22.05.2022

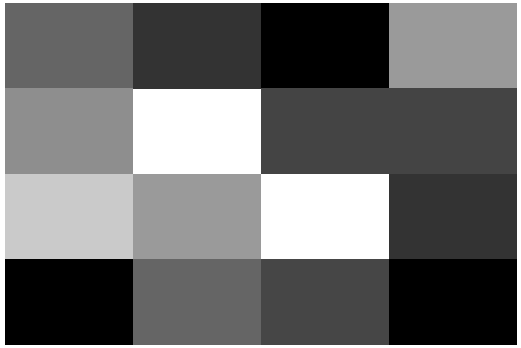
Introduction to Image Processing

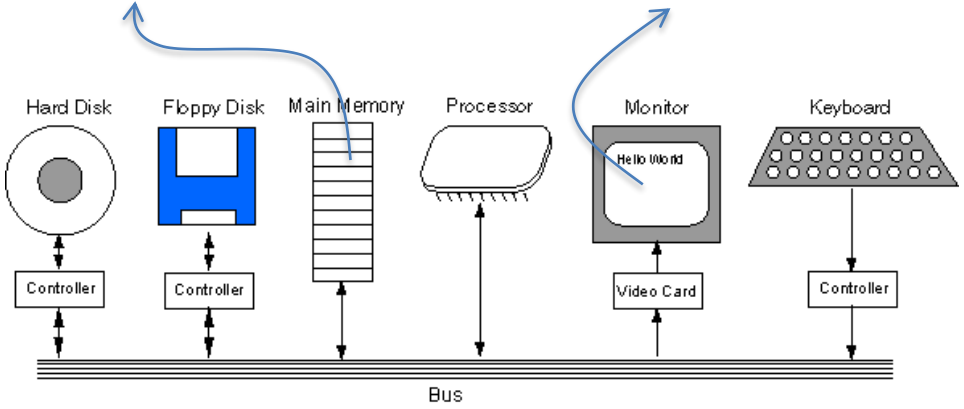
Ravi Kiran

What is a **digital** image ?

- An **array** of numbers representing color intensities

100	50	0	150
90	255	70	70
200	150	255	50
0	100	80	0





http://www.cs.ucc.ie/~gavin/cs1001/Notes/chap01/ch01_4.html

What is a **digital** image ?

- An **array** of numbers representing color intensities

100	50	0	150
90	255	70	70
200	150	255	50
0	100	80	0

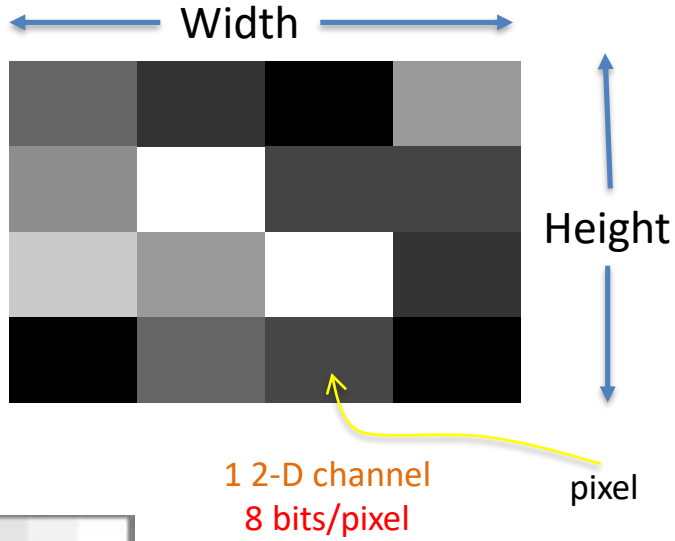
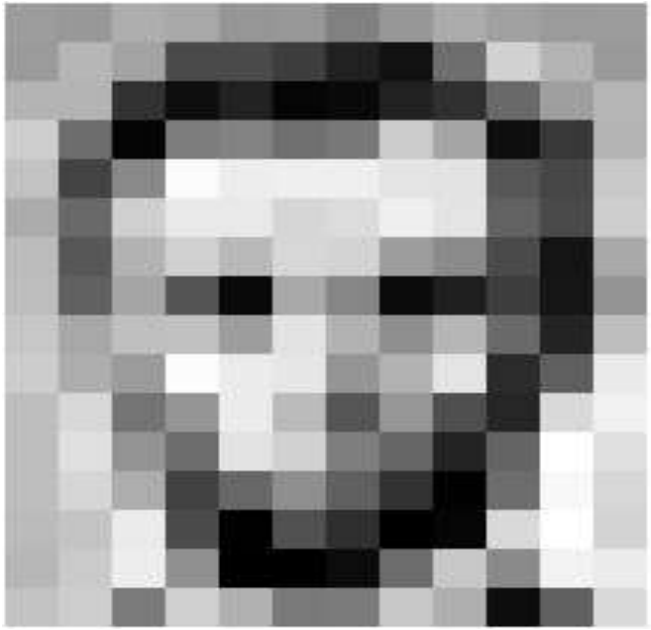


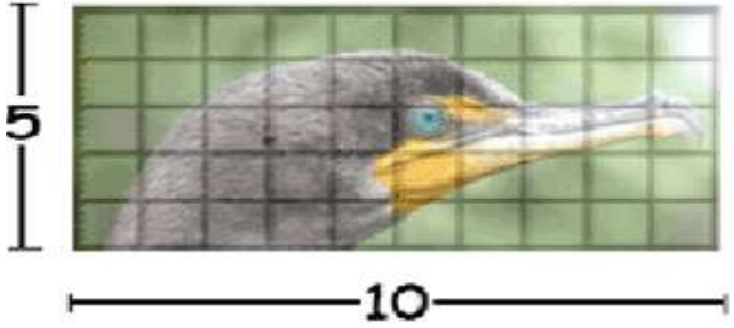
Image Representation (Grayscale)

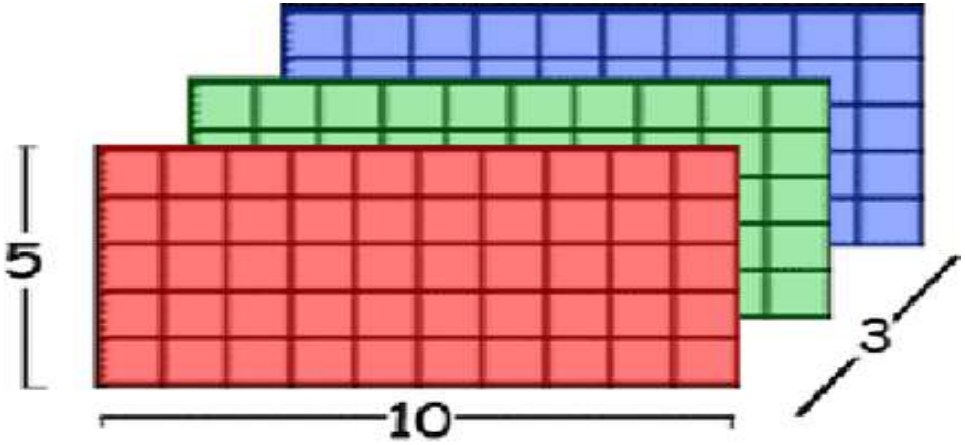


157	153	174	168	150	152	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	86	150	79	38	218	241
190	224	147	108	227	210	127	102	36	101	255	224
190	214	173	66	103	143	96	50	2	109	249	215
187	196	235	75	1	81	47	0	6	217	255	211
183	202	237	145	0	0	12	108	200	138	243	236
195	206	123	207	177	121	123	200	175	13	96	218

https://openframeworks.cc/ofBook/images/image_processing_computer_vision/

Image Representation (RGB)

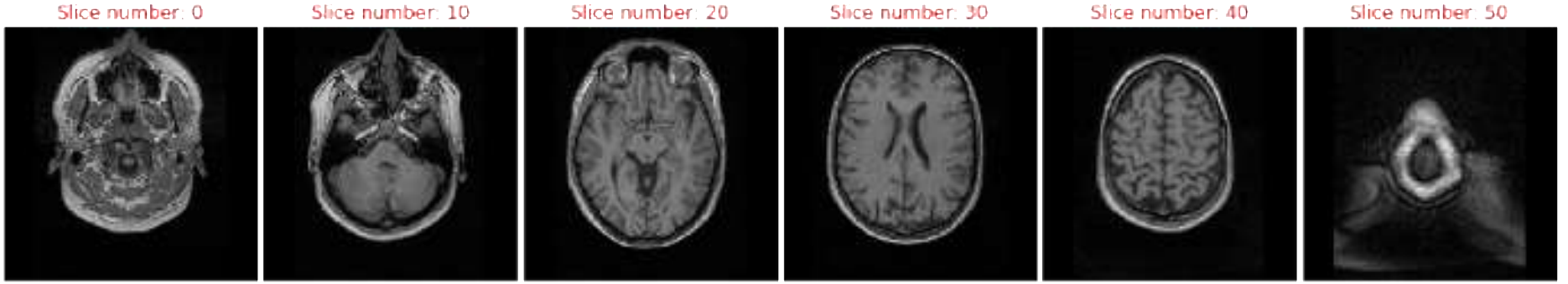




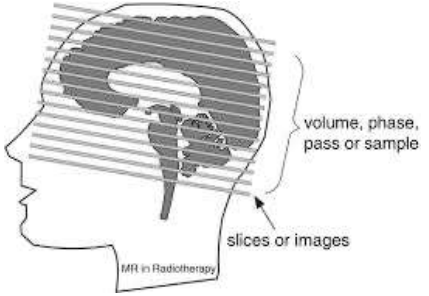
3 2-D channels
24 bits per pixel

Image Representation

fMRI image slices



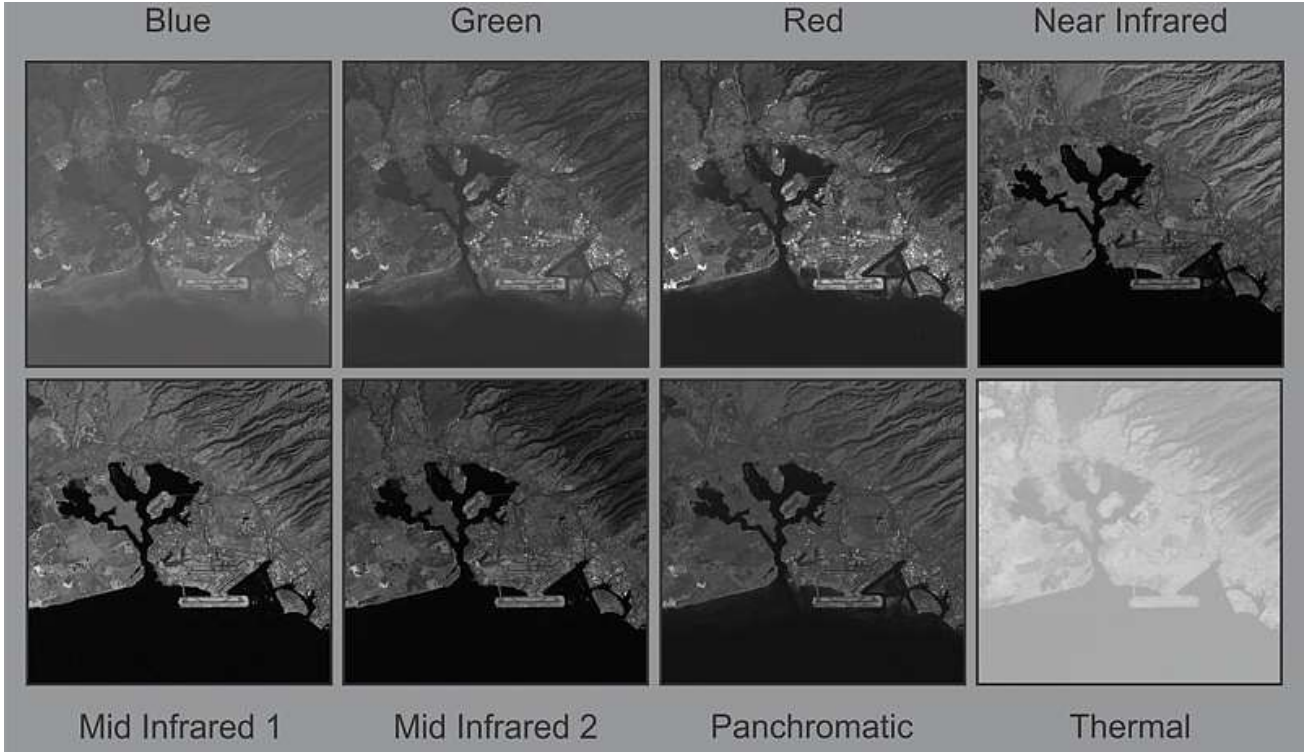
56 2-D channels



Types on images (classification on arrangement)

- Grayscale
- RGB
- Multispectral images
- Stereo images
- Multi-view images

Multi spectral images



Courtesy: LANDSAT

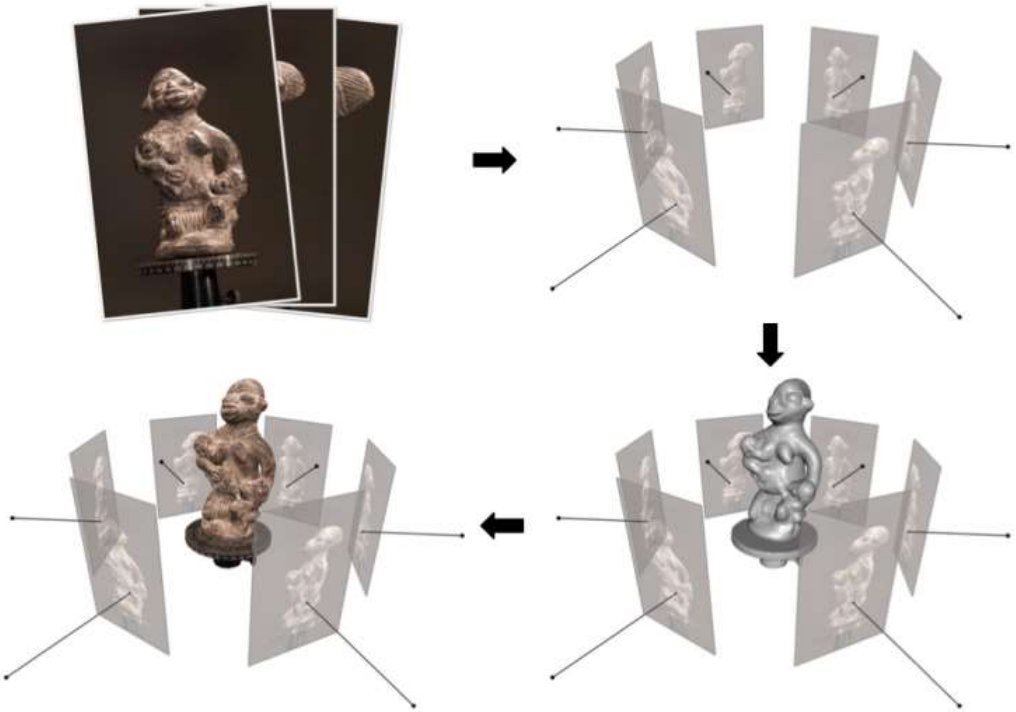
Stereo Images

courtesy: wikimedia.com

Stereo Images

courtesy: vision.middlebury.edu

Multi-view images



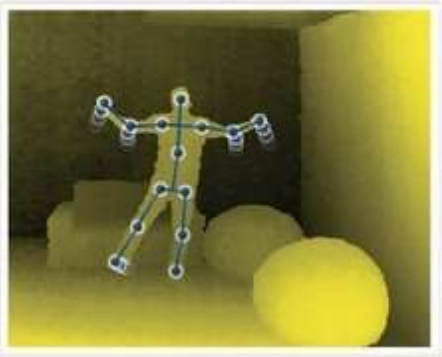
courtesy: Yasutaka Furukawa

courtesy: Sameer Agarwal

Kinect images

Color (RGB) Image

Depth Image

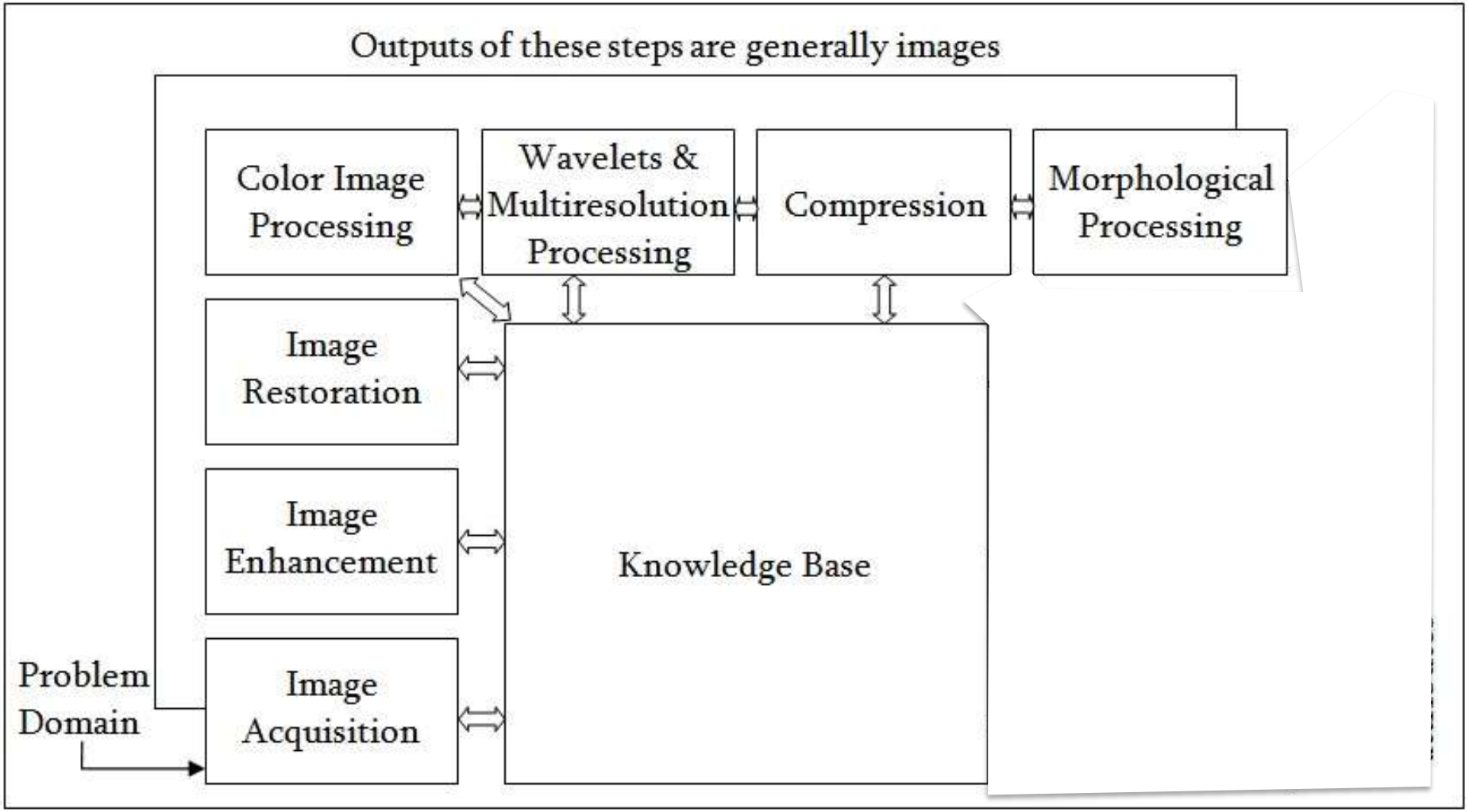


courtesy: kinect and prime sense

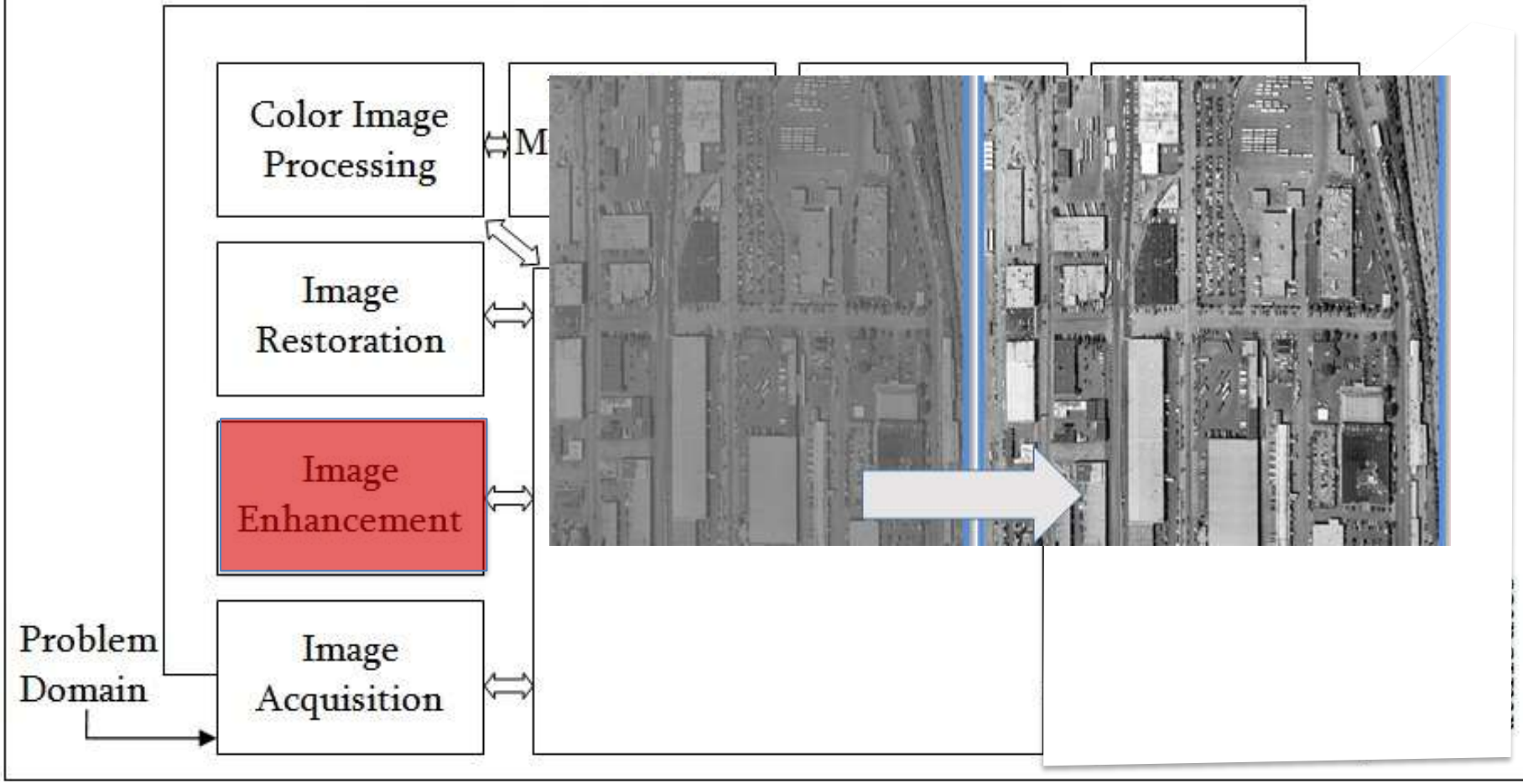
Digital Image **processing**

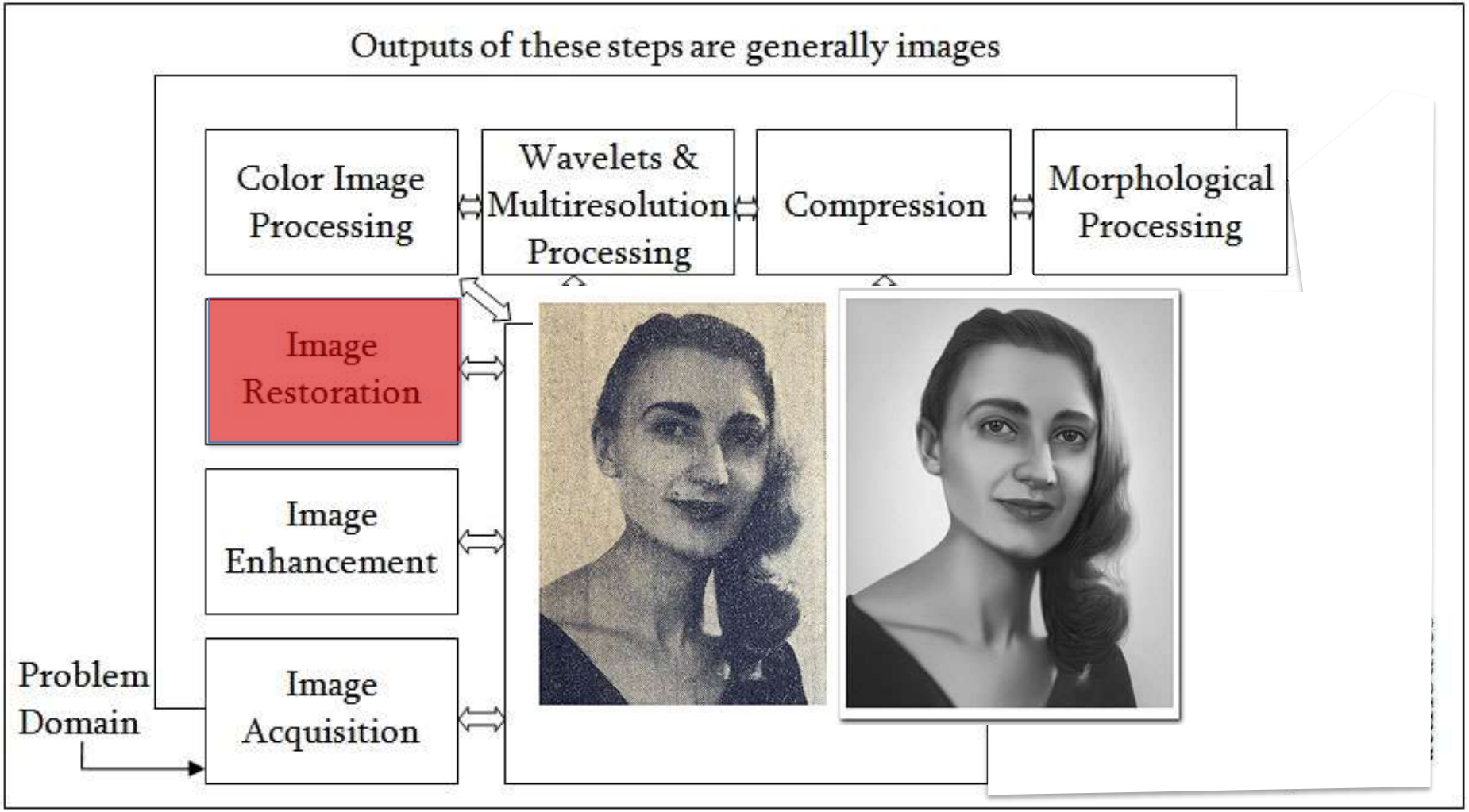
- Study and Development of algorithms that operate on an image
  - To create new image(s)
  - To retrieve its attributes

- Consumer-based view
  - For consumption by human eyes
  - For consumption by machine-based processes


Outputs of these steps are generally images
Color Image Processing
Wavelets & Multiresolution Processing
Compression
Morphological Processing
Image Restoration
Image Enhancement
Knowledge Base
Problem Domain
Image Acquisition

Outputs of these steps are generally images




Outputs of these steps are generally images
Color Image Processing
Wavelets & Multiresolution Processing
Compression
Morphological Processing
Image Restoration
Image Enhancement
Image Acquisition
Problem Domain

Outputs of these steps are generally images

Color Image Processing

Wavelets & Multiresolution Processing

Compression

Morphological Processing

Image Restoration

Image Enhancement

Image Acquisition

Problem Domain

Uncorrected Image

With .Look Applied

Outputs of these steps are generally images

Color Image Processing

Wavelets & Multiresolution Processing

Compression

Morphological Processing

Image Restoration

Image Enhancement

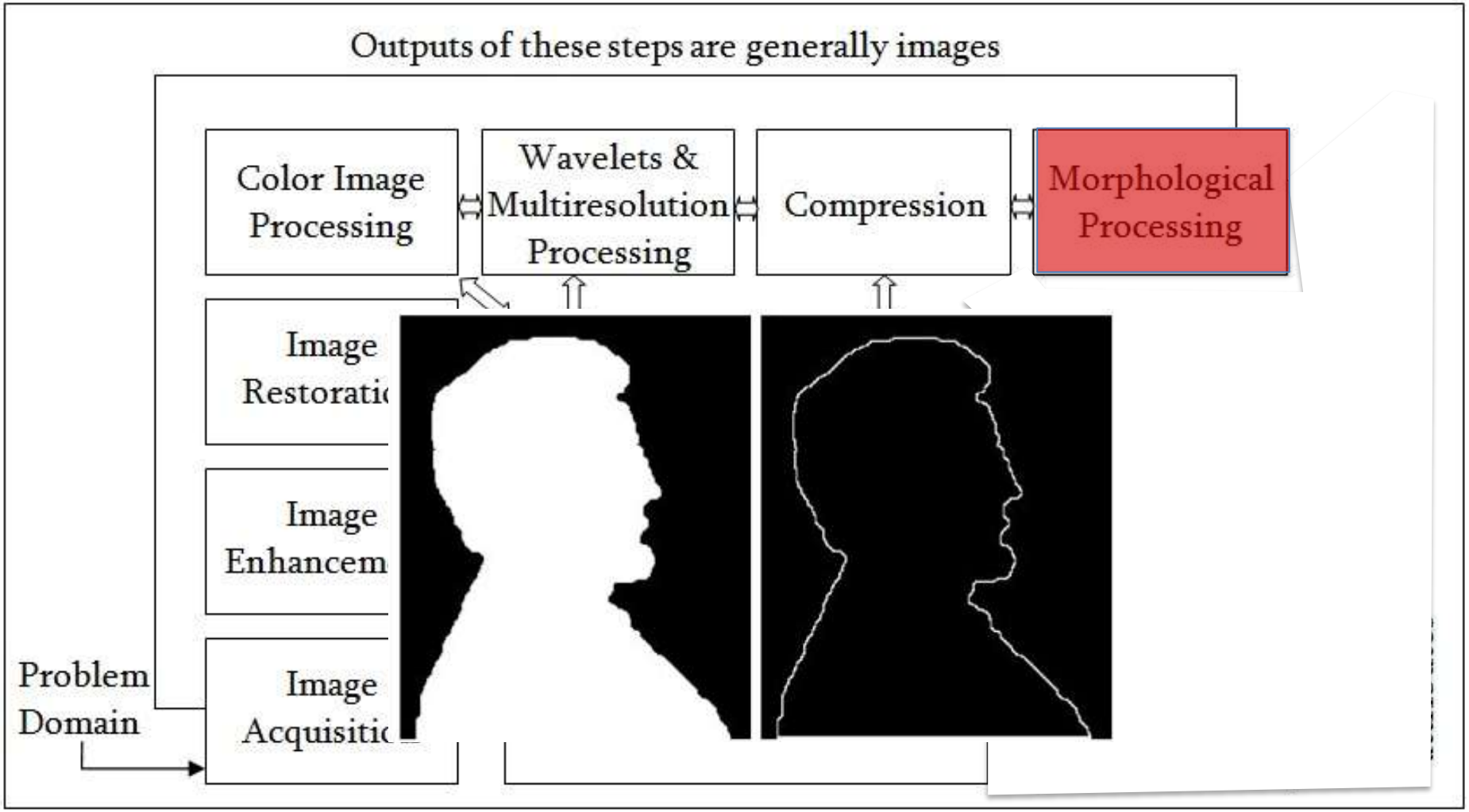
Image Acquisition

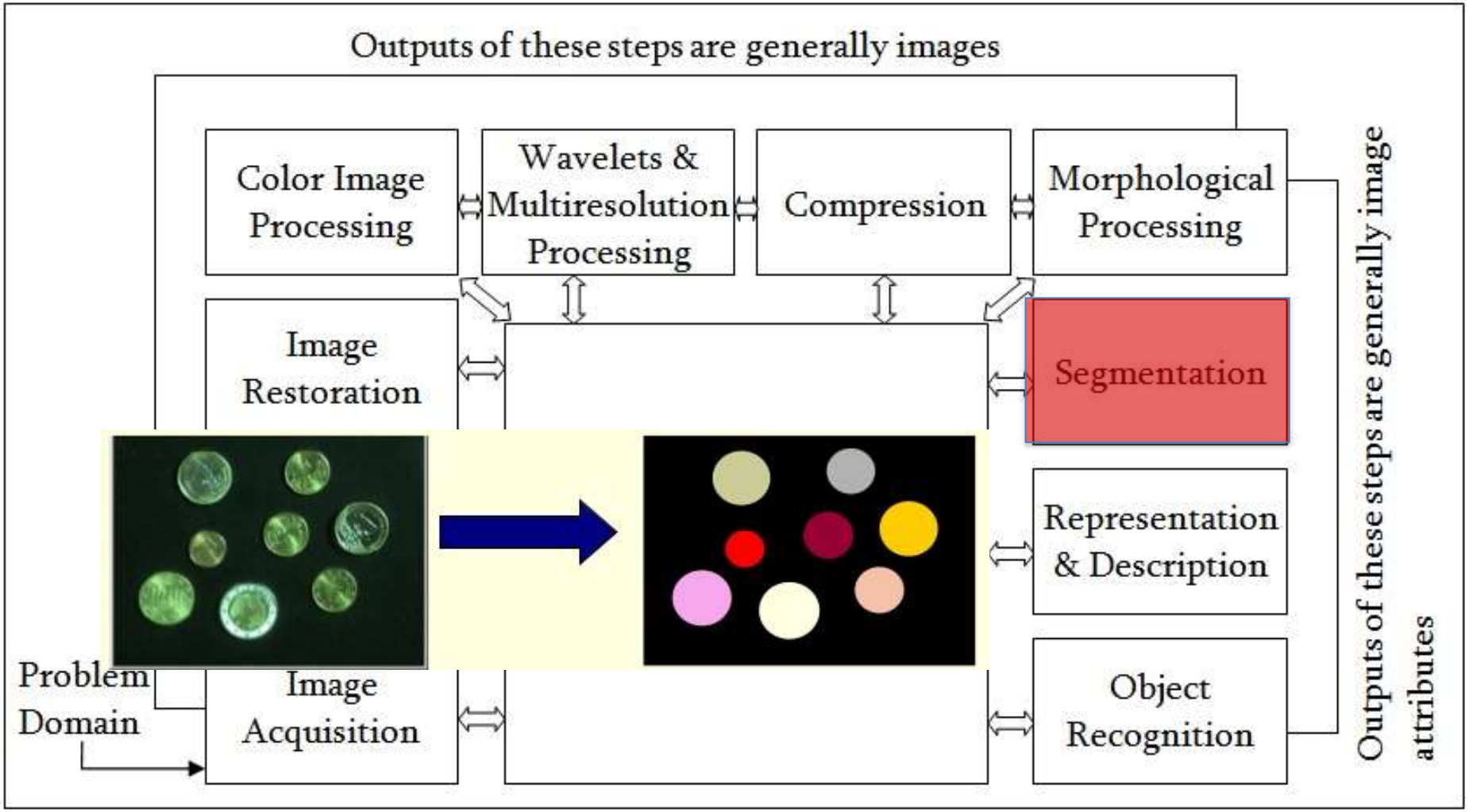
Problem Domain

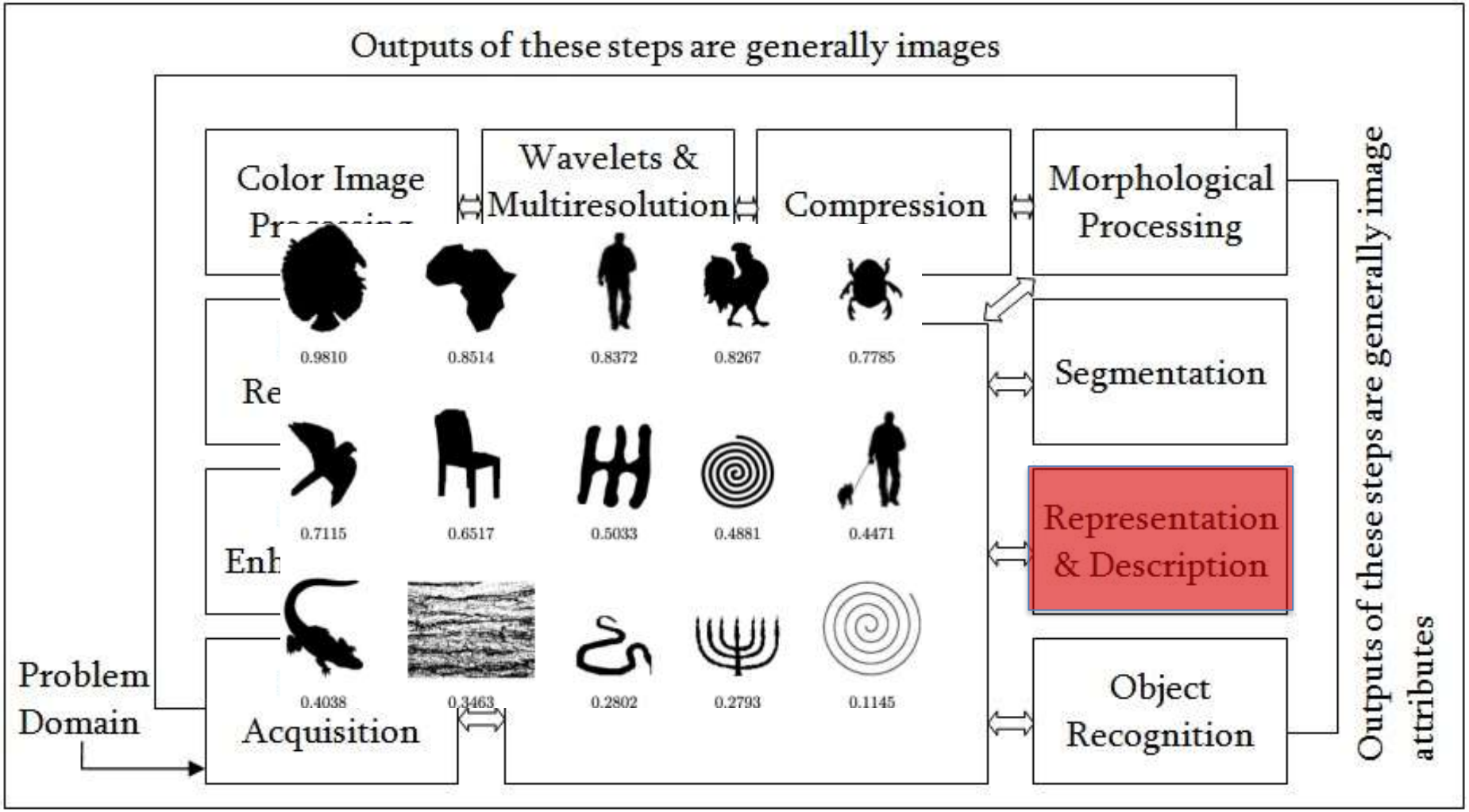
Size-270 KB

Size-22 KB

"Without Compression a CD store only 200 Pictures or 8 Seconds Movie"


Outputs of these steps are generally images
Color Image Processing
Wavelets & Multiresolution Processing
Compression
Morphological Processing
Image Restoratio
Image Enhancem
Image Acquisitio
Problem Domain


Outputs of these steps are generally images
Color Image Processing
Wavelets & Multiresolution Processing
Compression
Morphological Processing
Image Restoration
Segmentation
Representation & Description
Object Recognition
Image Acquisition
Problem Domain
Outputs of these steps are generally image attributes


Outputs of these steps are generally images
Color Image Processing
Wavelets & Multiresolution
Compression
Morphological Processing
Segmentation
Representation & Description
Object Recognition
Acquisition
Problem Domain
Outputs of these steps are generally image attributes
0.9810
0.8514
0.8372
0.8267
0.7785
0.7115
0.6517
0.5033
0.4881
0.4471
0.4038
0.3463
0.2802
0.2793
0.1145

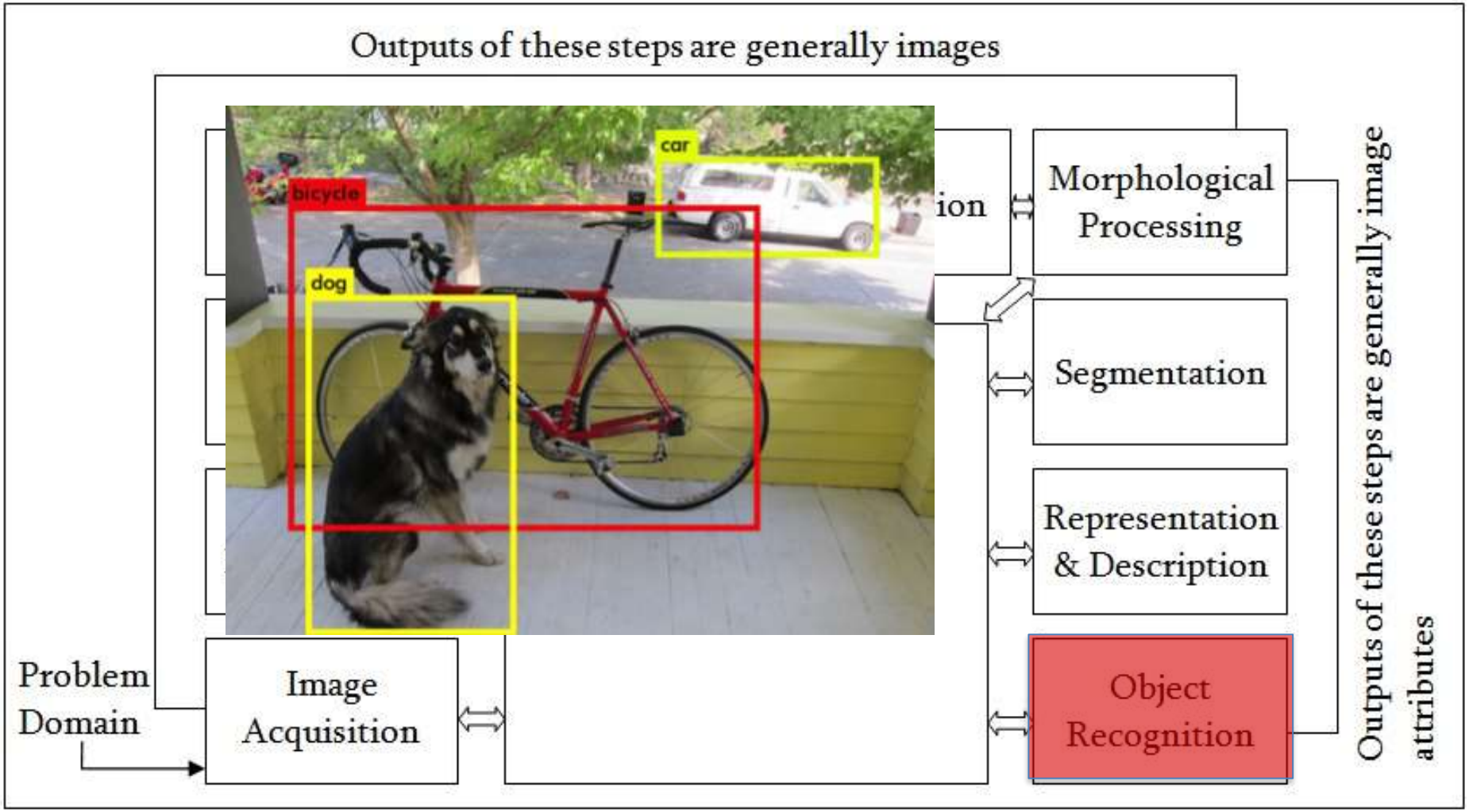
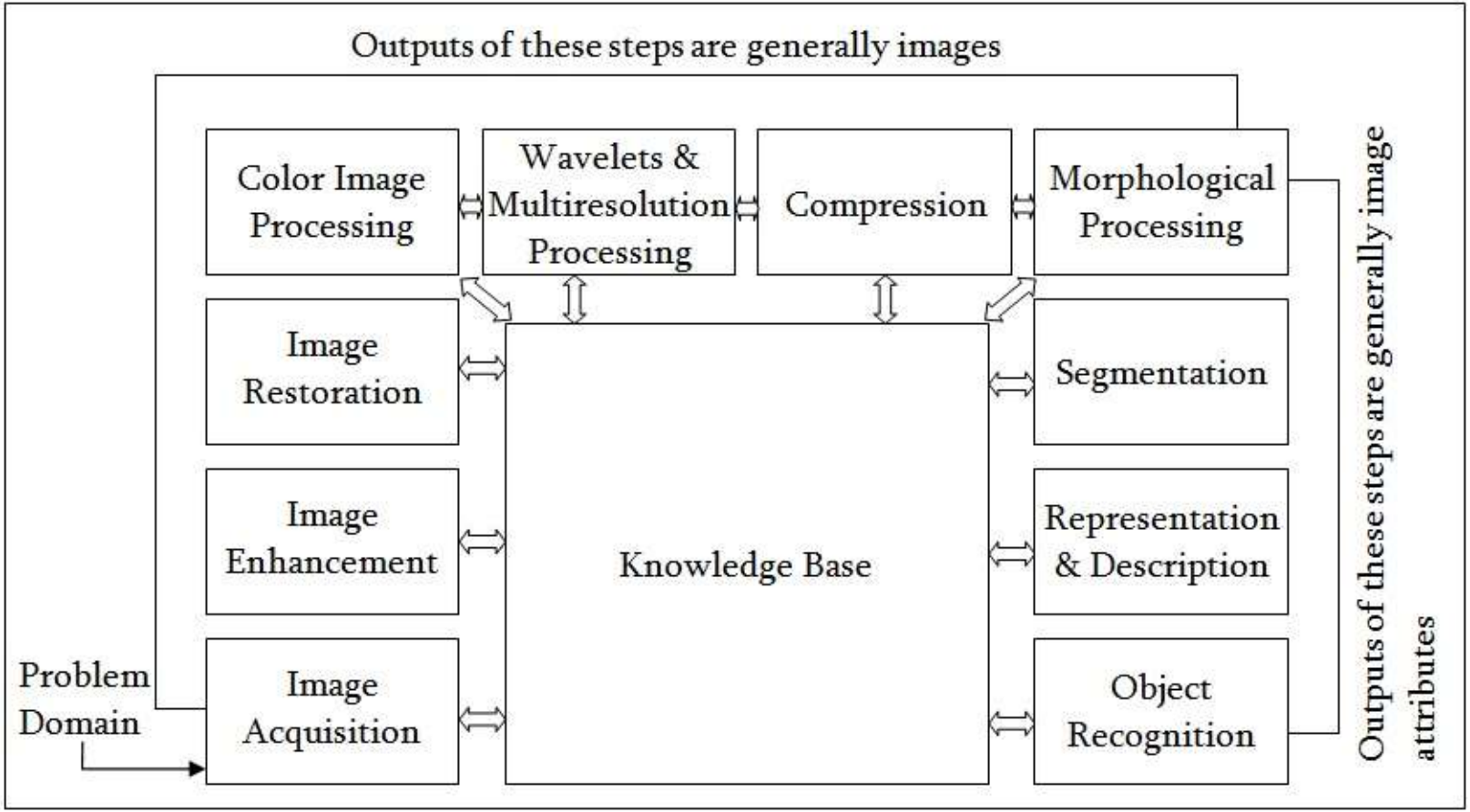

Outputs of these steps are generally images
car
bicycle
dog
ion
Morphological Processing
Segmentation
Representation & Description
Object Recognition
Outputs of these steps are generally image attributes
Problem Domain
Image Acquisition

Image Processing Tasks



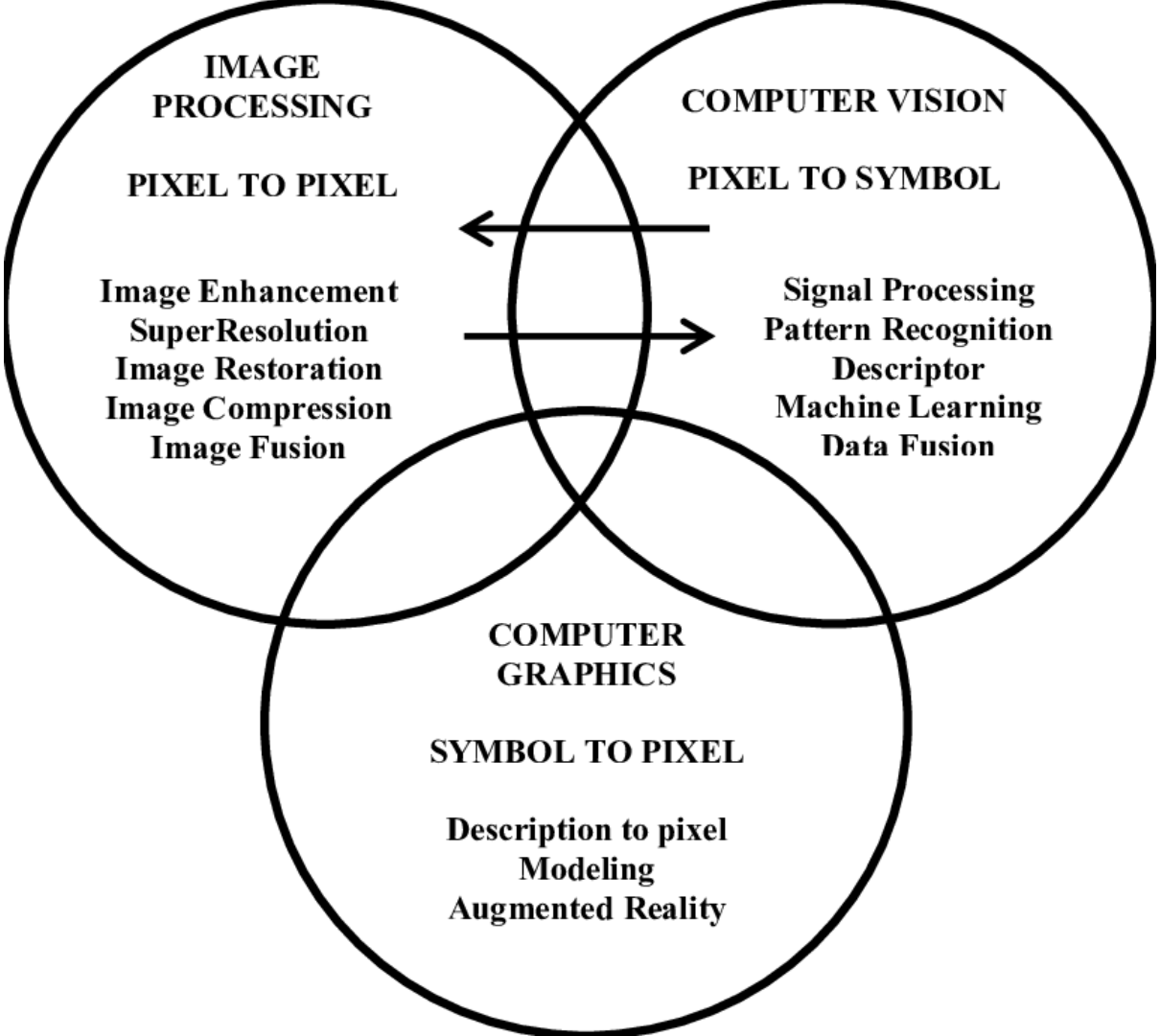


IMAGE PROCESSING
PIXEL TO PIXEL
Image Enhancement
SuperResolution
Image Restoration
Image Compression
Image Fusion
COMPUTER VISION
PIXEL TO SYMBOL
Signal Processing
Pattern Recognition
Descriptor
Machine Learning
Data Fusion
COMPUTER GRAPHICS
SYMBOL TO PIXEL
Description to pixel
Modeling
Augmented Reality

Computational Photography

Computer Graphics

+ easy to create new worlds
+ easy to manipulate objects/viewpoint
- Very hard to look realistic

Realism
Manipulation
Ease of capture

Photography

+ instantly realistic
+ easy to aquire
- very hard to manipulate objects/viewpoint

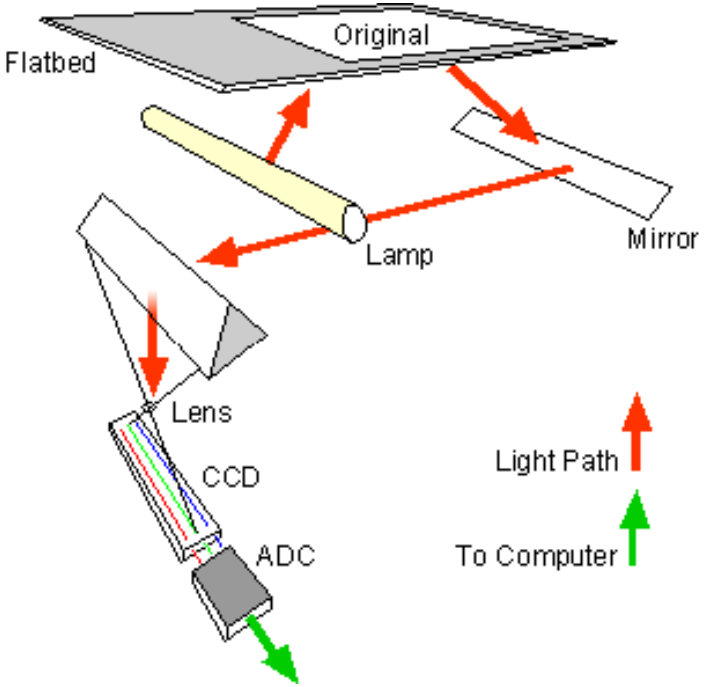
http://www.cs.toronto.edu/~guerzhoy/320/lec/Introduction.pdf

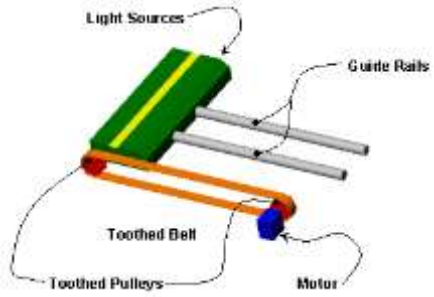
- **Digital Image Acquisition**
- Image Sampling and Quantization

How images are acquired

How images end up in digital form

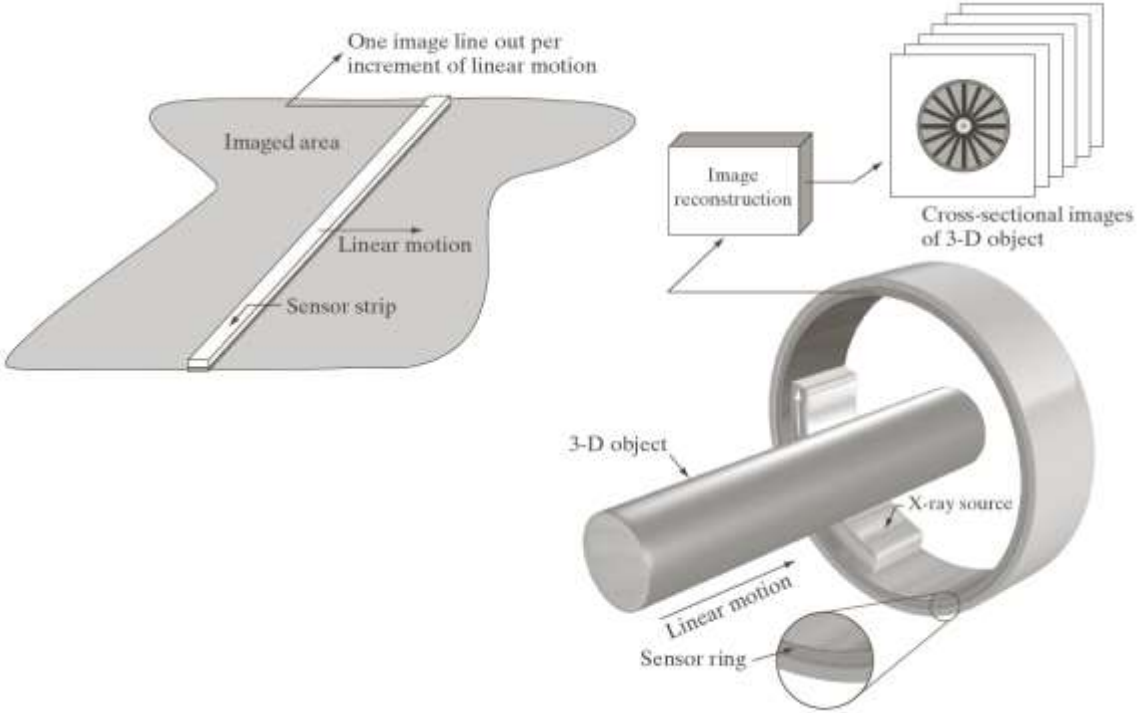
Image Sensing and Acquisition





How does a scanner work: https://www.youtube.com/watch?v=OpBDTjw9yho

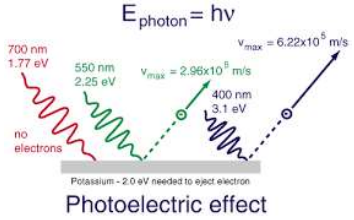
Image Sensing and Acquisition

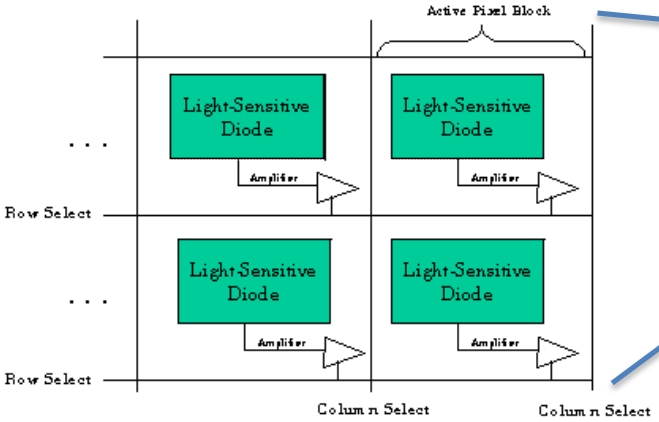


a b

FIGURE 2.14 (a) Image acquisition using a linear sensor strip. (b) Image acquisition using a circular sensor strip.

CMOS photo-electric sensor





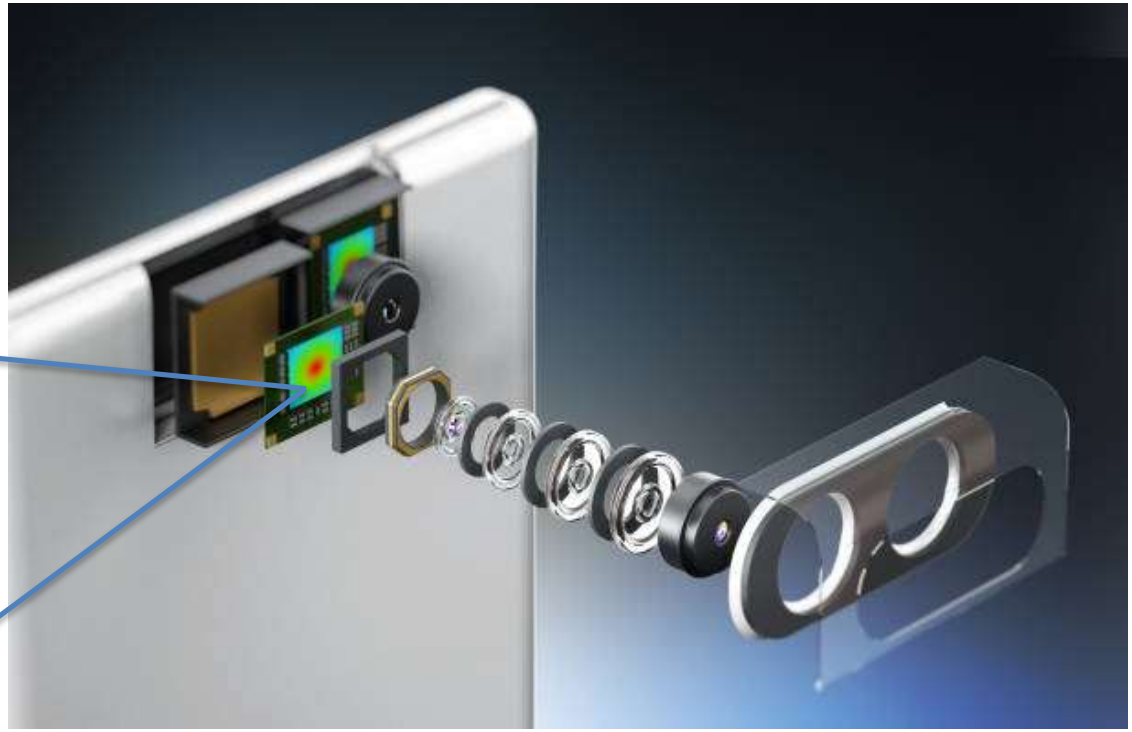
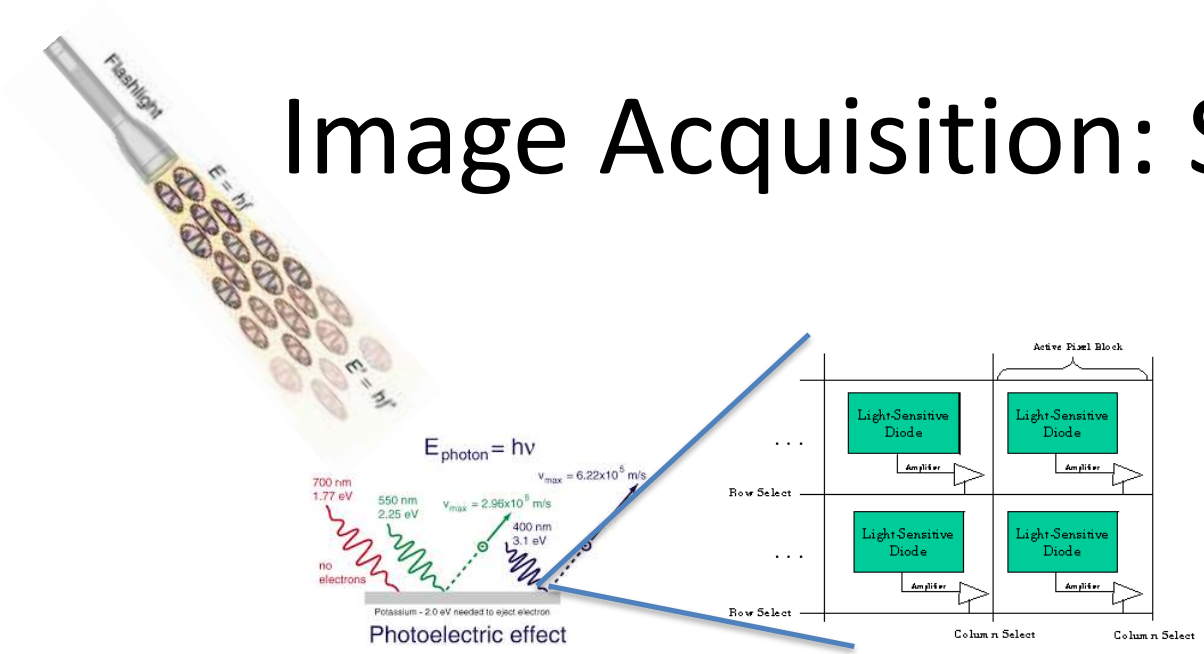
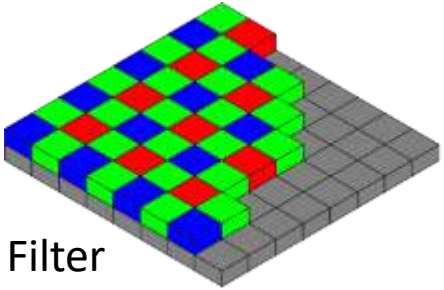


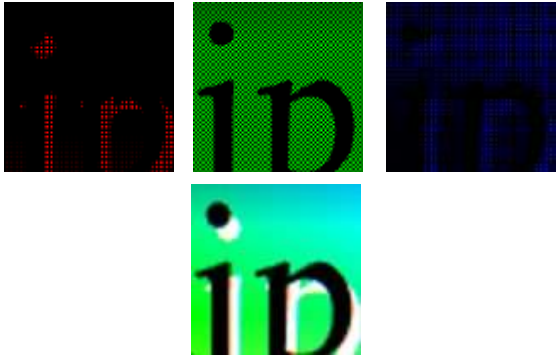
Image Acquisition: Summary





Bayer Filter

Demosaicing



- Digital Image Acquisition

How images are acquired

- **Image Sampling and Quantization**

How images end up in digital form

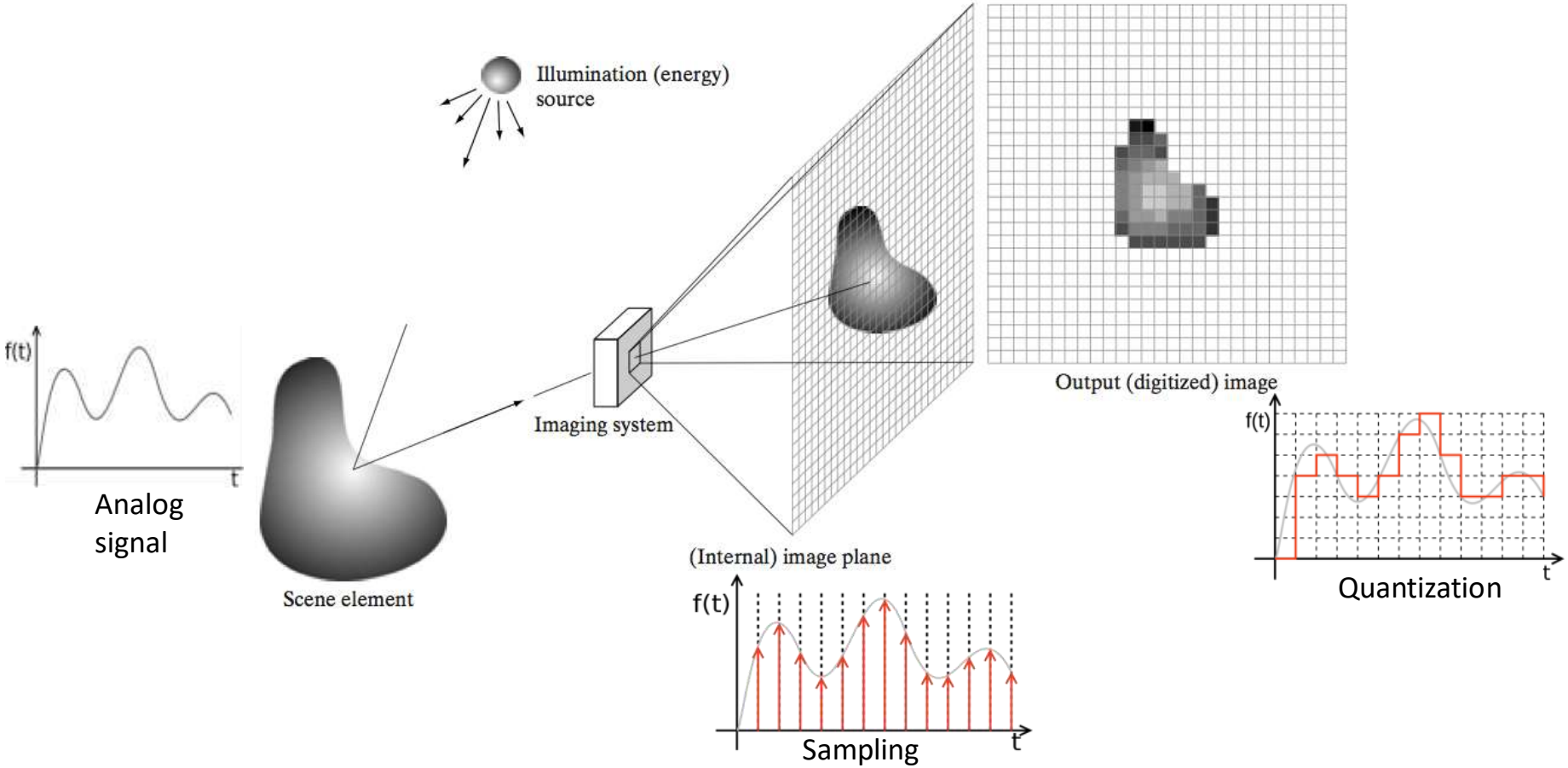


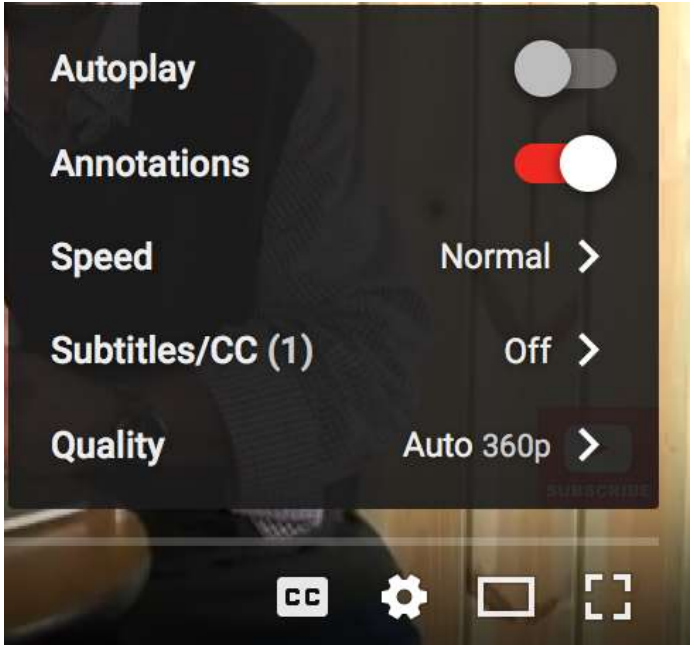
Image courtesy: Gonzalez and Woods

Cross-section of typical smartphone camera

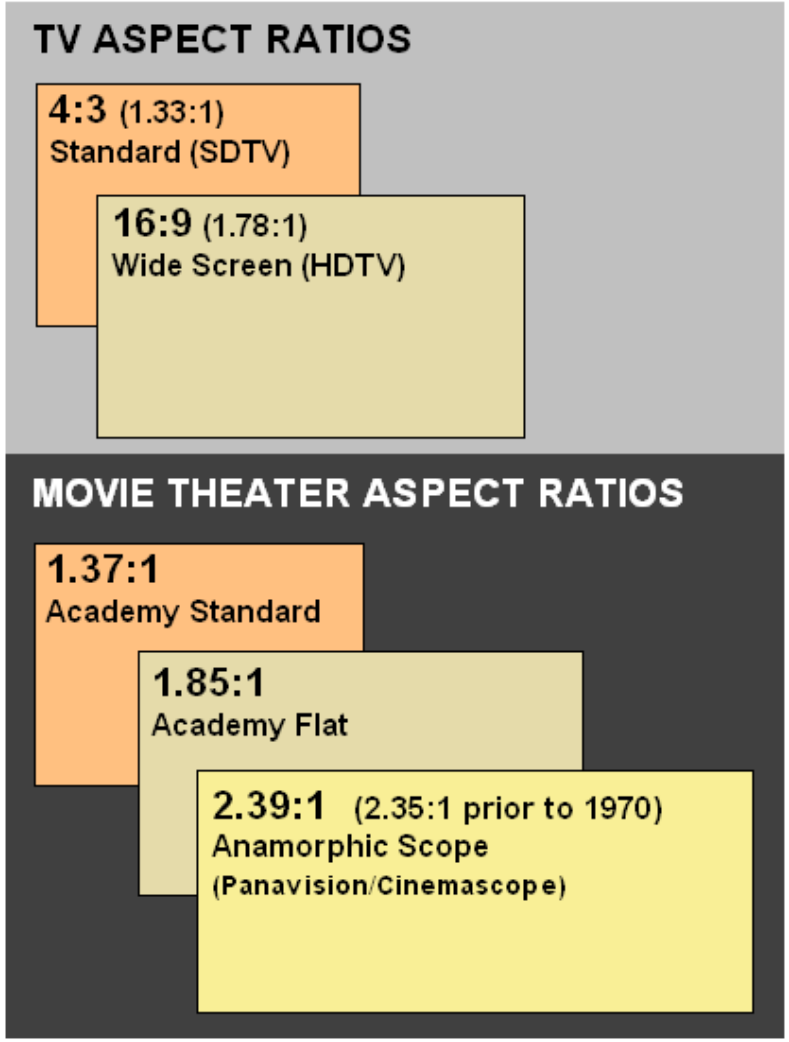
SONY
make.believe
α
Size matters. Larger sensor provides better quality images.
Sony 35mm Full frame
Exmor
Sony APS-C
Exmor
Micro Four Thirds
CX Format
Compact Camera
(1/2.3 format)
30 x
13 x
7.9 x
4.1 x
4 x
Approximate Size Ratio
PHOTOGRAPHY BAY

Full HD
1920 x 1080
4K
3840 x 2160
5K
5120 x 2160
8K
7680 x 4320

- Video resolution example
  - Settings in YouTube



Aspect Ratios



Key Terms

Standard Video Qualities: Standard combinations of aspect ratio and video resolution

Video Quality	4:3 Aspect Ratio Resolution	16:9 Aspect Ratio Resolution
360p	480 x 360	640 x 360
480p	640 x 480	854 x 480
720p	Not generally used	1280 x 720
1080p	Not generally used	1920 x 1280

Image acquisition process

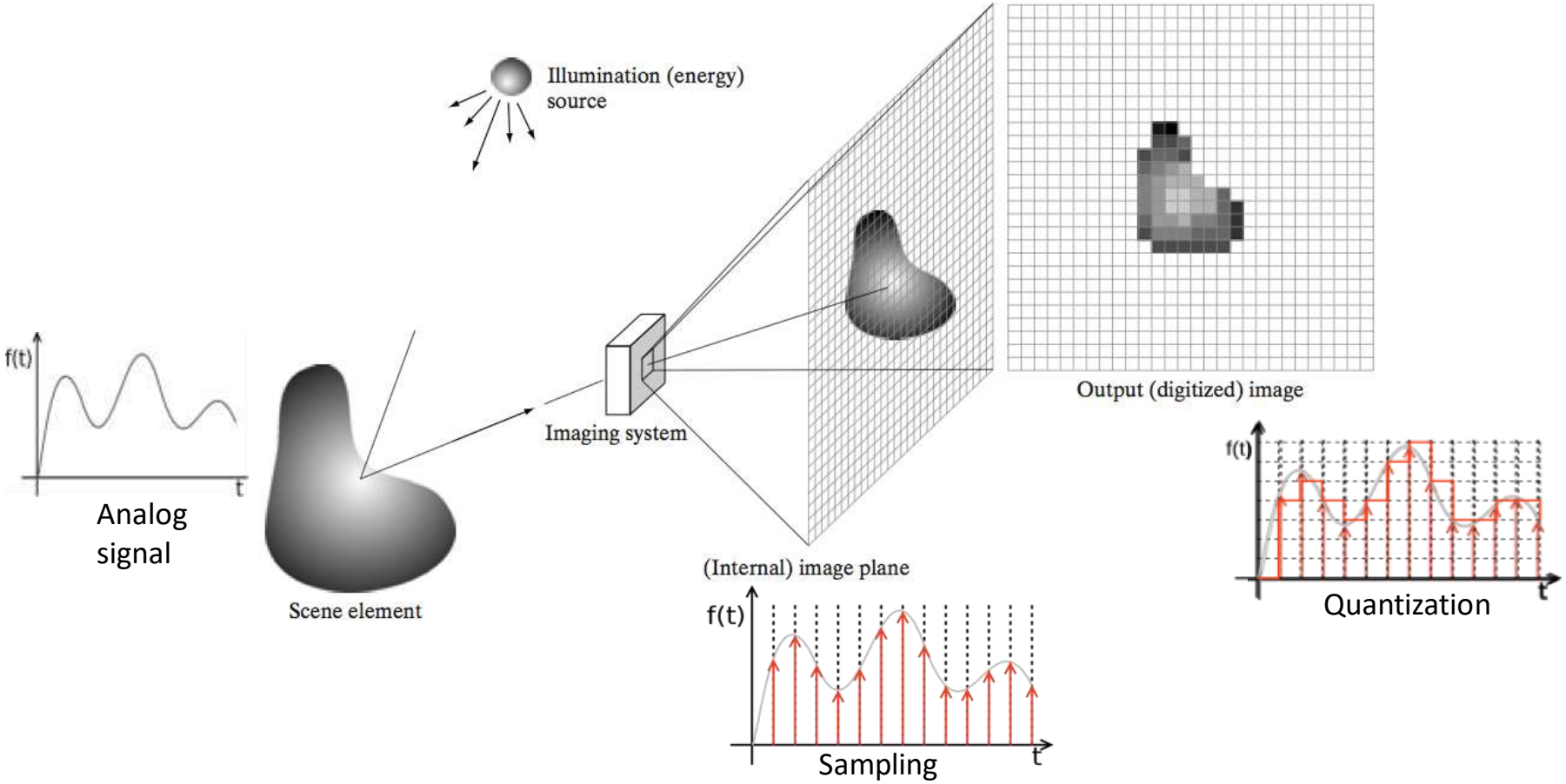
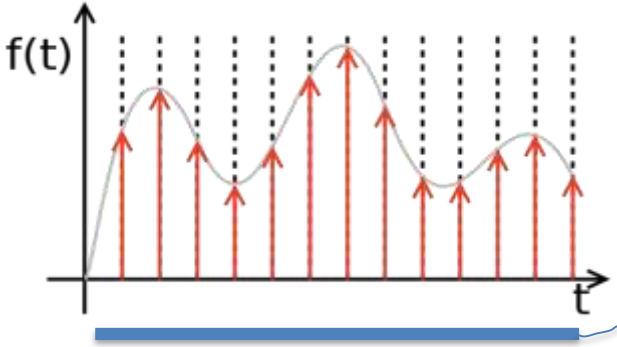


Image courtesy: Gonzalez and Woods

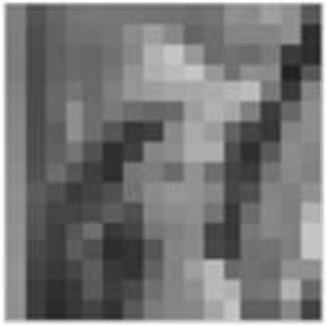
Summary



Sampling

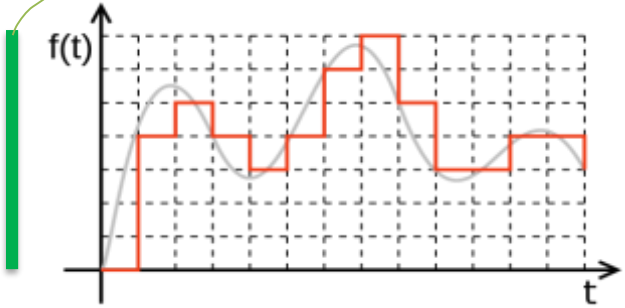
256 × 256

32 × 32



16 × 16

Quantization

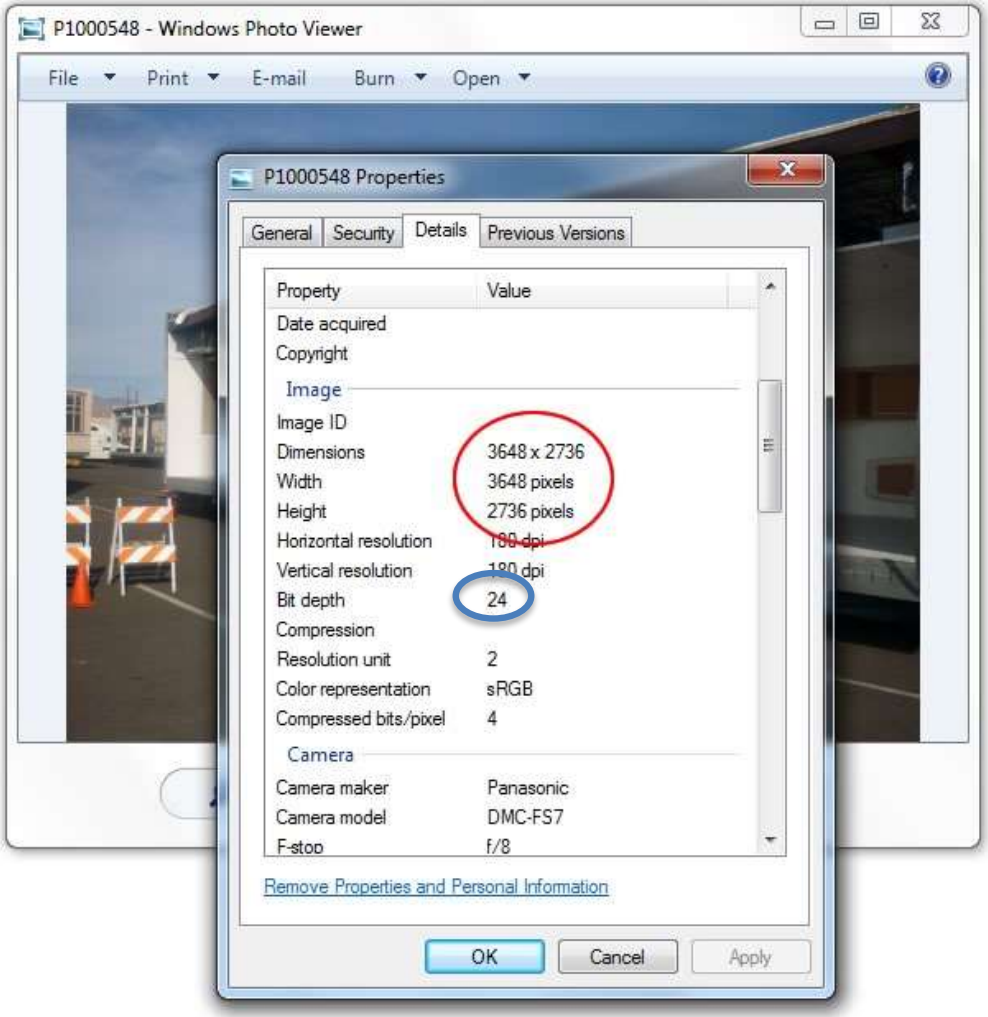


8 bits per pixel

4 bits per pixel

2 bits per pixel

1 bit per pixel


P1000548 - Windows Photo Viewer
File
Print
E-mail
Burn
Open
P1000548 Properties
General
Security
Details
Previous Versions
Property
Value
Date acquired
Copyright
Image
Image ID
Dimensions
3648 x 2736
Width
3648 pixels
Height
2736 pixels
Horizontal resolution
180 dpi
Vertical resolution
180 dpi
Bit depth
24
Compression
Resolution unit
2
Color representation
sRGB
Compressed bits/pixel
4
Camera
Camera maker
Panasonic
Camera model
DMC-FS7
F-stop
f/8
Remove Properties and Personal Information
OK
Cancel
Apply

Quantization

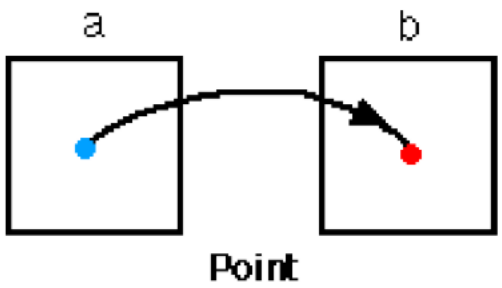
- In Hardware (# of voltage levels, # of bits)
- In Software (raw → JPEG)

Image Processing – Two Paradigms

- Directly manipulating pixels in spatial domain
- Manipulating in transform domain

Spatial Domain Processing

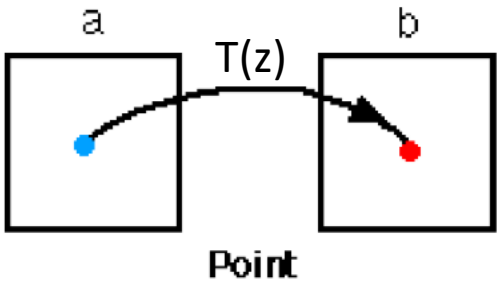
- Manipulating Pixels Directly in Spatial Domain
- 3 approaches
- 1. Point to Point



Intensity Transforms – Point to Point

- z = *a*(x,y)

- z' = b(x,y) = T(z) = T(*a*(x,y))



Intensity levels r: [0, L-1]

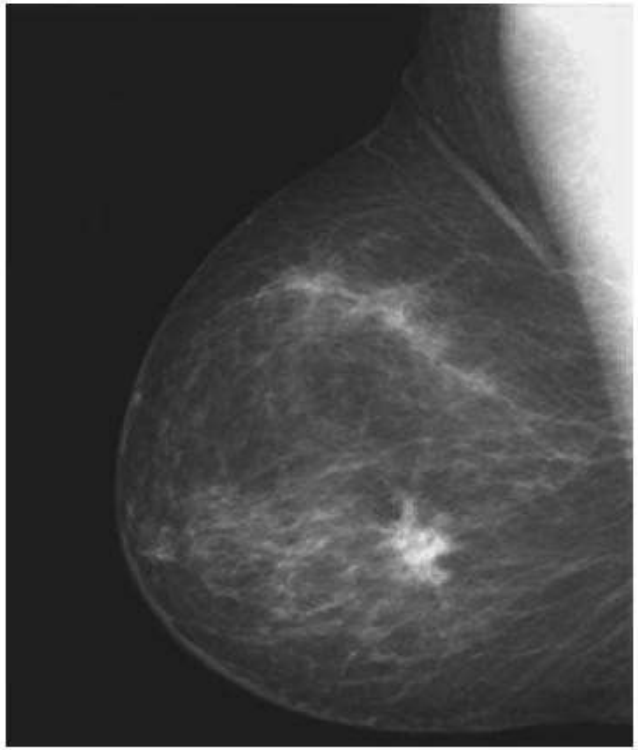
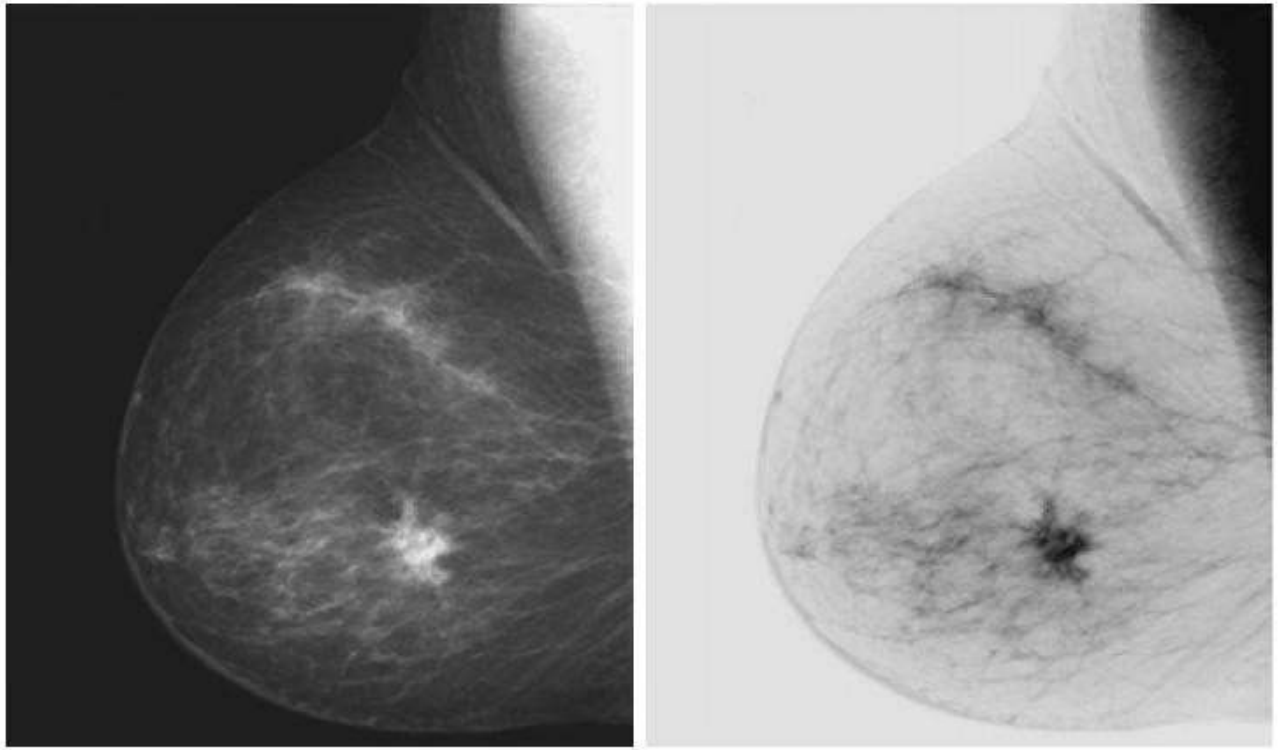


Image Negatives

Intensity levels r: [0,L-1]

$$s = T(r) = L - 1 - r$$



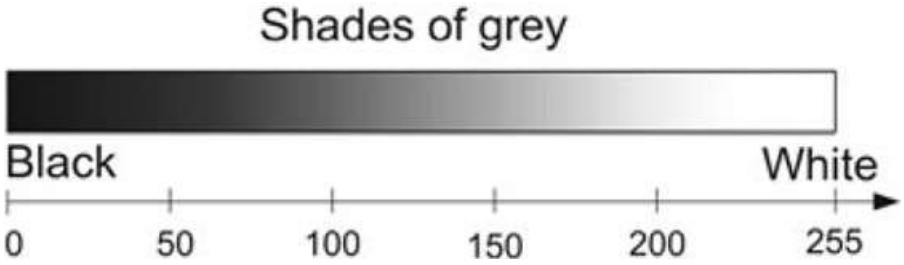
a b

FIGURE 3.4 (a) Original digital mammogram. (b) Negative image obtained using the negative transformation in Eq. (3.2-1). (Courtesy of G.E. Medical Systems.)

Intensity Transforms

- $T(z) = z + K$
- $T(z) = z - K$

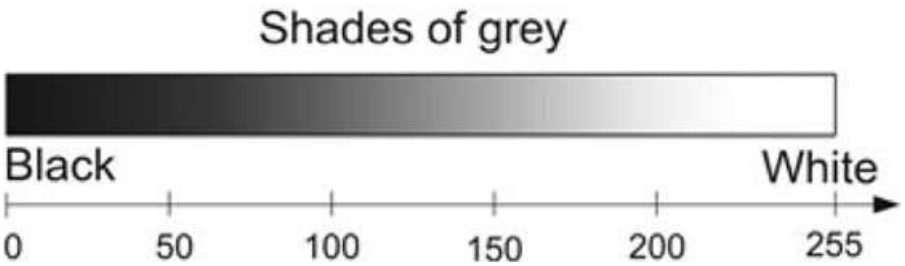
Demo:
https://colab.research.google.com/drive/11qIL0VKleZnONtPuxAryAf9WkUC7kEMI#scrollTo=WkBKnKz7aS6O&line=1&uniqifier=1



Data visualization: Map to display range

Normalize to range :

$$J = \texttt{round}\left(255 * \frac{I - min(I)}{max(I) - min(I)}\right)$$



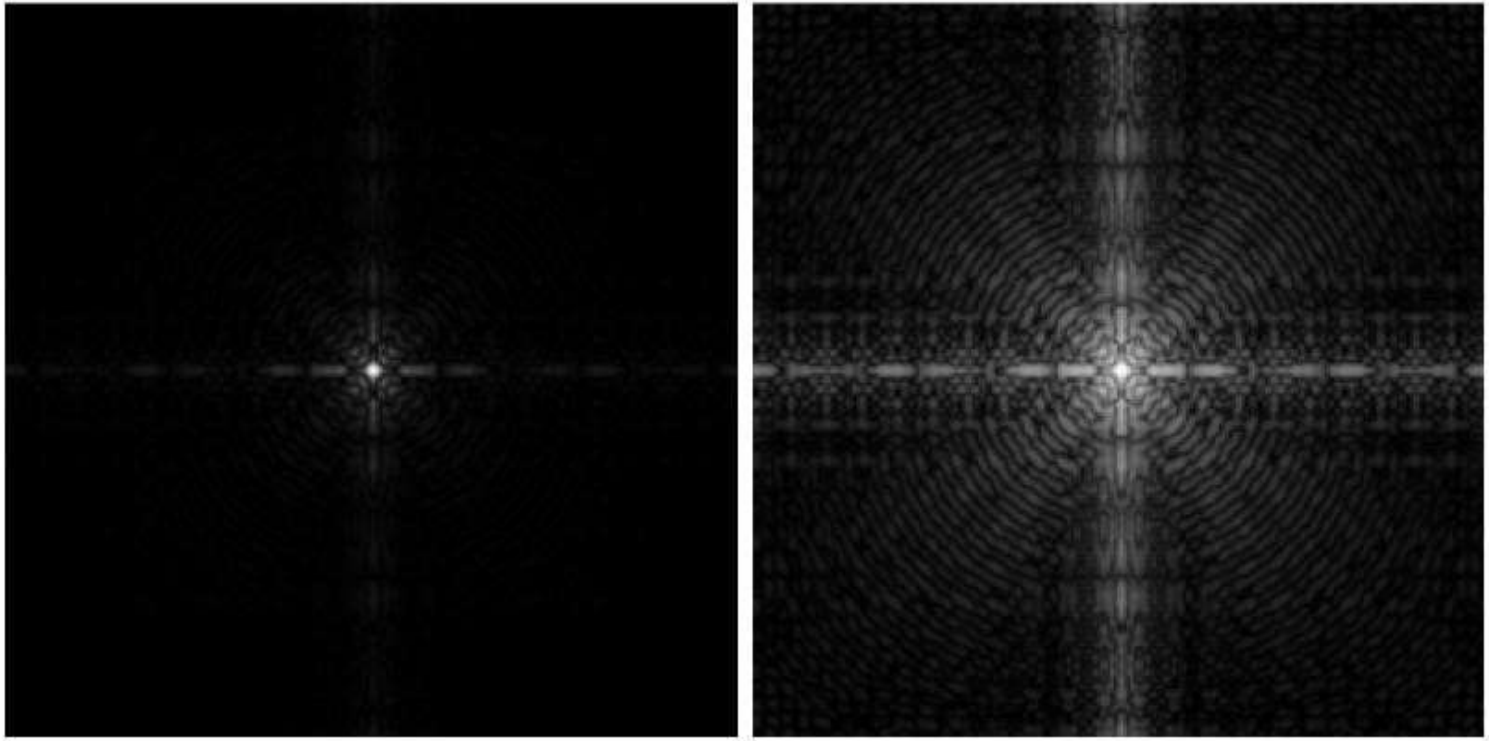
Storage v/s Display

- 8-bit image : [0,255]
- 4-bit image : [0,15]
- Demo: https://colab.research.google.com/drive/11qIL0VKleZnONtPuxAryAf9WkUC7kEMI#scrollTo=WkBKnKz7aS6O&line=1&uniqifier=1

Log Transformations

a b

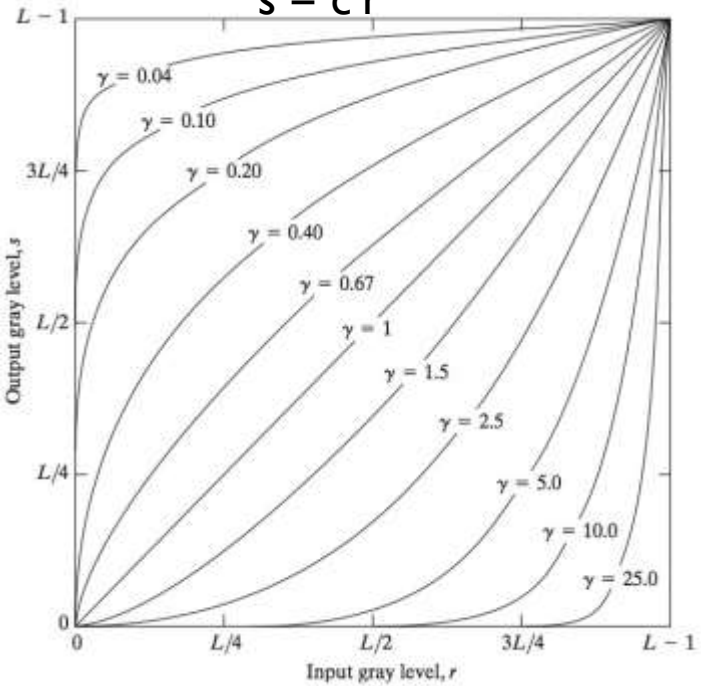
FIGURE 3.5
(a) Fourier spectrum.
(b) Result of applying the log transformation given in Eq. (3.2-2) with $c = 1$.

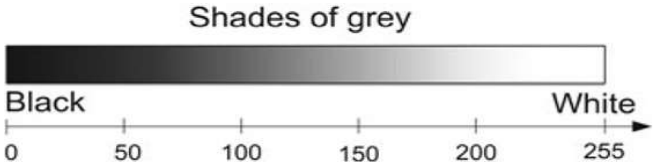


$$s = T(r) = c\ log(1 + r)$$

Power-Law Transformations

$s = c\,r^{\gamma}$



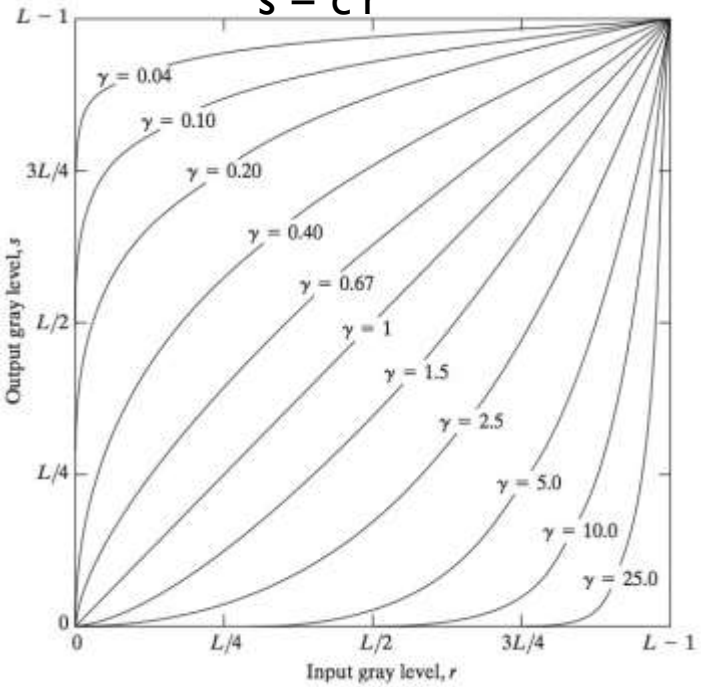


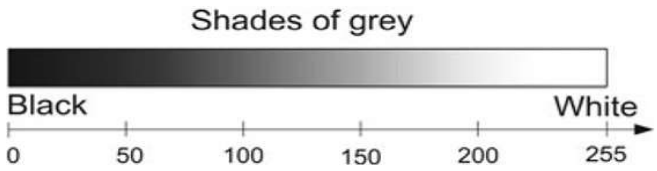
a b
c d

FIGURE 3.9 (a) Aerial image. (b)–(d) Results of applying the transformation in Eq. (3.2-3) with $c = 1$ and $\gamma = 3.0, 4.0$, and 5.0, respectively. (Original image for this example courtesy of NASA.)

Power-Law Transformations

$s = c\, r^{\gamma}$



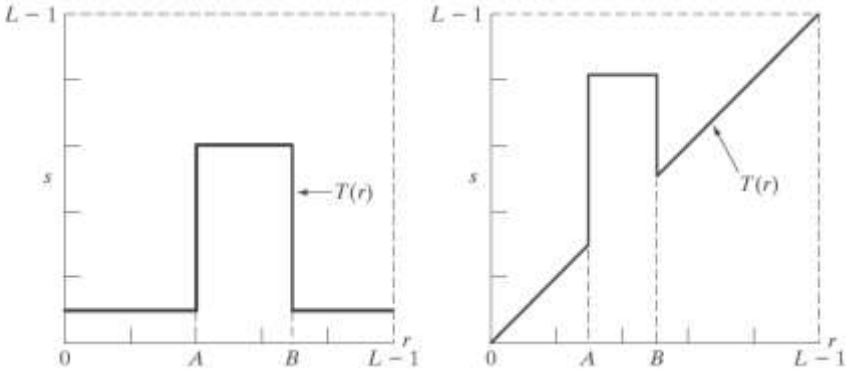


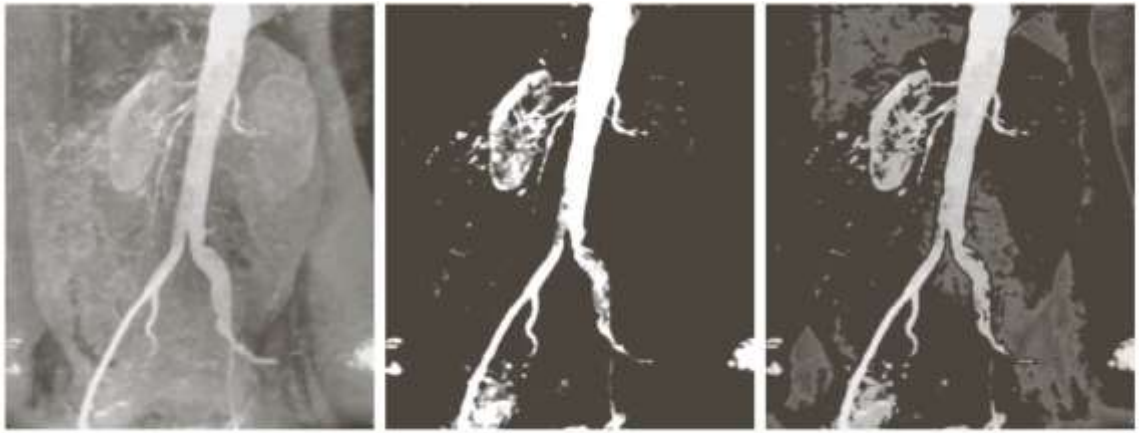
Demo:
https://colab.research.google.com/drive/11qIL0VKleZnONtPuxAryAf9WkUC7kEMI#scrollTo=aU5WQaqOpSCr&line=12&uniqifier=1

Intensity Slicing

a b

FIGURE 3.11 (a) This transformation highlights intensity range $[A, B]$ and reduces all other intensities to a lower level. (b) This transformation highlights range $[A, B]$ and preserves all other intensity levels.





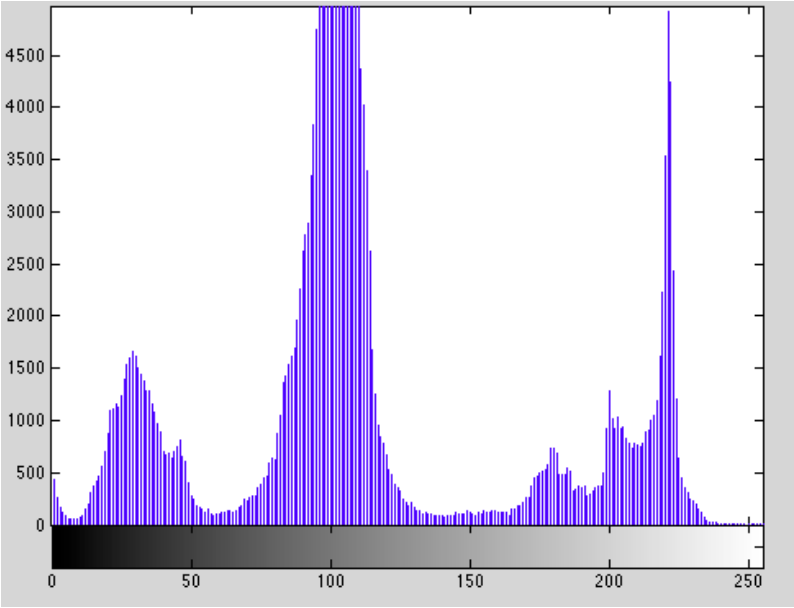
a b c

Histogram: An image representation + visualization

$$h_r(i) = n_i$$

i → intensity value, range [0,L-1]

n_i → number of pixels with intensity i



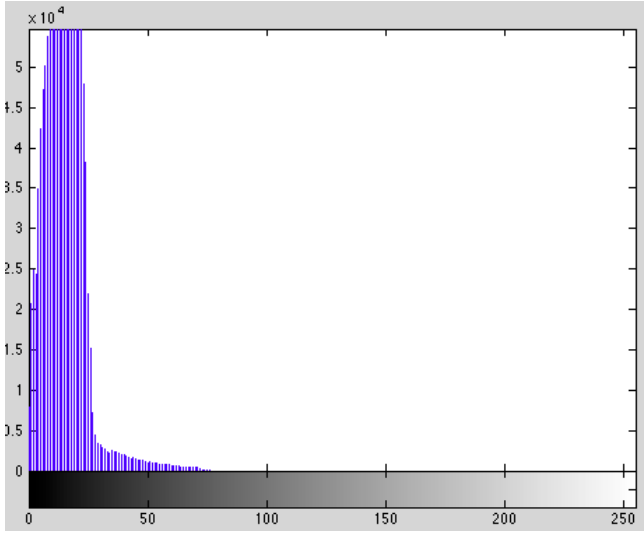
Histograms

- What can we infer from histograms?

Histogram viewing standard in most DSLR cameras

Histograms

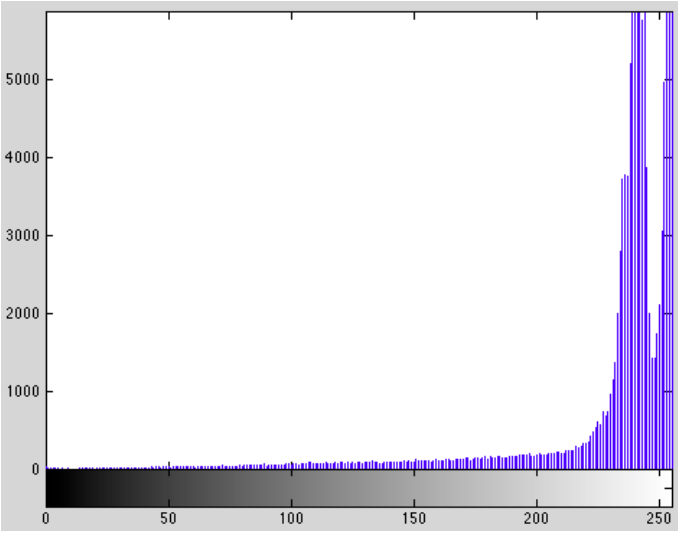
- Histograms and brightness



Under exposure

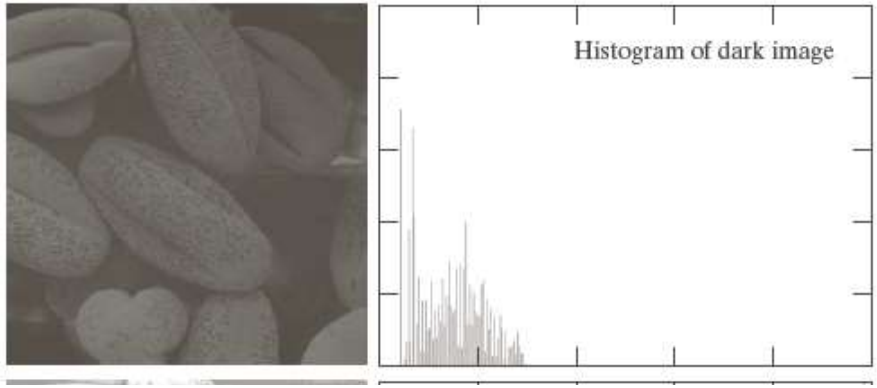
Histograms

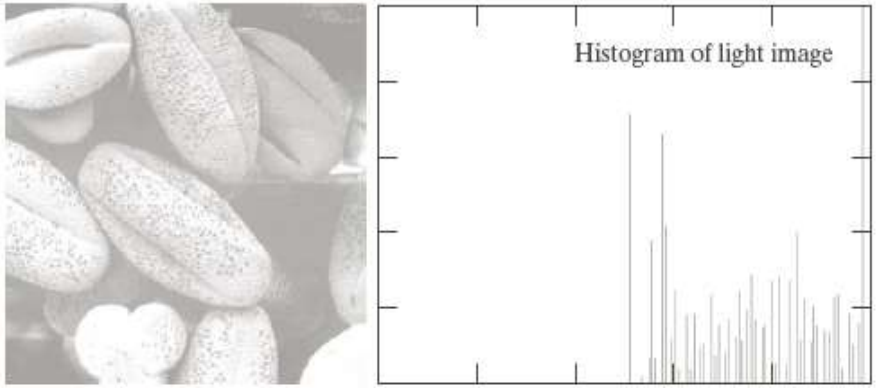
- Histograms and brightness



Over exposure

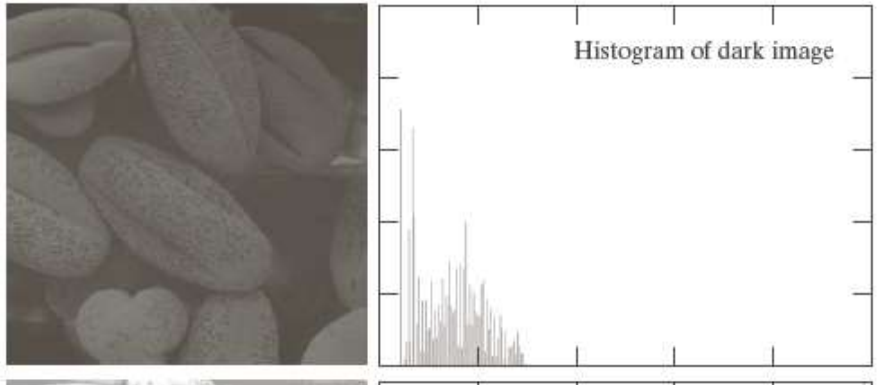
Histograms and Contrast

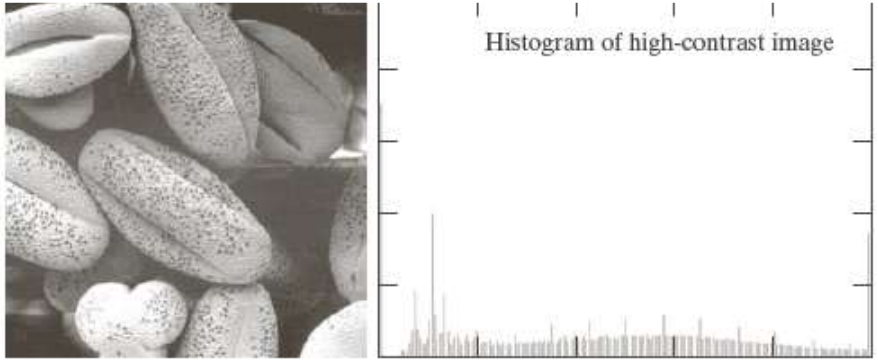


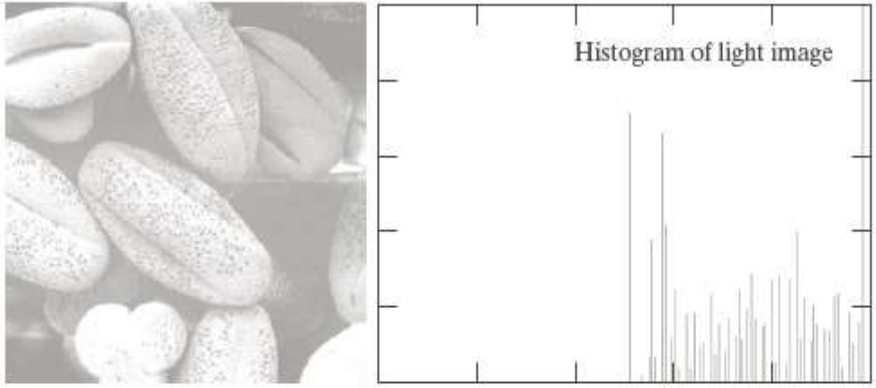


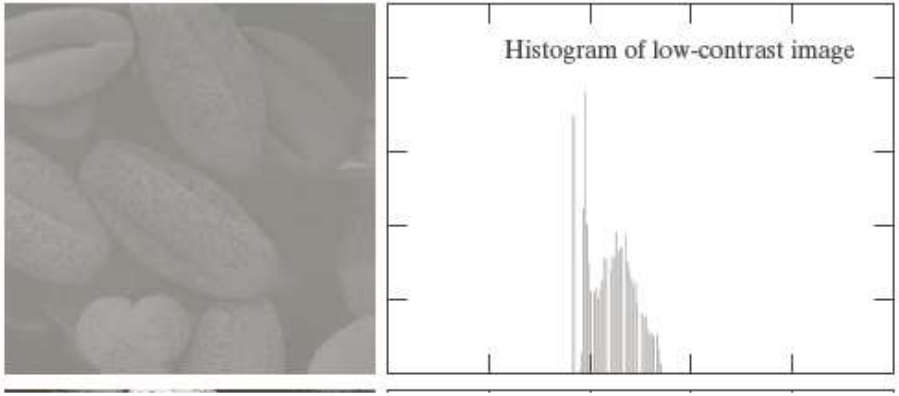
$$\frac{I_{max} - I_{min}}{I_{max} + I_{min}}$$

Histograms and Contrast

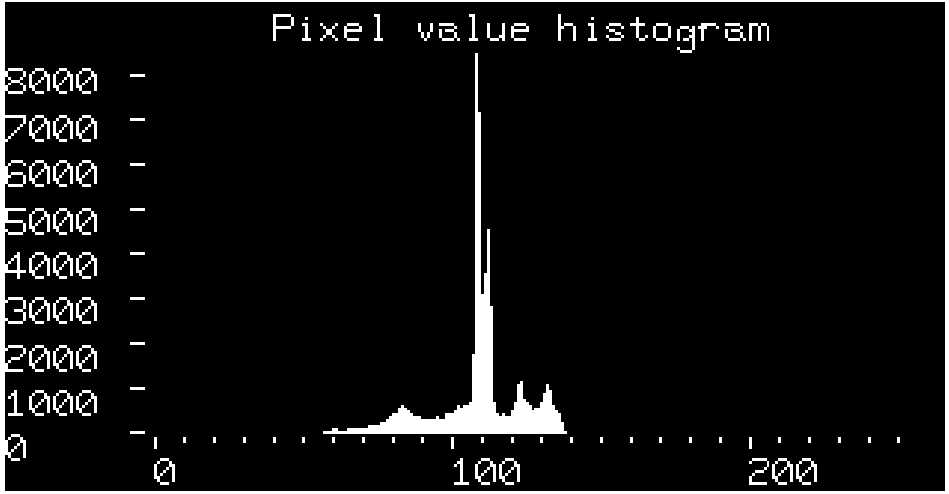








A low-contrast image and its histogram

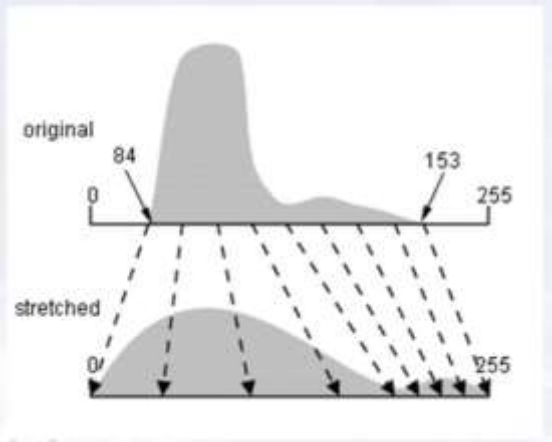


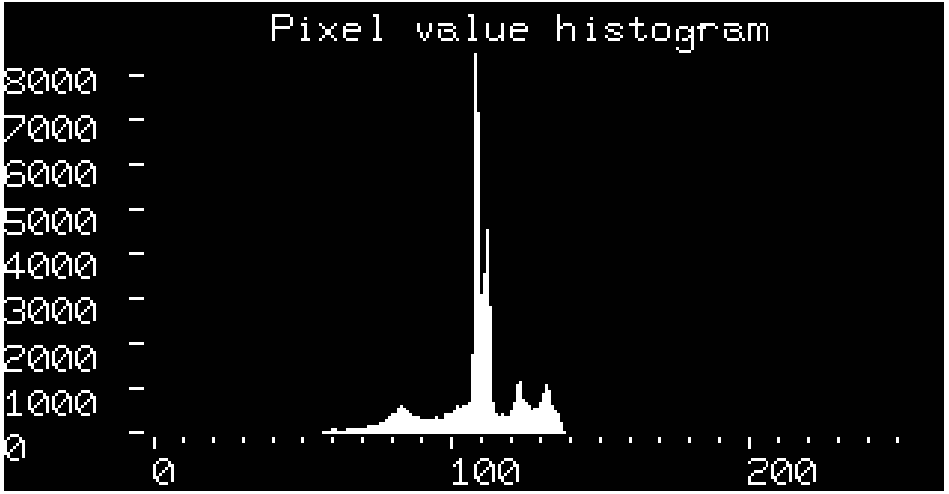
Contrast Stretching

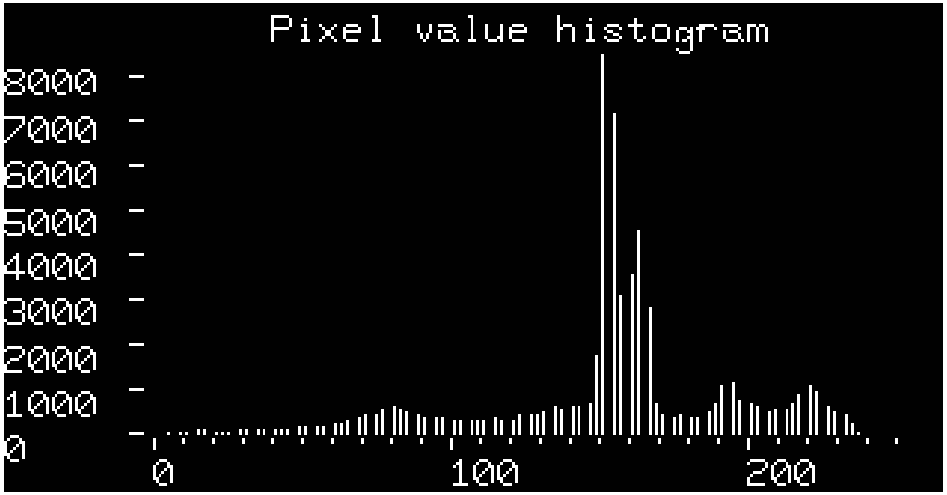
$$f_{\text{ac}}(a) = a_{\text{min}} + (a - a_{\text{low}}) \cdot \frac{a_{\text{max}} - a_{\text{min}}}{a_{\text{high}} - a_{\text{low}}}$$

If $a_{min} = 0$ and $a_{max} = 255$

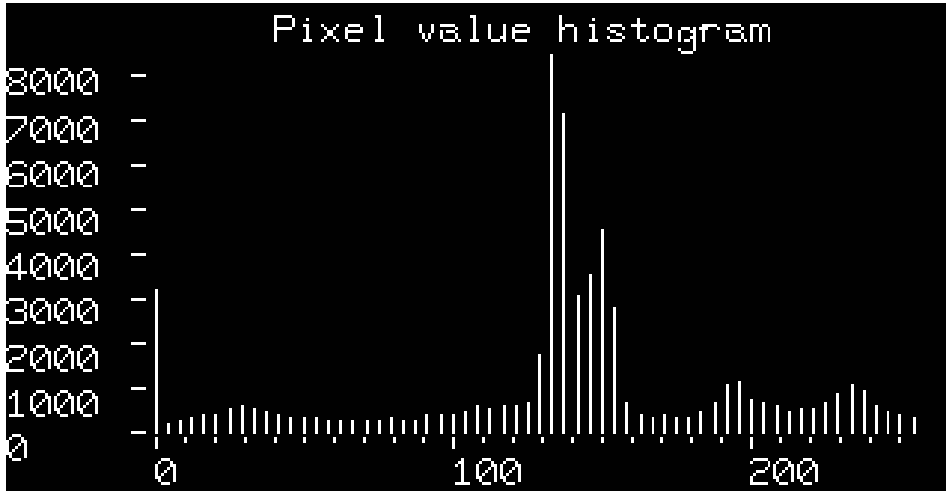
$$f_{\text{ac}}(a) = (a - a_{\text{low}}) \cdot \frac{255}{a_{\text{high}} - a_{\text{low}}}$$




Pixel value histogram
8000
7000
6000
5000
4000
3000
2000
1000
0
0
100
200

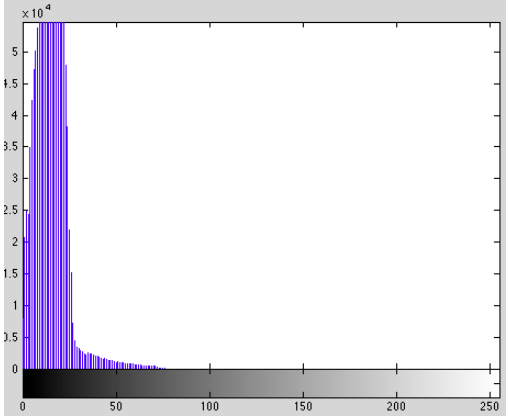

Pixel value histogram
8000
7000
6000
5000
4000
3000
2000
1000
0
0
100
200

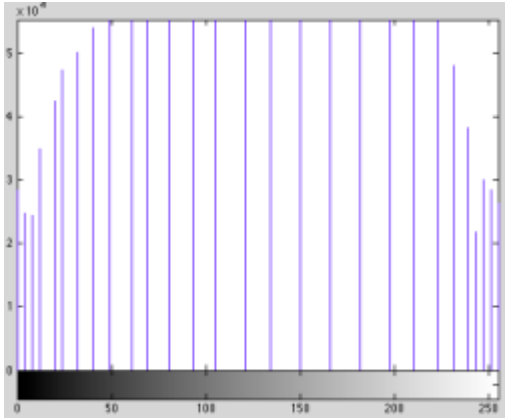
Are all intensities well represented ?



ALL INTENSITIES

Histogram Equalization

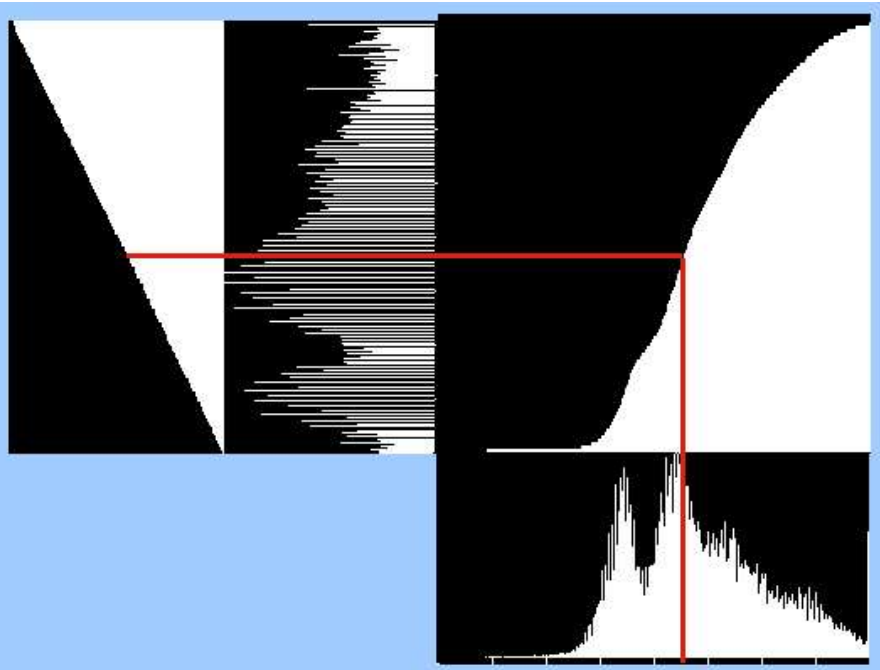




Histogram Equalization

Contrast Stretching

Histogram Equalization


Equalized histogram
Histogram of riginal image

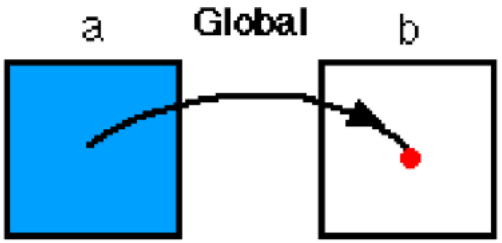
Histogram Equalization v/s Contrast Enhancement

Contrast Enhancement

Histogram equalization

Histogram Processing

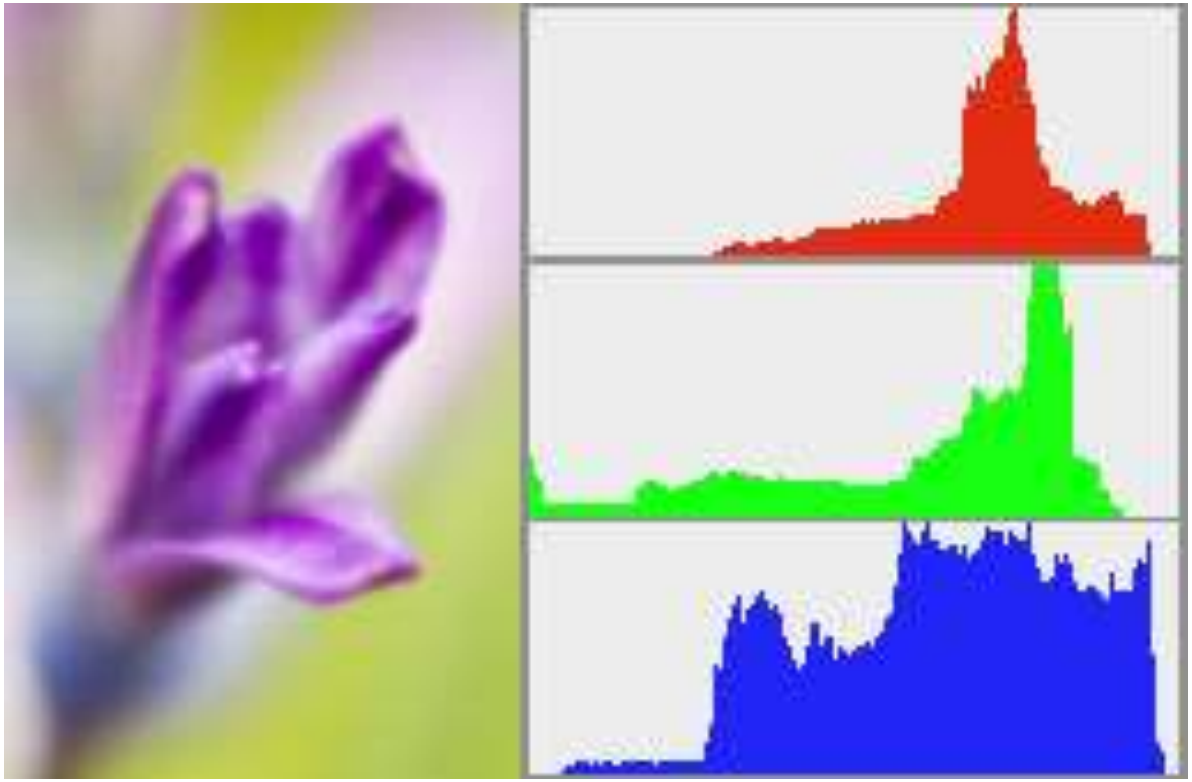
- Global to Point



Histogram : Discussion

- A visualization
- A useful statistical representation of image intensities
  - Not dependent on image size
- Drawbacks
  - No spatial information
  - Intensity-centric
  - Raw (unnormalized form): Image-size dependent
- Equalization:
  - An image ‘normalization’ approach
  - Improves global contrast, but can also boost noise

Histograms for RGB images



Histogram Specification / Matching [GW Section 3.3.2]

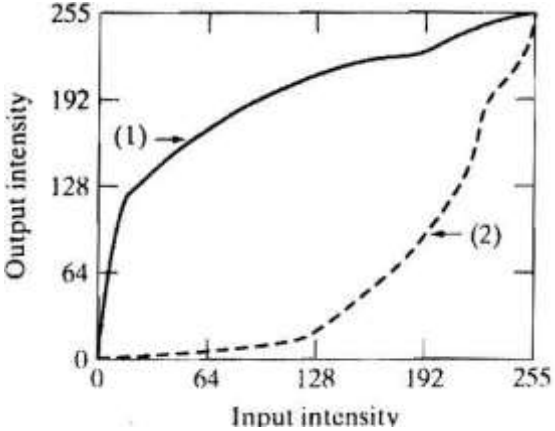
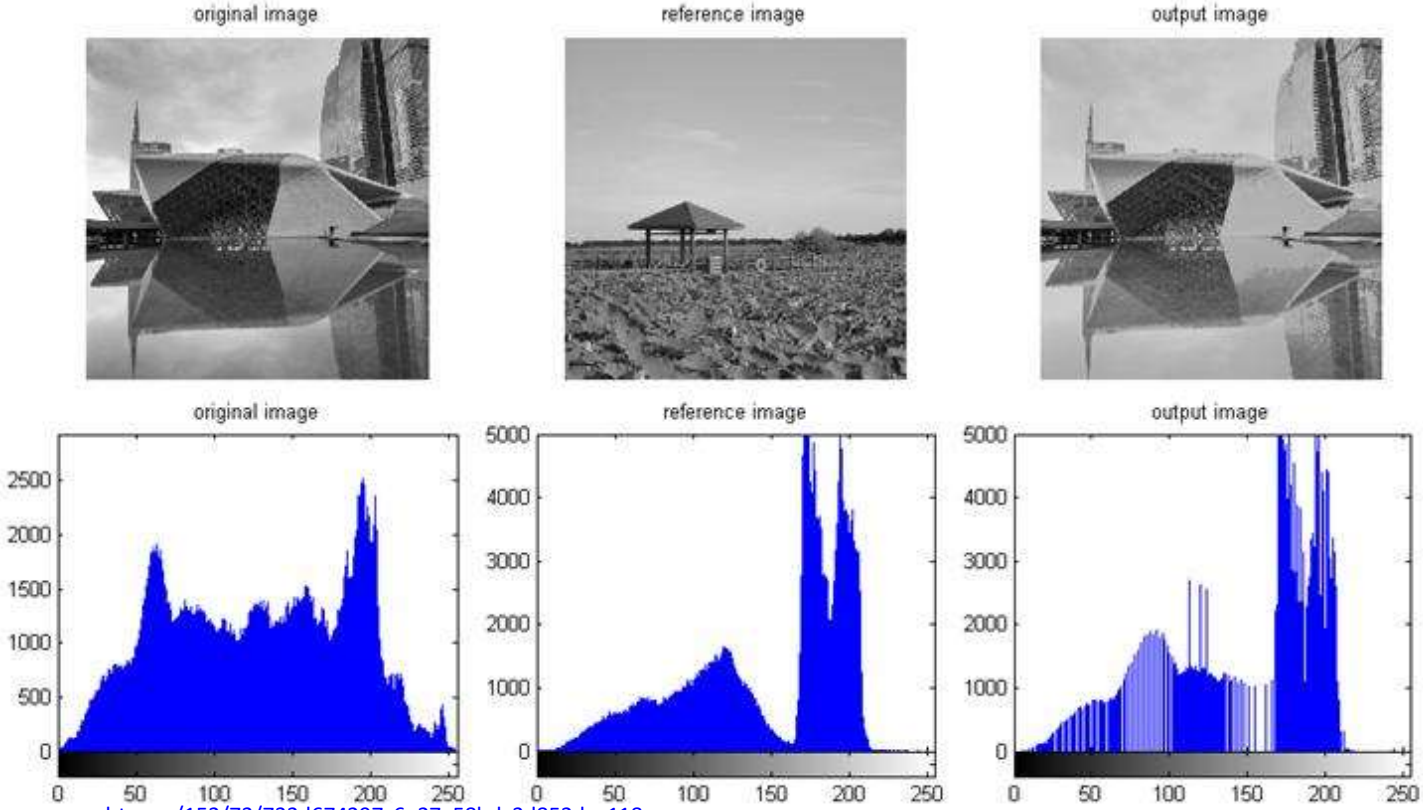


Image Courtesy: Gonzalez and Woods

Histogram specification (custom curve)

https://theiszm.wordpress.com/2010/07/06/5-image-enhancement-by-histogram-manipulation/

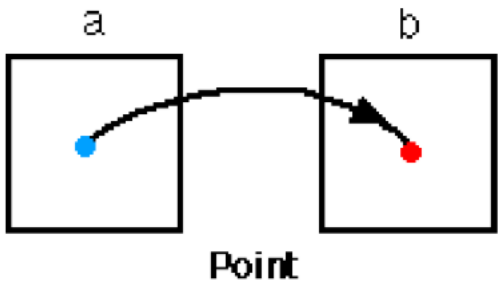
Histogram specification – style transfer (curve from a reference image)



https://images1.programmersought.com/152/73/733d674297c6e27a58bdc3d852dca118.png

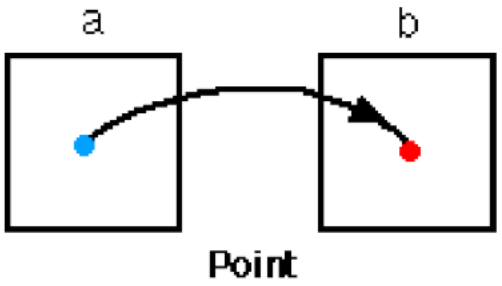
Summary

- Manipulating Pixels Directly in Spatial Domain
- 3 approaches
- 1. Point to Point



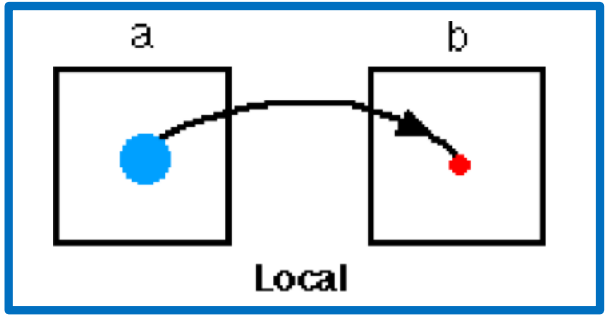
- Linear Intensity Transforms
  - E.g. Negative
- Non-linear Transforms
  - E.g. Logarithm
- Histogram

- Point to Point

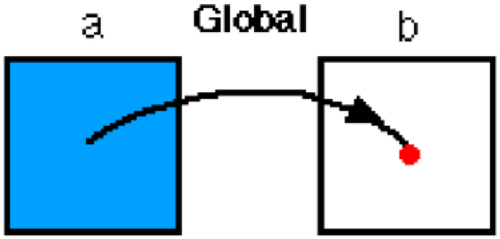


Intensity Transforms

- **Neighborhood to Point**

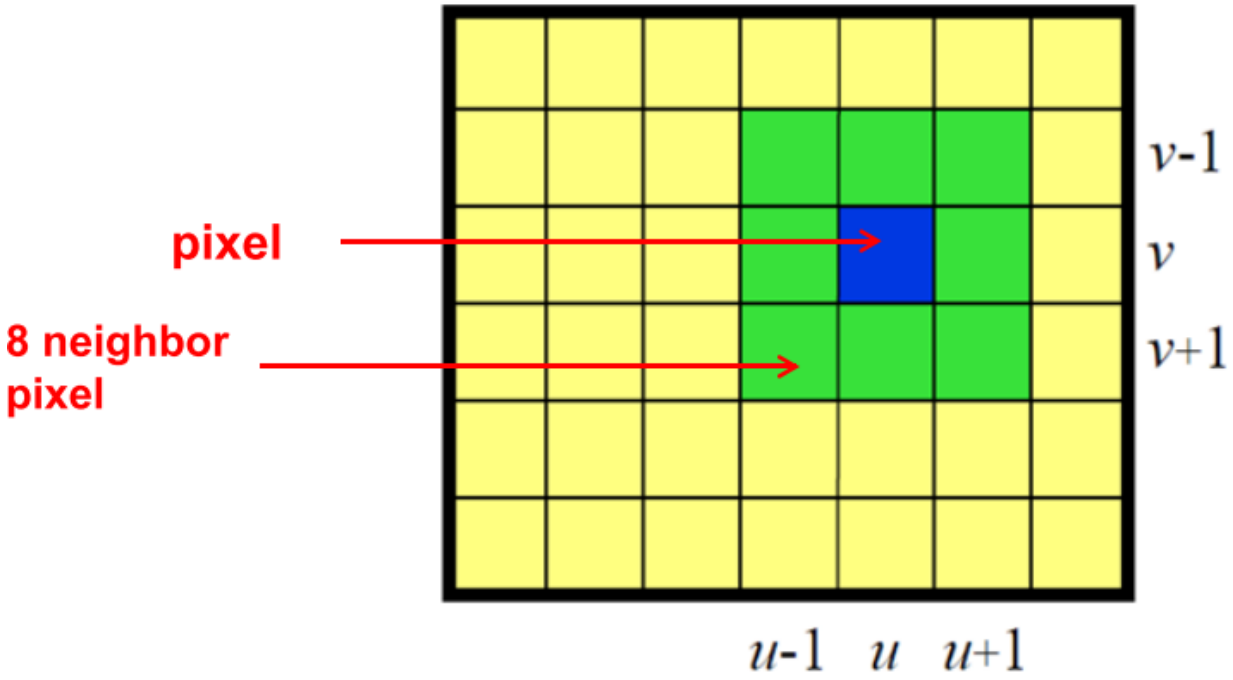


- Global Attribute to Point



Histogram Equalization

Neighborhood



What is Image Filtering?

- A process of transforming original image to create an output image with desirable properties

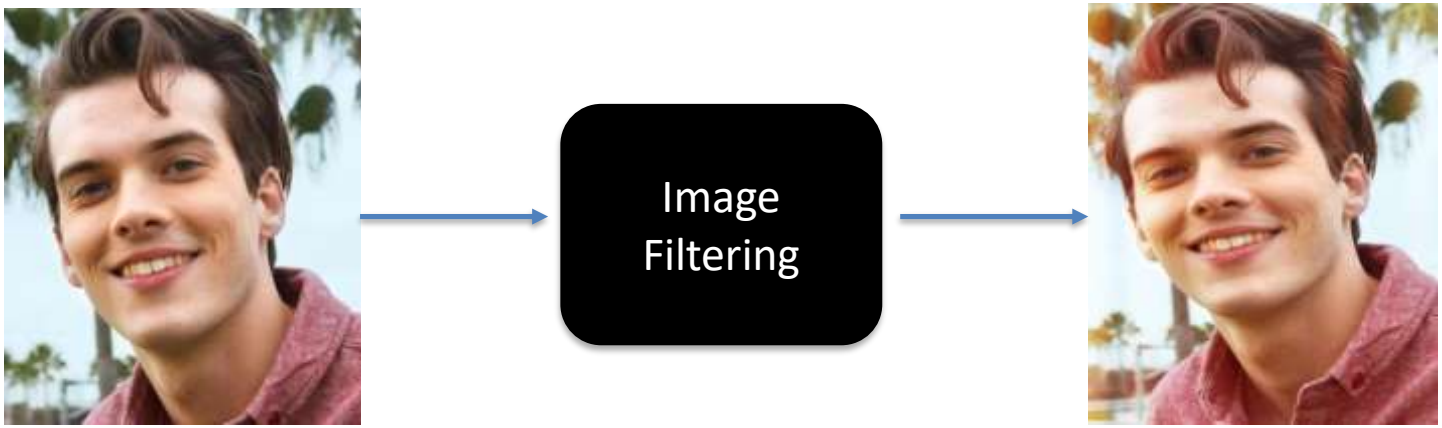
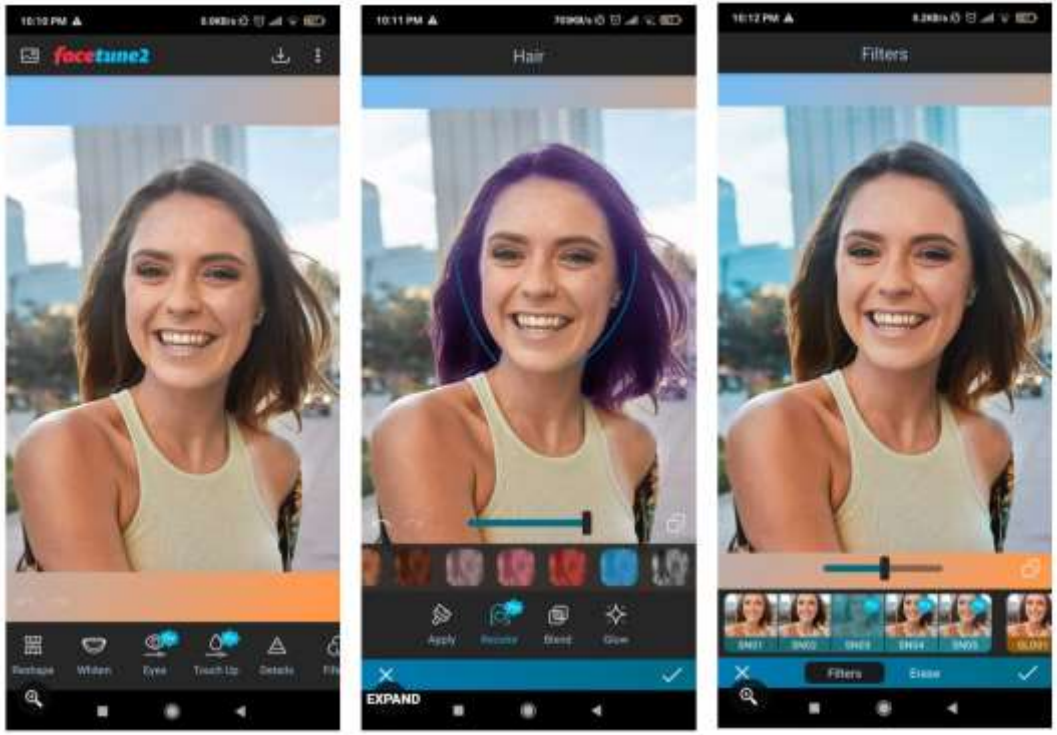


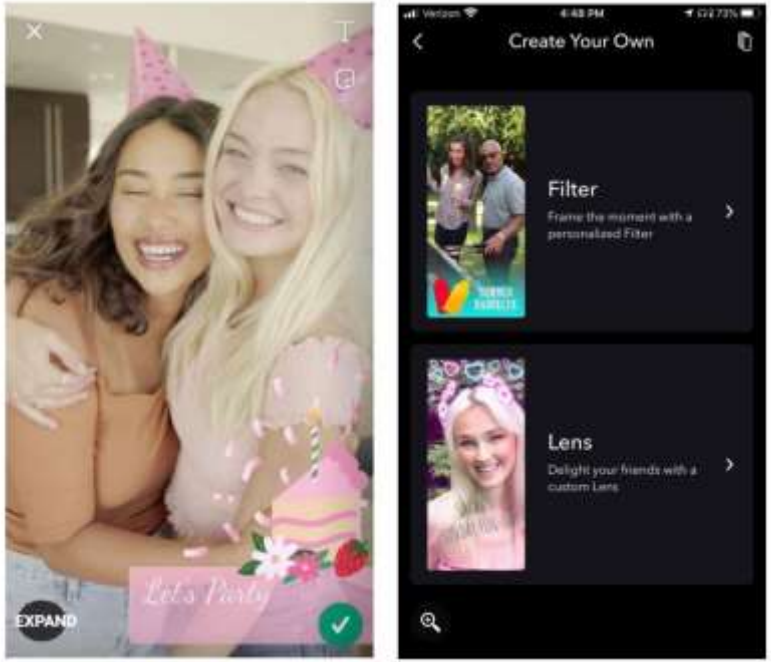
Image Courtesy: https://www.makeuseof.com/best-face-filter-apps-selfies/

Spatial Domain Filtering

Selfie Time…



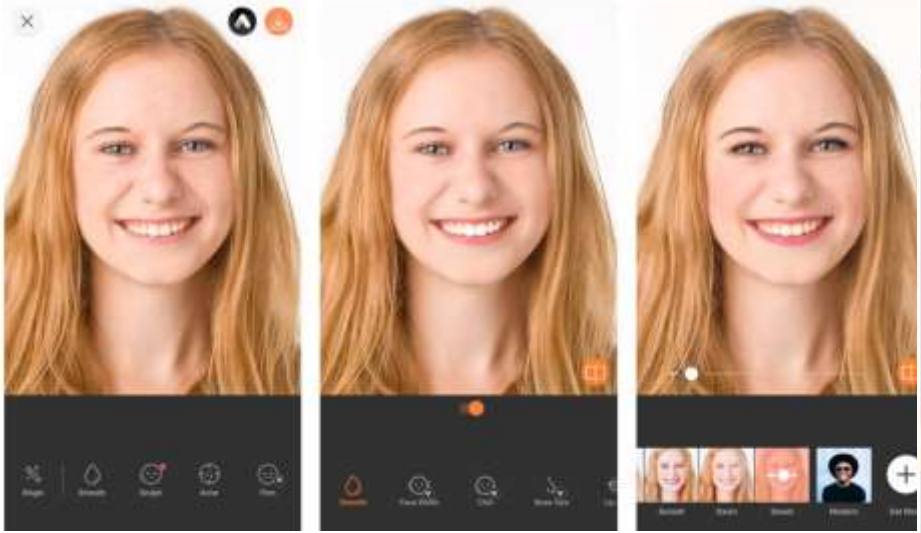
FaceTune2



Snapchat

Image Courtesy: https://www.makeuseof.com/best-face-filter-apps-selfies/

Selfie Time…

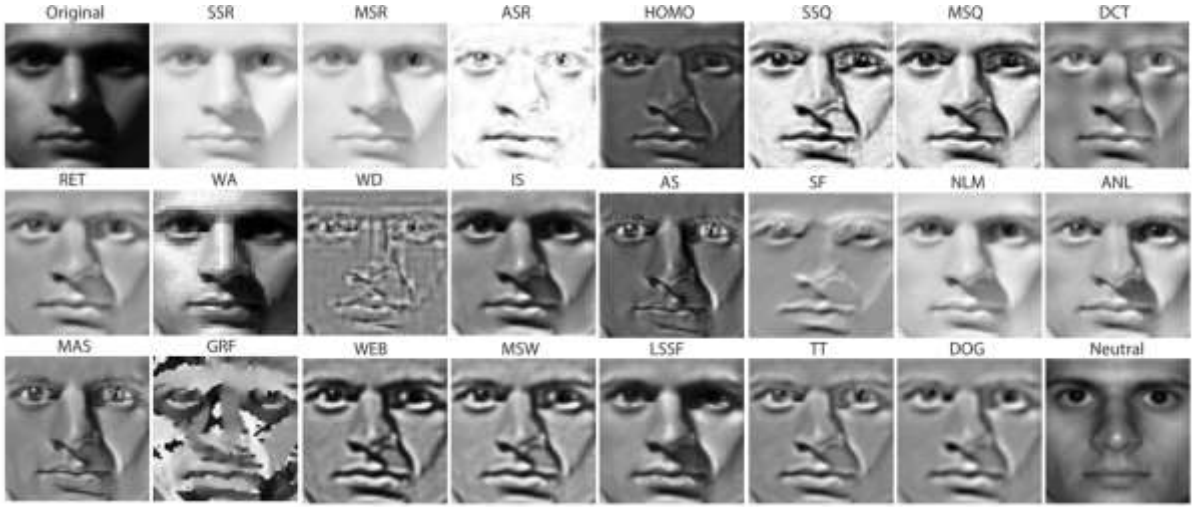


AirBrush

Cymera

Image Courtesy: https://www.makeuseof.com/best-face-filter-apps-selfies/

Filtering (Preprocessing)-Biometrics



INFace: MATLAB Toolbox for Illumination Invariant Face Recognition http://luks.fe.uni-lj.si/sl/osebje/vitomir/face_tools/INFace/

Jufei Xu et al., "Subspace-Based Discrete Transform Encoded Local Binary Patterns Representations for Robust Periocular Matching on NIST's Face Recognition Grand Challenge," IEEE Transactions on Image Processing, 2014

Filtering (Preprocessing)-Biometrics

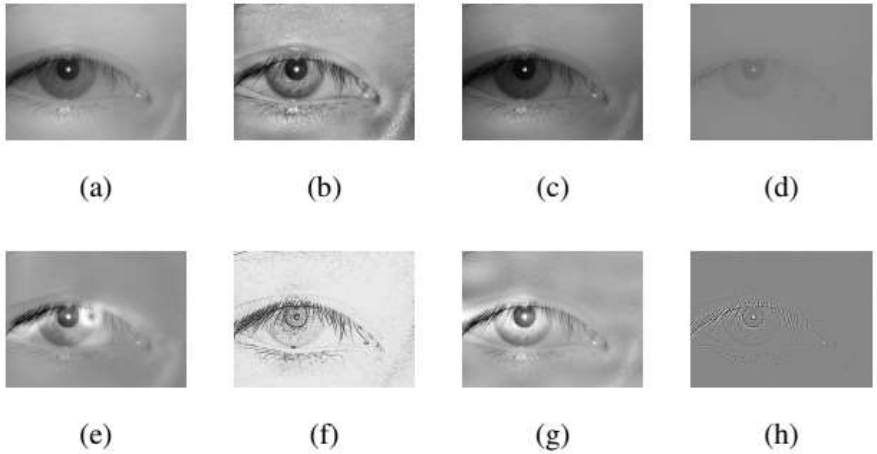
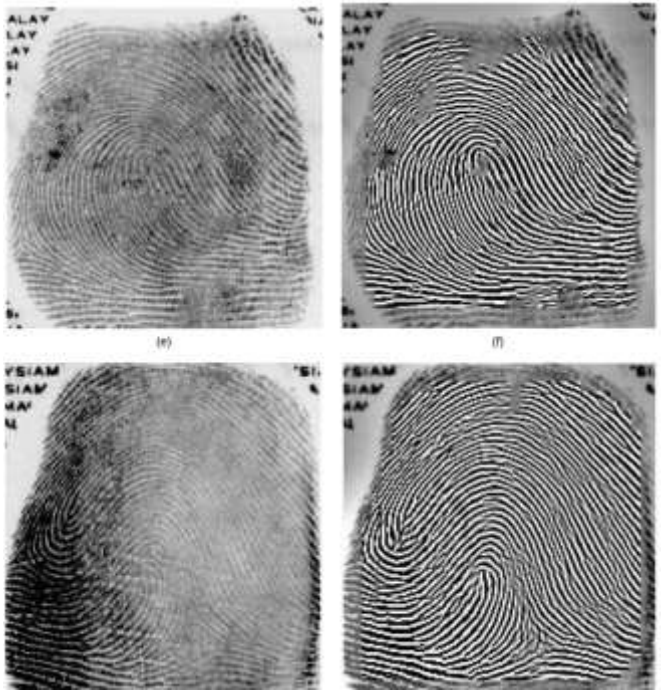


Fig. 4: An example of an NIR iris image subjected to seven illumination normalization schemes. (a) Original, (b) CLAHE, (c) Gamma correction, (d) Homomorphic filtering, (e) MSR, (f) SQI, (g) DCT normalization and (h) DoG.[4]

Banerjee et al., "Impact of Photometric Transformations on PRNU Estimation Schemes: A Case Study Using Near Infrared Ocular Images," IAPR International Workshop on Biometrics and Forensics, 2018

Enhancement-Biometrics



Hong et al., “Fingerprint Image Enhancement: Algorithm and Performance Evaluation ,” IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998

Mean/Average Filter (Smoothing)

```
M = 3
For each valid location [x,y] in S
        a ← Average of intensities in a M x M neighborhood centered on [x,y]
        D[x,y] = round(a)
```

120	190	140	150	200
17	21	30	8	27
89	123	150	73	56
10	178	140	150	18
190	14	76	69	87

x

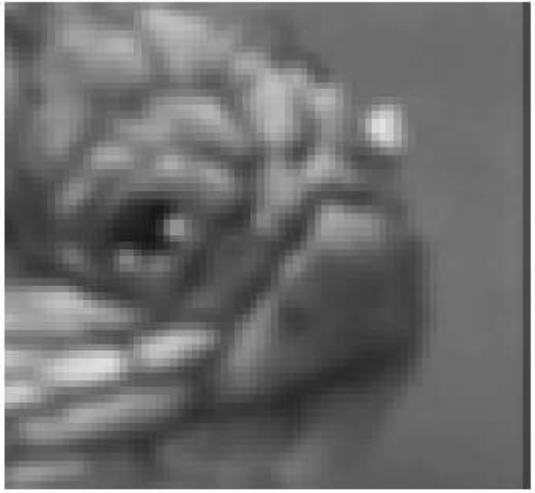
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

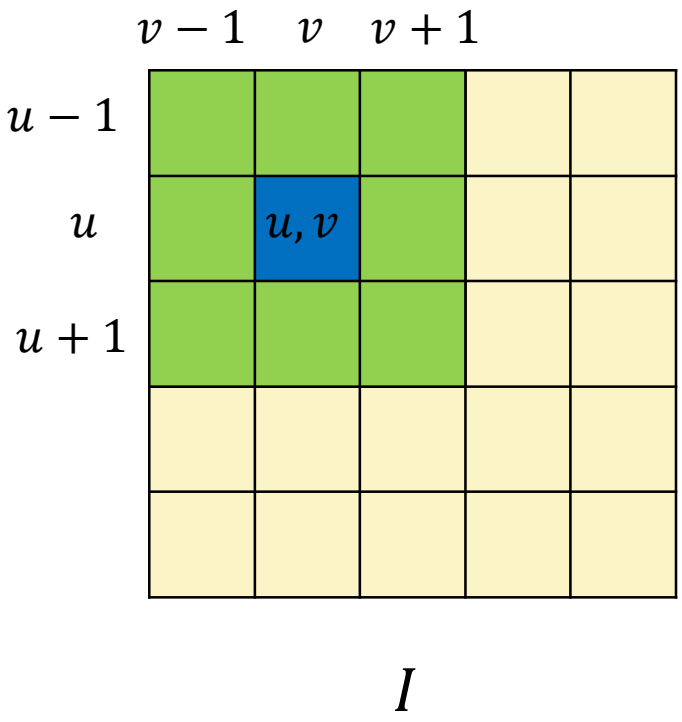
	98			

http://199.116.233.101/index.php?title=File:Animated-kernels.gif

$$1/9 * \begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$



Mean/Average Filter



Note: Coefficients sum to 1

H

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

3 x 3 Weight Mask / Kernel / Filter

$$I'(u,v) \leftarrow \frac{1}{9} \cdot \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j)$$

$$I'(u,v) \leftarrow \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j) \boxed{\bullet\ H(i,j)}$$

Effect of Mask Size

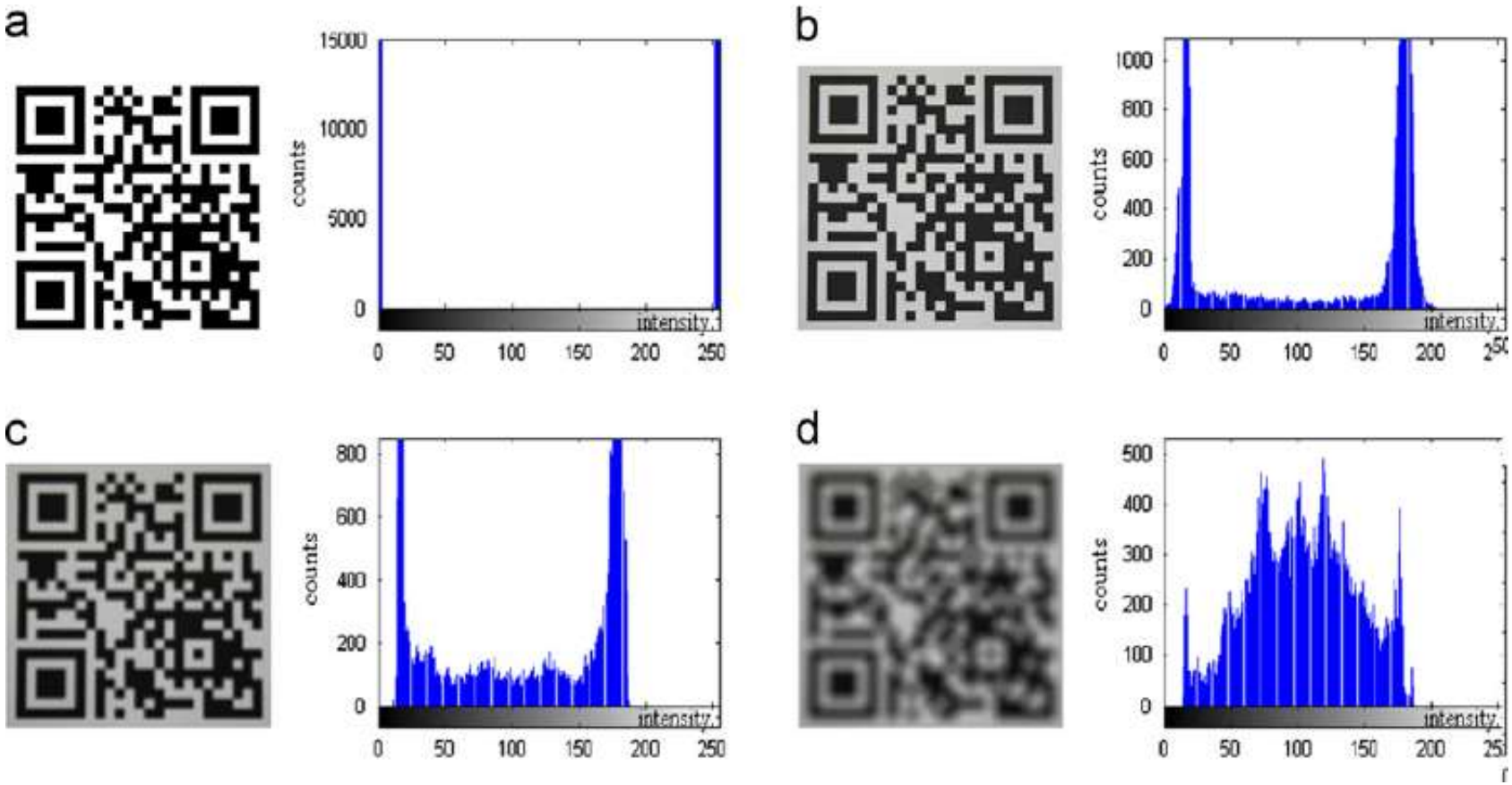
Original Image

[3x3]

[5x5]

[7x7]

Averaging – a histogram perspective



https://www.researchgate.net/figure/Histograms-of-barcode-images-with-different-levels-of-blur-a-a-synthetic-QR-code-image_fig3_224201499

Illustrations of Noisy Images

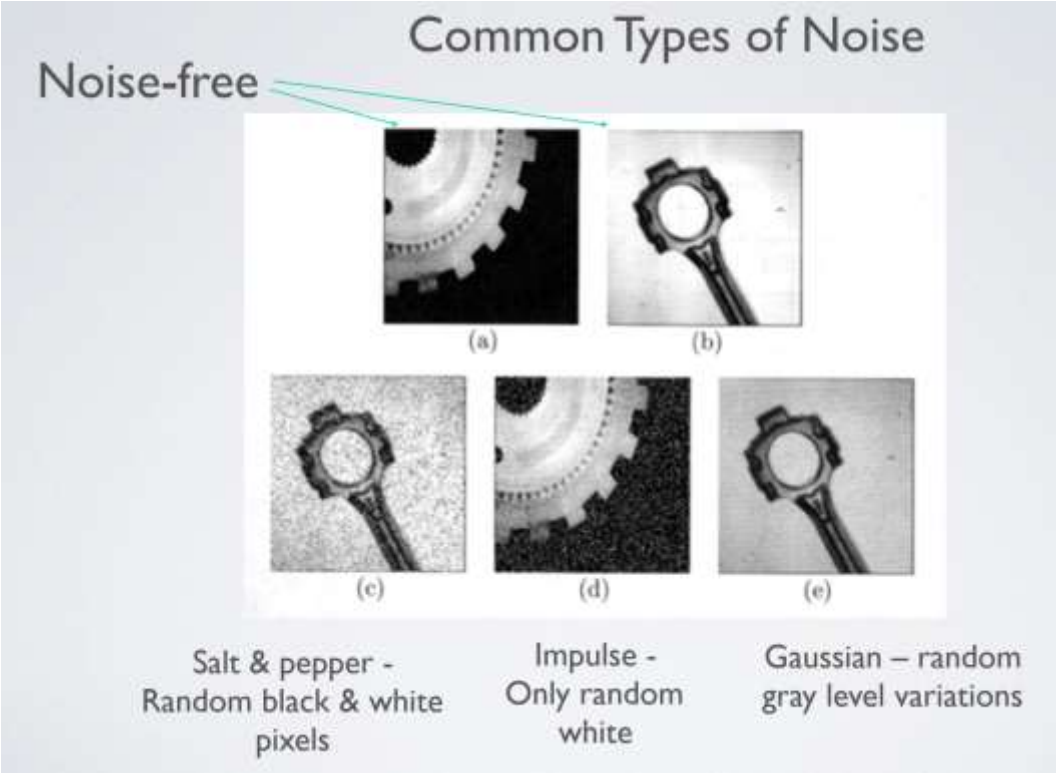
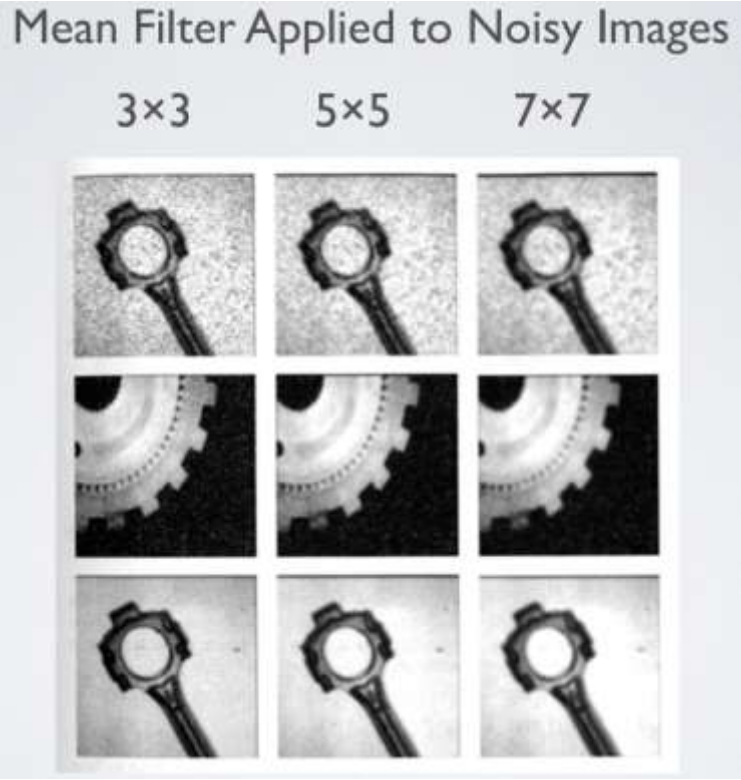
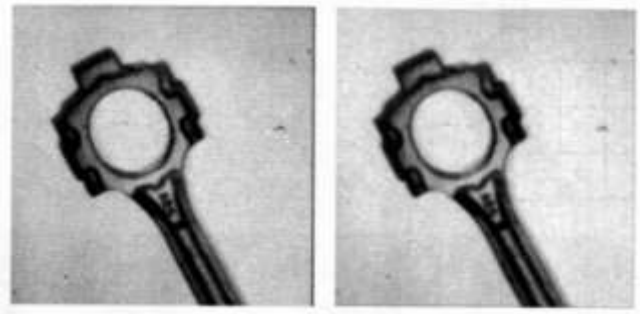


Image Courtesy: http://ece.uprm.edu/~mtoledo/web/6088/F2018/Slides/6_Filtering.pdf

Effect of Mean Filter on Noisy Images

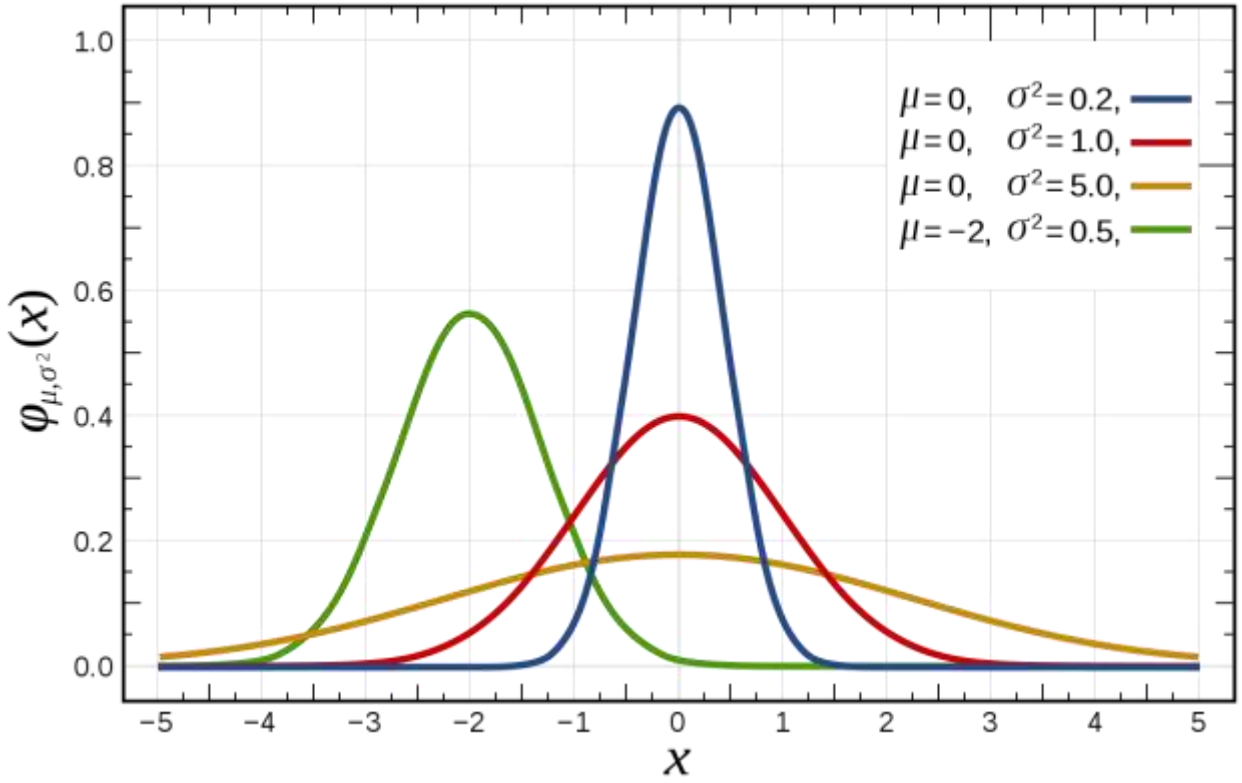




Can sometimes remove too much detail…

Image Courtesy: http://ece.uprm.edu/~mtoledo/web/6088/F2018/Slides/6_Filtering.pdf

Gaussian Function (1-D)



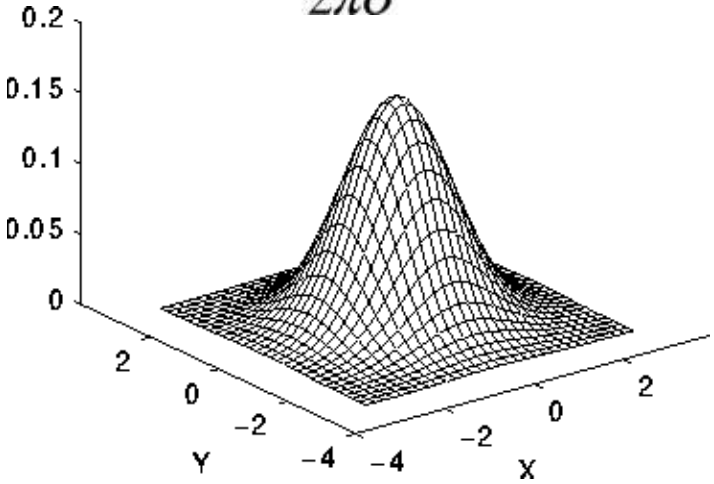
$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

μ: *Mean*, σ: *Standard deviation*

Gaussian Smoothing

- Mask weights are samples of a zero-mean 2-D Gaussian

$$G(x, y) = \frac{1}{2\pi\sigma^2}\exp\{-(x^2 + y^2)/2\sigma^2\}$$



$$\frac{1}{256}$$

1	4	6	4	1
4	16	26	16	4
6	26	43	26	6
4	16	26	16	4
1	4	6	4	1

5×5 Gaussian filter,σ=1

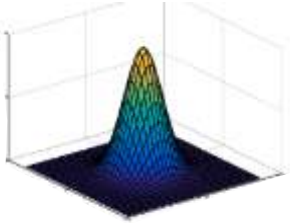
Gaussian Smoothing – Effect of sigma

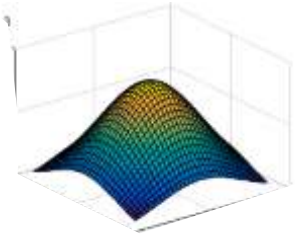
$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\{-(x^2 + y^2)/2\sigma^2\}$$

Original Image
(Sigma 0)

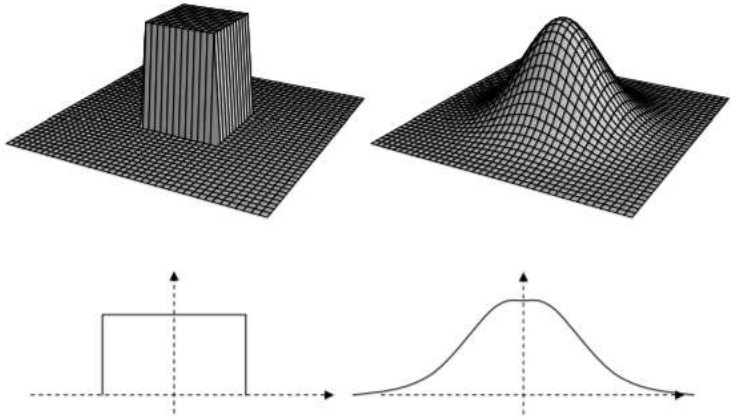
Gaussian Blur
(Sigma 0.7)

Gaussian Blur
(Sigma 2.8)



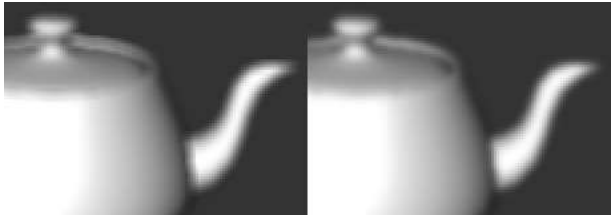


Averaging vs Gaussian filters



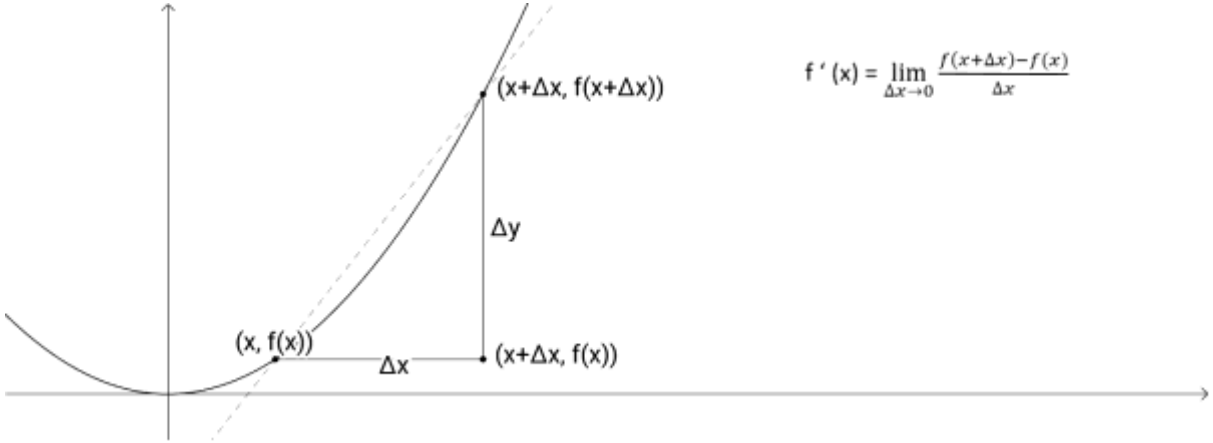
0	0	0	0	0
0	1	1	1	0
0	1	1	1	0
0	1	1	1	0
0	0	0	0	0

0	1	2	1	0
1	3	5	3	1
2	5	9	5	2
1	3	5	3	1
0	1	2	1	0



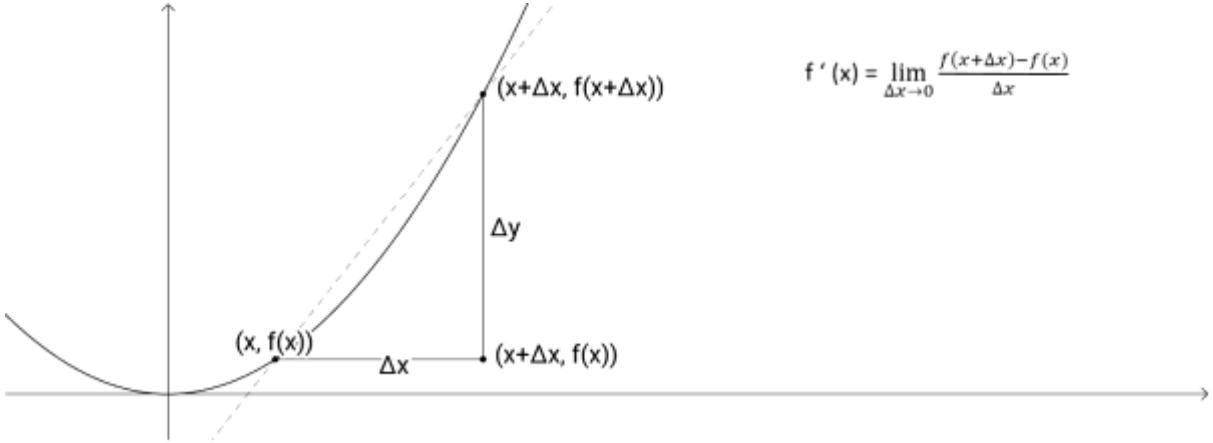
Smoother intensity transitions

Recap: Derivatives

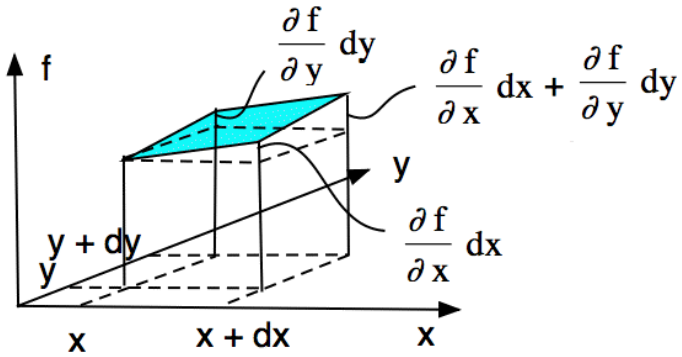


$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

Recap: Derivatives



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



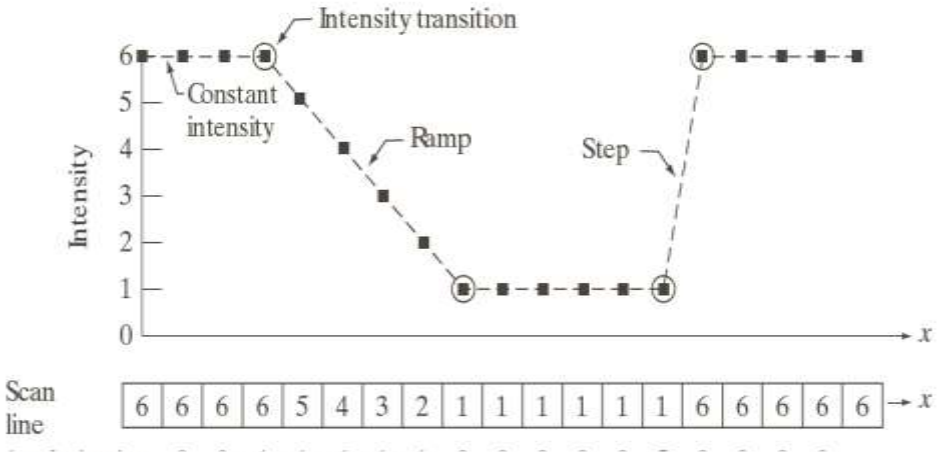
$$\frac{\partial f(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$$

$$\frac{\partial f(x,y)}{\partial y} = \lim_{\Delta y \to 0} \frac{f(x, y+\Delta y) - f(x,y)}{\Delta y}$$

First Derivative (Digital approximation)

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$$

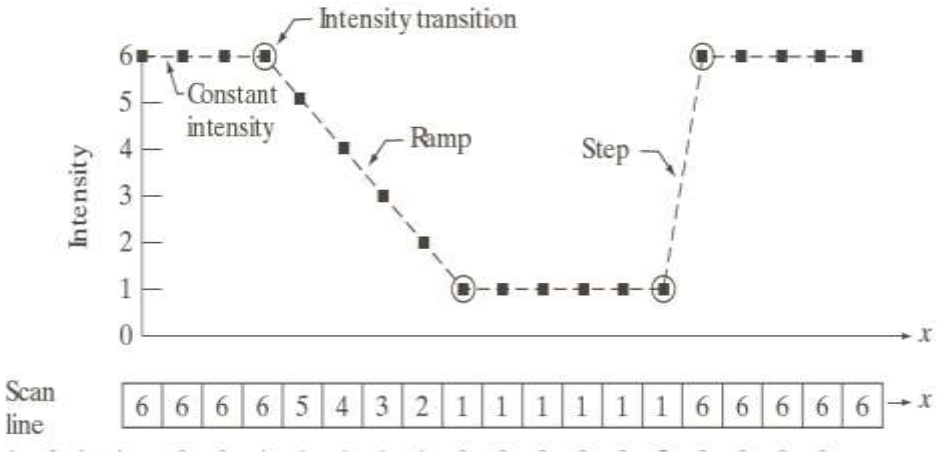
$$\frac{\partial f(x,y)}{\partial x} \sim f[x+1, y] - f[x, y]$$



- First Derivative (Digital approximation)

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$$

$$\frac{\partial f(x,y)}{\partial x} \sim f[x+1,y] - f[x,y]$$



- Second Derivative (Digital Approximation)

$$\frac{\partial^2 f(x,y)}{\partial x^2} \sim (f[x+1,y] - f[x,y]) - (f[x,y] - f[x-1,y])$$

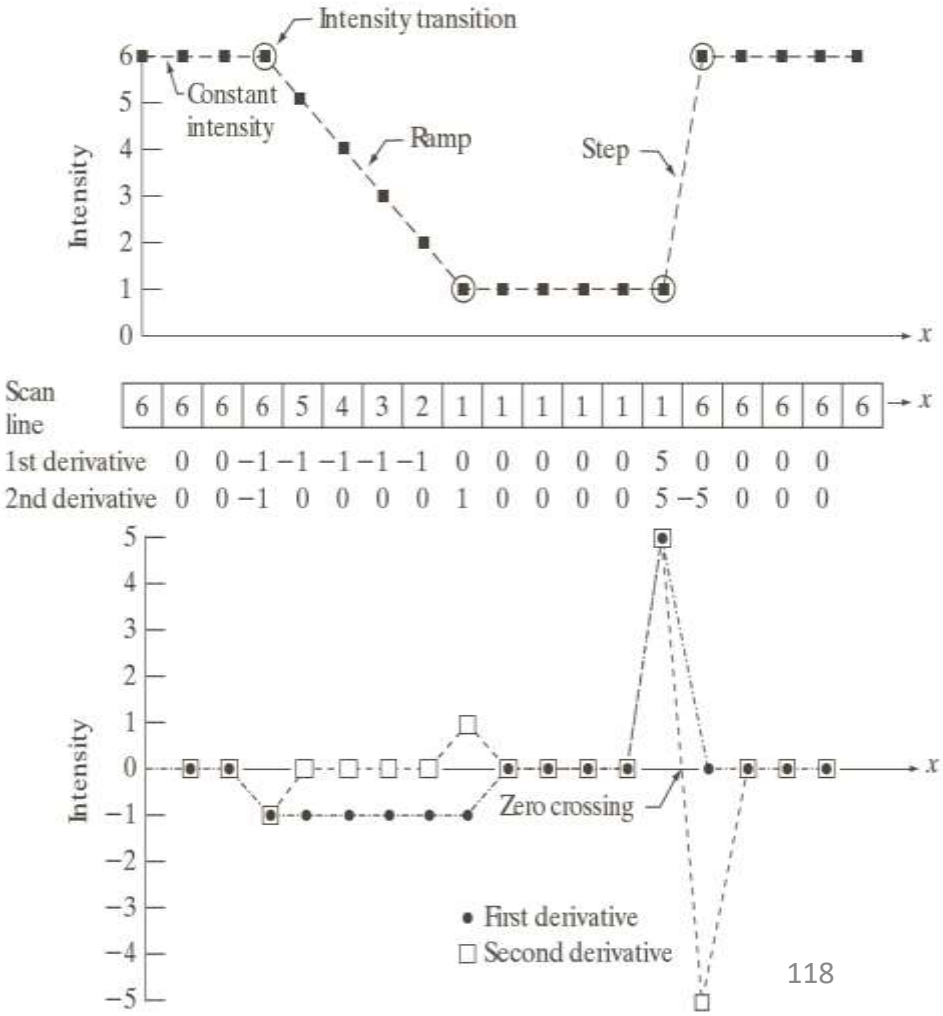
- First Derivative (Digital approximation)

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x, y) - f(x,y)}{\Delta x}$$

$$\frac{\partial f(x,y)}{\partial x} \sim f[x+1,y] - f[x,y]$$

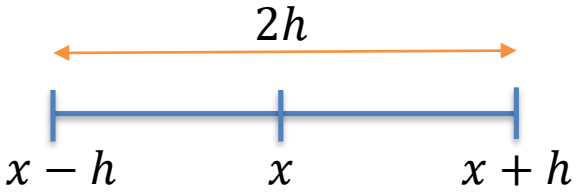
- Second Derivative (Digital Approximation)

$$\frac{\partial^2 f(x,y)}{\partial x^2} \sim (f[x+1,y] - f[x,y]) - (f[x,y] - f[x-1,y])$$



Alt: Derivative as symmetric Difference

$$\frac{\partial f(x,y)}{\partial x} \sim f[x+1,y] - f[x,y]$$



$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x-h)}{2h}$$

$$f'(x) = \lim_{h \to 0} \frac{1.f(x+h) + 0.f(x) - 1.f(x-h)}{2h}$$

$$f'(x) = \lim_{h \to 0} \frac{-1.f(x-h) + 0.f(x) + 1.f(x+h)}{2h}$$

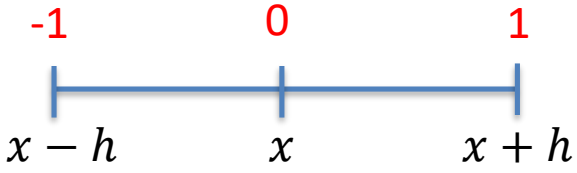


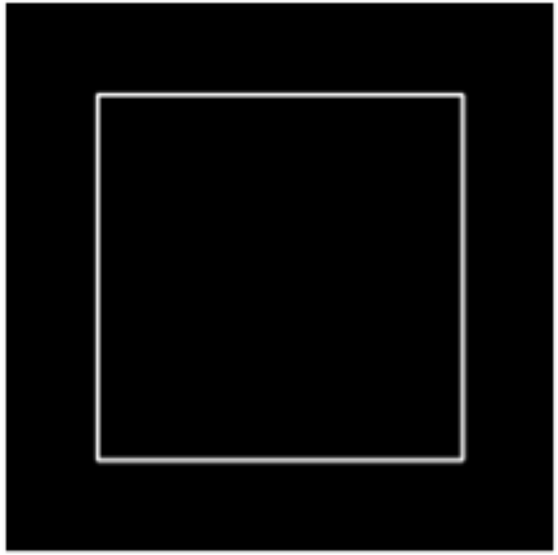
Image Gradient and Edges

$$\frac{f(x+h,y)-f(x-h,y)}{2h} \Rightarrow \begin{array}{|c|c|c|}\hline -1 & 0 & 1 \\ \hline \end{array}$$

x-derivative

$$\frac{f(x,y+h)-f(x,y-h)}{2h} \Rightarrow \begin{array}{|c|}\hline -1 \\ \hline 0 \\ \hline 1 \\ \hline \end{array}$$

y-derivative



Image

Gradient in x

Gradient in y

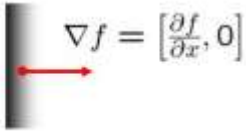
https://i1.wp.com/theailearner.com/wp-content/uploads/2019/05/gradxy.png?resize=768%2C280&ssl=1

Image gradient

The gradient of an image:

$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

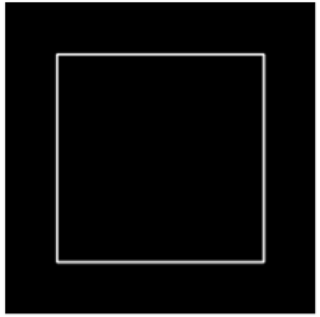
The gradient points in the direction of most rapid change in intensity



$$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$$

$$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$$

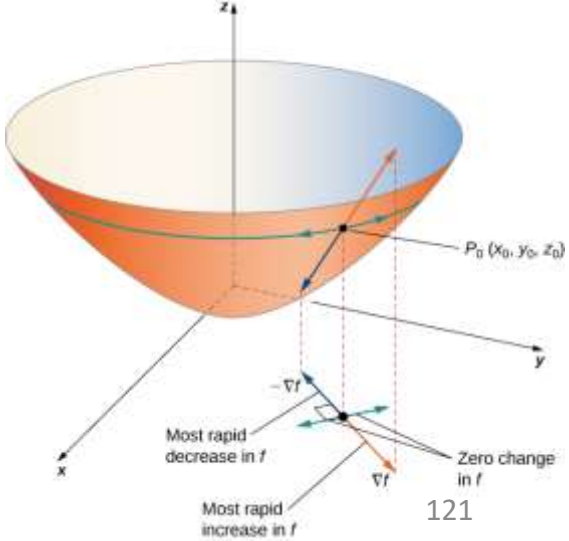
$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$



Image

Gradient in x

Gradient in y



Edge is perpendicular to gradient

Image | Gradient in x | Gradient in y

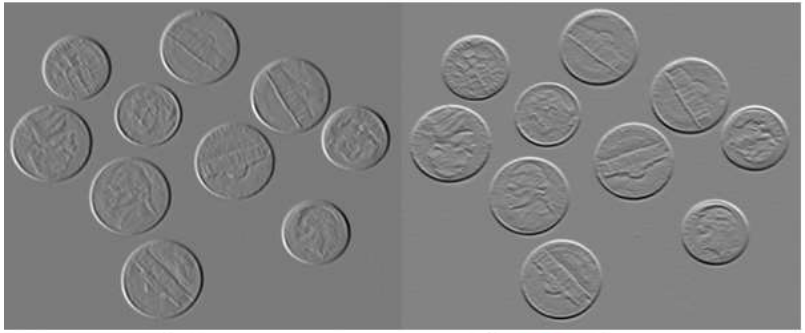
-1	0	+1
-1	0	+1
-1	0	+1

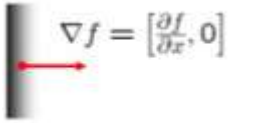
G_x

+1	+1	+1
0	0	0
-1	-1	-1

G_y

Gradient Magnitude and Orientation





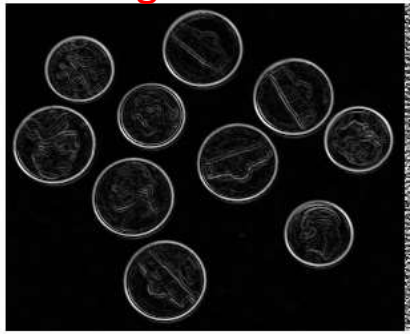
$\nabla f = \left[\frac{\partial f}{\partial x}, 0\right]$

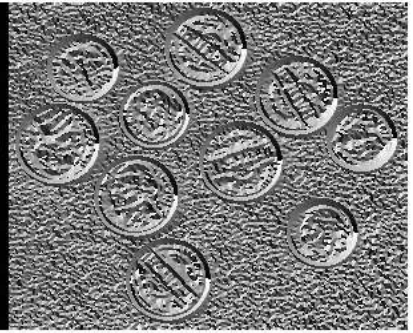
$\nabla f = \left[0, \frac{\partial f}{\partial y}\right]$

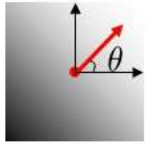
Magnitude

Orientation

$$\|\nabla f\| = \sqrt{\left(\frac{\partial f}{\partial x}\right)^2 + \left(\frac{\partial f}{\partial y}\right)^2}$$







$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$

$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

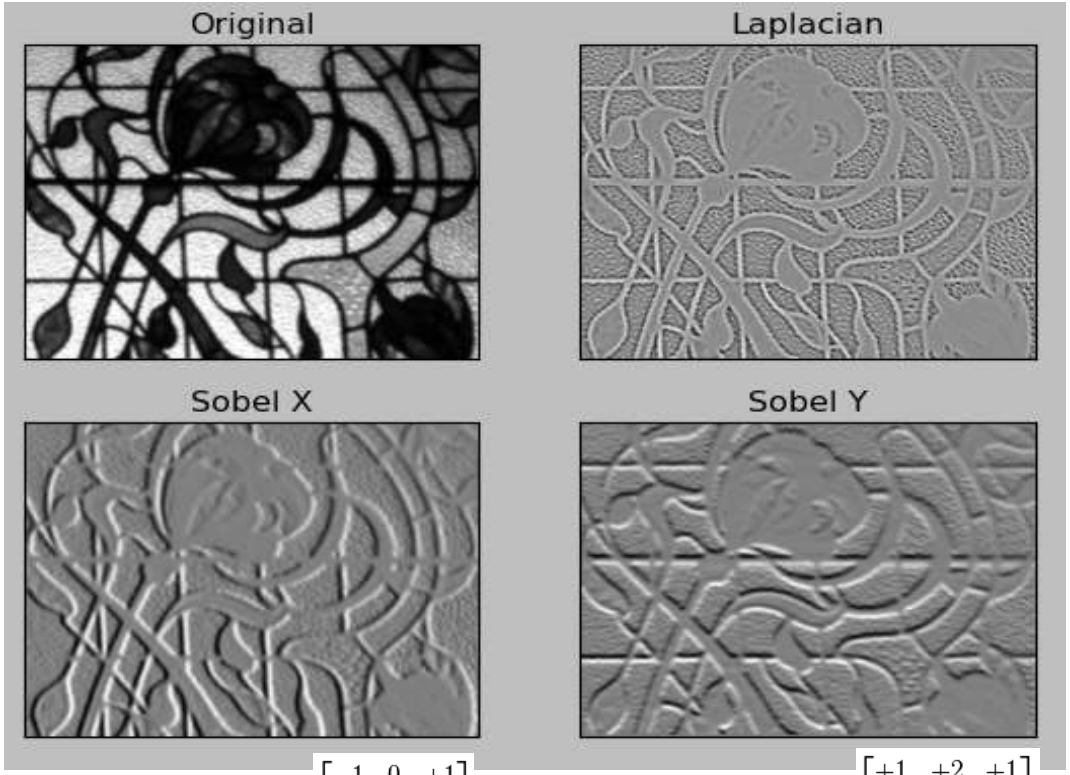
2-D Laplacian Filter

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$

$$\nabla^2 f = f(x+1, y) + f(x-1, y) + f(x, y+1) + f(x, y-1) - 4f(x, y)$$

0	1	0
1	-4	1
0	1	0

Edge Masks – Sobel , Laplacian



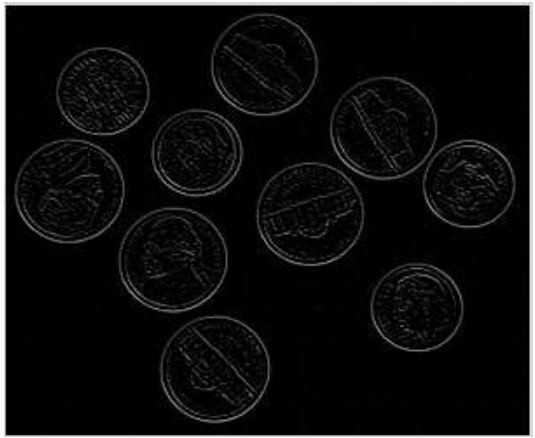
0	−1	0
−1	4	−1
0	−1	0

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

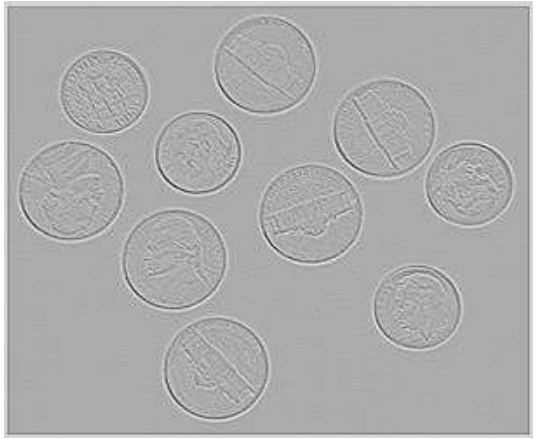
$$\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

Image Sharpening Using Laplacian

$I(u,v)$



$\nabla^2 I(u,v)$



$\nabla^2 I(u,v) + 128$

(For visualization)

$I'(u,v) = I(u,v) + \nabla^2 I(u,v)$

Unsharp masking

- What does blurring take away?

–

=

Difference between original and smoothed images

- Let's add it back:

+ k

=

Add original and weighted detail images

Corner cases, Padding

```
M = 3
For each valid location [x,y] in S
        a ← Average of intensities in a M x M neighborhood centered on [x,y]
        D[x,y] = round(a)
```

120	190	140	150	200
17	21	30	8	27
89	123	150	73	56
10	178	140	150	18
190	14	76	69	87

x

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

=

	98			

Edge Detection-Biometrics

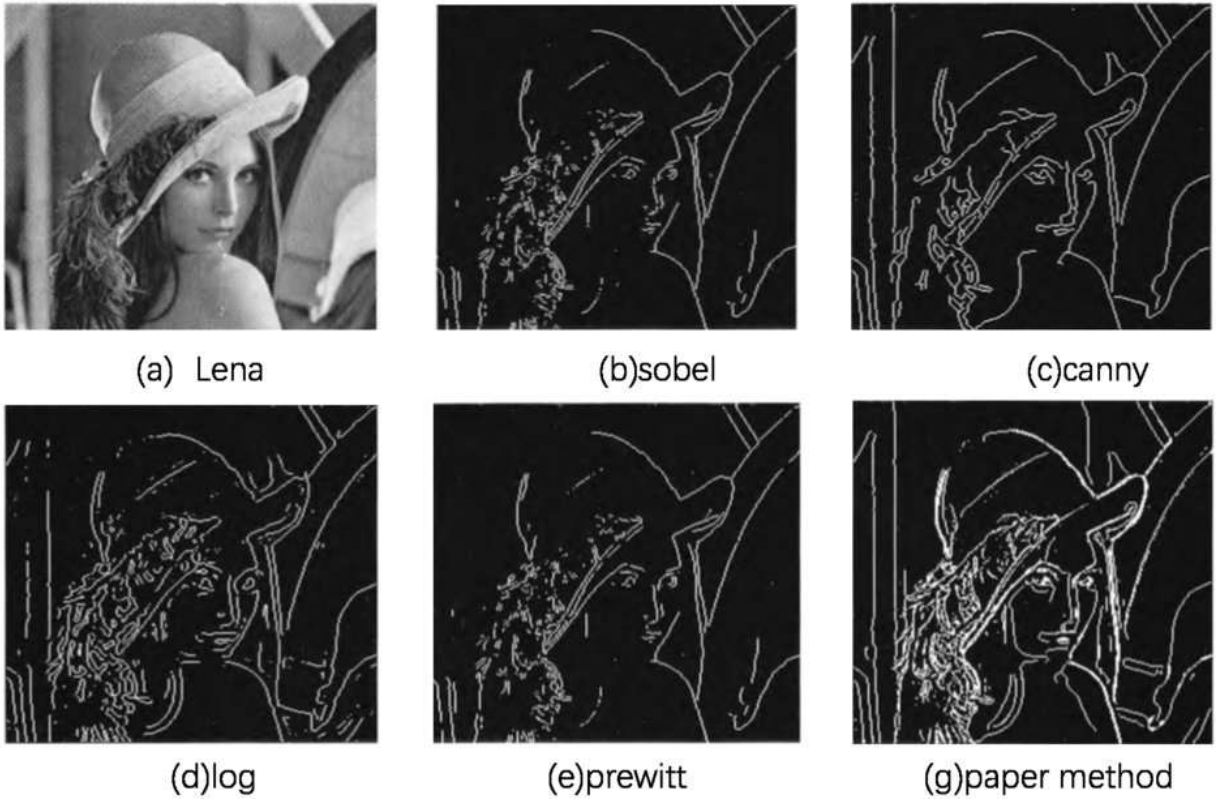


Image Courtesy: Hua et al., "An edge detection method with boundary reserved based on non-subsampled contourlet transform for remote sensing imagery," Optics and Photonics for Information Processing XI, SPIE, 2017

Edge Detection-Biometrics

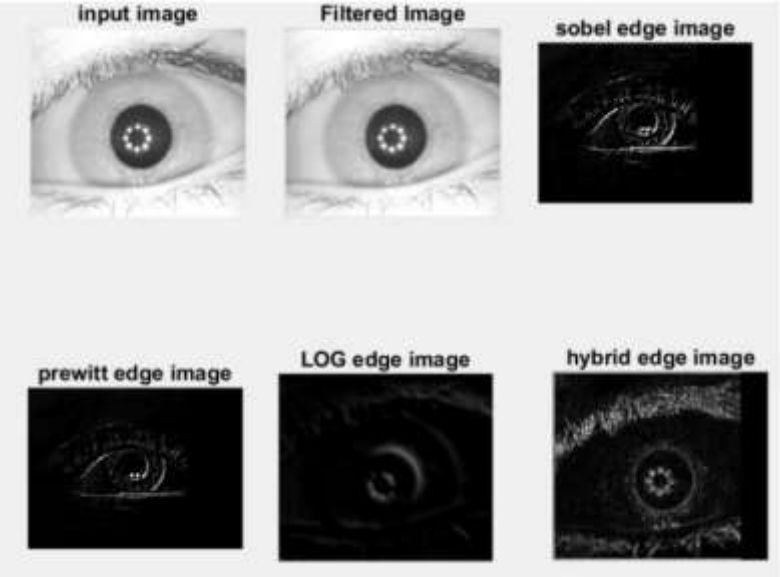
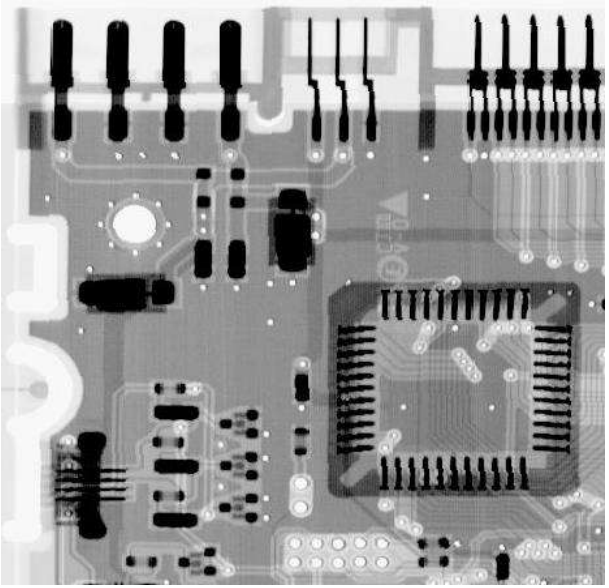


Image Courtesy: Kaur et al., “Comparison of edge detection techniques for iris recognition,” Intl. Journal of Computer Applications, 2016

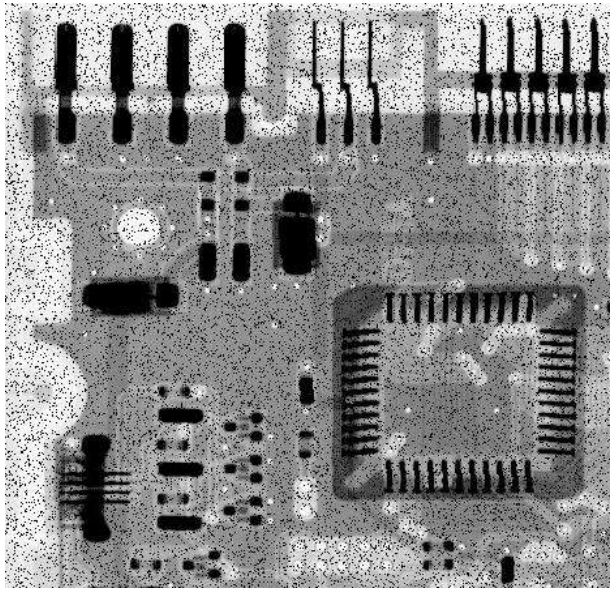
Spatial Domain Filtering - Approaches

- Linear (Average, Gaussian, Prewitt, Sobel, Laplacian)
- Non-linear

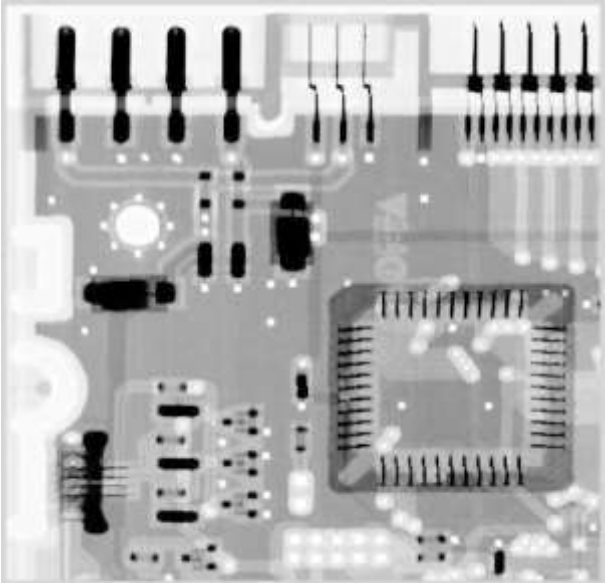
Non-linear Spatial Filters (max)



pepper noise



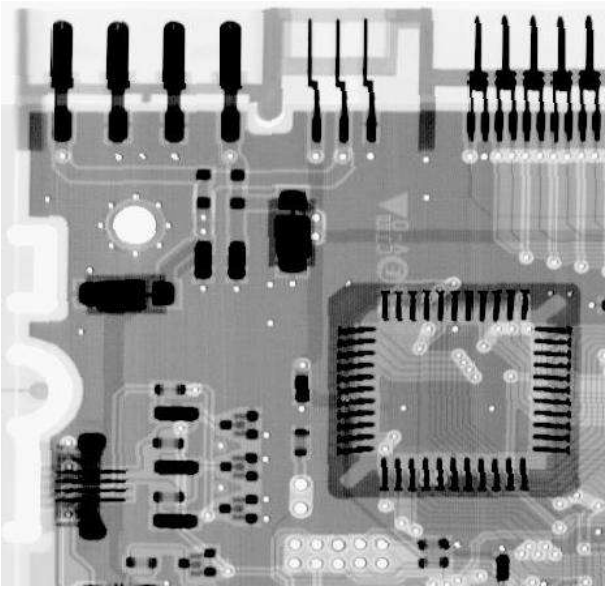
After applying max filter

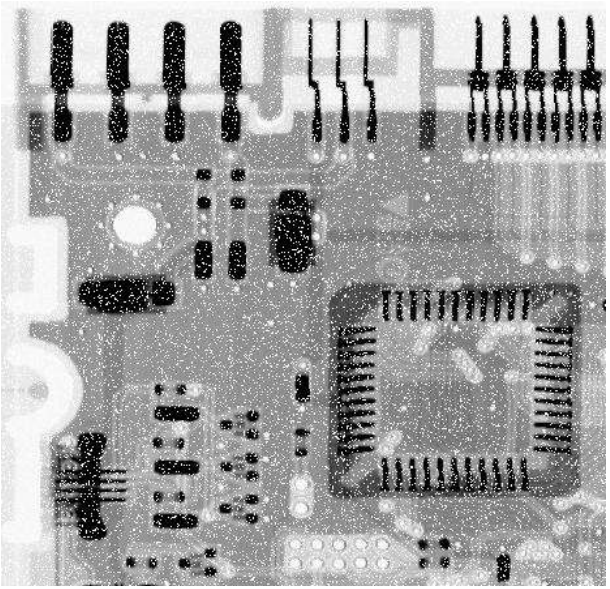


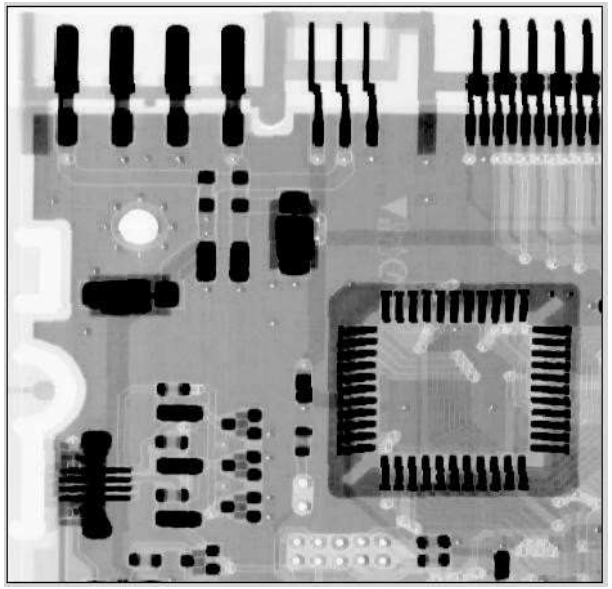
Non-linear Spatial Filters (min)

salt noise

After applying min filter



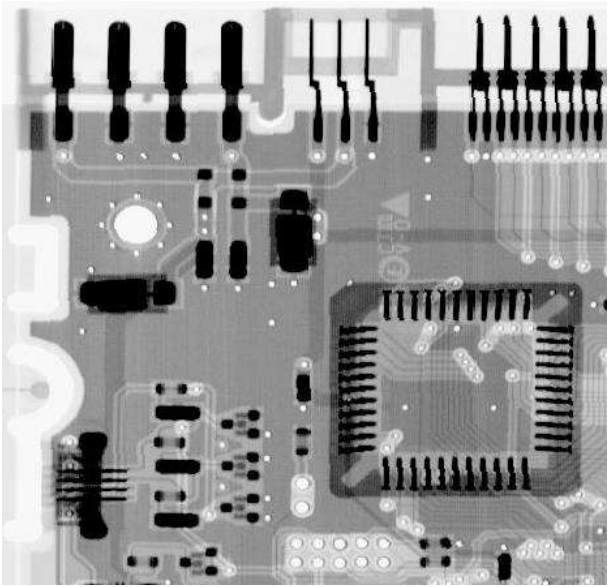


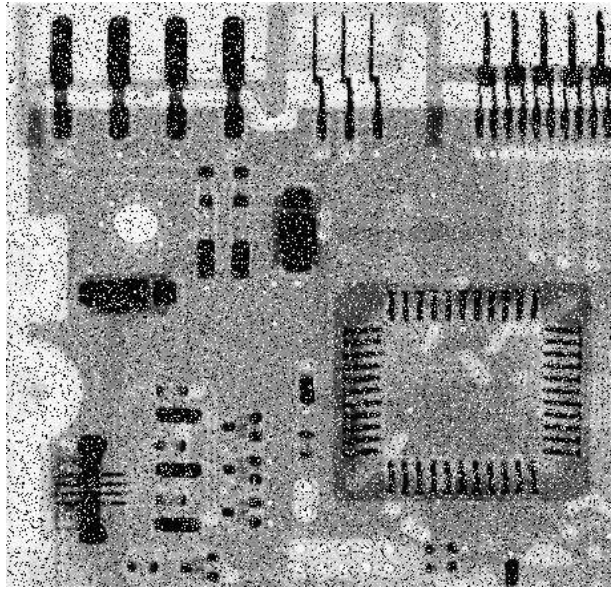


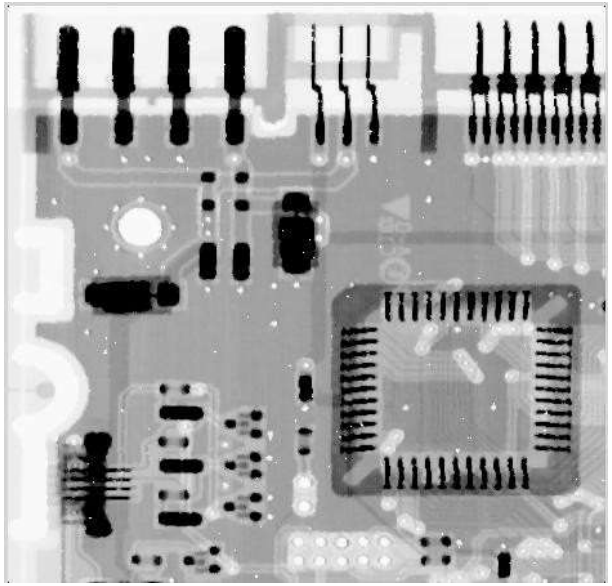
Non-linear Spatial Filters (median)

salt & pepper noise

After applying median filter







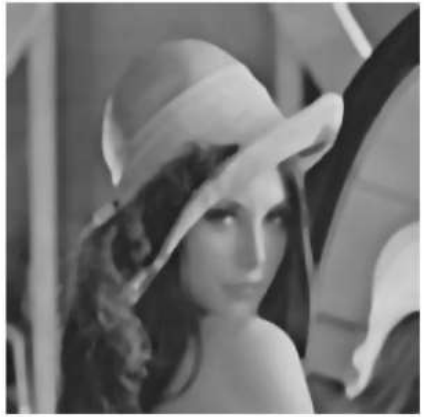
max, min, median → also known as rank / order statistic filters

- Mean: blurs image, removes simple noise, no details are preserved
- Gaussian: blurs image, preserves details only for small σ.
- Median: preserves some details, good at removing strong noise

original

3x3 mean

3x3 gaussian

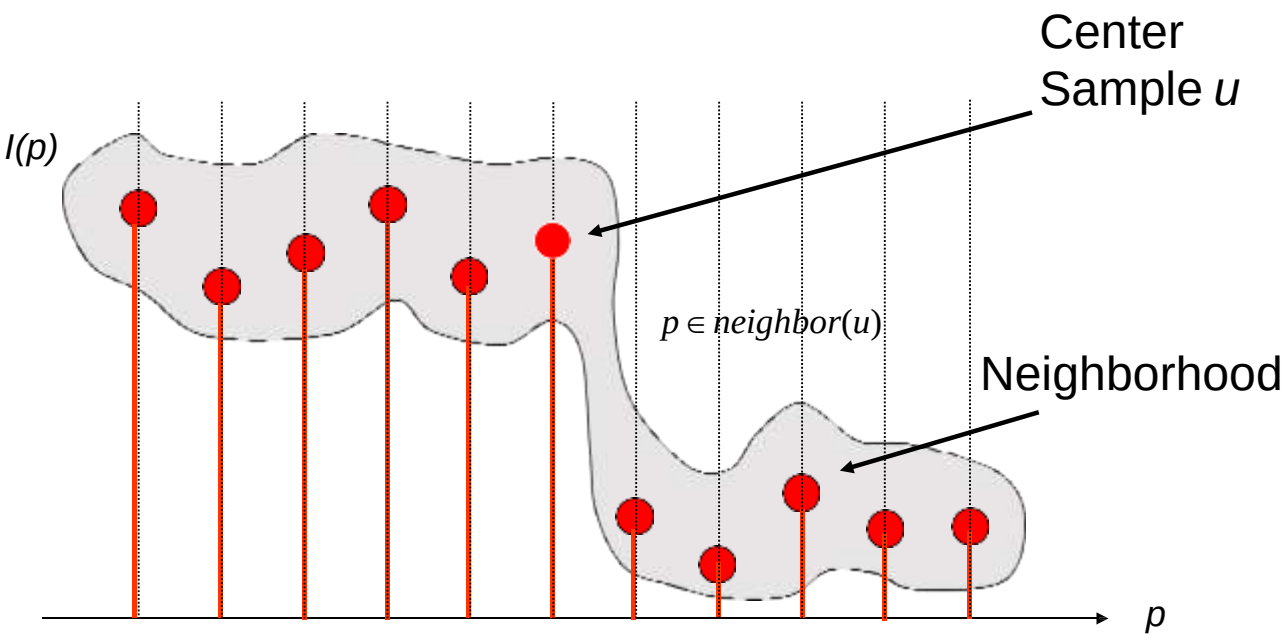


3x3 median

Bilateral Filtering (Edge preserving smoothing)

Original image taken from cs.cityu.edu.hk

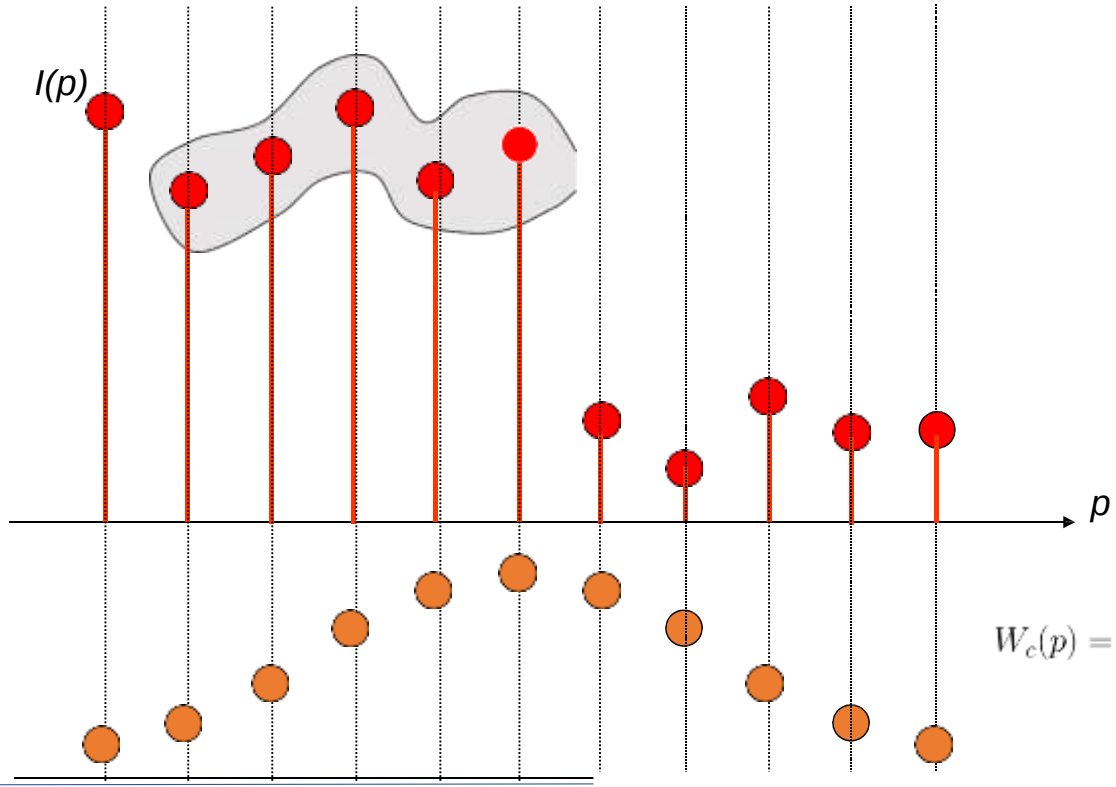
Bilateral Filters - 1D example



It is clear that in weighting this neighborhood,
we would like to preserve the step

Gaussian Weights

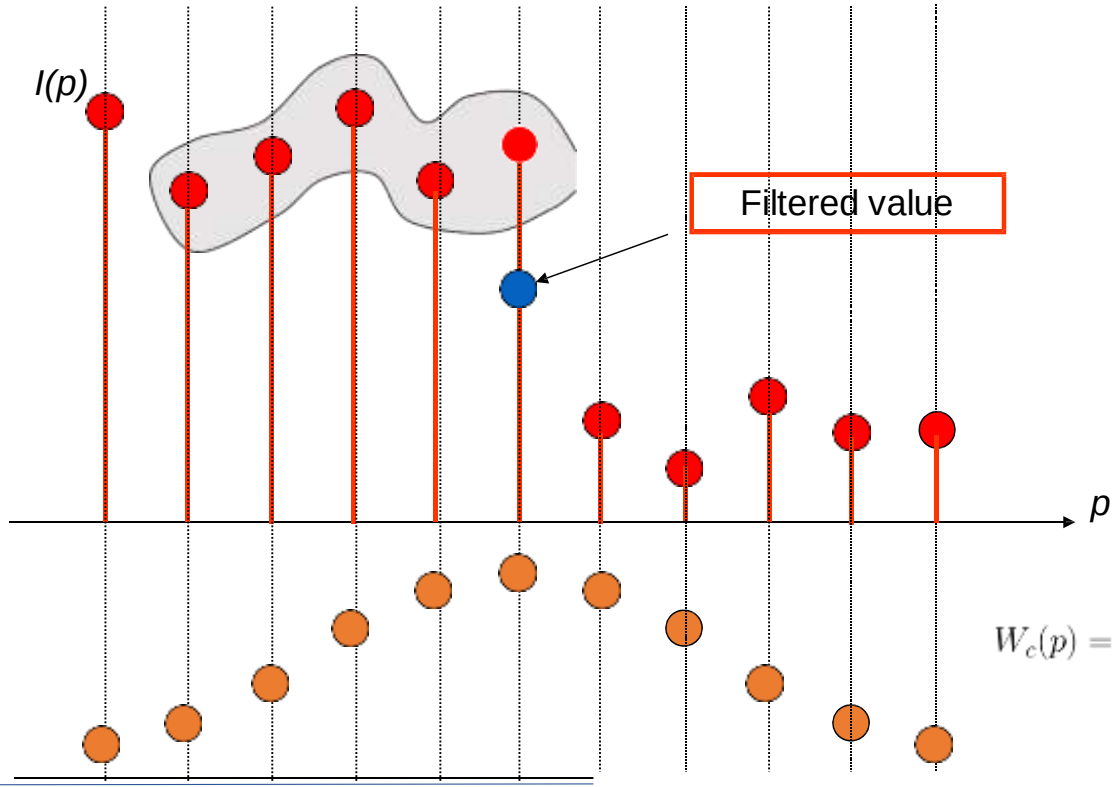
- Gaussian Weights



$$W_c(p) = exp(-\frac{(u-p)^2}{2\sigma_c^2})$$

Filtered Output

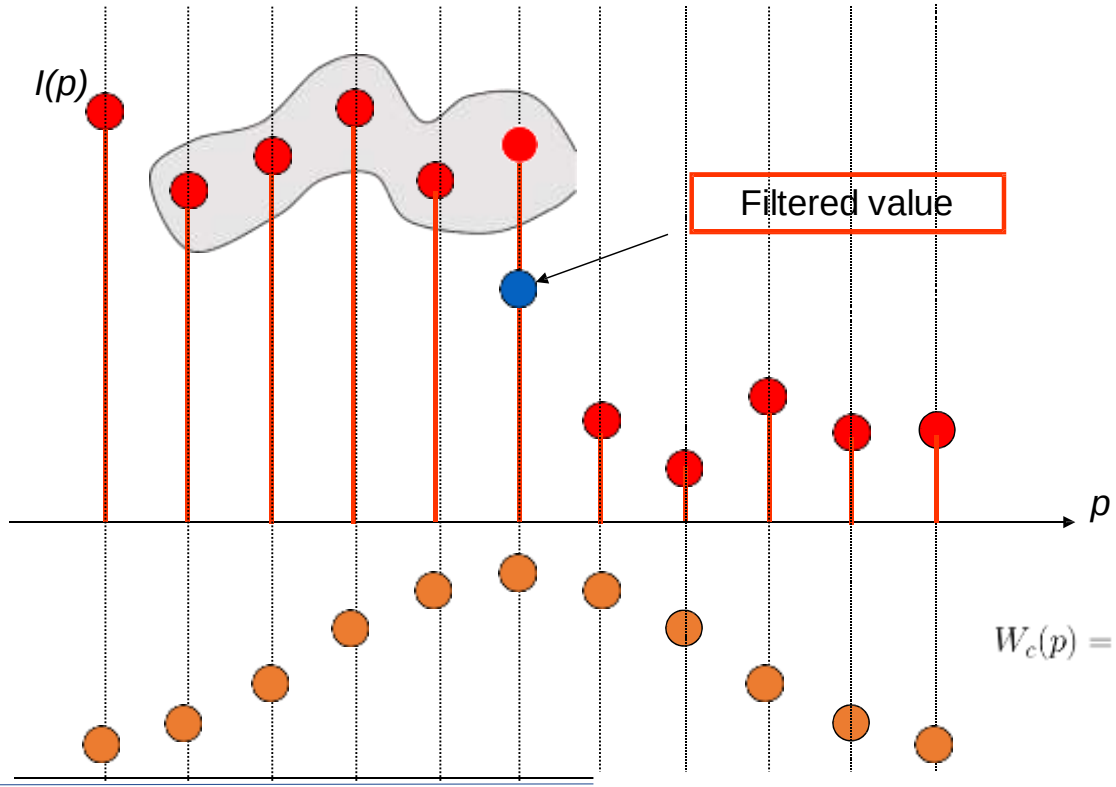
- Weighted sum on the $W_c(p)$



$$W_c(p) = exp(-\frac{(u-p)^2}{2\sigma_c^2})$$

Edge loss

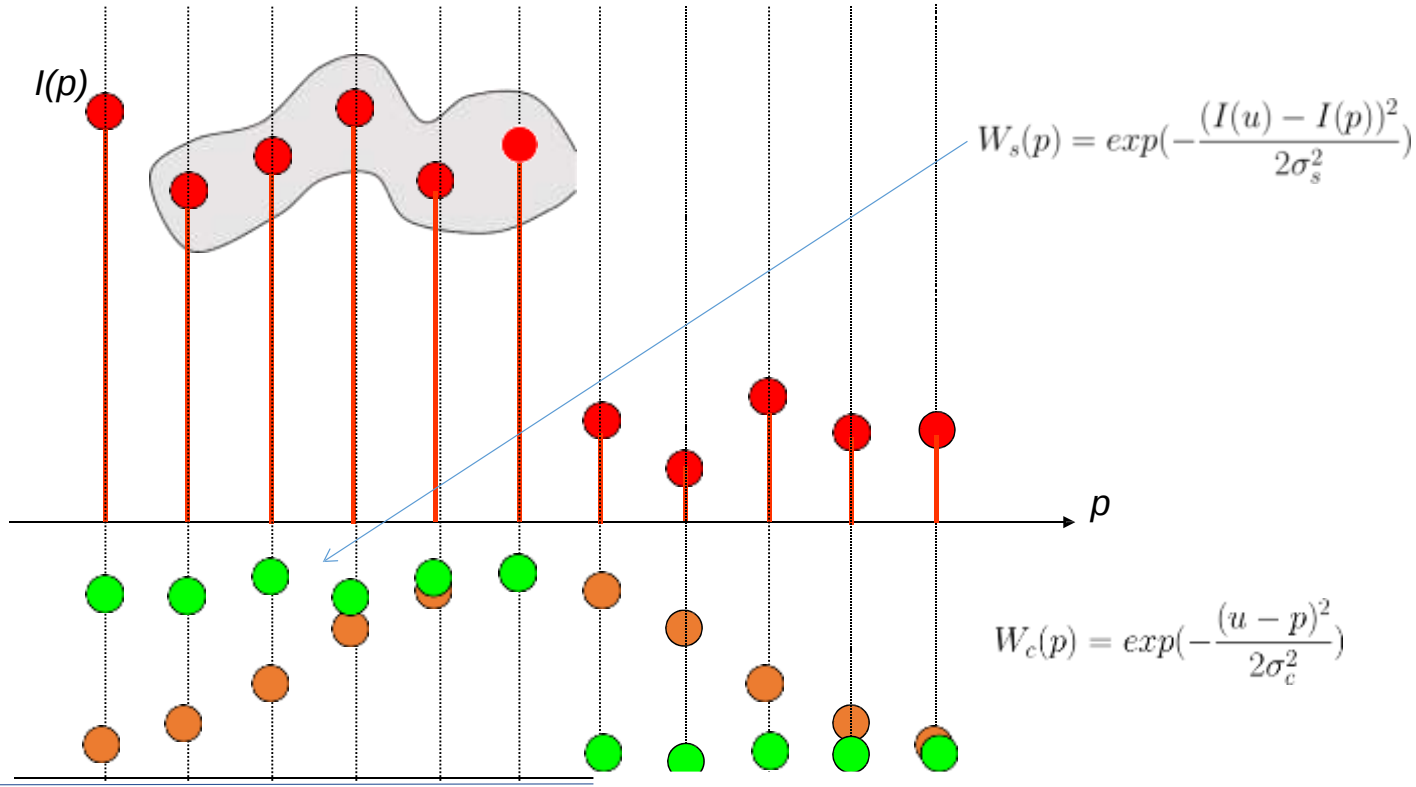
- Edge is smoothed/lost



$$W_c(p) = exp(-\frac{(u-p)^2}{2\sigma_c^2})$$

Photometric Weights

- Introducing Photometric weights



Bilateral Filtering

- Fitler Weights derived from both geometric and photometric distances

Denoise

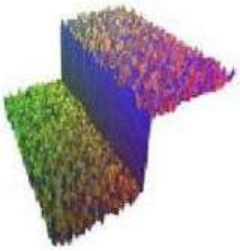
Feature preserving

$$I'(u) = \frac{\sum_{p \in N(u)} e^{-\frac{\|u-p\|^2}{2\sigma_c^2}} e^{-\frac{|I(u)-I(p)|}{2\sigma_s^2}} I(p)}{\sum_{p \in N(u)} e^{-\frac{\|u-p\|^2}{2\sigma_c^2}} e^{-\frac{|I(u)-I(p)|}{2\sigma_s^2}}}$$

Normalization

Bilateral Filter

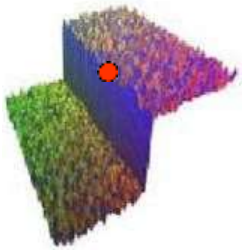
- Illustration of bilateral filter changes



input

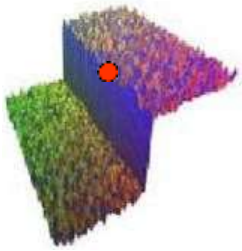
Bilateral Filter

- Illustration of bilateral filter changes

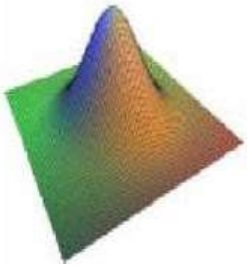


input

Bilateral Filter

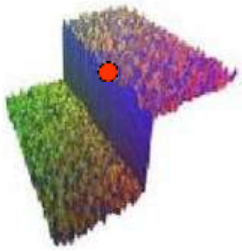


input

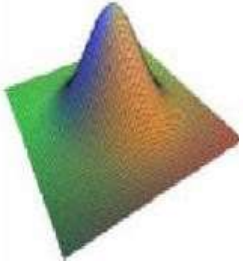


W_c

Bilateral Filter



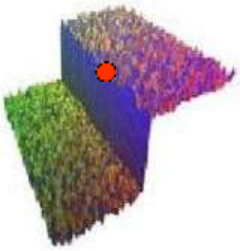
input



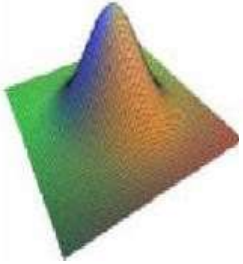
W_c

W_s

Bilateral Filter

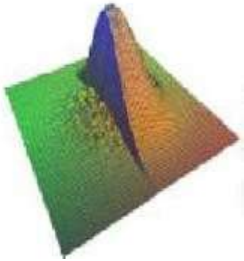


input



W_c

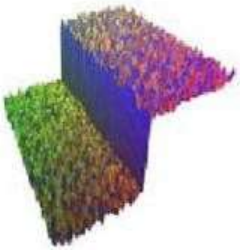
W_s



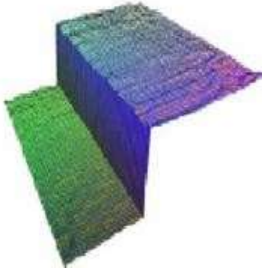
$W_s * W_c$

Bilateral Filter

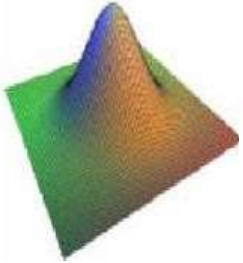
- Filtering process



input

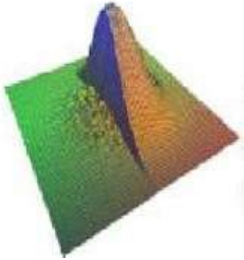


Output



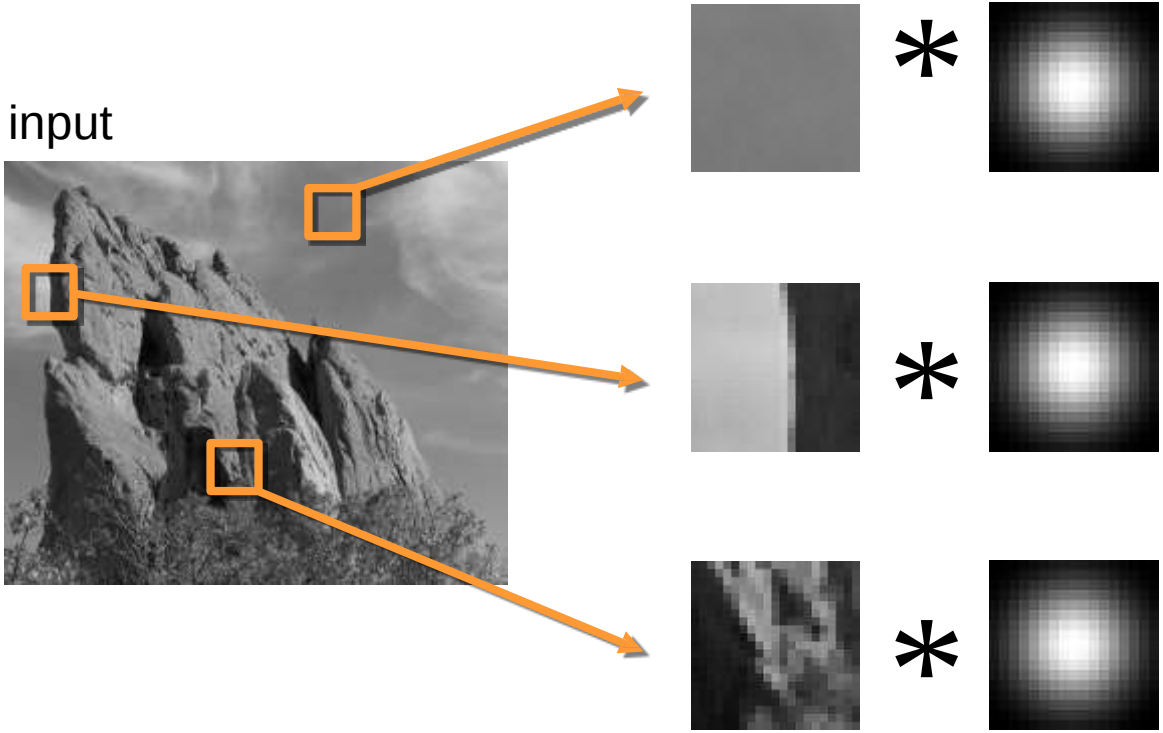
W_c

W_s



W_s*W_c

Usual Gaussian Filtering

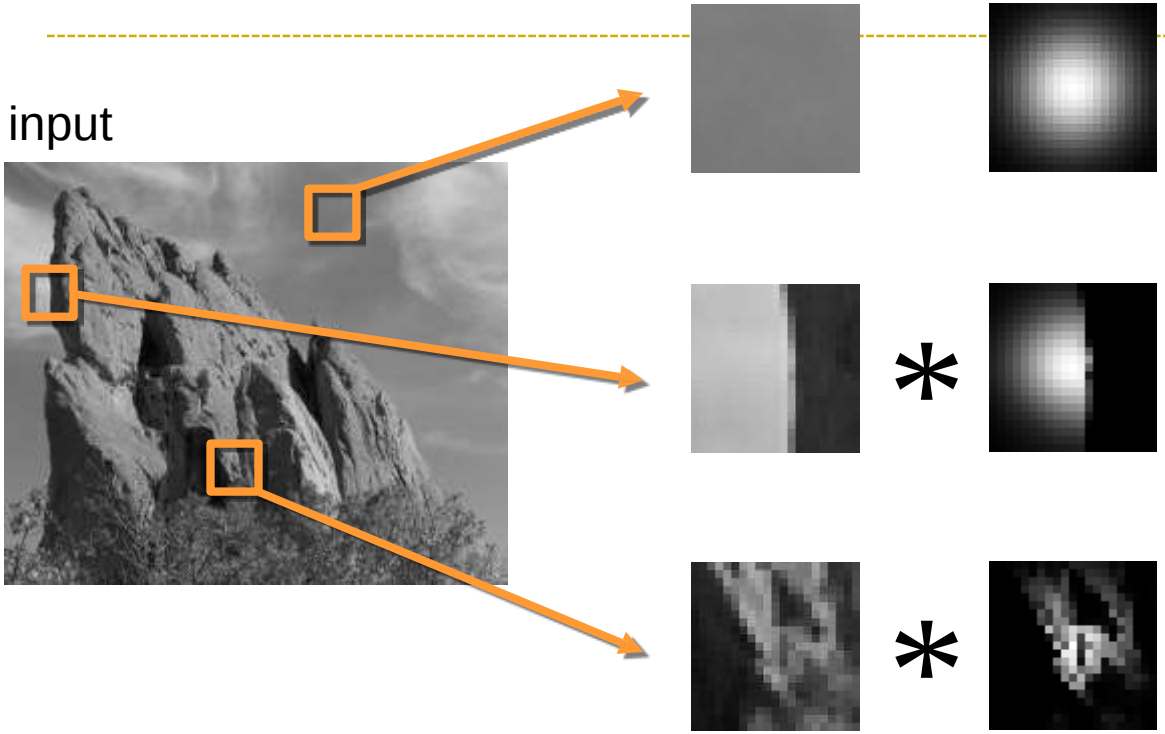


output

Same Gaussian kernel everywhere.

Source: Sylvain Paris

Bilateral Filtering



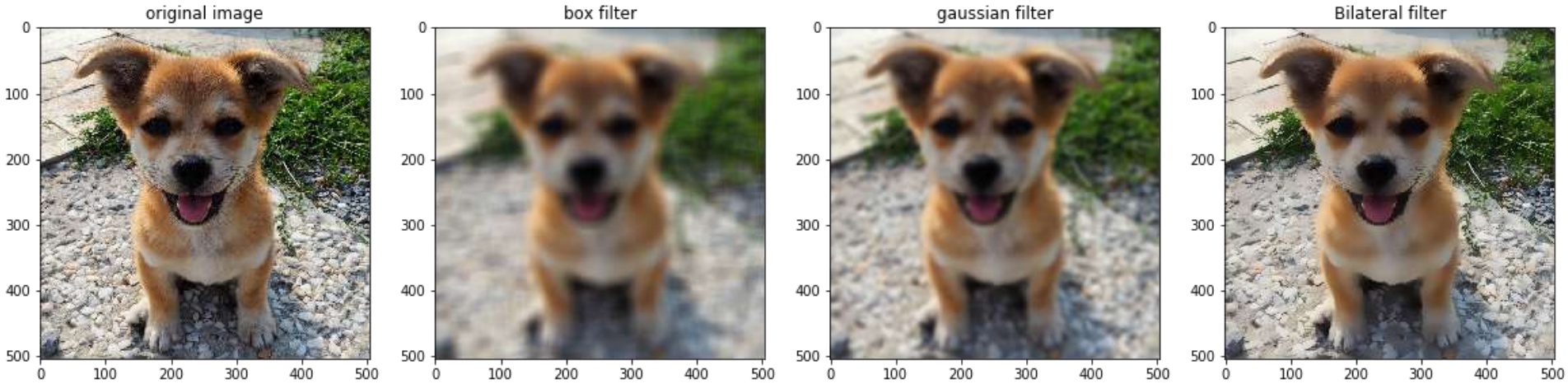
output

The kernel shape depends on the image content.

Source: Sylvain Paris

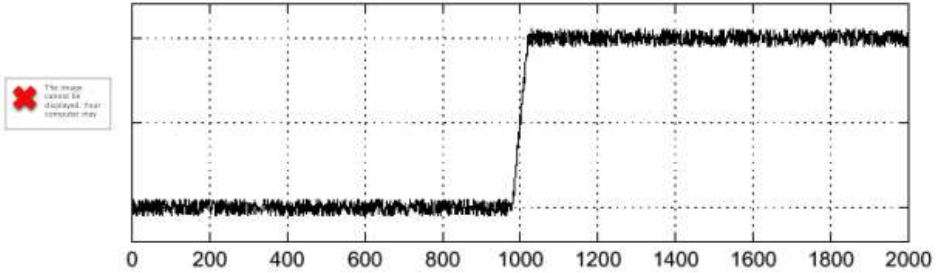
1 Iteration 2 Iterations 4 Iterations

Fig. 2.4 Iterations: the bilateral filter can be applied iteratively, and the result progressively approximates a piecewise constant signal. This effect can help achieve a limited-palette, cartoon-like rendition of images [72]. Here, $\sigma_s = 8$ and $\sigma_r = 0.1$.

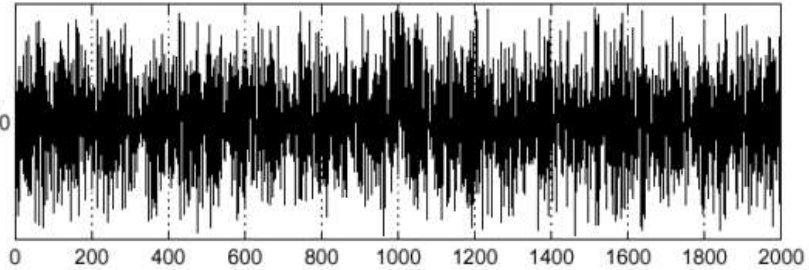

original image
box filter
gaussian filter
Bilateral filter
0
100
200
300
400
500

Effects of noise

Consider a single row or column of the image



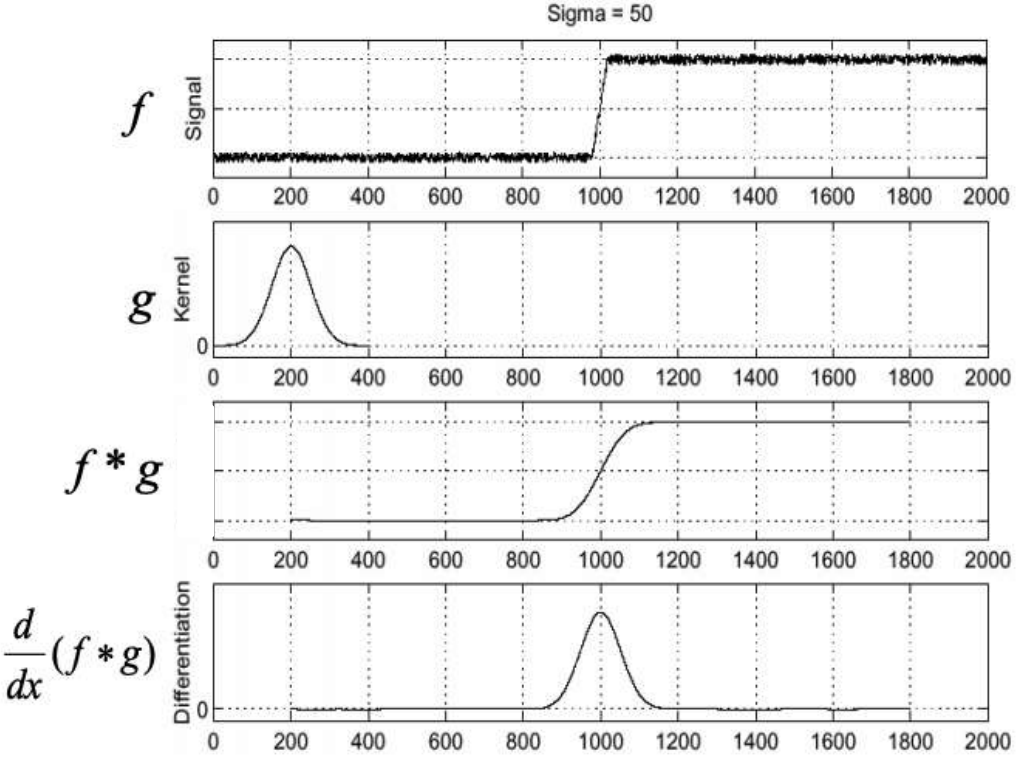
$\frac{d}{dx}f(x)$



Where is the edge?

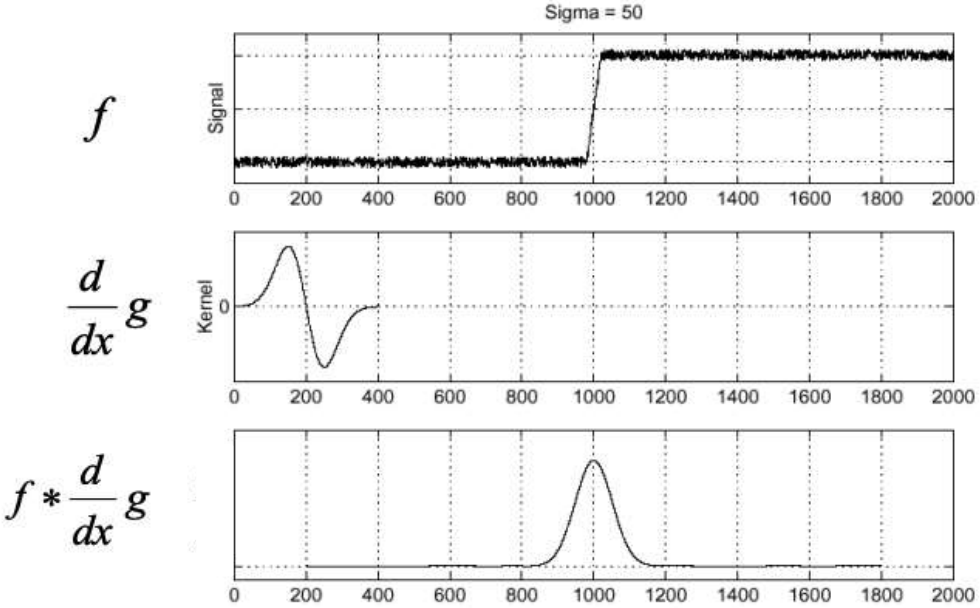
Source: S. Seitz

Solution: smooth first



$$\frac{d}{dx}(f * g) = f * \frac{d}{dx} g$$

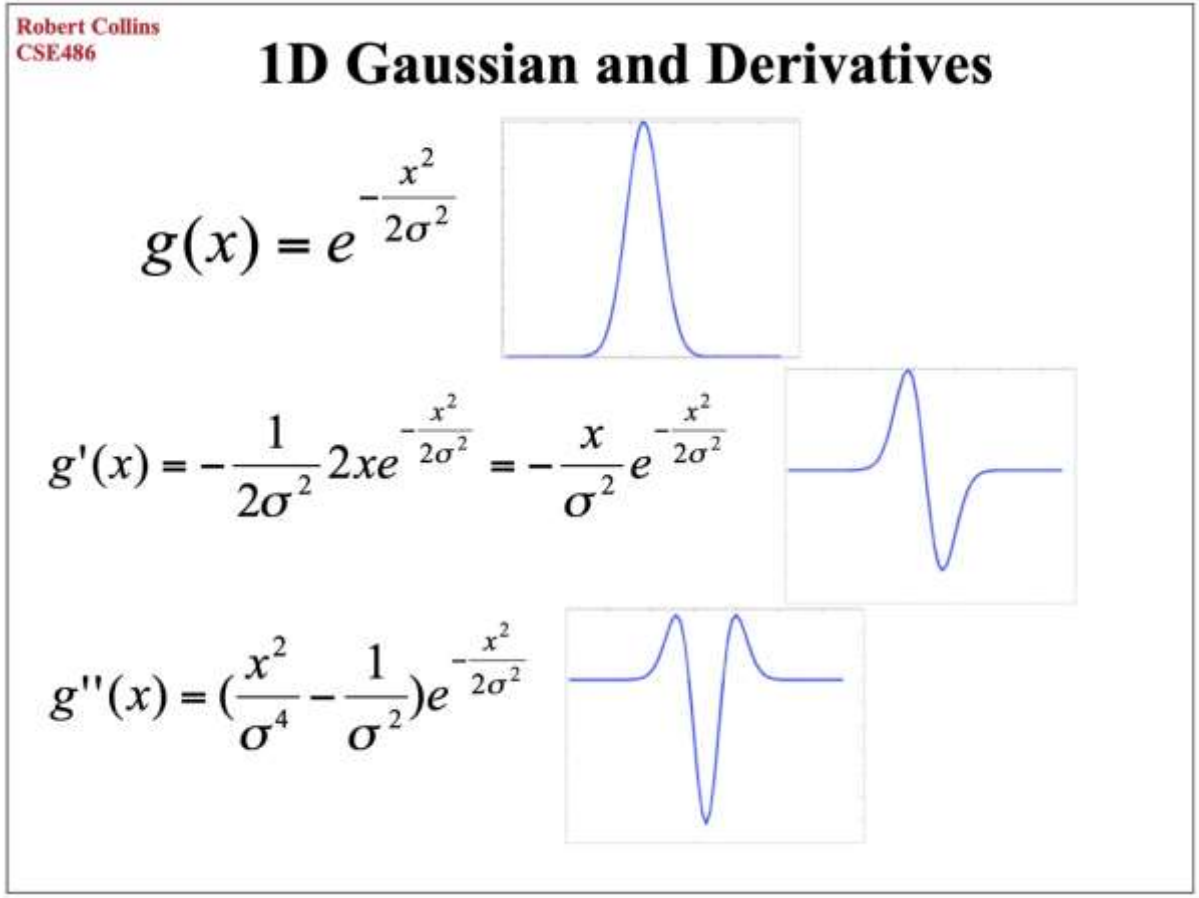
- This saves us one operation:



Source: S. Seitz

Other Important Filters

- Laplacian of Gaussian
  - Noise Suppression



Convolution / Linear Filters

- Smoothing (Average, Gaussian)
- Edge Filters (Prewitt, Sobel, Laplacian)

$$I'(u,v) \leftarrow \sum_{j=-1}^{1} \sum_{i=-1}^{1} I(u+i, v+j) \cdot H(i,j)$$

i \ j	-1	0	1
-1	a	b	c
0	d	e	f
1	g	h	i

120	190	140	150	200
17	21	30	8	27
89	123	150	73	56
10	178	140	150	18
190	14	76	69	87

x

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

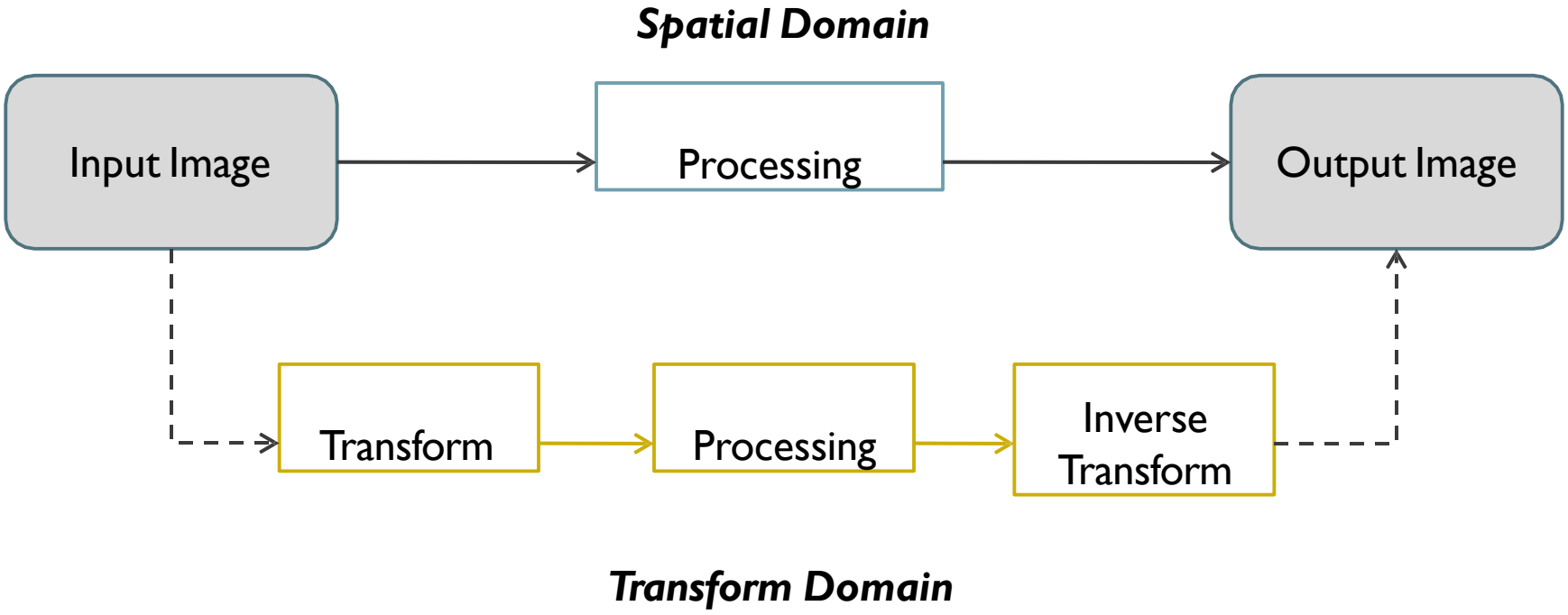
=

	98			

Image Processing – Two Paradigms

- Directly manipulating pixels in spatial domain
- Manipulating in transform domain

Spatial vs. Transform Domain Processing



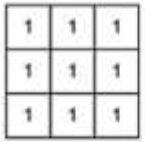
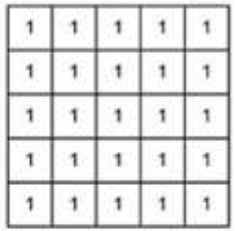
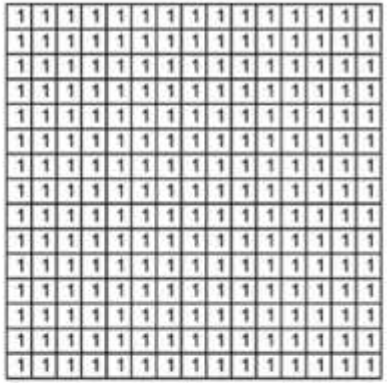
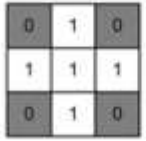
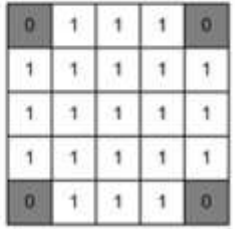
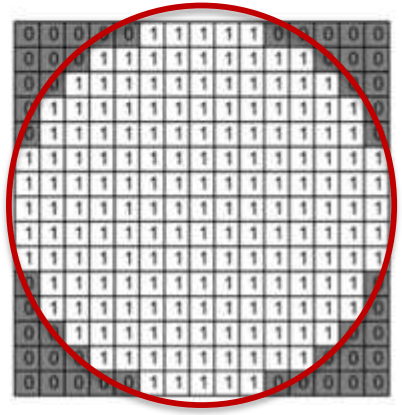
Introduction to Morphological Operators

Image – Set of Pixels

- Basic idea:
  - Object/Region = set of pixels (or coordinates of pixels)

- 0 = background
- 1 = foreground

Structuring Element

	3x3	5x5	15x15
Box			
Disc			

Structuring Element (Kernel)

- Can have varying sizes
- Have an origin
- Usually, element values are 0,1 and none(!)
  - For thinning, other values are possible
- Empty spots in the Structuring Elements are *don't care*'s!

Box

1	1	1
1	1	1
1	1	1

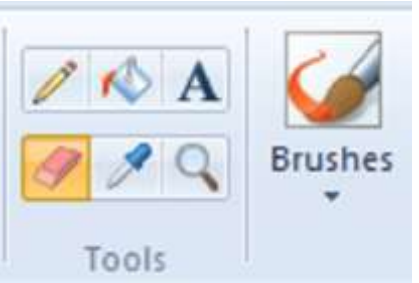
Disc

	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	1	1
1	1	1	1	1	1	1
	1	1	1	1	1	
		1	1	1		

1	1	
1	0	
1		0

1	1	1
1	10	1
1	1	1



Erosion

Erosion : Effect

1	1	1
1	1	1
1	1	1

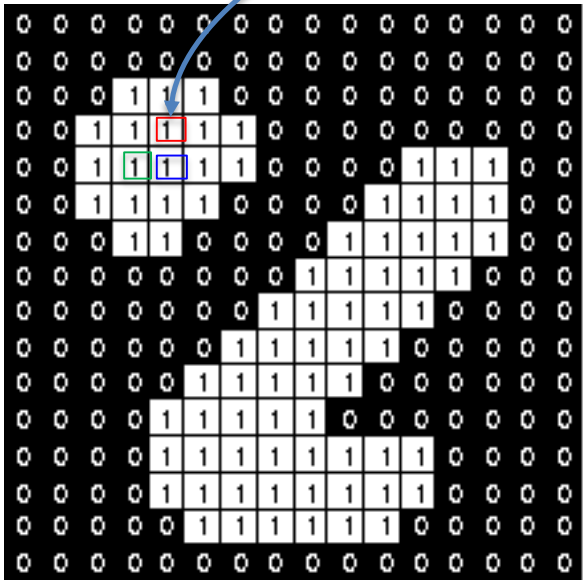
Set of coordinate points =

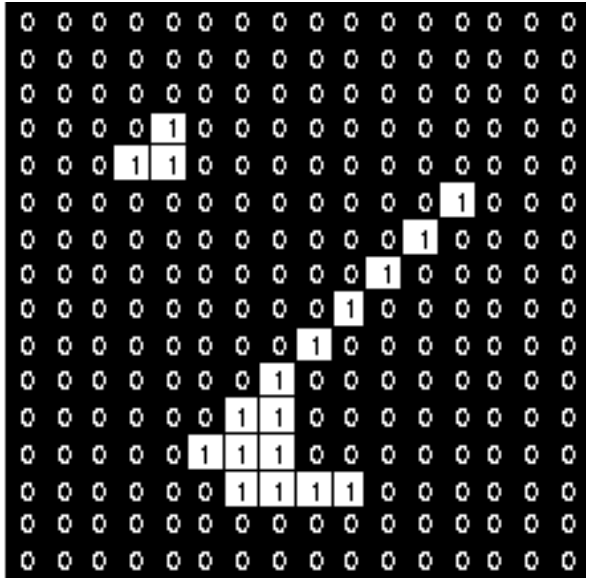
{ (-1, -1), (0, -1), (1, -1),

(-1, 0), (0, 0), (1, 0),

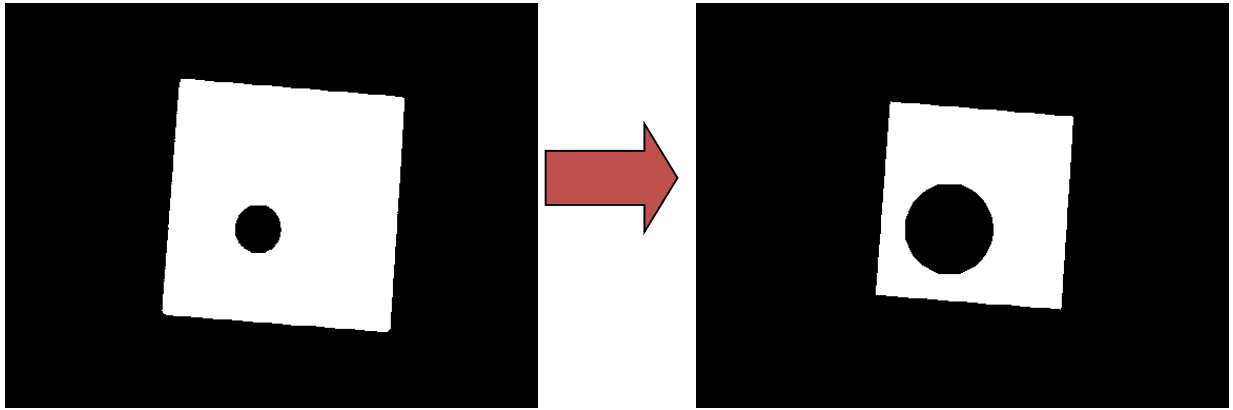
(-1, 1), (0, 1), (1, 1) }

If, for a particular location of Structuring Element (SE) origin, SE lies **fully within the region**, retain the location, else set to 0





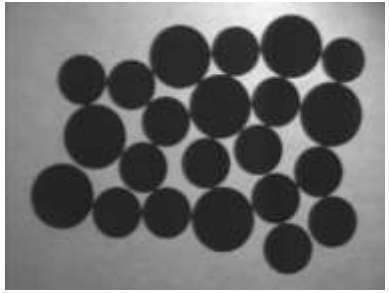
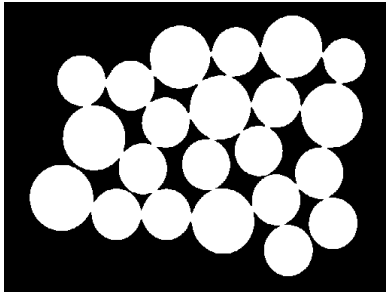
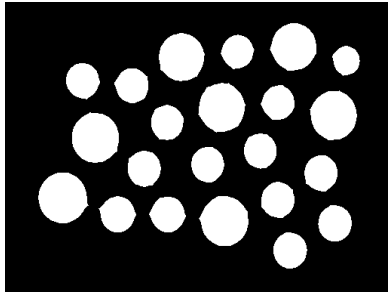
Another example of erosion



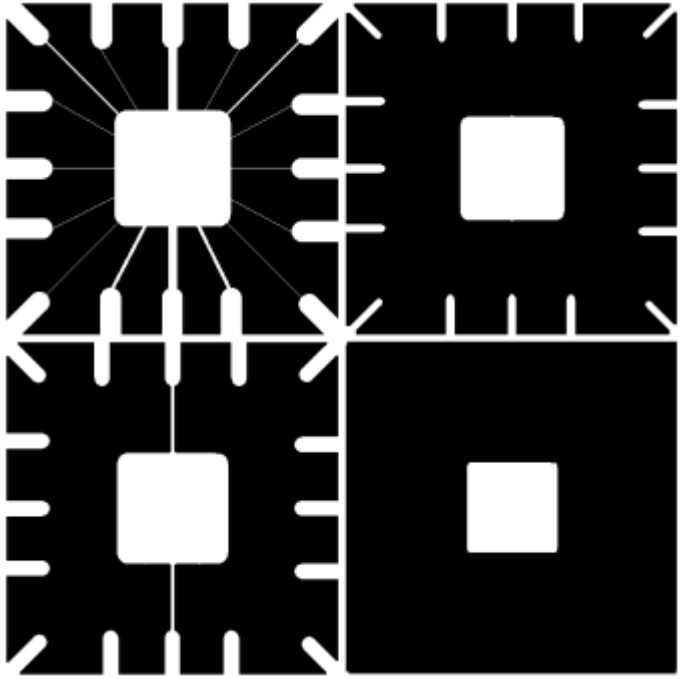
Erosion ➔ Image gets darker

Example: Counting coins

- Difficult because they touch each other!
- Solution: Binarization and Erosion separates them!

Erosion - example



a b
c d

FIGURE 9.8 An illustration of erosion.
(a) Original image.
(b) Erosion with a disk of radius 10.
(c) Erosion with a disk of radius 5.
(d) Erosion with a disk of radius 20.

From: Digital Image Processing, Gonzalez,Woods And Eddins

Erosion : Operation (min filter)

1	1	1
1	1	1
1	1	1

Set of coordinate points =

{ (-1, -1), (0, -1), (1, -1),

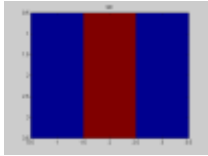
(-1, 0), (0, 0), (1, 0),

(-1, 1), (0, 1), (1, 1) }

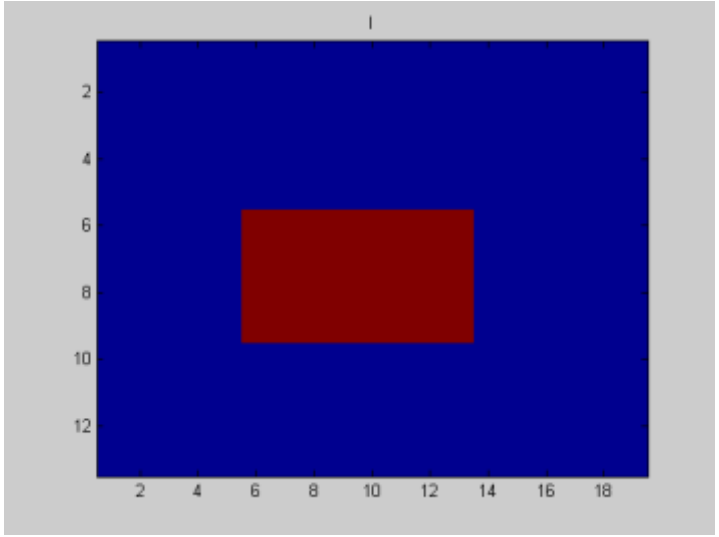
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	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	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	0	0	0
0	0	0	0	0	0	0	0	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	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0
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	0	0	1	0	0	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	0	0	1	0	0	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	0	0	0	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	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	0	0	0	0	0	0	0	0	0	0	0	0	0

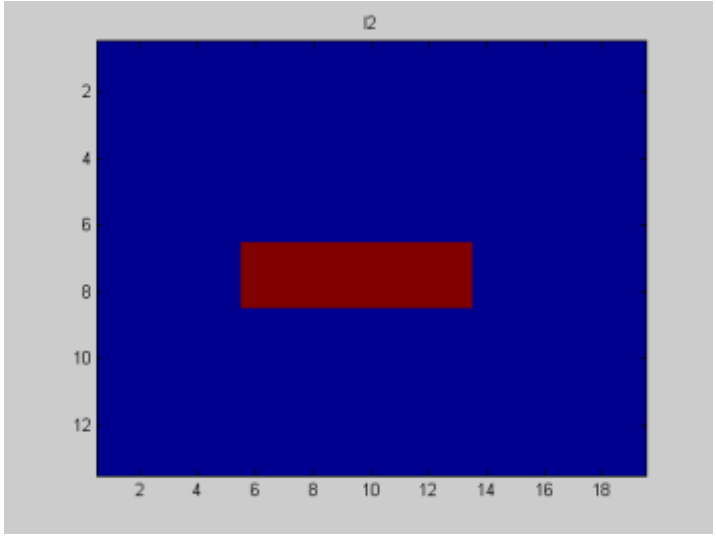
MATLAB code



SE = 3x3



I2



I3=imerode(I2,SE);

Erosion

- Shrinks foreground objects
- Foreground holes are enlarged
- Small (relative to structuring element size) foreground objects are removed.

Dilation

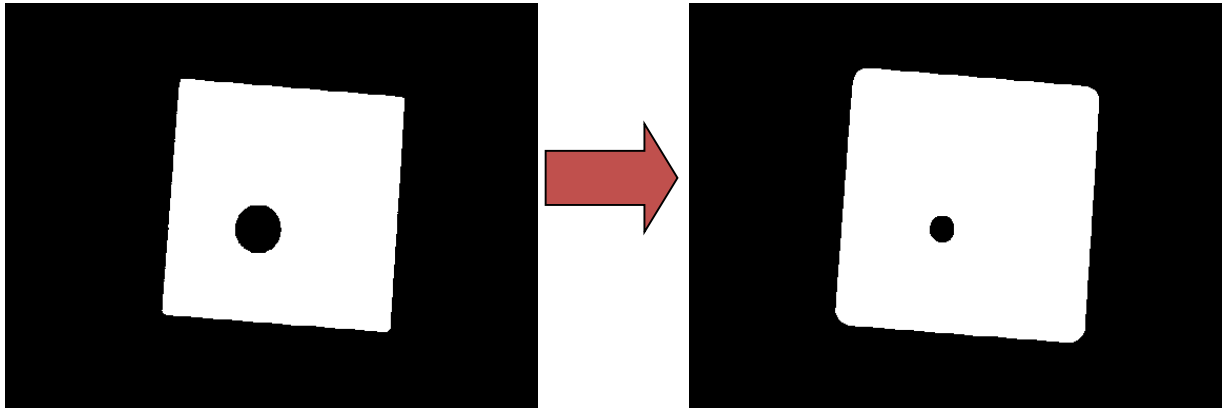
1	1	1
1	1	1
1	1	1

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	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\Rightarrow$

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1		0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0

Dilation Example



- Image gets lighter, more uniform intensity
- NOTE-1: SE = disk
- NOTE-2: Multiple iterations of dilation

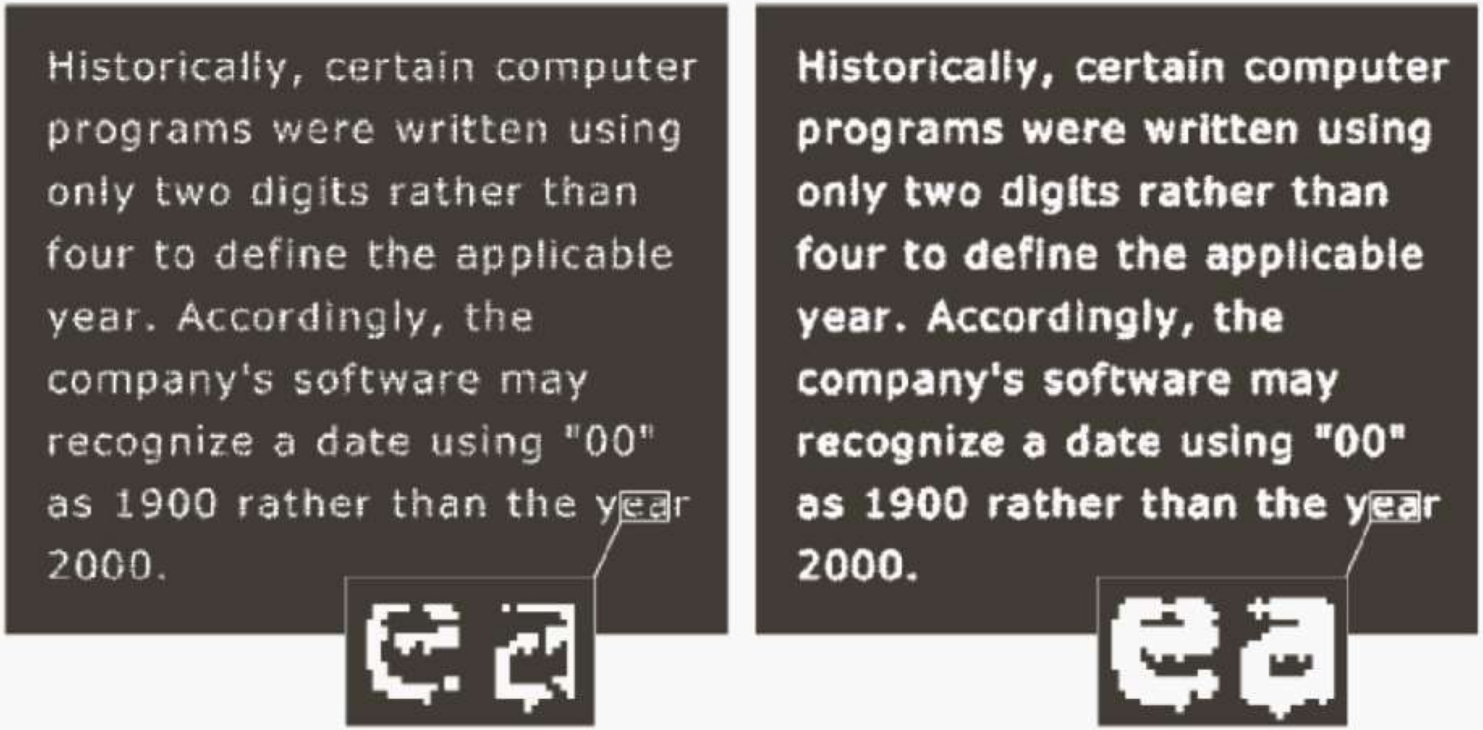
Dilation (max filter)

1	1	1
1	1	1
1	1	1

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	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

$\Rightarrow$

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	0	0	1	1	1	1	1	0
0	1	1	1	1	1	1	1	0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	1	1	1	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1		0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0



0	1	0
1	1	1
0	1	0

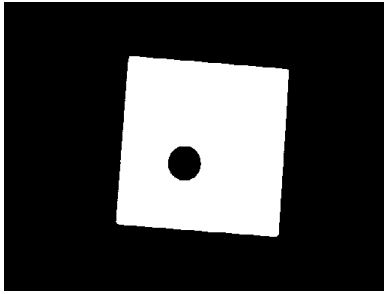
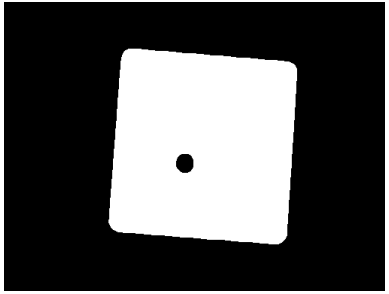
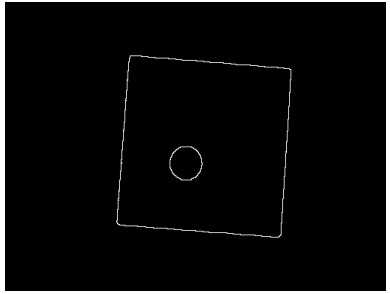
Figure 1: Image before (left) and after (right) dilation with the structuring element shown at the bottom

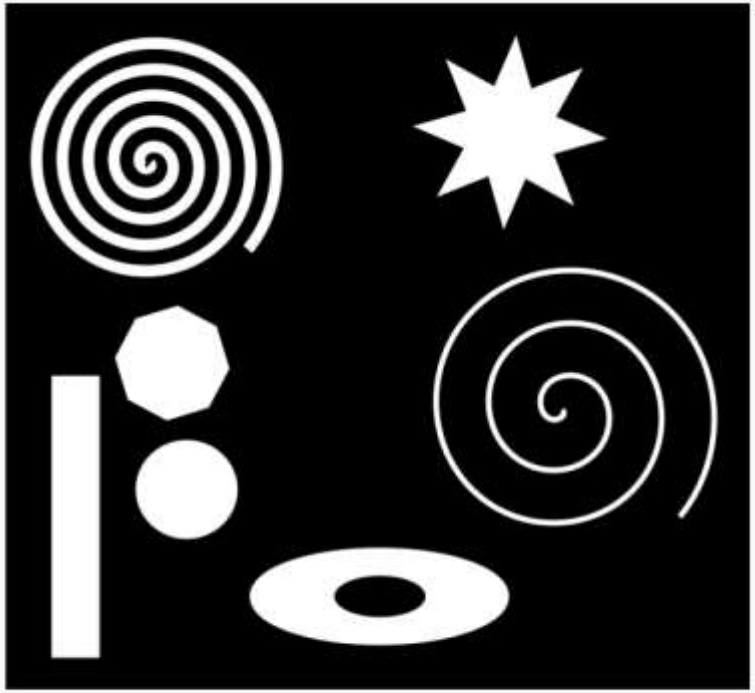
Dilation

- Expands foreground objects
- Foreground holes are shrunk

Boundary Detection

1. Dilate input image
2. Subtract input image from dilated image
3. Boundaries remain!

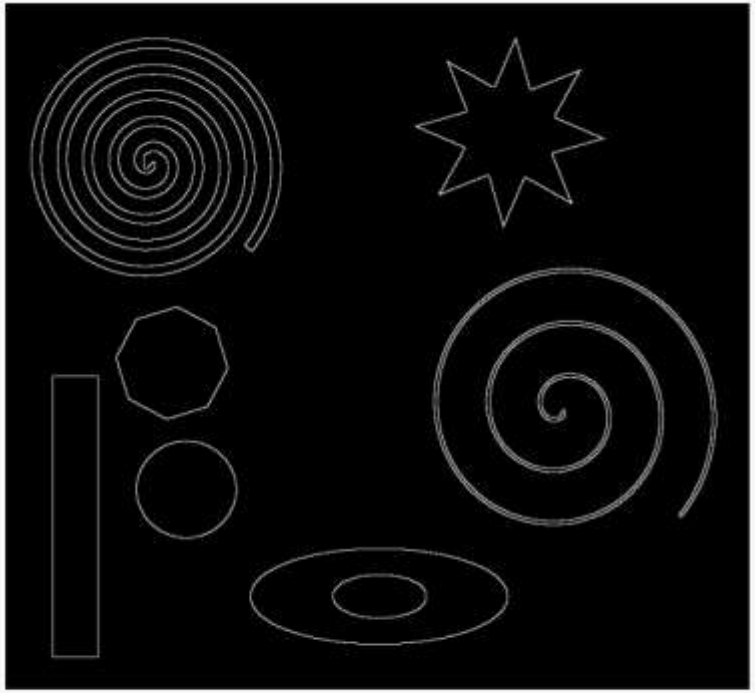






(a) f

1	1	1
1	1	1
1	1	1

(b) s



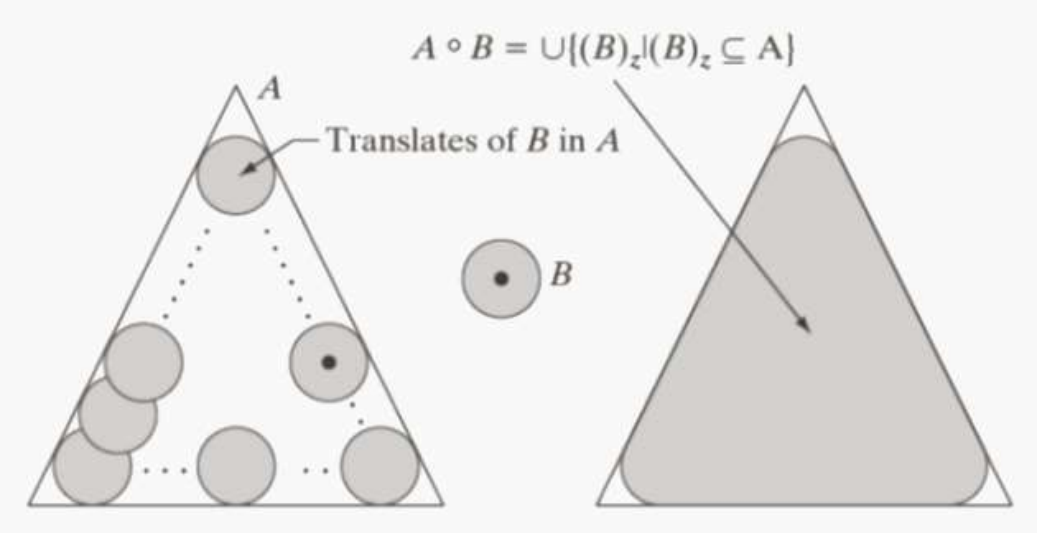
(c) $f - (f \ominus s)$

Opening and Closing

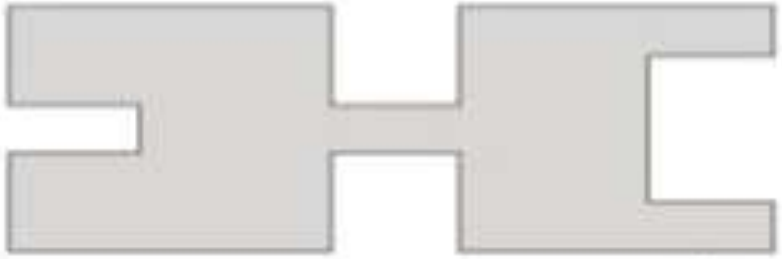
- Important operations
- Derived from the fundamental operations
  - Dilation
  - Erosion

Opening

- Take the structuring element (SE) and slide it around *inside* each foreground region.
  - All pixels which can be covered by the SE with the SE being entirely within the foreground region will be preserved.
  - All foreground pixels which can *not* be reached by the structuring element without lapping over the edge of the foreground object will be eroded away!

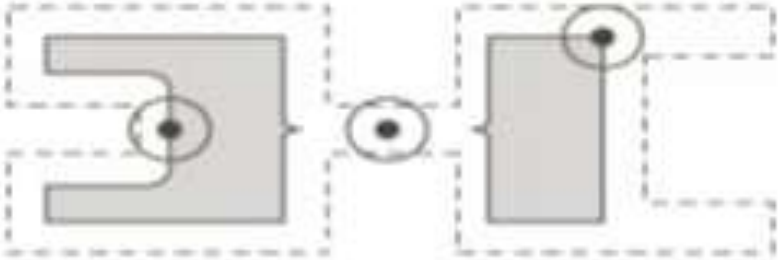


Opening



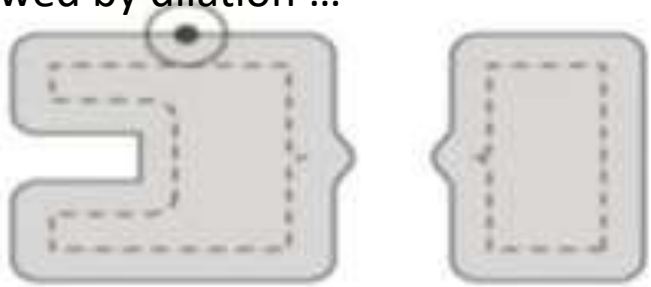
A

Erosion



$A \ominus B$

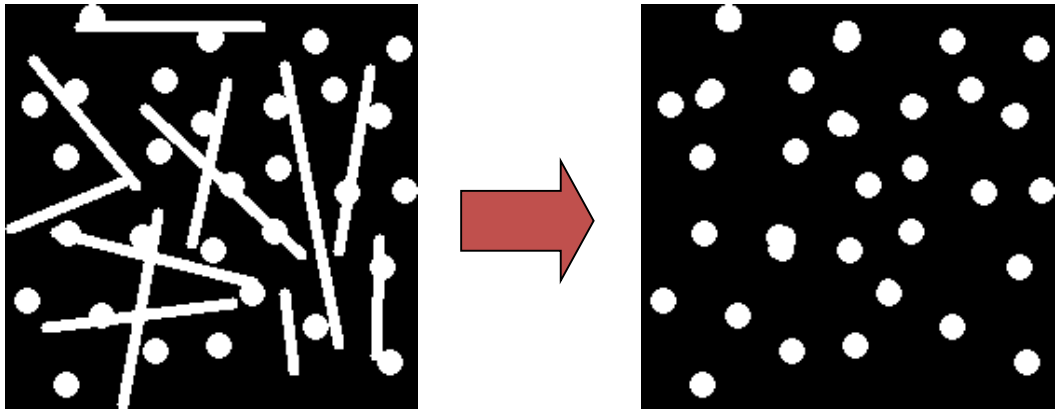
Followed by dilation …



$A \circ B = (A \ominus B) \oplus B$

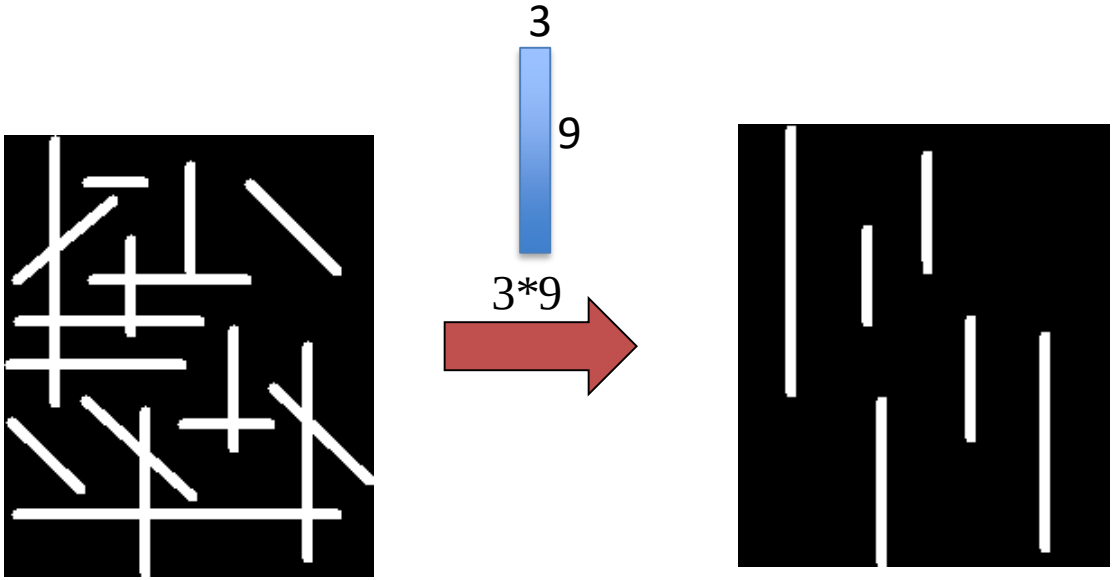
Opening: Example

- Opening with a 11 pixel diameter disc



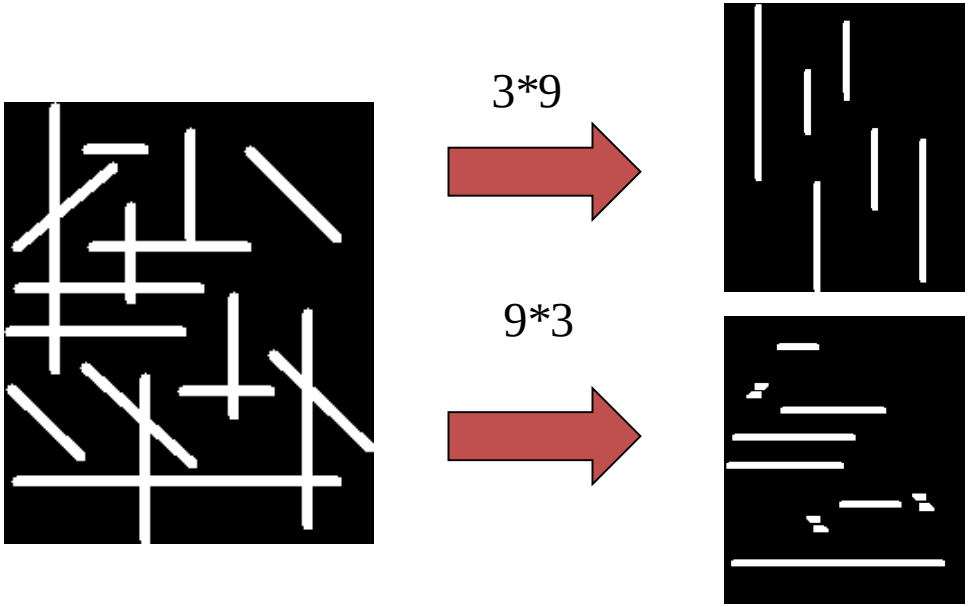
Opening: Another Example

- 3x9 Structuring Element

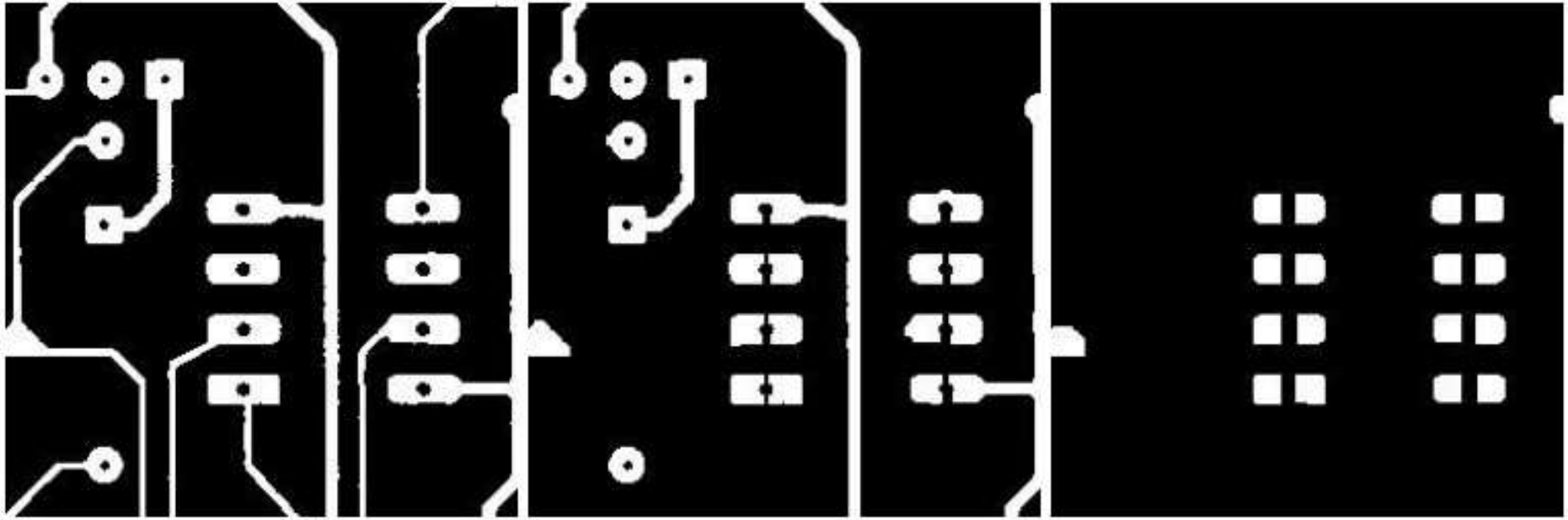


Opening: Another Example

- 3x9 and 9x3 Structuring Element



Opening

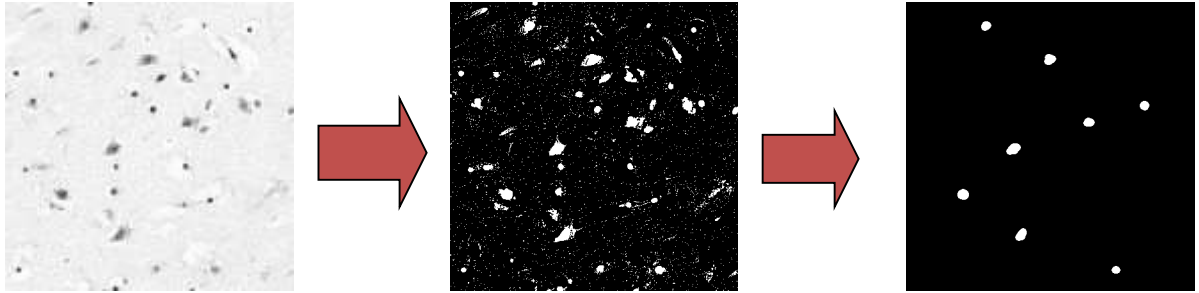


Binary image f $\quad$ $f \circ s$ (5×5 square) $\quad$ $f \circ s$ (9×9 square)

Results of opening with a square structuring element (www.mmorph.com/html/morph/mmopen.html/).

Use Opening for Separating Blobs

- Use large structuring element that fits into the big blobs
- Structuring Element: 11 pixel disc

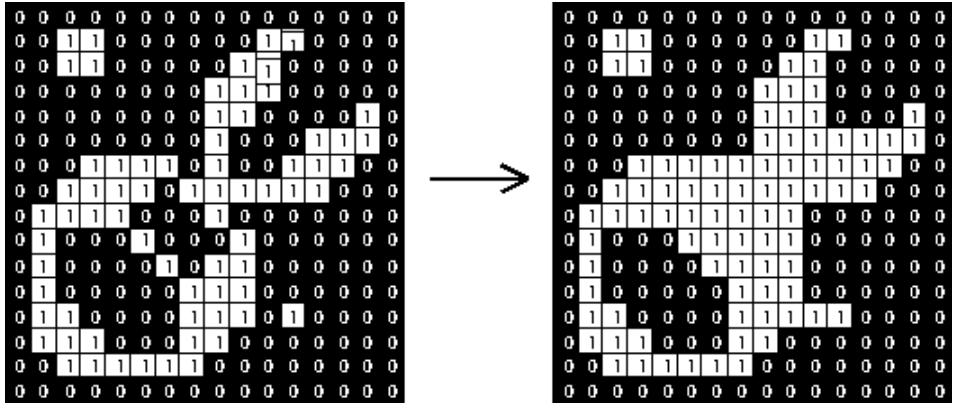


Opening

- Similar to Erosion
  - Spot and noise removal
  - Less destructive
- Erosion followed by Dilation
- the *same structuring element for both operations.*
- Opening is **idempotent:** Repeated application has no further effects!

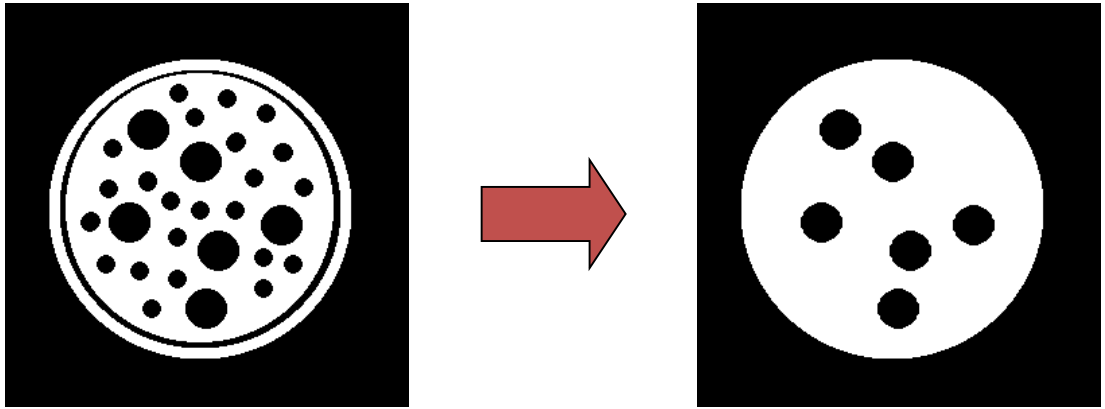
Closing (Dilation then Erosion)

- Structuring element: 3x3 square



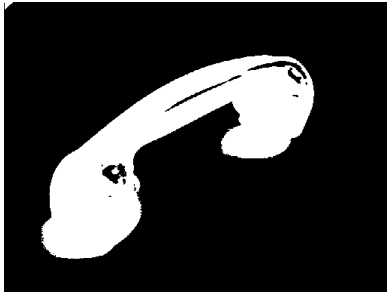
Closing: Example

- Closing operation with a 22 pixel disc
- Closes small holes in the foreground

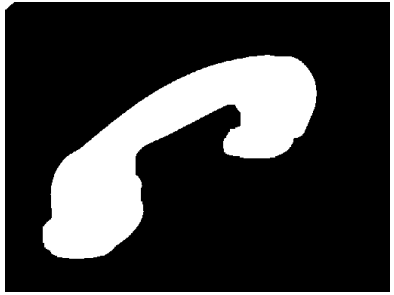


Closing Example 1

1. Threshold
2. Closing with disc of size 20



Thresholded



closed

Fingerprint problem

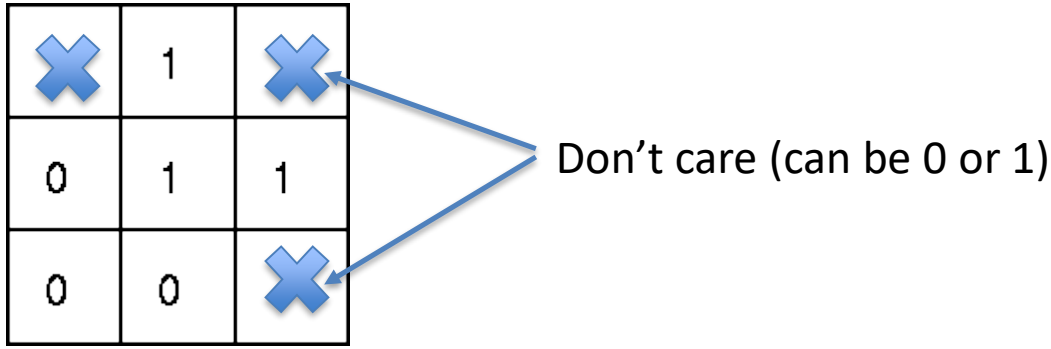
a b c

FIGURE 9.11 (a) Noisy fingerprint image. (b) Opening of image. (c) Opening followed by closing. (Original image courtesy of the National Institute of Standards and Technology.)

From: Digital Image Processing, Gonzalez,Woods And Eddins

Hit-and-miss Transform

- Look for particular patterns of foreground and background pixels.



- If matched, set pixel = 1

Example: Find right-angled convex corners

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	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	1	1	1	1	0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	0	0	1	1	1	1	1	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	0	0	0	0	0	1	1	1	1	0	0	0
0	0	1	1	0	0	0	0	0	1	1	1	1	0	0	0
0	0	0	1	0	0	0	0	0	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	0	0	0	0	0
0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0
0	0	1	1	1	1	1	1	1	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	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	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	0	0	0	0	1	0	0	0	1	0	0	0
0	0	1	0	0	1	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	0
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	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
0	0	1	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	1	0	0	0
0	0	1	0	0	0	0	0	0	0	0	1	0	0	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	0	0	1	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	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	0	0	0	0	0	0	0	0	0	0	0	0	0	0

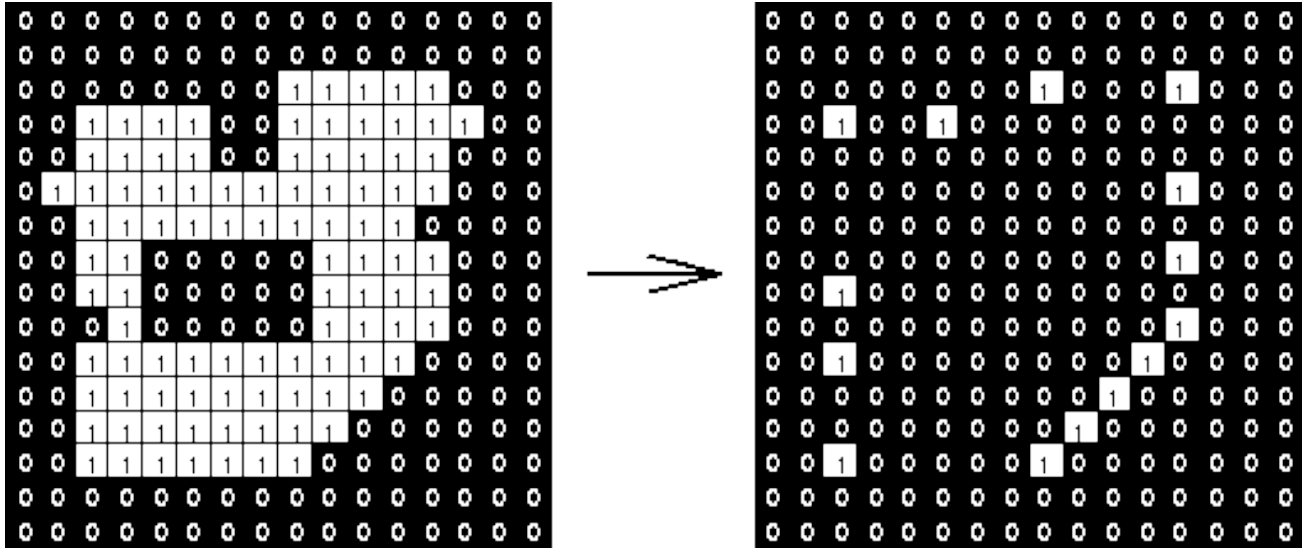
Example: Find right-angled convex corners

	1	
0	1	1
0	0	

	1	
1	1	0
	0	0

	0	0
1	1	0
	1	

0	0	
0	1	1
	1	



Sample HAM transforms

1)

0	0	0
0	1	0
0	0	0

Locate isolated points

Sample HAM transforms

1)

0	0	0
0	1	0
0	0	0

Locate isolated points

2)

0	1	0
0	0	0

Locate end points on thin lines

Sample HAM transforms

1)

0	0	0
0	1	0
0	0	0

Locate isolated points

2)

0	1	0
0	0	0

Locate end points on thin/tapering structures

3a)

	1	
	1	
1		1

3b)

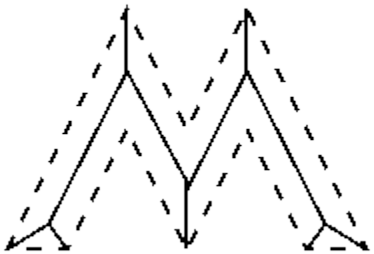
1		
	1	
1		1

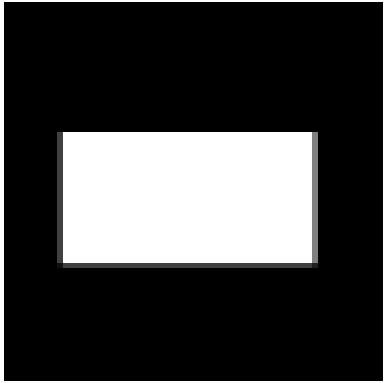
3c)

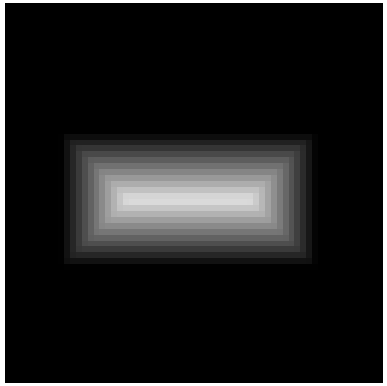
*	0	1
1	1	0
*	1	*

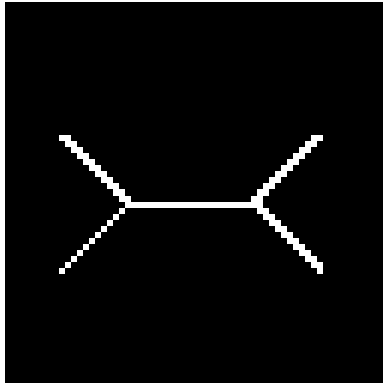
Locate triple junctions

Skeletonization

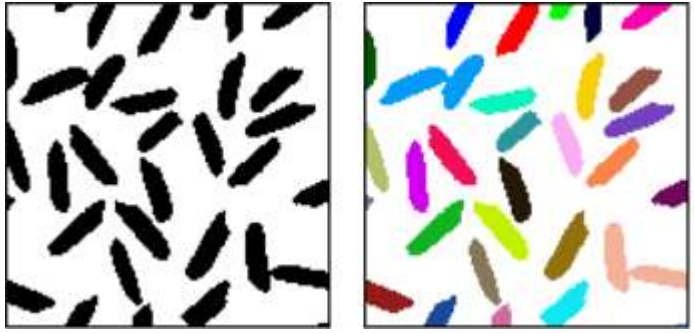


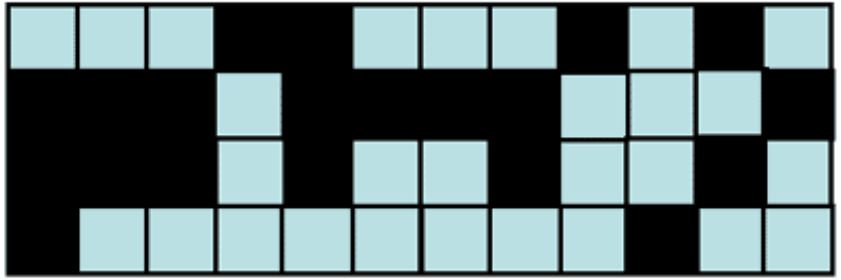




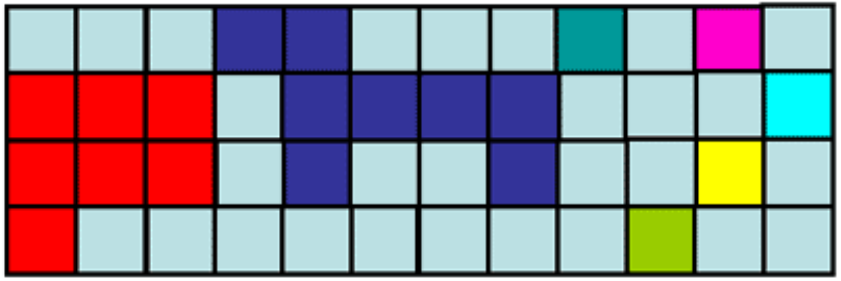


Components

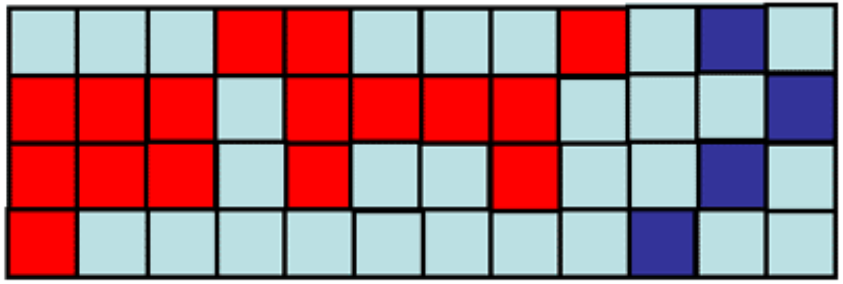




Binary image:
0 - objects;
1 -background

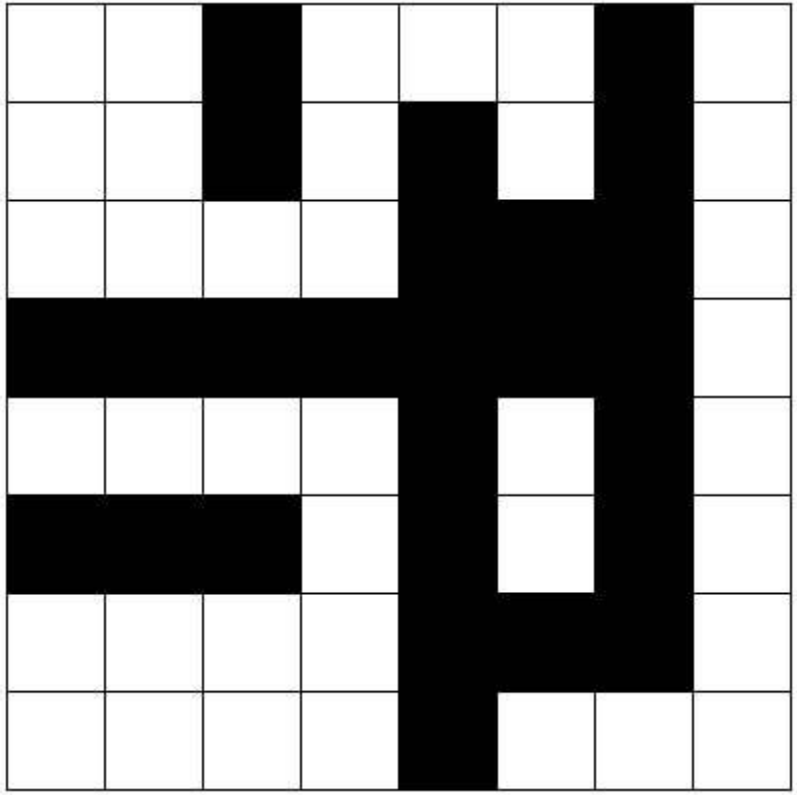


4-connected
objects +
8-connected
background

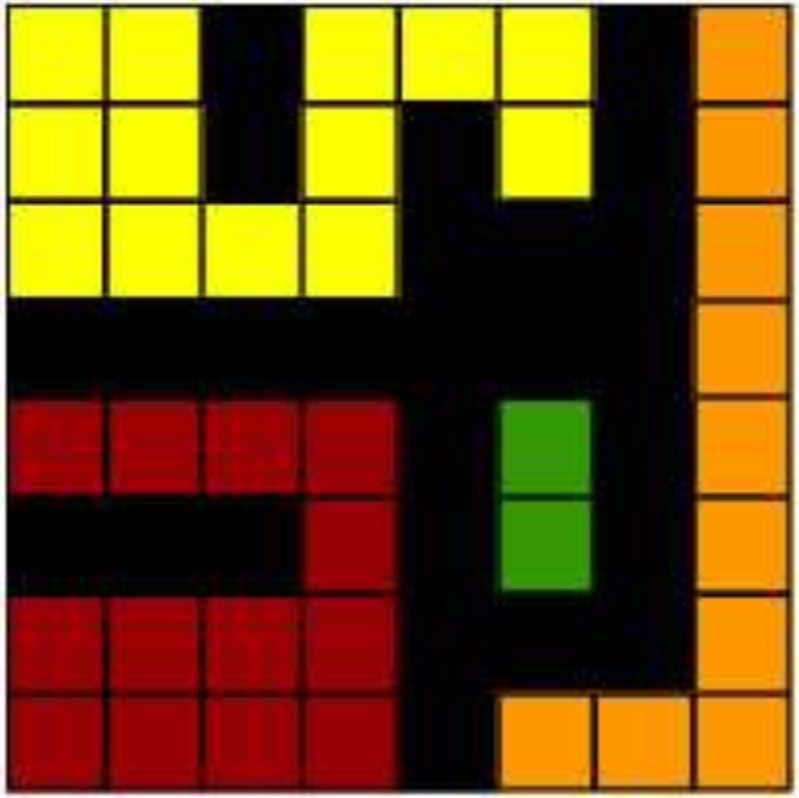


8-connected
objects +
8-connected
background

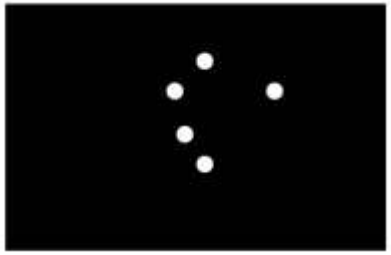
Two-Pass Algorithm for Connected Component Labelling



Two-Pass Algorithm for CCL

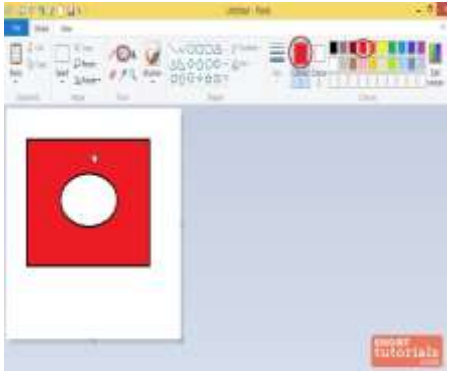


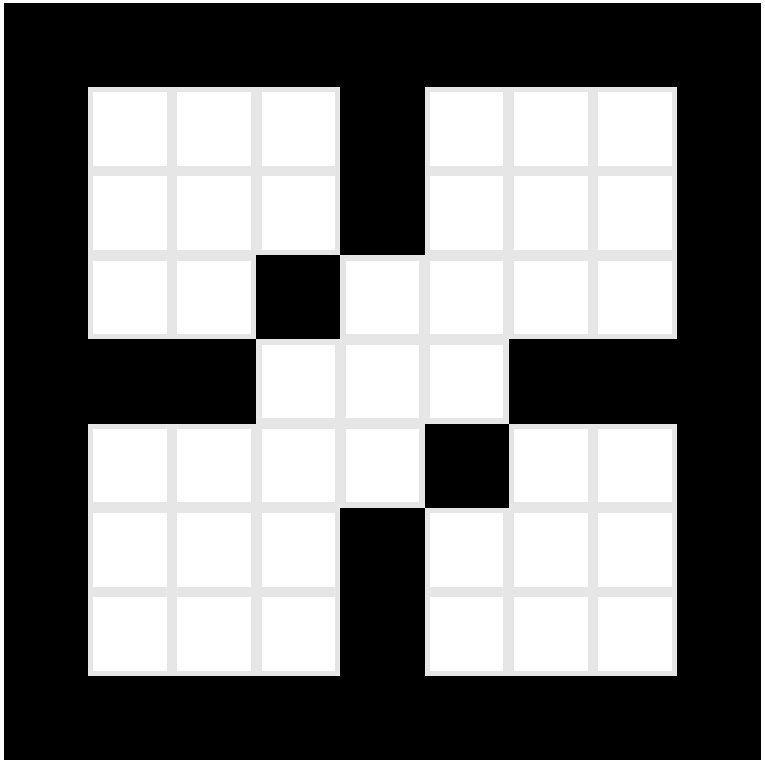
Connected Component Labeling



`bwlabel` `label2rgb`

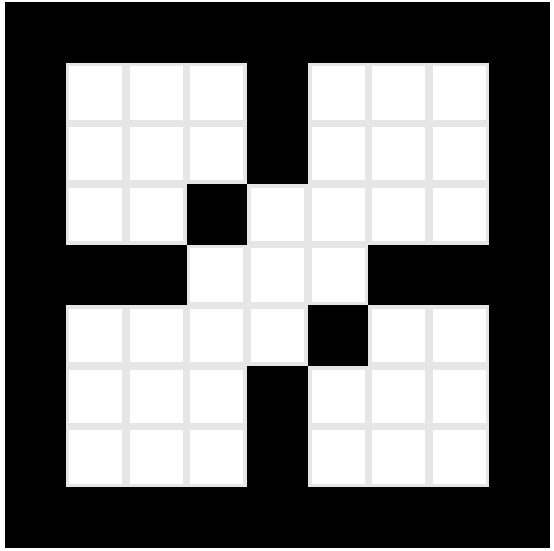
Flood-Fill





https://upload.wikimedia.org/wikipedia/commons/7/7e/Recursive_Flood_Fill_4_%28aka%29.gif

Flood-Fill Algorithm (4-conn)

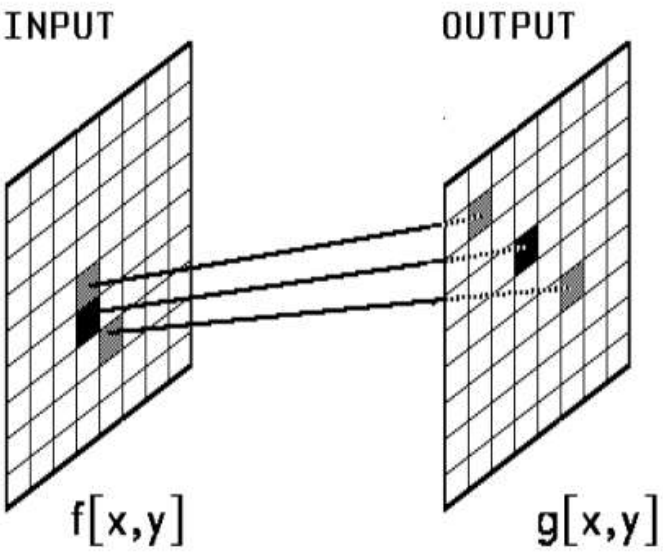


```
void floodFill(int x, int y, int fill, int old)
{
    if ((x < 0) || (x >= width)) return;
    if ((y < 0) || (y >= height)) return;
    if (getPixel(x, y) == old) {
        setPixel(fill, x, y);
        floodFill(x+1, y, fill, old);
        floodFill(x, y+1, fill, old);
        floodFill(x-1, y, fill, old);
        floodFill(x, y-1, fill, old);
    }
}
```

Many/More efficient versions exist !

Geometric Operations

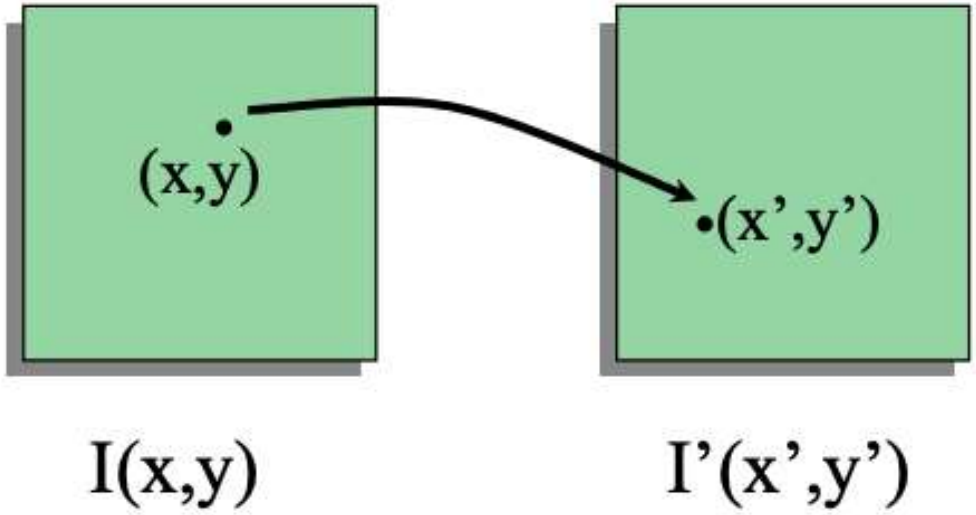
- Filters, Point Operations
  - modify color values (range) of pixels
  - domain (x,y) remains (mostly) fixed
- **Geometric transformation**
  - modify the positions of pixels
  - .. but keep their colors (mostly) unchanged



$$x \rightarrow f_x(x, y) = x'$$

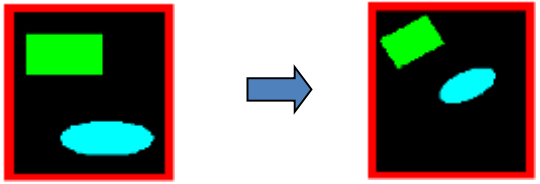
$$y \rightarrow f_y(x, y) = y'$$

$$I(x, y) = I'\left(f_x(x, y), f_y(x, y)\right)$$



Geometric Operations

- **Scale** - change image content size
- **Rotate** - change image content orientation
- **Reflect** - flip over image contents
- **Translate** - change image content position
- **Affine Transformation**
  - general image content linear geometric transformation

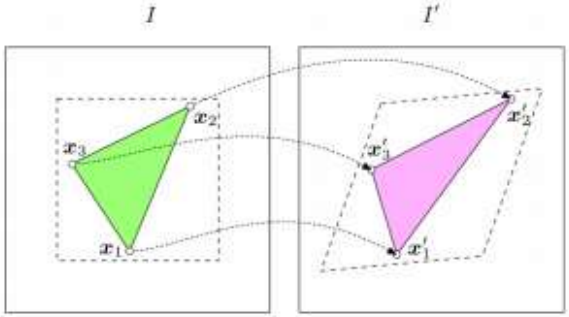


Affine (3-Point) Mapping

- Can use homogeneous coordinates to rewrite translation, rotation, scaling, etc as vector-matrix multiplication

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- **Affine mapping:** Can then derive values of matrix that achieve desired transformation (or combination of transformations)

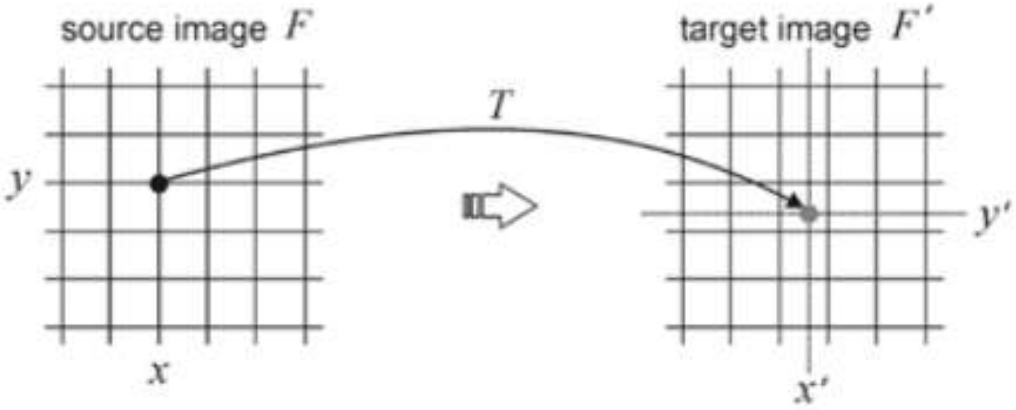


- Inverse of transform matrix is **inverse mapping**

https://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/slides/lecture11.pdf

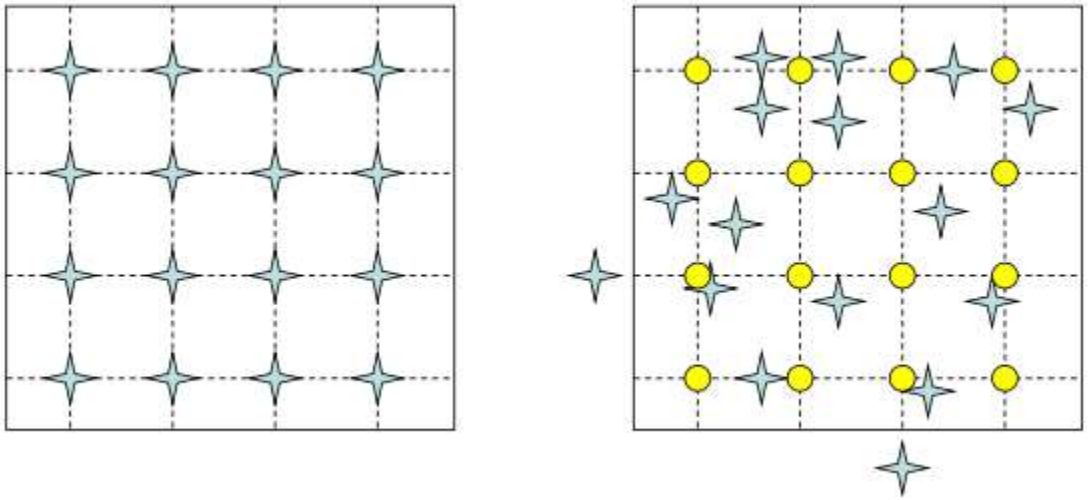
Interpolation methods

- Forward mapping



https://www.d.umn.edu/~dzhou/ME4060/ME%204060%20Chapter%207-1%20Geometria%20operation.pdf

Forward Warping



Warping points are often non-integer samples

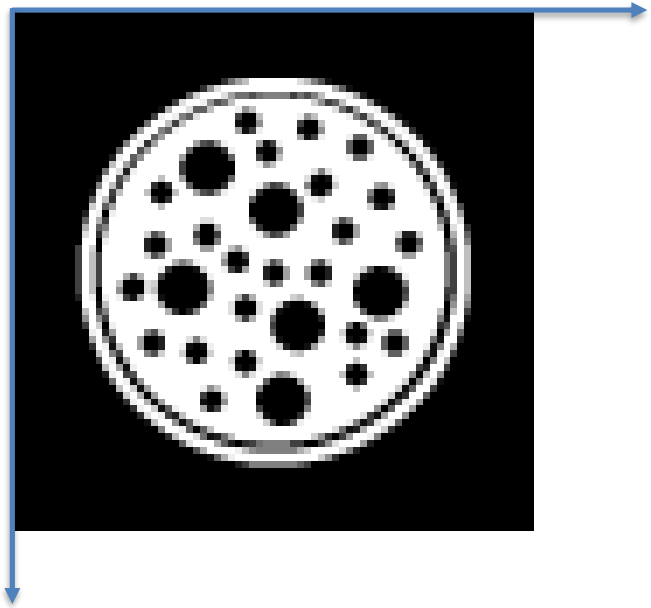
Many integer samples "o" are not assigned Values

http://eeweb.poly.edu/~yao/EL5123/lecture12_ImageWarping.pdf

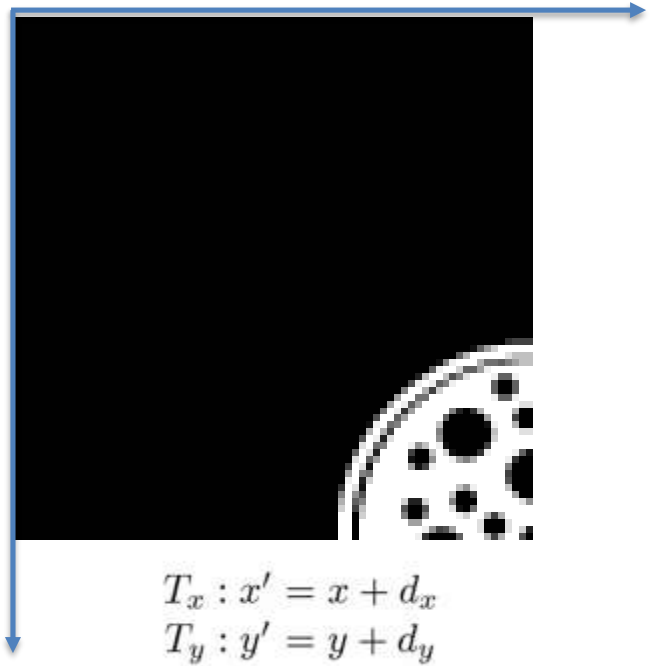
Translation

300

300



Translation by (150,150)



$$T_x : x' = x + d_x$$
$$T_y : y' = y + d_y$$

- Translation by (30.7, 30.7) ?

Two issues

- Two issues:
  - Dimensionality: The destination image may not be large enough to contain all of the processed samples
  - Discretization: Transformed locations are not integers: How can we place a source sample at a non-integer location in the destination?

(a) Source image. (b) Rotation. (c) Expanded view of rotated image.

Figure 7.2. Destination dimensionality under rotation.

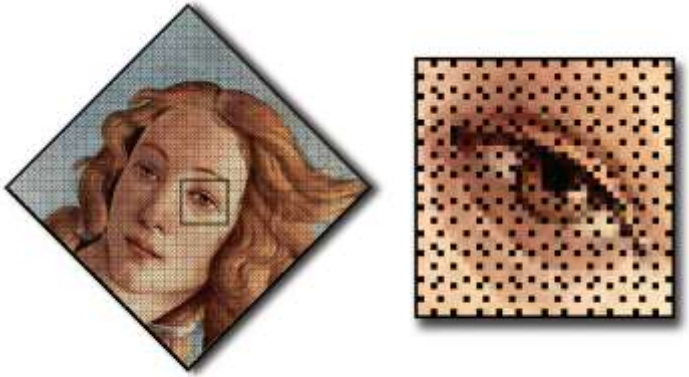
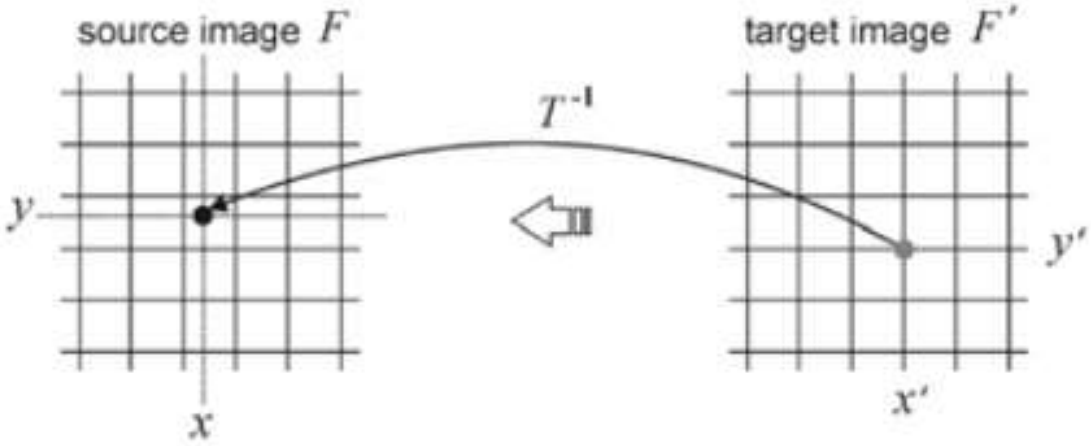


Figure 7.3. Rotation by forward mapping.

Solution: Backward mapping

Interpolation methods

- Backward mapping



- An empty destination image is created.
- Each location in the destination is mapped backwards onto the source.
- Source location may not be integer-valued coordinates.
- Sample value is obtained via interpolation.

T : Affine transform matrix

v : location in the destination image

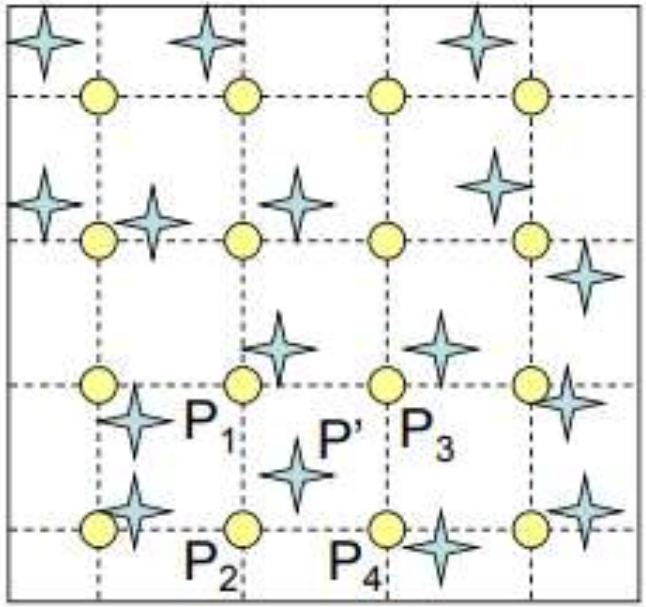
$v' = [x', y', 1]^T$

$$v' = Tv$$

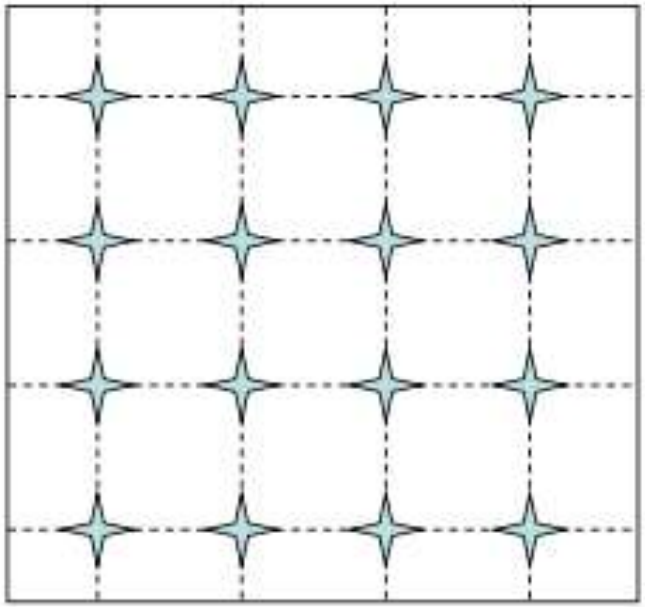
$$v = T^{-1}v'$$

https://www.d.umn.edu/~dzhou/ME4060/ME%204060%20Chapter%207-1%20Geometria%20operation.pdf

Inverse Warping

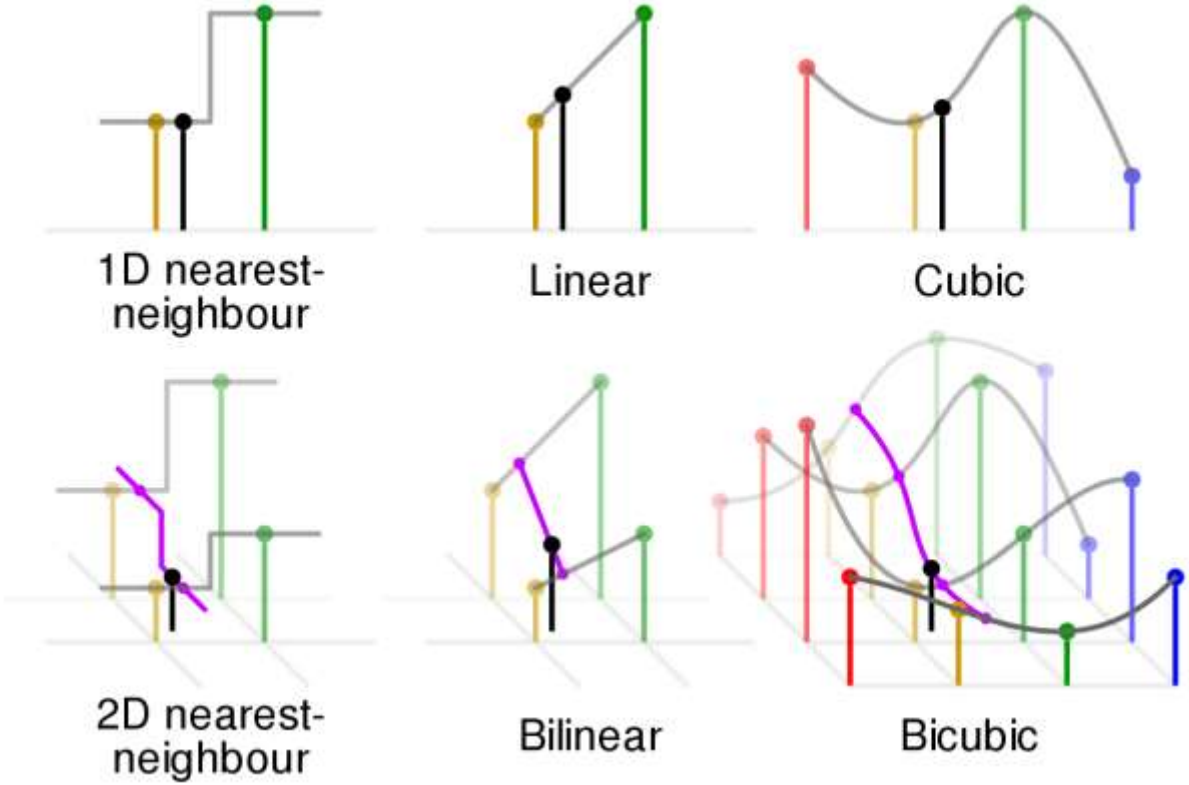


P' will be interpolated from P_1, P_2, P_3, and P_4

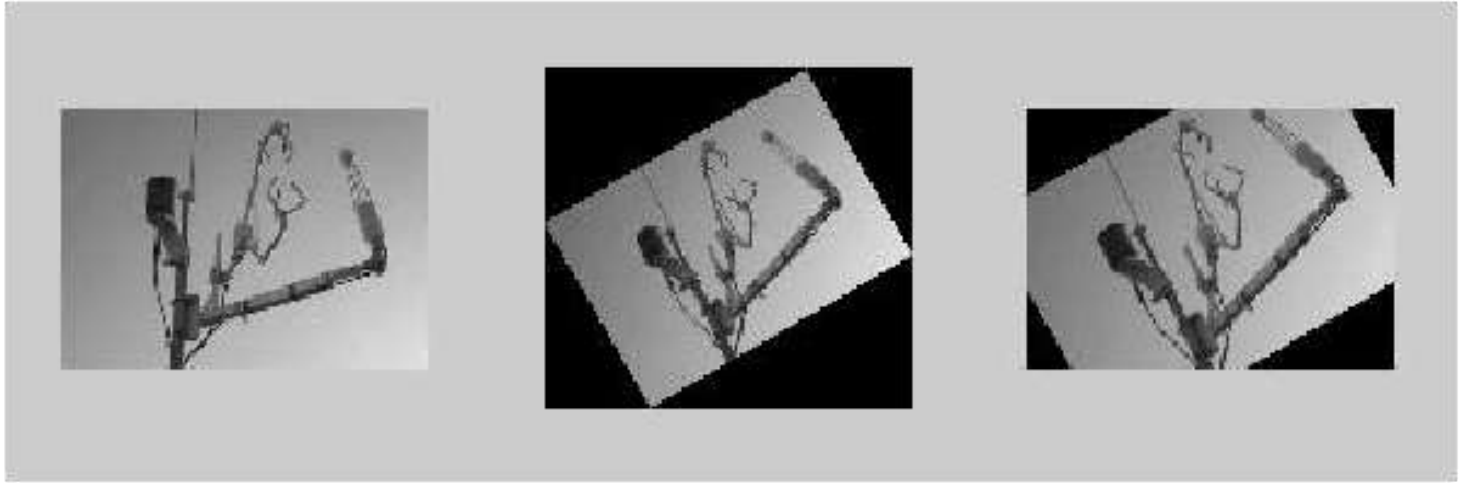


http://eeweb.poly.edu/~yao/EL5123/lecture12_ImageWarping.pdf

Interpolation Function



Example: `imrotate`



I

J = imrotate(I,30,`loose’)

J = imrotate(I,30,`crop’)

NOTE: By default, rotation is anticlockwise & about center of image

- MATLAB’s T = `imregtform()` rotates about top-left corner of image

B translation

B rotation

$$\text{Translation:} \quad x(k,l) = k + 50; y(k,l) = l;$$

$$\text{Rotation:} \quad x(k,l) = (k - x_0)cos(\theta) + (l - y_0)sin(\theta) + x_0;$$

$$y(k,l) = -(k - x_0)sin(\theta) + (l - y_0)cos(\theta) + y_0;$$

$x_0 = y_0 = 256.5$ the center of the image $\mathbf{A}$, $\theta = \pi/6$

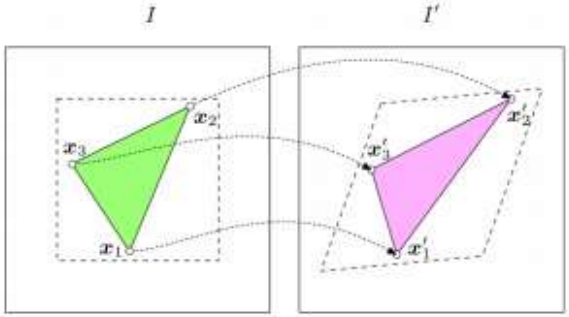
By Onur Guleyuz

Affine (3-Point) Mapping

- Can use homogeneous coordinates to rewrite translation, rotation, scaling, etc as vector-matrix multiplication

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- **Affine mapping:** Can then derive values of matrix that achieve desired transformation (or combination of transformations)



- Inverse of transform matrix is **inverse mapping**

https://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/slides/lecture11.pdf

WARP

SWIRL

WARP $x(u,v) = sign(u - x_0) * (u - x_0)^2 / x_0 + x_0; y(u,v) = v$

SWIRL

$$x(u,v) = (u - x_0)\cos(\theta) + (v - y_0)\sin(\theta) + x_0;$$
$$y(u,v) = -(u - x_0)\sin(\theta) + (v - y_0)\cos(\theta) + y_0;$$
$$r = ((u - x_0)^2 + (v - y_0)^2)^{1/2}, \theta = \pi r / 512.$$

By Onur Guleyuz

Geometric Transformation: Perspective

- Two aspects
  - Mapping (Type of transform)
  - Interpolation (Quality of transform)

Examples of Image Morphing

Cross Dissolve

I(t) = (1-t)*S+t*T

Mesh based

George Wolberg, "Recent Advances in Image Morphing", *Computer Graphics Intl. '96*, Pohang, Korea, June 1996.

http://eeweb.poly.edu/~yao/EL5123/lecture12_ImageWarping.pdf

Geometric Transforms and Registration

- Given: T (transformation)
- Determine: Effect of T on source image I, get target image O
- Variant:
  - Given : Source image I, Target image O
  - Determine : Transformation T

Image Registration

Aerial Photo Image

Orthophoto Image

https://in.mathworks.com/help/images/register-an-aerial-photograph-to-a-digital-orthophoto.html

Geometric Operations

- Some uses
  - Correct distortions introduced during imaging
  - Transformation: To create special effects (e.g. morphing)
  - Registration: Register two images taken of the same scene at different times/conditions

Summary of Morphological Filtering

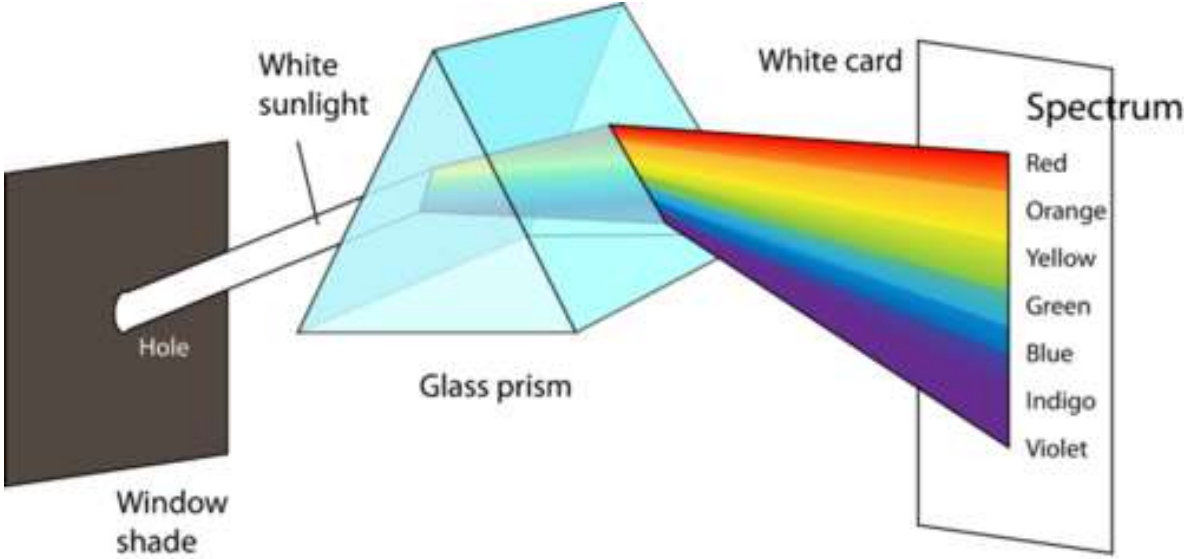
Operation	Equation	Comments (The Roman numerals refer to the structuring elements shown in Fig. 9.26).	MATLAB codes
Translation	$(A)_z = \{w \mid w = a + z, \quad \text{for } a \in A\}$	Translates the origin of A to point z.	circshift(A,z)
Reflection	$\hat{B} = \{w \mid w = -b, \quad \text{for } b \in B\}$	Reflects all elements of B about the origin of this set.	fliplr(flipud(B))
Complement	$A^c = \{w \mid w \notin A\}$	Set of points not in A.	~A or 1-A
Difference	$A - B = \{w \mid w \in A, w \notin B\} = A \cap B^c$	Set of points that belong to A but not to B.	A &~B
Dilation	$A \oplus B = \{z \mid (\hat{B})_z \cap A \neq \varnothing\}$	"Expands" the boundary of A. (I)	imdilate(A,B)
Erosion	$A \ominus B = \{z \mid (B)_z \subseteq A\}$	"Contracts" the boundary of A. (I)	imerode(A,B)
Opening	$A \circ B = (A \ominus B) \oplus B$	Smoothes contours, breaks narrow isthmuses, and eliminates small islands and sharp peaks. (I)	imopen(A,B)
Closing	$A \bullet B = (A \oplus B) \ominus B$	Smoothes contours, fuses narrow breaks and long thin gulfs, and eliminates small holes. (I)	imclose(A,B)

Slide courtesy: EE465: Introduction to Digital Image Processing

What is Color?

- Physical Phenomenon
- Physiopsychological Phenomenon

Color as a Physical Phenomenon

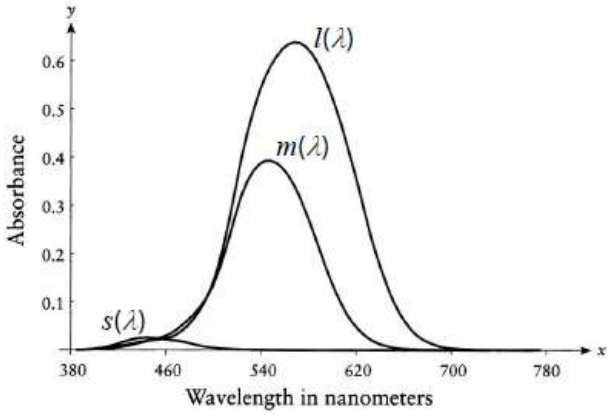


What is Color?

- Physical Phenomenon
- Physiopsychological Phenomenon

Biology of Color Vision

Cones come in three varieties: L, M, and S.



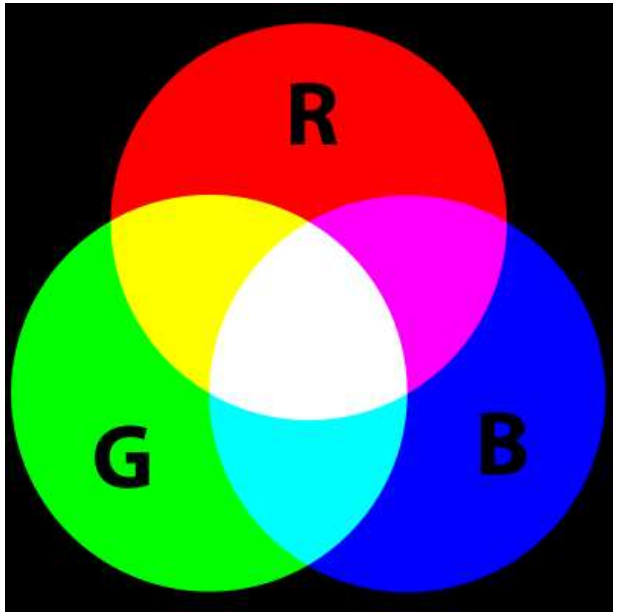
Cone photopigment absorption (Glassner, 1.1)

Cones are active under high light levels, i.e., they are responsible for **photopic** vision.

Primary Colors (color as three numbers)

- Additive (CRT displays, projectors etc.)
  - Combining red, green and blue light produces lighter colors, offering a good contrast to dark screens.
  - (Usually) Combined on a surface which reflects all light that falls on it
  - Easy to get **black** – don't shine any light

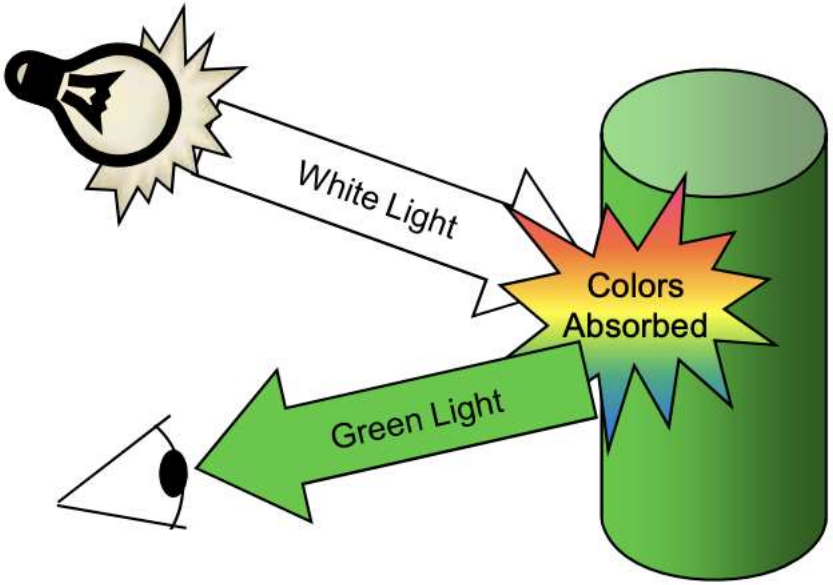
Magenta = Red + Blue
Cyan = Blue + Green
Yellow = Green + Red



Mantis shrimp
12 primary color receptors

'Perceived' color

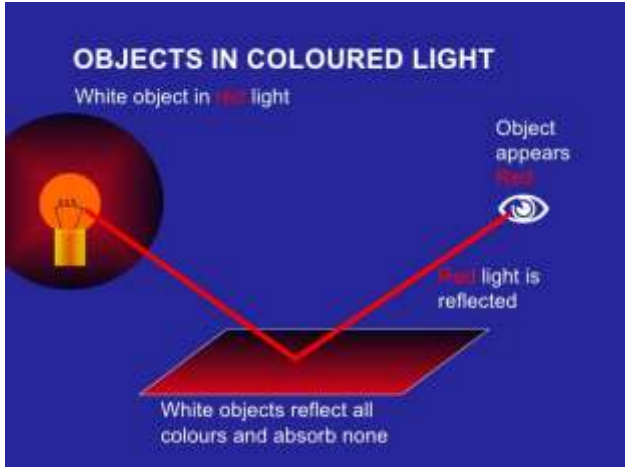
- Color is determined by nature of light reflected from an object
- E.g. green object reflects light from 500-570nm wavelength, absorbs other wavelengths

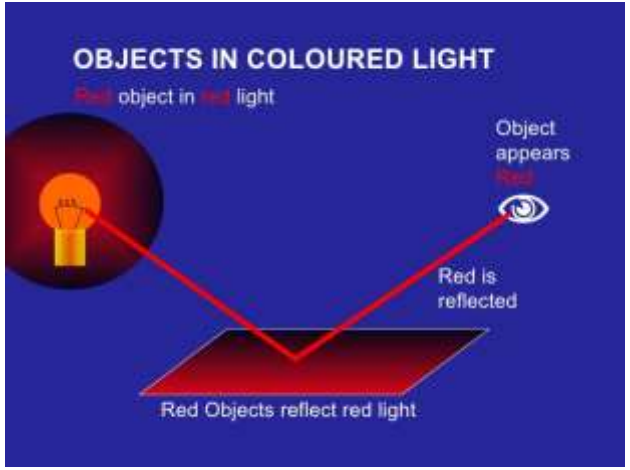


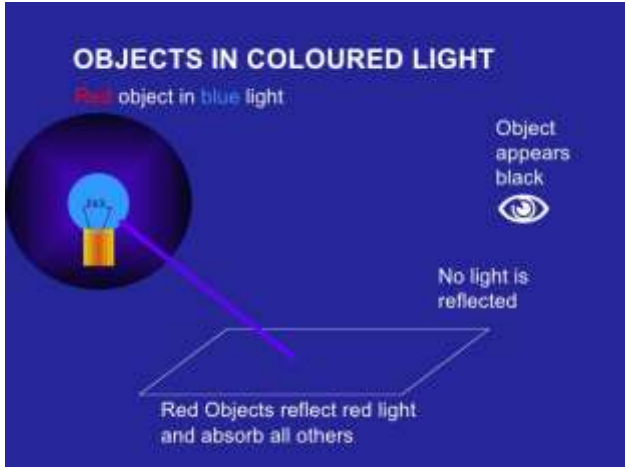
©The University of Waikato Te Whare Wānanga o Waikato|www.sciencelearn.org.nz

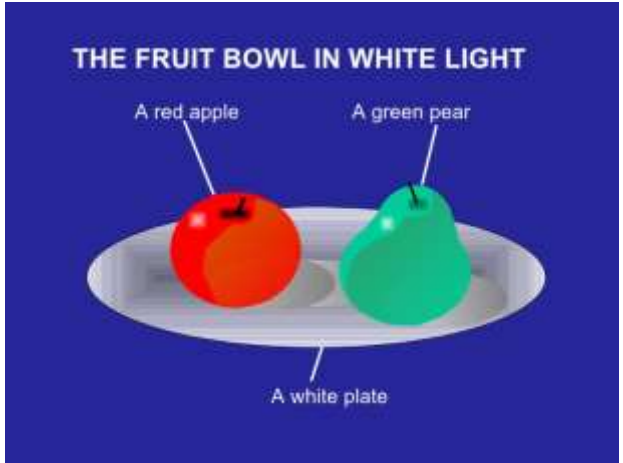
Courtesy: CNikou

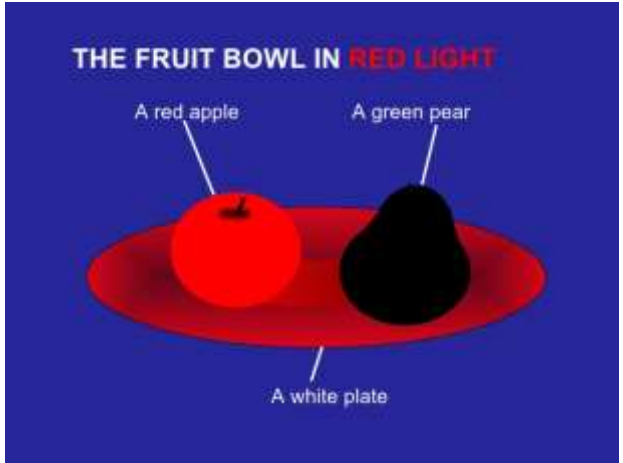
Perceived Color = f(Light, Object)

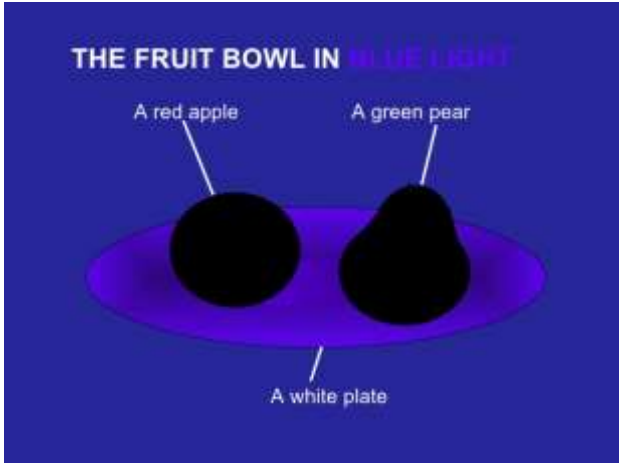












Full color

Red

Green

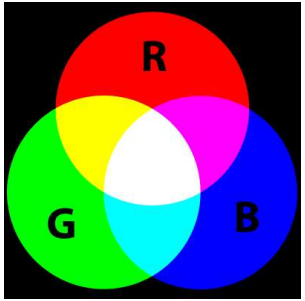
Blue

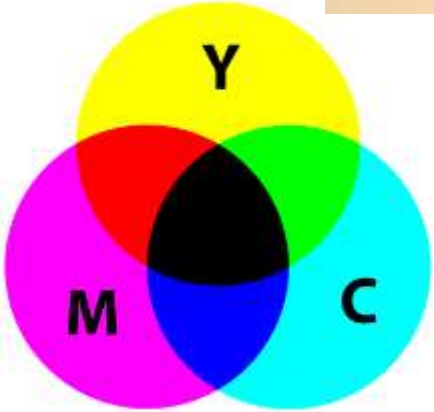
RGB channels

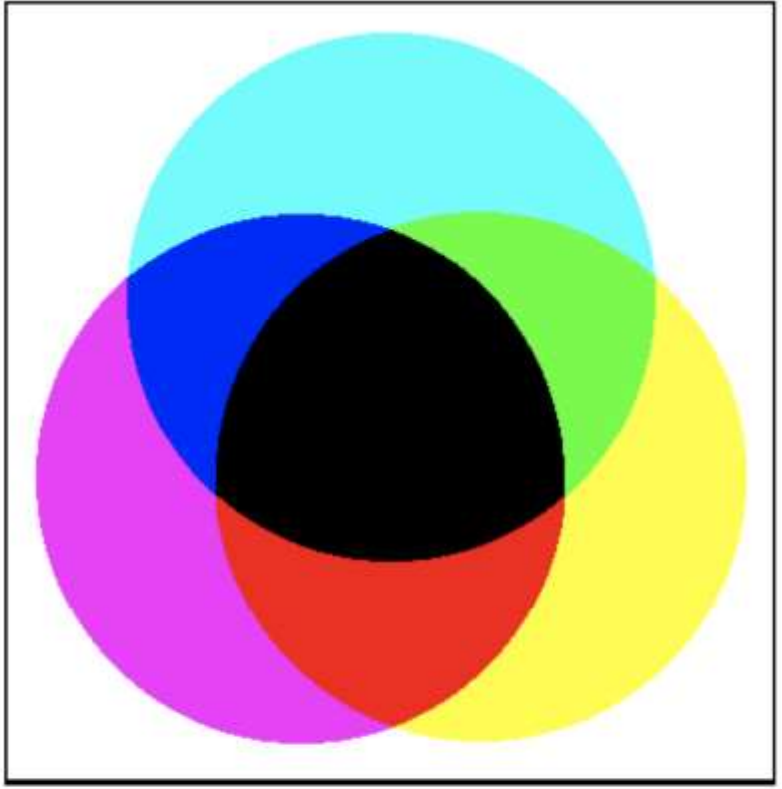
Primary Colors

Viewed under white light

- Subtractive
  - Paints, Printing
  - Starting with white paper, each addition of a colored ink "subtracts" available color from the starting medium.
  - For pure white , leave the paper unprinted (nothing can be whiter than the paper it is printed upon).



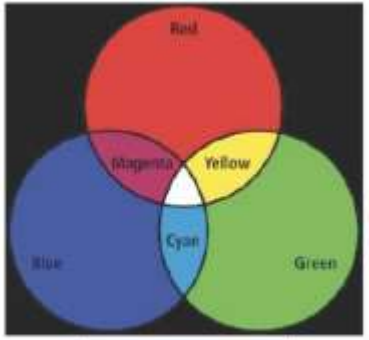




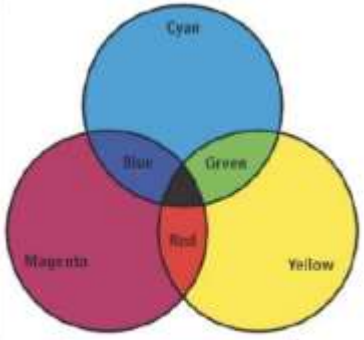
Passive displays, such as color inkjet printers, **absorb** light instead of emitting it. Combinations of **cyan**, **magenta** and **yellow** inks are used. This is a **subtractive** color model.

Why CMYK in printers

- Cyan, Magenta, and Yellow are lighter than red, green, and blue.
- CMY cover most lighter color ranges quite easily compared to RGB.
- BUT CMY by itself will not be able to create very deep dark colors (e.g. “true black,”)
- Black (designated “K” for “key color”) is added to CMY to achieve a much wider range of colors.

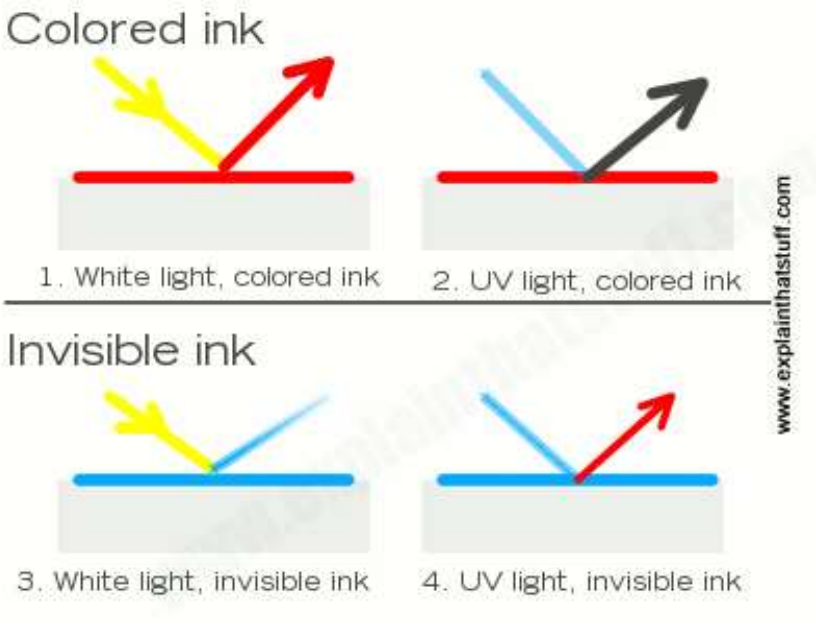


Magenta = Red + Blue
Cyan = Blue + Green
Yellow = Green + Red



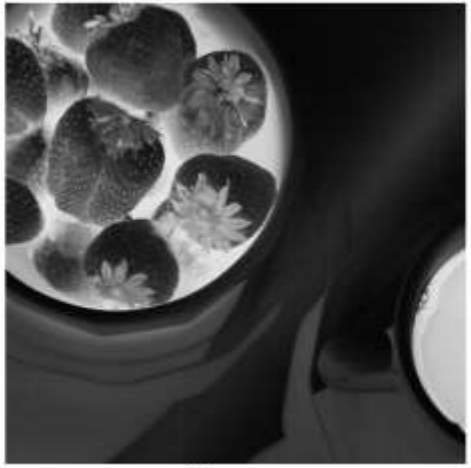
Magenta = White - Green
Cyan = White - Red
Yellow = White - Blue

White light shown as yellow for display

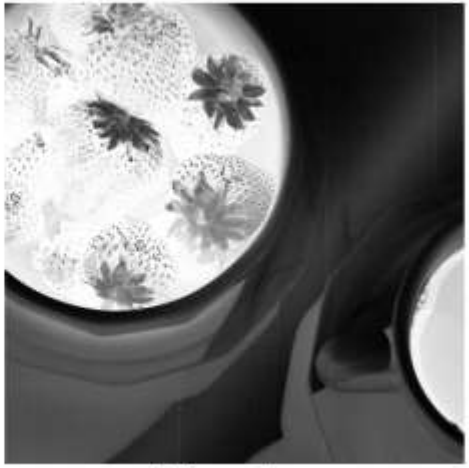


The Da Vinci Code

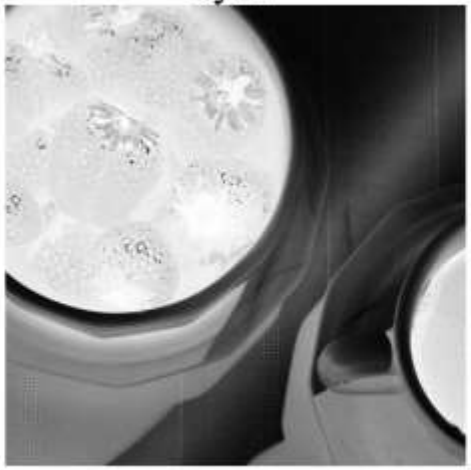
Full color



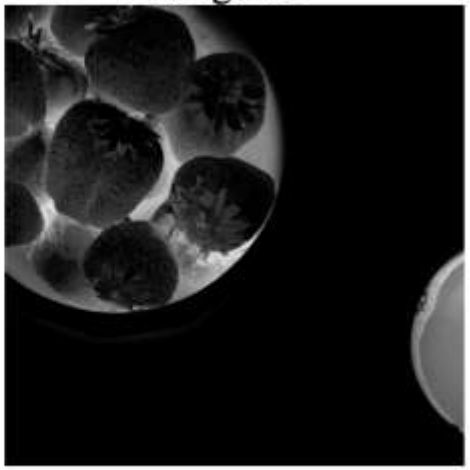
Cyan



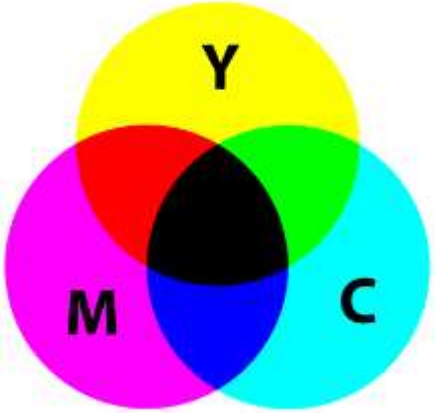
Magenta

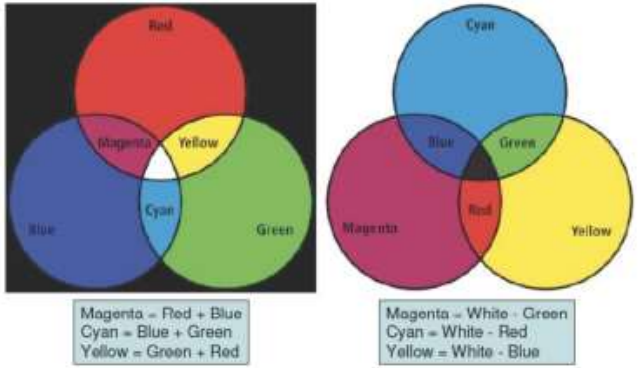


Yellow



Black





Original | Red Band | Green Band | Blue Band

Cyan | Magenta | Yellow

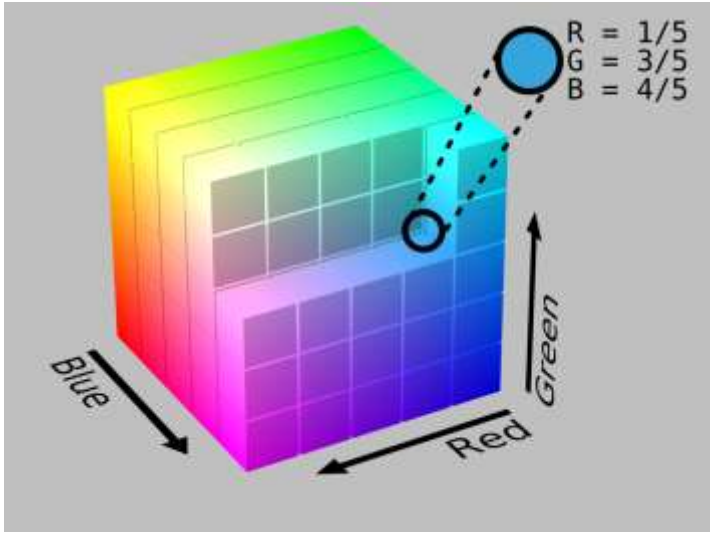
No Red | No Green | No Blue

Color Models

- RGB
- HSI / HSY
- CIE LAB

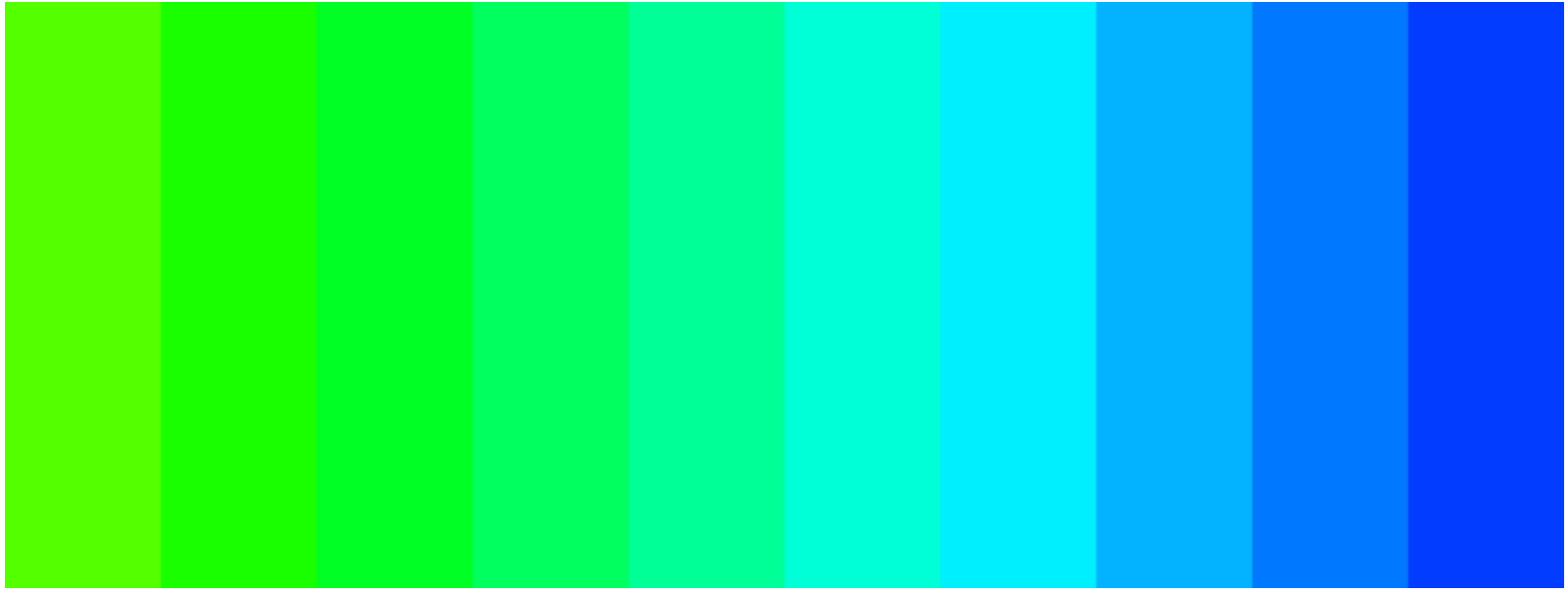
RGB color space

- Primary colors
- Additive color model $f(x, y) = \alpha_1 R + \alpha_2 G + \alpha_3 B$
- Perceptually non uniform



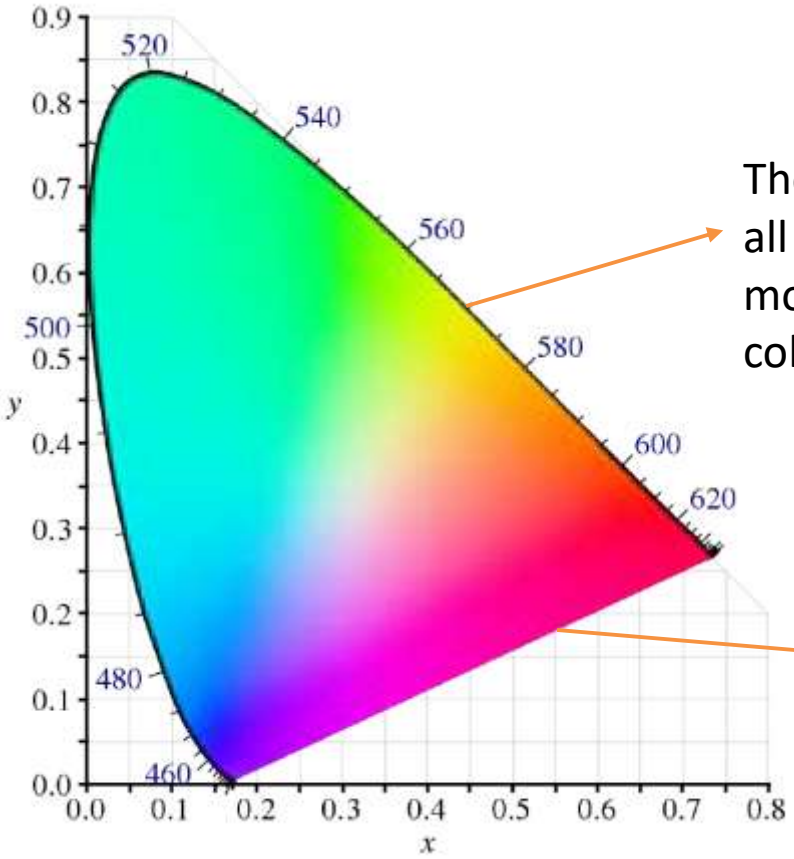
Courtesy: wikipedia

RGB : Non-Uniform Perceptual Space



CIE 1931 (x,y) chromaticity diagram

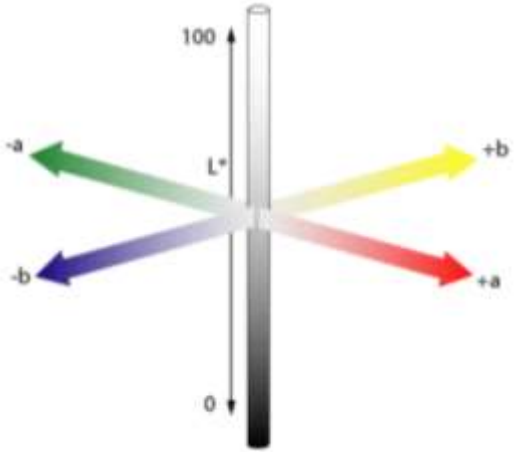
- Yxy - luminance + the two most distinctive chrominance components
- Helps separate luminance and chromaticity
- A convenient representation for color values

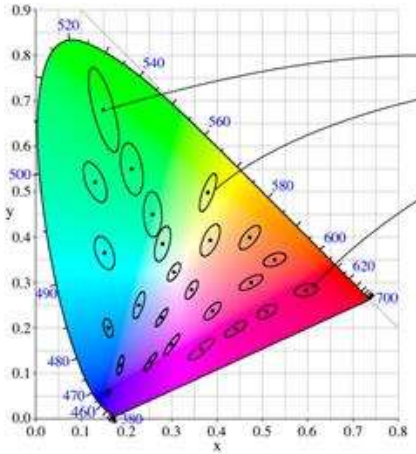


The rim represents all of the pure monochromatic colors

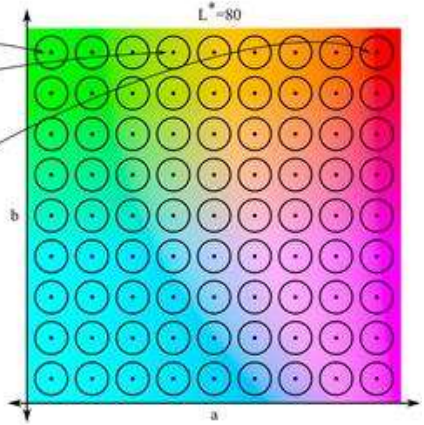
Purple Line

CIE Lab color space





CIE 1931



CIELAB (CIE 1976 L*,a*,b*)

Components:

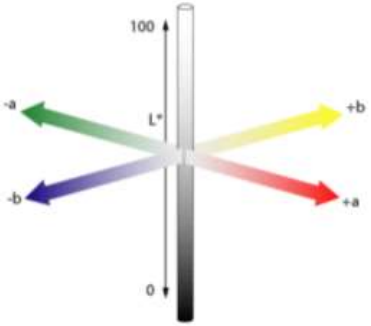
- Luminosity L
- Color components
  - a* : color hue, saturation along green-red axis
  - b* : color hue, saturation along blue-yellow axis

- Makes changes in CIE space linear with respect to human perception
- Popular in high-quality photographic applications

- Luminosity L
- Color components
  - a* : color hue and saturation along green-red axis
  - b* : color hue and saturation along blue-yellow axis

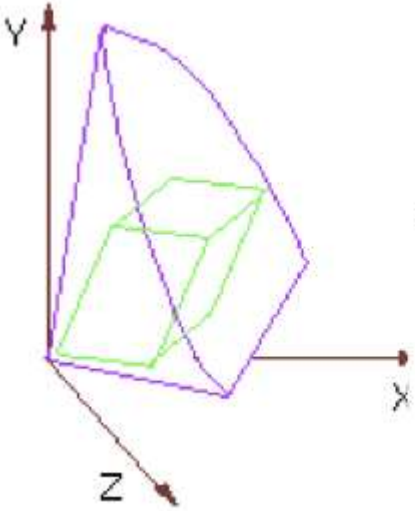
Example of L*a*b* Components

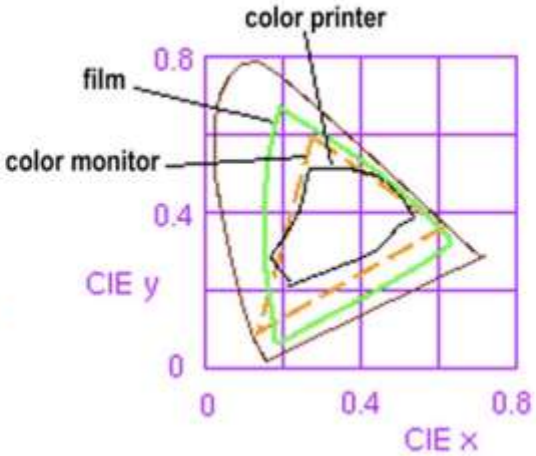
L^* a^* b^*

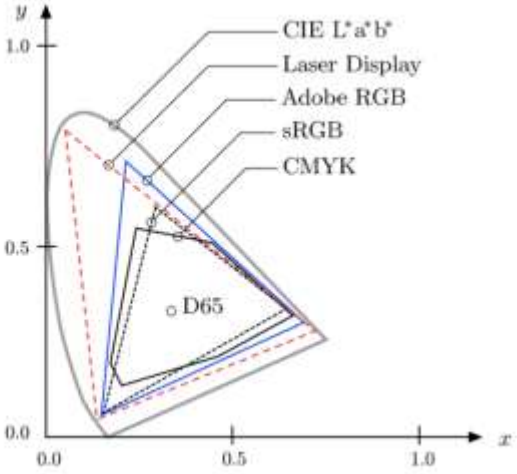


Device Color Gamuts

- The RGB color cube sits within CIE color space
- We can use the CIE chromaticity diagram to compare the gamuts of various devices
- E.g. compare color printer and monitor color gamuts





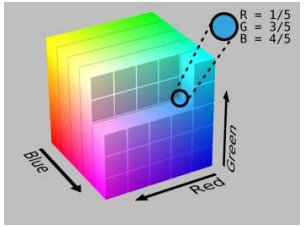


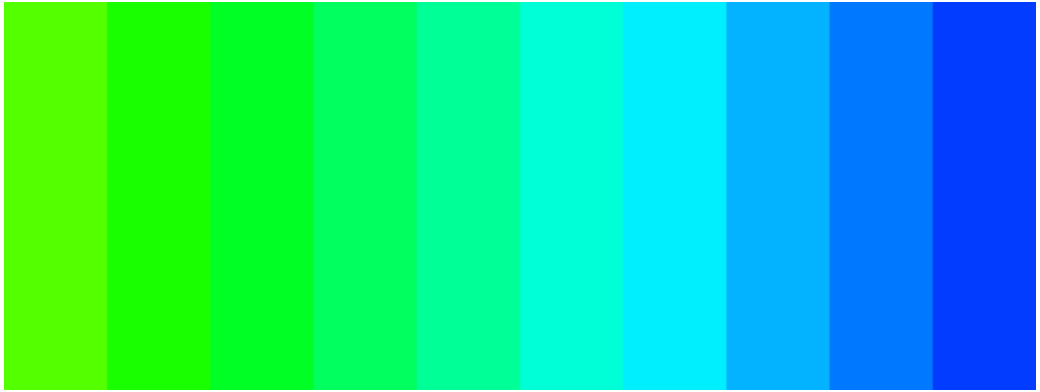
Measuring Color Differences

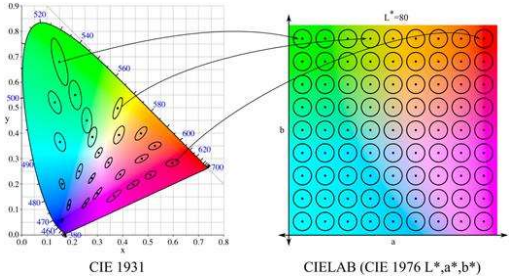
- Due to its uniformity with respect to human perception, differences between colors in L*a*b* color space can be determined as **euclidean distance**

$$\begin{aligned}\mathrm{ColorDist}_{\mathrm{Lab}}(\mathbf{C}_1, \mathbf{C}_2) &= \|\mathbf{C}_1 - \mathbf{C}_2\| \\ &= \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2}\end{aligned}$$

Perceptually Uniform Color Spaces



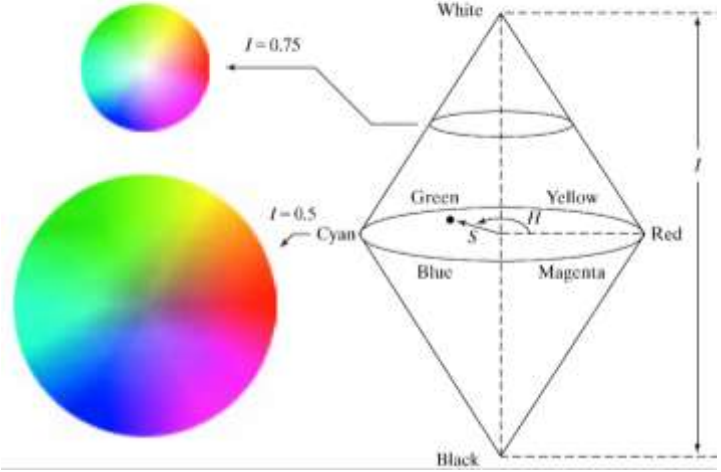




https://matplotlib.org/3.1.1/tutorials/colors/colormaps.html

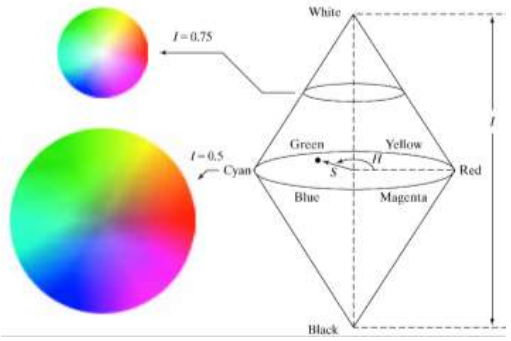
HSI color space

- **HUE**
  - A subjective measure of color
  - Average human eye can perceive ~200 different colors
- **Saturation**
  - Relative purity of the color. Mixing more "white" with color reduces its saturation.
  - Pink has the same **hue** as **red** but less **saturation**
- **Intensity**
  - The brightness or darkness of an object



RGB : great for color generation
HSI : great for color description

Example: RGB to HSV Conversion



Original RGB image

H_{HSV} S_{HSV} V_{HSV}

HSV values in grayscale

White Balancing

Courtesy: wahlmanphotography.com

Courtesy: wallcreations.com.au

White Balancing

White Balancing

White Balancing

```
im = double(imread('lighthouse.jpg'));

RGBw = [246 169 87];

im1(:,:,1) = im(:,:,1)*255/RGBw(1);
im1(:,:,2) = im(:,:,2)*255/RGBw(2);
im1(:,:,3) = im(:,:,3)*255/RGBw(3);
```

White Balancing

Incandescent lighting

Fluorescent lighting

Sunlight

Camera Flash

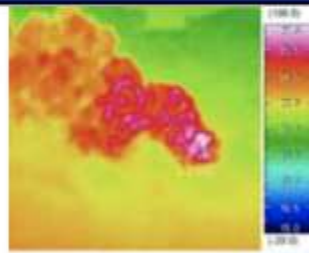
Cloudy

Shadow

courtesy: seventhgrove.com

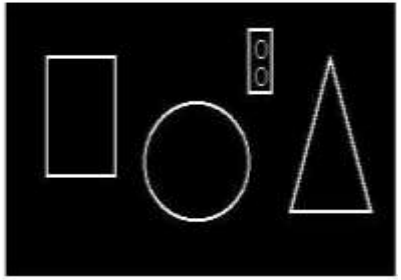
Pseudocolor Image Processing

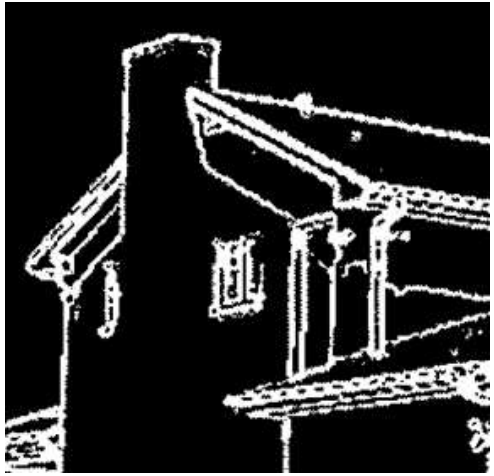
- Pseudocolor (also called false color) image processing consists of assigning colors to grey values based on a specific criterion.
- The principle use of pseudocolor image processing is for human visualisation.
  - Humans can discern between thousands of color shades and intensities, compared to only about two dozen or so shades of grey.

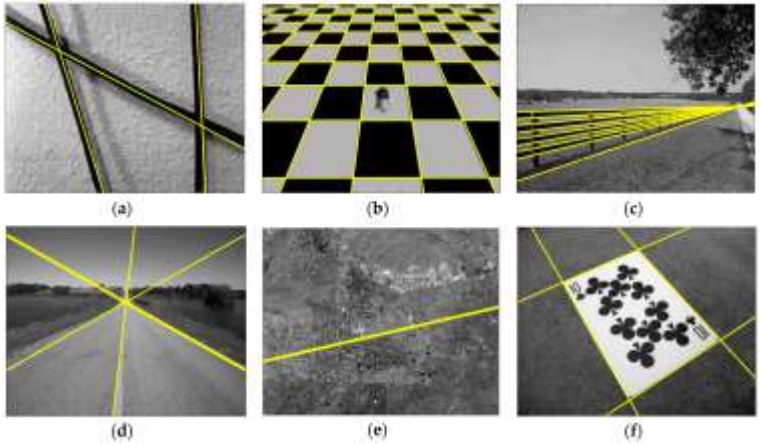


Hough Transform

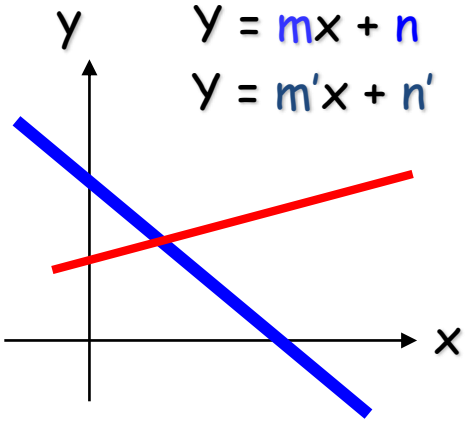
- Straight lines
- Circles
- Algebraic curves
- Arbitrary specific shapes in an image







Hough Transform for Lines: Image and Parameter Spaces



$y = mx + n$

$y = m'x + n'$

$y_1 = m\,x_1 + n$

$y_2 = m\,x_2 + n$

$y_N = m\,x_N + n$

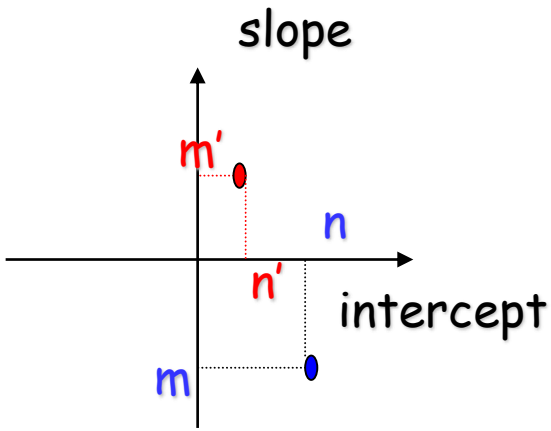


Image Space

Parameter Space

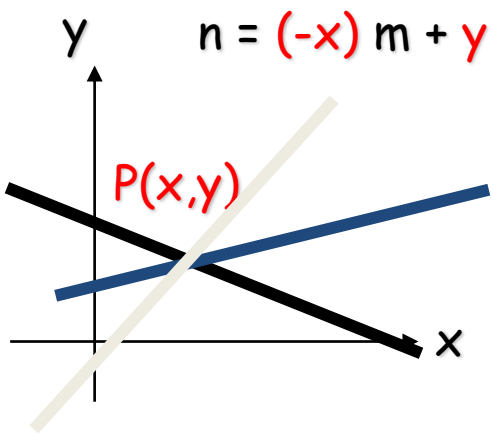
Line in Img. Space ~ Point in Param. Space

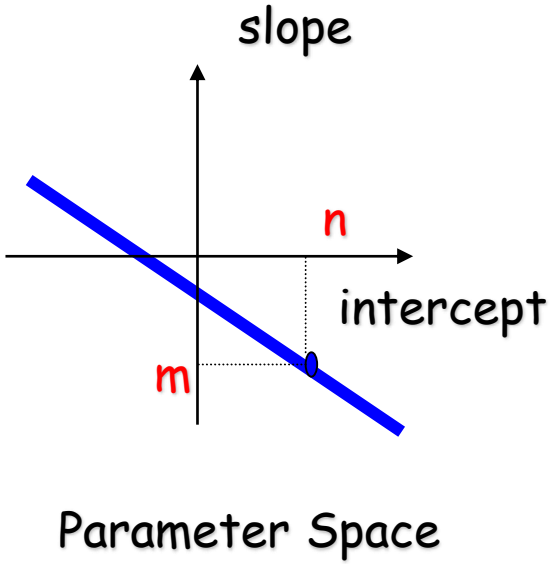
Image Parameter Spaces

- Image Space
  - Lines
  - Points
  - Collinear points
- Parameter Space
  - Points
  - Lines
  - Intersecting lines

Hough Transform Technique

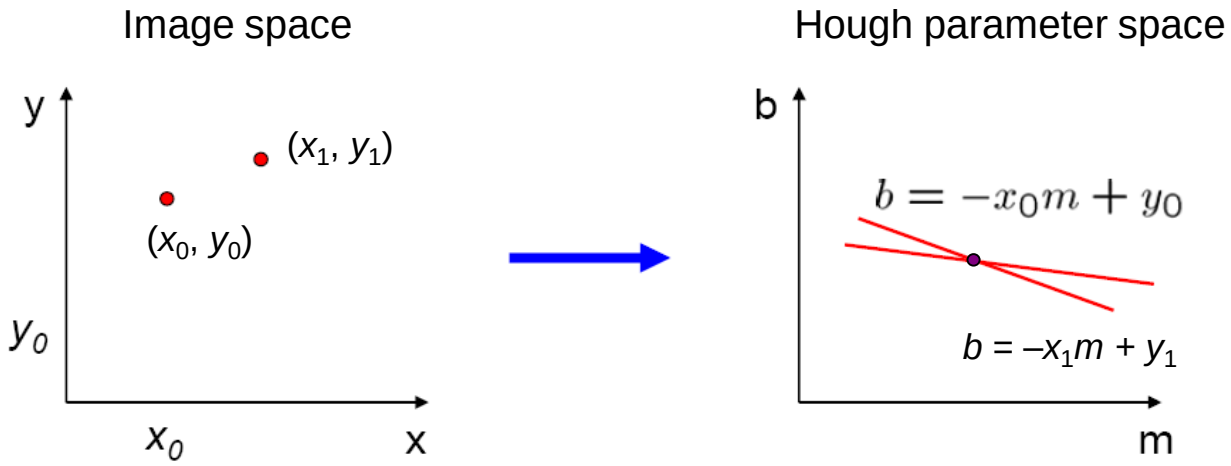
- Given an edge point, there is an infinite number of lines passing through it (Vary m and n).
  - These lines can be represented as a line in parameter space.





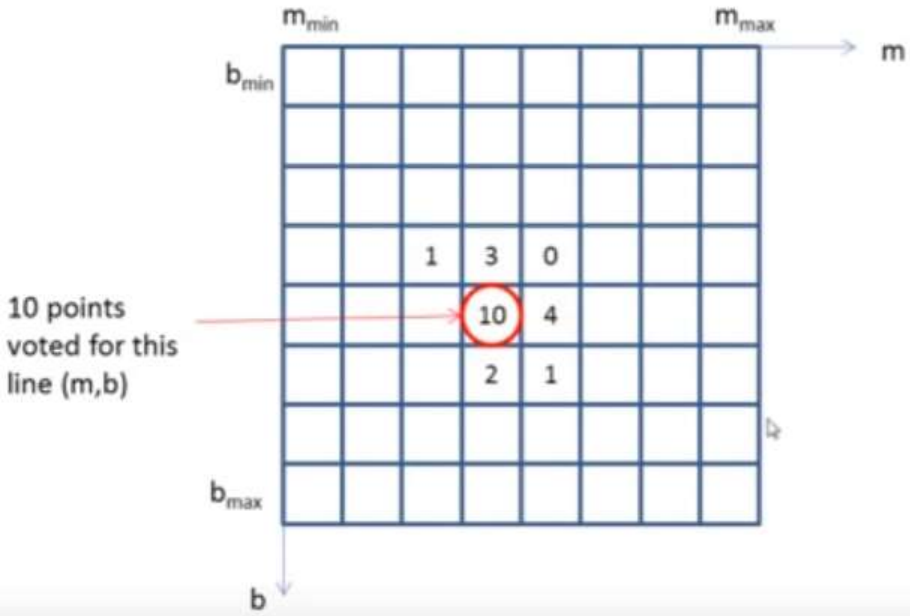
Parameter space representation

- Where is the line that contains both (x_0, y_0) and (x_1, y_1)?
  - It is the intersection of the lines $b = -x_0 m + y_0$ and $b = -x_1 m + y_1$



Hough Transform Algorithm

- Initialize an accumulator array A(m,b) to zero
- For each edge element (x,y), increment all cells that satisfy b = -x m + y
- Local maxima in A(m,b) correspond to lines

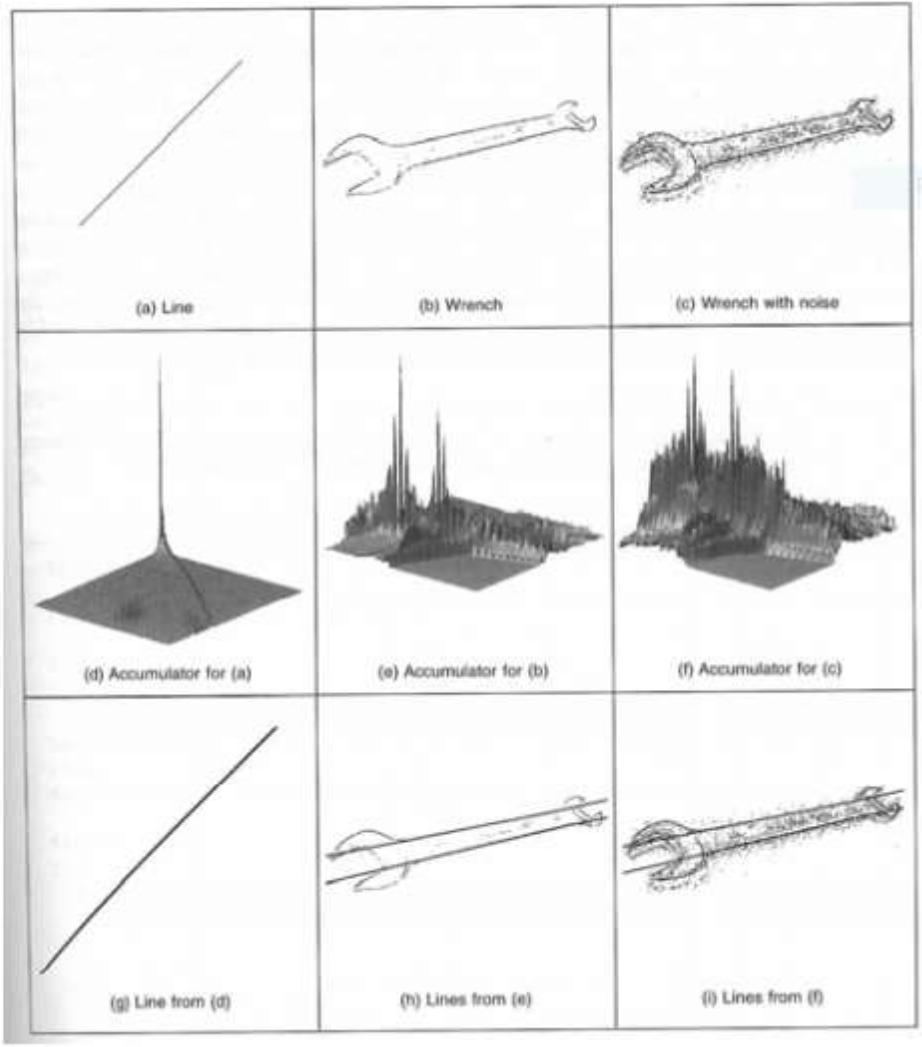


https://www.youtube.com/watch?v=uDB2qGqnQ1g

Thresholded edge images

Visualizing the accumulator space
The height of the peak will be defined by the number of pixels in the line.

Thresholding the accumulator space and superimposing this onto the edge image

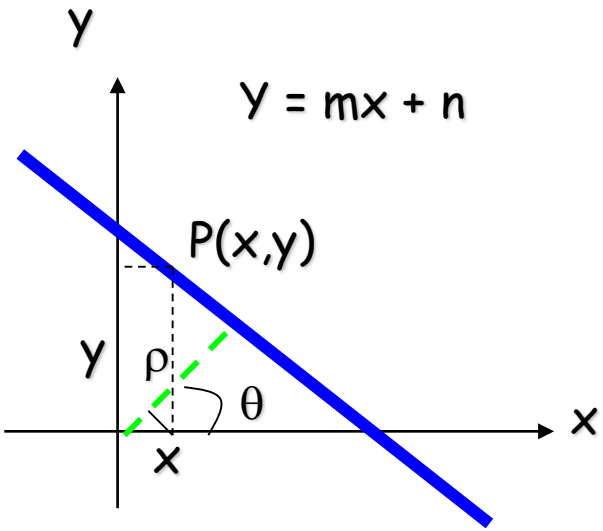


Practical Issues with This Hough Parameterization

- The slope of the line is $-\infty < m < \infty$
  - The parameter space is INFINITE
- The representation $y = mx + n$ does not express lines of the form $x = k$

Solution:

- Use the “Normal” equation of a line:



$\rho = x\cos\theta + y\sin\theta$

θ Is the line orientation

ρ Is the distance between the origin and the line

Consequence:

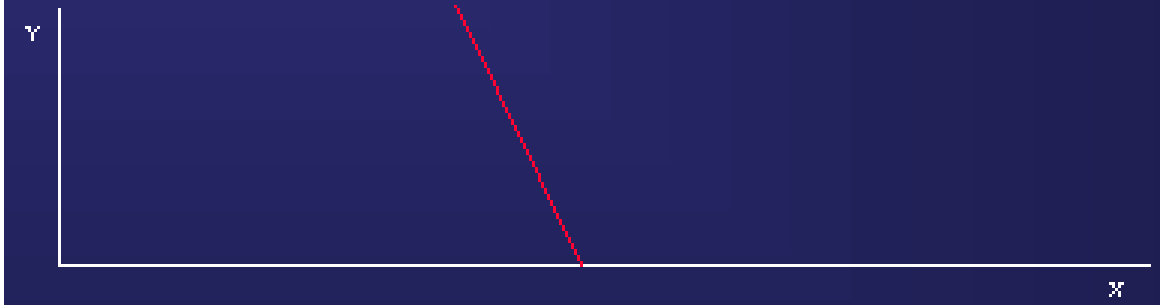
- A Point in Image Space is now represented as a SINUSOID
  - $\rho = x \cos\theta + y \sin\theta$

New Parameter Space for Hough based on trigonometric functions

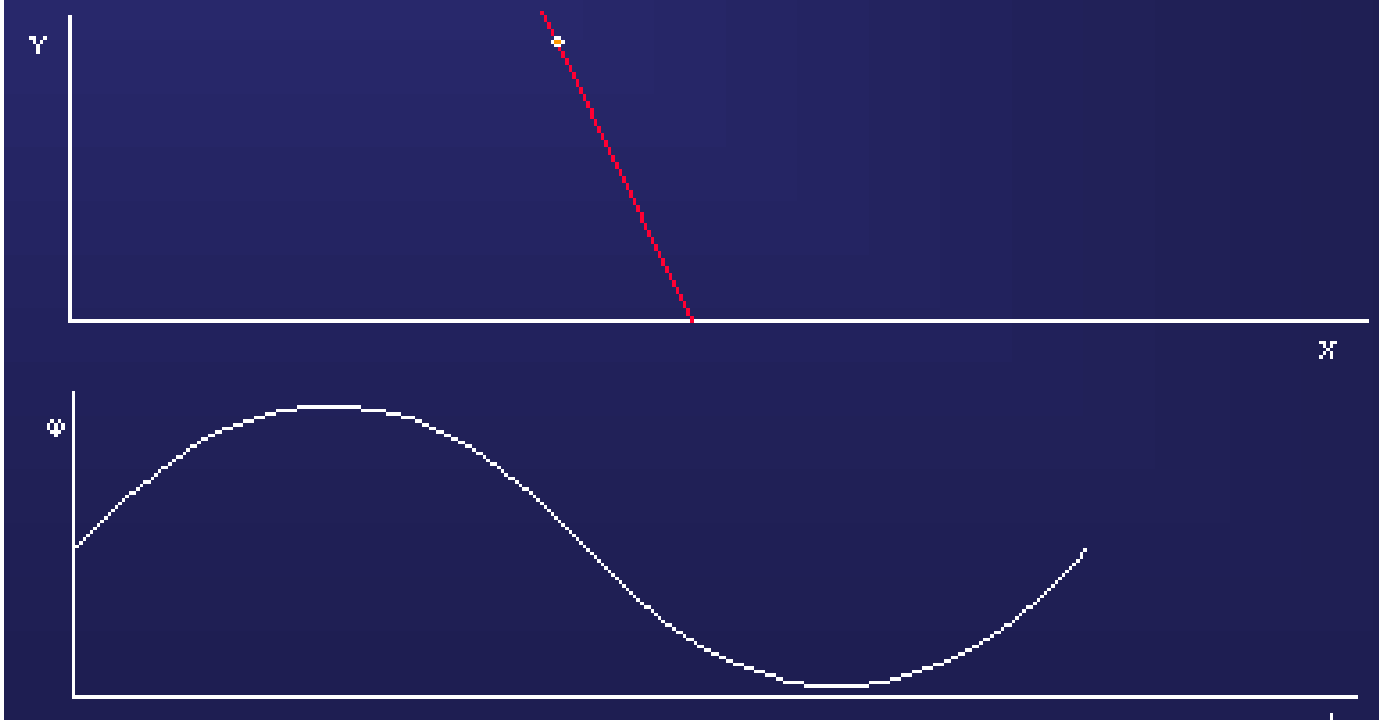
- Use the parameter space (ρ, θ)
- The new space is FINITE
  - $0 < \rho < D$, where D is the image diagonal.
  - $-\pi < \theta < \pi$
- The new space can represent all lines
  - $Y = k$ is represented with $\rho = k$, $\theta=90$
  - $X = k$ is represented with $\rho = k$, $\theta=0$

SHT: Another Viewpoint

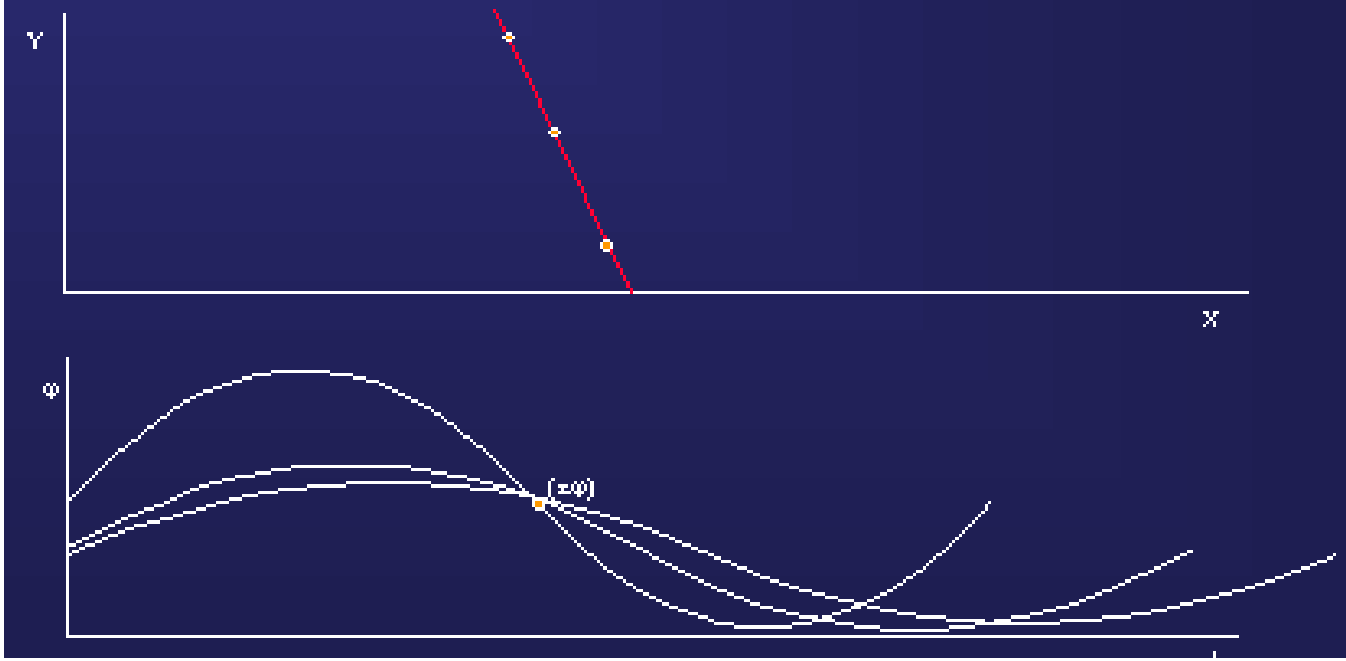
- Since any (r,φ) represents a point in parameter space we may plot all possible (r,φ) points for each (x,y) poi



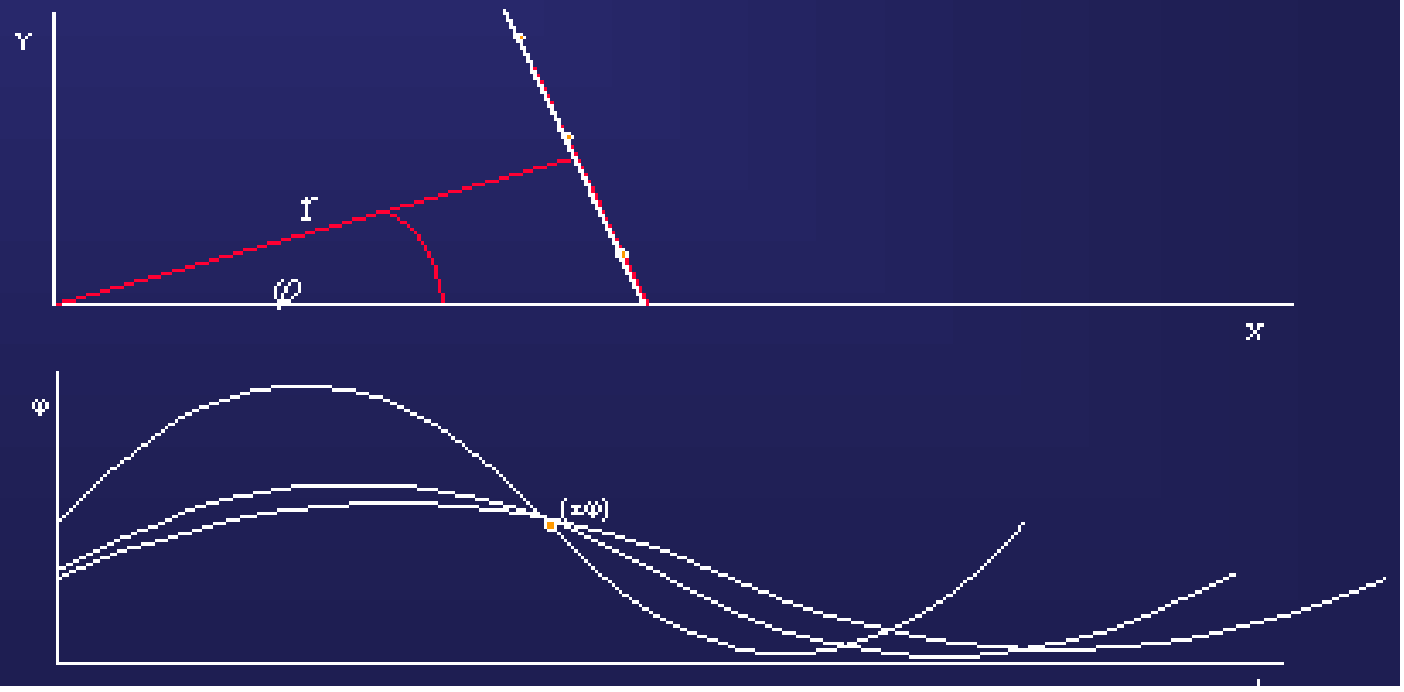
- Taking any particular (x,y) point its set of (r,φ) points is a sinusoid through parameter space



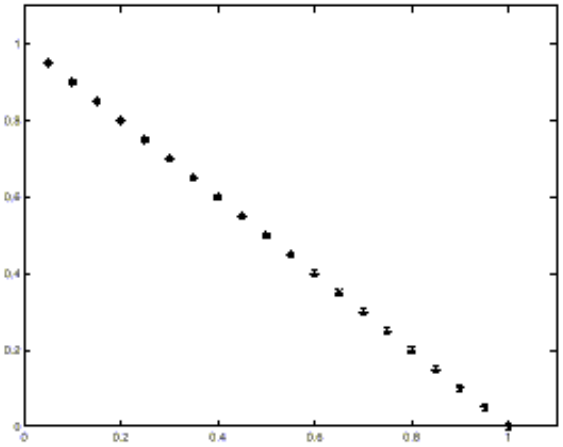
- The line upon which the (x,y) points lie is then given by the (r,φ) pair on which all the sinusoids *agree* (i.e. intersect)

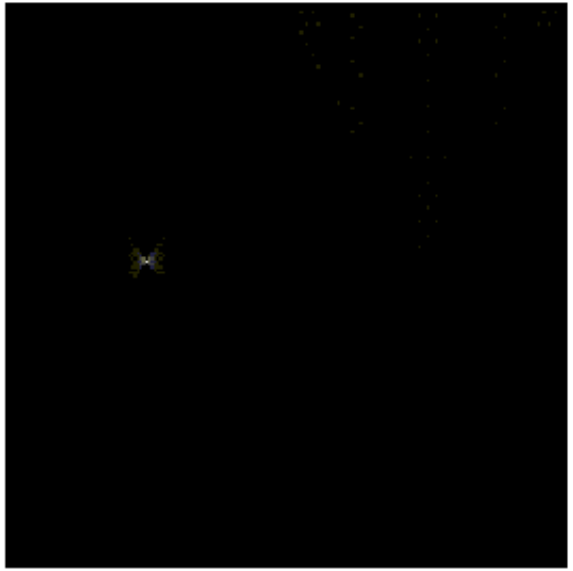


- Finding this intersection we then have the required (r,φ) parameters and therefore the line in the image

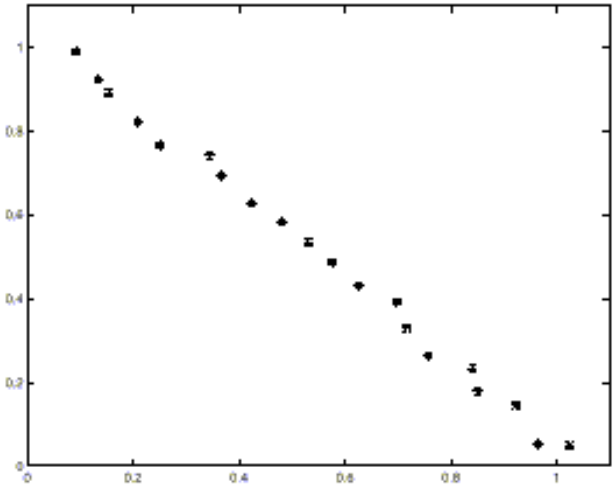


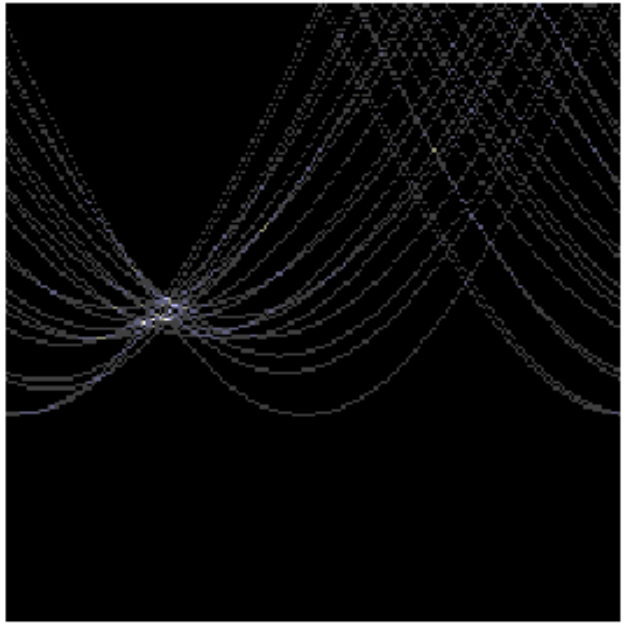
Hough Transform – cont.



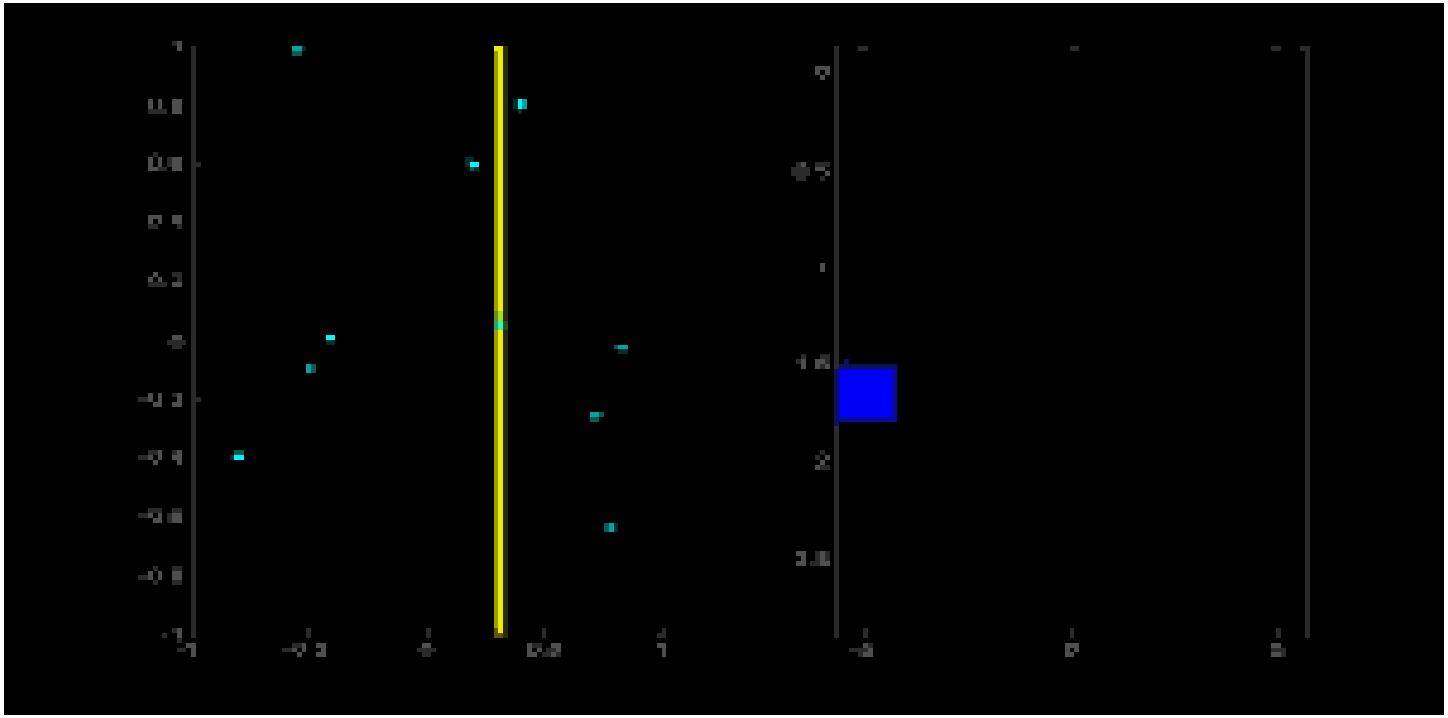


Hough Transform – cont.





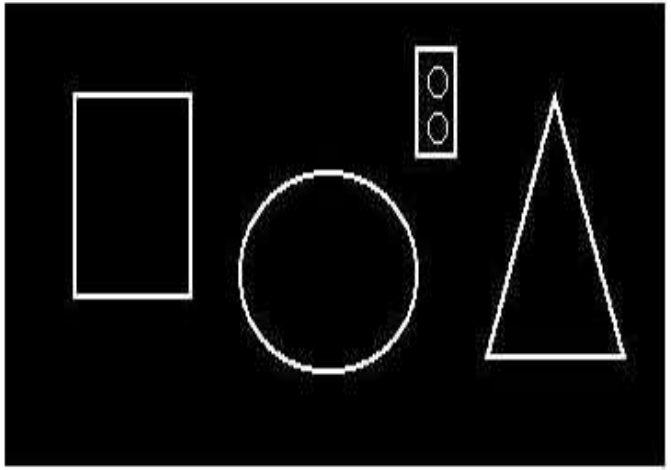
Animation



http://homepages.inf.ed.ac.uk/amos/hough.html

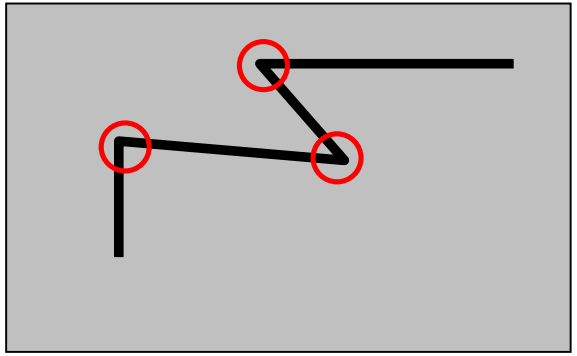
Hough Transform Generalizations

- It locates straight lines (SHT) - standard, simple HT
- It locates straight line intervals
- It locates circles
- It locates algebraic curves
- It locates arbitrary specific shapes in an image
  - **But you pay progressively for complexity of shapes by time and memory usage**



Harris corner detector

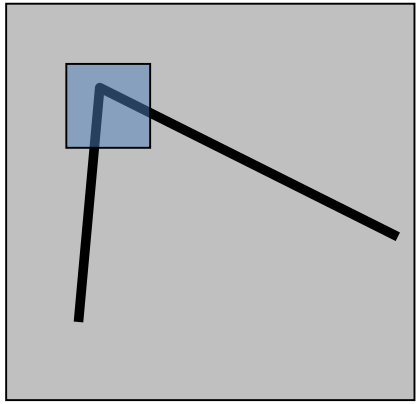
- C.Harris, M.Stephens. “A Combined Corner and Edge Detector”. 1988



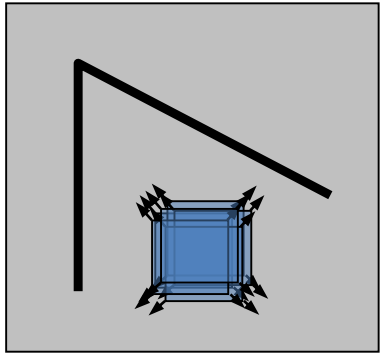
from Rick Szeliski’s lecture notes, CSE576, Spring 05

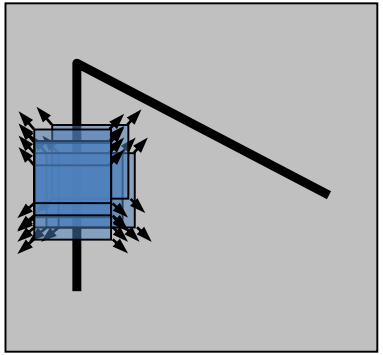
The Basic Idea

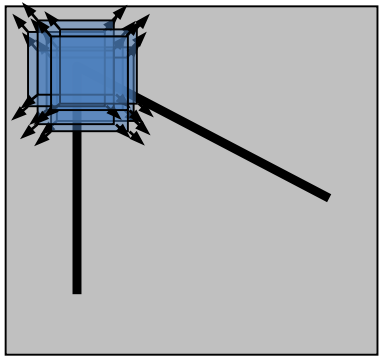
- We should easily recognize the point by looking through a small window
- Shifting a window in *any direction* should give *a large change* in intensity



Harris Detector: Basic Idea







"flat" region:
no change in
all directions

"edge":
no change along
the edge direction

"corner":
significant
change in all
directions

Harris Detector: Mathematics

Change of intensity for the shift [*u*,*v*]:

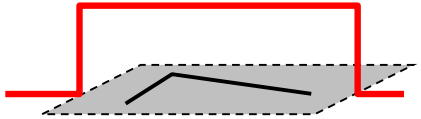
$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2$$

Window function

Shifted intensity

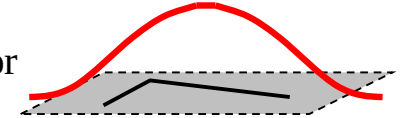
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

Harris Detector: Mathematics

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

Harris Detector: Mathematics

$$E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2$$

For small shifts $[u,v]$ we have a *bilinear* approximation:

$$E(u,v) \cong [u,v] \quad M \quad \begin{bmatrix} u \\ v \end{bmatrix}$$

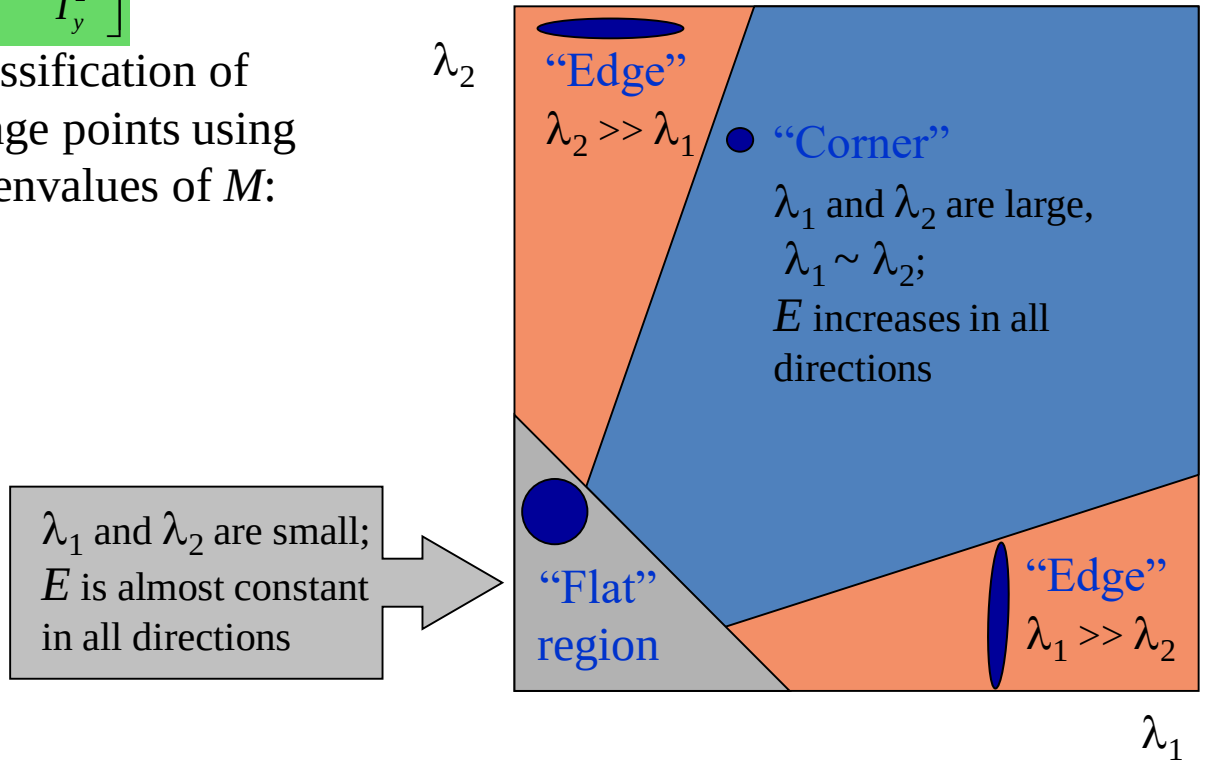
where M is a 2×2 matrix computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Harris Detector: Mathematics

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Classification of image points using eigenvalues of M:



Harris Detector: Mathematics

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Measure of corner response:

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$
$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

(k – empirical constant, k = 0.04-0.06)

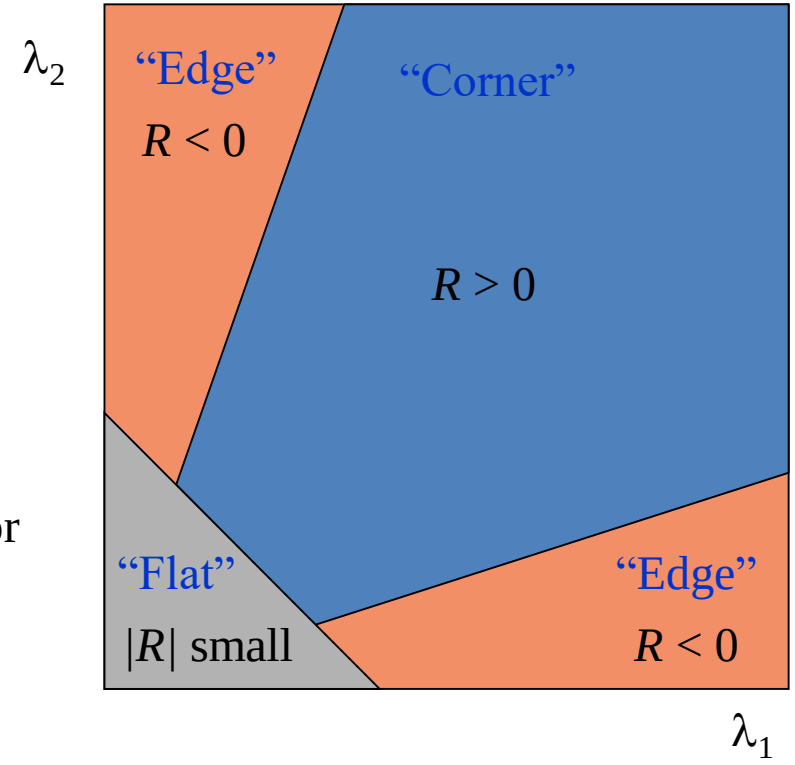
Harris Detector: Mathematics

$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

$$\det M = \lambda_1 \lambda_2$$

$$\operatorname{trace} M = \lambda_1 + \lambda_2$$

- R depends only on eigenvalues of M
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Harris Detector

- The Algorithm:
  - Find points with large corner response function R (R > threshold)
  - Take the points of local maxima of R

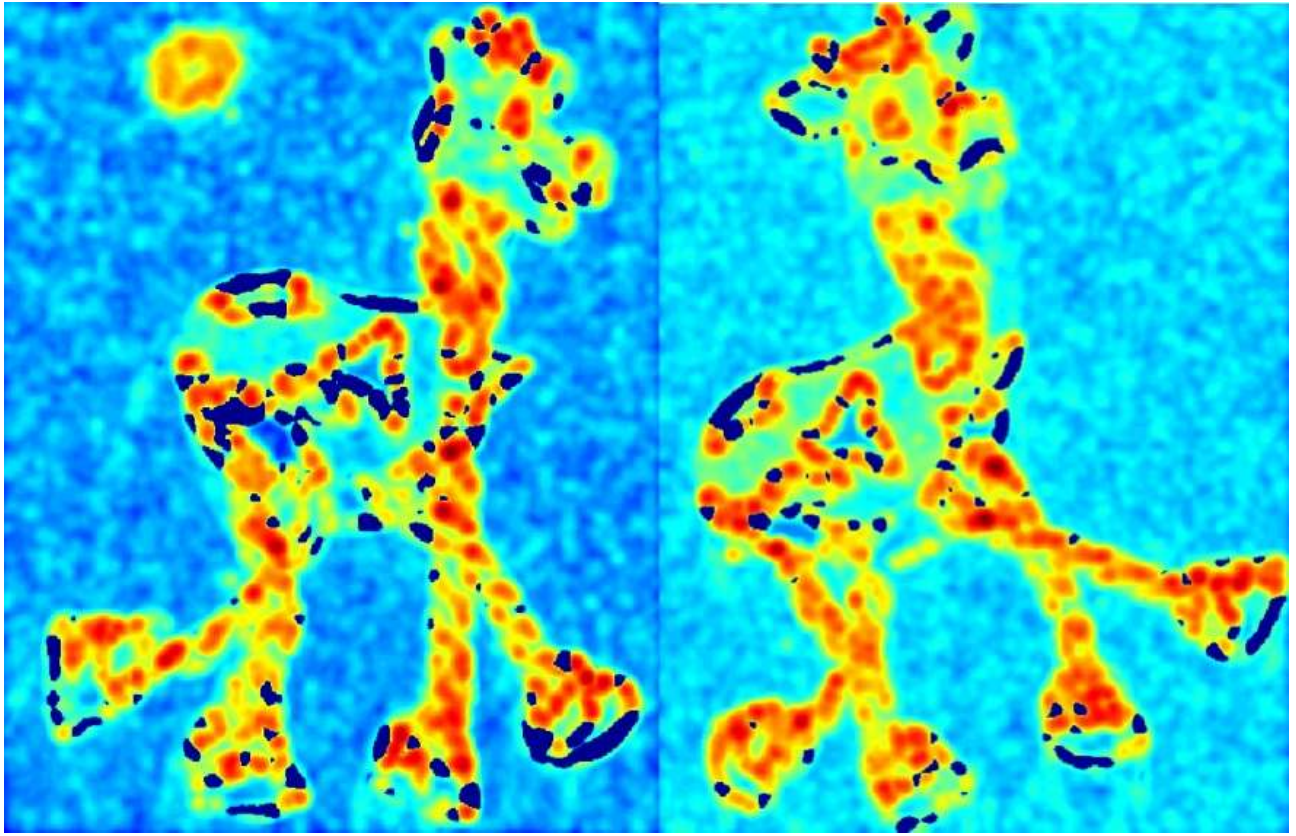
Harris Detector: Workflow

Harris Detector: Workflow

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

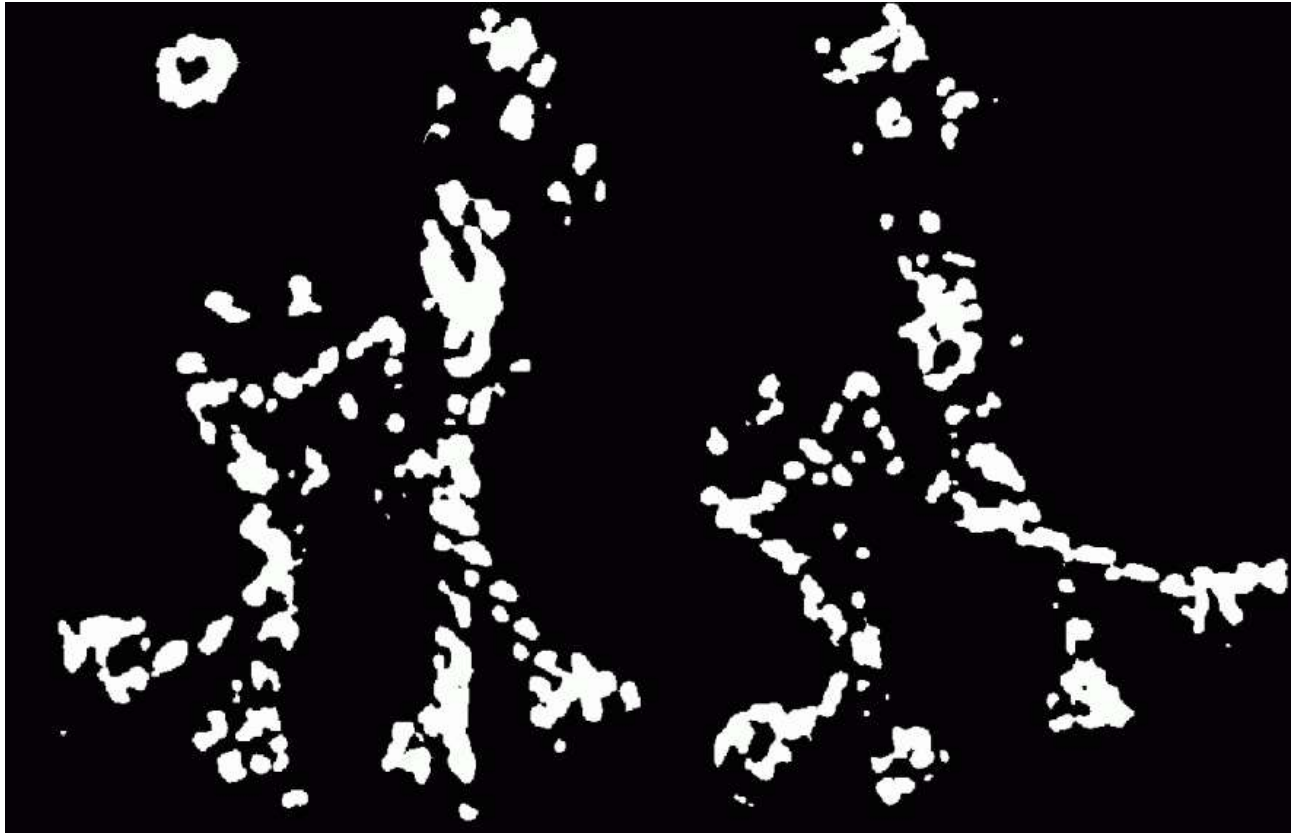
$$R = \det M - k\left(\operatorname{trace} M\right)^2$$

Compute corner response R



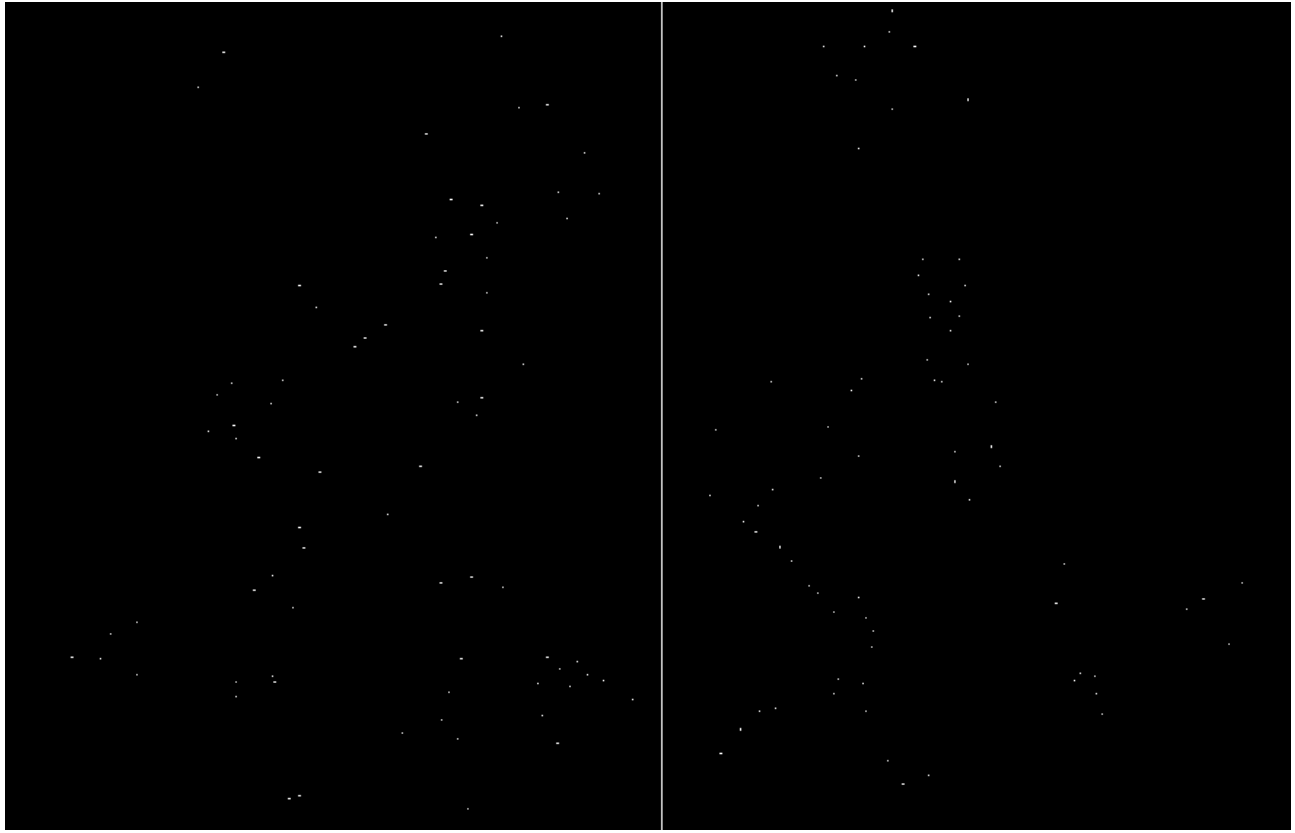
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

Take only the points of local maxima of R



Harris Detector: Workflow

Harris Detector: Summary

- Average intensity change in direction $[u,v]$ can be expressed as a bilinear form:

$$E(u,v) \cong [u,v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

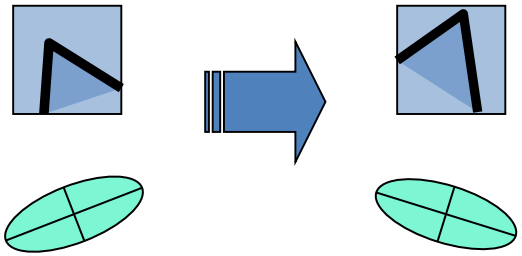
- Describe a point in terms of eigenvalues of M: *measure of corner response*

$$R = \lambda_1 \lambda_2 - k(\lambda_1 + \lambda_2)^2$$

- A good (corner) point should have a *large intensity change* in *all directions*, i.e. R should be large positive

Harris Detector: Some Properties

- Rotation invariance



Ellipse rotates but its shape (i.e. eigenvalues) remains the same

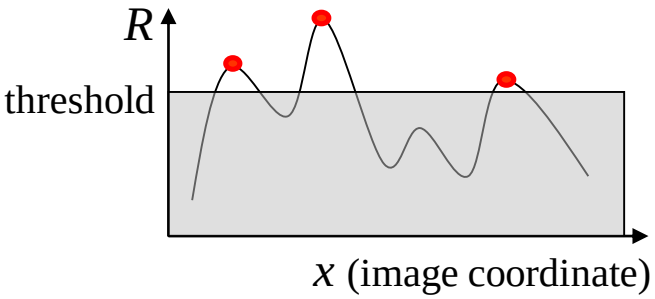
Corner response R is invariant to image rotation

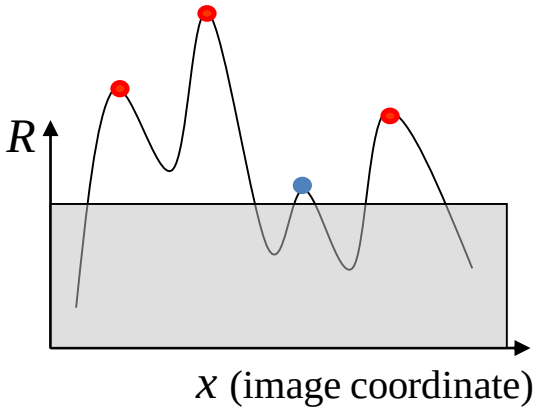
Harris Detector: Some Properties

- Partial invariance to *affine intensity* change

✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

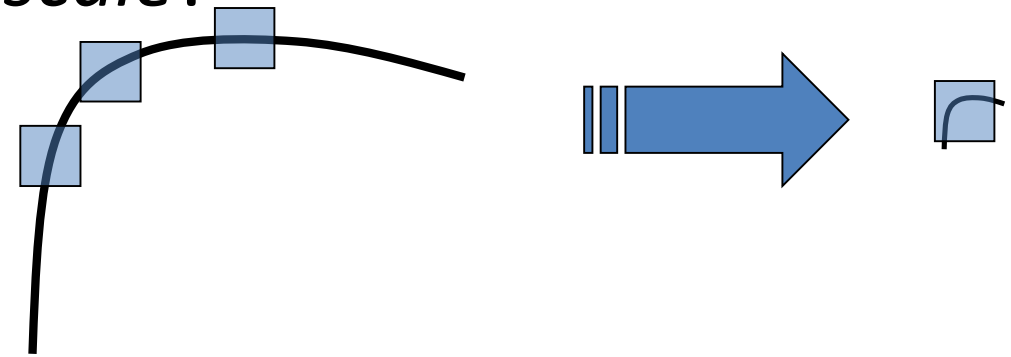
✓ Intensity scale: $I \rightarrow a\ I$





Harris Detector: Some Properties

- But: non-invariant to *image scale*!



All points will be classified as edges

Corner !

Image Segmentation

Partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image

Image Segmentation

Partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image

2. into linear structures, such as
   - line segments
   - curve segments

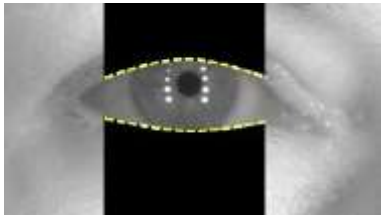
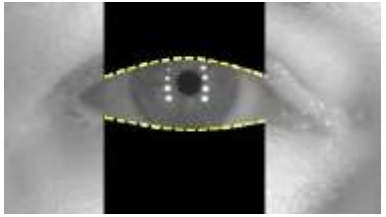


Image Segmentation

Partitioning an image into a collection of connected sets of pixels.

1. into regions, which usually cover the image

2. into linear structures, such as
   - line segments
   - curve segments



3. into 2D shapes, such as
   - circles
   - ellipses
   - ribbons (long, symmetric regions)

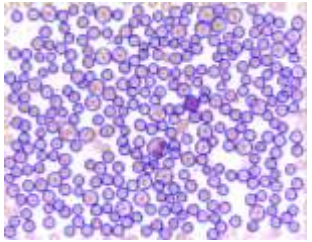


Image Segmentation - Approaches

- Edge-based
- Thresholding
- Region-growing
- Morphological Watersheds
- Motion

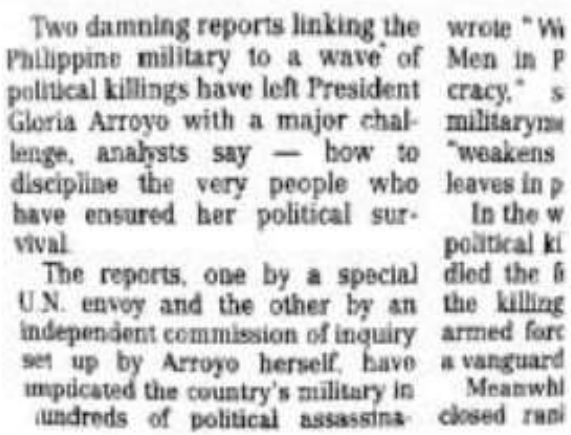
Segmentation: Thresholding based approaches

Two class Segmentation: Motivating example

- Separate pixels associated with object of interest from background

Two damning reports linking the Philippine military to a wave of political killings have left President Gloria Arroyo with a major challenge, analysts say — how to discipline the very people who have ensured her political survival.

The reports, one by a special U.N. envoy and the other by an independent commission of inquiry set up by Arroyo herself, have implicated the country's military in hundreds of political assassina-



Two damning reports linking the Philippine military to a wave of political killings have left President Gloria Arroyo with a major challenge, analysts say — how to discipline the very people who have ensured her political survival.

The reports, one by a special U.N. envoy and the other by an independent commission of inquiry set up by Arroyo herself, have implicated the country's military in hundreds of political assassina-

Thresholding

- Separate pixels associated with object of interest from background

- Given a image f(x,y), the segmented image g(x,y) is given by:

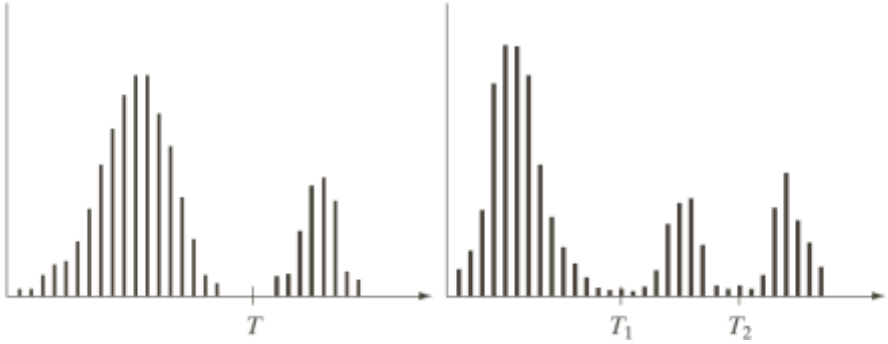
$$g(x,y) = \begin{cases} 1 & \text{if } f(x,y) > T \\ 0 & \text{if } f(x,y) \leq T \end{cases}$$

If T is constant over the entire image → Global Thresholding
If T changes over the image → Variable Thresholding

The main question is: **How to find T?**

Thresholding

- How to find T?
- One Idea is to explore the intensity histograms (if there is clear separation)



a b

FIGURE 10.35 Intensity histograms that can be partitioned (a) by a single threshold, and (b) by dual thresholds.

Thresholding: Role of Noise

- Clear separation?

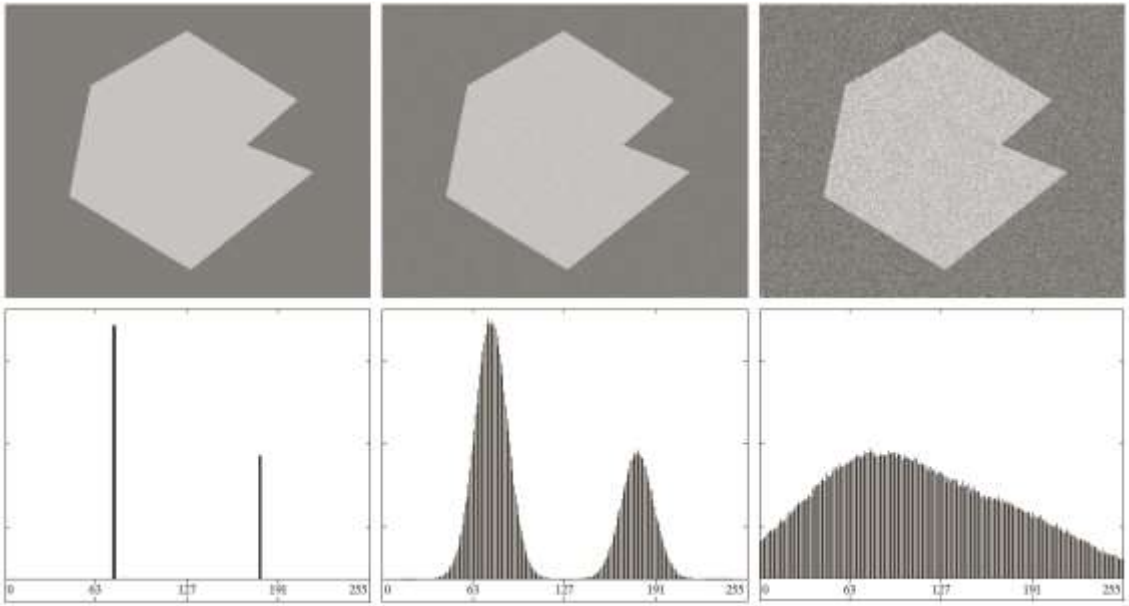
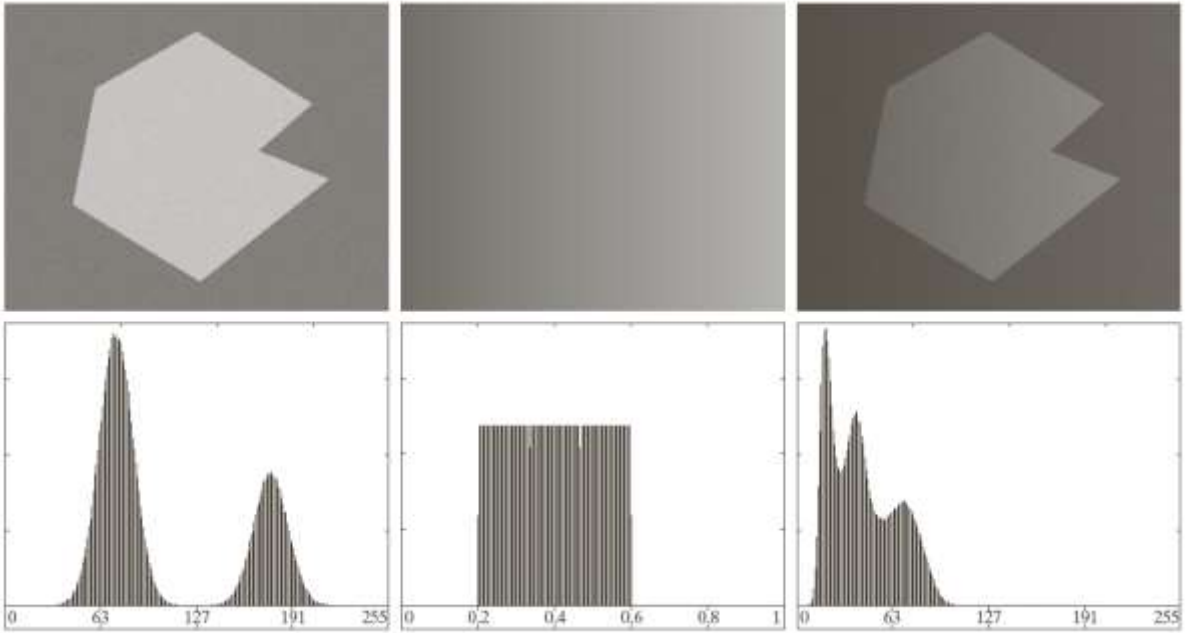


FIGURE 10.36 (a) Noiseless 8-bit image. (b) Image with additive Gaussian noise of mean 0 and standard deviation of 10 intensity levels. (c) Image with additive Gaussian noise of mean 0 and standard deviation of 50 intensity levels. (d)–(f) Corresponding histograms.

Thresholding: Role of Illumination and Reflectance



a b c
d e f

FIGURE 10.37 (a) Noisy image. (b) Intensity ramp in the range [0.2, 0.6]. (c) Product of (a) and (b). (d)–(f) Corresponding histograms.

Choosing a threshold is something of a "black art":

```
n=imread('nodules1.jpg');
figure(1); imshow(n);
n1=im2bw(n,0.35);
n2=im2bw(n,0.75);
figure(2), imshow(n1);
figure(3), imshow(n2);
```

n: Original image n1: Threshold too low n2: Threshold too high

Finding T: Basic Global Thresholding

Iterative approach

1. Select an initial estimate of global threshold T
2. Segment the image using T
   - This will produce two groups of pixels (G1 and G2)
3. Compute the average (mean) intensity values m1 and m2 for the pixels in G1 and G2 respectively
4. Compute a new threshold value T_new = (m1+m2)/2
5. If |T_new – T| < eps, stop.
6. Else, set T = T_new. Go to Step 2.

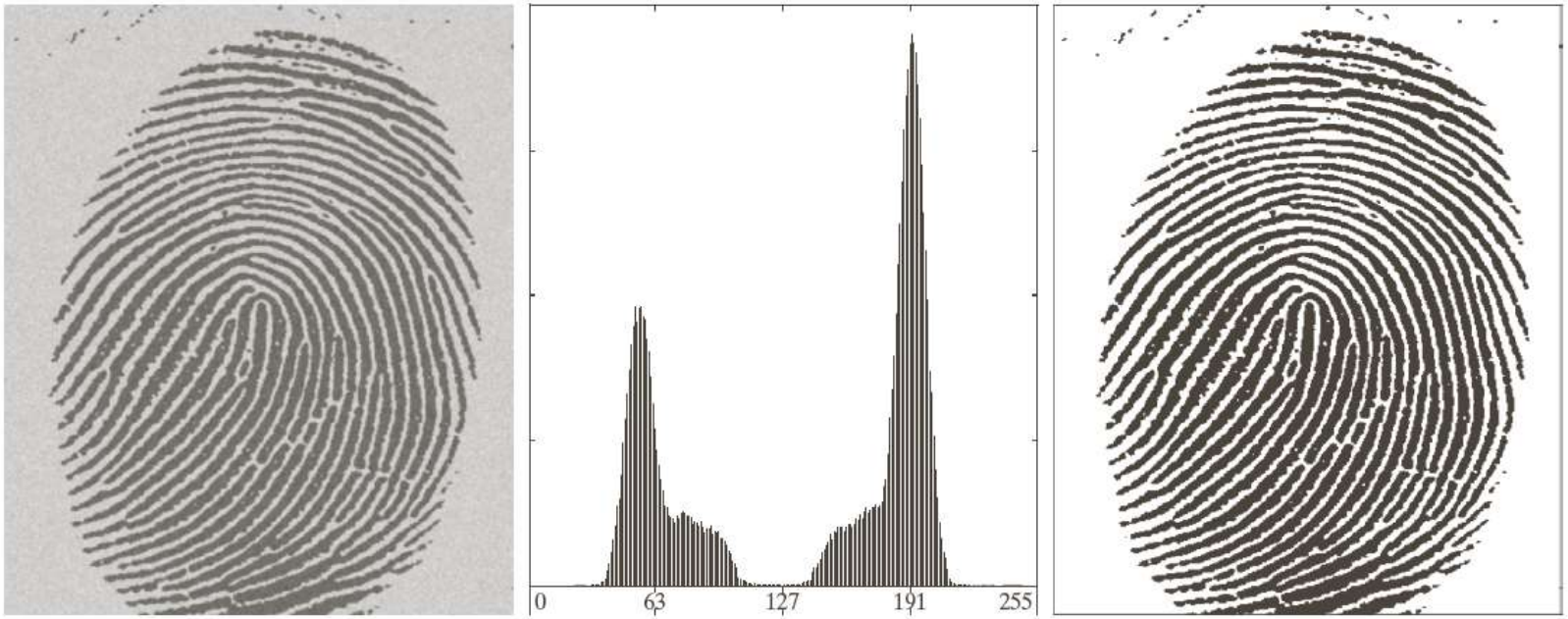
Finding T: Basic Global Thresholding

Iterative approach

1. Select an initial estimate of global threshold T
2. Segment the image using T
   - This will produce two groups of pixels (G1 and G2)
3. Compute the average (mean) intensity values m1 and m2 for the pixels in G1 and G2 respectively
4. Compute a new threshold value T_new = (m1+m2)/2
5. If |T_new – T| < eps, stop.
6. Else, set T = T_new. Go to Step 2.

Matlab function: `opthr`

Basic Global Thresholding

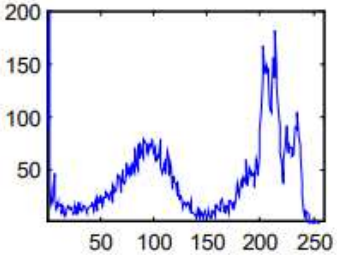


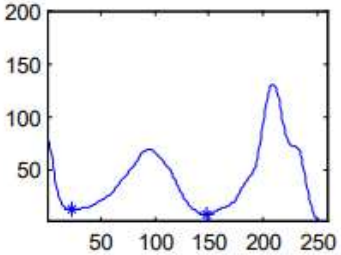
a b c

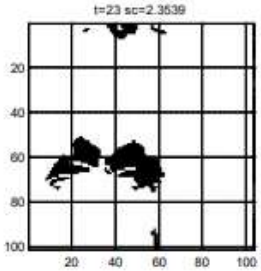
FIGURE 10.38 (a) Noisy fingerprint. (b) Histogram. (c) Segmented result using a global threshold (the border was added for clarity). (Original courtesy of the National Institute of Standards and Technology.)

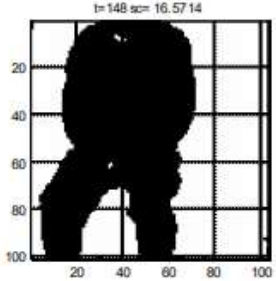
Global Thresholding

T is usually located at the valley/ one of the valleys

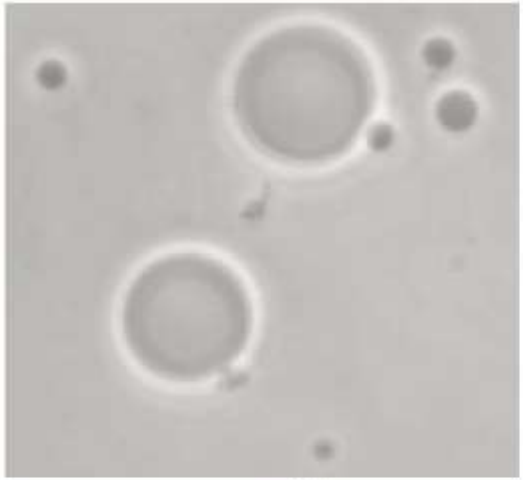


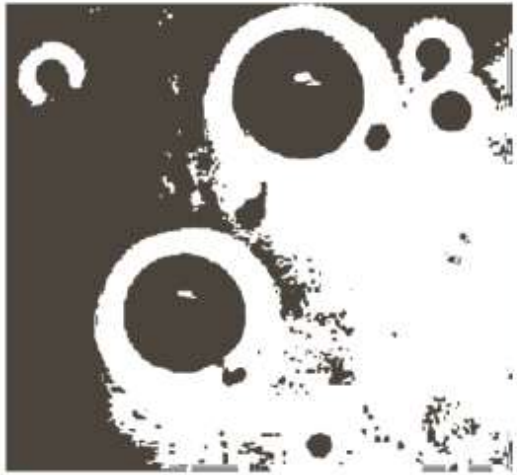


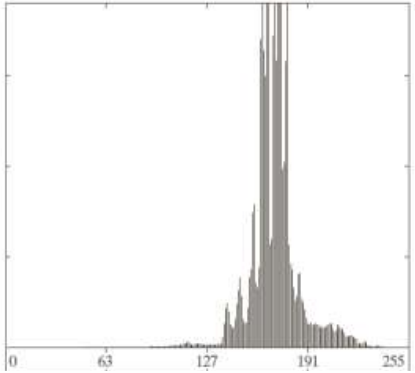




Basic Global Thresholding

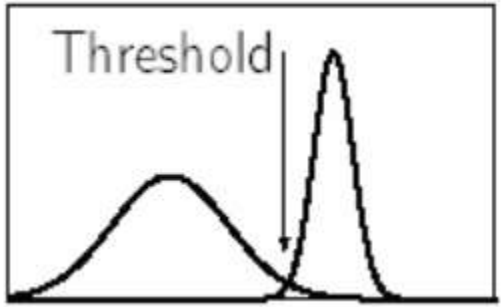


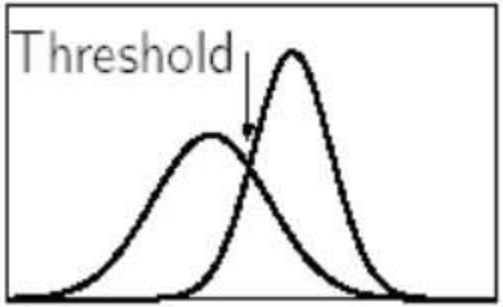




- No valleys
- BG intensity close to ROI (region-of-interest) intensity

Determining FG/BG separation





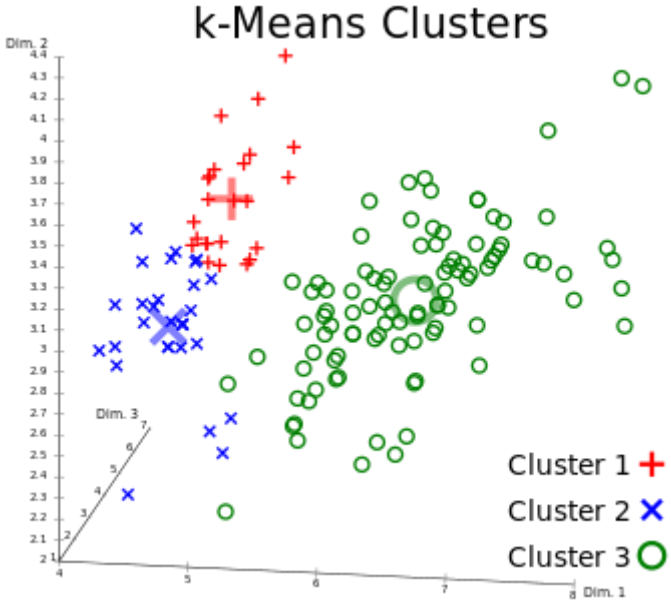
Global Thresholding: Otsu's Method

- Based on histograms
- Automatically finds the optimal threshold <u>maximizing the between class variance</u>
- Proposed in 1975

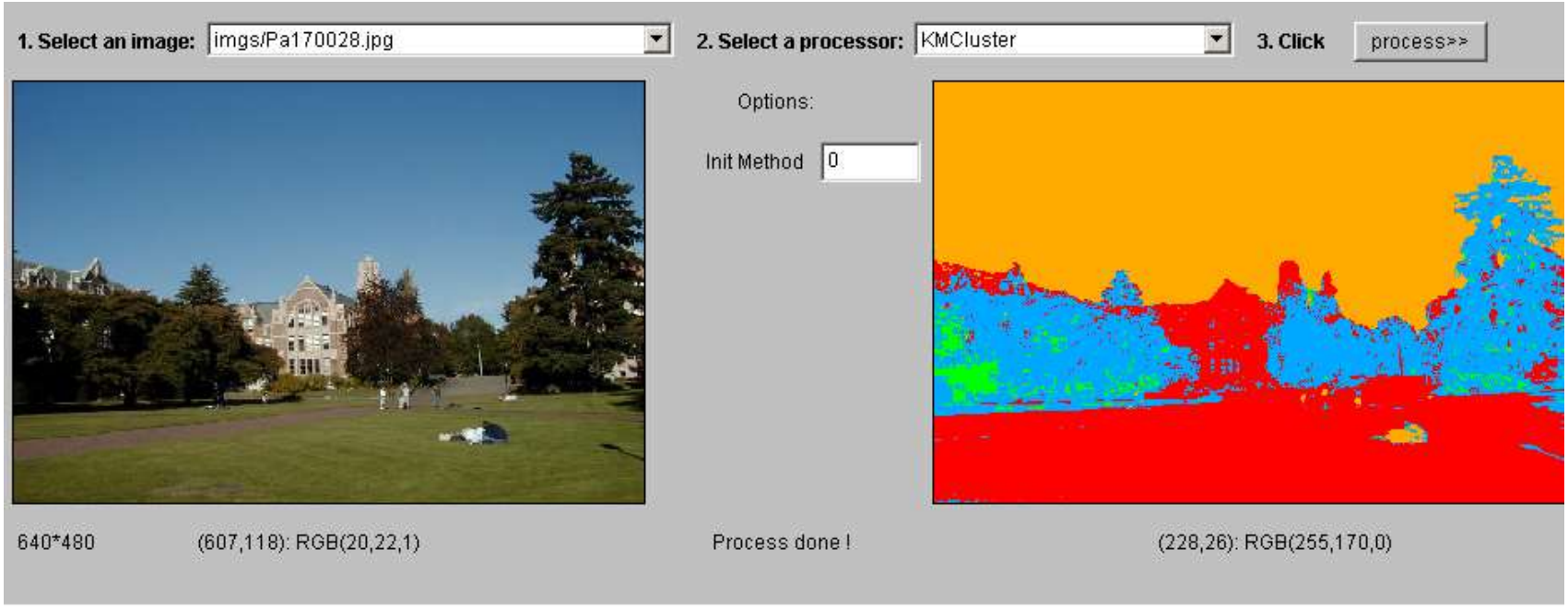
- Preliminaries:
  - What is the formula for mean and variance of intensities in an image ?
  - What does variance measure ?
  - A probabilistic / normalized-histogram perspective for mean, variance

Clustering

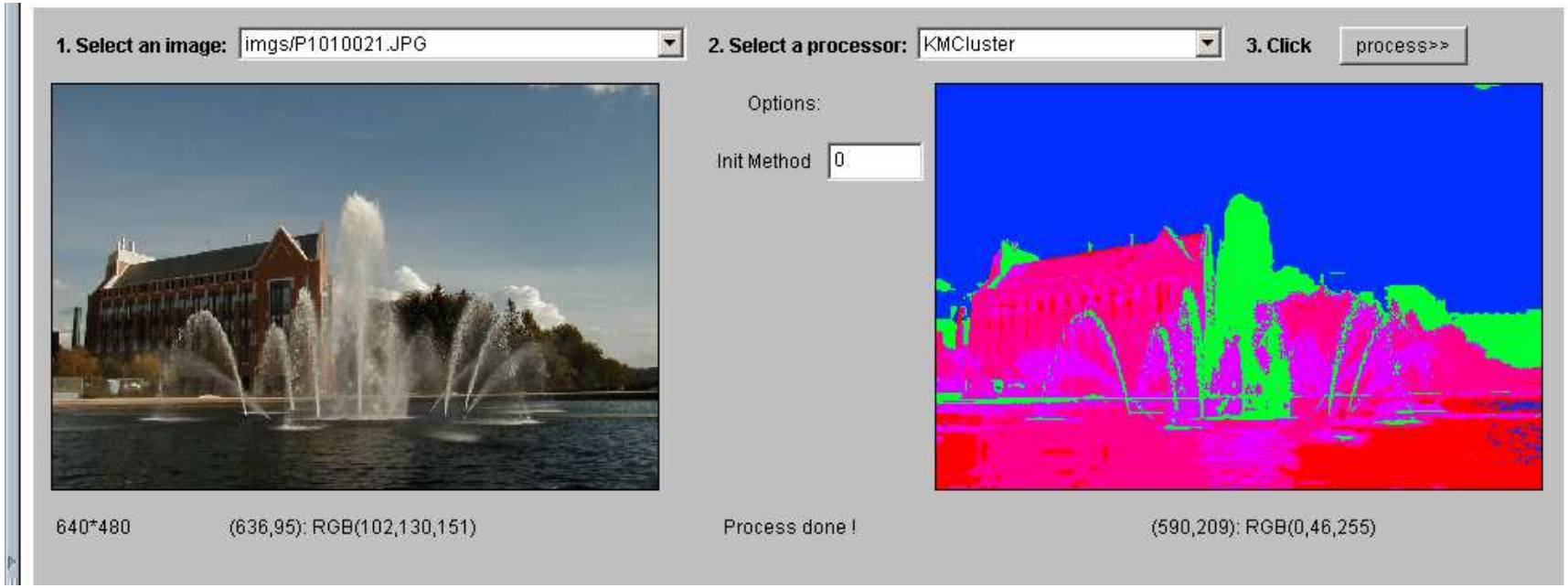
- Clustering using extracted features + spatial coordinates
  - Features like color, texture etc.
  - Algorithms like k-means, agglomerative clustering, spectral clustering etc.



K-Means Example 1



K-Means Example 2



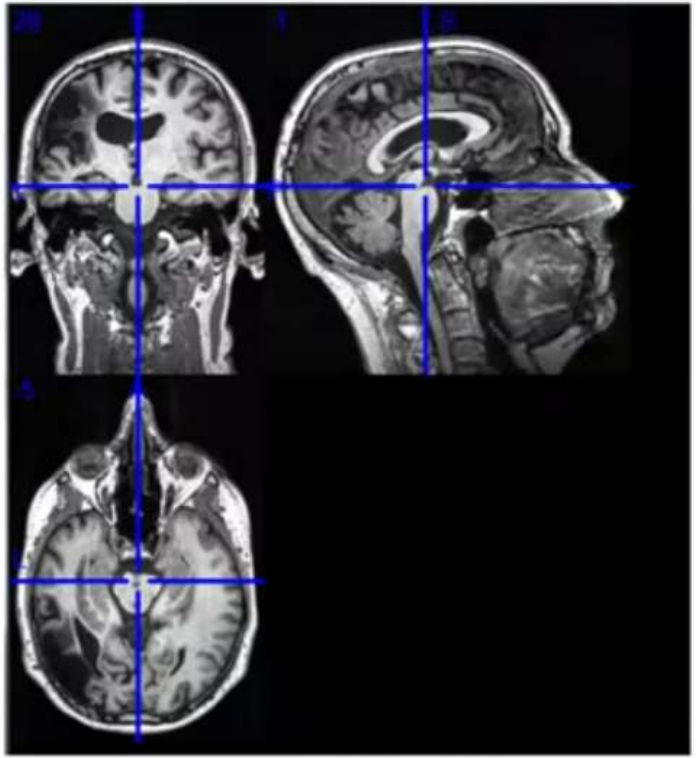
Clustering

- Clustering using extracted features + spatial coordinates
  - Features : color, texture etc.
  - Algorithms : k-means, agglomerative clustering, spectral clustering etc.

Clustering

- Clustering using extracted features + spatial coordinates
  - Features : color, texture etc.
  - Algorithms : k-means, agglomerative clustering, spectral clustering etc.

Clustering-Medical Images



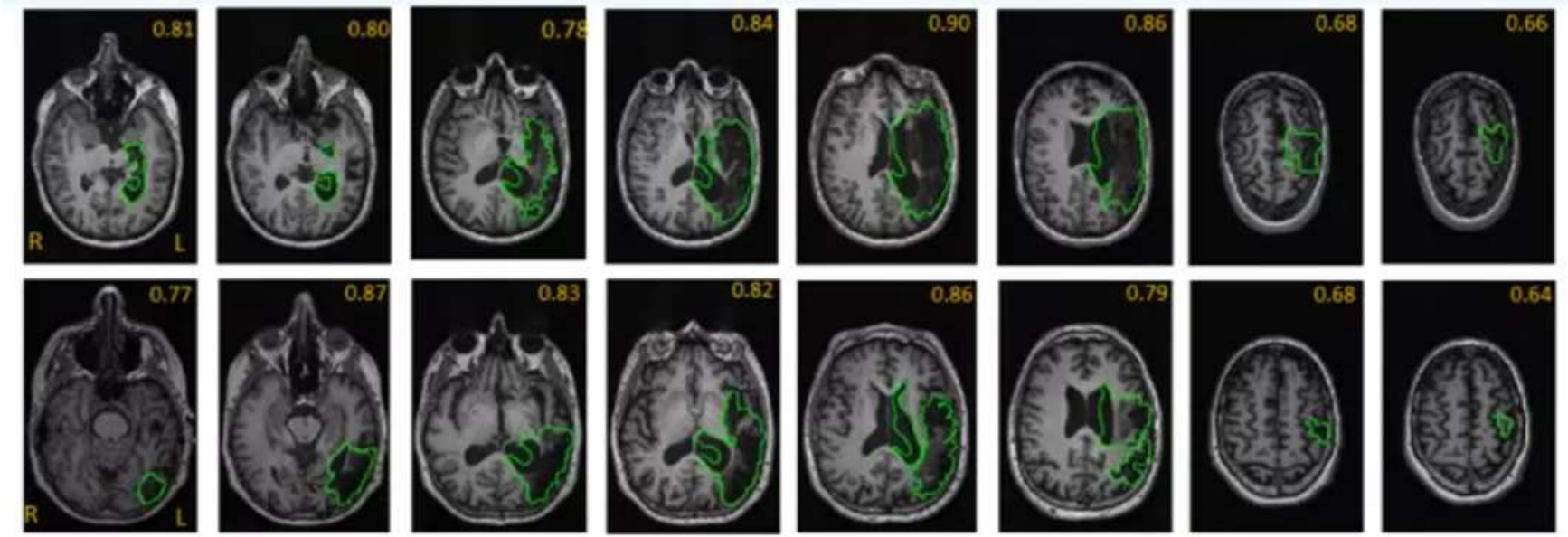
Voxels i.e 3D pixels

Clustering in 4 dimensional space i.e
[x, y, z, I(x,y,z)]

There are three groups: White matter, Gray matter and CSF

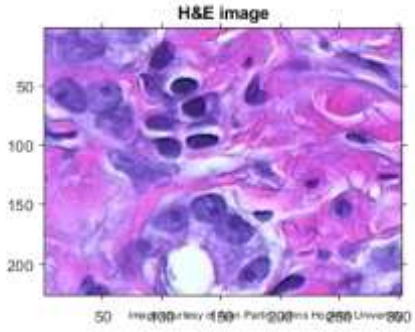
Source: Aggelos K. Katsaggelos

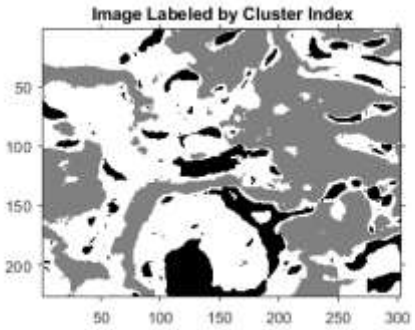
Clustering-Medical Images

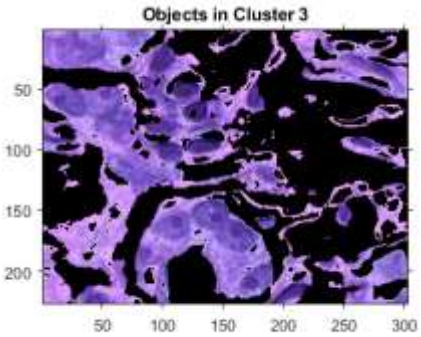


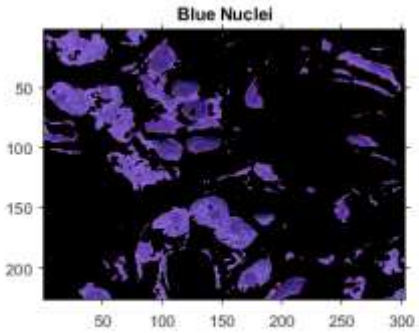
Important both for diagnosis + analysis of recovery

Source: Aggelos K. Katsaggelos









RGB -> L*a*b

Clustering on a,b

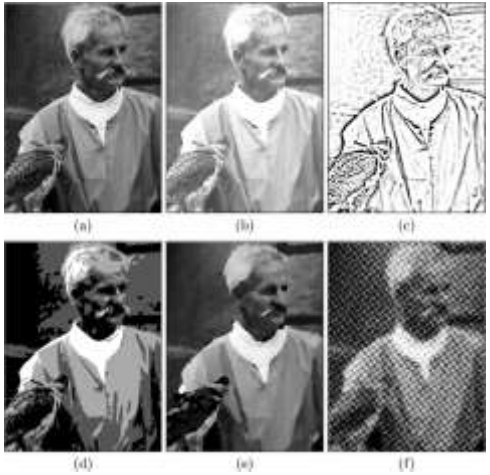
Get intensity

Threshold L

https://in.mathworks.com/help/images/color-based-segmentation-using-k-means-clustering.html

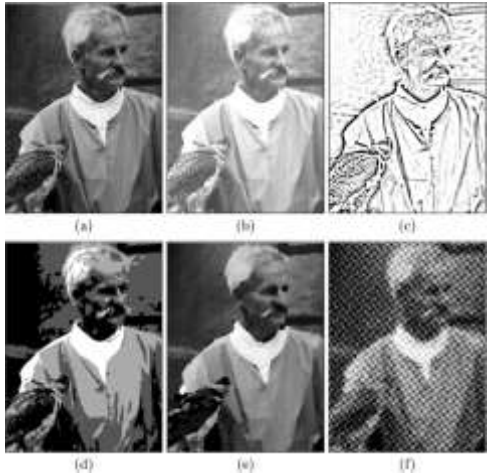
How to tell if 2 Images are same?

- Pixel by pixel comparison?
  - Makes sense only if pictures taken from same angle, same lighting, etc
- Noise, quantization, etc introduces differences
  - Human may say images are same even with numerical differences



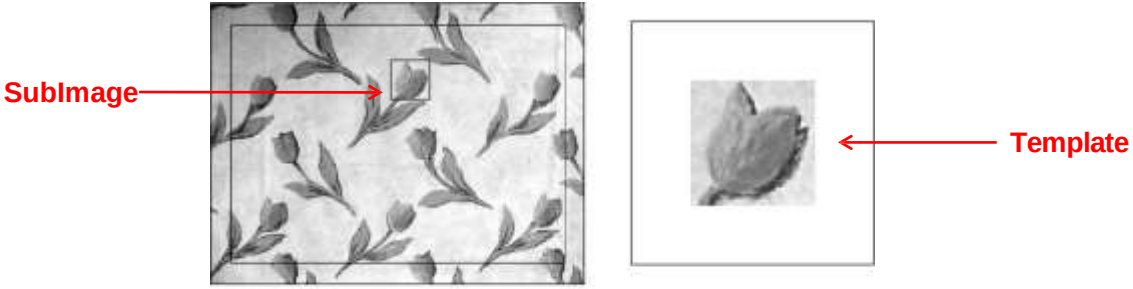
Comparing Images

- Better approach: Template matching
  - Identify similar sub-images (called template) within 2 images
- Applications?
  - Match left and right picture of stereo images
  - Find particular pattern in scene
  - Track moving pattern through image sequence



Template Matching

- Basic idea
  - Move given pattern (template) over search image
  - Measure difference between template and sub-images at different positions
  - Record positions where highest similarity is found



(a) original image I (b) reference image R

Template Matching

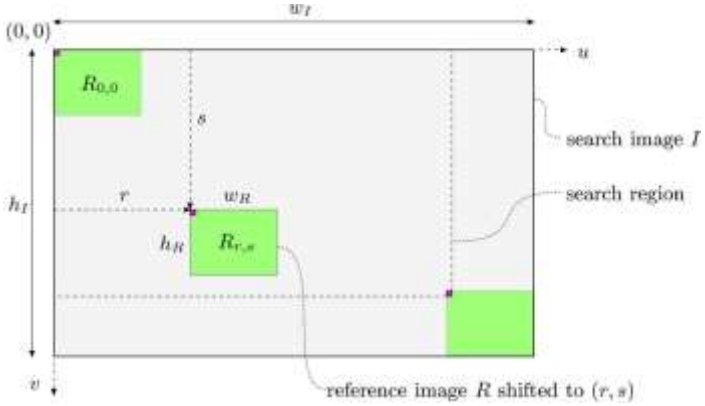
- Difficult issues?
  - What is distance (difference) measure?
  - What levels of difference should be considered a match?
  - How are results affected when brightness or contrast changes?

(a) original image I (b) reference image R

Template Matching in Intensity Images

- We can restate template matching problem as:
- Finding the offset *(r,s)* such that the similarity between the shifted reference image $R_{r,s}$ and corresponding subimage I is a maximum

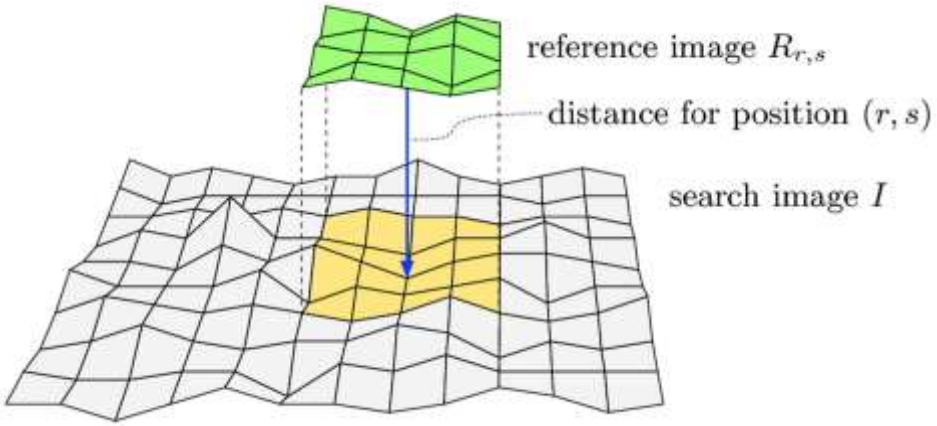
$$R_{r,s}(u,v) = R(u-r, v-s)$$



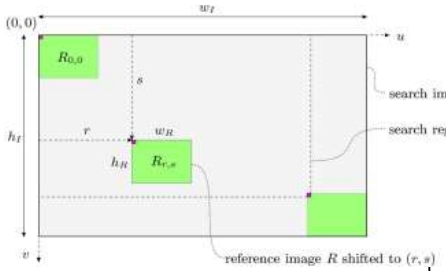
- Solving this problem involves solving many sub-problems

Distance between Image Patterns

- Many measures proposed to compute distance between the shifted reference image $R_{r,s}$ and corresponding subimage I



Distance between Image Patterns



- Many measures proposed to compute distance between the shifted reference image $R_{r,s}$ and corresponding subimage I
- Sum of absolute differences:

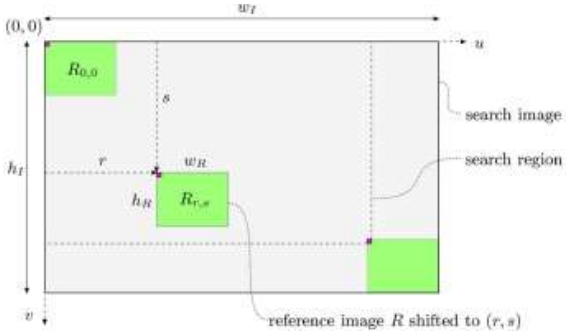
$$\mathrm{d}_A(r,s) = \sum_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

- Maximum difference:

$$\mathrm{d}_M(r,s) = \max_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

- Sum of squared differences (also called N-dimensional Euclidean distance):

$$\mathrm{d}_E(r,s) = \Big[\sum_{(i,j)\in R} \big(I(r+i, s+j) - R(i,j)\big)^2 \Big]^{1/2}$$



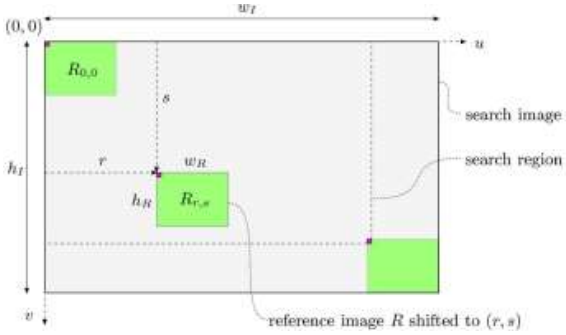
Distance and Correlation

- Best matching position between shifted reference image $R_{r,s}$ and subimage I minimizes square of d_E which can be expanded as

$$
\begin{aligned}
d_E^2(r,s) &= \sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2 \\
&= \underbrace{\sum_{(i,j)\in R} I^2(r+i,s+j)}_{A(r,s)} + \underbrace{\sum_{(i,j)\in R} R^2(i,j)}_{B} - \underbrace{2\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}_{C(r,s)}
\end{aligned}
$$

- *B* term is a constant, independent of *r, s* and can be ignored
- *A* term is sum of squared values within subimage *I* at current offset *r, s*

Distance and Correlation



$$\mathrm{d}_E^2(r,s) = \sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2$$

$$= \underbrace{\sum_{(i,j)\in R} I^2(r+i,s+j)}_{A(r,s)} + \underbrace{\sum_{(i,j)\in R} R^2(i,j)}_{B} - \underbrace{2\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}_{C(r,s)}$$

- *C(r,s)* term is **linear cross correlation** between *I* and *R* defined as

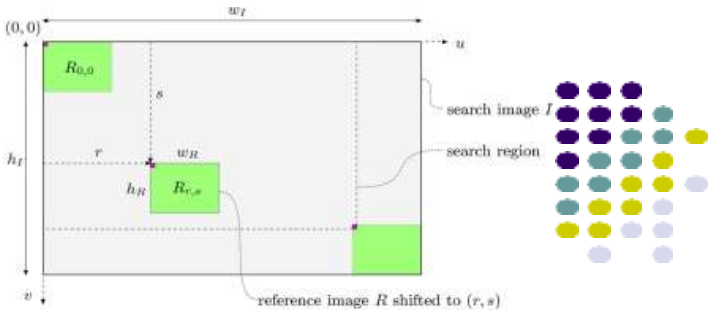
$$(I \circledast R)(r,s) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(r+i,s+j)\cdot R(i,j)$$

- Since *R* and *I* are assumed to be zero outside their boundaries

$$\sum_{i=0}^{w_R-1} \sum_{j=0}^{h_R-1} I(r+i,s+j)\cdot R(i,j) = \sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)$$

- If A constant, Min value of $d_E^2(r,s)$ corresponds to max value of $(I \circledast R)(r,s)$

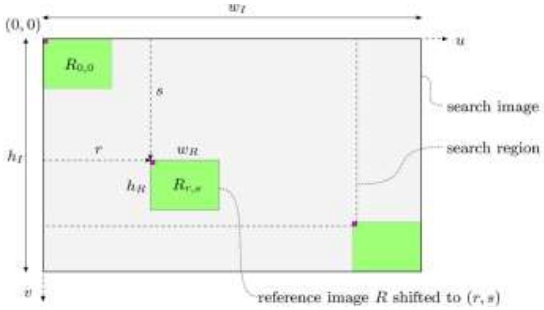
$$
\begin{aligned}
\mathrm{d}_E^2(r,s) &= \sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2 \\
&= \underbrace{\sum_{(i,j)\in R} I^2(r+i,s+j)}_{A(r,s)} + \underbrace{\sum_{(i,j)\in R} R^2(i,j)}_{B} - 2\underbrace{\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}_{C(r,s)}
\end{aligned}
$$



- Unfortunately, *A* term is not constant in most images
- Range of C(r,s) not bounded. Dependent on size of template.
- Not invariant to changes in intensity amplitude.

$$d_E^2(r,s) = \sum_{(i,j)\in R} \left(I(r+i,s+j) - R(i,j)\right)^2$$

$$= \underbrace{\sum_{(i,j)\in R} I^2(r+i,s+j)}_{A(r,s)} + \underbrace{\sum_{(i,j)\in R} R^2(i,j)}_{B} - 2\underbrace{\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}_{C(r,s)}$$



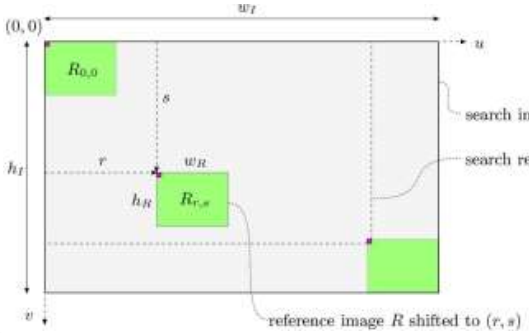
Normalized Cross Correlation

$$C_N(r,s) =$$

$$= \frac{\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}{\left[\sum_{(i,j)\in R} I^2(r+i,s+j)\right]^{1/2} \cdot \left[\sum_{(i,j)\in R} R^2(i,j)\right]^{1/2}}$$

- $C_N(r,s)$ is a local distance measure, is in [0,1] range
- $C_N(r,s) = 1$ indicates maximum match
- $C_N(r,s) = 0$ indicates images are very dissimilar

Correlation Coefficient



- **Correlation coefficient:** Use differences between *I* and *R* and their average values

$$C_L(r,s) = \frac{\sum_{(i,j)\in R} \left(I(r+i,s+j) - \bar{I}(r,s)\right) \cdot \left(R(i,j) - \bar{R}\right)}{\left[\sum_{(i,j)\in R} \left(I(r+i,s+j) - \bar{I}_{r,s}\right)^2\right]^{1/2} \cdot \underbrace{\left[\sum_{(i,j)\in R} \left(R(i,j) - \bar{R}\right)^2\right]^{1/2}}_{S_R^2 = K \cdot \sigma_R^2}}$$

where the average values are defined as

$$\bar{I}_{r,s} = \frac{1}{K} \cdot \sum_{(i,j)\in R} I(r+i,s+j) \quad \text{and} \quad \bar{R} = \frac{1}{K} \cdot \sum_{(i,j)\in R} R(i,j)$$

- *K* is number of pixels in reference image *R*
- $C_L(r,s)$ can be rewritten as

$$C_L(r,s) = \frac{\sum_{(i,j)\in R} \left(I(r+i,s+j) \cdot R(i,j)\right) - K \cdot \bar{I}_{r,s} \cdot \bar{R}}{\left[\sum_{(i,j)\in R} I^2(r+i,s+j) - K \cdot \bar{I}_{r,s}^2\right]^{1/2} \cdot S_R}$$

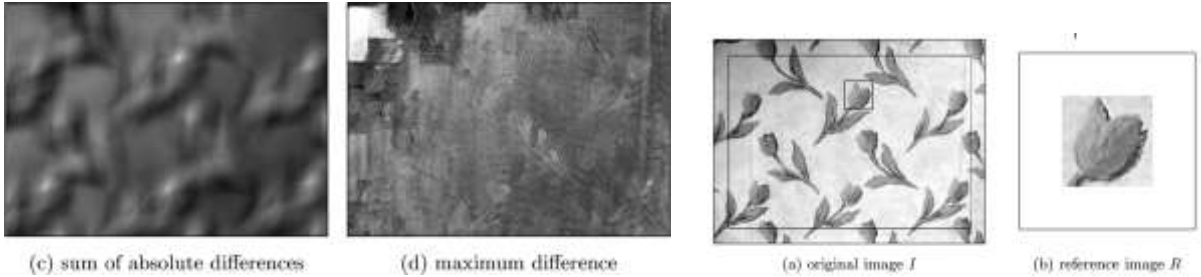
Examples and Discussion

- We now compare these distance metrics
- **Original image *I*:** Repetitive flower pattern
- **Reference image *R*:** one instance of repetitive pattern extracted from *I*

(a) original image I (b) reference image R

- Now compute various distance measures for this *I* and *R*

Examples and Discussion


(c) sum of absolute differences
(d) maximum difference
(a) original image I
(b) reference image R

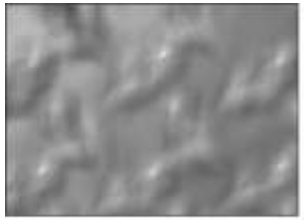
- **Sum of absolute differences:** performs okay but affected by global intensity changes

$$\mathrm{d}_A(r,s) = \sum_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

- **Maximum difference:** Responds more to lighting intensity changes than pattern similarity

$$\mathrm{d}_M(r,s) = \max_{(i,j)\in R} |I(r+i, s+j) - R(i,j)|$$

Examples and Discussion



(e) sum of squared distances

(f) global cross correlation

(a) original image I

(b) reference image R

- **Sum of squared (euclidean) distances:** performs okay but affected by global intensity changes

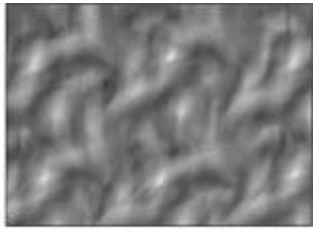
$$\mathrm{d}_E(r,s) = \Big[\sum_{(i,j)\in R} \big(I(r+i,s+j) - R(i,j)\big)^2 \Big]^{1/2}$$

- **Global cross correlation:** Local maxima at true template position, but is dominated by high-intensity responses in brighter image parts

$$(I \circledast R)(r,s) = \sum_{i=-\infty}^{\infty} \sum_{j=-\infty}^{\infty} I(r+i,s+j) \cdot R(i,j)$$

Examples and Discussion

(g) normalized cross correlation



(h) correlation coefficient

(a) original image I

(b) reference image R

- **Normalized cross correlation:** results similar to euclidean distance (affected by global intensity changes)

$$\frac{\sum_{(i,j)\in R} I(r+i,s+j)\cdot R(i,j)}{\left[\sum_{(i,j)\in R} I^2(r+i,s+j)\right]^{1/2}\cdot\left[\sum_{(i,j)\in R} R^2(i,j)\right]^{1/2}}$$

- **Correlation coefficient:** yields best results. Distinct peaks produced for all 6 template instances, unaffected by lighting

$$C_L(r,s)=\frac{\sum_{(i,j)\in R}\left(I(r+i,s+j)\cdot R(i,j)\right)\;-\;K\cdot\bar{I}_{r,s}\cdot\bar{R}}{\left[\sum_{(i,j)\in R} I^2(r+i,s+j)\;-\;K\cdot\bar{I}^2_{r,s}\right]^{1/2}\cdot S_R}$$

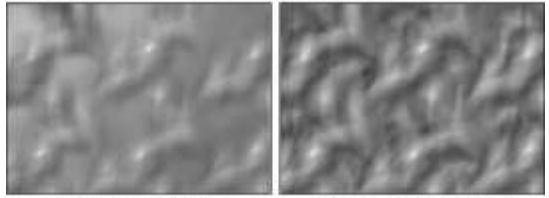
Effects of Changing Intensity

- To explore effects of globally changing intensity, raise intensiy of reference image *R* by 50 units
- Distinct peaks disappear in **Euclidean distance**
- **Correlation coefficient** unchanged, robust measure in realistic lighting conditions

(a) original image I

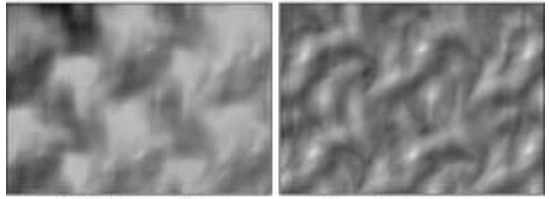
(b) reference image R

Original reference image: R



(a) Euclidean distance (b) correlation coefficient

Modified reference image: $R' = R + 50$



(c) Euclidean distance (d) correlation coefficient

Shape of Template

- Template does not have to be rectangular
- Some applications use circular, elliptical or custom-shaped templates
- Non-rectangular templates stored in rectangular array, but pixels in template marked using a mask
- More generally, a weighted function can be applied to template elements

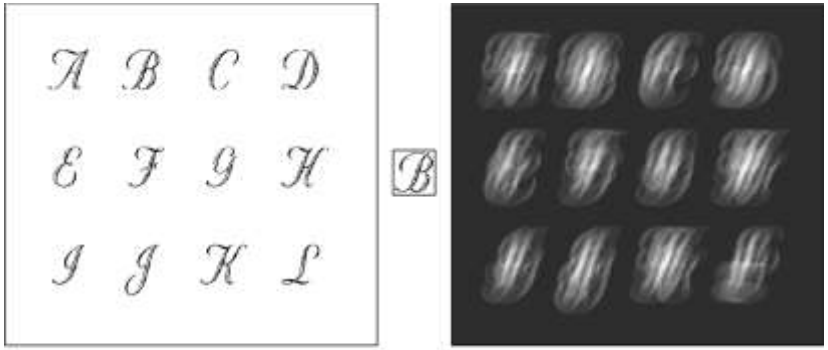
Matching under Rotation and Scaling

- **Simple Approach:**
  - Store multiple rotated and scaled versions of template
  - Computationally prohibitive

- Alternate approaches:
  - Matching in logarithmic-polar space (complicated!)
  - Affine matching: Use local statistical features invariant under affine image transformations (including rotation and scaling)

Matching Binary Images

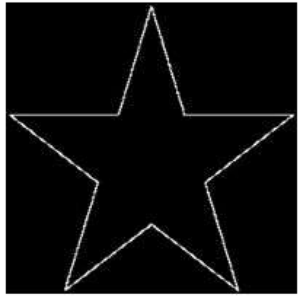
- **Direct Comparison:**
  - Count the number of identical pixels in search image and template
  - Small total difference when most pixels are same
- Problem: Small shift, rotation or distortion of image create high distance
- Need a more tolerant measure



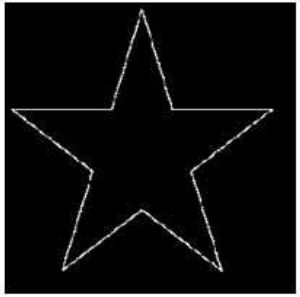
(a) (b) (c)

Detecting Shapes Without Texture

- Normalized correlation does not work well.
- Slight misalignments have a great impact on the correlation score.

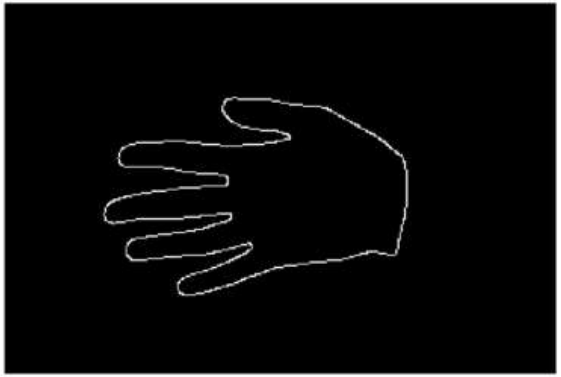


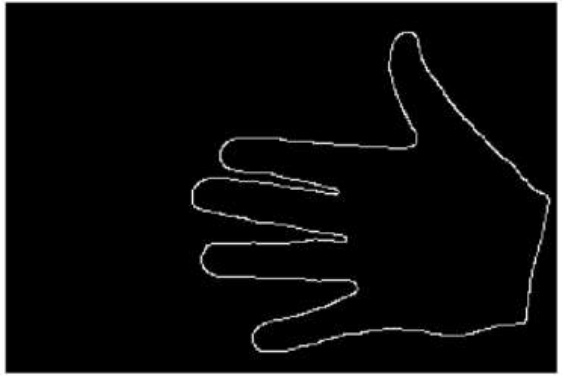
star1



star3

combined





The Distance Transform

- For every position (u,v) in the search image *I*, record distance to closest foreground pixel
- Recall: In binary image, convention is

$$FG(I) = \{\boldsymbol{p} \mid I(\boldsymbol{p}) = 1\}$$
$$BG(I) = \{\boldsymbol{p} \mid I(\boldsymbol{p}) = 0\}$$

- Distance transform is defined as

$$D(\boldsymbol{p}) = \min_{\boldsymbol{p}' \in FG(I)} \mathrm{dist}(\boldsymbol{p}, \boldsymbol{p}')$$

- Examples of distance measures are **Euclidean distance**

$$\mathrm{d}_E(\boldsymbol{p}, \boldsymbol{p}') = \|\boldsymbol{p} - \boldsymbol{p}'\| = \sqrt{(u-u')^2 + (v-v')^2} \quad \in \mathbb{R}^+$$

- Or **Manhattan distance**

$$\mathrm{d}_M(\boldsymbol{p}, \boldsymbol{p}') = |u - u'| + |v - v'| \quad \in \mathbb{N}_0$$

Distance Transform Example

- Example using Manhattan distance

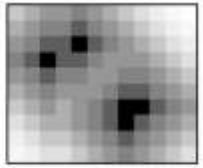
binary image

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	0	0	0	**1**	0	0	0	0	0	0	0
0	0	**1**	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	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	**1**	**1**	0	0	0
0	0	0	0	0	0	0	**1**	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	0	0	0

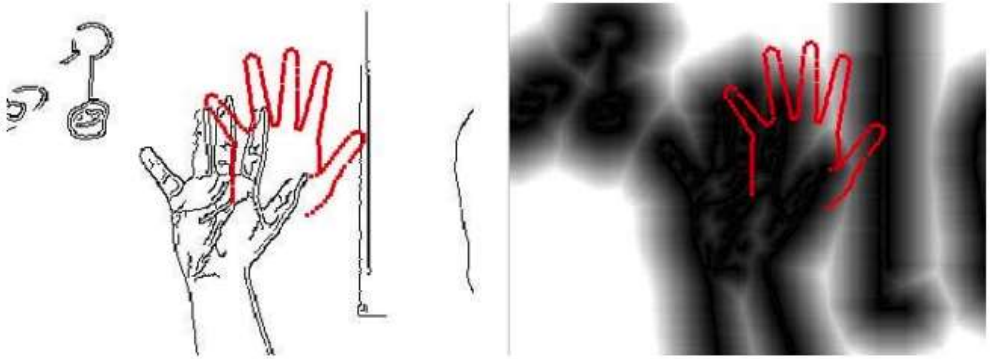
$\rightarrow$

distance transform

5	4	3	3	2	3	4	5	6	7	8	9
4	3	2	2	1	2	3	4	5	6	7	8
3	2	1	1	**0**	1	2	3	4	5	6	7
2	1	**0**	1	1	2	3	3	3	4	5	6
3	2	1	2	2	3	3	2	2	3	4	5
4	3	2	3	3	3	2	1	1	2	3	4
5	4	3	4	3	2	1	**0**	**0**	1	2	3
6	5	4	4	3	2	1	**0**	1	2	3	4
7	6	5	5	4	3	2	1	2	3	4	5
8	7	6	6	5	4	3	2	3	4	5	6



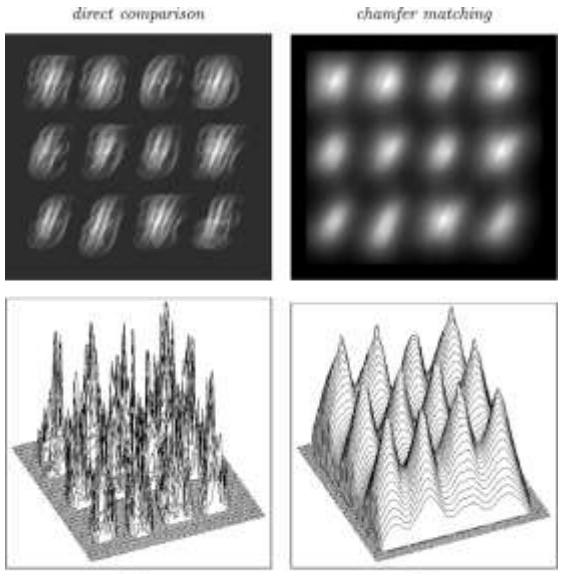
Chamfer Matching



- Chamfer score is average nearest distance from template points to image points
- Nearest distances are readily obtained from the distance image
- Computationally inexpensive

Comparing Direct Pixel comparison and Chamfer Matching

- Chamfer match score Q much smoother than direct comparison
  - Distinct peaks in places of high similarity



(a) (b)

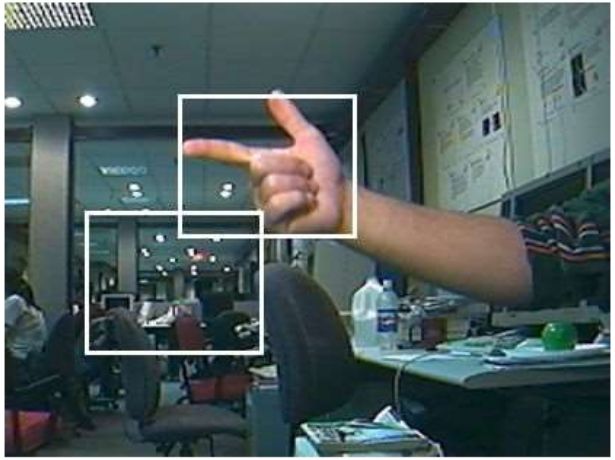
- Chamfer Matching
  - Variant to scale and rotation
  - Sensitive to small shape changes
  - Need large number of template shapes

  But
  - Robust to clutter
  - Computationally cheap

Reducing the cost of correlation matching

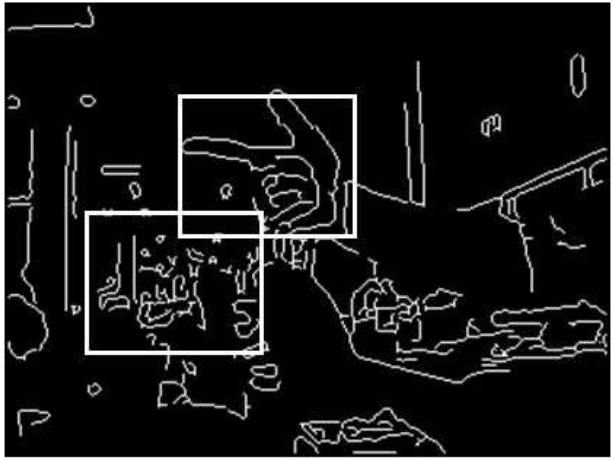
- A number of factors lead to large costs in correlation matching:
  - the image N is much larger than the template M, so we have to perform correlation matching of M against every nxn window of N

Results for Single Scale Search



- What is causing the false result?

Results for Single Scale Search



- What is causing the false result?
  - Window with lots of edges.
- How can we refine these results?