Image Processing & Vision

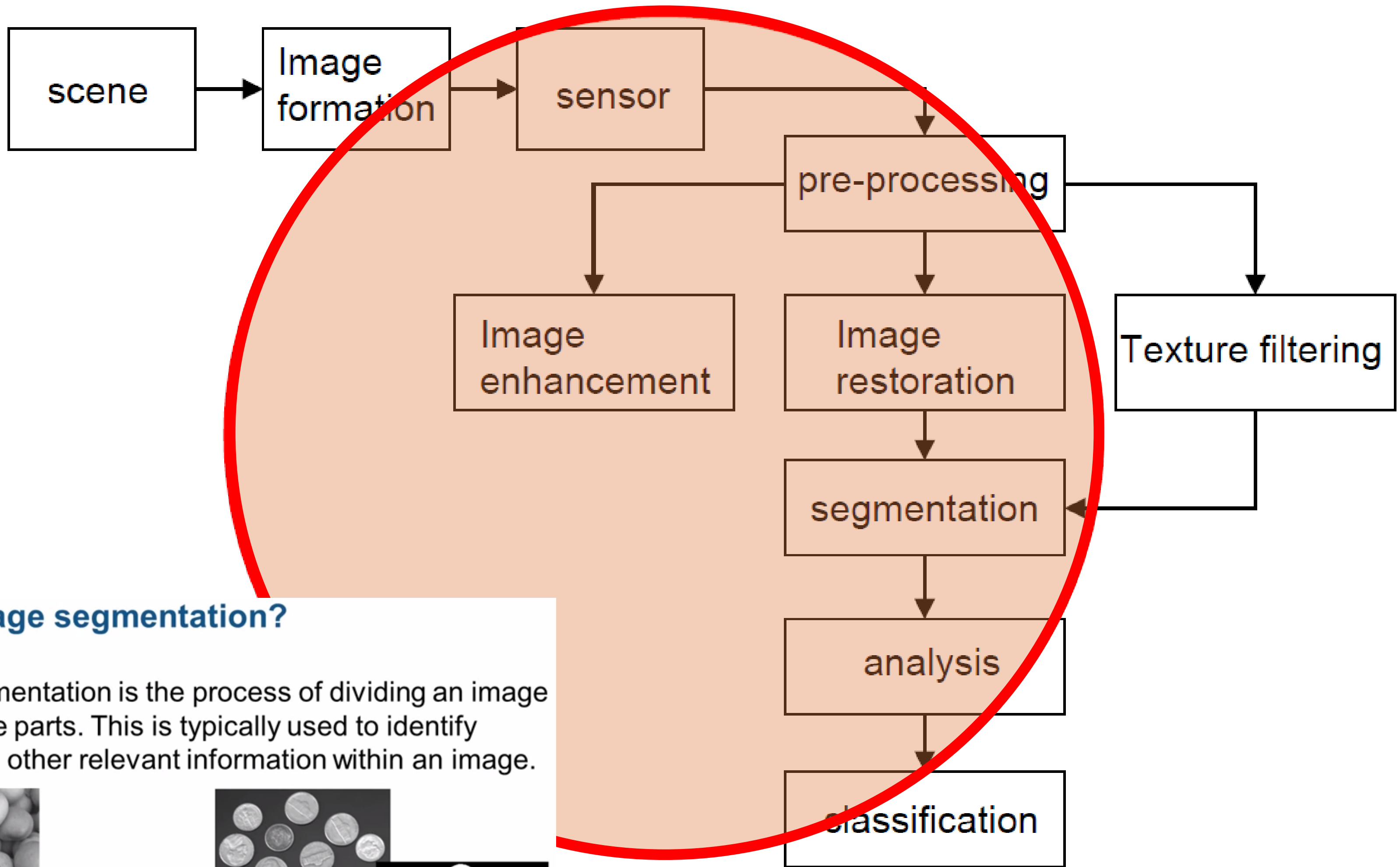
Week 2

Content

- **Recap:**
  - Matlab, DIPimage, loading and display image

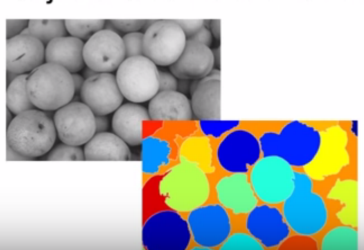
- Image filtering 1

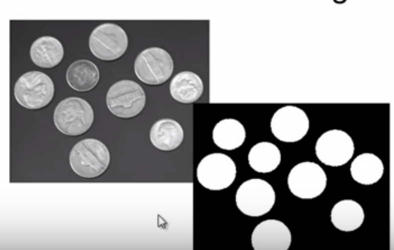
Image Analysis Paradigm



What is image segmentation?

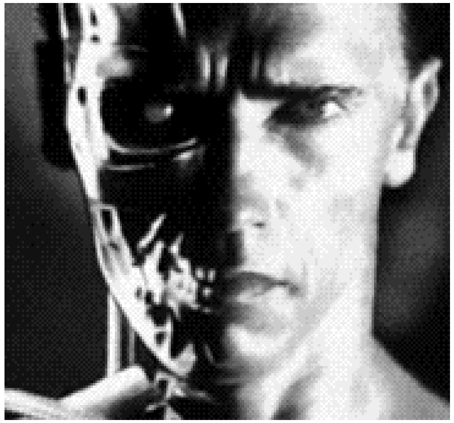
- Image segmentation is the process of dividing an image into multiple parts. This is typically used to identify objects and other relevant information within an image.

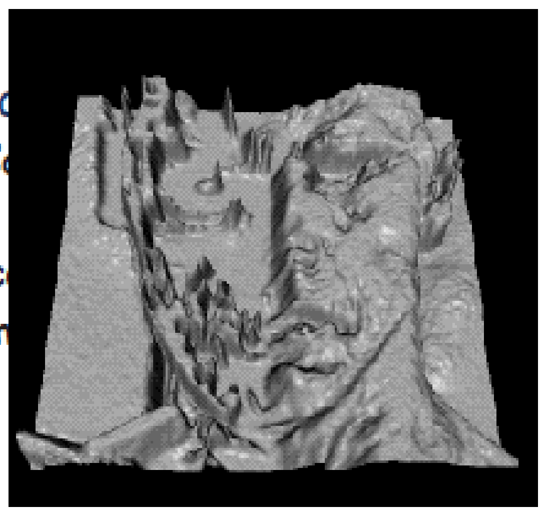


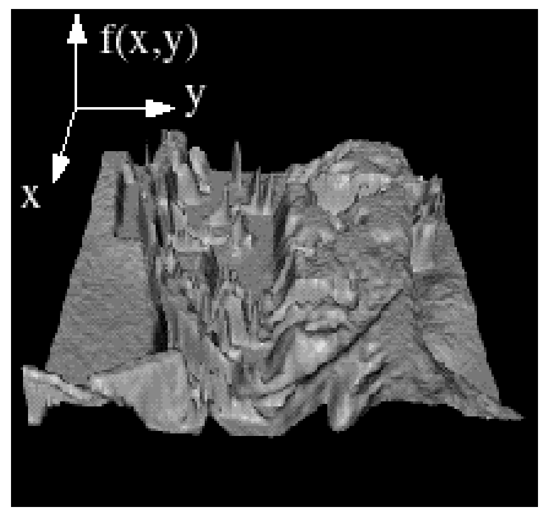


What are images…

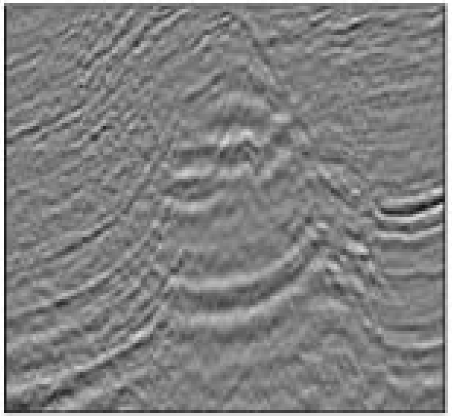
- An imag … of
  position
  - Radia
  - Acous
  - Surfa
- 2-D ima
  - Projec
- 3-D ima
  - CT: Co
  - MRI:
  - Optic
  - Seism

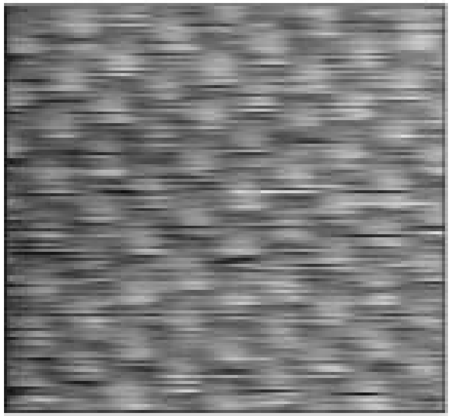




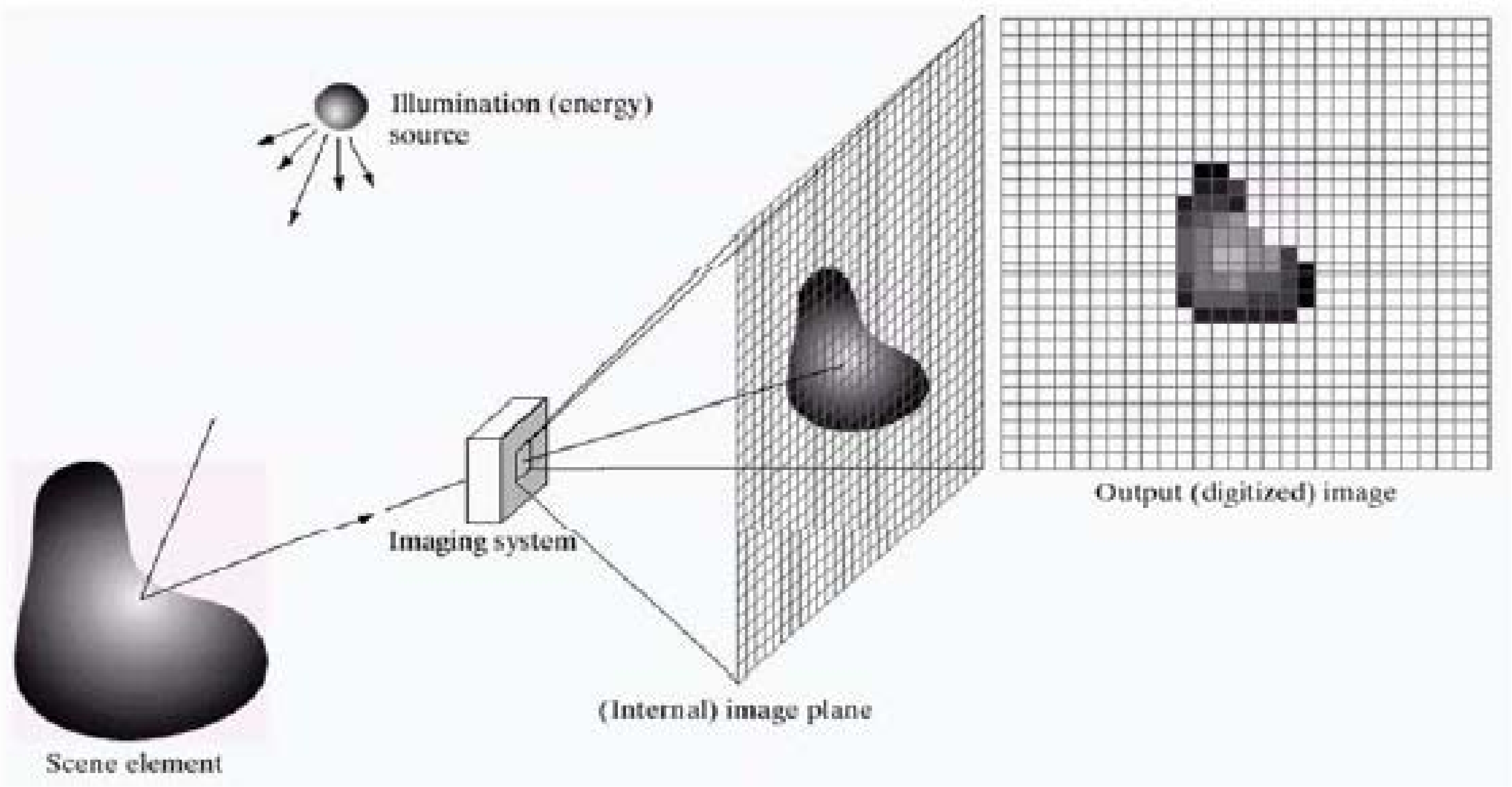


Examples of Images





Digital Image Acquisition Process



Sampling and Quantization

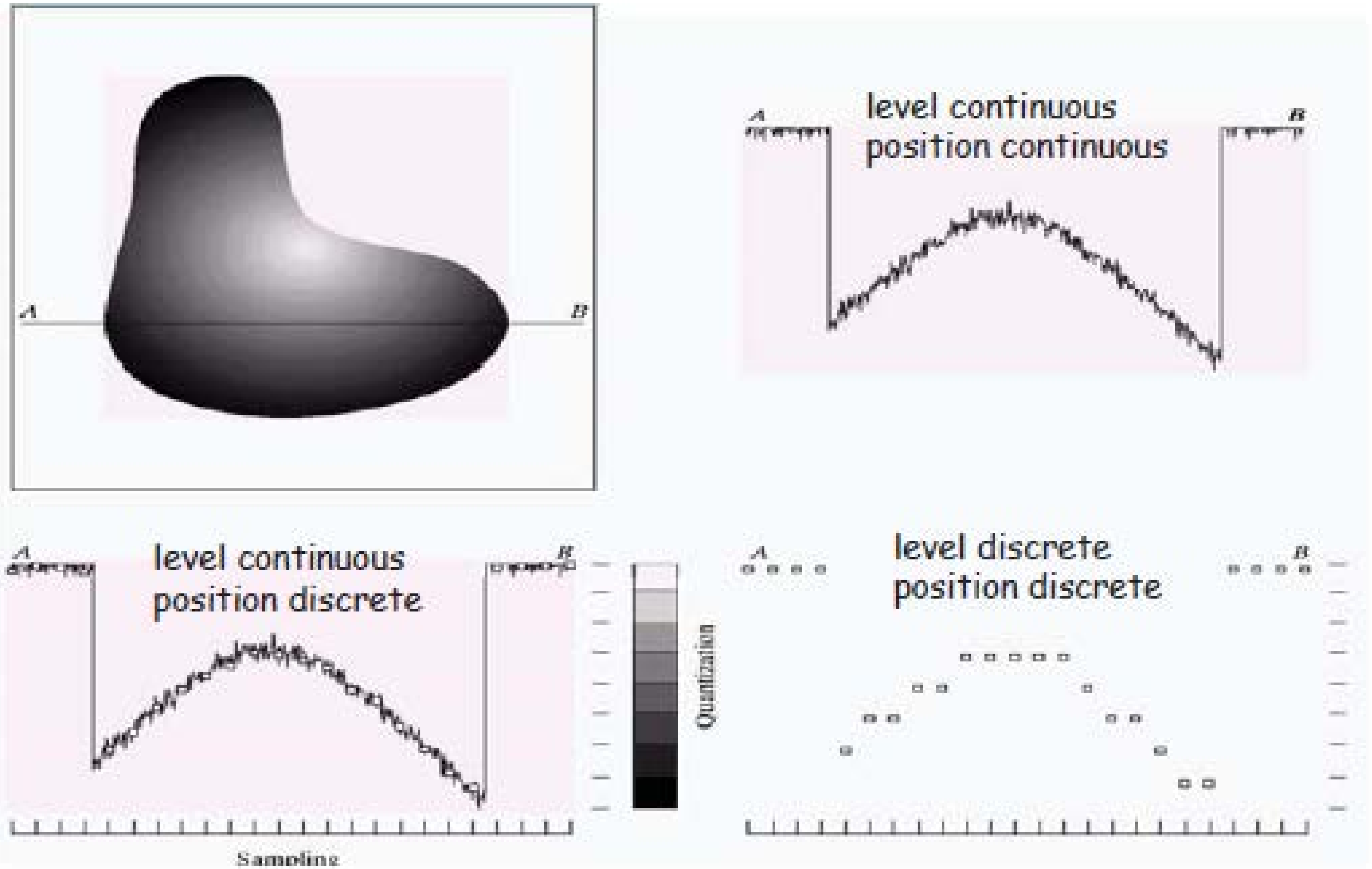
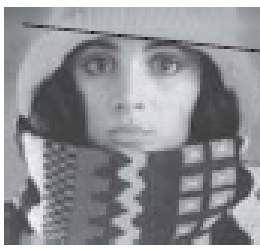


Image Representation

Image can be represented as a matrix

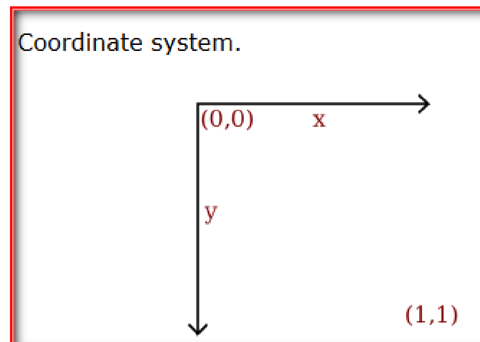


I(x,y)

$$
\begin{pmatrix}
100 & 103 & 105 & 100 & 104 & 103 & 198 & 202 & 201 & 197 \\
103 & 103 & 100 & 102 & 103 & 104 & 200 & 199 & 207 & 199 \\
96 & 106 & 98 & 105 & 107 & 202 & 195 & 189 & 203 & 201 \\
105 & 110 & 103 & 96 & 104 & 200 & 190 & 187 & 210 & 209 \\
99 & 98 & 109 & 110 & 108 & 199 & 199 & 189 & 210 & 200 \\
103 & 102 & 101 & 105 & 157 & 200 & 201 & 206 & 198 & 211 \\
89 & 110 & 107 & 103 & 199 & 210 & 219 & 199 & 205 & 201 \cdots \\
100 & 103 & 104 & 178 & 203 & 201 & 199 & 199 & 201 & 205 \\
105 & 103 & 98 & 189 & 199 & 198 & 201 & 200 & 198 & 203 \\
 & & & & & \vdots & & & &
\end{pmatrix}
$$

M(m,n)

I(5,4)=M(5,4)=104

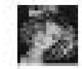


Sampling

256x256

128x128

64x64



32x32

16x16

Effects of Quantization


8 bit


7 bit


6 bit


5 bit

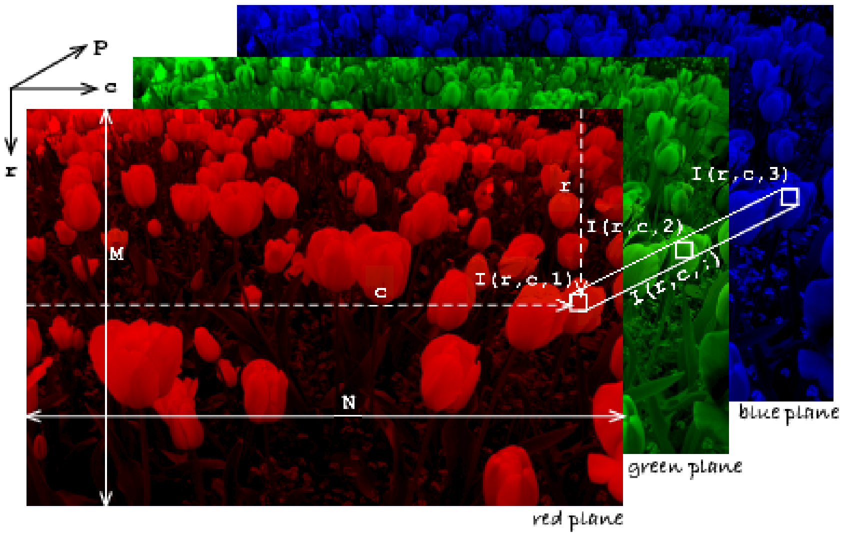

4 bit


3 bit

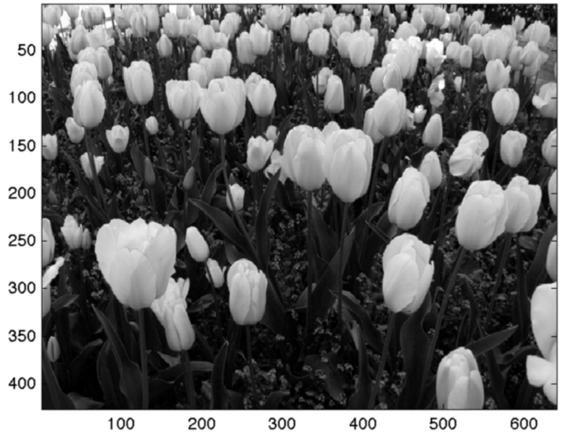

2 bit


1 bit

Color channels

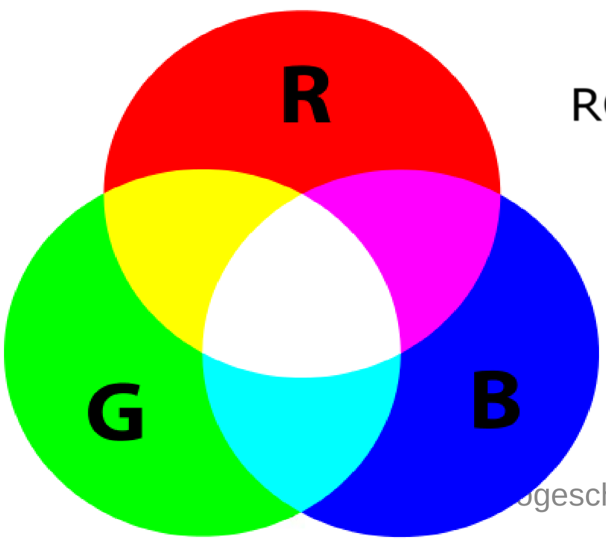


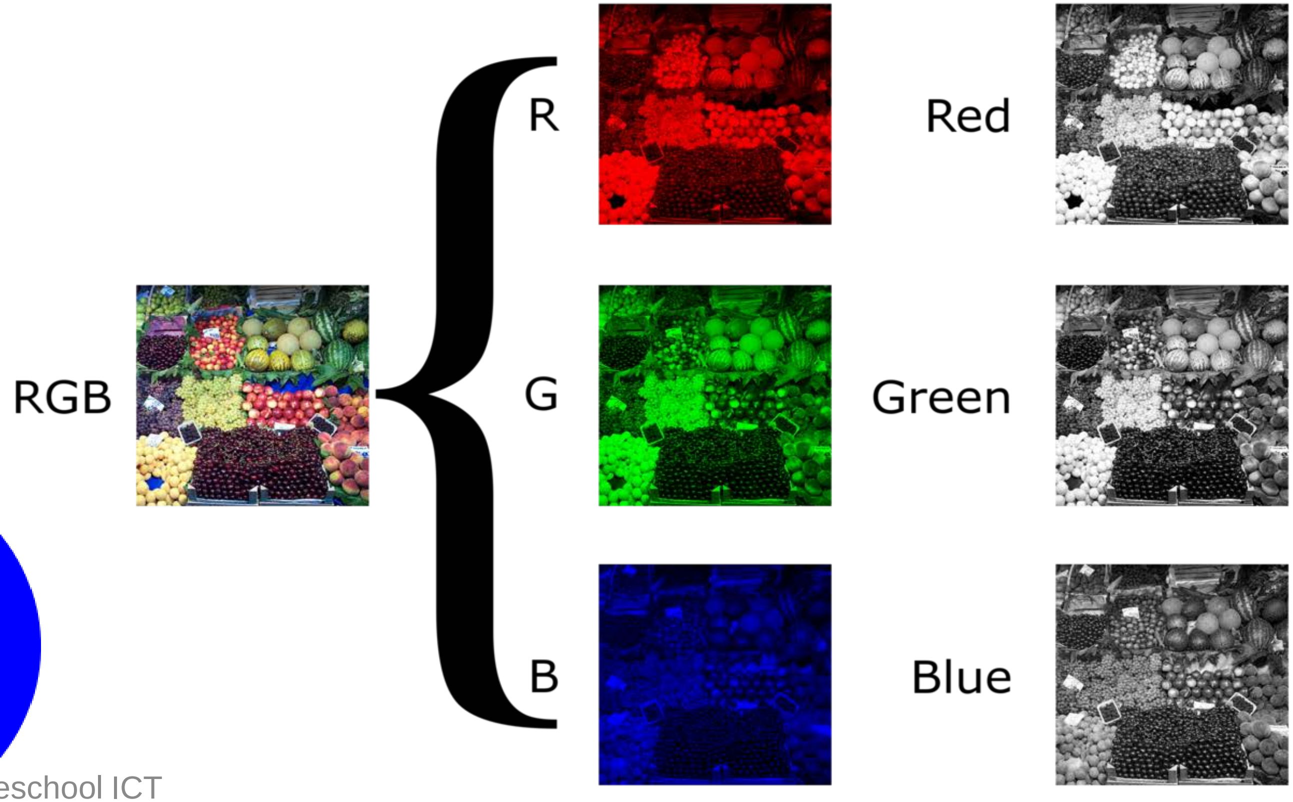
multi-channel images
(Blue, Green, Red)



Example:

- color channel splitting of a full RGB color image
  - Here is The column at left shows the isolated color channels in natural colors, while at right there are their grayscale equivalence
  - https://nl.mathworks.com/help/matlab/creating_plots/image-types.html



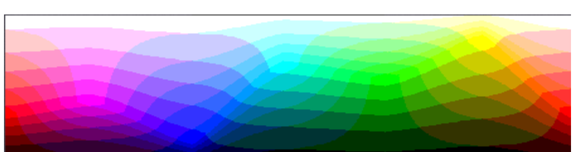


Bit depth/channels

- Bit depth is the number of bits per pixel
- Quantifies how many unique colors are available in an image's color palette

Bits Per Pixel	Number of Colors Available	Common Name(s)
1	2	Monochrome
2	4	CGA
4	16	EGA
8	256	VGA
16	65536	XGA, High Color
24	16777216	SVGA, True Color
32	16777216 + Transparency	
48	281 Trillion	

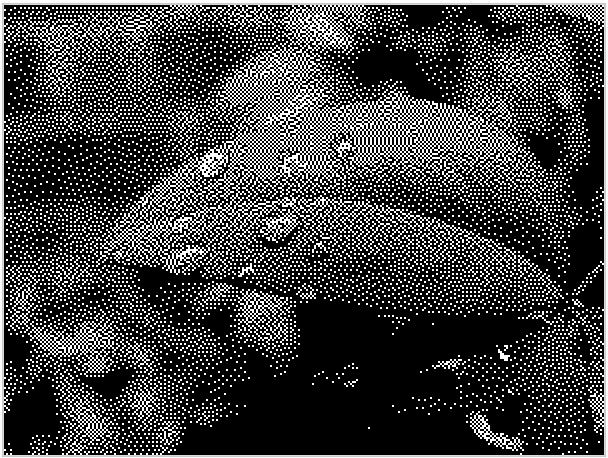
Every color pixel in a digital image is created through some combination of the three primary colors: red, green, and blue.

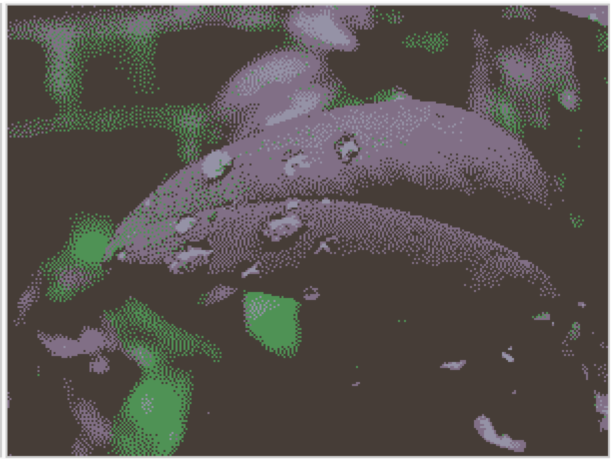


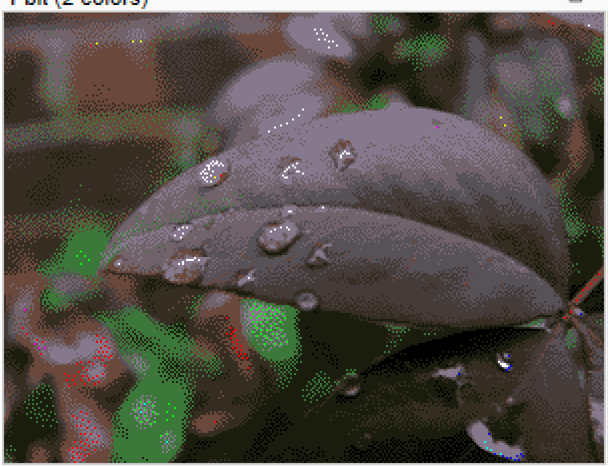
8 bits

24 bits

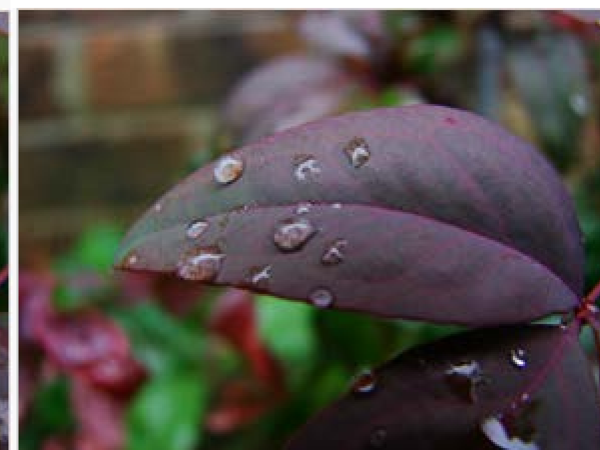
Bit depth


1 bit (2 colors)

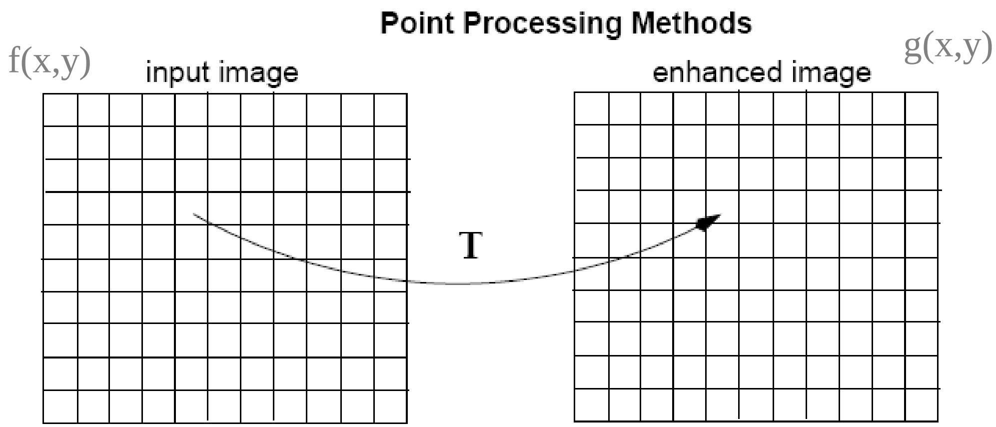

2 bits (4 colors)


4 bits (16 colors)


8 bits (256 colors)

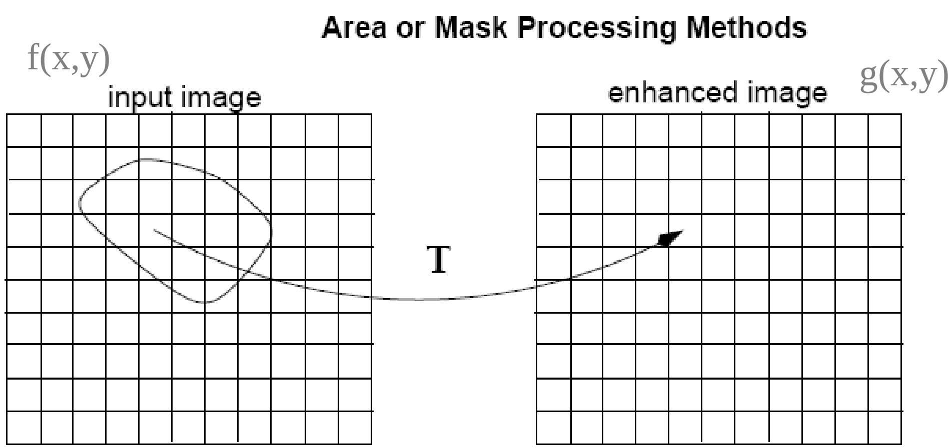

24 bits (16,777,216 colors, "truecolor")

Spatial Domain Methods



$$g(x,y) = T[f(x,y)]$$

T operates on 1 pixel



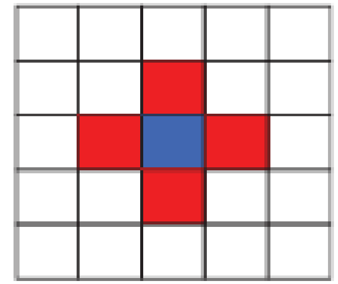
$$g(x,y) = T[f(x,y)]$$

T operates on a neighborhood of pixels

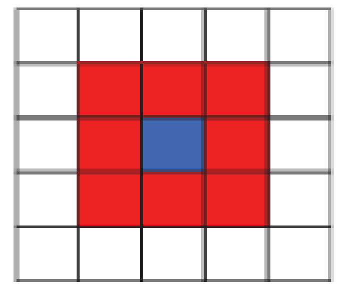
Connectivity

p=

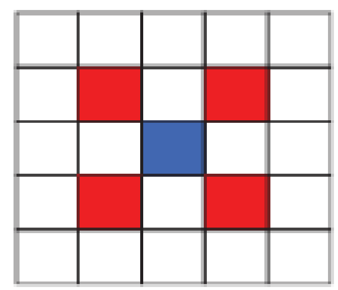
4-connected neighbours $N_4(p)$



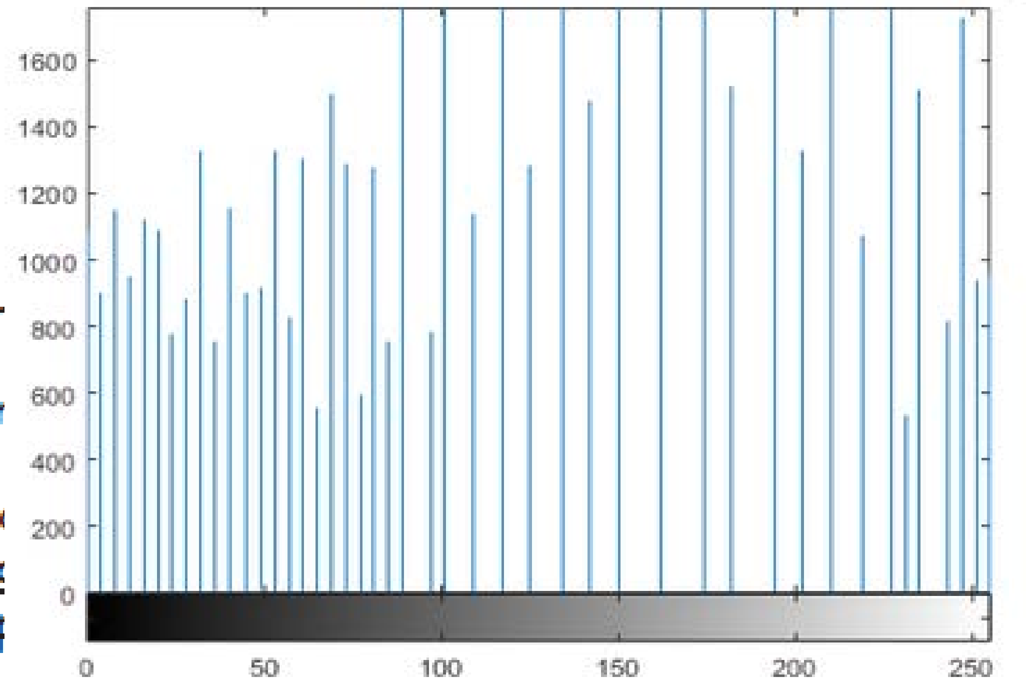
8-connected neighbours $N_8(p)$



Diagonal neighbours $N_D(p)$



Histogram

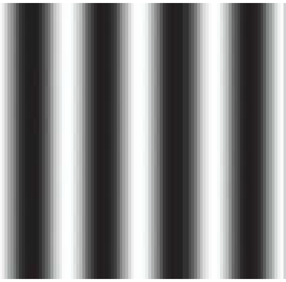


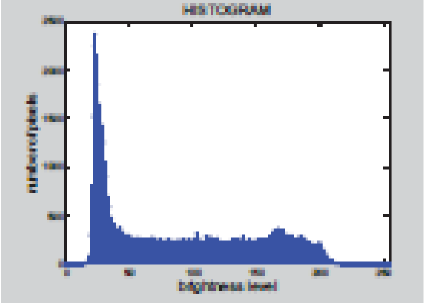
Histogram H

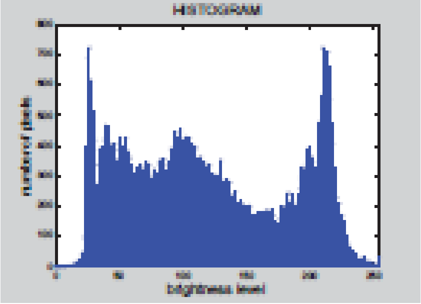
- a count of the n
  an image
- easy to comput
  levels in an imag
- related to the p
  image

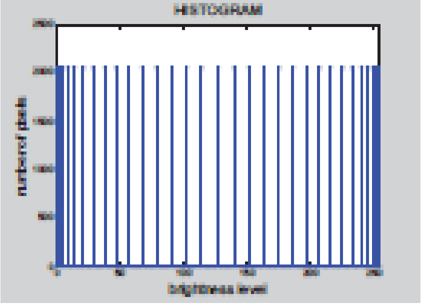
- H(i)=number of pixels in image with gray level I
- Gray level sometimes called brightness or intensity
- Number of levels (also called number of bins) depends on gray level quantization (e.g. 8 bit image has 256 levels)
- H(i) is a discrete function
- $H(i) \geq 0$ and $H(i) \leq$ #pixels in image for all i.

Histogram Examples



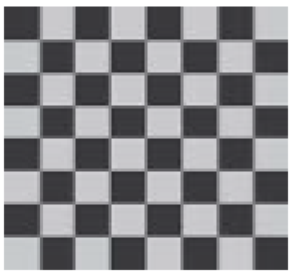


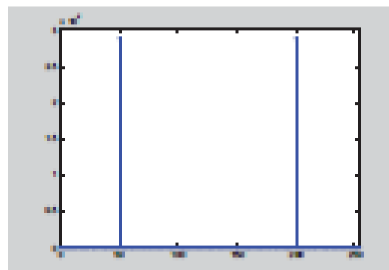


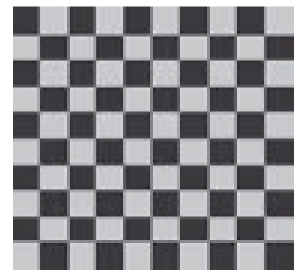


Histogram Properties

Every image has a unique histogram but not vice versa







Total number of pixels is sum over all brightness levels i

$$\#\text{pixels} = \sum_{i=0}^{i=max} H(i)$$

Hint

Image Filtering

Image Filtering

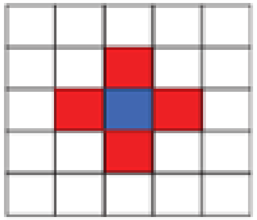
- Basic (low level) image processing operations:
  - Filtering
  - Enhancement
- Usually important pre-processing steps in computer vision systems (often in hardware)

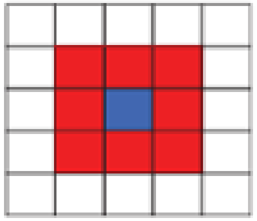
Content

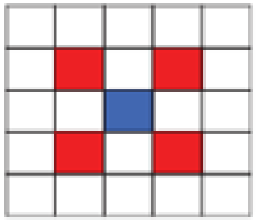
- Linear and nonlinear spatial filtering
  - linear filters: smoothen/sharpen the edges in an image
  - Understand how the two operations are related
- Edge enhancement←→edge detection

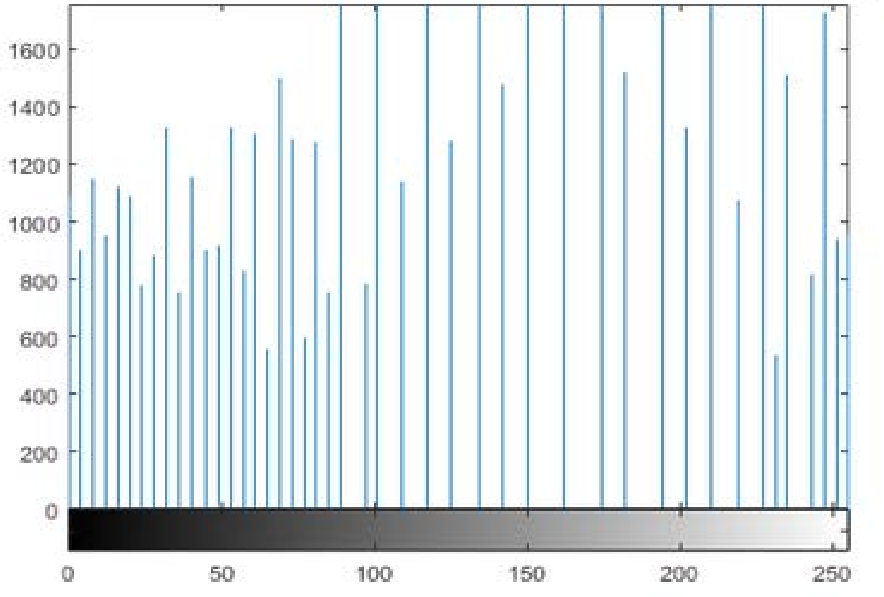
Image Filtering operations

- Simple image operations:
  - 'neighbourhood'
    - More general filtering operations use!!!
  - 'pointwise'
    - Histogram equalisation









Fontys

Linear & non-linear filters

Basically there are two types of filters:

Linear Filters

- GaussianFilter
- DerivativeFilter
- MeanFilter
- GradientFilter …

Nonlinear Filters

- MedianFilter
- MinFilter
- MaxFilter
- KuwaharaFilter …

Linear filters can be implemented by a **convolution**
But non-linear filters can not.

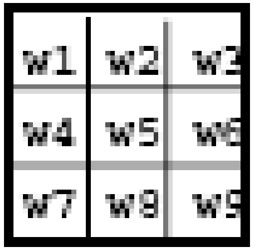
Convolution Kernels

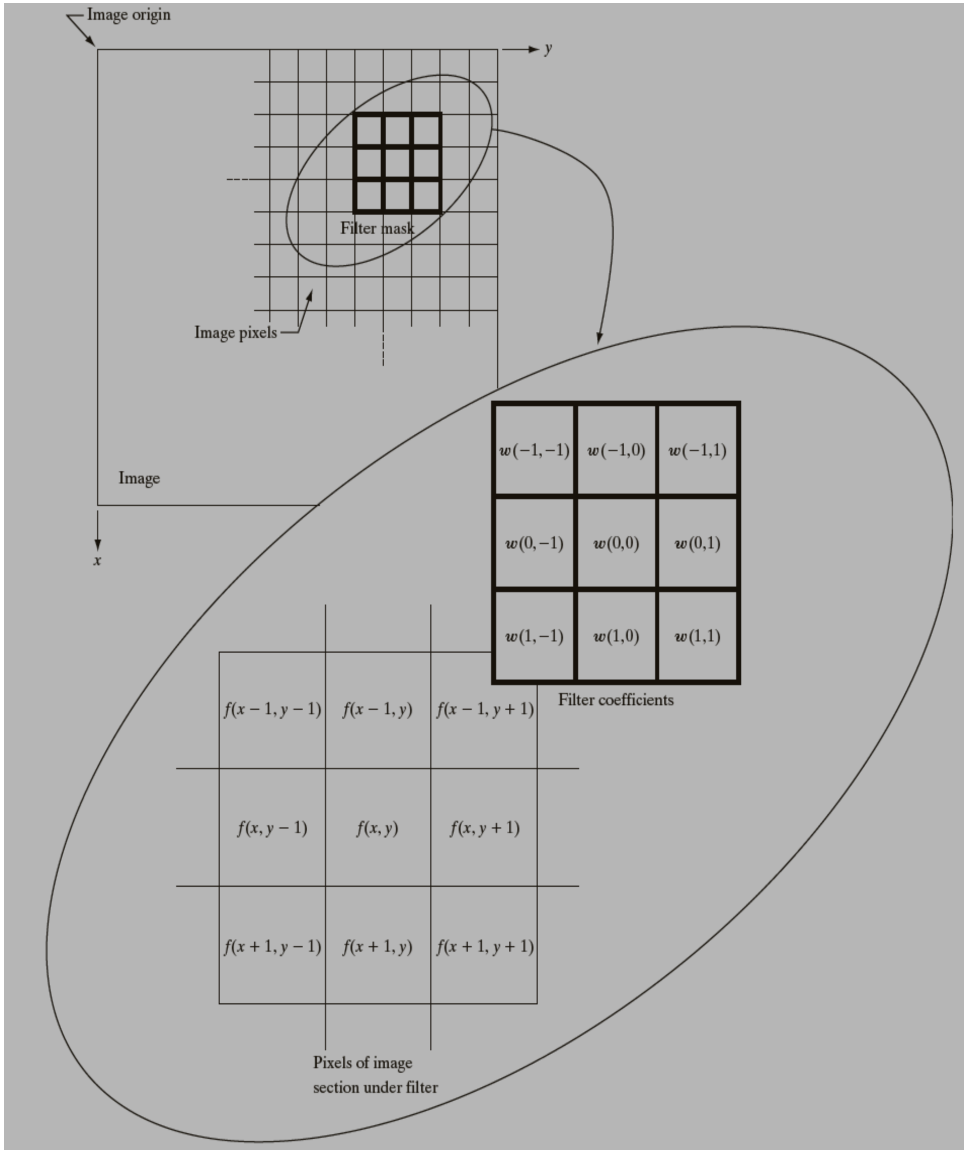
- DiskMatrix
- BoxMatrix …

Linear Operation

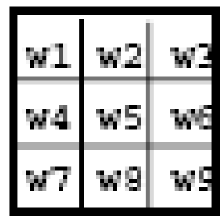
- Typically linear combinations of pixel values.
  - e.g., weight pixel values and add them together.
- Applications: by using different weights.
  - e.g., smoothing, sharpening, edge detection.

mask





Example



10	5	3
4	6	1
1	1	8

Local image neighborhood

0	0	0
0	0.5	0
0	1	0.5

mask

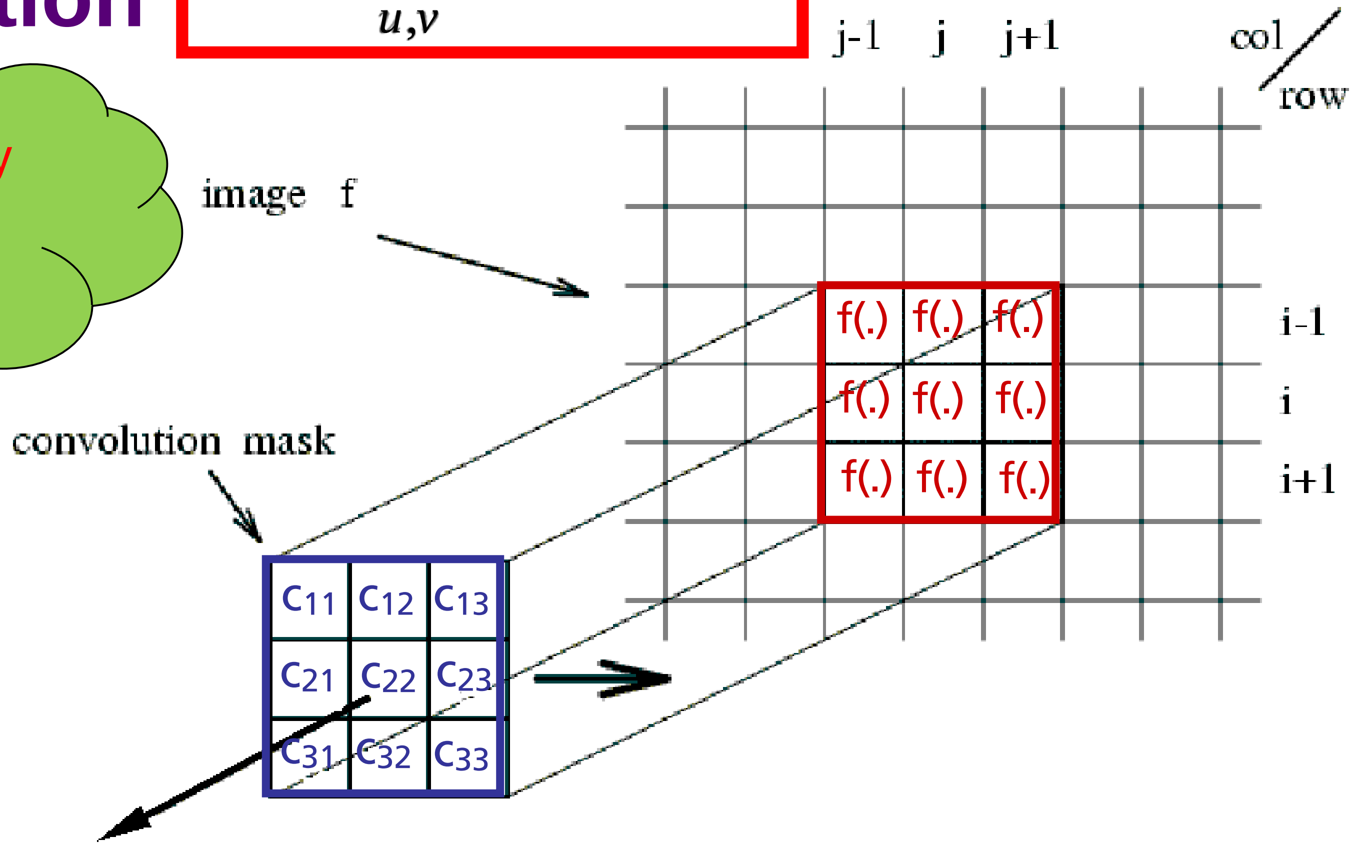
	8	

Modified image data

Convolution

$$R_{ij} = \sum_{u,v} H_{i-u,j-v} F_{uv}$$

'overlap-multiply -add' with 'convolution mask'



$$o(i,j) = c_{11}\, f(i-1,j-1) + c_{12}\, f(i-1,j) + c_{13}\, f(i-1,j+1) + c_{21}\, f(i,j-1) + c_{22}\, f(i,j) + c_{23}\, f(i,j+1) + c_{31}\, f(i+1,j-1) + c_{32}\, f(i+1,j) + c_{33}\, f(i+1,j+1)$$

Linear filter: Gaussian Filtering(demo)

- Properties
  - Output is a linear function of the input
  - Output is a shift-invariant function of the input
- Gaussian kernel gives less weight to pixels further from the center of the window
- This kernel is an approximation of a Gaussian function:

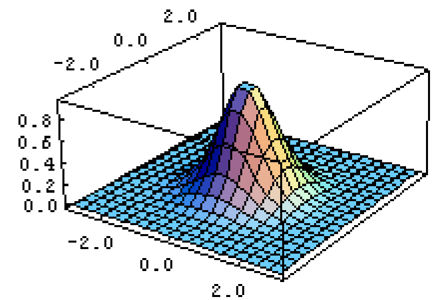
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$F[x, y]$

$$\frac{1}{16}\boxed{\begin{array}{|c|c|c|}\hline 1 & 2 & 1 \\ \hline 2 & 4 & 2 \\ \hline 1 & 2 & 1 \\ \hline\end{array}}$$

$H[u, v]$

$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$



Demo

- http://setosa.io/ev/image-kernels/