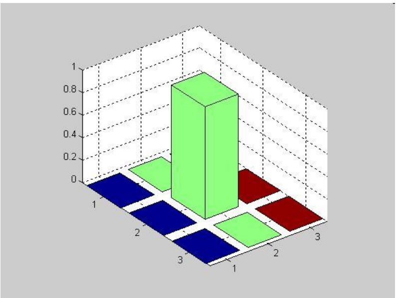
- Most common natural model

https://studiousguy.com/real-life-examples-normal-distribution/

Gaussian filter - properties

- Most common natural model
- Smooth function, it has infinite number of derivatives
- It is Symmetric
- Fourier Transform of Gaussian is Gaussian.
- Convolution of a Gaussian with itself is a Gaussian.
- Gaussian is separable; 2D convolution can be performed by two 1-D convolutions
- There are cells in eye that perform Gaussian filtering.

Filtering Examples - 1



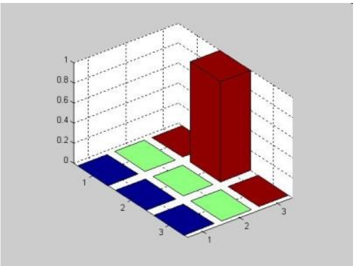
*

0	0	0
0	1	0
0	0	0

=

Filtering Examples - 2

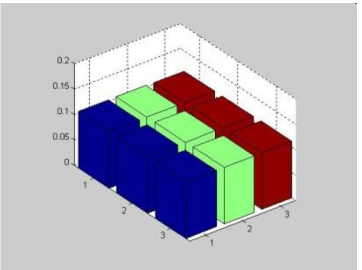
*



0	0	0
1	0	0
0	0	0

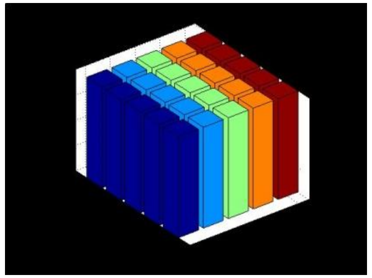
=

Filtering Examples - 3



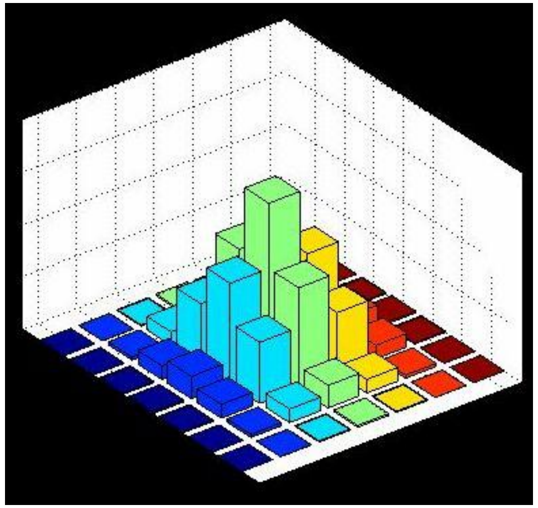
$$* \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$

Filtering Examples - 4



$$* \frac{1}{25} \begin{array}{|c|c|c|c|c|} \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline 1 & 1 & 1 & 1 & 1 \\ \hline \end{array} =$$

Filtering Examples - 5

 $*$  $=$ 

Gaussian Smoothing

Filtering Examples - 6

Gaussian Smoothing

Smoothing by Averaging

Filtering Examples - 7

After additive
Gaussian Noise

After Averaging

After Gaussian Smoothing

Example: box filter

What does it do?

- Replaces each pixel with an average of its neighborhood

$$g[\cdot\,,\cdot\,]$$

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

Slide credit: David Lowe (UBC)

Image filtering

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

$$h[m,n] = \sum_{k,l} g[k,l]\; f[m+k, n+l]$$

Credit: S. Seitz

Image filtering

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10							

$$h[m,n] = \sum_{k,l} g[k,l]\, f[m+k, n+l]$$

Credit: S. Seitz

Image filtering

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20						

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$$g[\cdot,\cdot] \quad \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20	30					

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$$g[\cdot,\cdot] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
				?					

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$g[\cdot\,,\cdot\,]$ $\frac{1}{9}$

1	1	1
1	1	1
1	1	1

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
						?			
				50					

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Image filtering

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Example: box filter

What does it do?

- Replaces each pixel with an average of its neighborhood
- Achieve smoothing effect (remove sharp features)

$$g[\cdot\,,\cdot\,]$$

$$\frac{1}{9}\begin{array}{|c|c|c|}\hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline\end{array}$$

Slide credit: David Lowe (UBC)

Smoothing with Box filter

Practice with kernels

Original

0	0	0
0	1	0
0	0	0

?

Practice with kernels

Original

0	0	0
0	1	0
0	0	0

Filtered
(no change)

Practice with kernels

Original

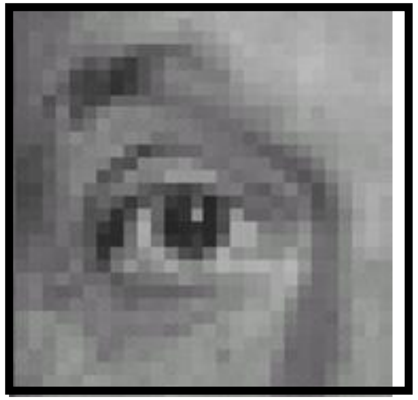
0	0	0
0	0	1
0	0	0

?

Practice with kernels

Original

0	0	0
0	0	1
0	0	0



Shifted left
By 1 pixel

Practice with kernels

Original

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

(Note that filter sums to 1)

Practice with kernels

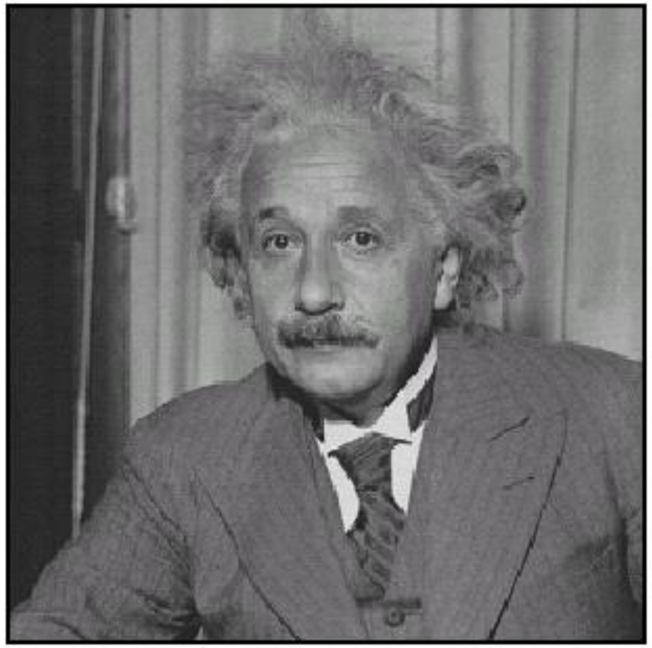
Original

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

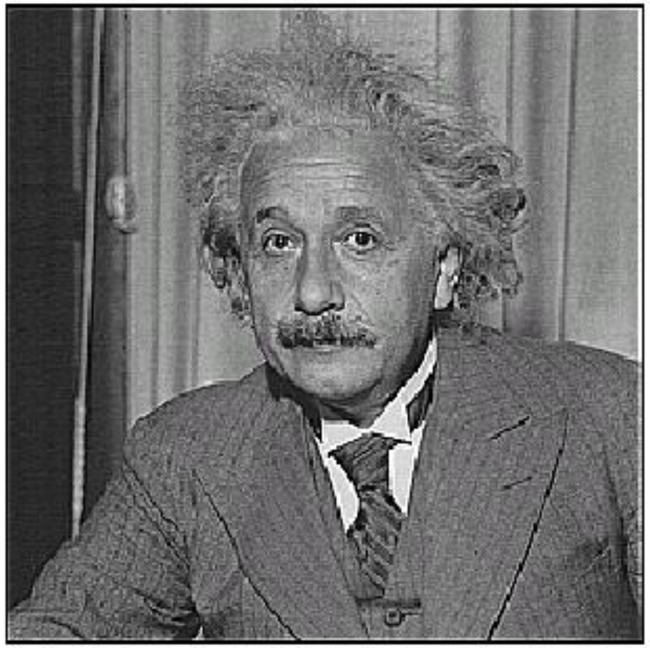
Sharpening filter

- Accentuates differences with local average

Sharpening Filter

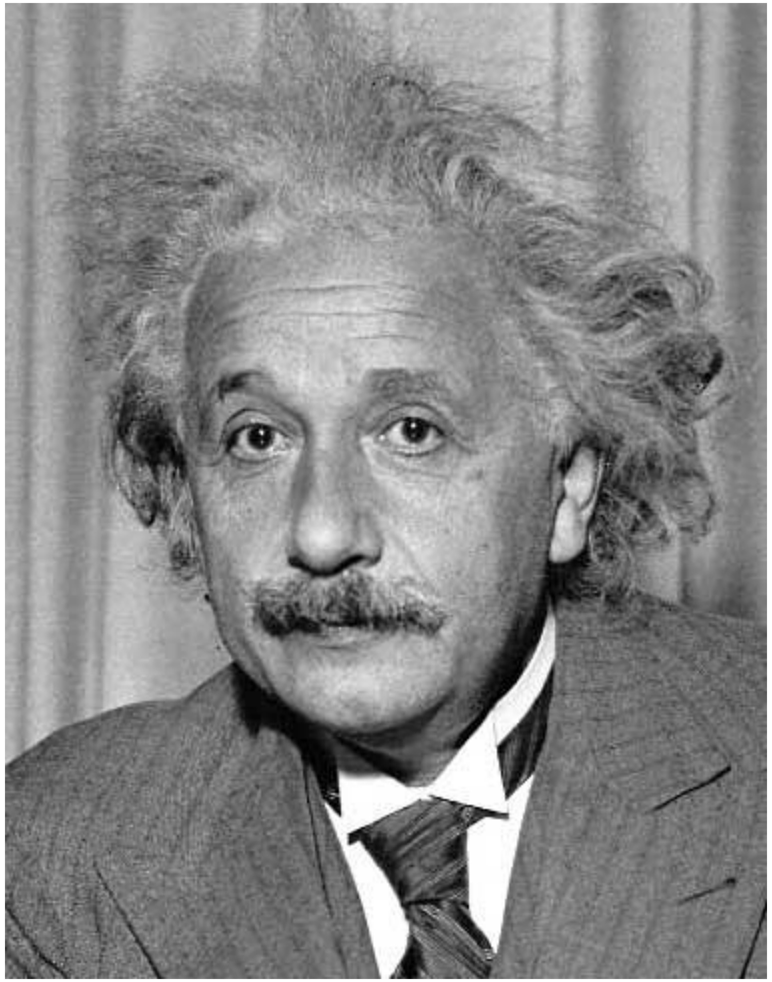


before



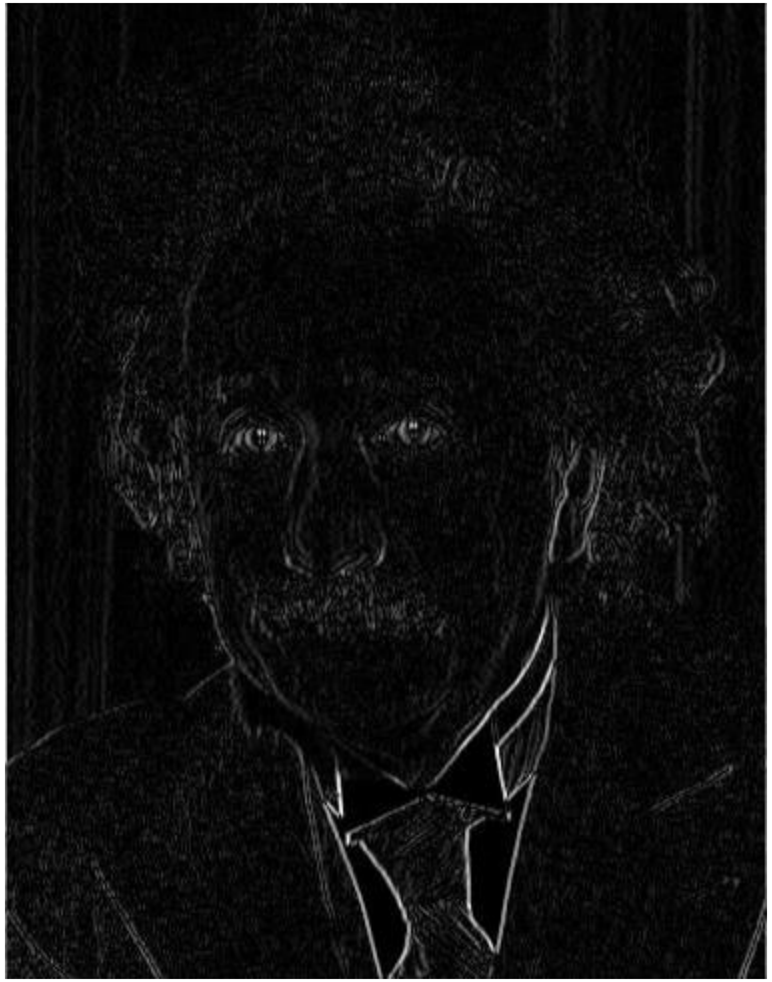
after

Sobel Filtering



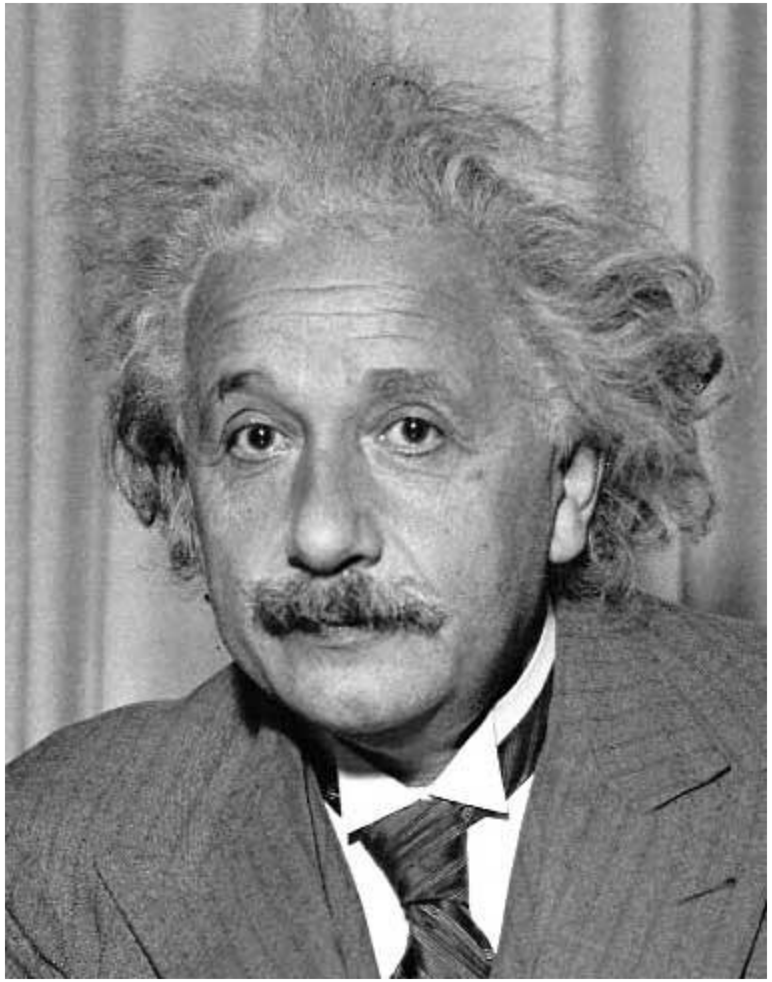
1	0	-1
2	0	-2
1	0	-1

Sobel



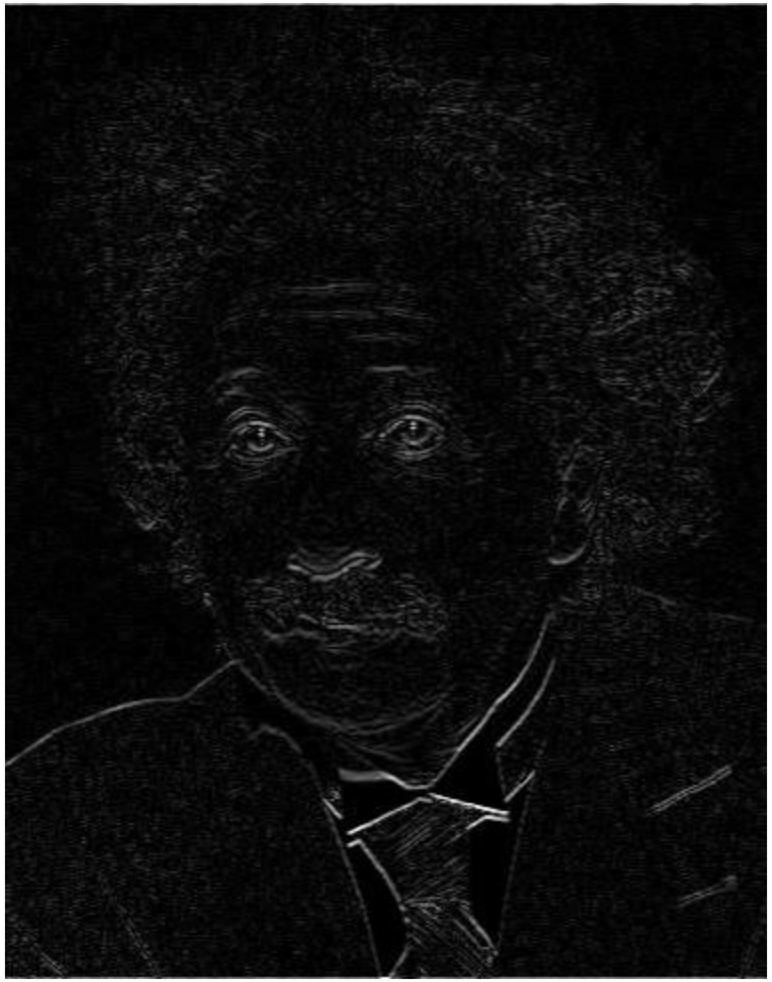
Vertical Edge
(absolute value)

Sobel Filtering



1	2	1
0	0	0
-1	-2	-1

Sobel



Horizontal Edge
(absolute value)

Key properties of linear filters

Linearity:

filter(f_1 + f_2) = filter(f_1) + filter(f_2)

Shift invariance: same behavior regardless of pixel location

filter(shift(f)) = shift(filter(f))

Any linear, shift-invariant operator can be represented as a convolution

More properties

- Commutative: $a * b = b * a$
  - Conceptually no difference between filter and signal
  - particular filtering implementations might break this equality
- Associative: $a * (b * c) = (a * b) * c$
  - Often apply several filters one after another: $(((a * b_1) * b_2) * b_3)$
  - This is equivalent to applying one filter: a * $(b_1 * b_2 * b_3)$
- Distributes over addition: $a * (b + c) = (a * b) + (a * c)$
- Scalars factor out: $ka * b = a * kb = k\ (a * b)$
- Identity: unit impulse e = [0, 0, 1, 0, 0], $a * e = a$

Median Filter

- A **Median Filter** operates over a window by selecting the median intensity in the window.
- Advantage?
- Is it same as convolution?

Image filtering - mean

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
				?					

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Image filtering - mean

$$g[\cdot\,,\cdot\,] \quad \frac{1}{9}\begin{bmatrix}1&1&1\\1&1&1\\1&1&1\end{bmatrix}$$

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
				50					

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

Image filtering - median

$f[.,.]$

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

$h[.,.]$

	0	10	20	30	30				
				?					

$$h[m,n]=\sum_{k,l} g[k,l]\, f[m+k,n+l]$$

Credit: S. Seitz

Image filtering - median

$f[.,.]$

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

$h[.,.]$

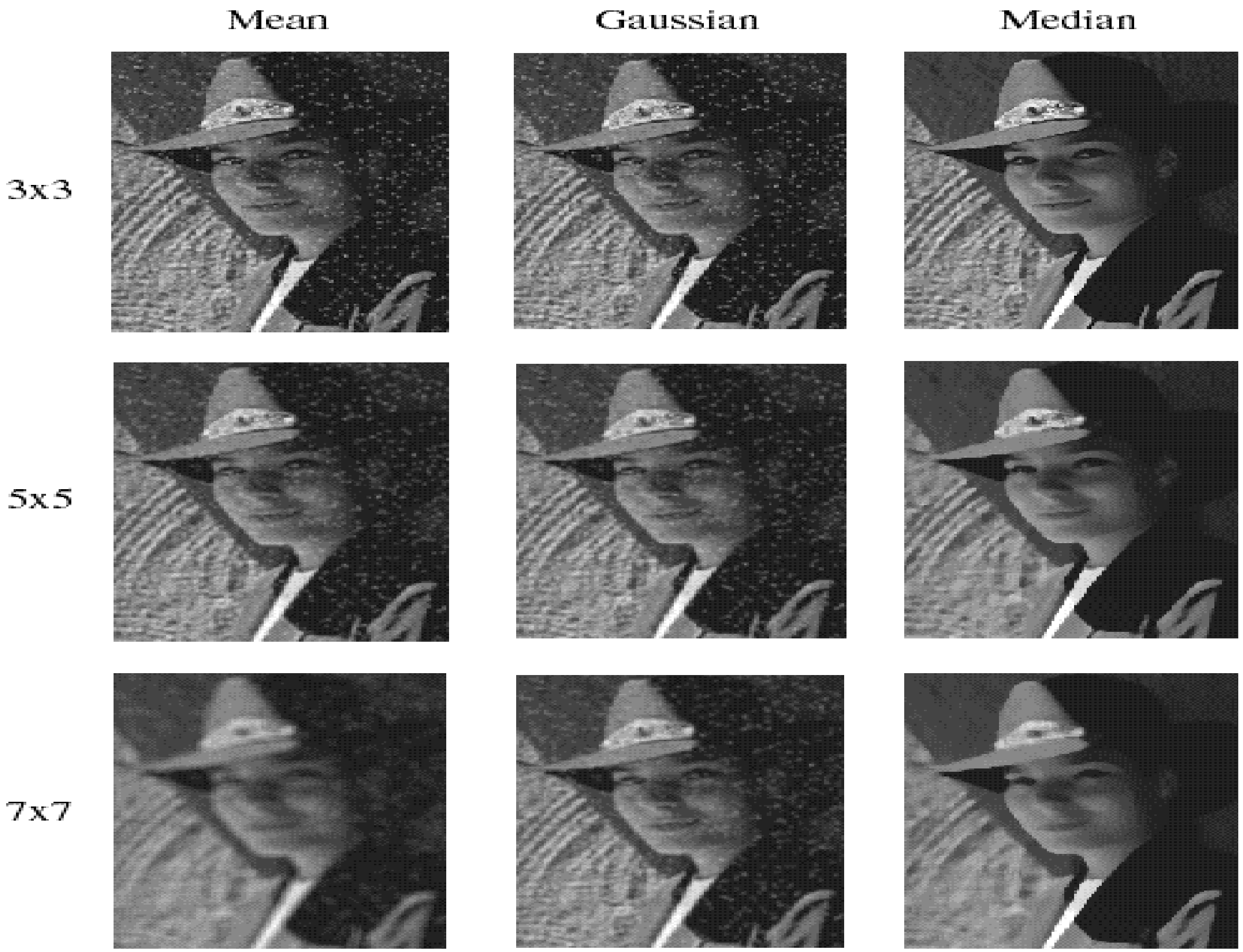
	0	10	20	30	30				
				90					

$$h[m,n]=\sum_{k,l} g[k,l]\; f[m+k,n+l]$$

Credit: S. Seitz

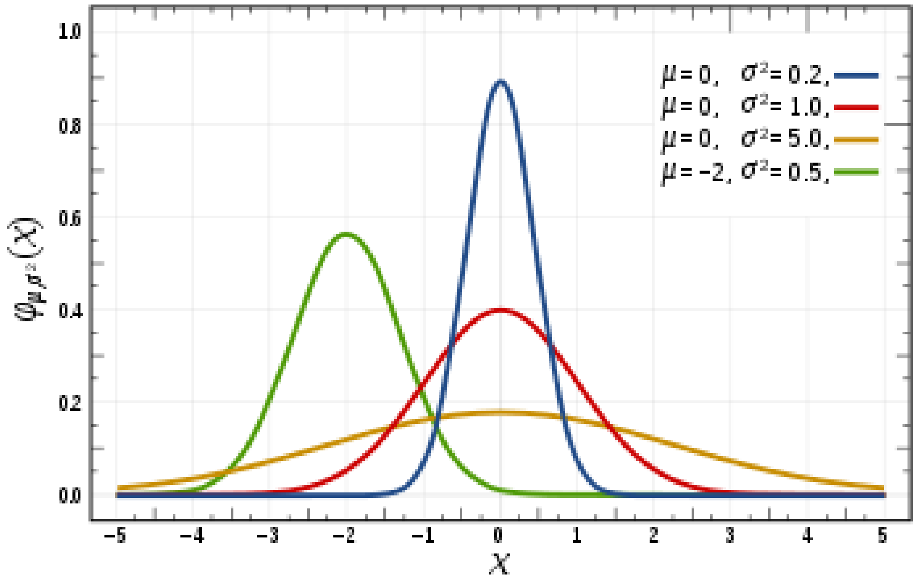
UCF Center for Research in Computer Vision

Median Filter



Practical matters

- How big should the filter be?
  - Values at edges should be near zero
  - Gaussians have infinite extent…
  - Rule of thumb for Gaussian: set filter half-width to about 3 σ



James Hays

Practical matters

What about near the edge?

- The filter window falls off the edge of the image
- Need to extrapolate
- methods:
  - clip filter (black)
  - wrap around
  - copy edge
  - reflect across edge

Source: S. Marschner

Questions?

UCF CENTER FOR RESEARCH IN COMPUTER VISION

CAP5415
Computer Vision

Yogesh S Rawat

yogesh@ucf.edu

HEC-241

Questions?

Image Filtering

Lecture 3

Image Derivates

Derivatives

- Derivative: rate of change
  - Speed is a rate of change of a distance, X=V.t
  - Acceleration is a rate of change of speed, V=a.t

Derivative

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x) = f_x$$

$$y = x^2 + x^4$$

$$\frac{dy}{dx} = 2x + 4x^3$$

$$y = \sin x + e^{-x}$$

$$\frac{dy}{dx} = \cos x + (-1)e^{-x}$$

Discrete Derivative

$$\frac{df}{dx} = \lim_{\Delta x \to 0} \frac{f(x) - f(x - \Delta x)}{\Delta x} = f'(x)$$

$$\frac{df}{dx} = \frac{f(x) - f(x-1)}{1} = f'(x)$$

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

Discrete Derivative / Finite Difference

$$\frac{df}{dx} = f(x) - f(x-1) = f'(x)$$

Backward difference

$$\frac{df}{dx} = f(x) - f(x+1) = f'(x)$$

Forward difference

$$\frac{df}{dx} = f(x+1) - f(x-1) = f'(x)$$

Central difference

Example: Finite Difference

$$f(x) = 10 \quad 15 \quad 10 \quad 10 \quad 25 \quad 20 \quad 20 \quad 20$$

$$f'(x) = 0 \quad 5 \quad -5 \quad 0 \quad 15 \quad -5 \quad 0 \quad 0$$

$$f''(x) = 0 \quad 5 \quad 10 \quad 5 \quad 15 \quad -20 \quad 5 \quad 0$$

Derivative Masks

Backward difference	[-1 1]
Forward difference	[1 -1]
Central difference	[-1 0 1]

Derivative in 2-D

Given function $f(x, y)$

Gradient vector $\nabla f(x, y) = \begin{bmatrix} \dfrac{\partial f(x, y)}{\partial x} \\ \dfrac{\partial f(x, y)}{\partial y} \end{bmatrix} = \begin{bmatrix} f_x \\ f_y \end{bmatrix}$

Gradient magnitude $|\nabla f(x, y)| = \sqrt{f_x^2 + f_y^2}$

Gradient direction $\theta = \tan^{-1} \dfrac{f_x}{f_y}$

Derivative of Images

Derivative masks $f_x \Rightarrow \frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$ $f_y \Rightarrow \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \qquad I_x = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & \boxed{10} & \boxed{10} & 0 & 0 \\ 0 & 10 & 10 & 0 & 0 \\ 0 & 10 & 10 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

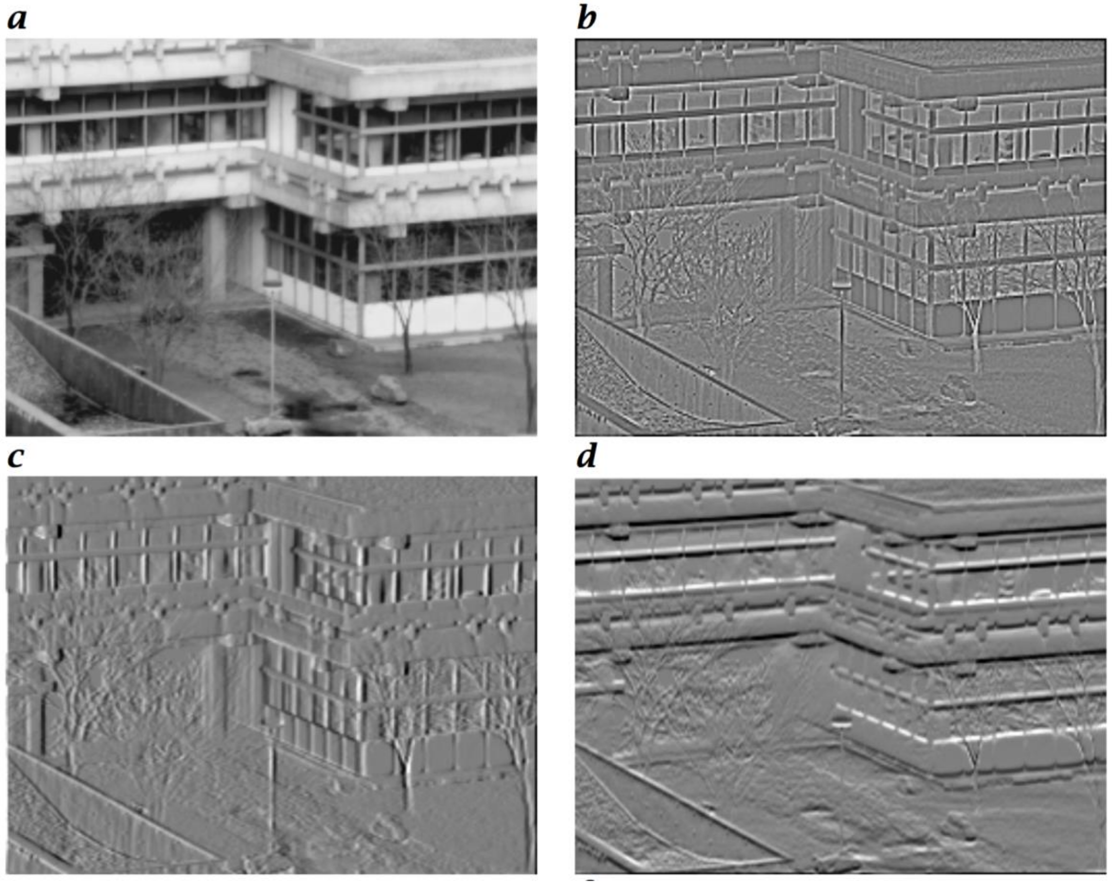
Derivative of Images

Derivative masks

$$f_x \Rightarrow \frac{1}{3}\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad f_y \Rightarrow \frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

$$I = \begin{bmatrix} 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \\ 10 & 10 & 20 & 20 & 20 \end{bmatrix} \qquad I_y = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Example



a. Original image
b. Laplacian operator
c. Horizontal derivative
d. Vertical derivative

Questions?

Sources for this lecture include materials from works by Mubarak Shah, S. Seitz, James Tompkin and Ulas Bagci