11.01.2025

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY
HYDERABAD

Digital Image Processing – Recap (Part 2)

Ravi Kiran, CVIT

Makarand Tapaswi, CVIT

CVIT

Administrivia

- Make sure you have all accepted A0 on Github Classroom. If you have any issues, get in touch with Sanika (TA).
- A1 will be released today (Jan 11th Saturday). Submission due: 22nd Jan (look out for Moodle announcement)

Announcements

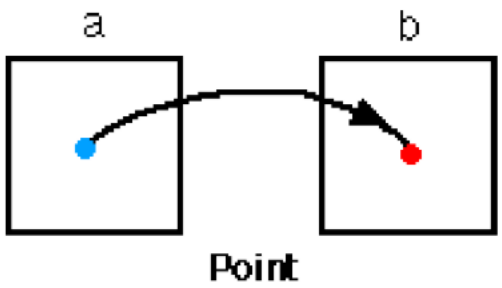
- **Update** on one-time bonus rule for assignments
  - You are allowed to use a maximum of 3 days for assignment extensions
  - NEW: You can split them across assignments, BUT each usage needs to be at least 1 day
    - ➔ you can use the assignment extension bonus for at most 3 of your assignments
  - If you fill the extension form and don't use the bonus days, we will still count it. Don't fill the form unless you are absolutely sure !
  - Any email ids other than your official IIIT id will be ignored

Image Processing – Two Paradigms

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

Spatial Domain Processing

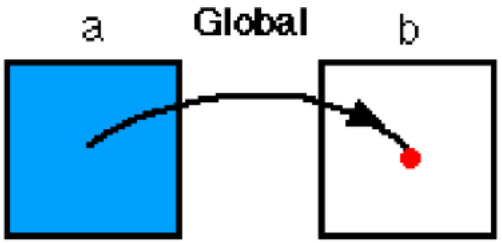
- Manipulating Pixels Directly in Spatial Domain
- 3 approaches
- 1. Point to Point



Spatial Domain Processing

- Manipulating Pixels Directly in Spatial Domain
- 3 approaches

- 3. Global to Point



Histogram

$$h_r(i) = n_i$$

i → intensity value, range [0,L-1]

n_i → number of pixels with intensity i

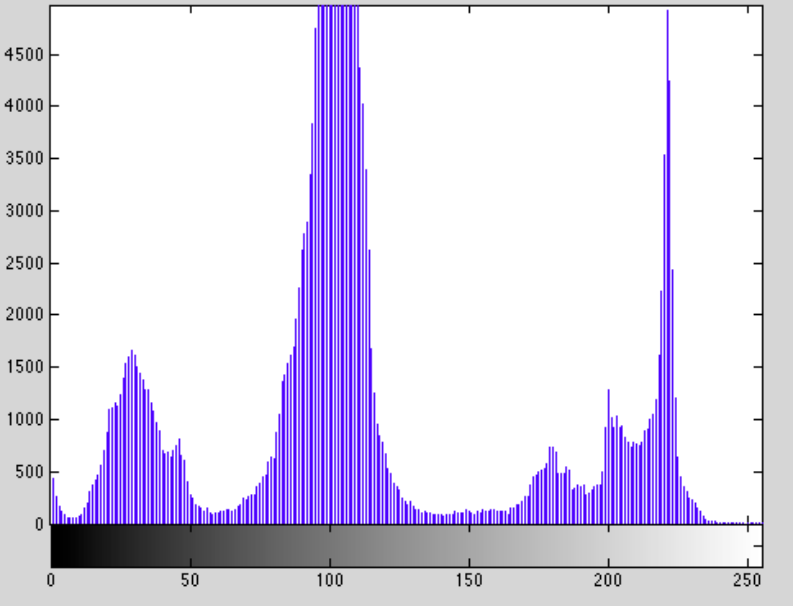
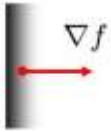


Image gradient

The gradient of an image:

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

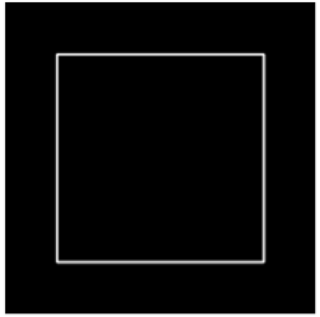
The gradient points in the direction of most rapid change in intensity



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

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

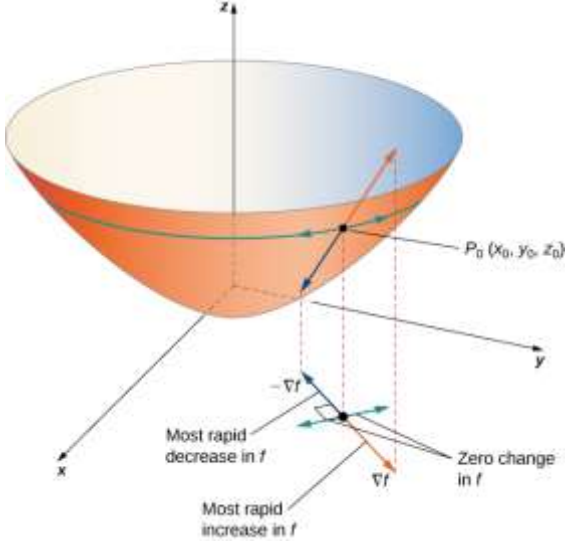
$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$



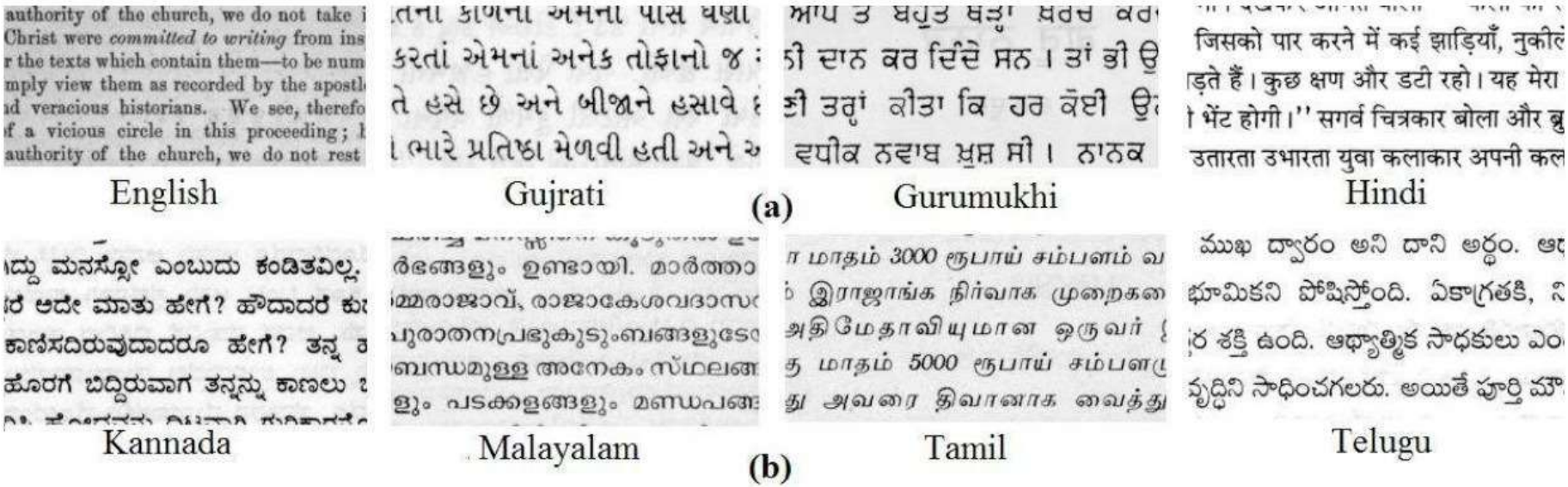
Image

Gradient in x

Gradient in y

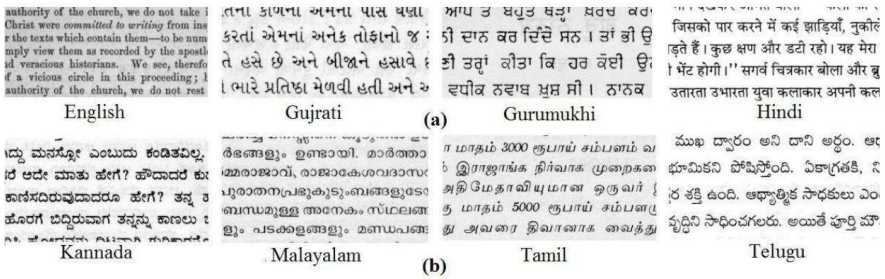


Task: Given a document image, predict the script class

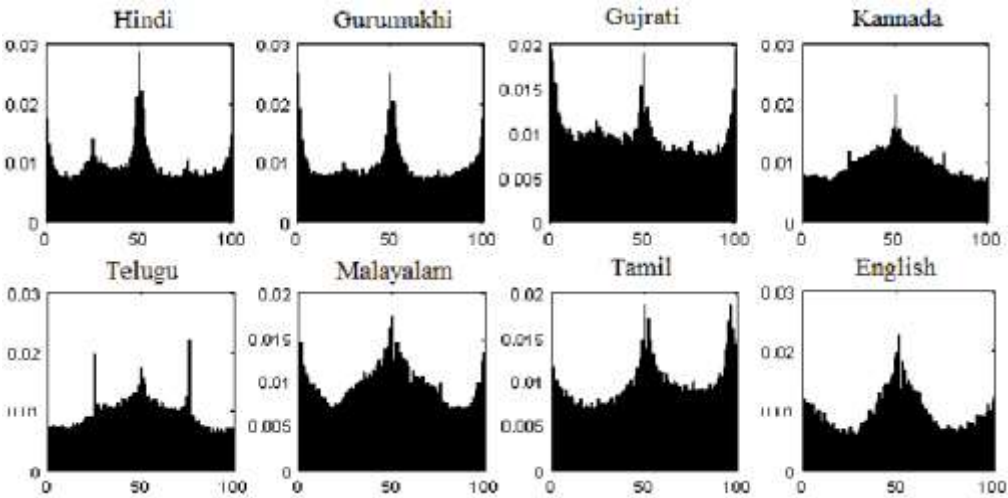


https://www.researchgate.net/publication/234830833_Curvature_feature_distribution_based_classification_of_Indian_scripts_from_document_images

Task: Given a document image, predict the script class



Histogram of edge directions →

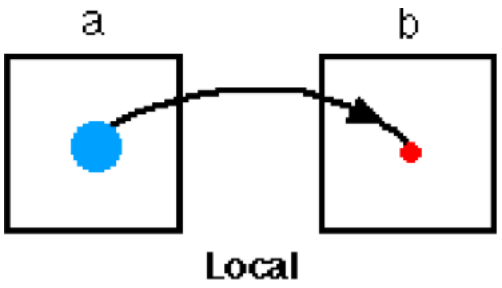


https://www.researchgate.net/publication/234830833_Curvature_feature_distribution_based_classification_of_Indian_scripts_from_document_images

Spatial Domain Processing

- Manipulating Pixels Directly in Spatial Domain
- 3 approaches

- 2. Neighborhood to Point



48	219	168	145	244	189	120	58
49	218	87	94	133	35	17	148
174	151	74	179	224	3	252	194
77	127	87	139	44	228	149	135
138	229	136	113	250	51	108	163
38	210	185	177	69	76	131	53
178	164	79	158	64	169	85	97
96	209	214	203	223	73	110	200

Current pixel

3 × 5 neighbourhood

Spatial Domain Filtering

- Linear
  - Blurring/Smoothing (suppressing edges, high local contrast)
    - Mean, Gaussian
  - Gradient /Edge Detection (emphasizing edges)
    - Prewitt, Sobel (first derivative approximations)
    - Laplacian (second derivative approximations)
    - Highboost/Unsharp (enhance edges)
    - Laplacian of Gaussian (edge detection with noise suppression)
- Non Linear
  - min,max, median
  - Bilateral filter

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

=

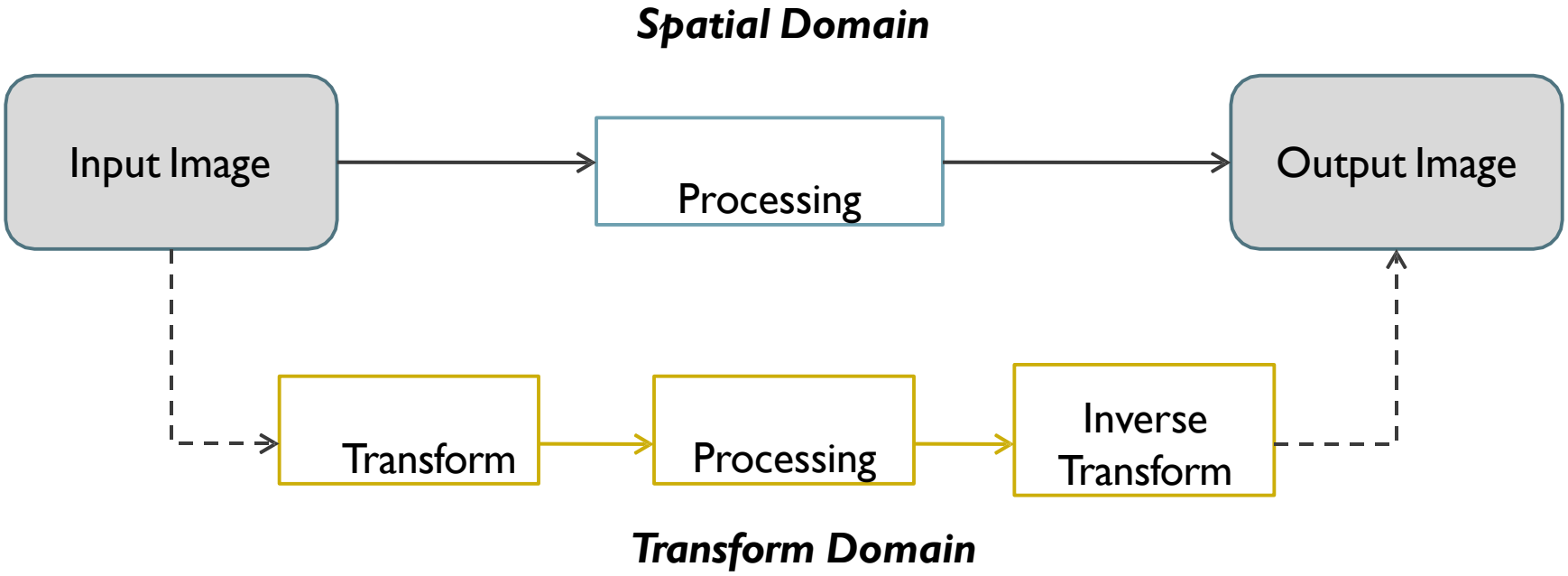
	98			

Image Enhancement

Contrast Stretching
(new intensities
span full range)

Histogram Equalization
(new image histogram is uniform
distribution)

Spatial vs. Transform Domain Processing



Spatial vs. Transform Domain Processing

Bandhani / Bandhej

Tie Dye

Spatial vs. Transform Domain Processing

Transform (Tie)

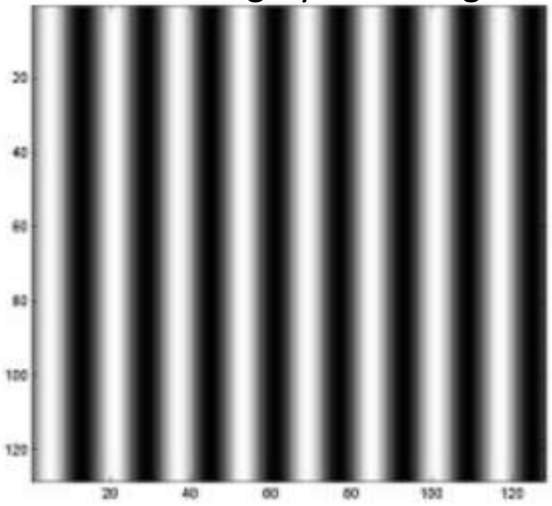
Process (Dye)

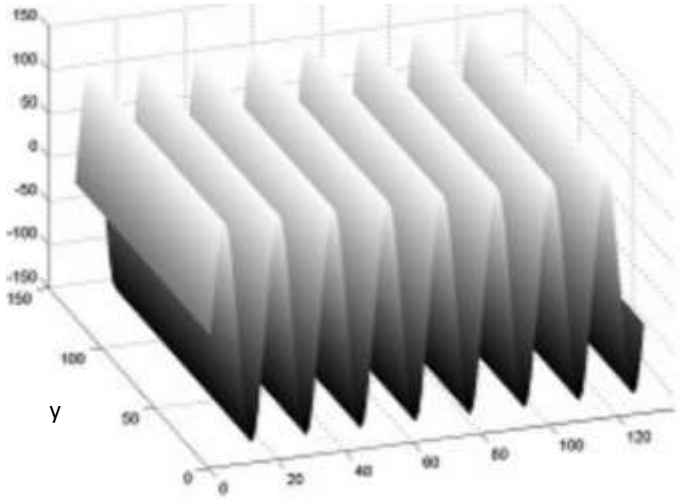
Inverse Transform (Untie)

Periodic Images

$$I(x, y) = 128\ sin(2\pi x/16)$$

128 x 128 grayscale image

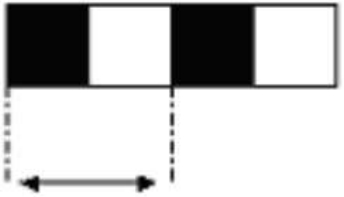




Sinusoid pattern repeats every 16 pixels
f = 1/16 cycles/pixel

Periodic Images

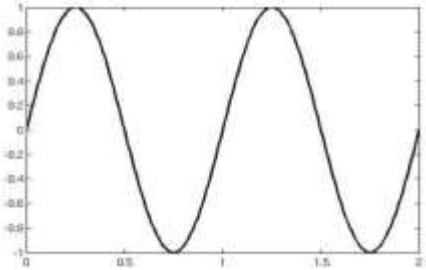
Spatial period = Minimal # of pixels between two identical patterns in a “periodic” image



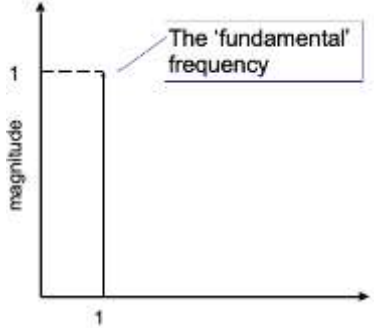
minimal period : 2 pixels

$$\Rightarrow \quad \nu_{max} = \frac{1}{\text{minimal period}} = \frac{1}{2}$$

$$y = \sin(2\pi s)$$

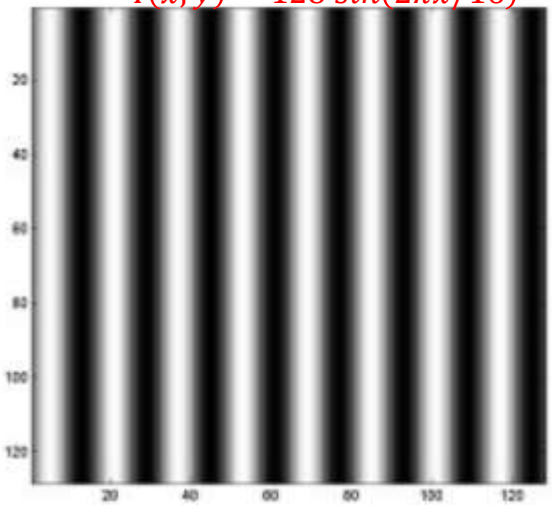


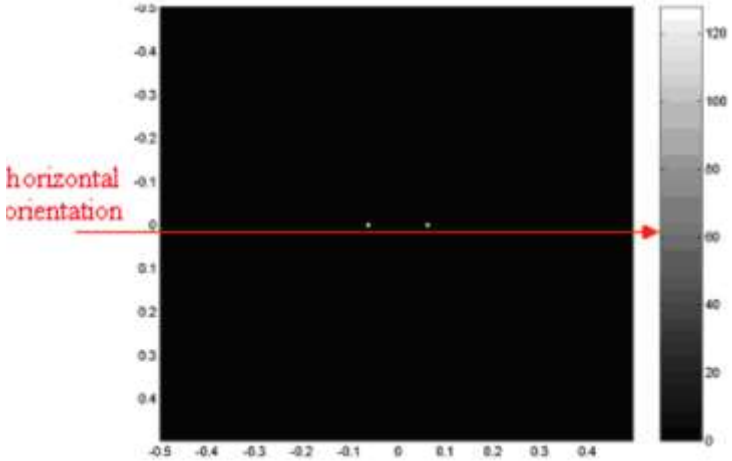
spatial distance (s)



Frequency (# of cycles per unit distance)

$$I(x, y) = 128\ sin(2\pi x/16)$$



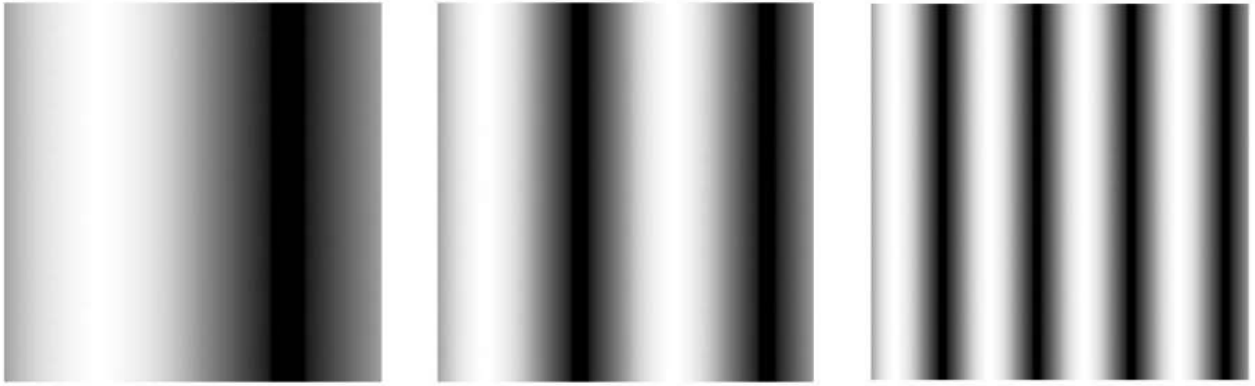


Sinusoid pattern repeats every 16 pixels
f = 1/16 cycles/pixel

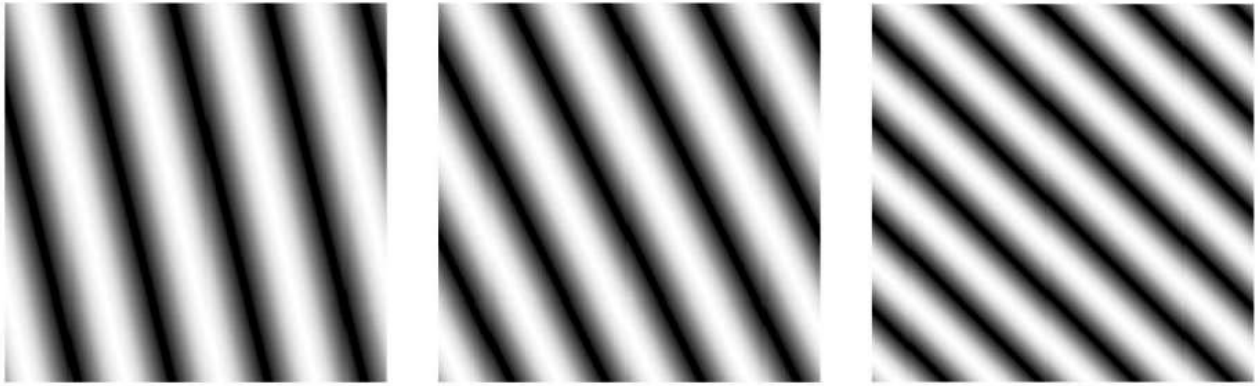
Spatial domain

Frequency domain

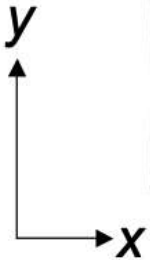
- Intensity images for $s(x,y) = \sin\left[2\pi\left(u_0 x + v_0 y\right)\right]$



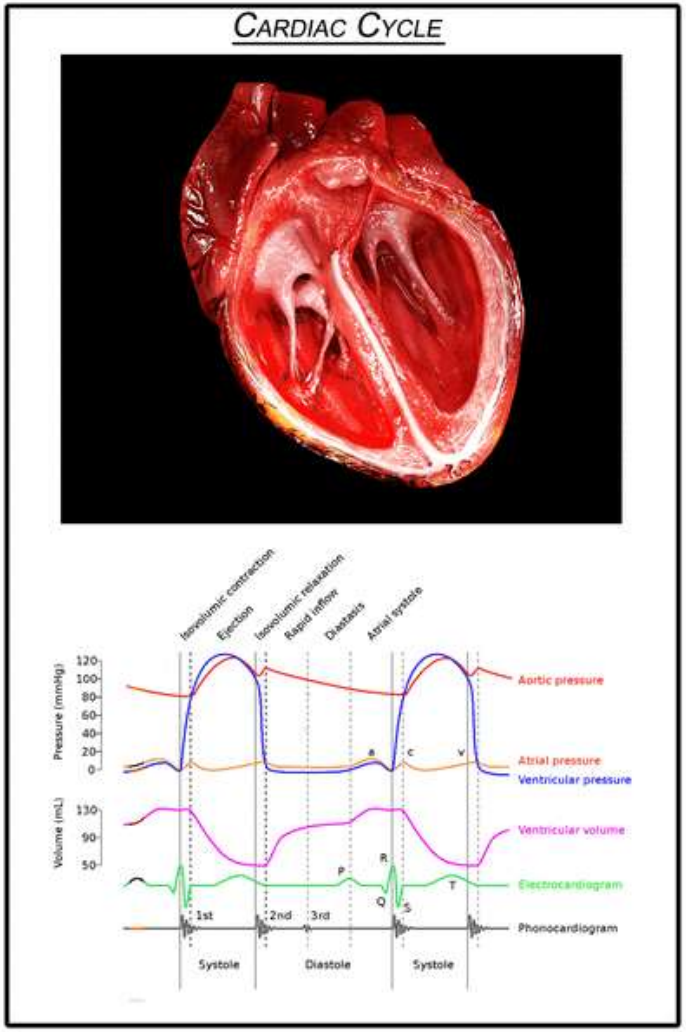
$u_0 = 1, v_0 = 0$ $u_0 = 2, v_0 = 0$ $u_0 = 4, v_0 = 0$



$u_0 = 4, v_0 = 1$ $u_0 = 4, v_0 = 2$ $u_0 = 4, v_0 = 4$



Many natural
phenomena (signals)
are periodic
but not necessarily
sinusoidal



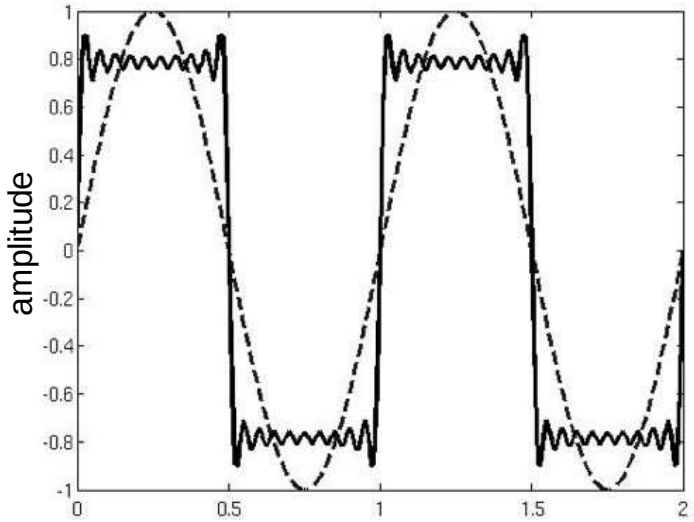
https://commons.wikimedia.org/wiki/File:Cardiac-Cycle-Animated.gif

Fourier Series

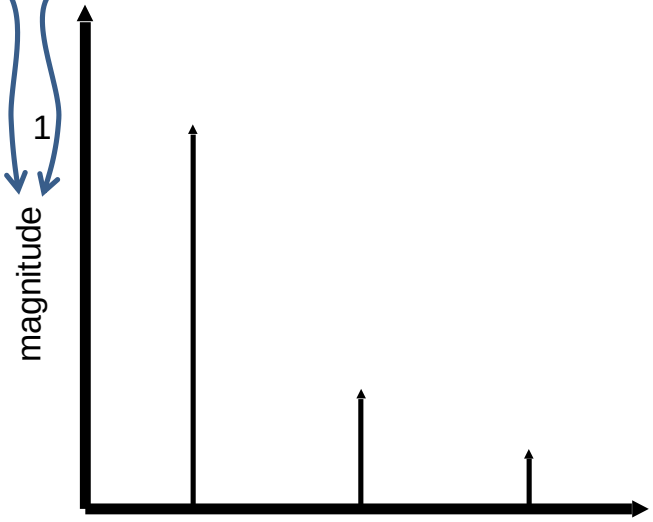
Approximate **periodic functions** with sines and cosines

Fourier Series

$$y(t) = \sum_n a_n \sin(nf \times 2\pi t) + \sum_n b_n \cos(nf \times 2\pi t)$$



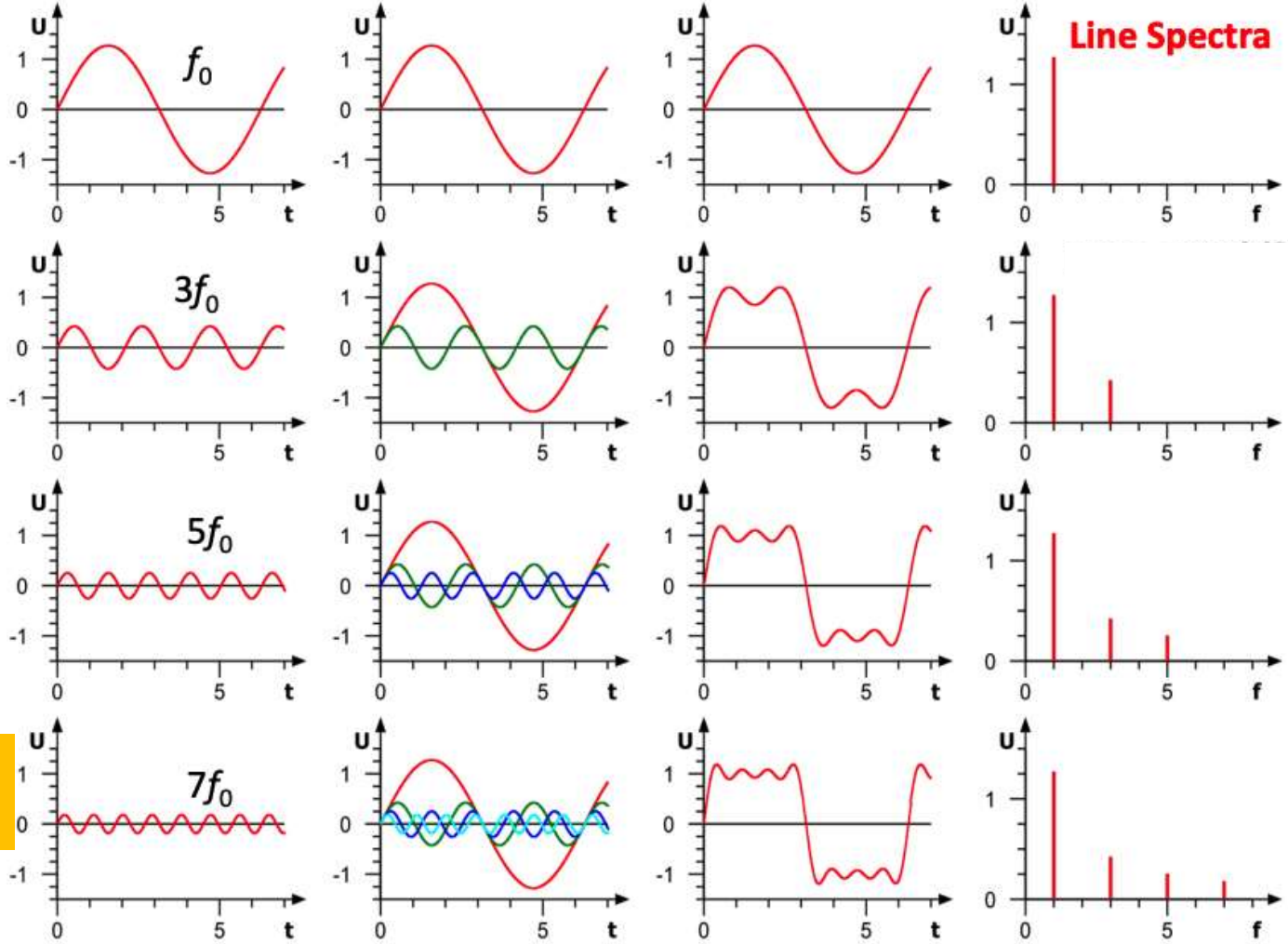
time/length



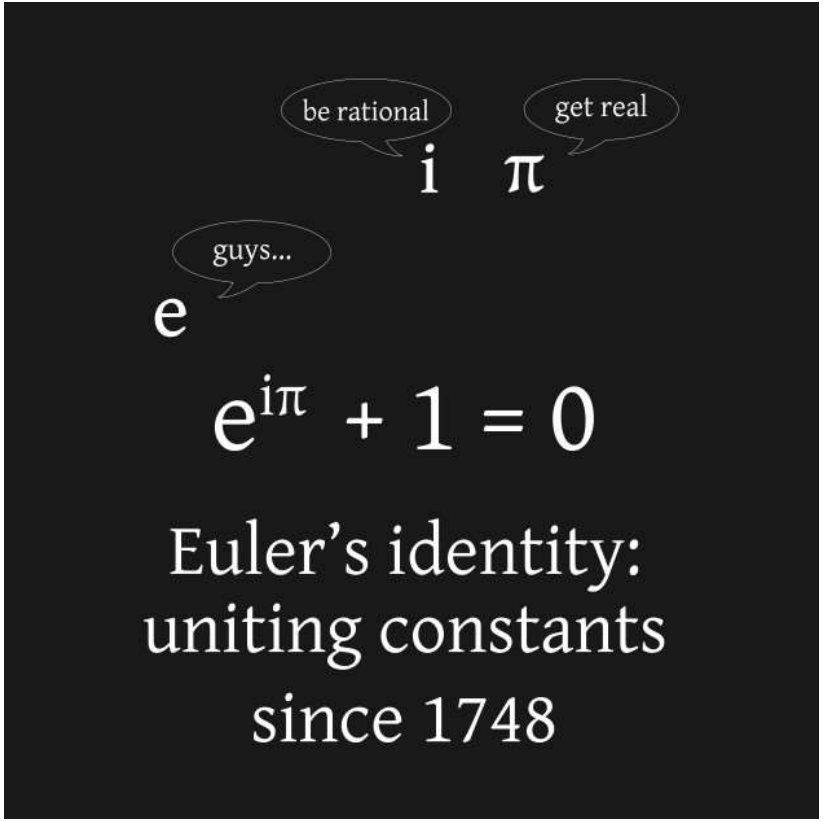
frequency

Function ⟷ **Frequency Spectrum**

Example: Periodic Square Wave as Sum of Sinusoids

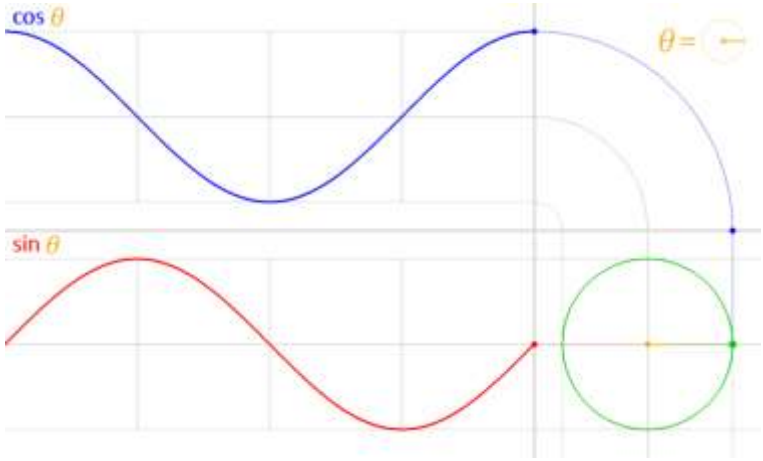


Fourier Basis



$$e^{it} = \cos t + i \sin t$$

$$i = \sqrt{-1}$$



https://res.cloudinary.com/teepublic/image/private/s--s4GX0hG3--/t_Preview/b_rgb:191919,c_limit,f_jpg,h_630,q_90,w_630/v1486157785/production/designs/1177382_1.jpg

Fourier Series in terms of complex coefficients

$$f(t) = a_0 + \sum_{m=1}^{\infty} a_m \cos\left(\frac{2\pi mt}{T}\right) + \sum_{n=1}^{\infty} b_n \sin\left(\frac{2\pi nt}{T}\right)$$

$$= \sum_{m=0}^{\infty} a_m \cos\left(\frac{2\pi mt}{T}\right) + \sum_{n=1}^{\infty} b_n \sin\left(\frac{2\pi nt}{T}\right)$$

$$a_0 = \frac{1}{T}\int_0^T f(t)dt$$

$$a_m = \frac{2}{T}\int_0^T f(t)\cos\left(\frac{2\pi mt}{T}\right)dt$$

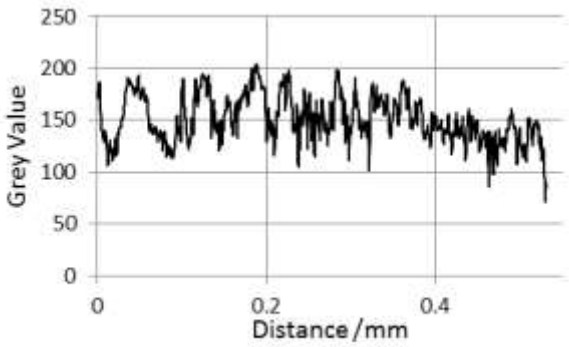
$$b_n = \frac{2}{T}\int_0^T f(t)\sin\left(\frac{2\pi mt}{T}\right)dt$$

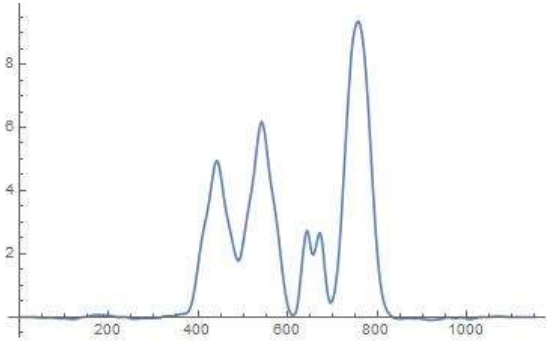
$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{i\frac{2\pi nt}{T}}$$

$$c_n = \frac{1}{T}\int_0^T f(t)e^{-i\frac{2\pi nt}{T}}dt$$

http://www.thefouriertransform.com/series/complexcoefficients.php

What if f(t) is non-periodic ?





$$\boxed{f(t) = \sum_{n=-\infty}^{\infty} c_n e^{i\frac{2\pi n t}{T}}}$$

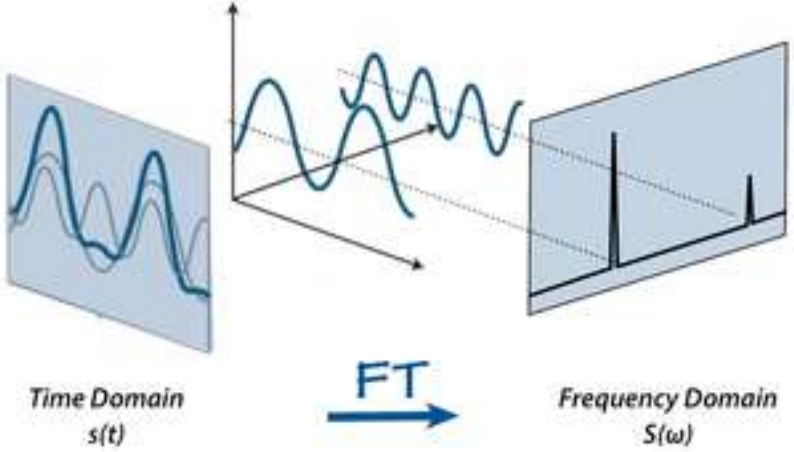
https://i.stack.imgur.com/fKznF.jpg

Fourier Transform

Approximate **non-periodic functions** with complex sinusoids

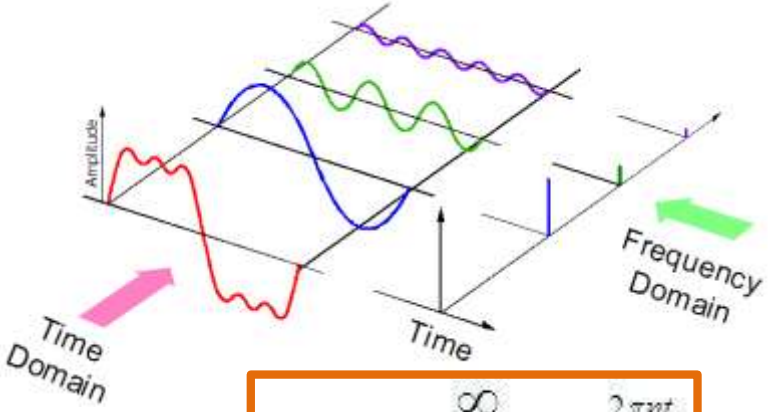
Fourier Transform vs Series

Fourier Transform



$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$$

Fourier Series (periodic only)



$$g(t) = \sum_{n=-\infty}^{\infty} c_n e^{i\frac{2\pi n t}{T}}$$

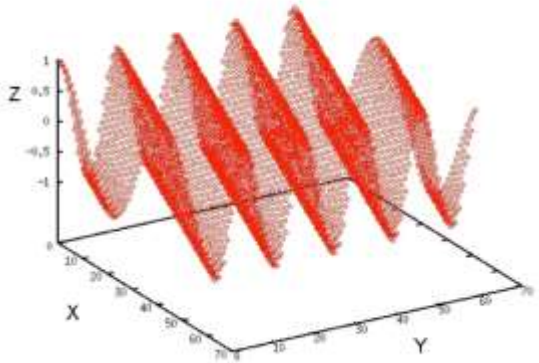
$$c_n = \frac{1}{T}\int_0^T f(t) e^{-i\frac{2\pi n t}{T}} dt$$

2D DFT and IDFT

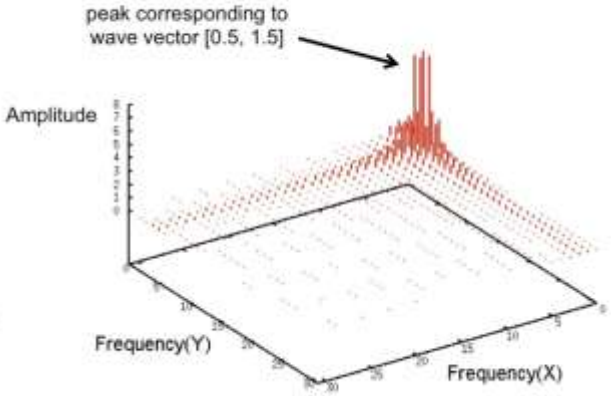
Transform

$$F[m,n] = \sum_{y=0}^{y=(N-1)} \sum_{x=0}^{x=(M-1)} f[x,y]\, e^{-2\pi j\left(\frac{mx}{M}+\frac{ny}{N}\right)}$$

$\cos(0.5\pi x + 1.5\pi y)$

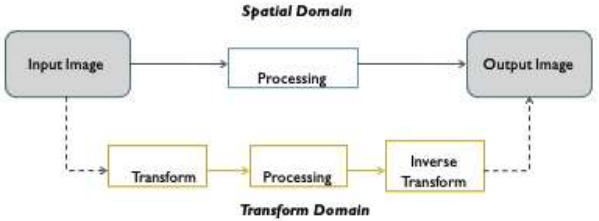


$\mathrm{FFT}[\cos(0.5\pi x + 1.5\pi y)]$

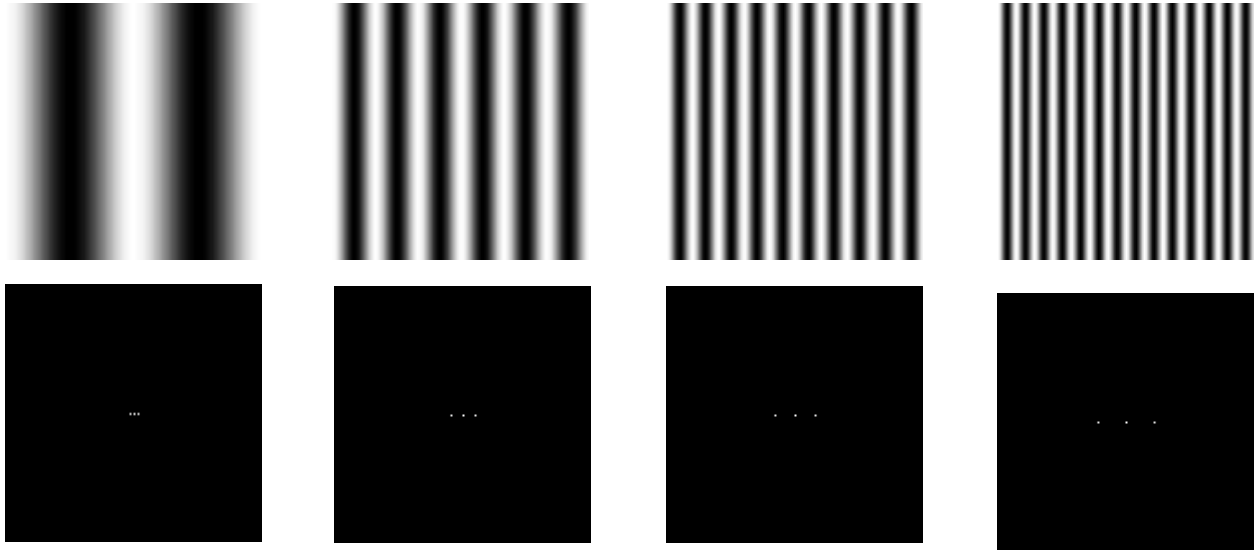


Inverse Transform

$$f[x,y] = \frac{1}{MN} \sum_{n=0}^{n=(N-1)} \sum_{m=0}^{m=(M-1)} F[m,n]\, e^{2\pi j\left(\frac{mx}{M}+\frac{ny}{N}\right)}$$

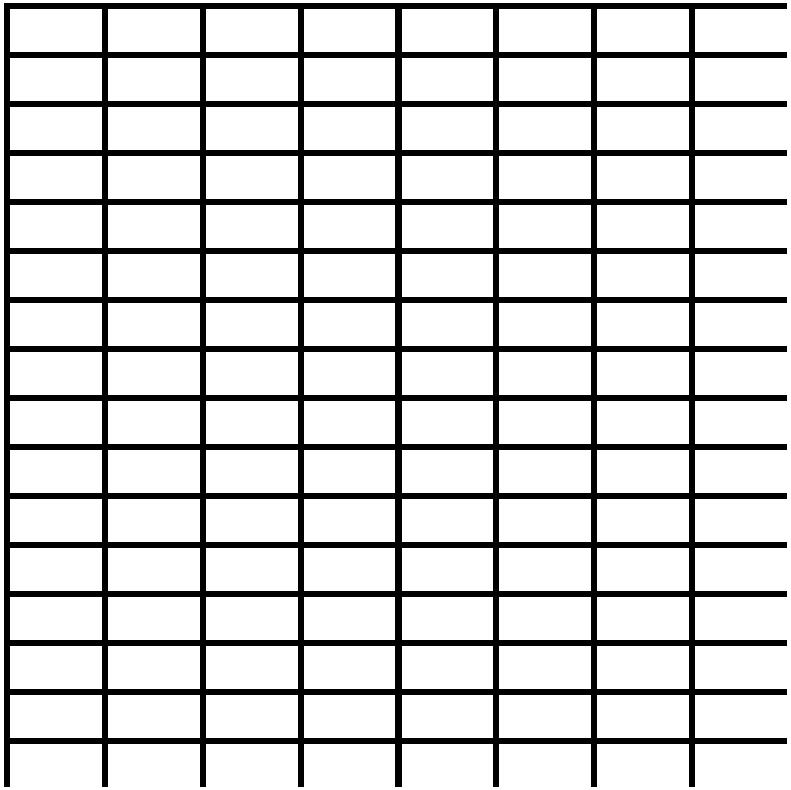


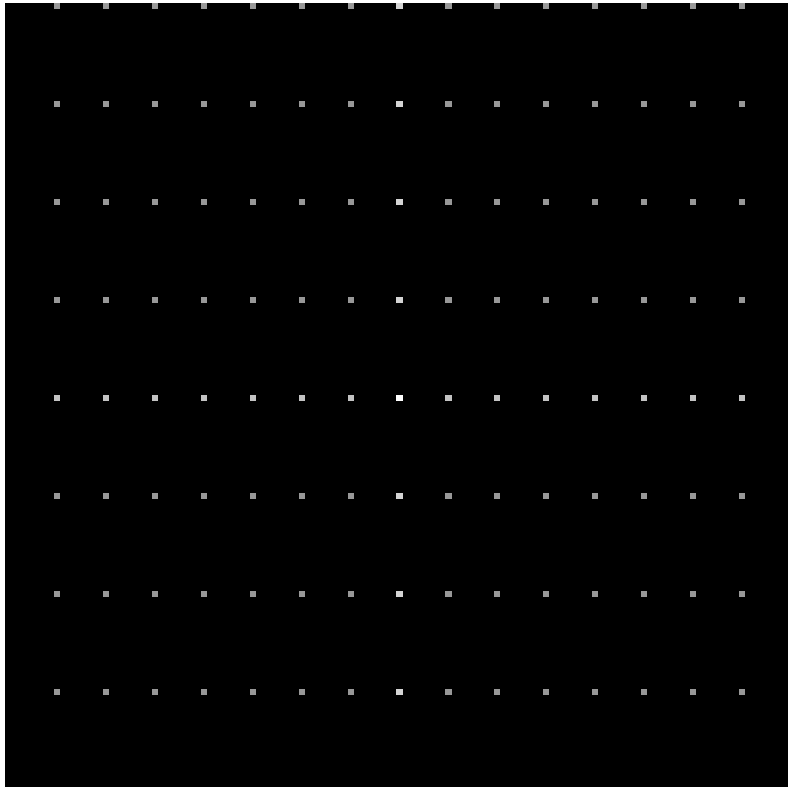
DFT Example



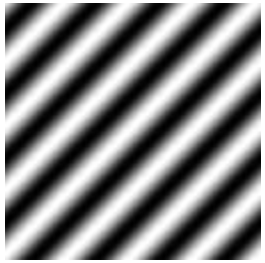
Courtesy: http://cns-alumni.bu.edu/~slehar/fourier/fourier.html

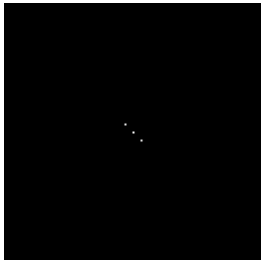
Example





DFT Example (Rotation)

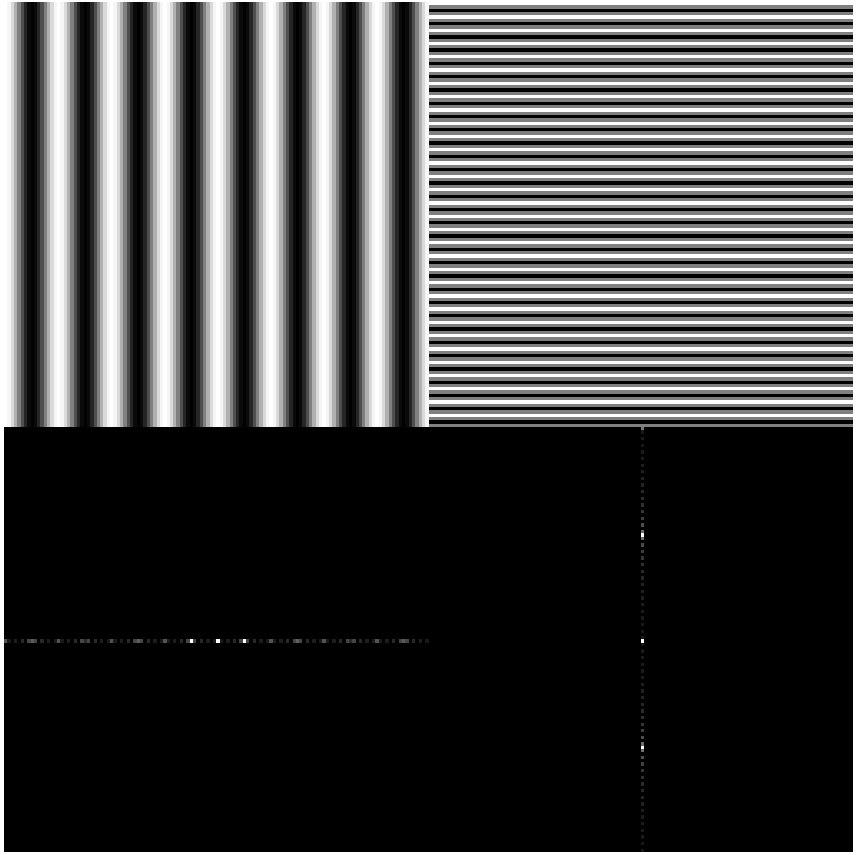




Courtesy: http://cns-alumni.bu.edu/~slehar/fourier/fourier.html

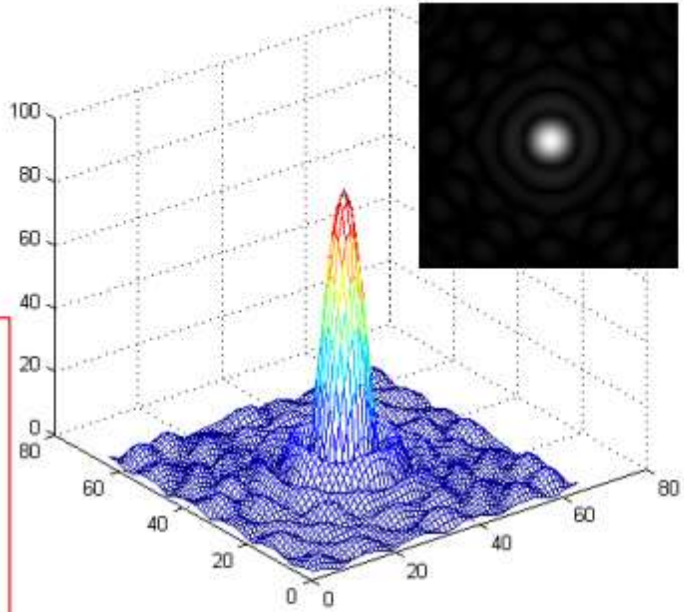
https://www.cs.unm.edu/~brayer/vision/fourier.html

DFT for simple ‘spatial’ patterns

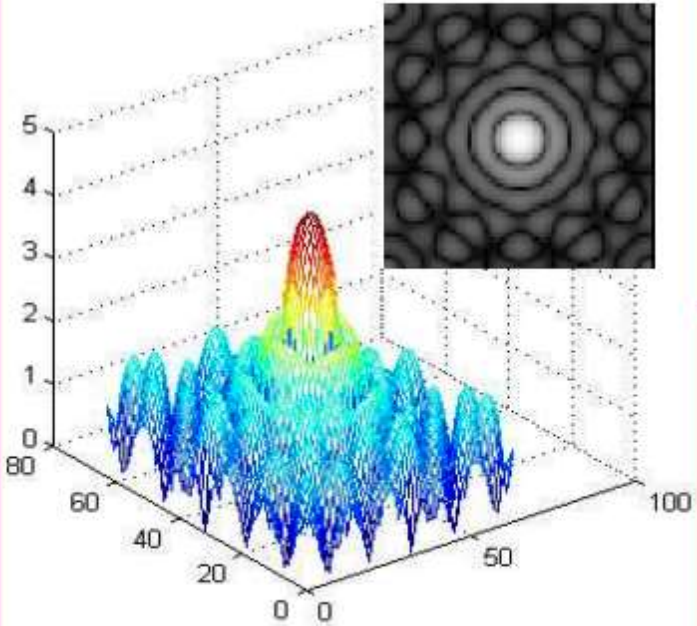


$f(x,y)$

(64x64)

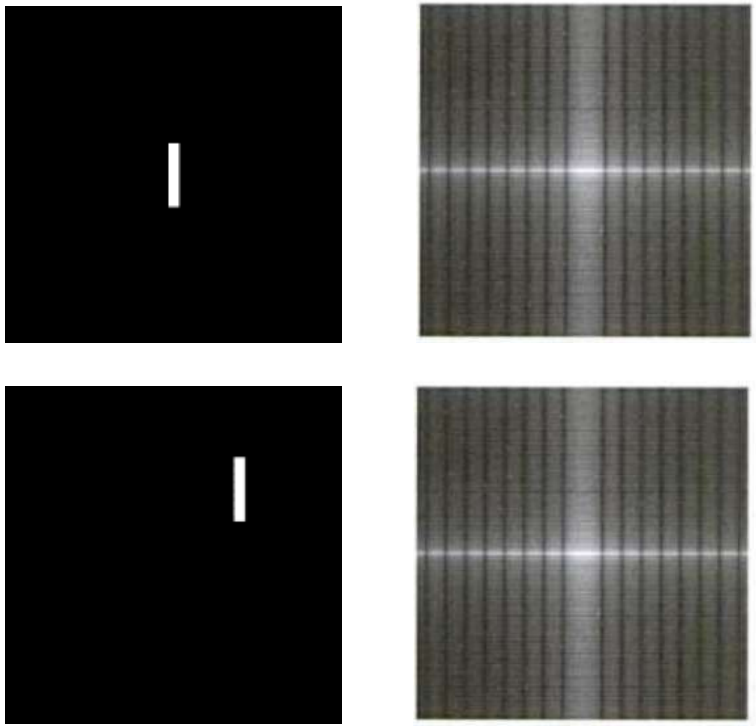


$|F(u,v)|$

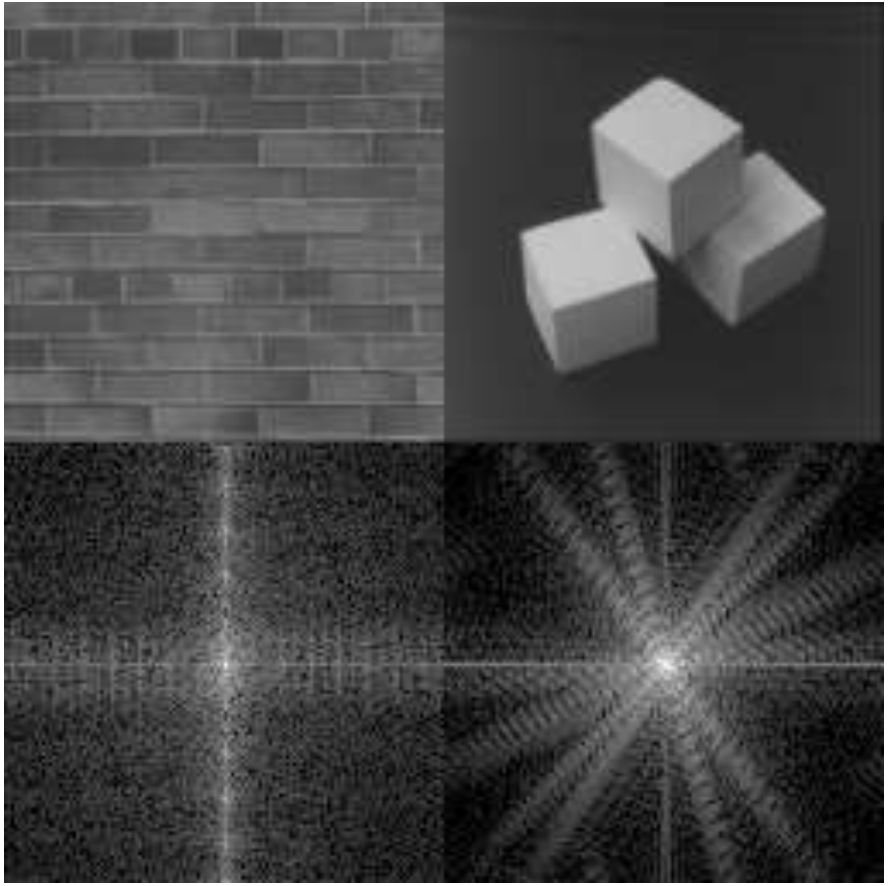


$log(1+|F(u,v)|)$

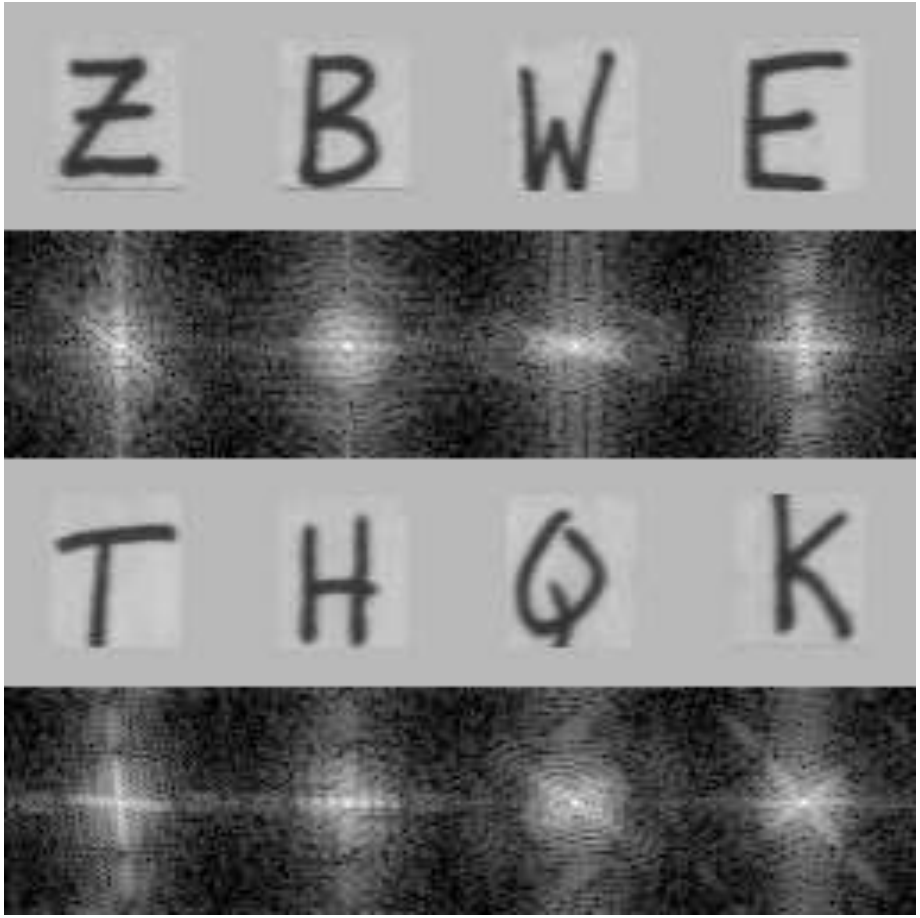
DFT Example (Translation - Magnitude)



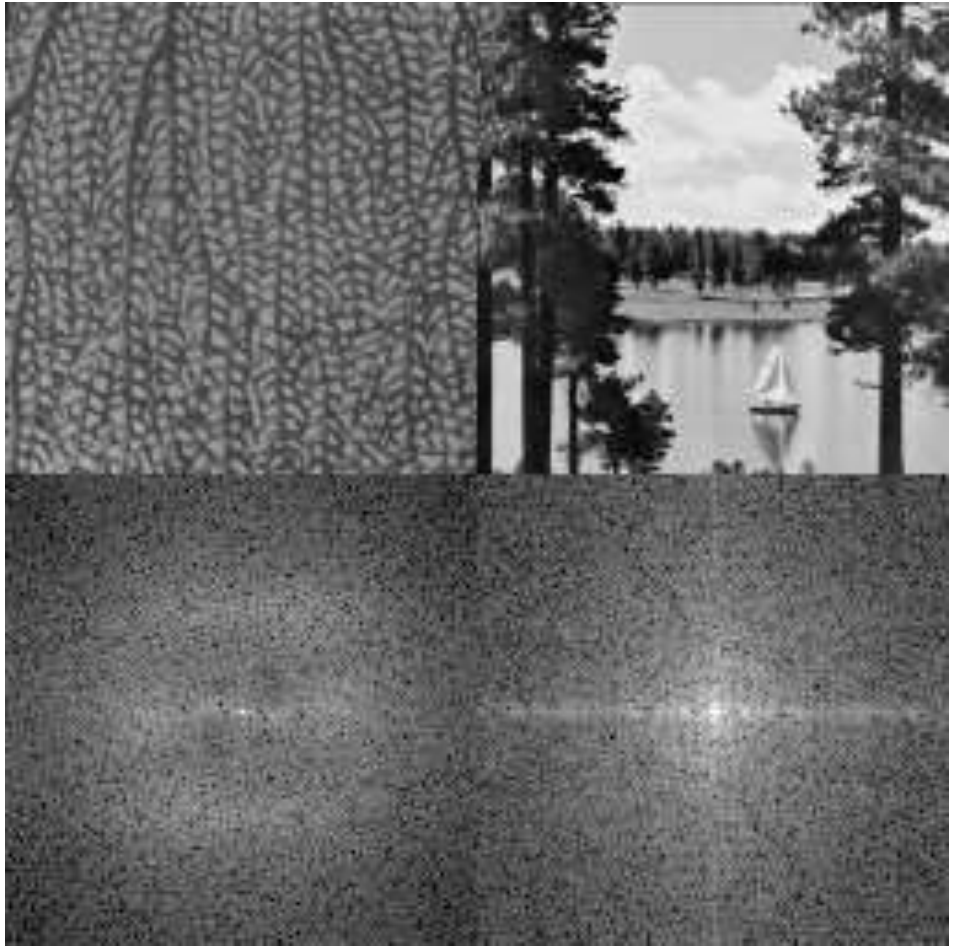
Some examples of images and spectra



Some examples of images and spectra



Some examples of images and spectra



- DFT has an efficient version called FFT (Fast Fourier Transform)

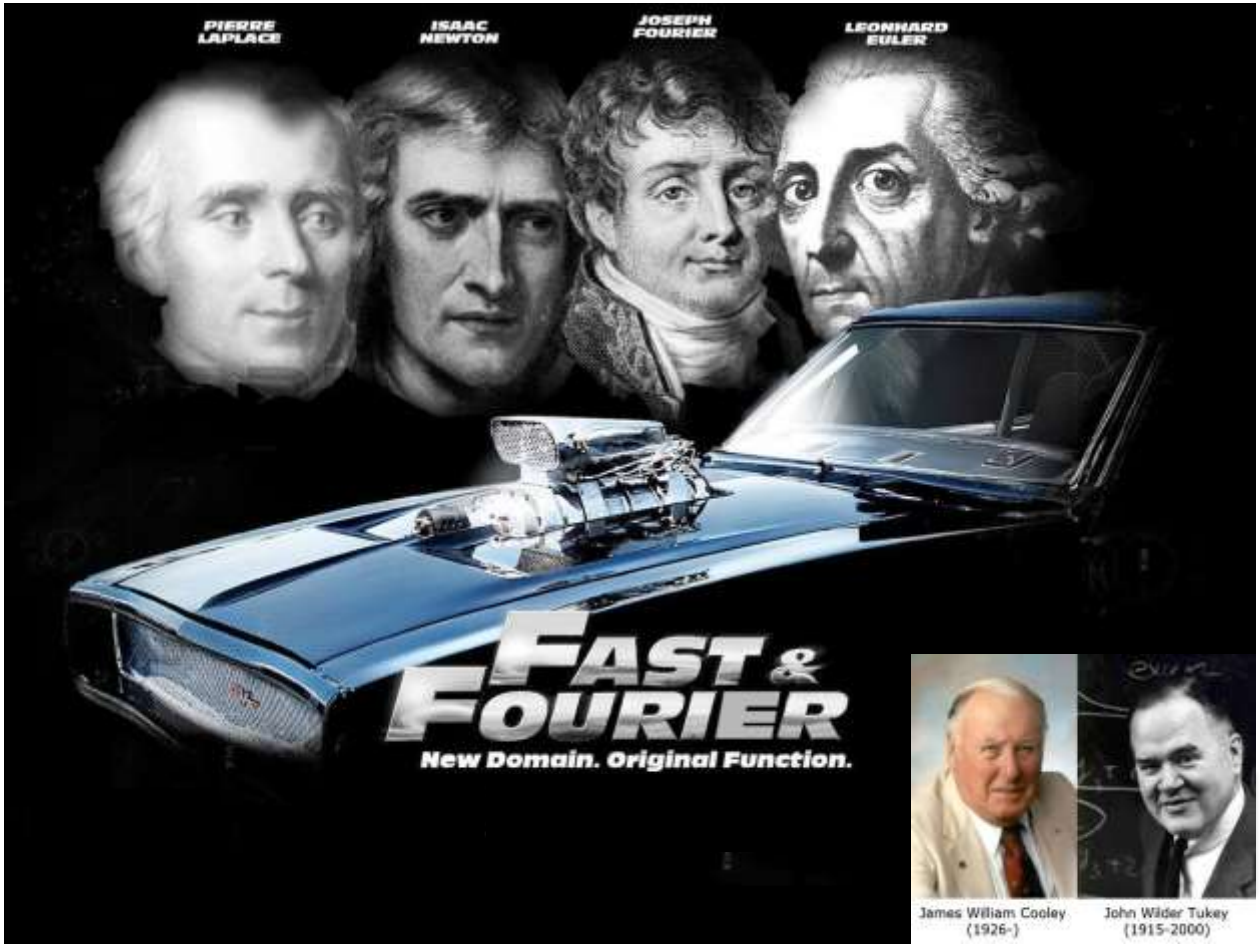
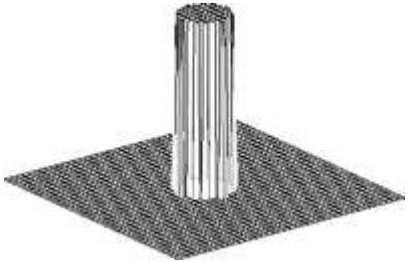
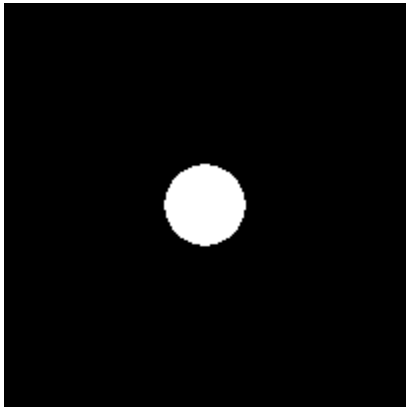


Image Enhancement and Filtering in Frequency Domain

Ideal Low Pass Filters

$$H(u,v) = \begin{cases} 1 \text{ if } D(u,v) \le D_0 \\ 0 \text{ if } D(u,v) > D_0 \end{cases}$$





where $D(u,v) = [(u - M/2)^2 + (v - N/2)^2]^{1/2}$

D_0 → cut off frequency

Image courtesy: Gonzalez and Woods

Ideal Low Pass Filters

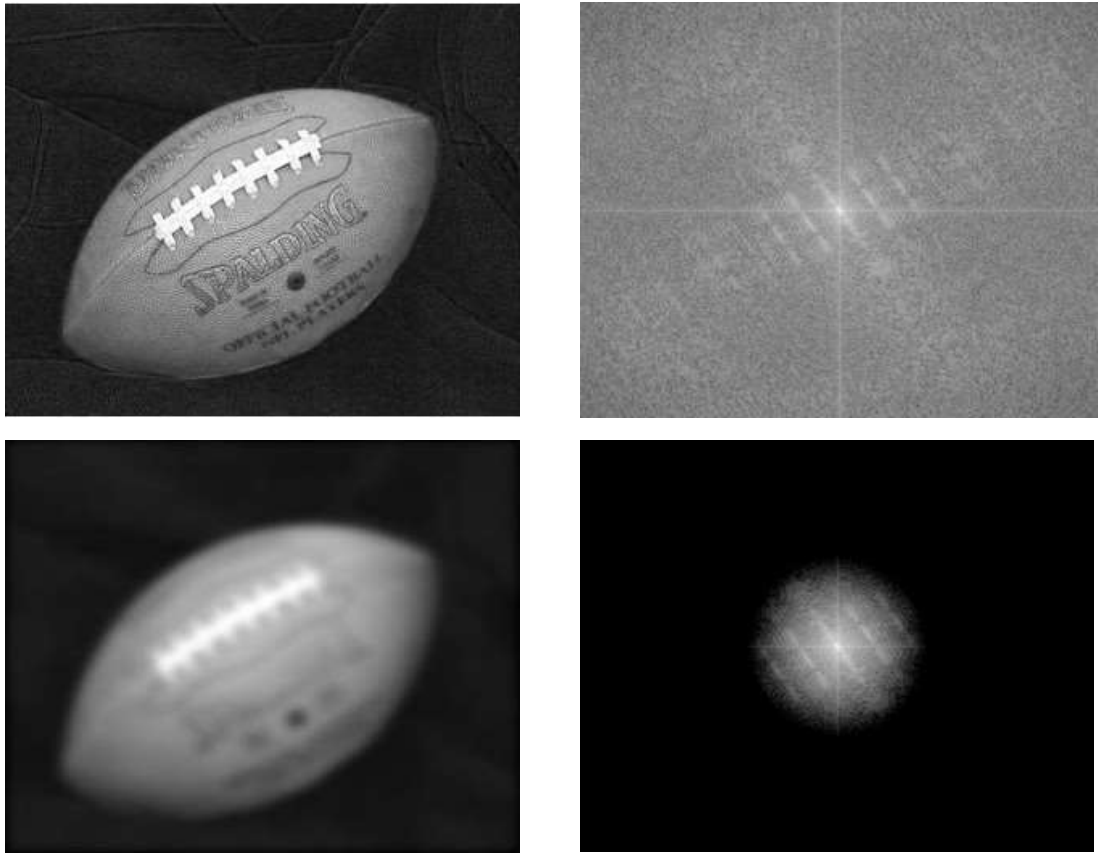
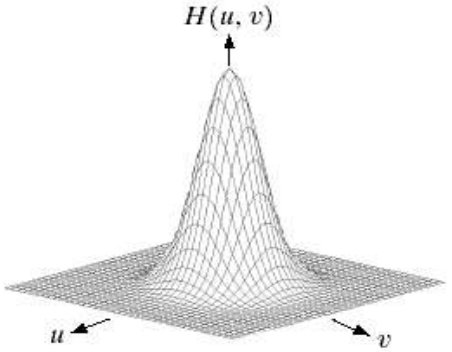
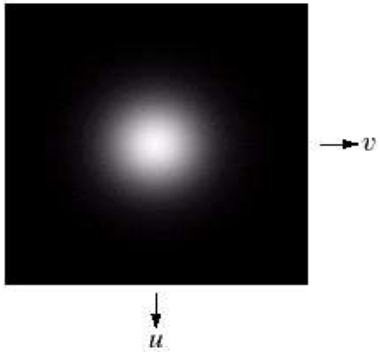
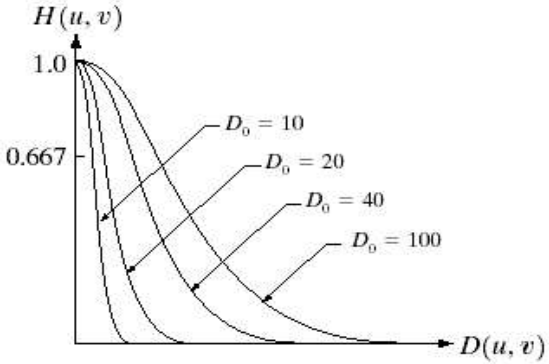


Image courtesy: cs.uregina.ca

Gaussian Low Pass Filters



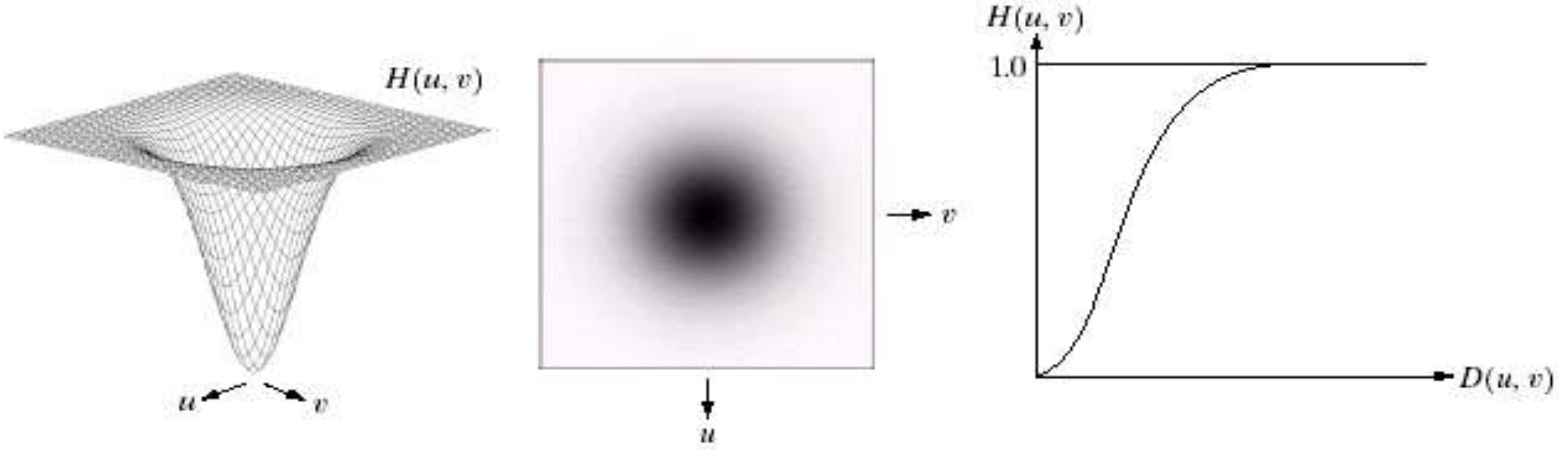




$$H(u,v) = e^{-D^2(u,v)/2D_0{}^2}$$

Image courtesy: Gonzalez and Woods

Gaussian High Pass Filters

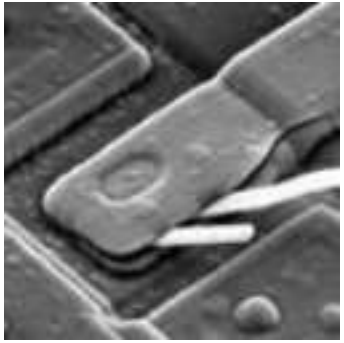


$$H(u,v) = 1 - e^{-D^2(u,v)/2D_0^2}$$

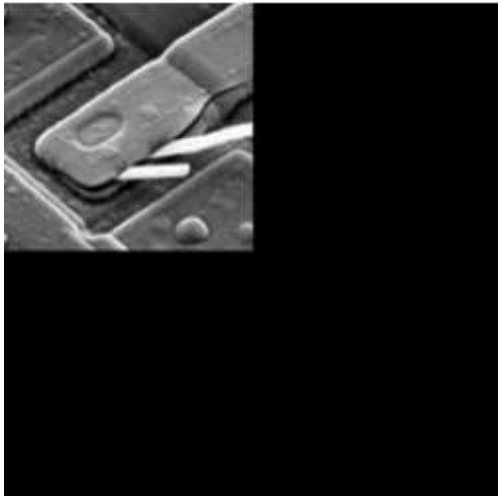
Image courtesy: Gonzalez and Woods

- Low-pass = remove high frequency = remove rapid changes in intensity = smoothing
- High-pass = remove low frequency = retain rapid changes in intensity = gradient-like = retain edges

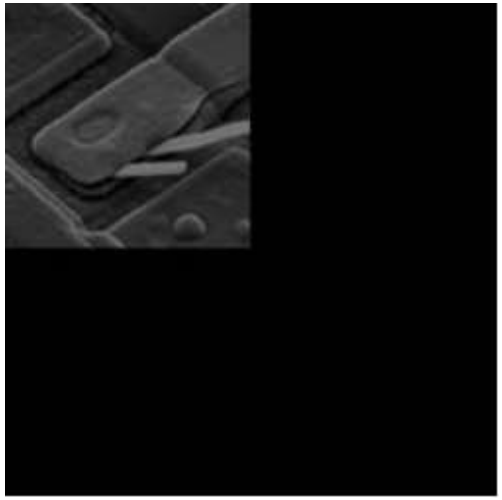
Recipe for transform domain processing



Given: M x N image f

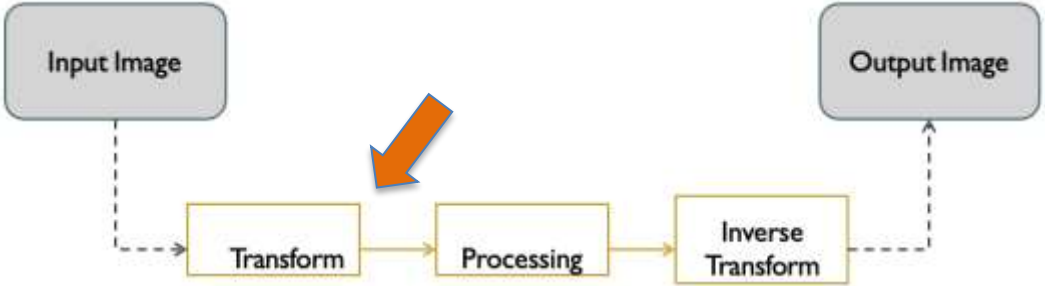


1: pad f_p to size P x Q
where P = 2M, Q = 2N

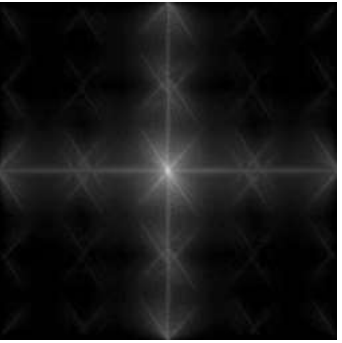


2: Multiply f_p by $(-1)^{(x+y)}$

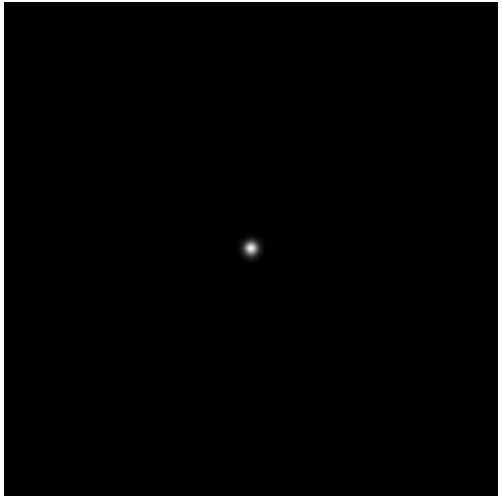
3: Compute $F_p = DFT(f_p)$



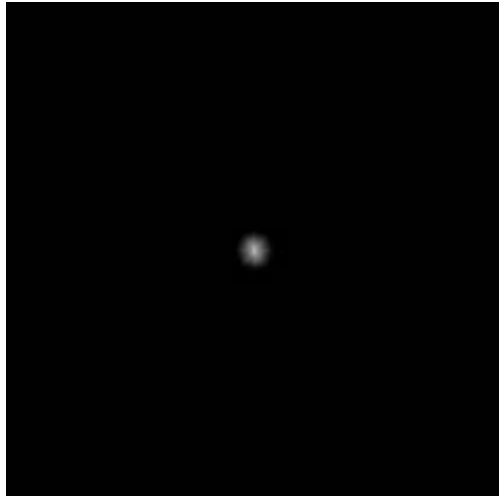
Recipe for transform domain processing



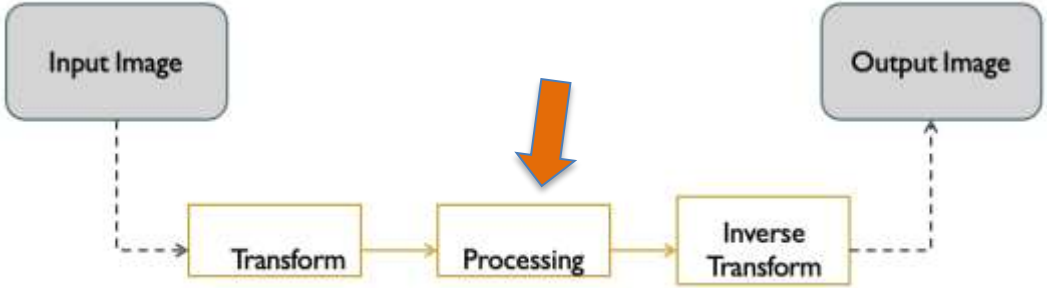
F_p - Fourier Spectrum of f_p



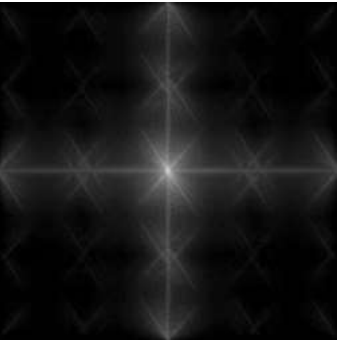
4: Centered Gaussian low pass spectral filter H



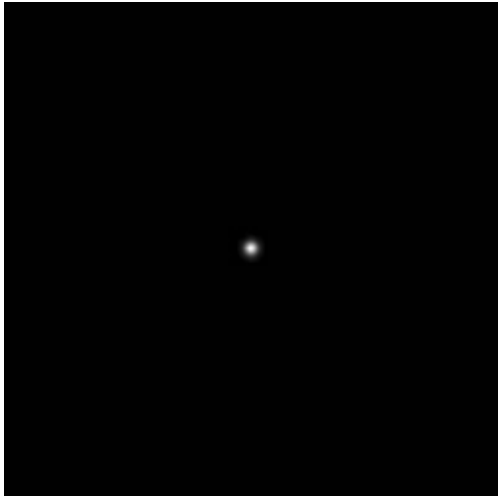
5: Compute $G_p = \mathrm{H}F_p$



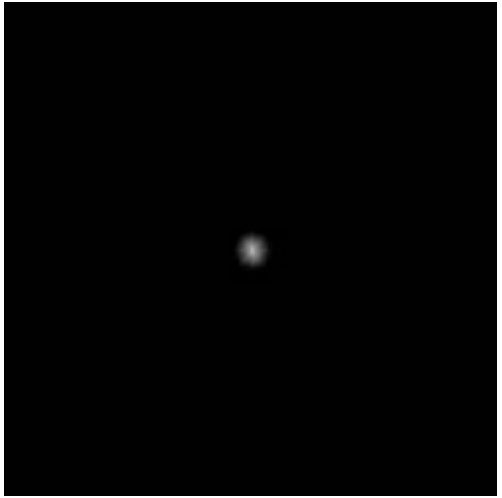
Recipe for transform domain processing



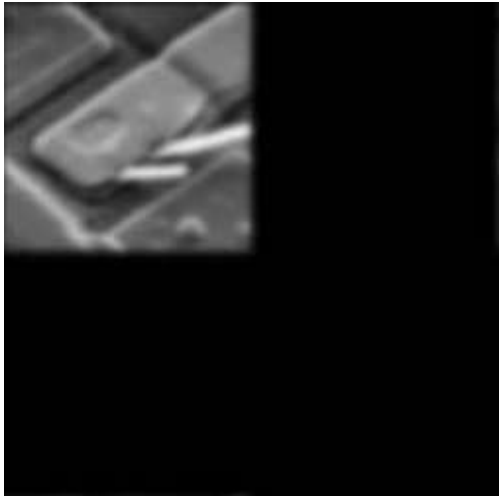
F_p - Fourier Spectrum of f_p



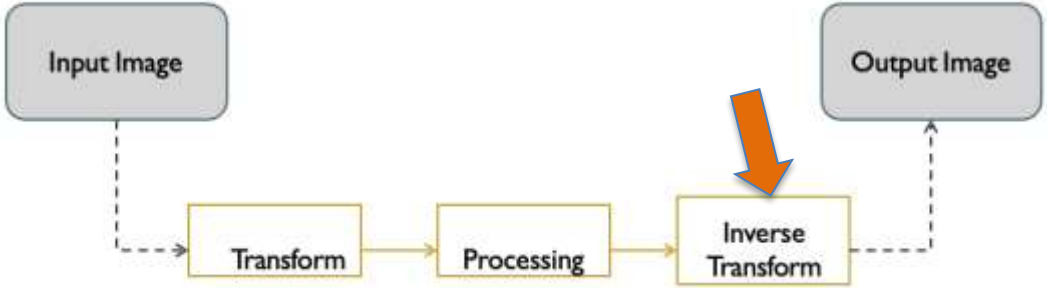
4: Centered Gaussian low pass spectral filter H

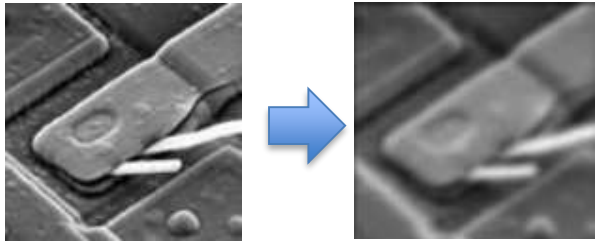


5: Compute $G_p = \mathrm{H}F_p$



6: Compute $\mathrm{Re}[IDFT\{G_p\}](-1)^{(x+y)}$

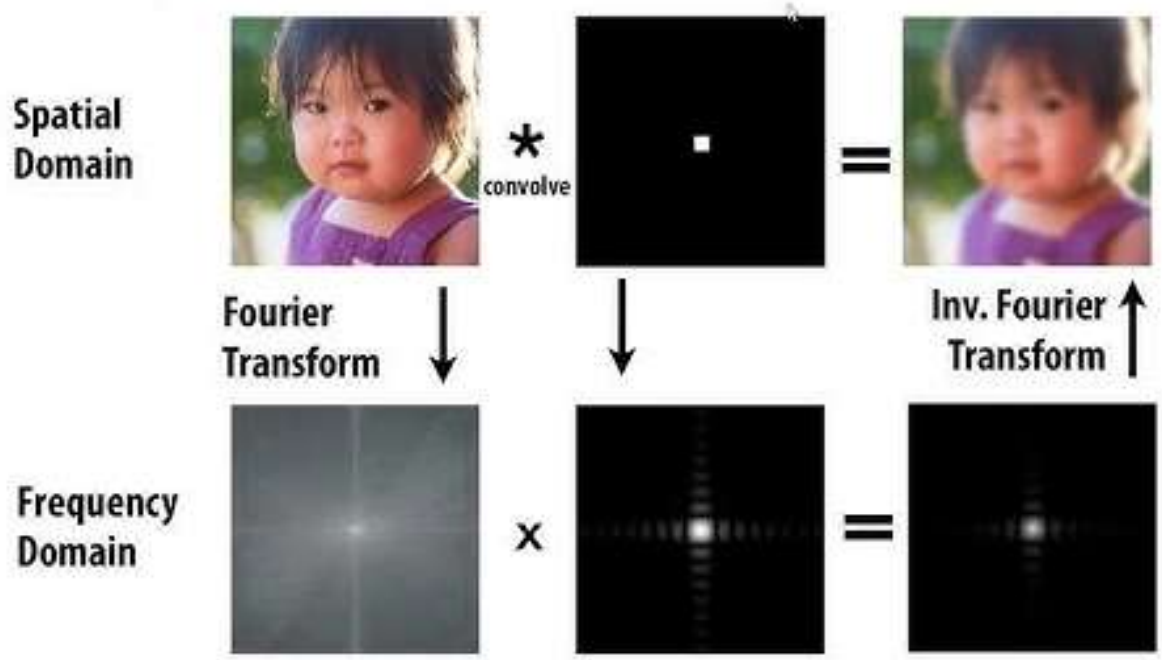




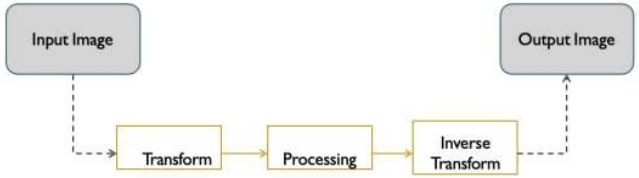
7: Filtered result

Convolution theorem

Convolution in the spatial domain is equal to multiplication in the frequency domain, and vice versa

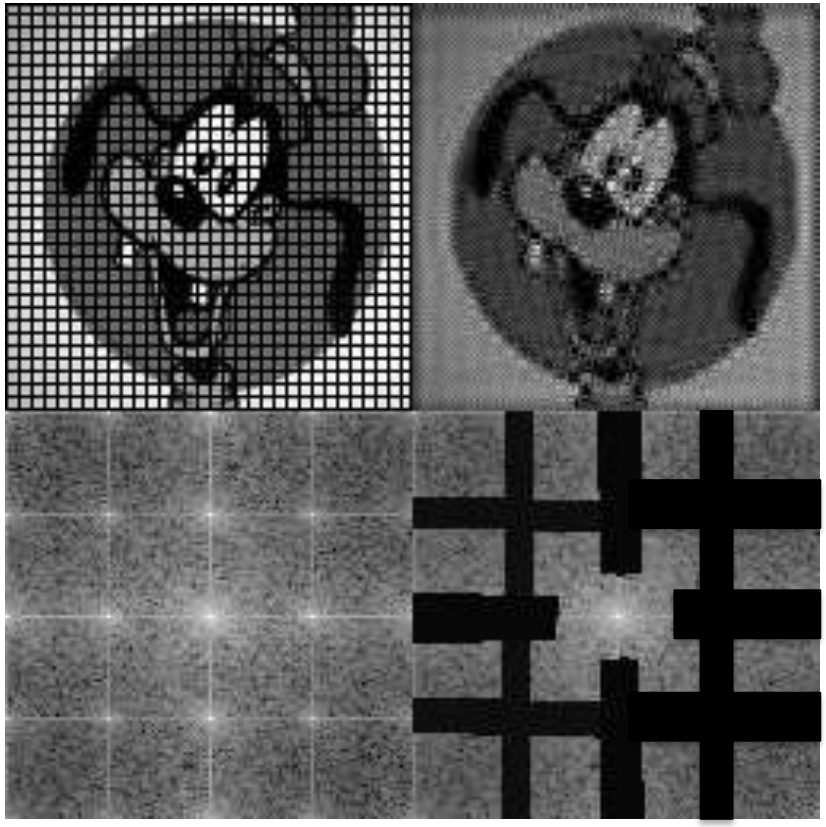


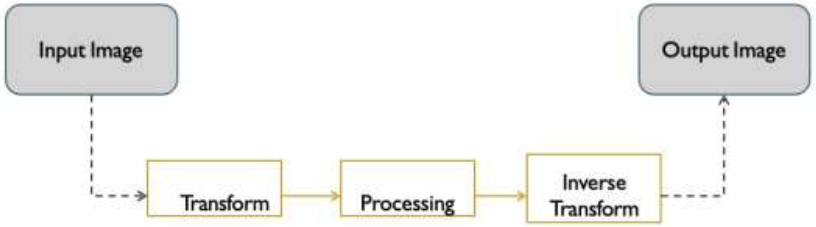
Stanford CS248, Winter 2020



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

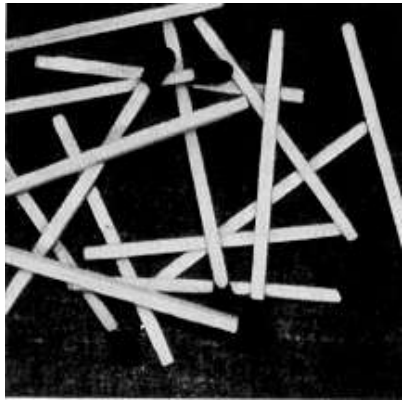
Artifact removal

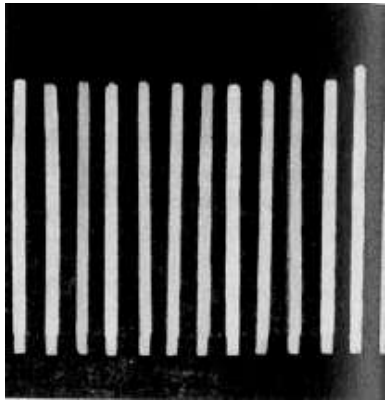




Spatial redundancy

- Interpixel redundancy exists ➔ pixel values are correlated
- i.e., a pixel value can be reasonably predicted by its neighbors





histograms

auto-correlation

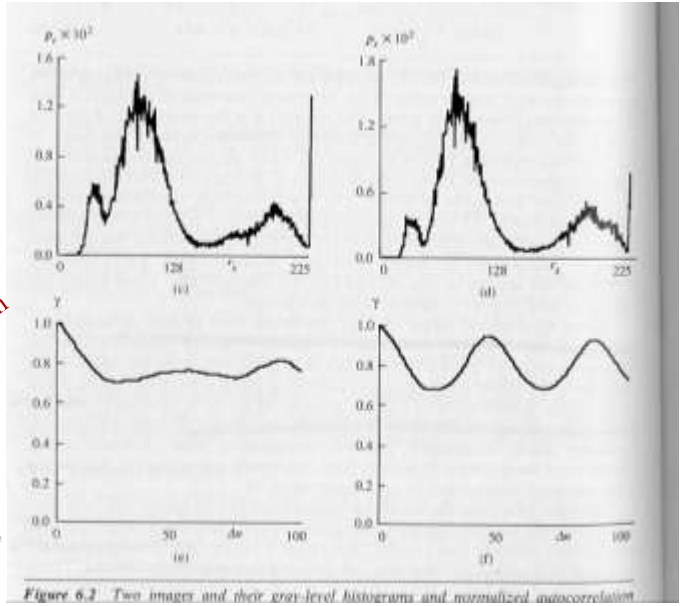


Figure 6.2 Two images and their gray-level histograms and normalized autocorrelation

$$f(x)\, o\, g(x) = \int_{-\infty}^{\infty} f(x) g(x+a)\, da$$

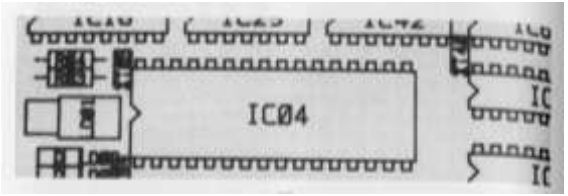
auto-correlation: f(x)=g(x)

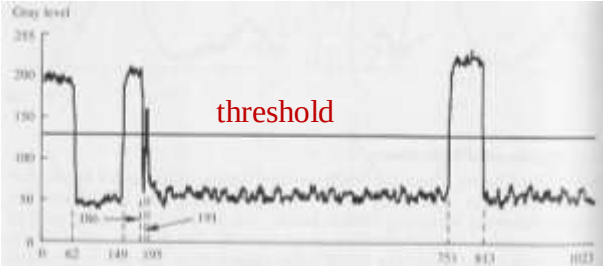
Interpixel redundancy (cont'd)

- To reduce interpixel redundancy, some kind of transformation must be applied on the data (e.g., thresholding, DFT, DWT)

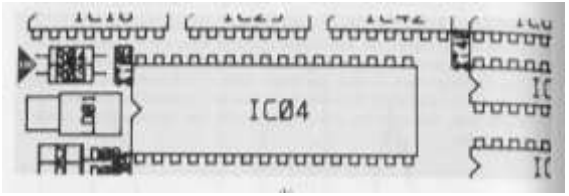
Example:

original





thresholded



11 ……0000……..11…..000…..

Run-length encoding:

(1,63) (0,87) (1,37) (0,5) (1,4) (0, 556) (1,62) (0,210)

Using 11 bits/pair: (1+10) bits/pair

88 bits are required (compared to 1024 !!)

Spatial and temporal redundancy

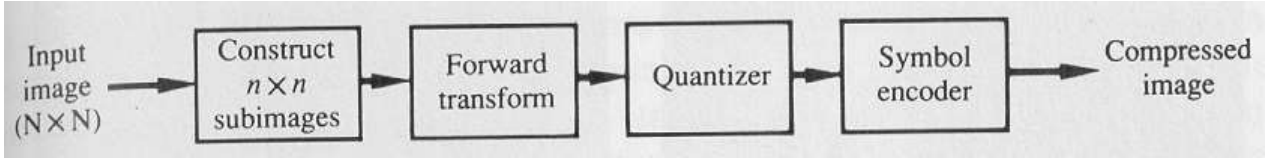
frame t

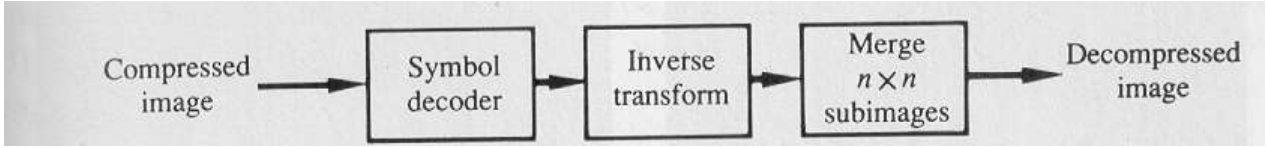
frame t+1

Spatial and temporal redundancy

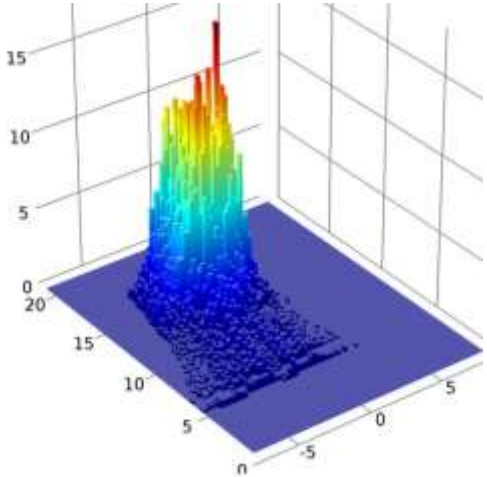
Lossy Compression

- Transform the image into some other domain to reduce interpixel redundancy.



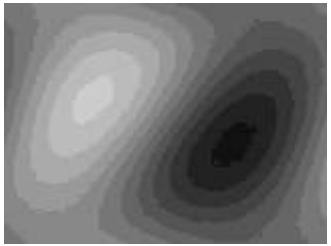


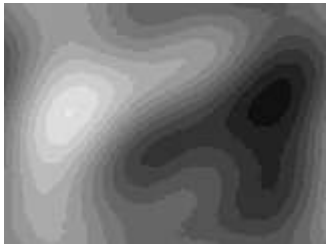
$$f[x,y] = \frac{1}{MN} \sum_{n=0}^{n=(N-1)} \sum_{m=0}^{m=(M-1)} F[m,n]\, e^{2\pi j\left(\frac{mx}{M} + \frac{ny}{N}\right)}$$



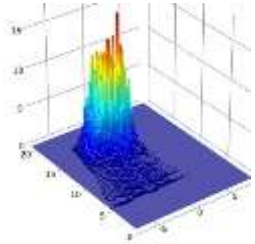
$$F[m,n] = \sum_{y=0}^{y=(N-1)} \sum_{x=0}^{x=(M-1)} f[x,y]\, e^{-2\pi j\left(\frac{mx}{M} + \frac{ny}{N}\right)}$$

$$f[x,y] = \frac{1}{MN} \sum_{n=0}^{n=(N-1)} \sum_{m=0}^{m=(M-1)} F[m,n]\, e^{2\pi j\left(\frac{mx}{M} + \frac{ny}{N}\right)}$$





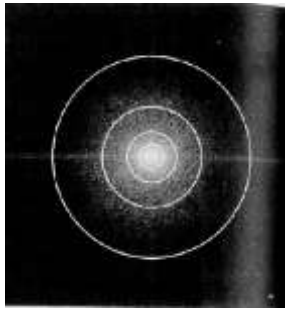
From 50% of the lowest frequencies



Adding up to 50% lowest frequencies

Example: Fourier Transform

$$f(x,y) = \frac{1}{N}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{\frac{j2\pi(ux+vy)}{N}}, \quad \text{x,y=0,1,...,N-1}$$



Note that the magnitude of the FT decreases, as *u*, *v* increase!

K << N

$$\hat{f}(x,y) = \frac{1}{N}\sum_{u=0}^{K-1}\sum_{v=0}^{K-1} F(u,v)e^{\frac{j2\pi(ux+vy)}{N}}, \quad \text{x,y=0,1,...,N-1}$$

$\sum_{x,y}(\hat{f}(x,y) - f(x,y))^2$ is very small !!

Transform Selection

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u,v)h(x,y,u,v)$$

- T(u,v) can be computed using various transformations, for example:
  - DFT
  - DCT (Discrete Cosine Transform)
  - KLT (Karhunen-Loeve Transformation) or Principal Component Analysis (PCA)

- JPEG using DCT for handling interpixel redundancy.

DCT (Discrete Cosine Transform)

[proposed by Nasir Ahmed, T. Natarajan, K.R.Rao (1972)]

All Real

Forward: $$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)cos(\frac{(2x+1)u\pi}{2N})cos(\frac{(2y+1)v\pi}{2N}),$$

u, v=0,1,...,N-1

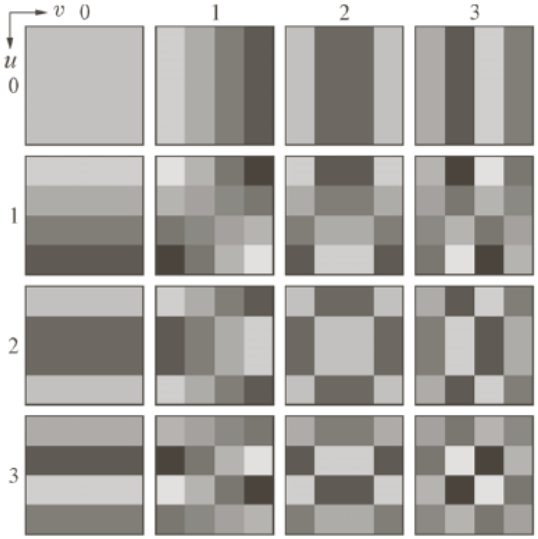
Inverse: $$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)cos(\frac{(2x+1)u\pi}{2N})cos(\frac{(2y+1)v\pi}{2N}),$$

x, y=0,1,...,N-1

$$\alpha(u) = \begin{cases} \sqrt{1/N} & \text{if u=0} \\ \sqrt{2/N} & \text{if u>0} \end{cases} \quad \alpha(v) = \begin{cases} \sqrt{1/N} & \text{if v=0} \\ \sqrt{2/N} & \text{if v>0} \end{cases}$$

DCT (cont'd)

- Basis functions for a 4x4 image (i.e., cosines of different frequencies).

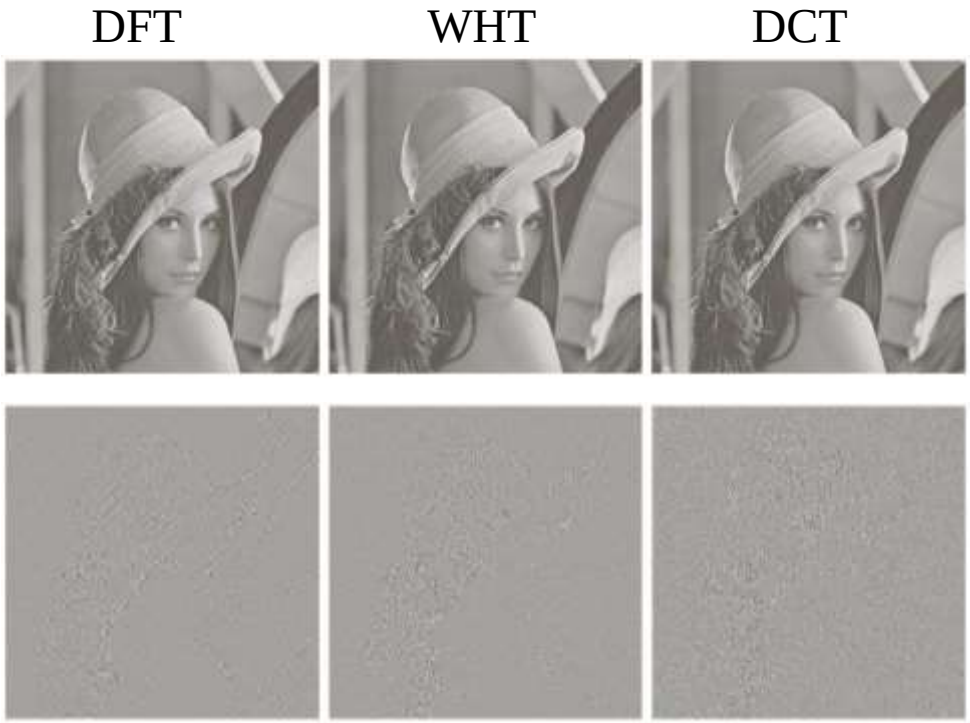


DCT (cont'd)

Using
8 x 8 sub-images
yields 64 coefficients
per sub-image.

Reconstructed images
by truncating
50% of the
coefficients

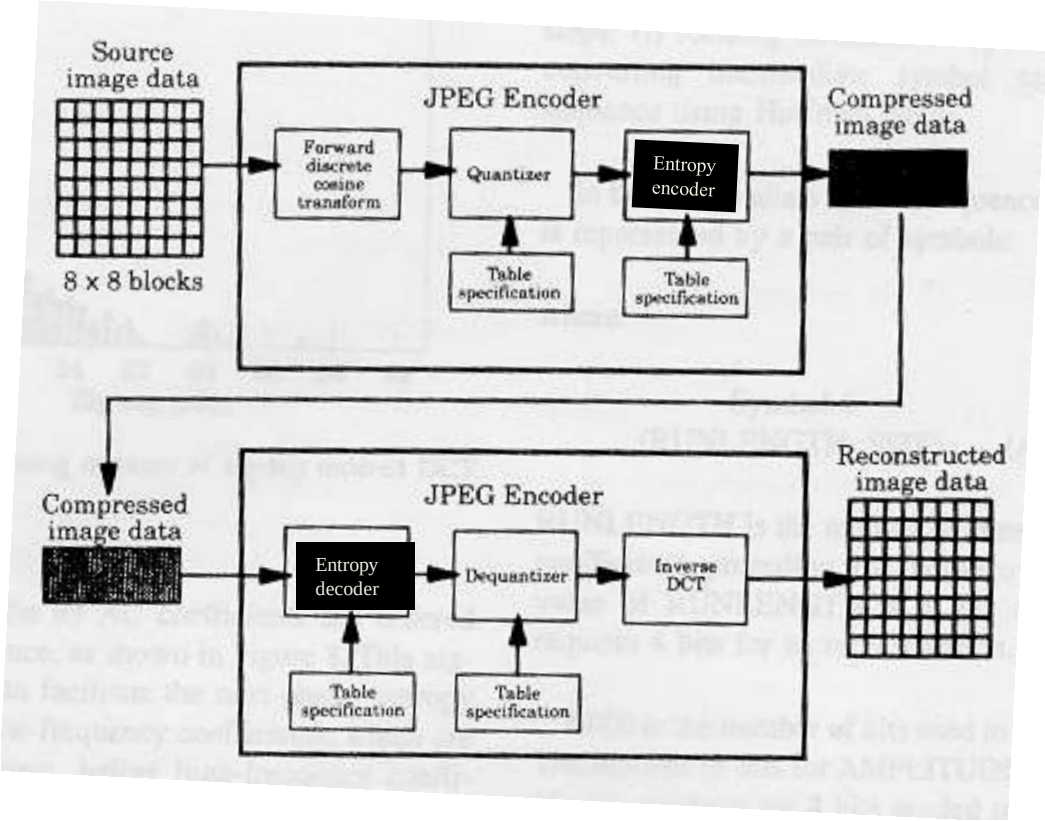
More compact
transformation



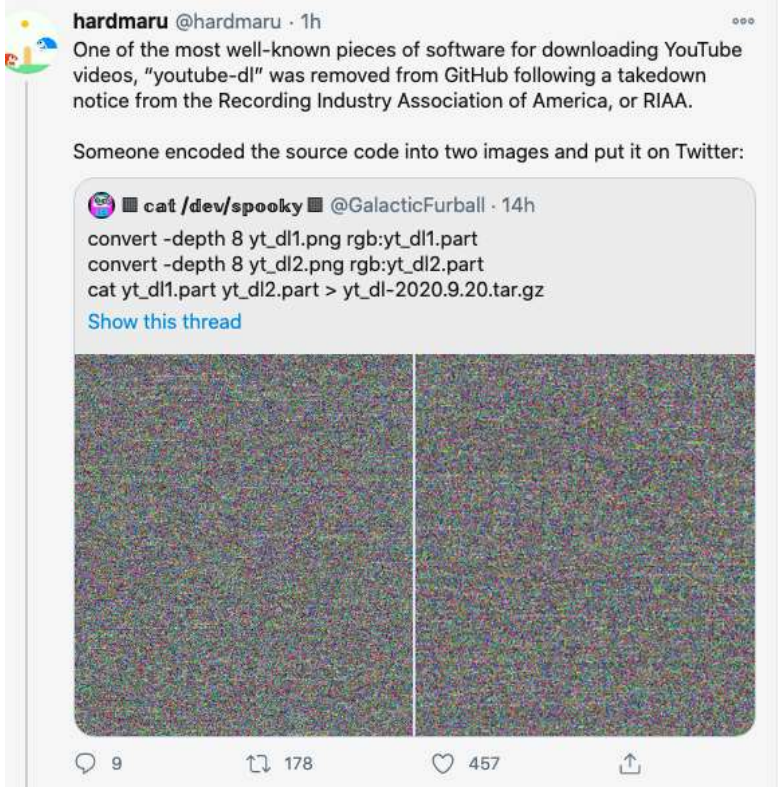
RMS error: 2.32 1.78 1.13

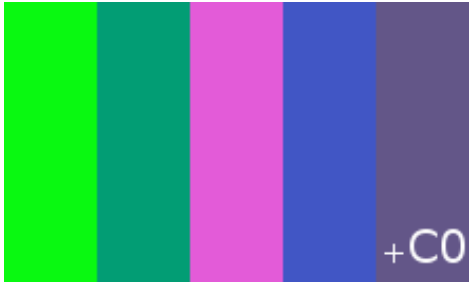
JPEG Compression

Accepted as an international image compression standard in 1992.



Encoding data as images





AACS LA DVD Decryption Code

Binary Images

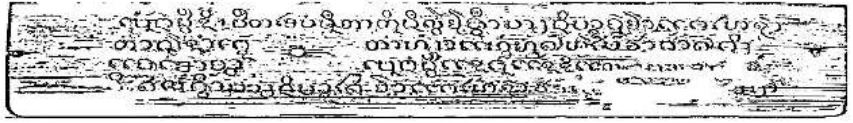
Plant Phenotyping

Document Image Analysis

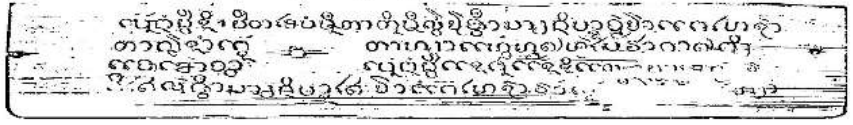
a) RGB image

b) Noise reduction image

c) Binary image by Otsu's algorithm



d) Binary image by Niblack's algorithm



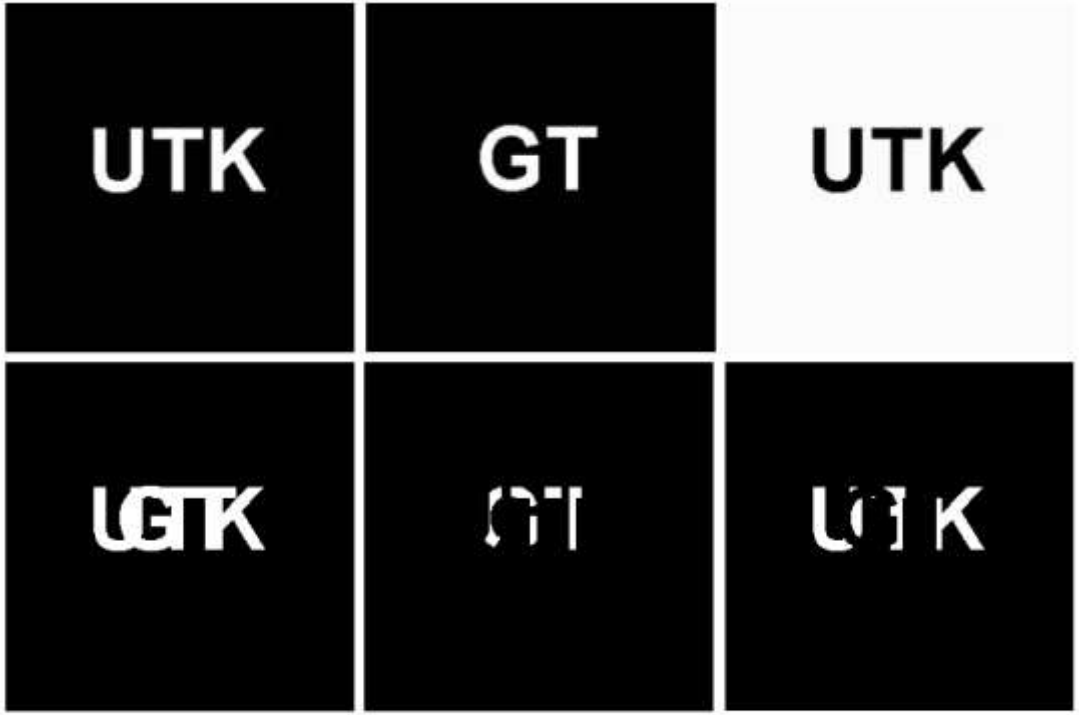
e) Binary image by Sauvola's algorithm

Figure 2. Samples of palm leaf images

Background Subtraction

Set Operations on Binary Images

A B



From: Digital Image Processing, Gonzalez,Woods And Eddins

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

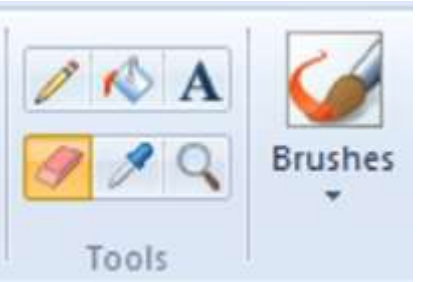
Disc

	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 : 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	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

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	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	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

Erosion

- Simple application of **pattern matching**

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

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	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	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	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

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 (max filter)

1	1	1
1	1	1
1	1	1

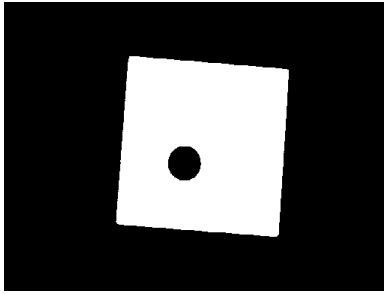
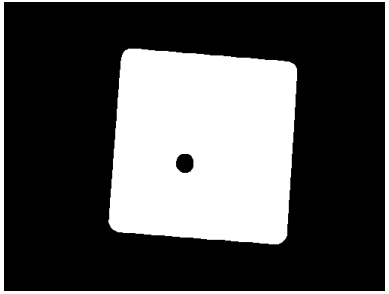
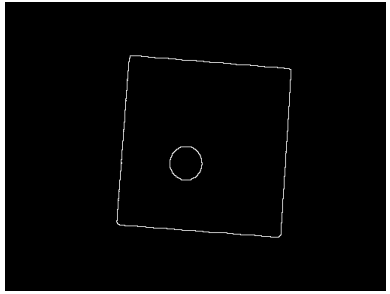
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0
0	0	1	1	1	1	1	0	0	0	0	1	1	1	0	0
0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0
0	0	0	1	1	0	0	0	0	1	1	1	1	1	0	0
0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0
0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0
0	0	0	0	0	0	1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

Boundary Detection

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

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

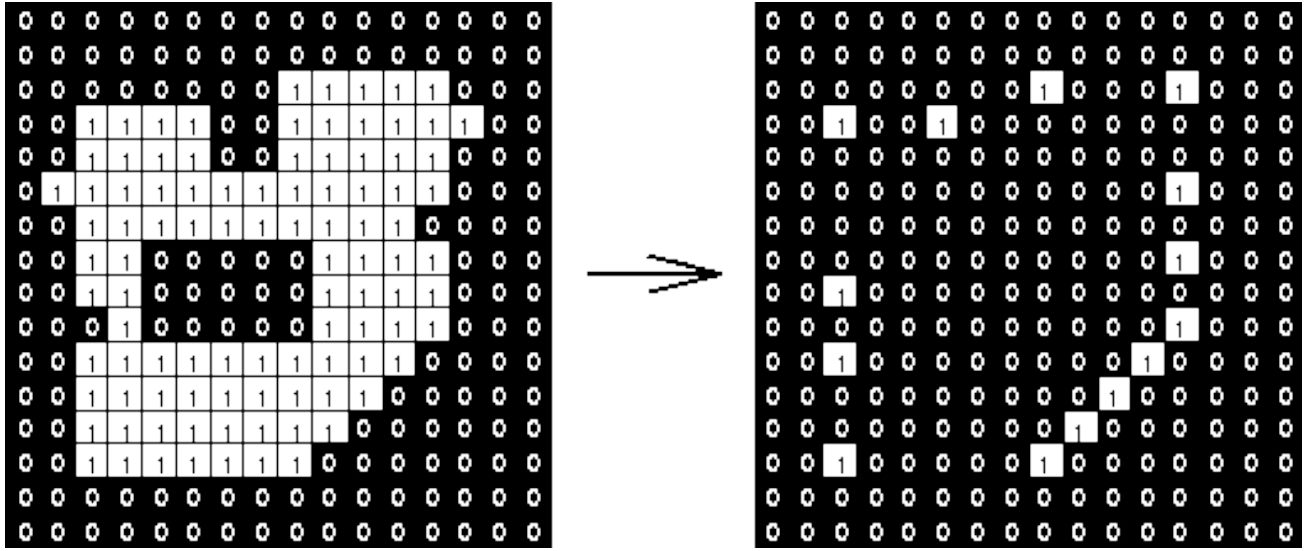
Example: Find right-angled convex corners

	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

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

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	*

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

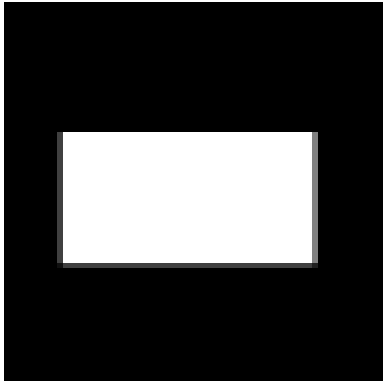
Distance Transform
(with chessboard distance metric)

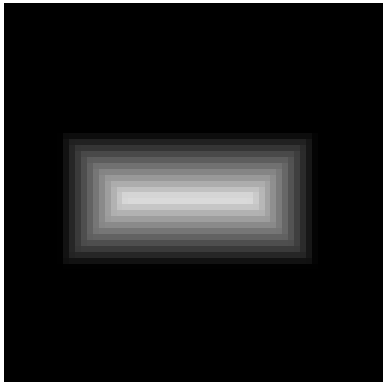
0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	1	1	1	1	1	0
0	0	0	0	0	0	0

Binary Image

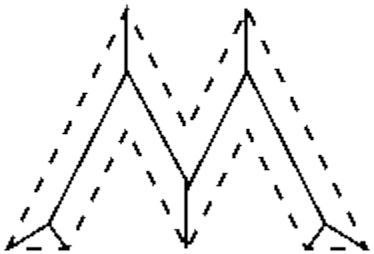
0	0	0	0	0	0	0
0	1	1	1	1	1	0
0	1	2	2	2	1	0
0	1	2	3	2	1	0
0	1	2	2	2	1	0
0	1	1	1	1	1	0
0	0	0	0	0	0	0

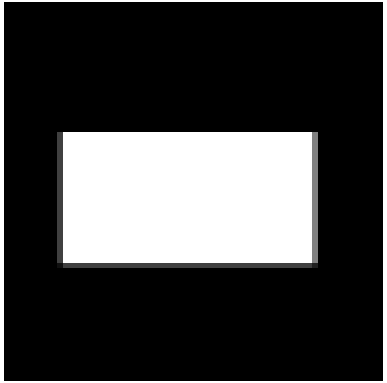
Distance transformation

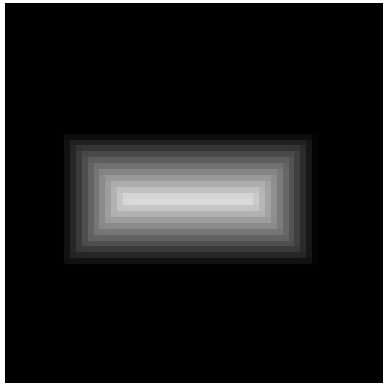


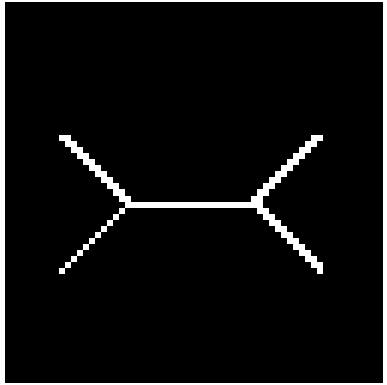


Skeletonization

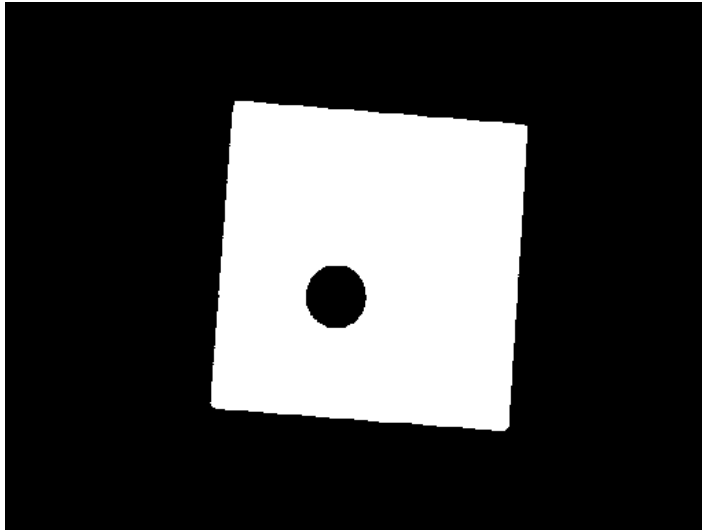


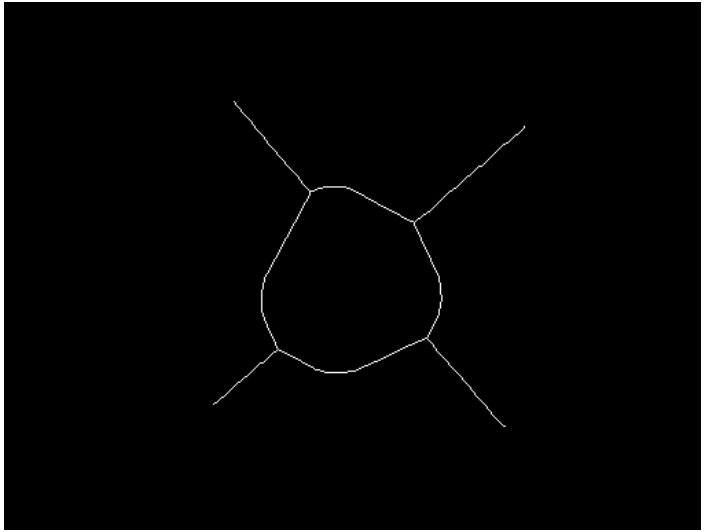




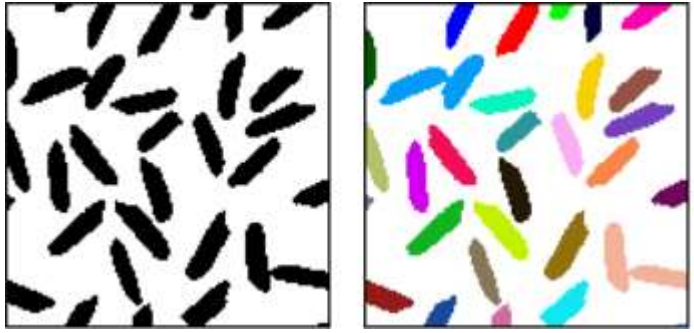


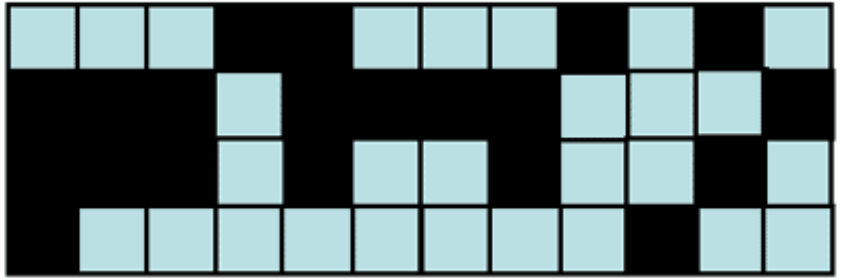
Skeletonization - Example



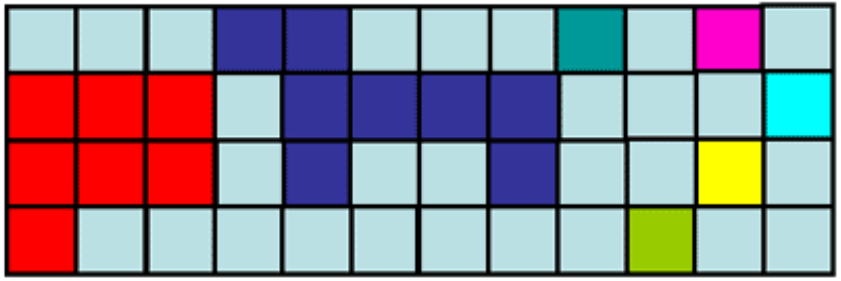


Components

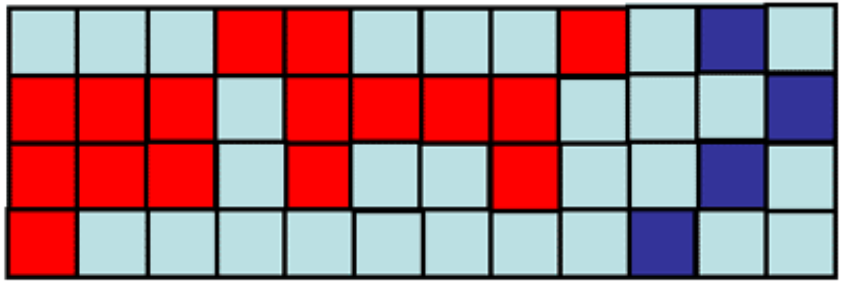




Binary image:
0 - objects;
1 -background

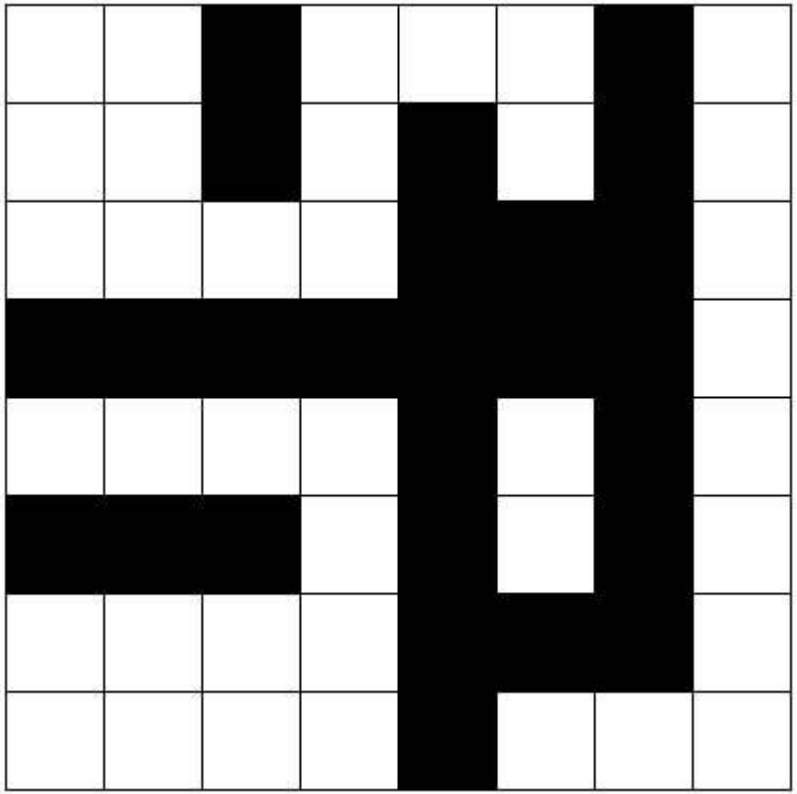


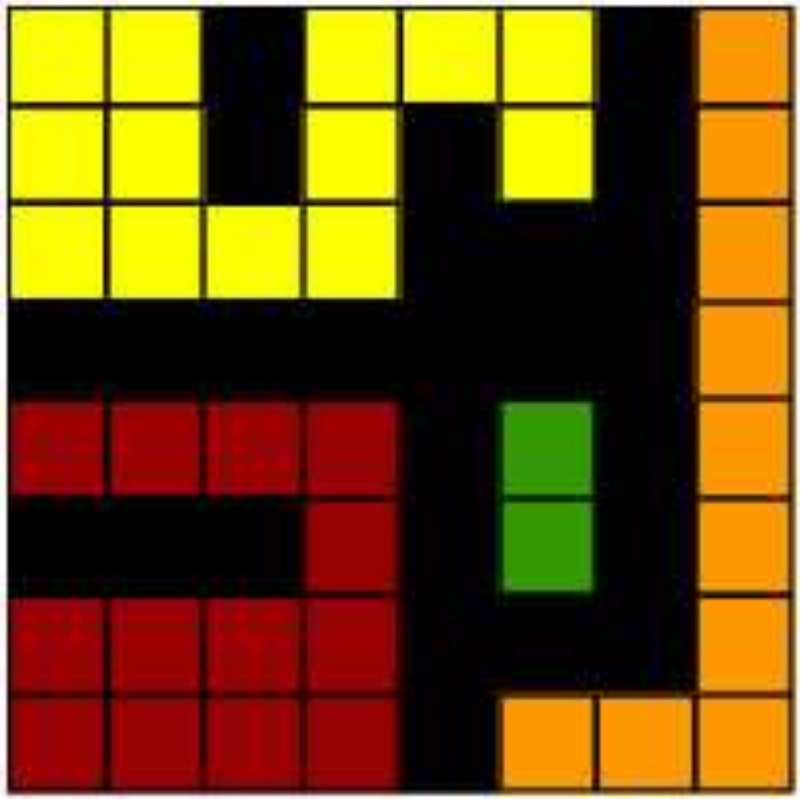
4-connected
objects +
8-connected
background



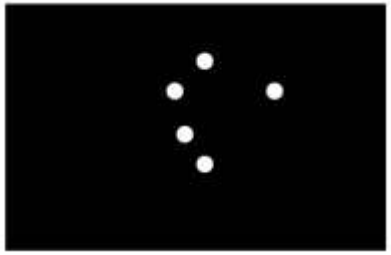
8-connected
objects +
8-connected
background

Connected Component Labeling



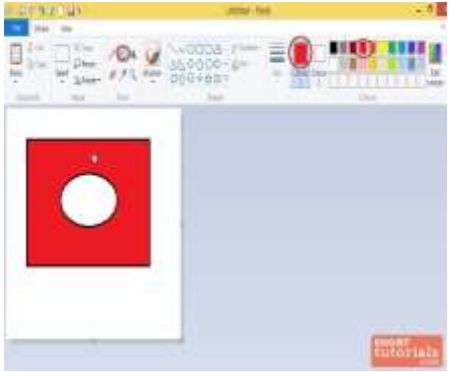


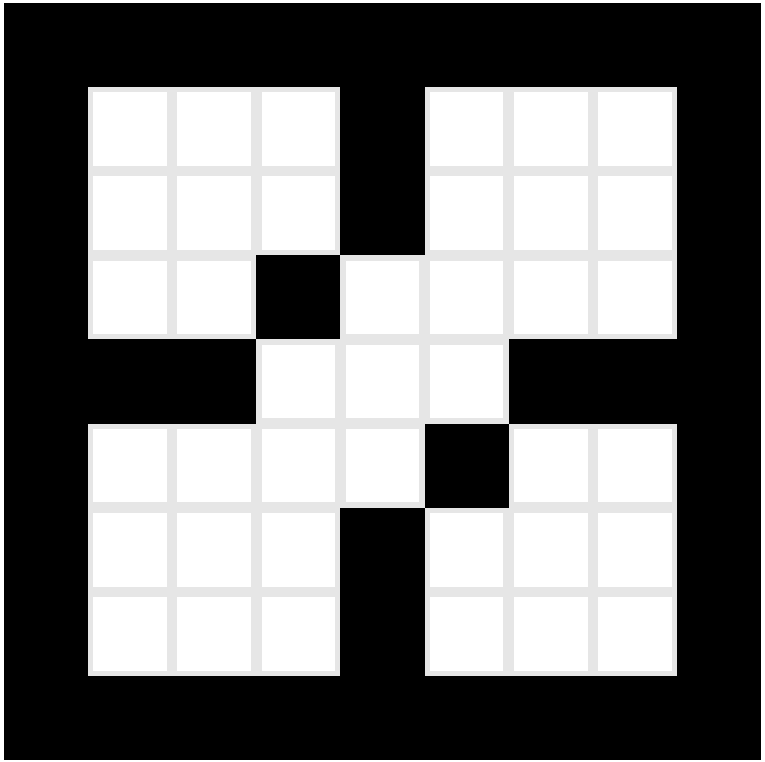
Connected Component Labeling



`bwlabel` `label2rgb`

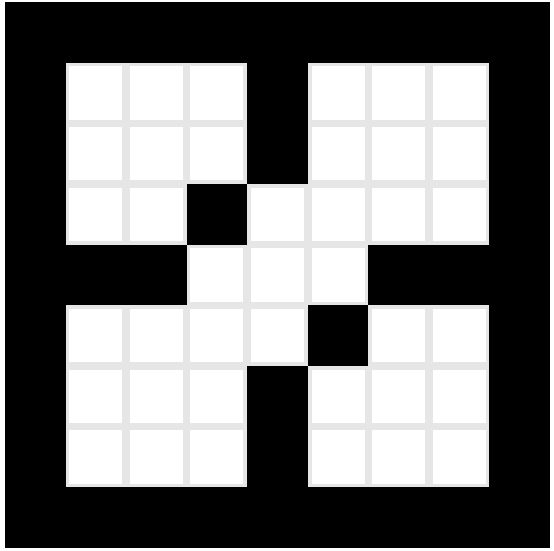
Flood-Fill





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

Flood-Fill Algorithm (4-conn)

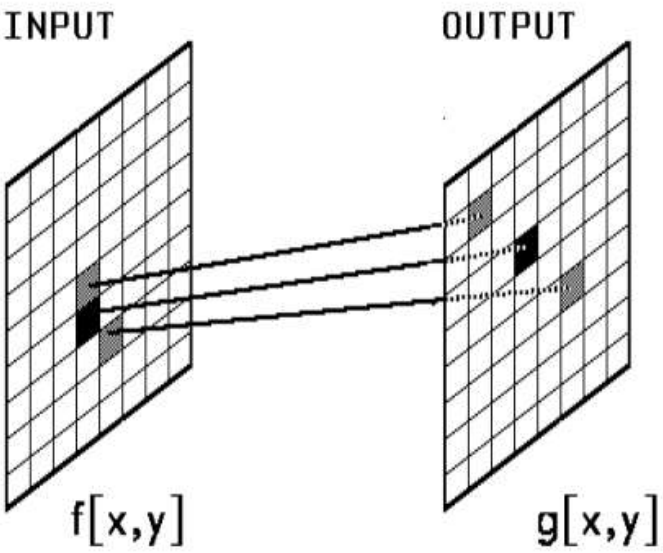


```
void floodFill(int x, int y, int fill, int old)
{

}
```

getPixel(x,y)
setPixel(x,y,color)

Geometric Operations

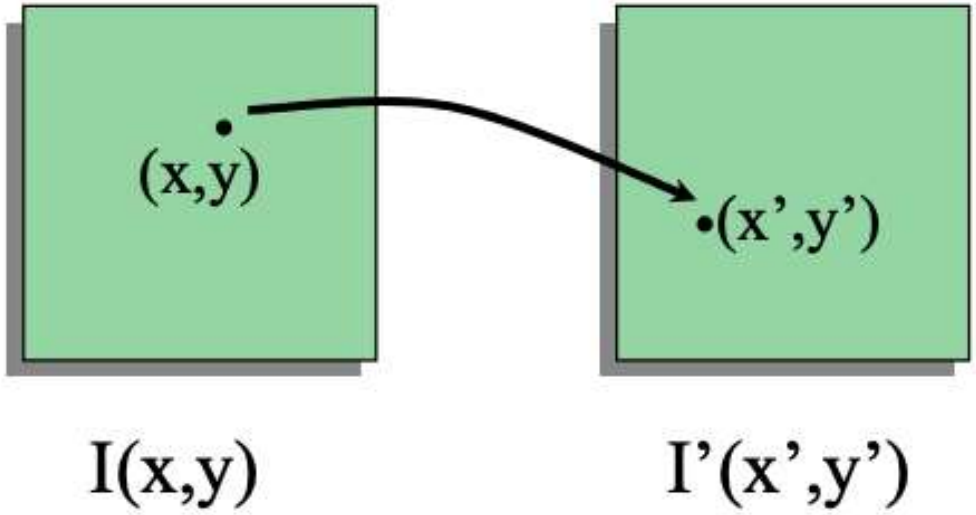
- Filters, Point Operations
  - modify color values (range) of pixels
  - domain (x,y) remains (mostly) fixed
- **Geometric transformation**
  - modify the positions of pixels
  - .. but keep their colors (mostly) unchanged



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

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

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



Common Geometric Operations

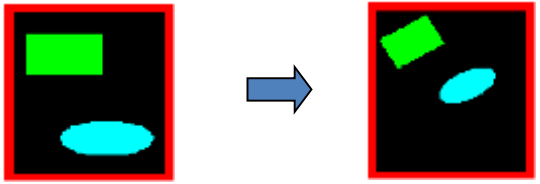
- **Scale** - change image content size

- **Rotate** - change image content orientation

- **Reflect** - flip over image contents

- **Translate** - change image content position

- **Affine Transformation**
  - general image content linear geometric transformation



Simple Mappings

- **Translation:** (shift) by a vector (d_x, d_y)

$$\begin{aligned} T_x : x' &= x + d_x \\ T_y : y' &= y + d_y \end{aligned} \quad \text{or} \quad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} d_x \\ d_y \end{pmatrix}$$

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

Simple Mappings

- **Translation:** (shift) by a vector (d_x, d_y)

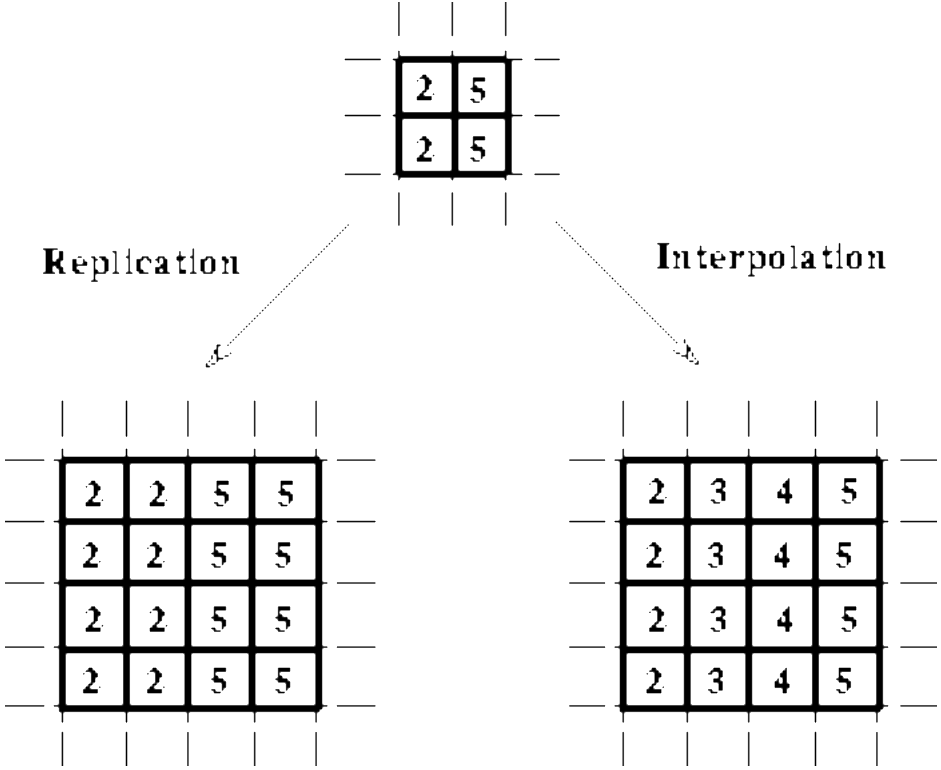
$$\begin{aligned} T_x : x' &= x + d_x \\ T_y : y' &= y + d_y \end{aligned} \quad \text{or} \quad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} d_x \\ d_y \end{pmatrix}$$

- **Scaling:** (contracting or stretching) along x or y axis by a factor s_x or s_y

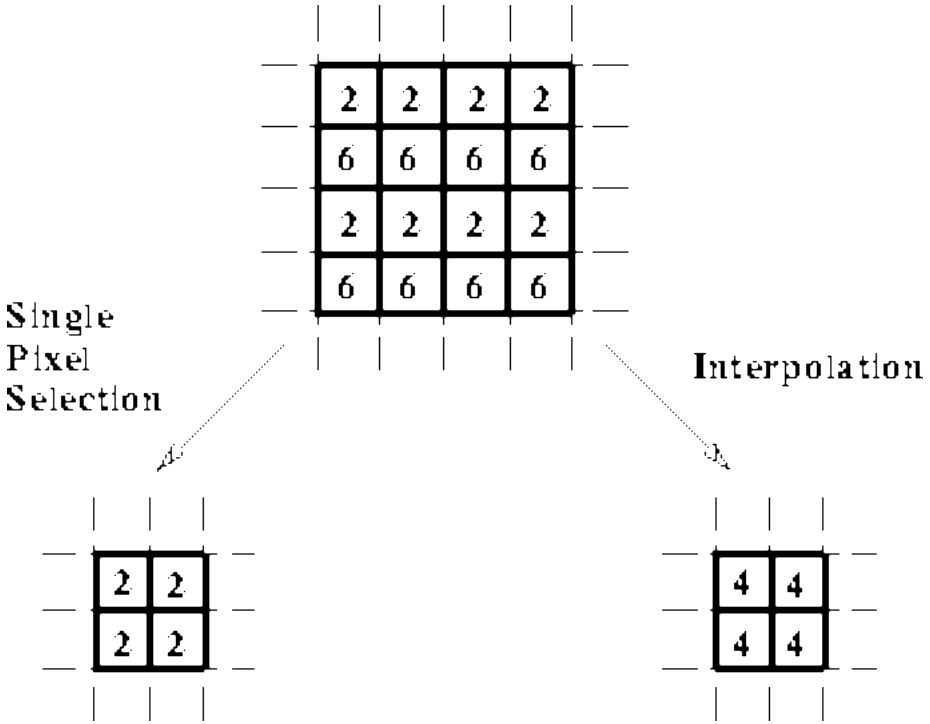
$$\begin{aligned} T_x : x' &= s_x \cdot x \\ T_y : y' &= s_y \cdot y \end{aligned} \quad \text{or} \quad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} s_x & 0 \\ 0 & s_y \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

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

Scaling (Stretch)



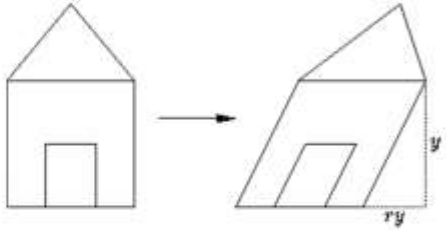
Scaling (Shrink)



Simple Mappings

- **Shearing:** along x and y axis by factor b_x and b_y

$$\begin{aligned} T_x : x' &= x + b_x \cdot y \\ T_y : y' &= y + b_y \cdot x \end{aligned} \quad \text{or} \quad \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} 1 & b_x \\ b_y & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$



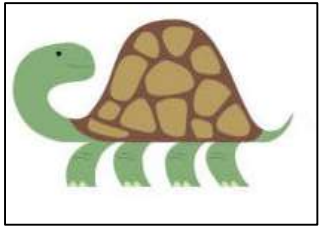
- **Rotation:** the image by an angle α

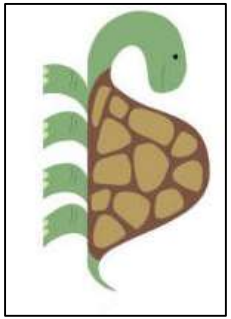
$$\begin{aligned} T_x : x' &= x \cdot \cos\alpha - y \cdot \sin\alpha \\ T_y : y' &= x \cdot \sin\alpha + y \cdot \cos\alpha \end{aligned}$$

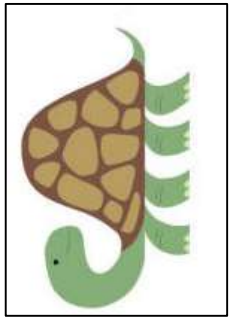
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

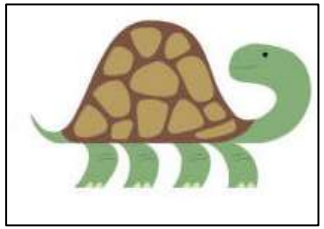
https://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/slides/lecture11.pdf

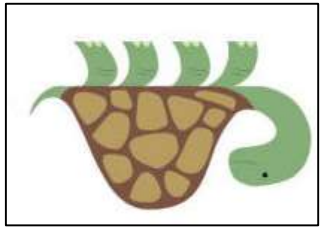
90, 180 rotations , Flipping

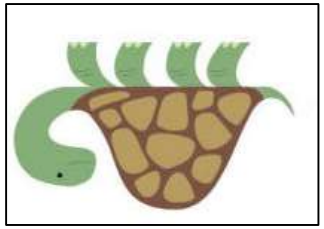






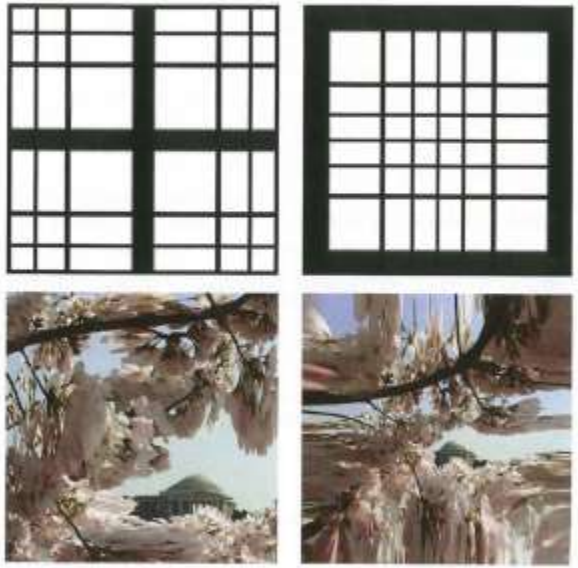






- **Image warping:** we can use a function to select which pixel somewhere else in the image to look up
- For example: apply function on both texel coordinates (x, y)

$$x = x + y * \sin(\pi * x)$$



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

Homogeneous Coordinates

- Notation useful for converting scaling, translation, rotating into point-matrix multiplication
- To convert ordinary coordinates into homogeneous coordinates

$$\boldsymbol{x} = \begin{pmatrix} x \\ y \end{pmatrix} \quad \text{converts to} \quad \hat{\boldsymbol{x}} = \begin{pmatrix} \hat{x} \\ \hat{y} \\ h \end{pmatrix} = \begin{pmatrix} h\,x \\ h\,y \\ h \end{pmatrix}$$

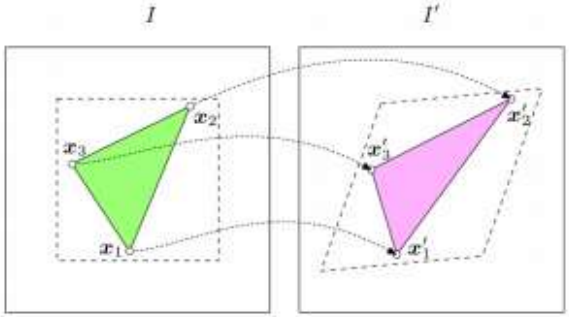
https://web.cs.wpi.edu/~emmanuel/courses/cs545/S14/slides/lecture11.pdf

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)

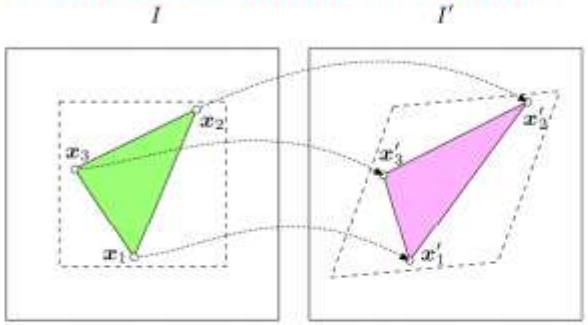


- Inverse of transform matrix is **inverse mapping**

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

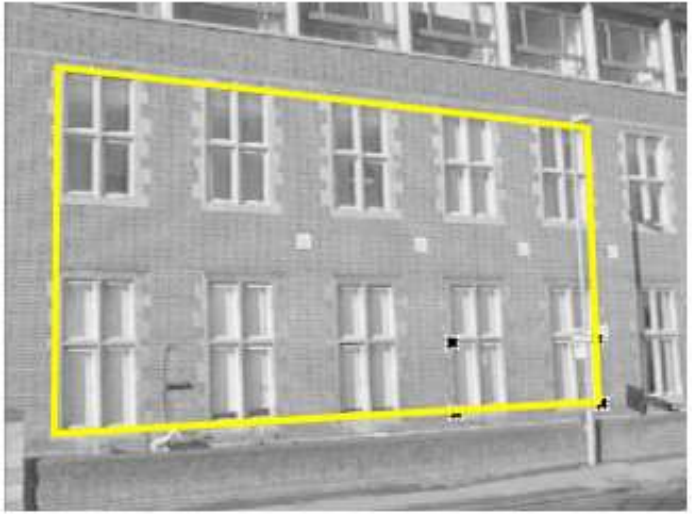
Affine (3-Point) Mapping

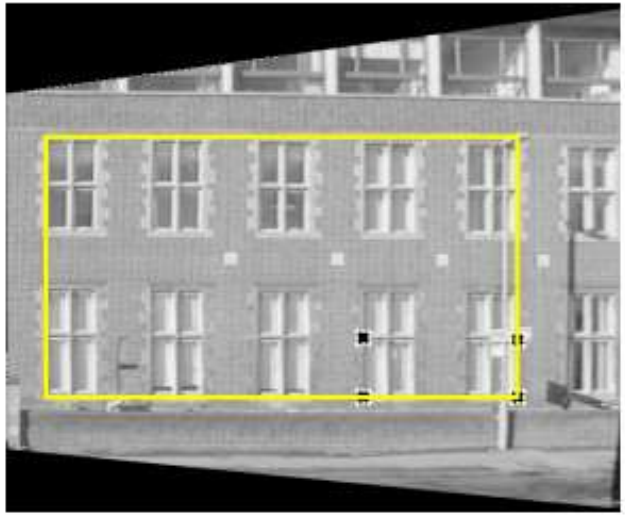
- **What's so special about affine mapping?**



- Maps
  - straight lines -> straight lines,
  - triangles -> triangles
  - rectangles -> parallelograms
  - Parallel lines -> parallel lines
- Distance ratio on lines do not change

Homography



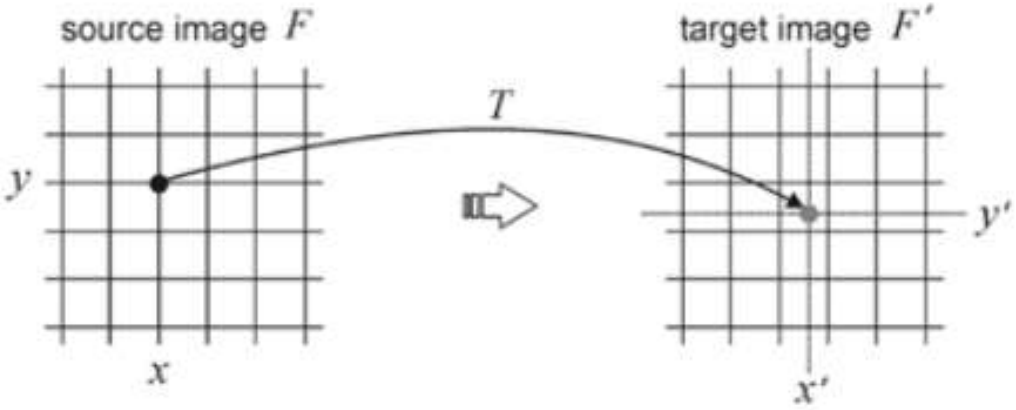


from Hartley & Zisserman

$$\begin{pmatrix} x_1' \\ x_2' \\ x_3' \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \times \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}$$

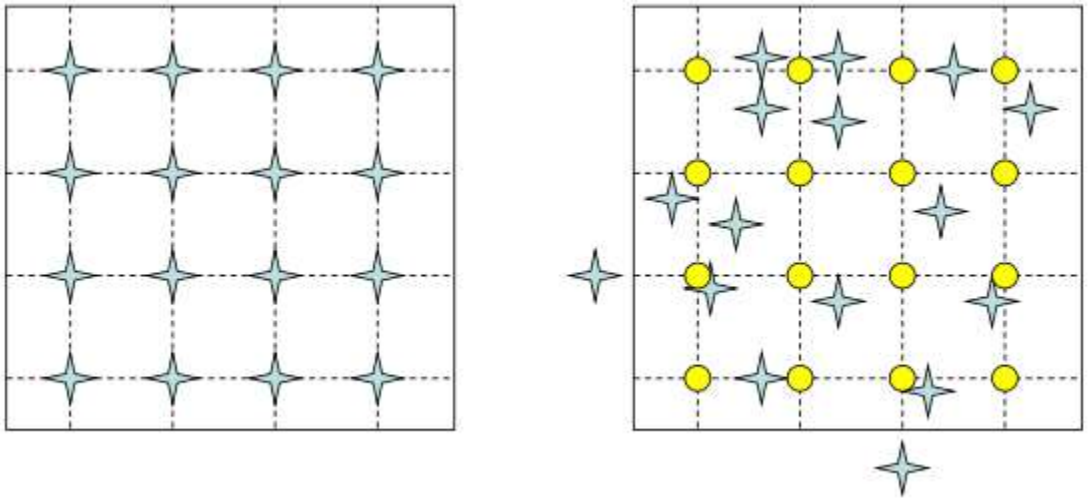
Interpolation methods

- 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

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.

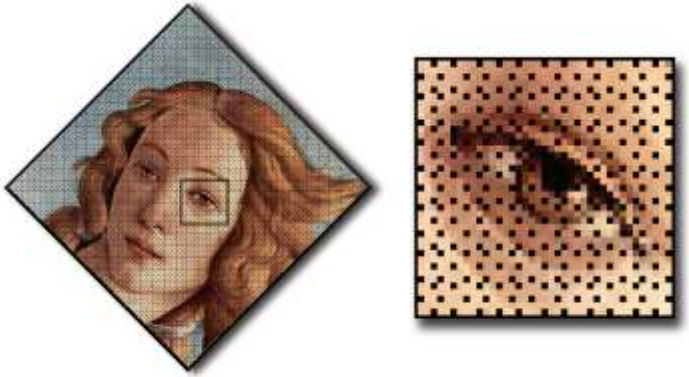
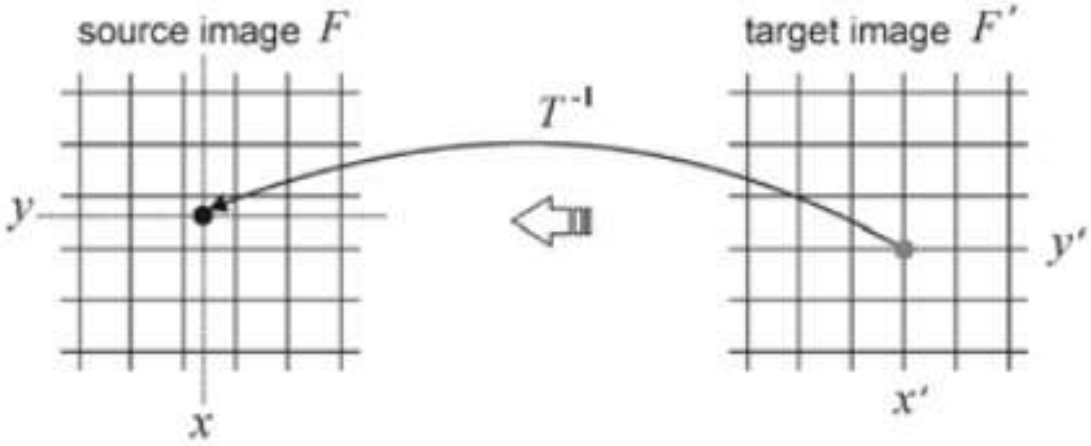


Figure 7.3. Rotation by forward mapping.

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

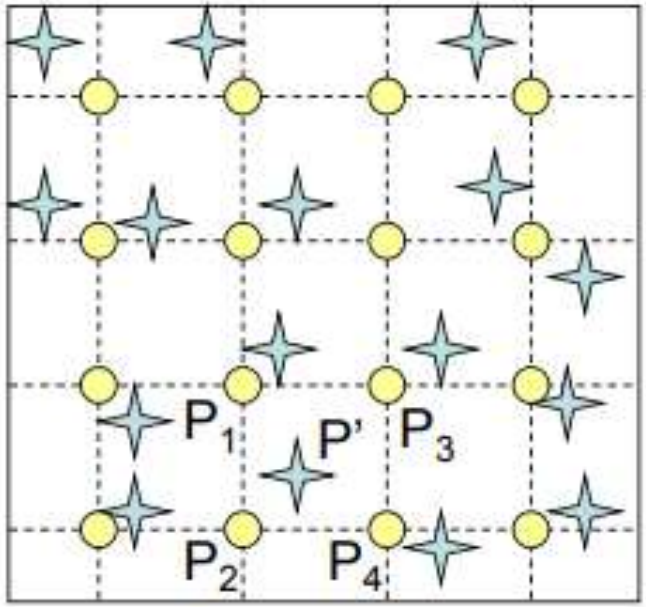
$v' = [x', y', 1]^T$

$$v' = Tv$$

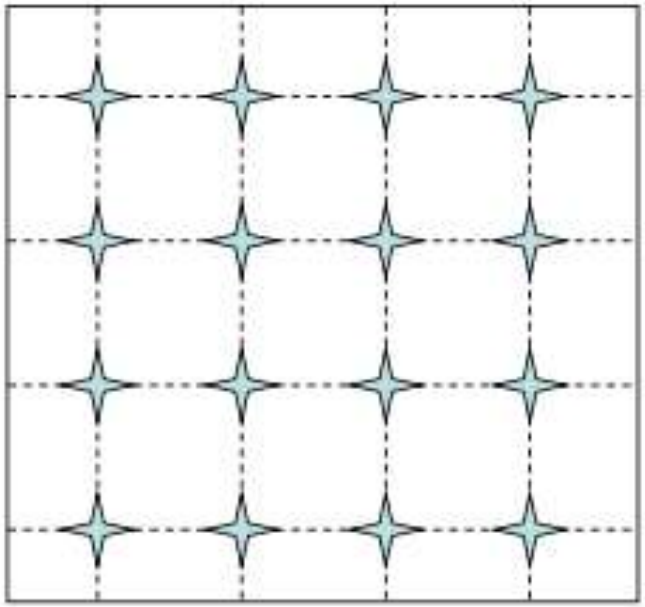
$$v = T^{-1}v'$$

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

Inverse Warping

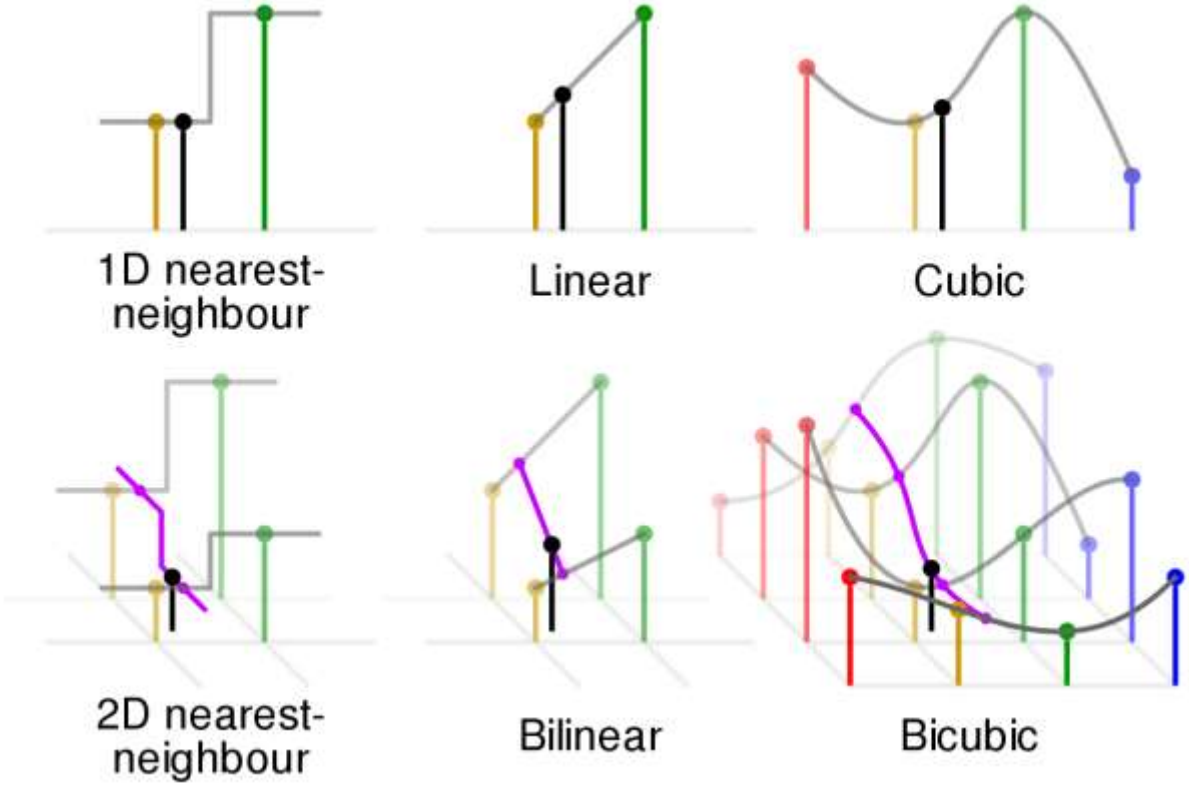


P' will be interpolated from P_1, P_2, P_3, and P_4

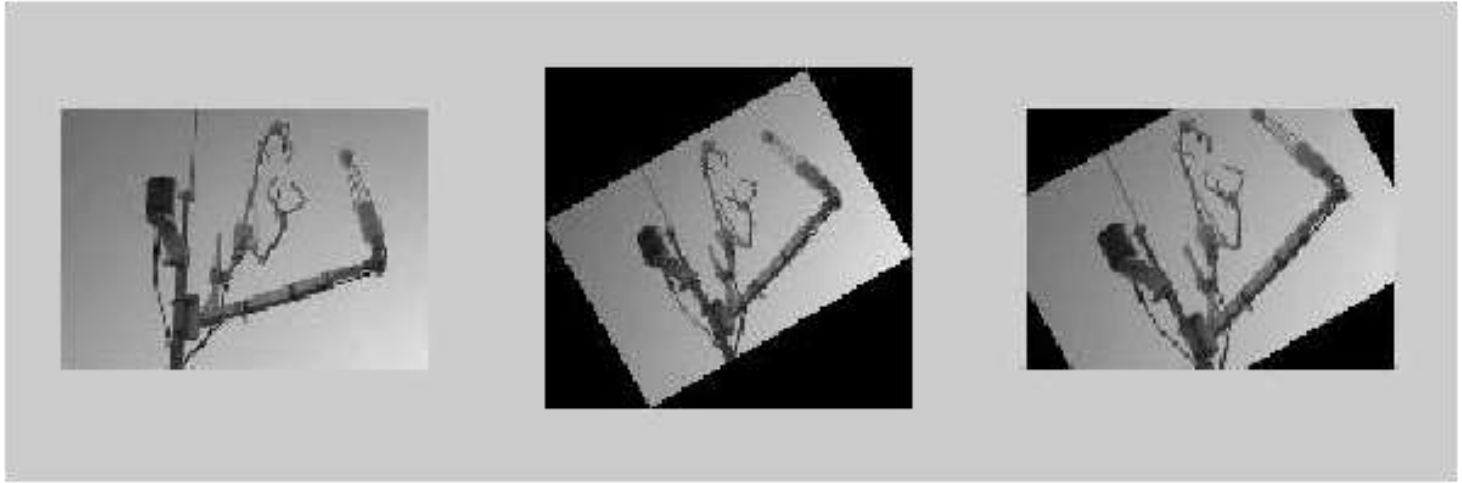


http://eeweb.poly.edu/~yao/EL5123/lecture12_ImageWarping.pdf

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 $\mathbf{A}$, $\theta = \pi/6$

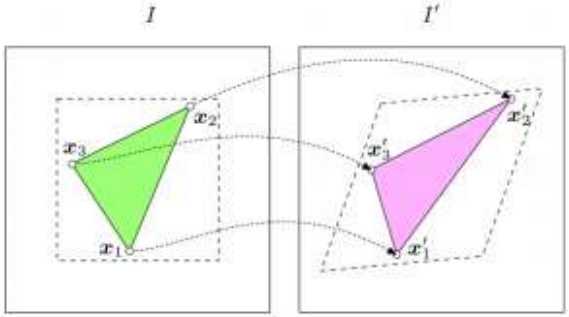
By Onur Guleyuz

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)

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

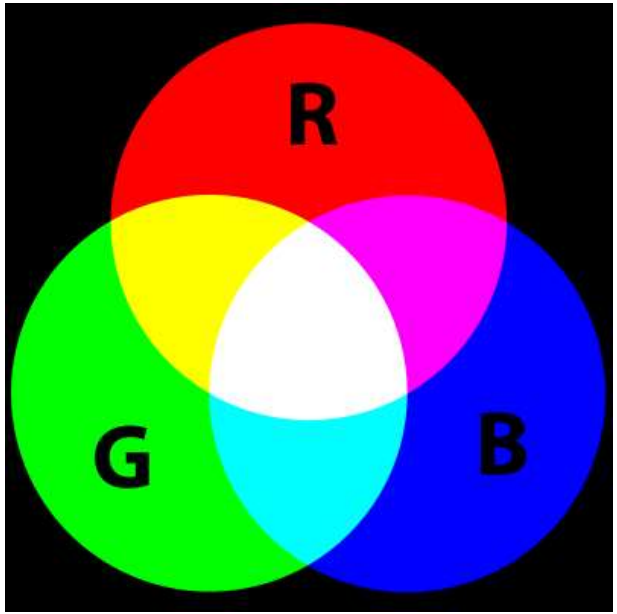
Geometric Operations

- 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

Primary Colors (color as three numbers)

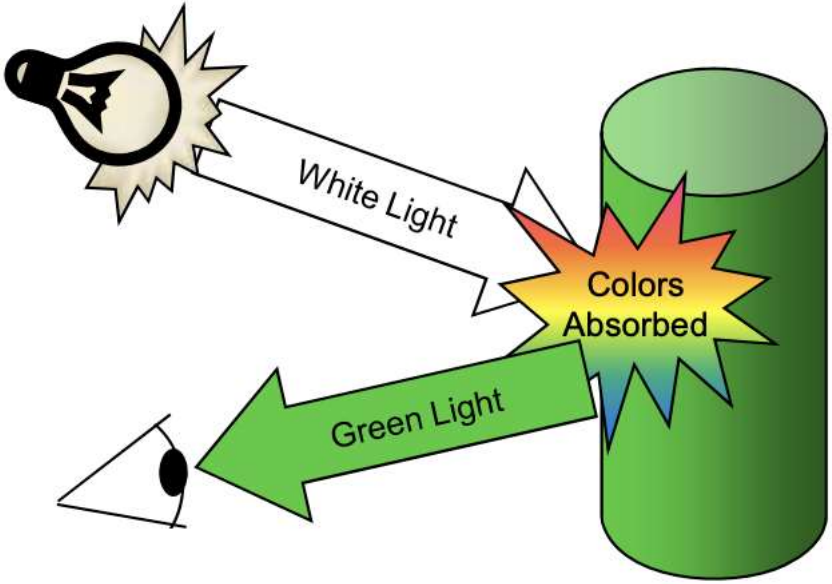
- Additive (CRT displays, projectors etc.)
  - Combining red, green and blue light produces lighter colors, offering a good contrast to dark screens.
  - (Usually) Combined on a surface which reflects all light that falls on it
  - Easy to get **black** – don't shine any light (CRT)

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



'Perceived' color

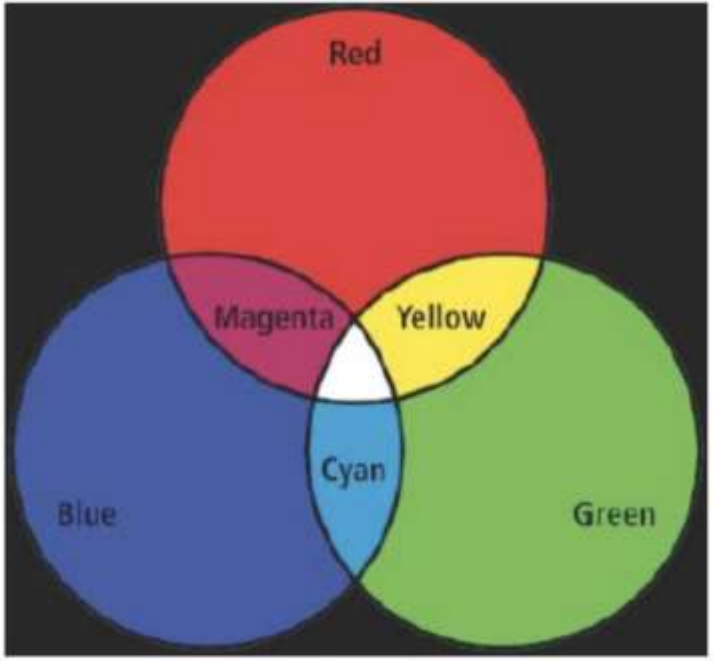
- Color is determined by nature of light reflected from an object
- E.g. green object reflects light from 500-570nm wavelength, absorbs other wavelengths



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

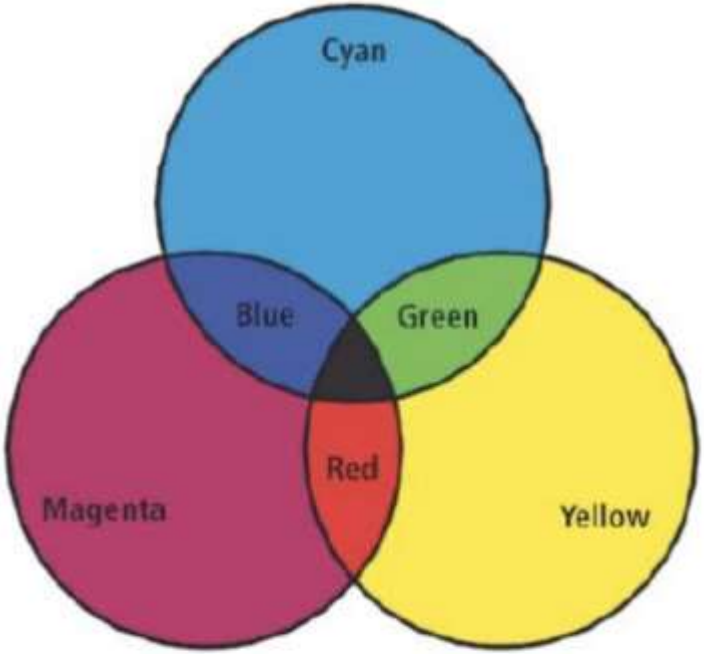
Courtesy: CNikou

Additive

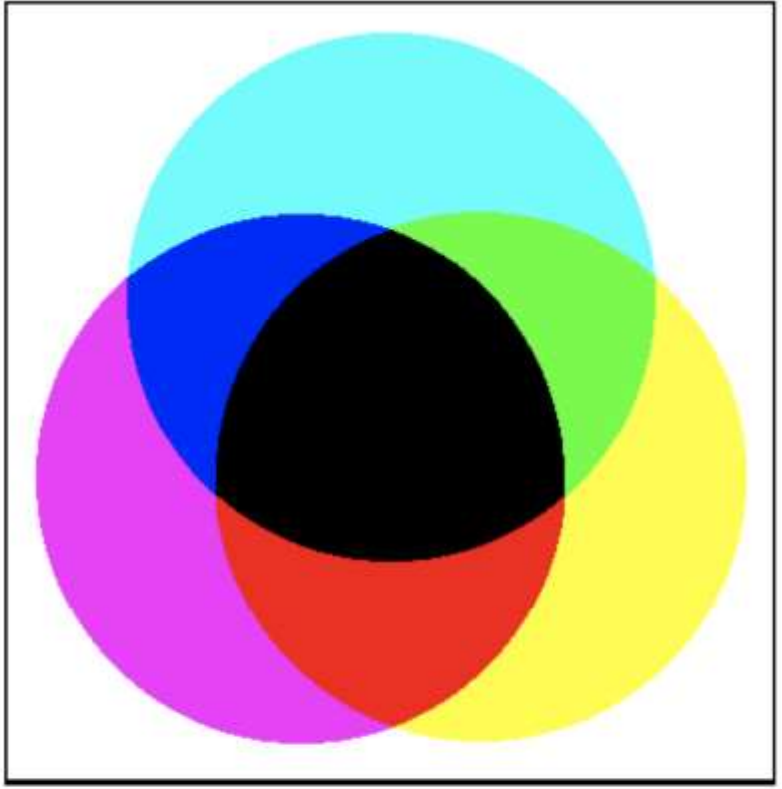


Magenta = Red + Blue
Cyan = Blue + Green
Yellow = Green + Red

Subtractive



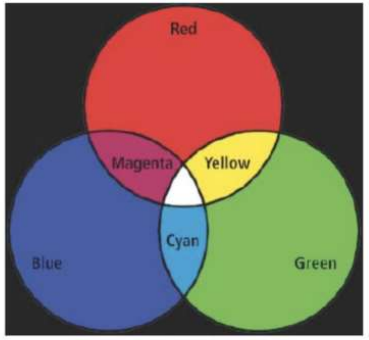
Magenta = White - Green
Cyan = White - Red
Yellow = White - Blue



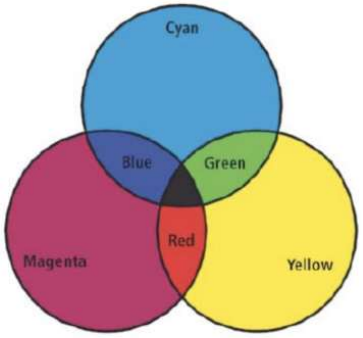
Passive displays, such as color inkjet printers, **absorb** light instead of emitting it. Combinations of **cyan**, **magenta** and **yellow** inks are used. This is a **subtractive** color model.

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



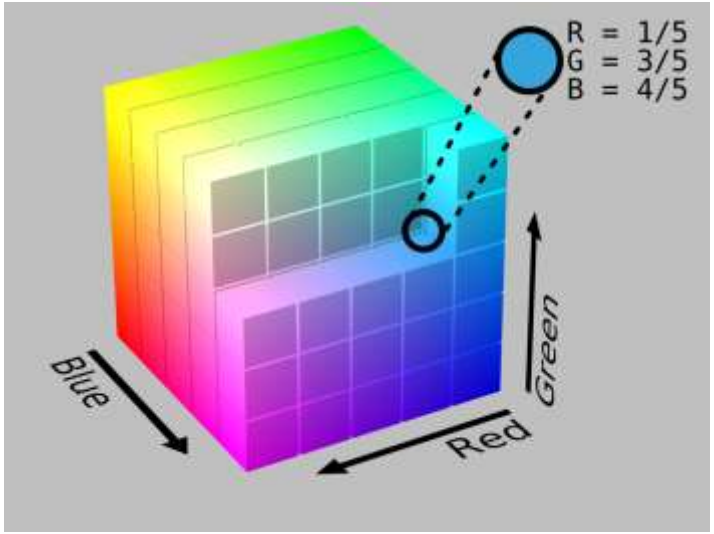
Magenta = White - Green
Cyan = White - Red
Yellow = White - Blue

Color Models

- RGB
- HSI / HSY
- CIE LAB

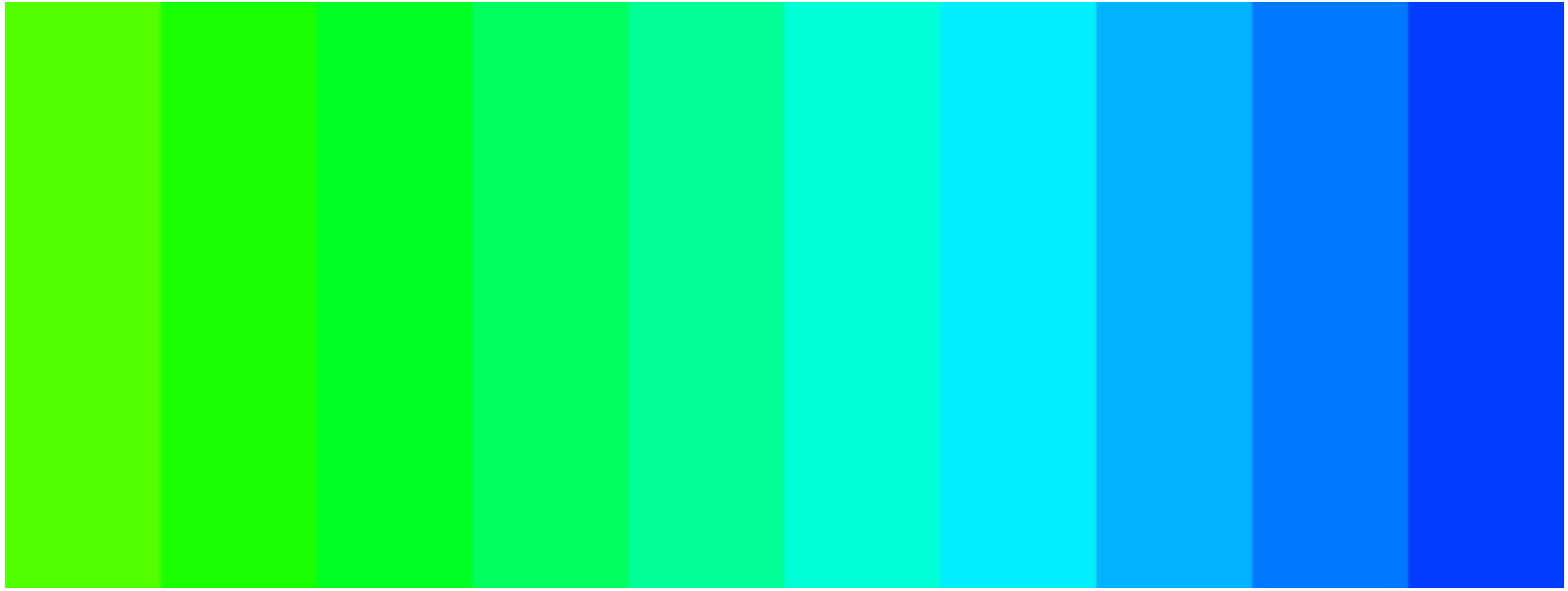
RGB color space

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

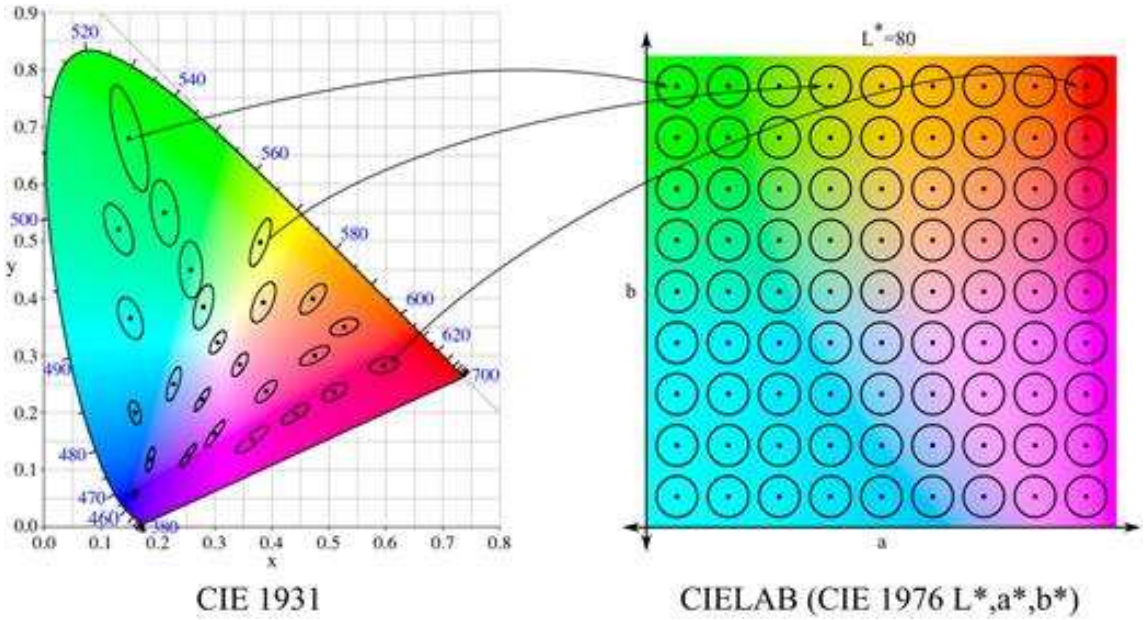


Courtesy: wikipedia

RGB : Non-Uniform Perceptual Space

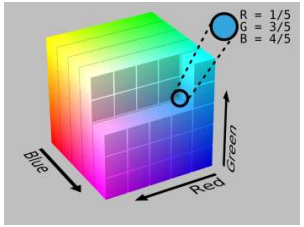


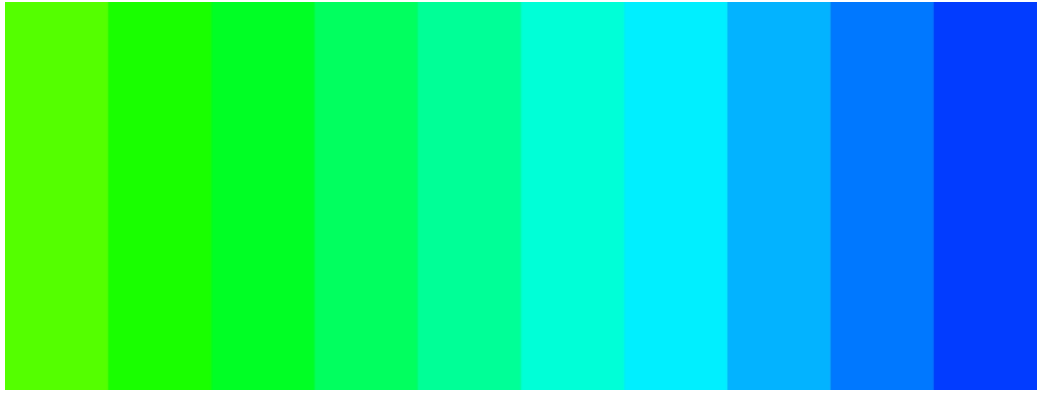
CIE Lab color space

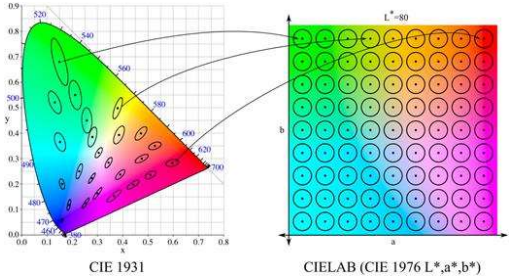


Ideal scenario

Perceptually Uniform Color Spaces



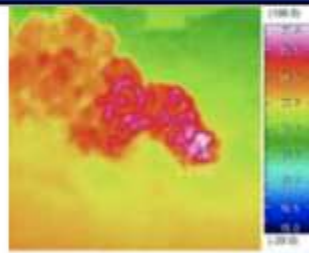




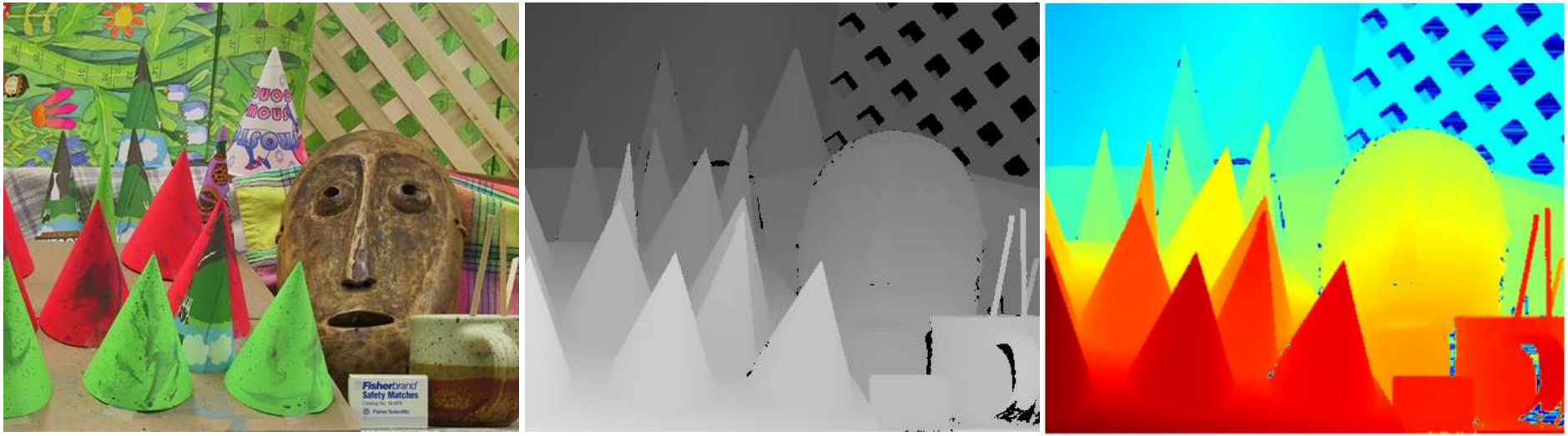
https://matplotlib.org/3.1.1/tutorials/colors/colormaps.html

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.

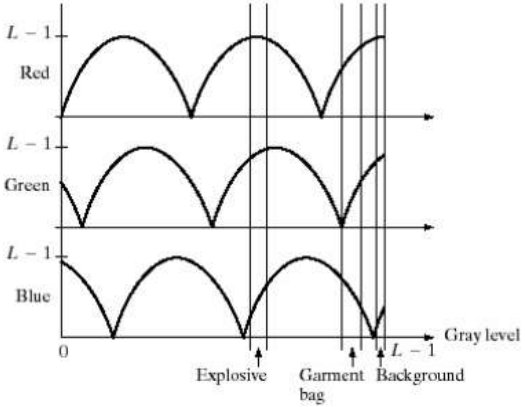


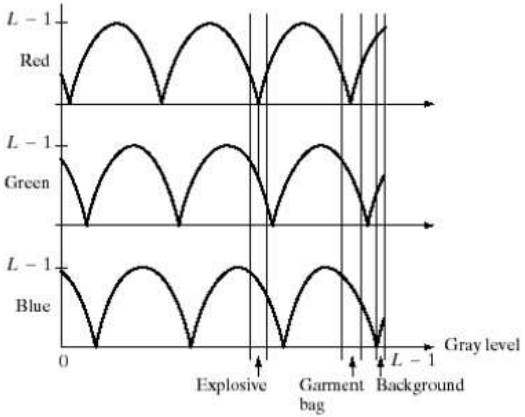
Pseudo color Image Processing

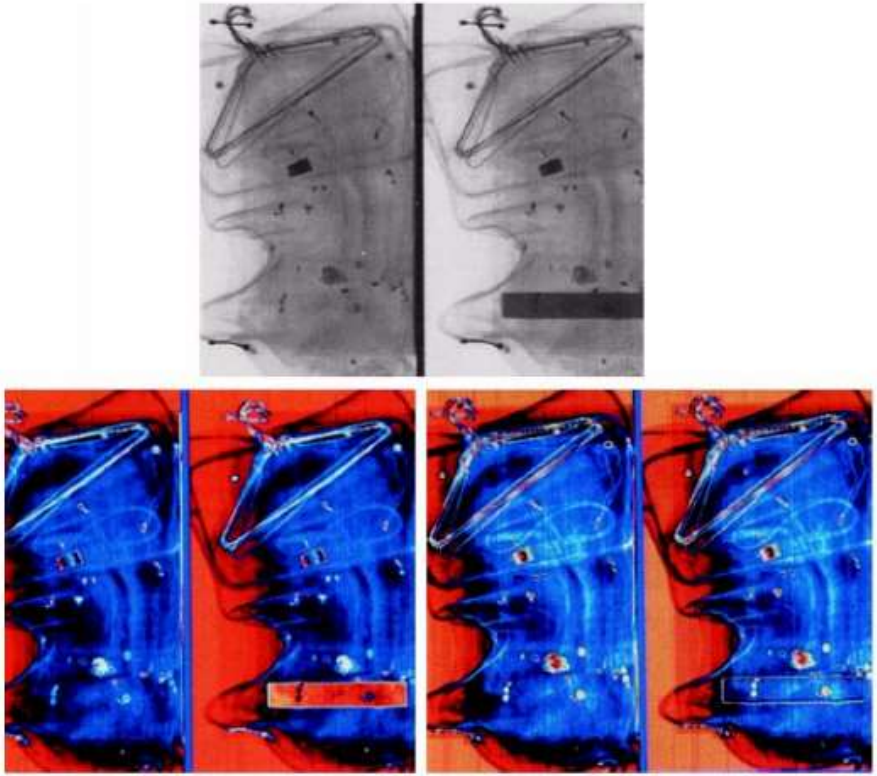


courtesy: middlebury dataset

Pseudo color Image Processing (Transformations)

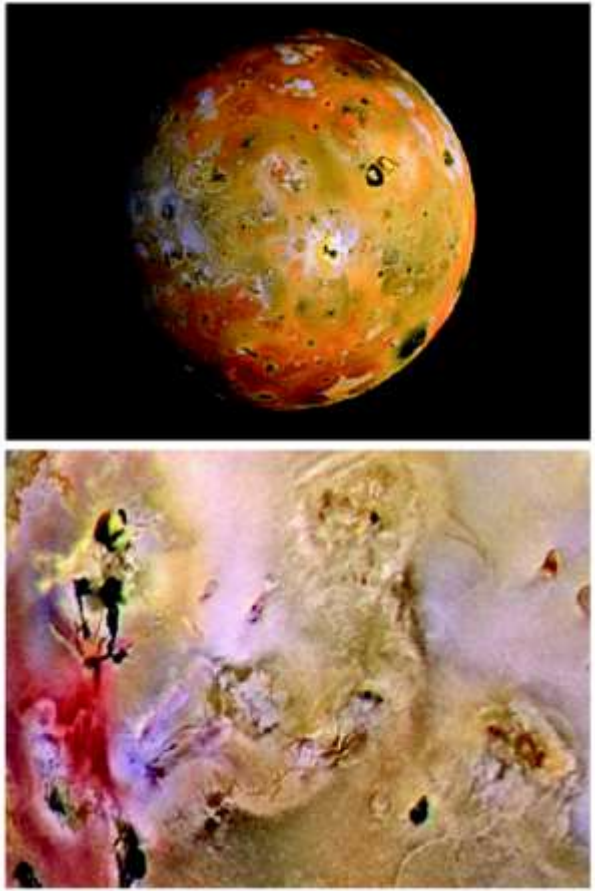






courtesy: Gonzalez and Woods

Pseudo color Image Processing (Multi Spectral)



courtesy: Gonzalez and Woods

RGBA space

- A (alpha) for transparency (important in image editing)

$$I_{out} = \alpha I_{foreground} + (1 - \alpha) I_{background}$$

Trending applications: Image enhancement in RGB

Example: Vintage effect

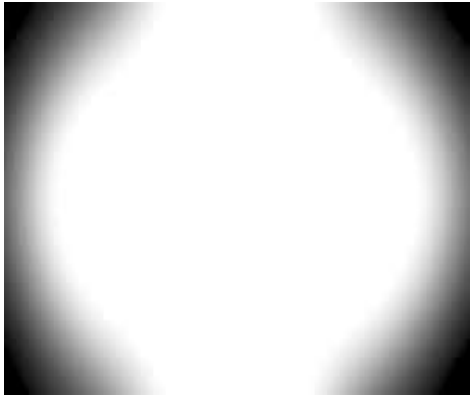
Example: Vintage effect

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

% Extract each colour plane
R = im(:,:,1); % Red
G = im(:,:,2); % Green
B = im(:,:,3); % Blue

% Create sepia tones for each channel
%(these number can be edited to create different styles)
outR= (R * .293) + (G *.769) + (B * .210);
outG = (R * .249) + (G *.686) + (B * .188);
outB = (R * .172) + (G *.534) + (B * .151);
```

Example: Vignetting effect

```
texture = imread(texture_path);
texture = imresize(texture,[size(out,1) size(out,2)]);
texture = double(rgb2gray(texture))/255;

out1(:,:,1) = double(out(:,:,1)) .* double(texture);
out1(:,:,2) = double(out(:,:,2)) .* double(texture);
out1(:,:,3) = double(out(:,:,3)) .* double(texture);
```

×



=

E.g. Contrast enhancement in color images:

- **Basic (popular) approach:**
  - ⇒ human eye - 5 more sensitive to brightness contrast then color contrast
  - ⇒ can achieve good contrast enhancement on brightness component alone!
  - ⇒ typically:

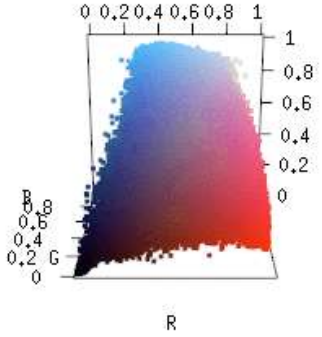
Grayscale quantization

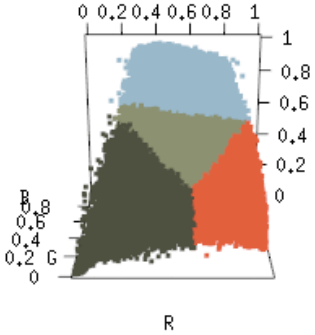
Original
(256 colors)

8 colors

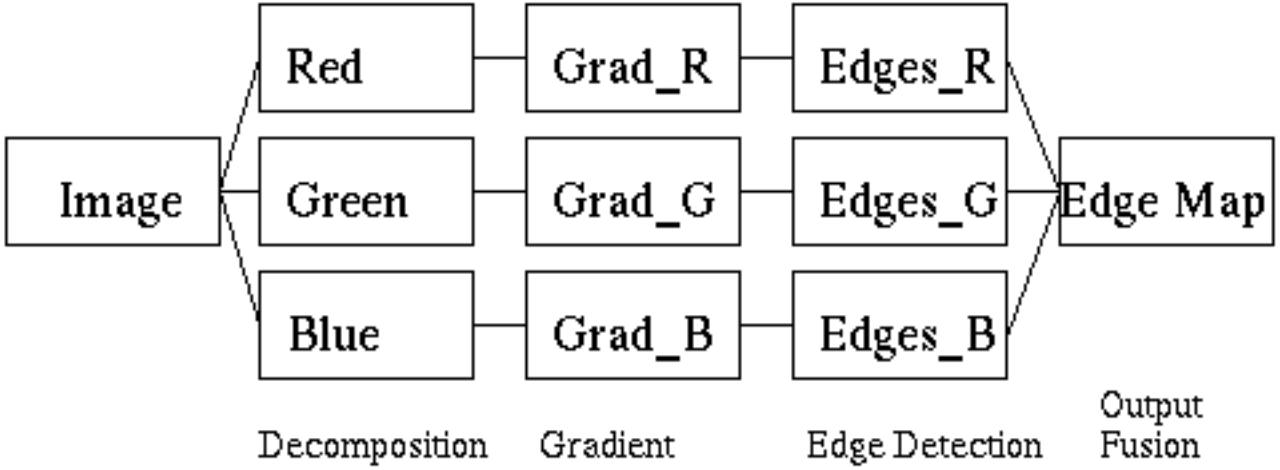
4 colors

Color Quantization (k-means on RGB)

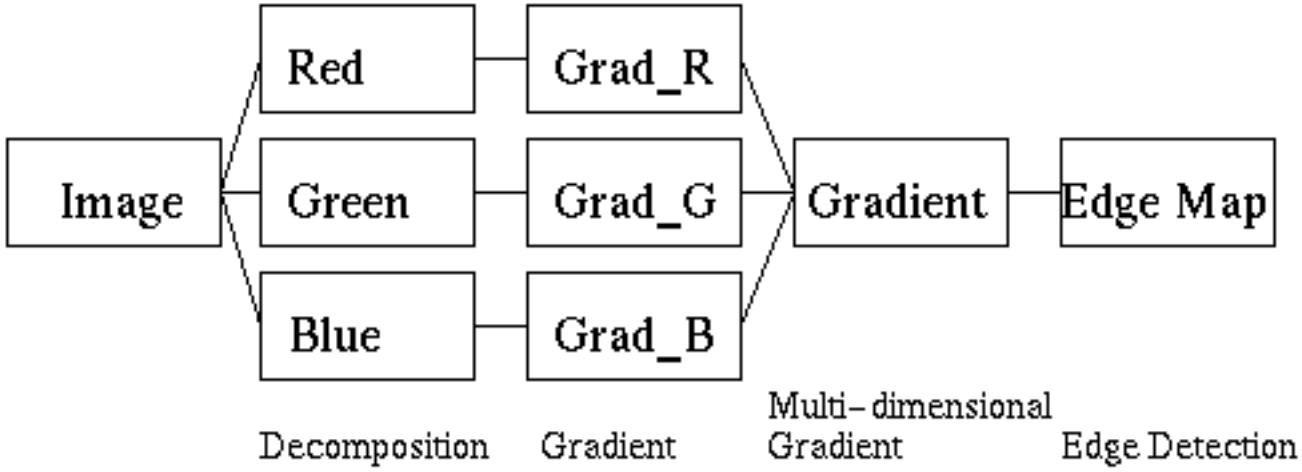




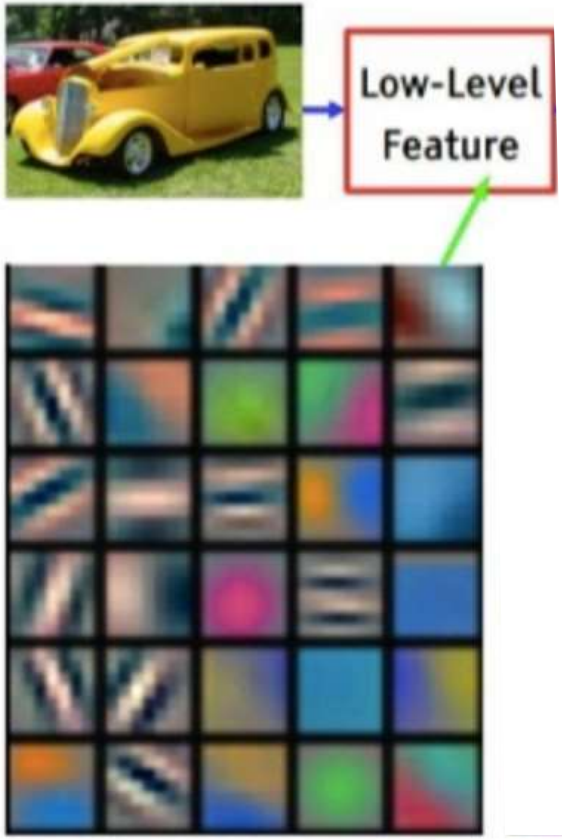
Color Edge Detection



Color Edge Detection



Convolutional Neural Network



Edge Detection Example With Smoothing

Original Image

Horizontal Gradient Component

Vertical Gradient Component

Combined Edge Image

Oriented Line Detection

−1	−1	−1
2	2	2
−1	−1	−1

Horizontal

−1	−1	2
−1	2	−1
2	−1	−1

+45°

−1	2	−1
−1	2	−1
−1	2	−1

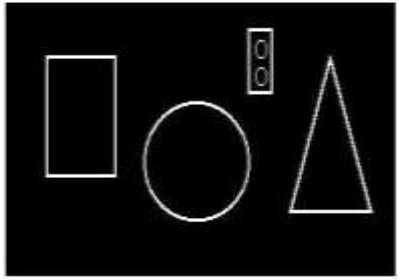
Vertical

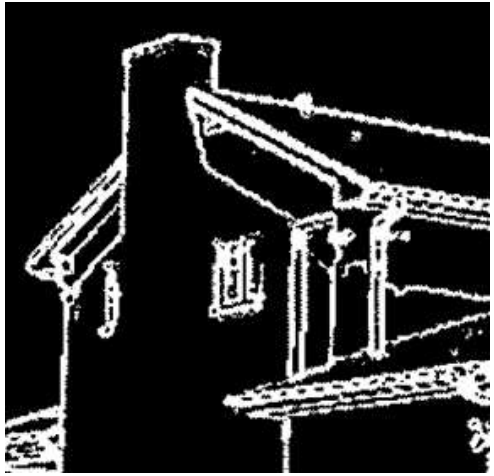
2	−1	−1
−1	2	−1
−1	−1	2

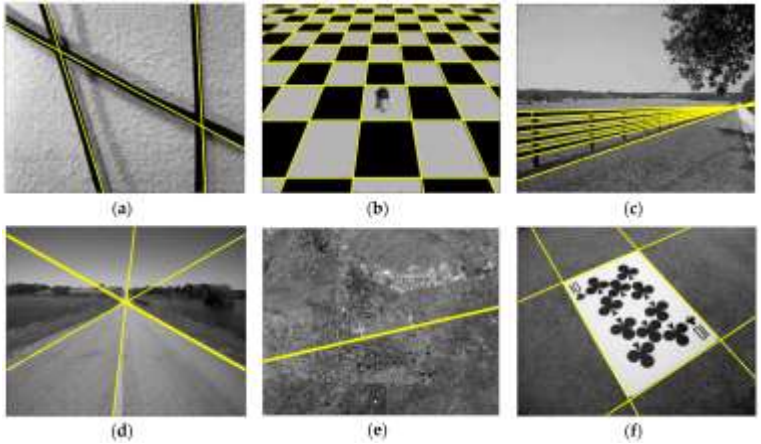
−45°

Hough Transform

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







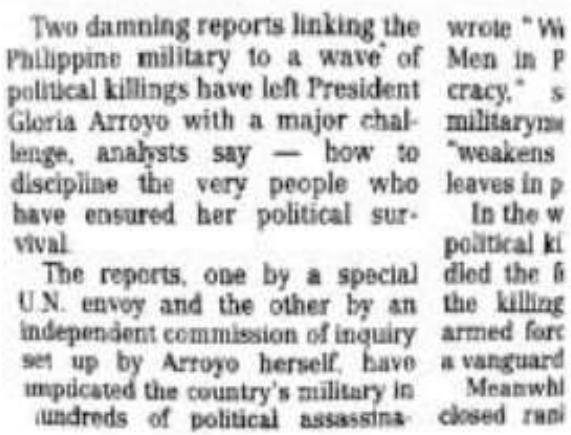
Segmentation: Thresholding based approaches

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?**

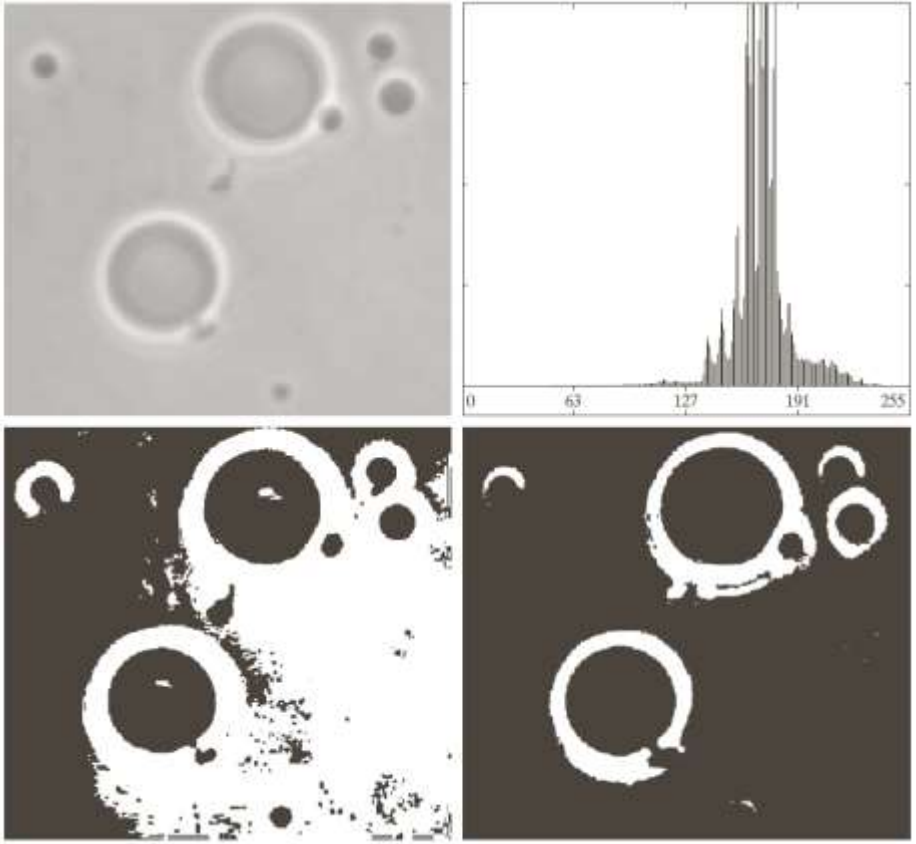
Choosing a threshold is something of a "black art":

```
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);
```

a b c

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

Otsu's Method

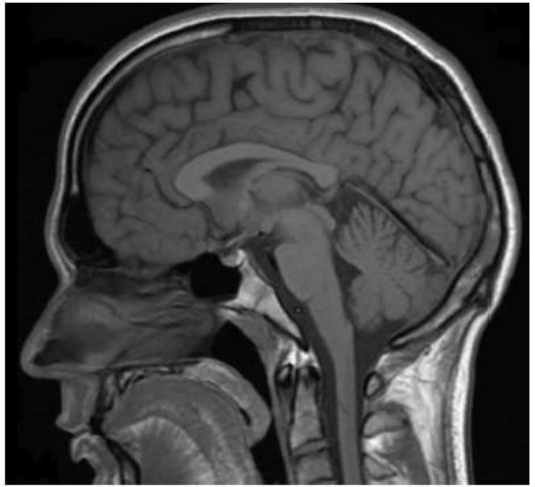


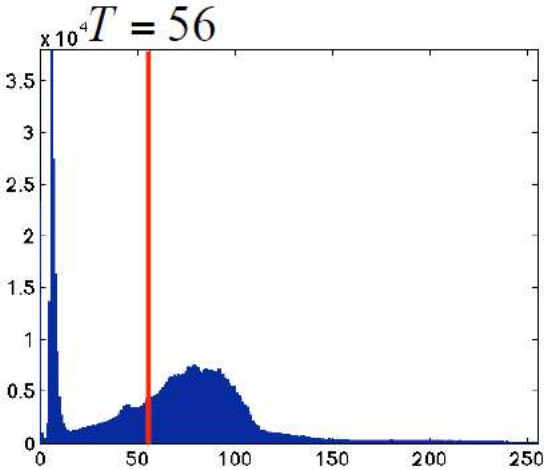
a b
c d

FIGURE 10.39 (a) Original image. (b) Histogram (high peaks were clipped to highlight details in the lower values). (c) Segmentation result using the basic global algorithm from Section 10.3.2. (d) Result obtained using Otsu's method. (Original image courtesy of Professor Daniel A. Hammer, the University of Pennsylvania.)

T = 181

Otsu’s Method





Global segmentation: main limitation

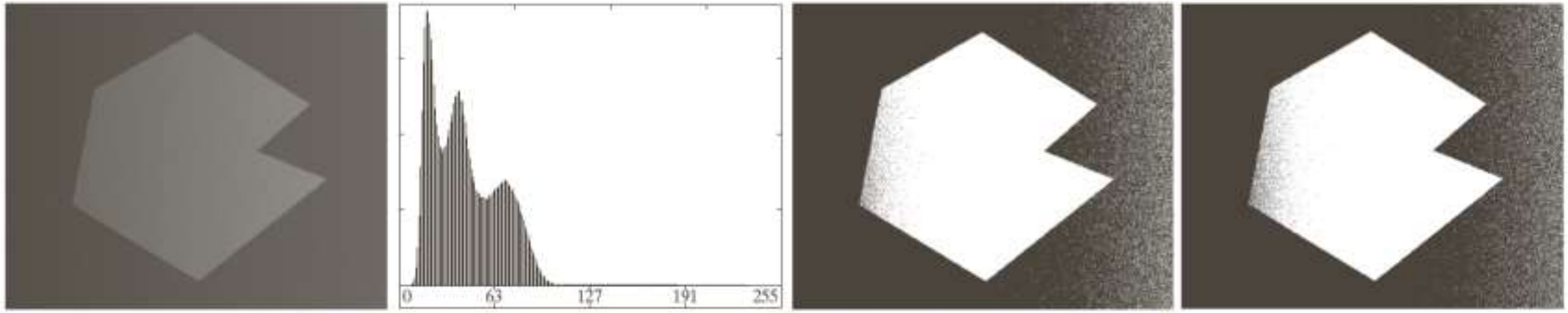
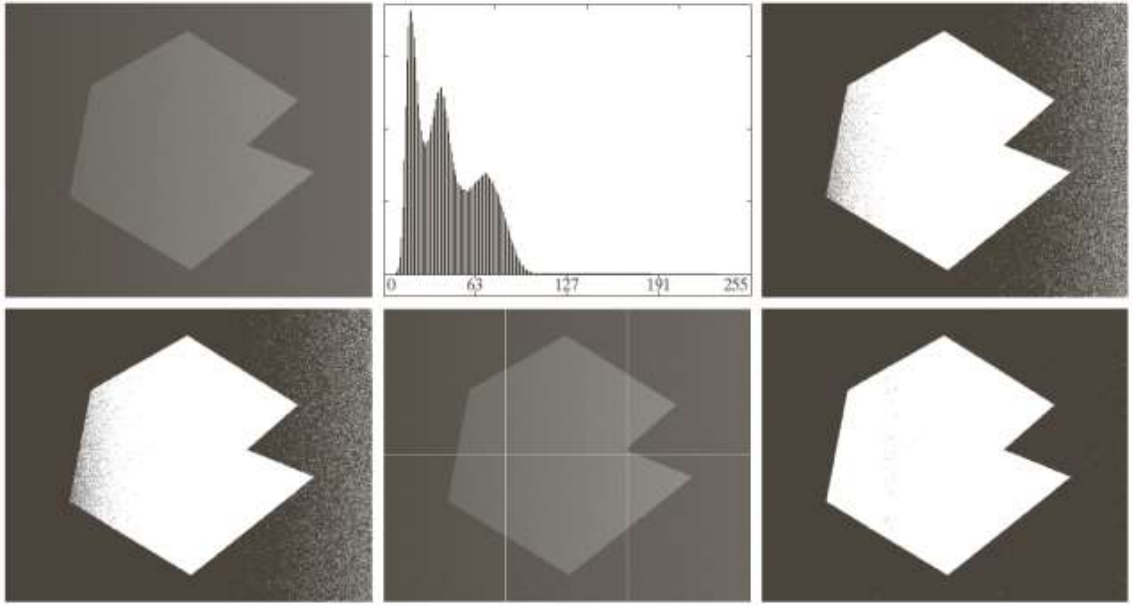


Image subdivision + variable Thresholding



a b c
d e f

FIGURE 10.46 (a) Noisy, shaded image and (b) its histogram. (c) Segmentation of (a) using the iterative global algorithm from Section 10.3.2. (d) Result obtained using Otsu's method. (e) Image subdivided into six subimages. (f) Result of applying Otsu's method to each subimage individually.

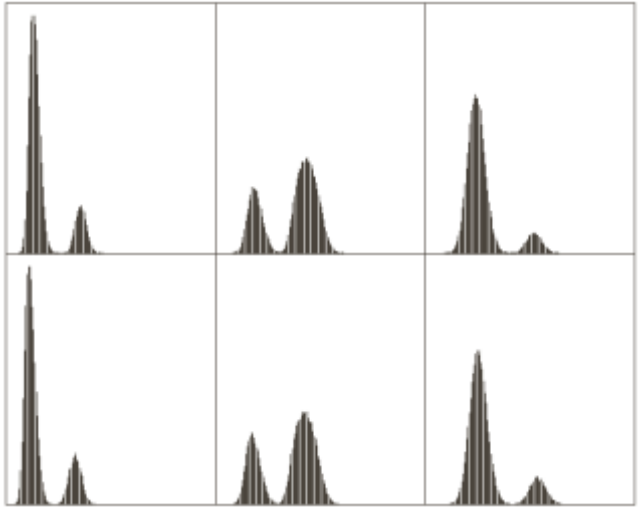
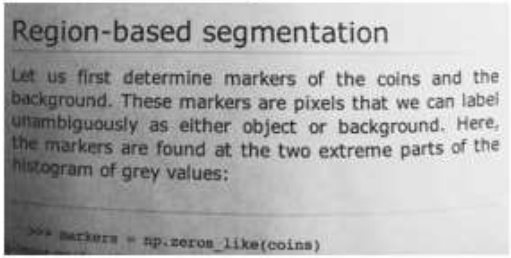


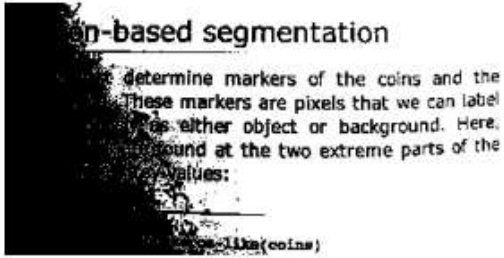
FIGURE 10.47 Histograms of the six subimages in Fig. 10.46(e).

Sauvola-Niblack Binarization

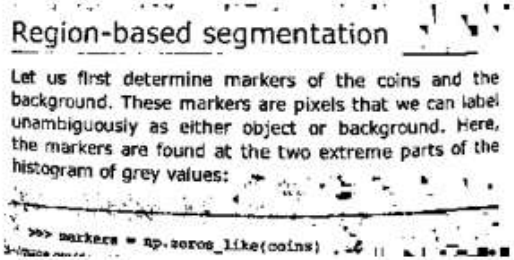
Original



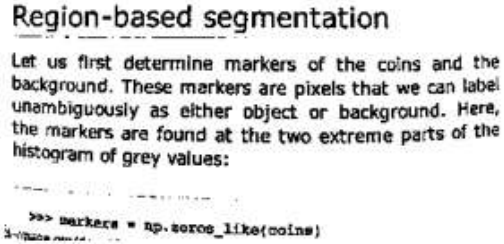
Global Threshold



Niblack Threshold



Sauvola Threshold



https://scikit-image.org/docs/0.24.x/auto_examples/segmentation/plot_niblack_sauvola.html

degraded images

ideal image

Blurred image

- What caused the image to blur?
  - Camera: translation, shake, out-of-focus …
  - Environment: scattered and reflected light
  - Device noise: CCD/CMOS sensor and circuitry
  - Quantization noise
- Can we improve the image, or "undo" the effects?

Degradations

- original

- optical blur

- motion blur

- spatial quantization (discrete pixels)

- additive intensity noise

a model for image distortion

- Image restoration
  - Use a priori knowledge of the degradation
  - Modeling the degradation and apply the inverse process
  - Formulate and evaluate objective criteria of goodness

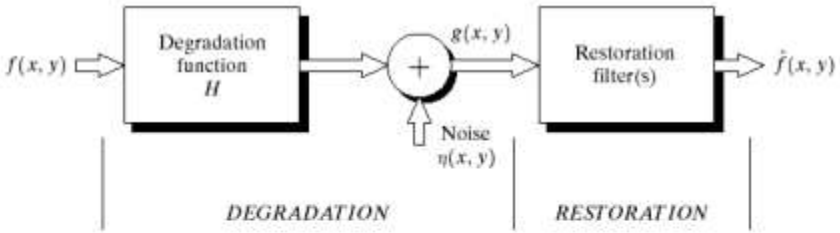


FIGURE 5.1 A model of the image degradation/restoration process.

$$g(x, y) = H[f(x, y)] + \eta(x, y)$$

$\longrightarrow$ design restoration filters such that $\hat{f}(x, y)$ is as close to $f(x, y)$ as possible.

who said distortion is a bad thing?

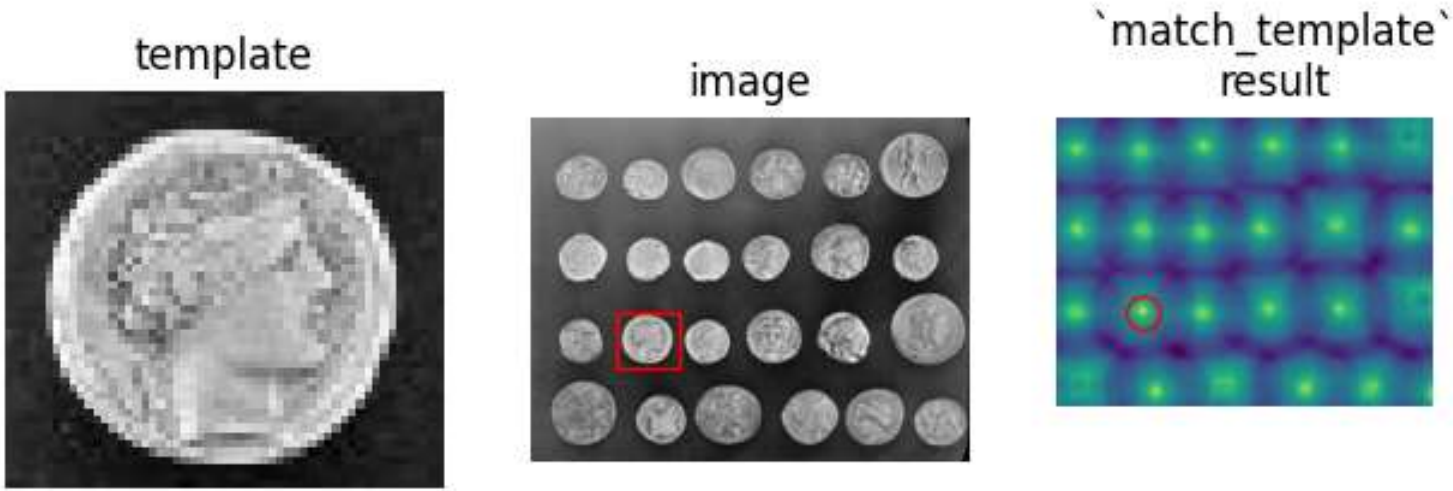
blur ...

noise ...

geometric ...

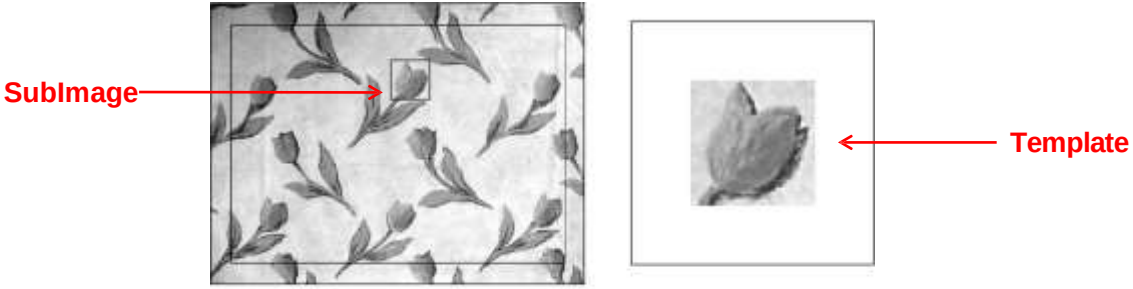
© Declan Mccullagh Photography, mccullagh.org

Template Matching



Template Matching

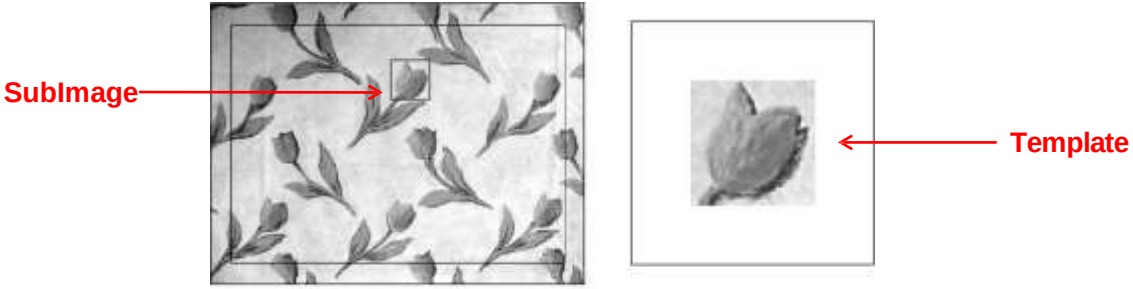
- Basic idea
  - Move given pattern (template) over search image
  - Measure difference between template and sub-images at different positions
  - Record positions where highest similarity is found



(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?

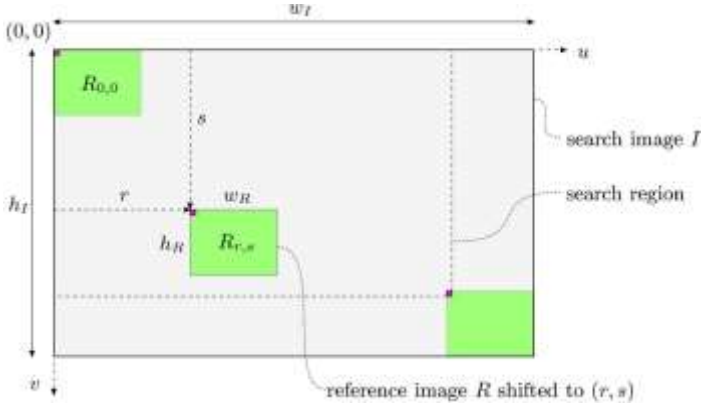


(a) original image I (b) reference image R

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

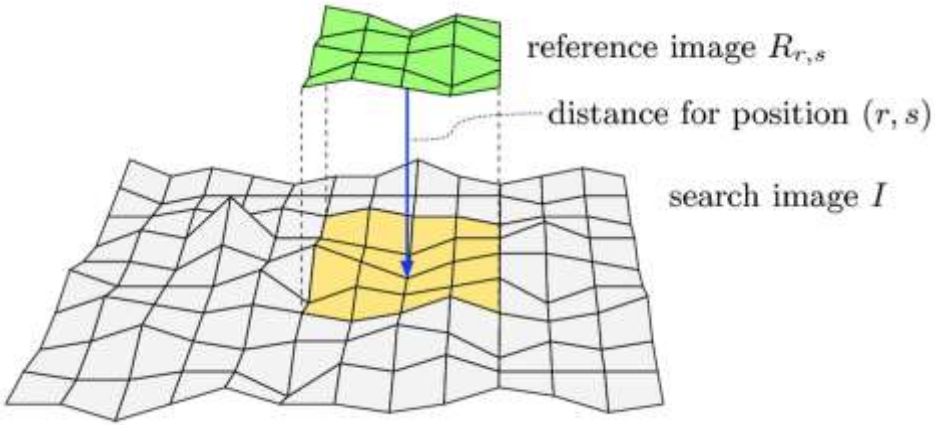
$$R_{r,s}(u,v) = R(u-r, v-s)$$



- Solving this problem involves solving many sub-problems

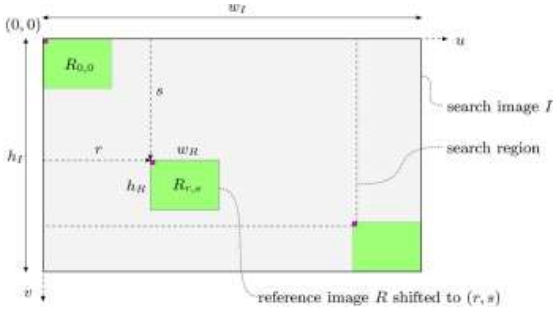
Distance between Image Patterns

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



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

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



<u>Normalized</u> Cross Correlation

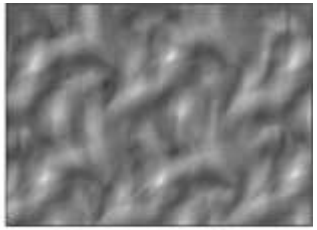
$$C_N(r,s) =$$

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

- $C_N(r,s)$ 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

Examples and Discussion

(g) normalized cross correlation



(h) correlation coefficient

(a) original image I

(b) reference image R

- **Normalized cross correlation**: (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. unaffected by lighting

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

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

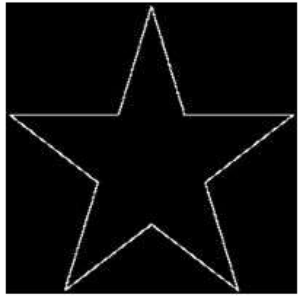
Matching under Rotation and Scaling

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

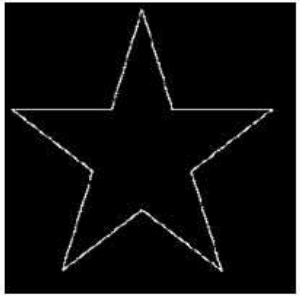
- Alternate approaches:
  - Matching in logarithmic-polar space (complicated!)
  - Affine matching: Use local statistical features invariant under affine image transformations (including rotation and scaling)

Detecting Shapes Without Texture

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

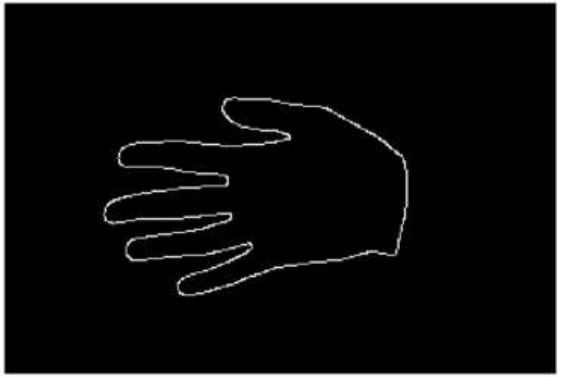


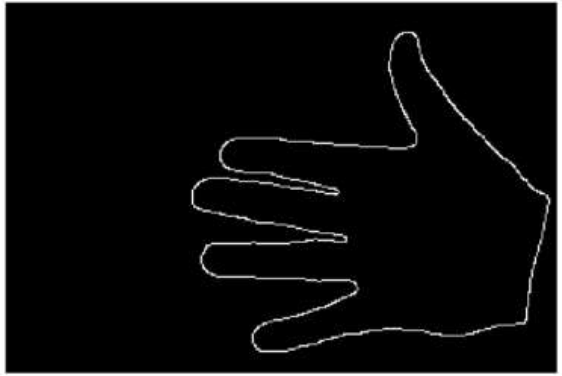
star1



star3

combined





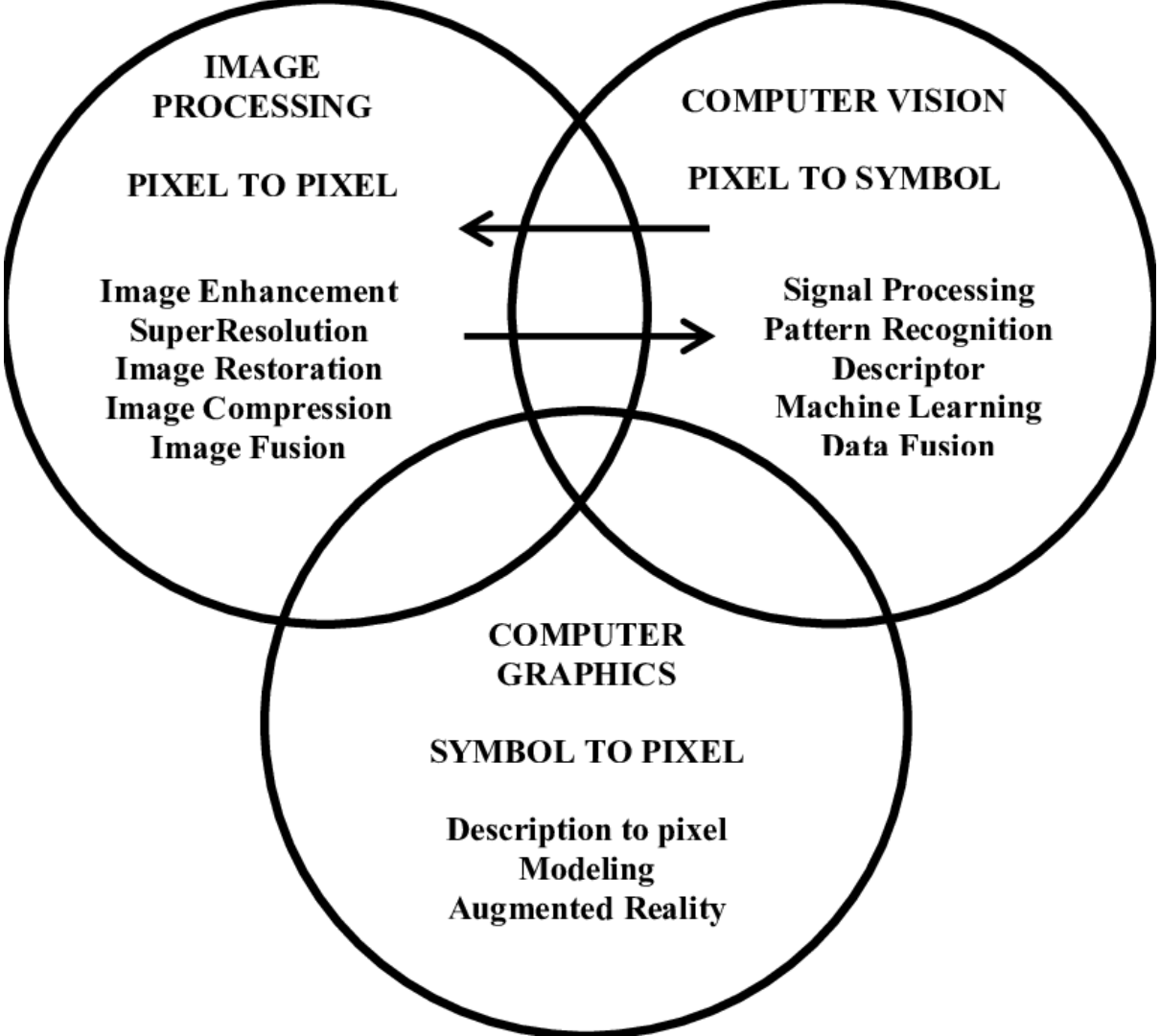
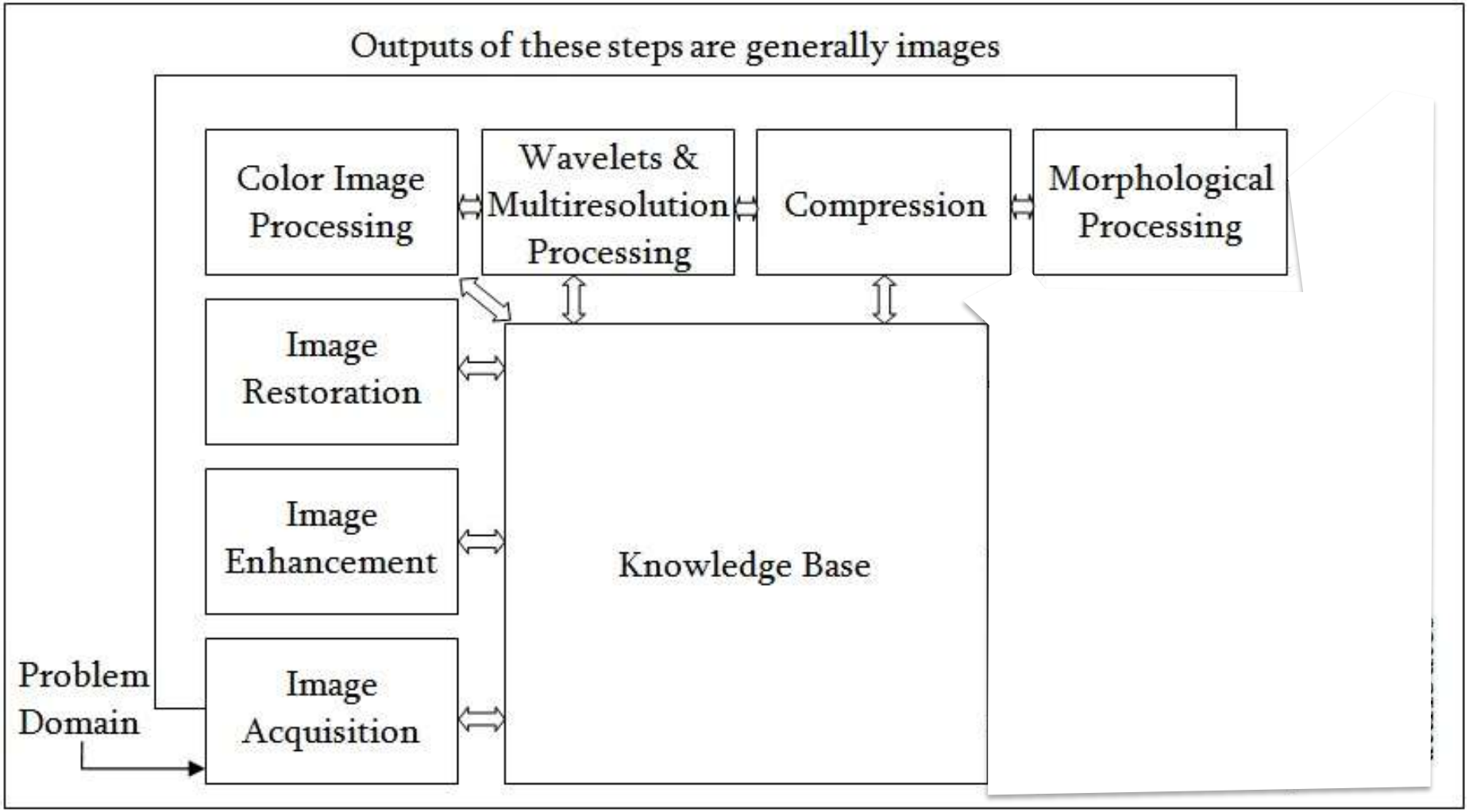
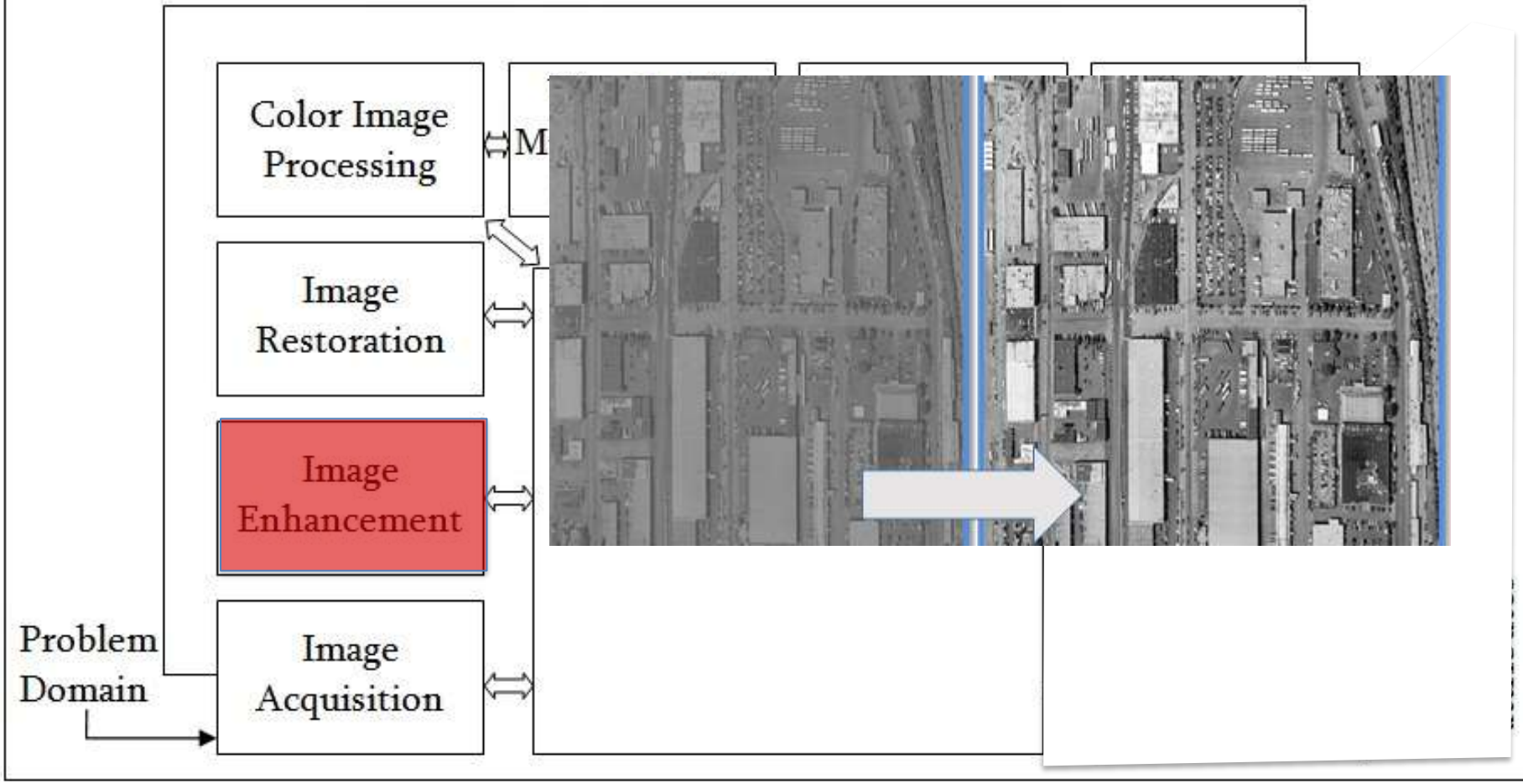
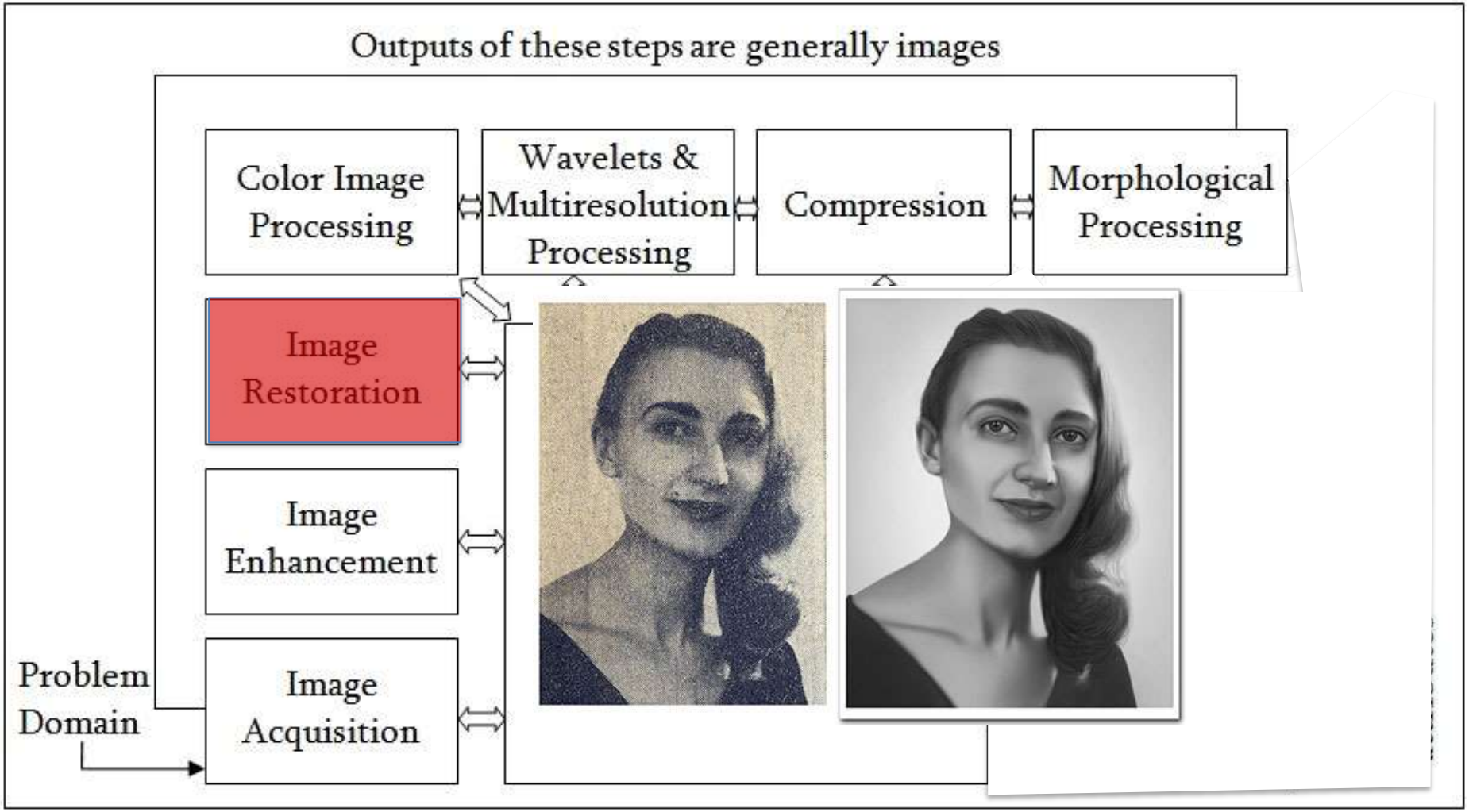


IMAGE PROCESSING
PIXEL TO PIXEL
Image Enhancement
SuperResolution
Image Restoration
Image Compression
Image Fusion
COMPUTER VISION
PIXEL TO SYMBOL
Signal Processing
Pattern Recognition
Descriptor
Machine Learning
Data Fusion
COMPUTER GRAPHICS
SYMBOL TO PIXEL
Description to pixel
Modeling
Augmented Reality


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

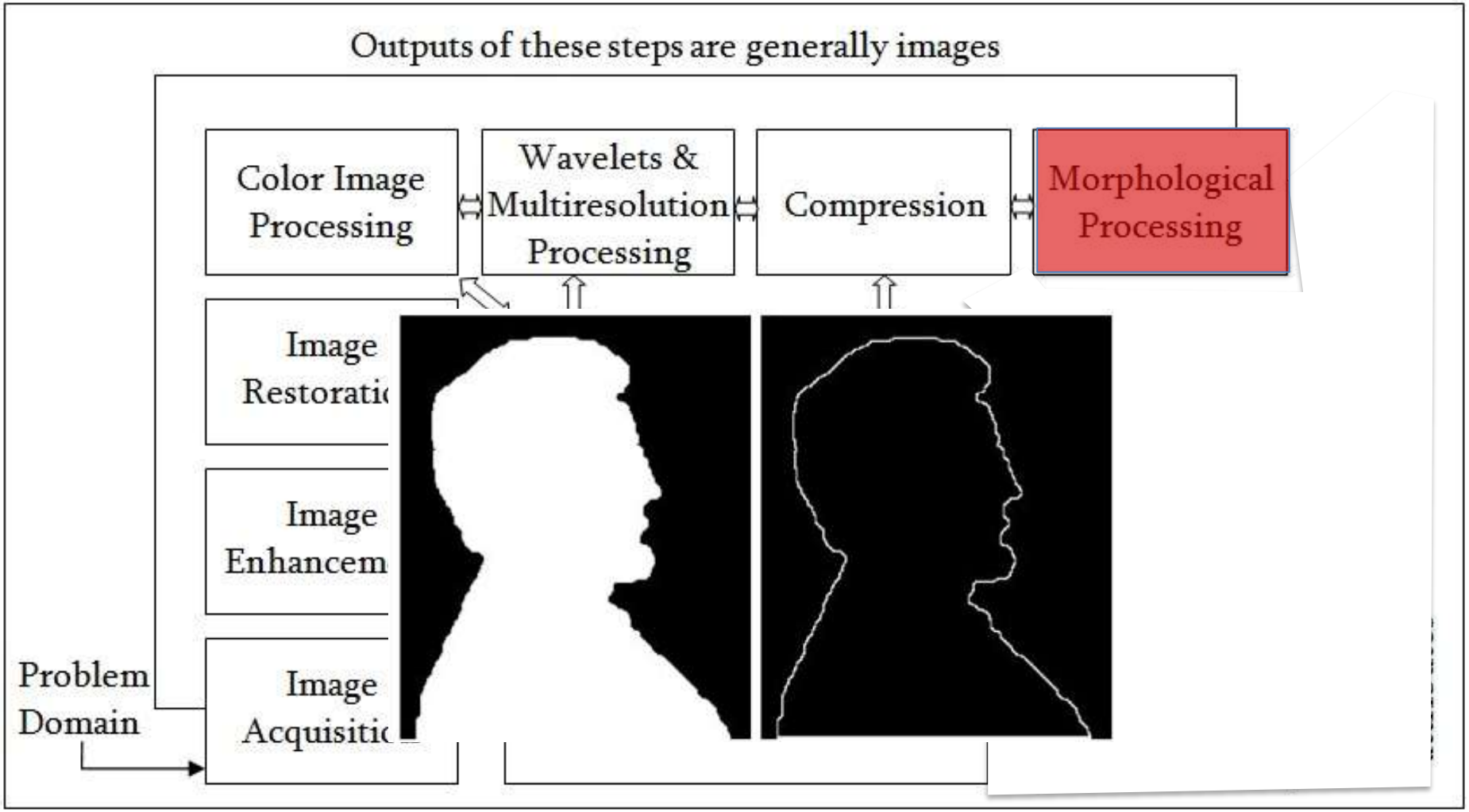
Image Acquisition

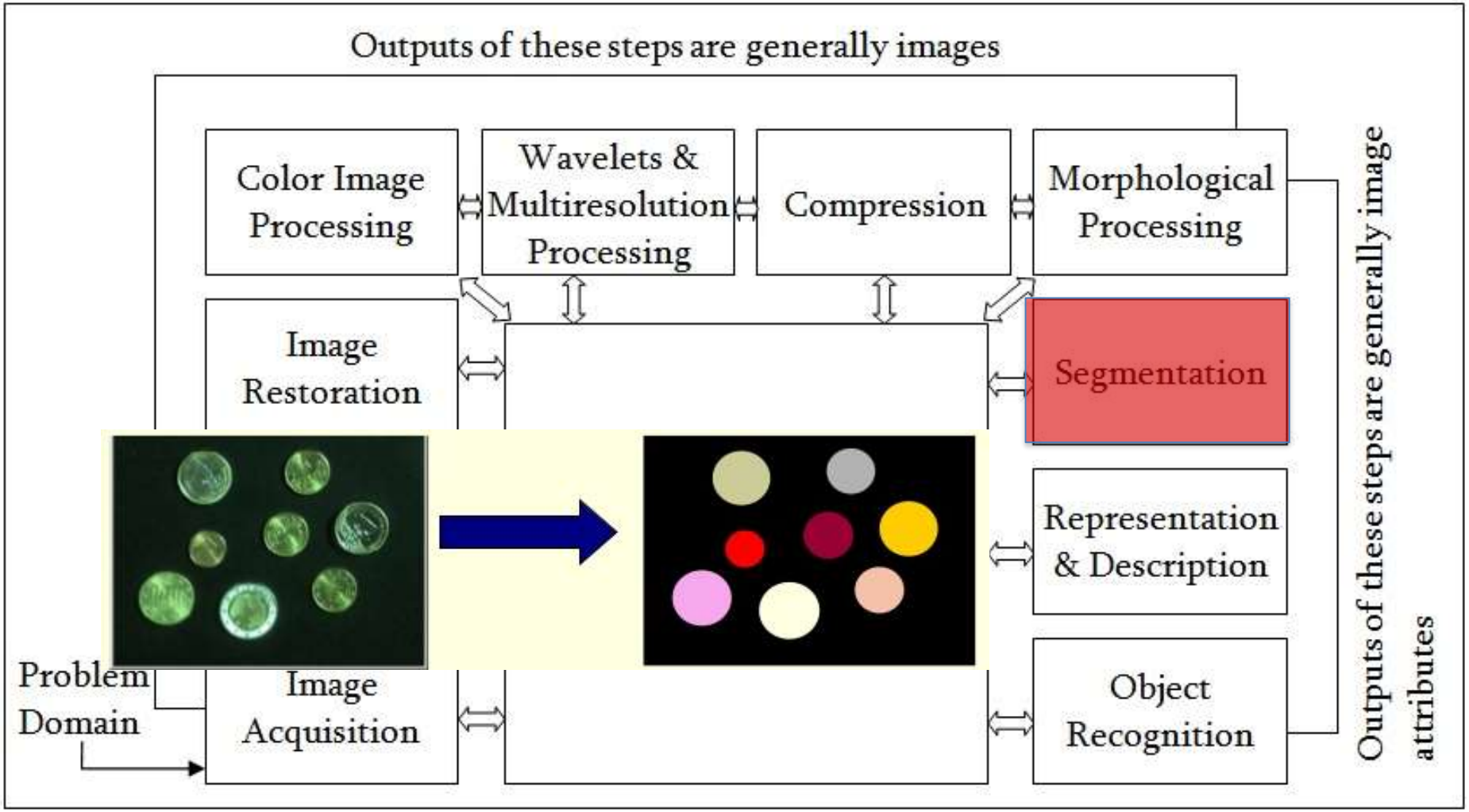
Problem Domain

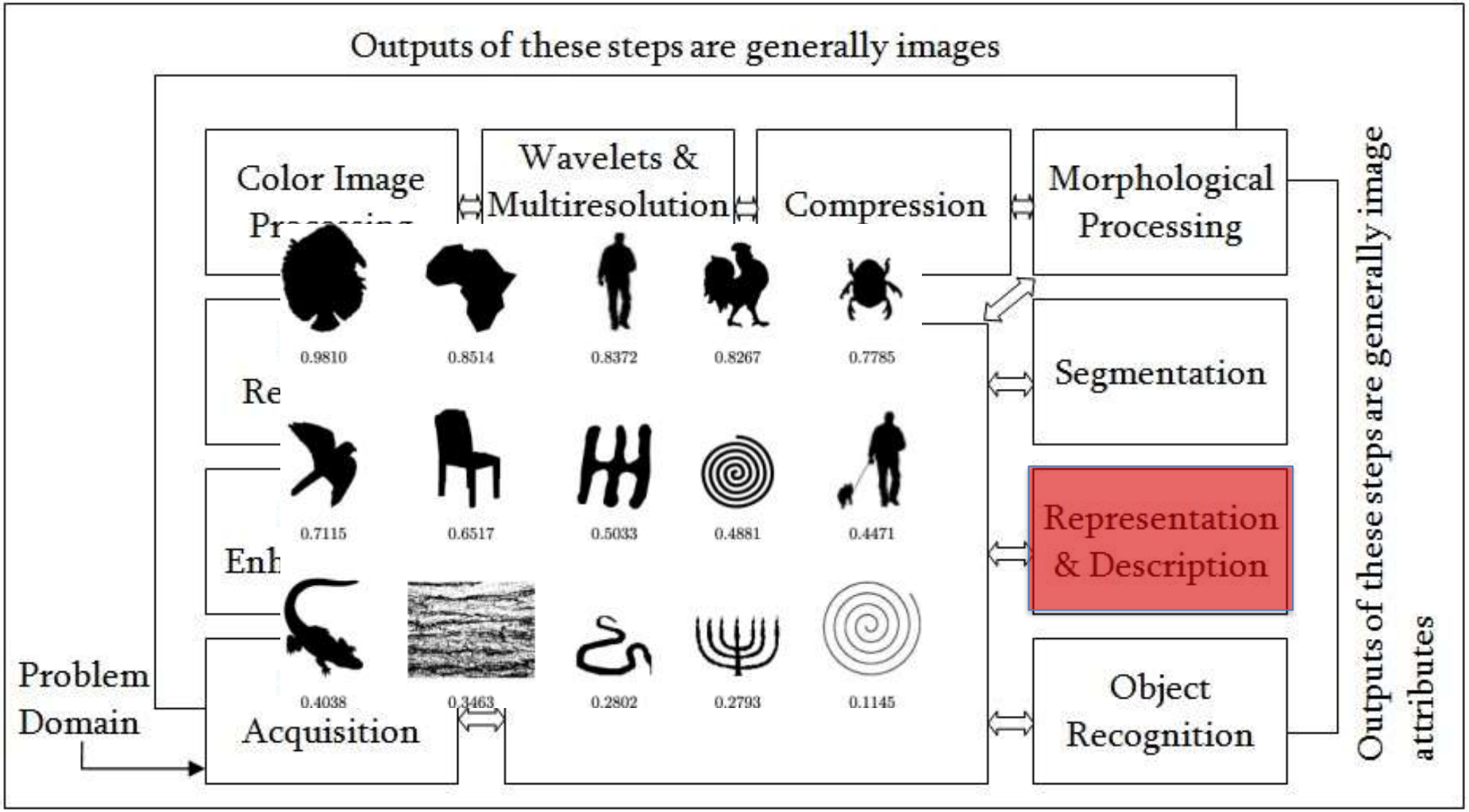
Size-270 KB

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

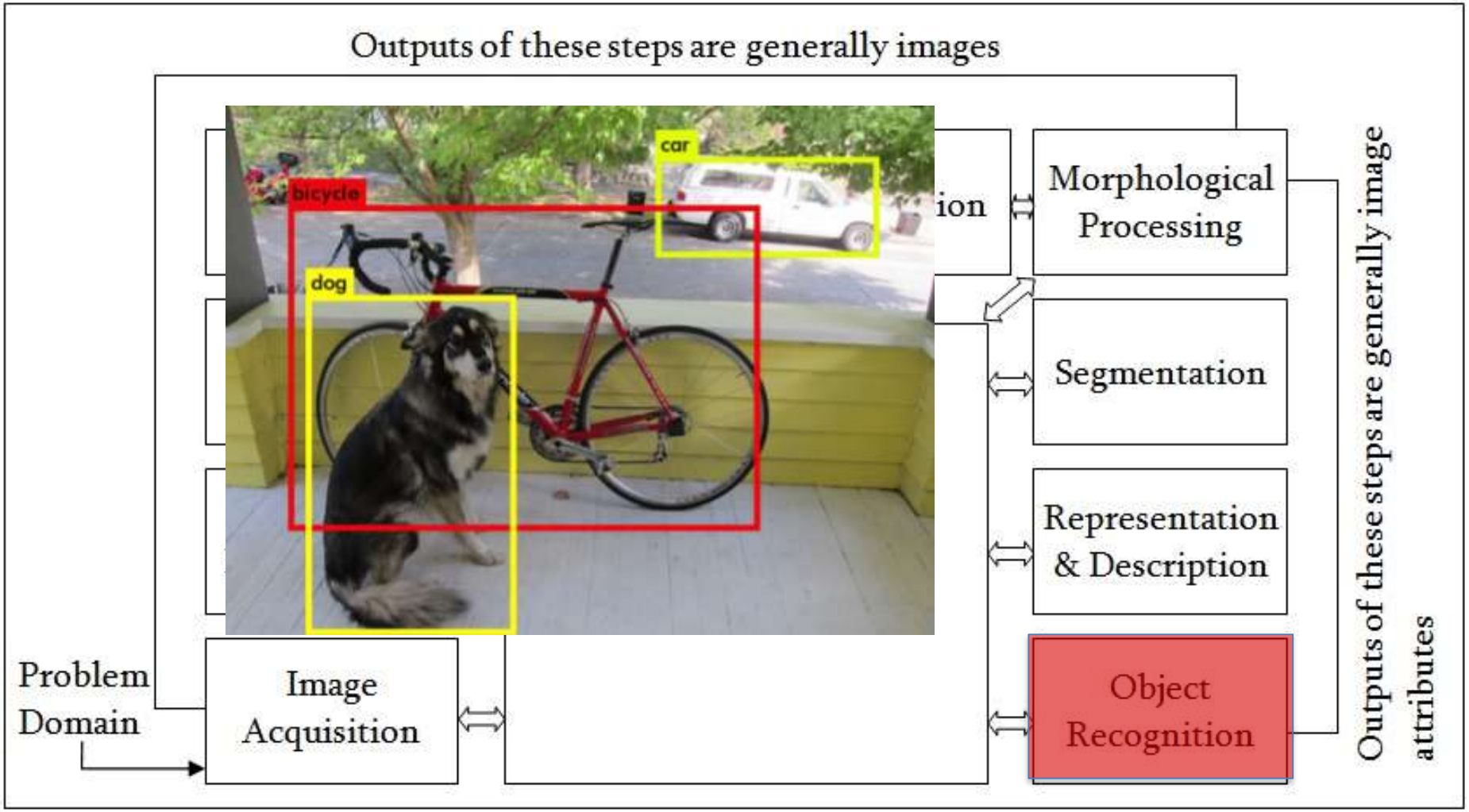
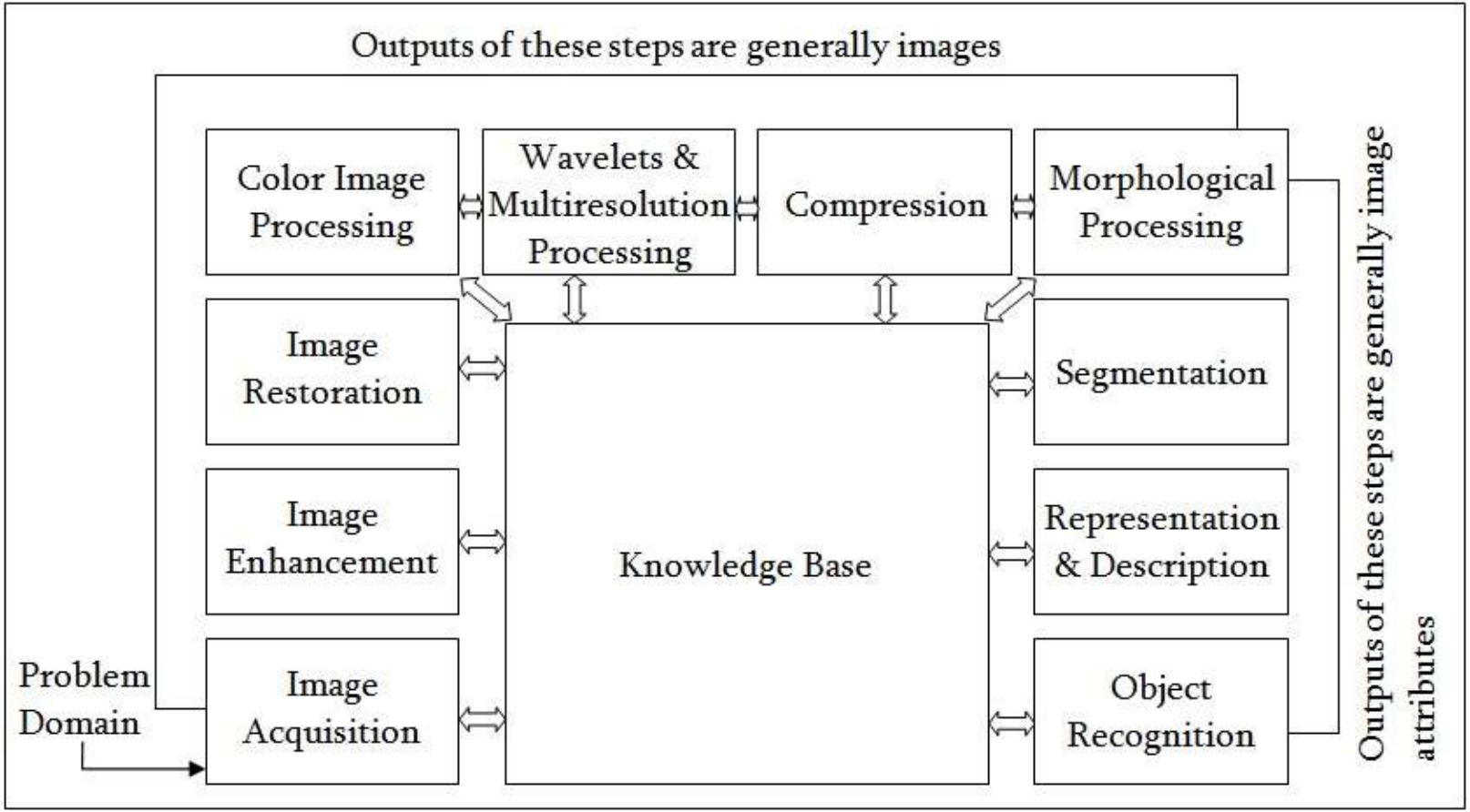

Outputs of these steps are generally images
car
bicycle
dog
ion
Morphological Processing
Segmentation
Representation & Description
Object Recognition
Outputs of these steps are generally image attributes
Problem Domain
Image Acquisition

Image Processing Tasks



Tasks of interest: Contrast adjustment

Image courtesy: mathworks

Tasks of interest: Edge Detection

Image courtesy: mathworks

Tasks of interest: Feature detection + stitching

Image courtesy: opencv

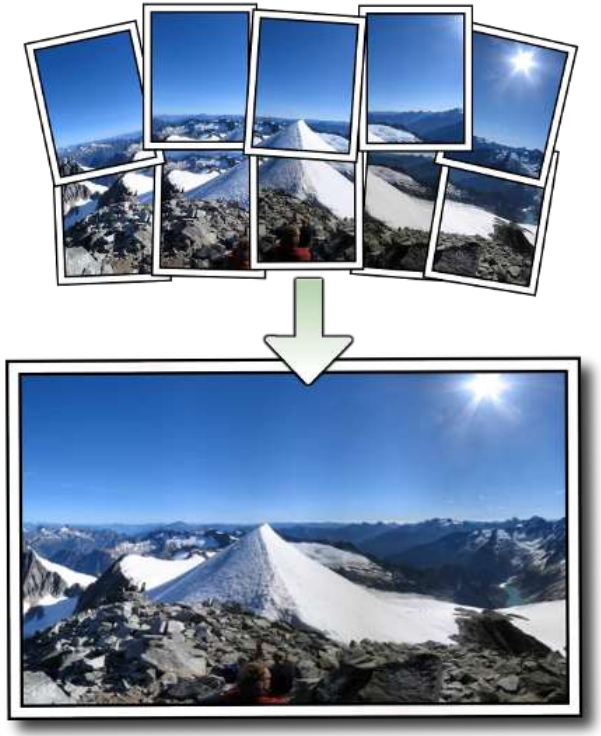


Image courtesy: autostitch

Tasks of interest: Noise Removal

Total variation denoising [Chambolle JMIV 2004]

Tasks of interest: Haze Removal

Single Image Haze Removal [He et al. CVPR 2009]

Retouch Personal Photos!

©Images taken from the web.

Tasks of interest: Artistic enhancement

Image courtesy: webneel.com

Image courtesy: Jon Morse

BW to Color

Mayabazar (1957), Vijaya Vauhini Studios

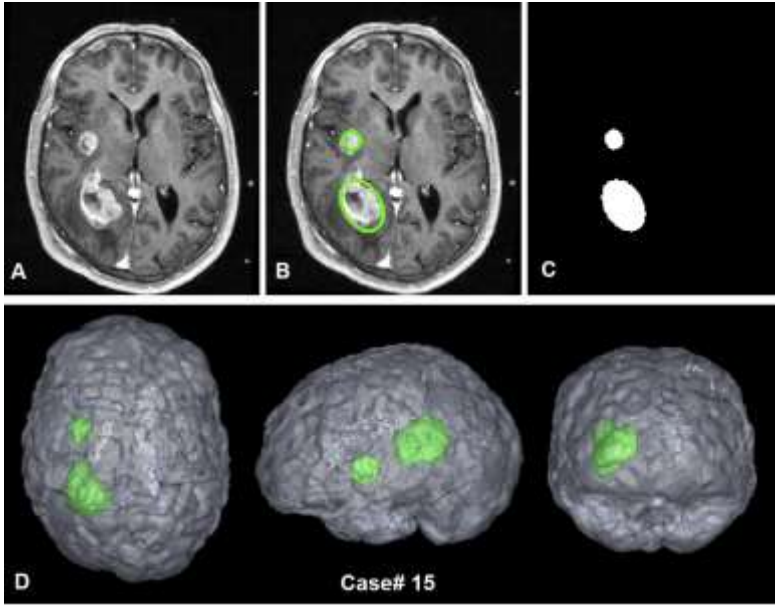
Colorized (2010)

Tasks of interest: Cinematic Grading

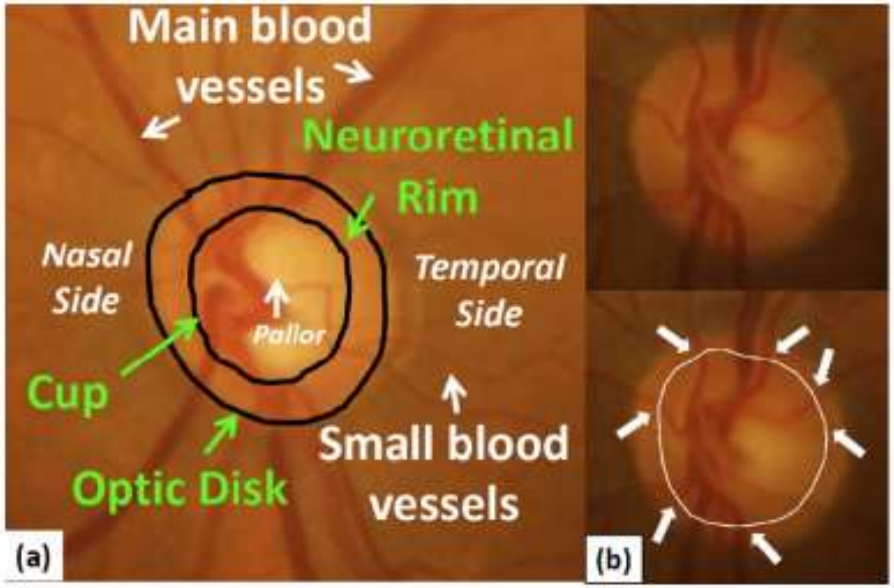
Mission Impossible - Ghost Protocol, Paramount Pictures

Image courtesy: juanmelara.com

Tasks of interest: Segmentation

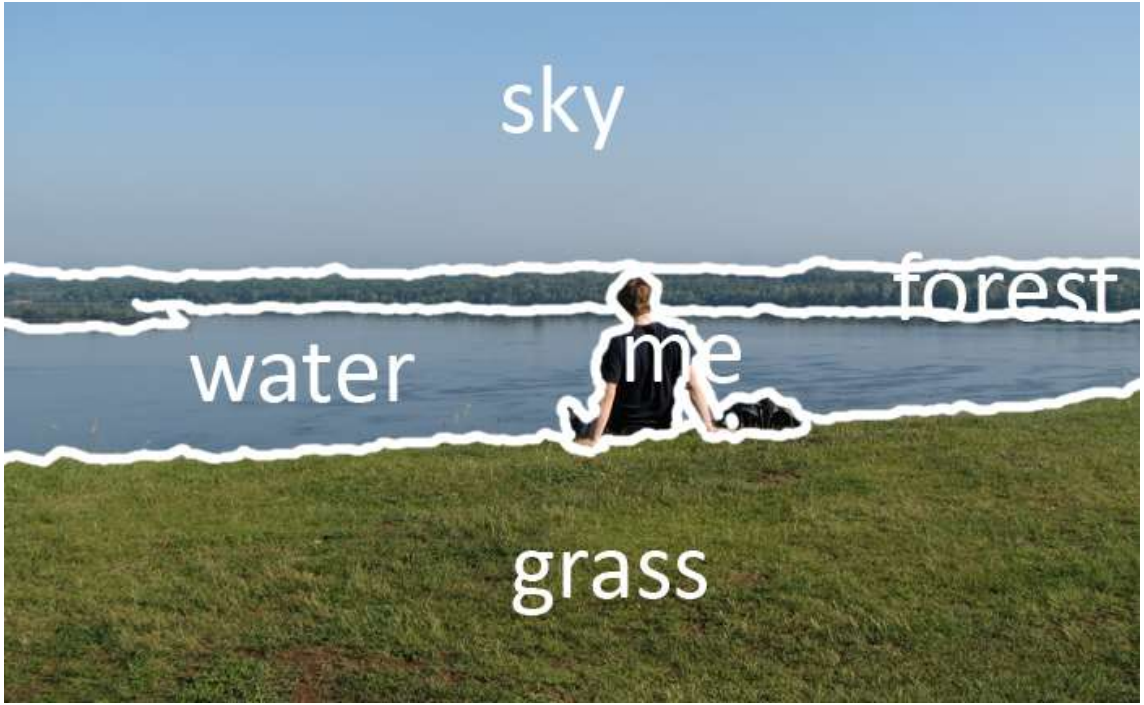


Tumour Segmentation [Yu et al. MICCAI 2010]



Cup Segmentation [Joshi and Sivaswamy 2011]

Tasks of interest: Segmentation



Courtesy: Roman Shapovalov

Tasks of interest: Compression

Original Image (1.2 mb)

Compressed JPEG Image (100 kb)

Tasks of interest: Inpainting

DAMAGED

RESTORED

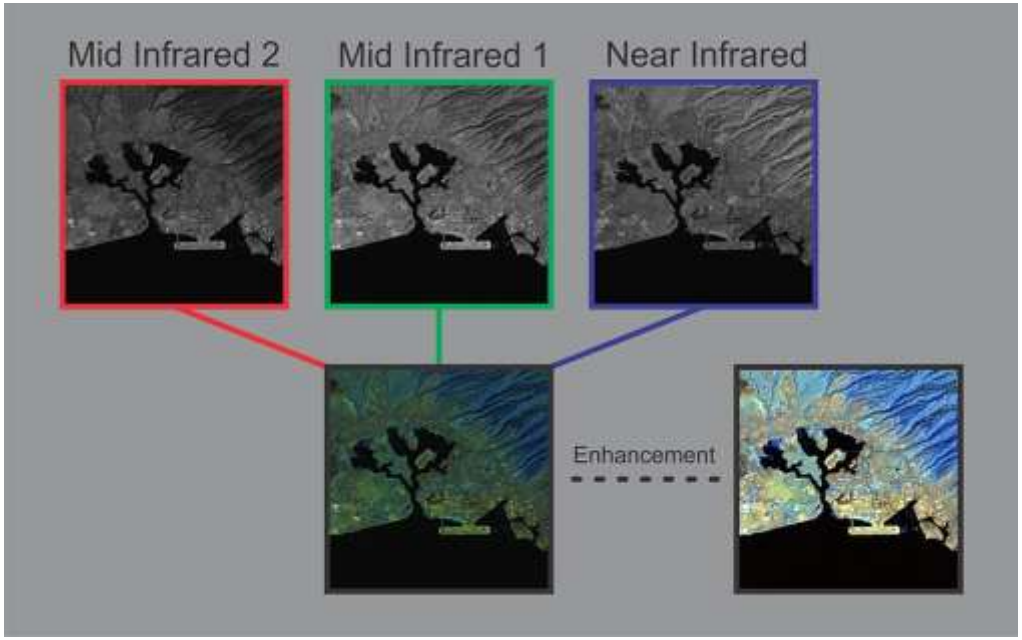
Bertalmio et al. SIGGRAPH 2010

Tasks of interest: Special effects

courtesy: wachowski brothers (matrix)

courtesy: Miller et al. (sin city)

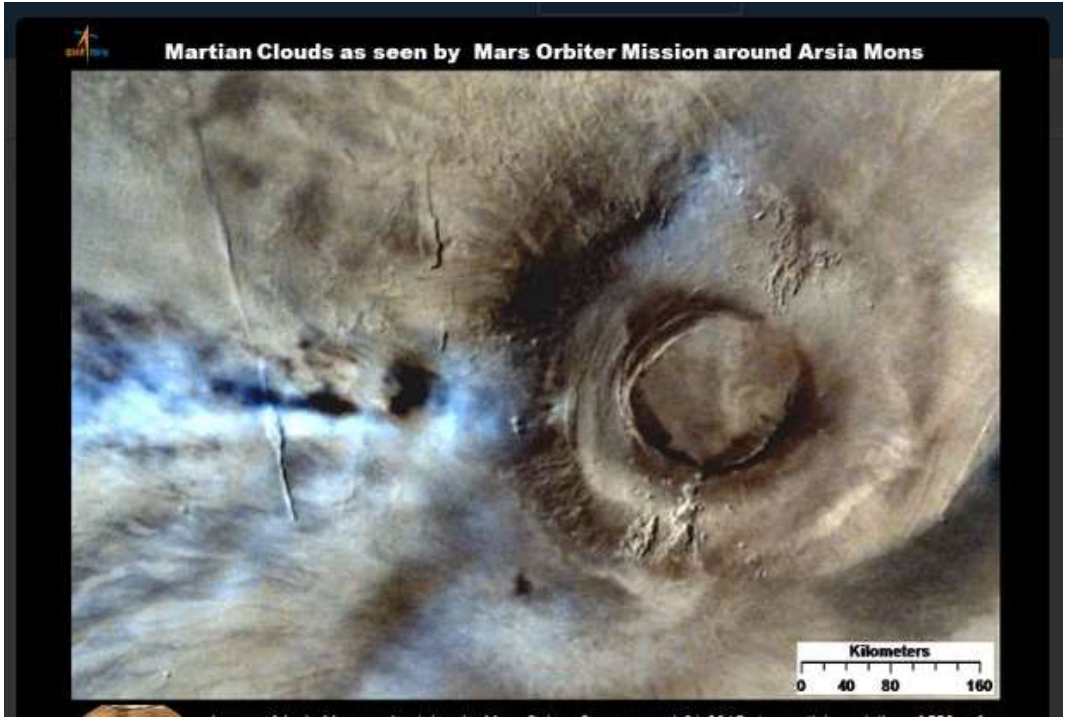
Tasks of interest: Satellite imaging

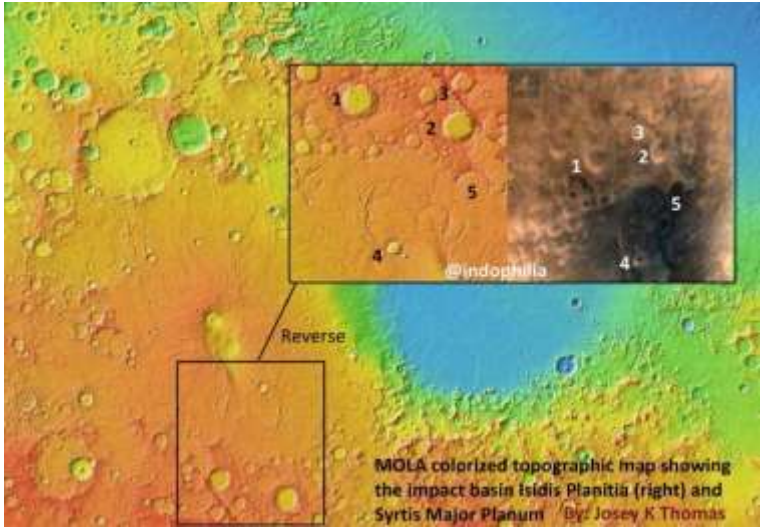


Terrain classification, weather predictions etc.

courtesy: NASA

Tasks of interest: Astronomy

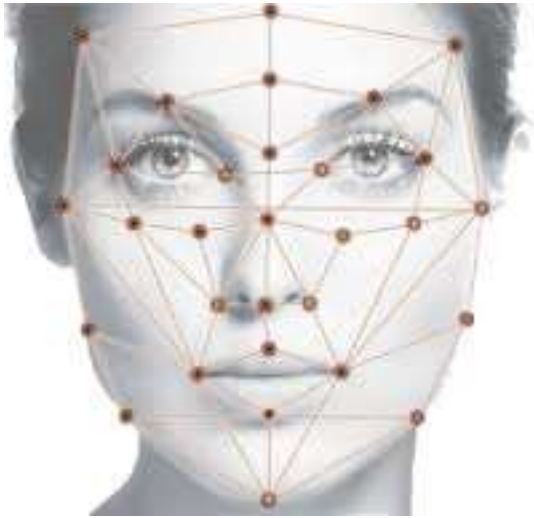




courtesy: ISRO

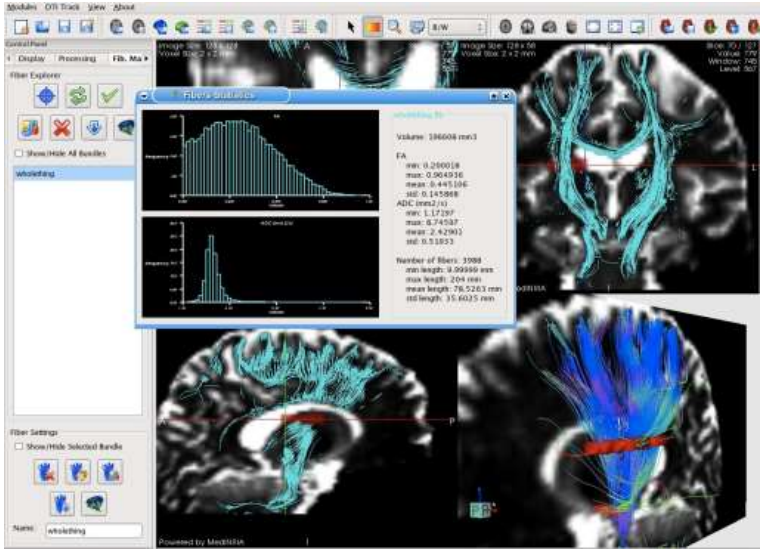
Tasks of interest: Biometrics

courtesy: dqindia.com



courtesy: heyce.com

Tasks of interest: Medicine

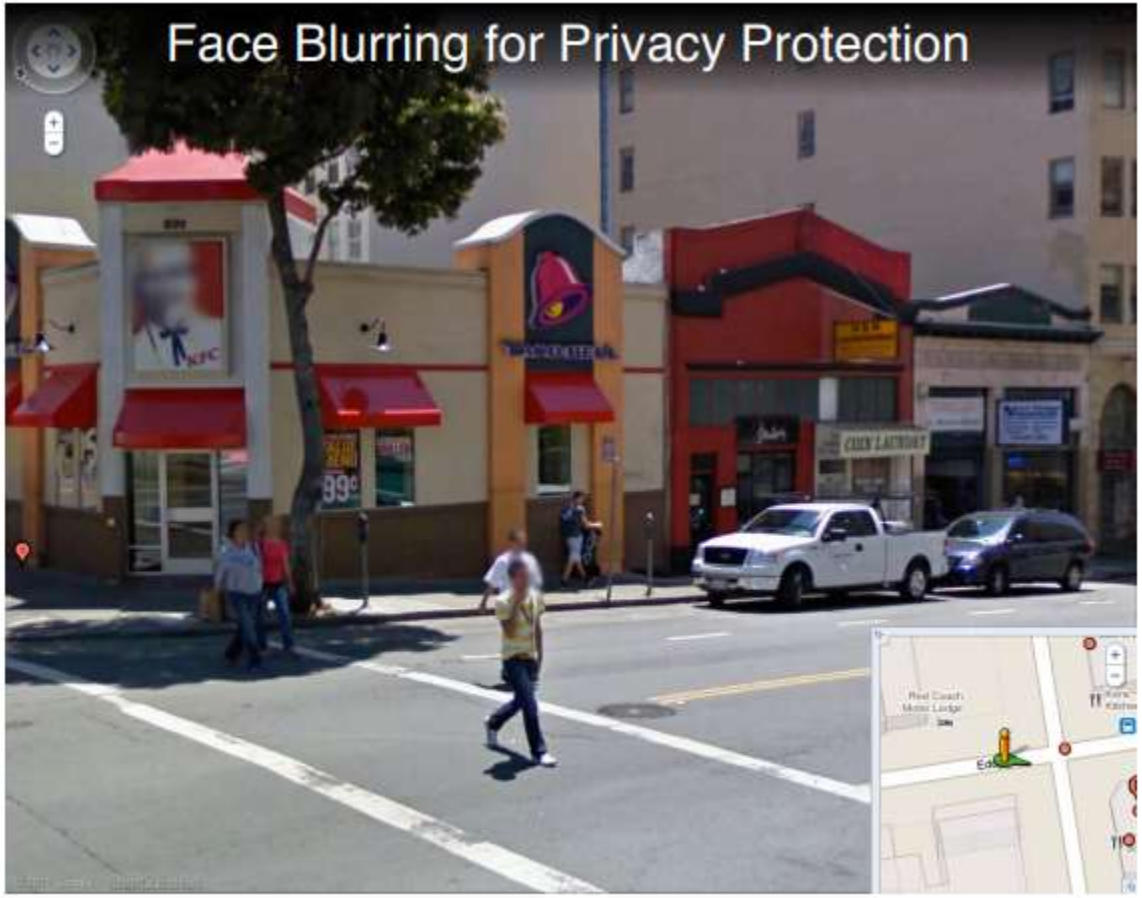


Courtesy: medINRIA

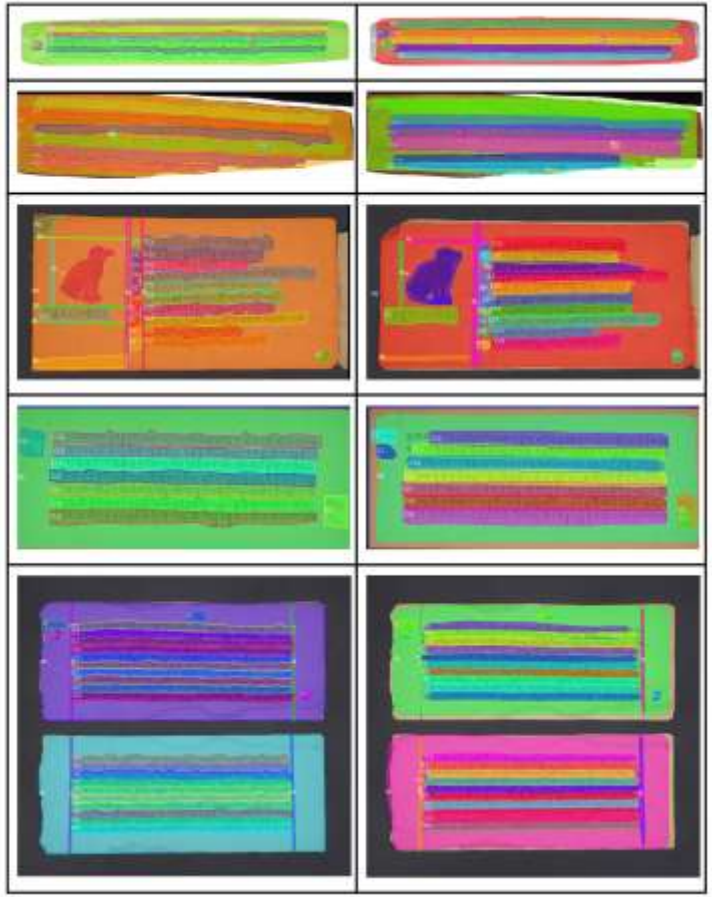
Courtesy: Naoki Suzuki

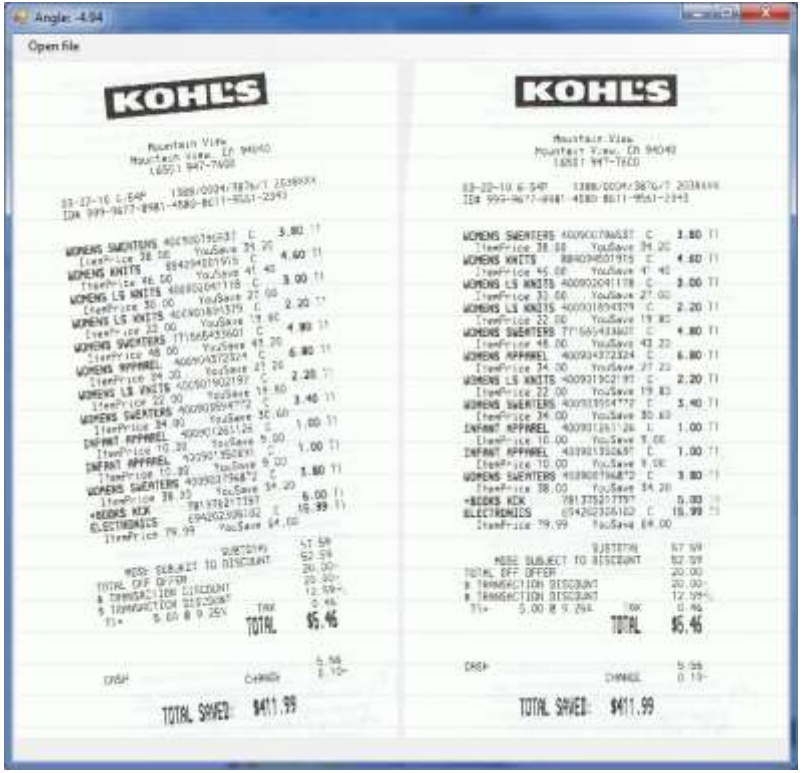
Tasks of interest: Driverless Vehicle Systems

Face Blurring for Privacy Protection



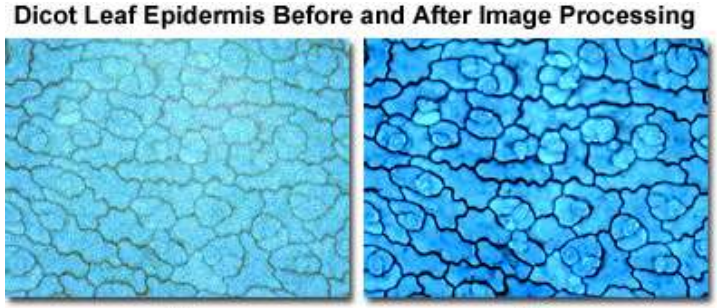
Tasks of interest: Document Image Analysis





Tasks of interest: Many more

- Biology
- HCI
- Number Plate recognition
- Gesture recognition



Courtesy: Olympus

Courtesy: Perviverzov et al. 2012

Courtesy: researchdesignlab.com

Resources

- Image Processing Libraries
  - scikit-image
  - **opencv-python**
  - Kornia
  - PIL
- P.S. Libraries are written by humans (mostly!) – can contain bugs!

The dangers behind image resizing

The expectation that the behaviour for image resizing is the same among the libraries can cause unforeseen problems...

https://zuru.tech/blog/the-dangers-behind-image-resizing