https://tinyurl.com/dhcp7h3w

Computer Vision 1

Neighborhood Processing
Image Filtering
Fourier Transform

Dr. Martin Oswald, Dr. Dimitris Tzionas, Dr. Arun Mukundan,
(m.r.oswald, d.tzionas, a.mukundan)@uva.nl

University of Amsterdam

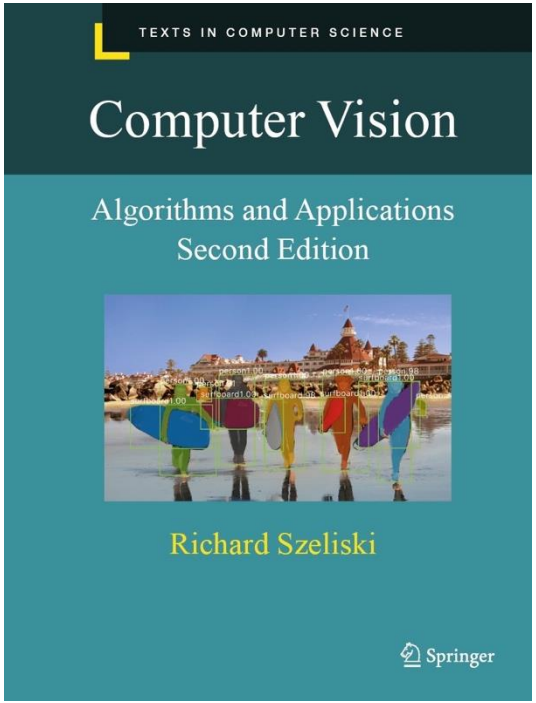
computer vision lab

Outline

- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Textbook

- 3.2 Linear filtering
- 3.3 More neighborhoos operators
- 3.4 Fourier transforms



[https://szeliski.org/Book/]

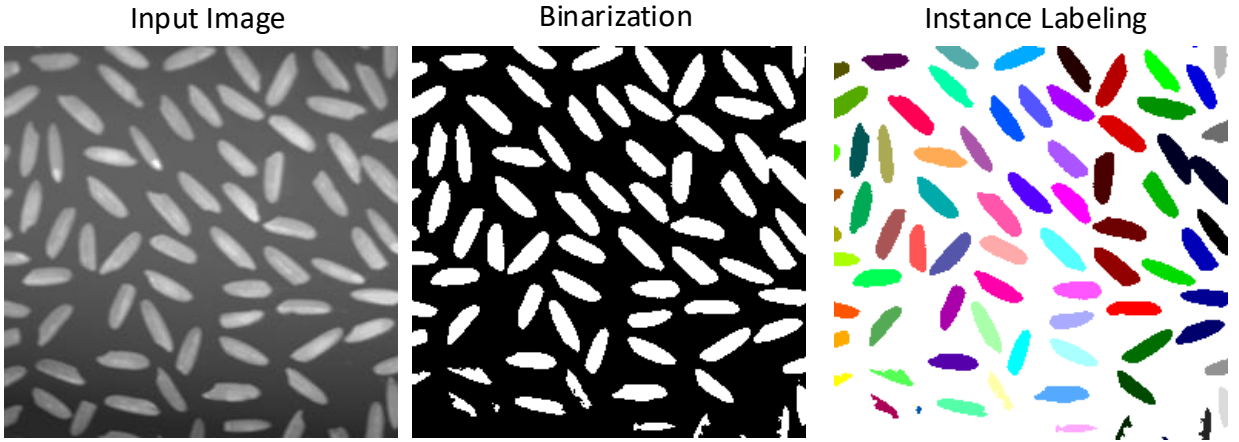
Outline

- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Simple Image Processing Examples

Instance Segmentation and Counting

- Rice grains



- Organs

Thresholding

Binarize input signal:

Input Image

Binarized Image $B(i,j) = \begin{cases} 255 & \text{if } I(i,j) > \tau \\ 0 & \text{otherwise} \end{cases}$ Threshold

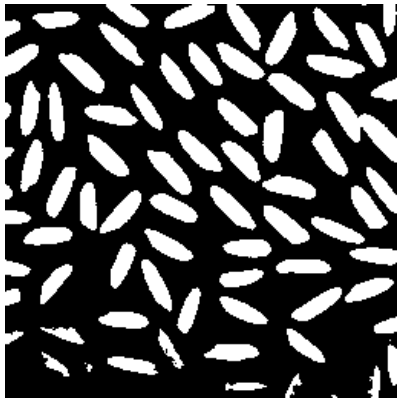
Why binary ?

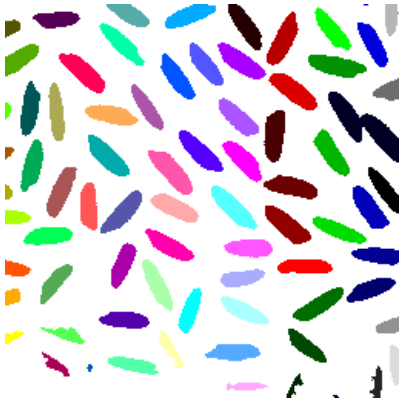
Outline

- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Connected Components Analysis

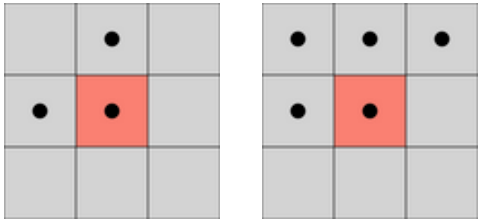
- Connected Components Analysis is a multiple-class labeling operation.
- It performs the change from pixels to regions.
- All pixels in the same region are given the same label.



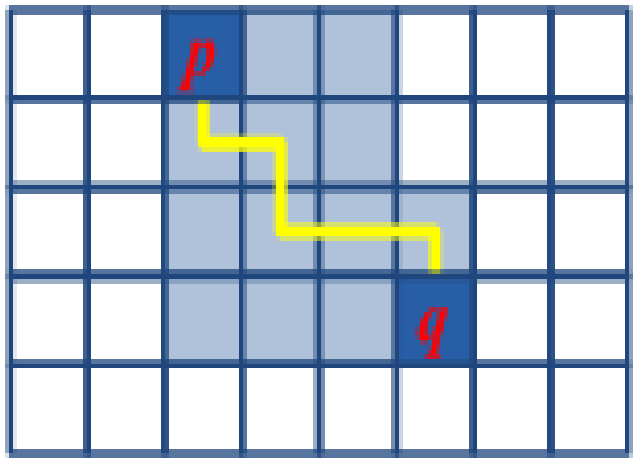


Connected Component Operators

- Definition of Connected Component
  - Two pixels p and q belong to the same connected component C if there is a sequence of 1-pixels $(p_0, p_1, \ldots p_n)$, where
    - $p_0 = p$
    - $p_n = q$
    - $p_{i-1}, p_i : i = 1, \ldots, n$ are neighbors



4-connectivity 8-connectivity



Region property computation

For each connected region, we can compute many of its properties, e.g.

- Areas
- Perimeter
- Centre of gravity
- circularity,
- Major axis
- Minor axis
- mean and standard deviation of radial distance
- bounding box
- extremal axis length from bounding box
- second order moments (row, column, mixed)
- …

Region property computation

First Pass:

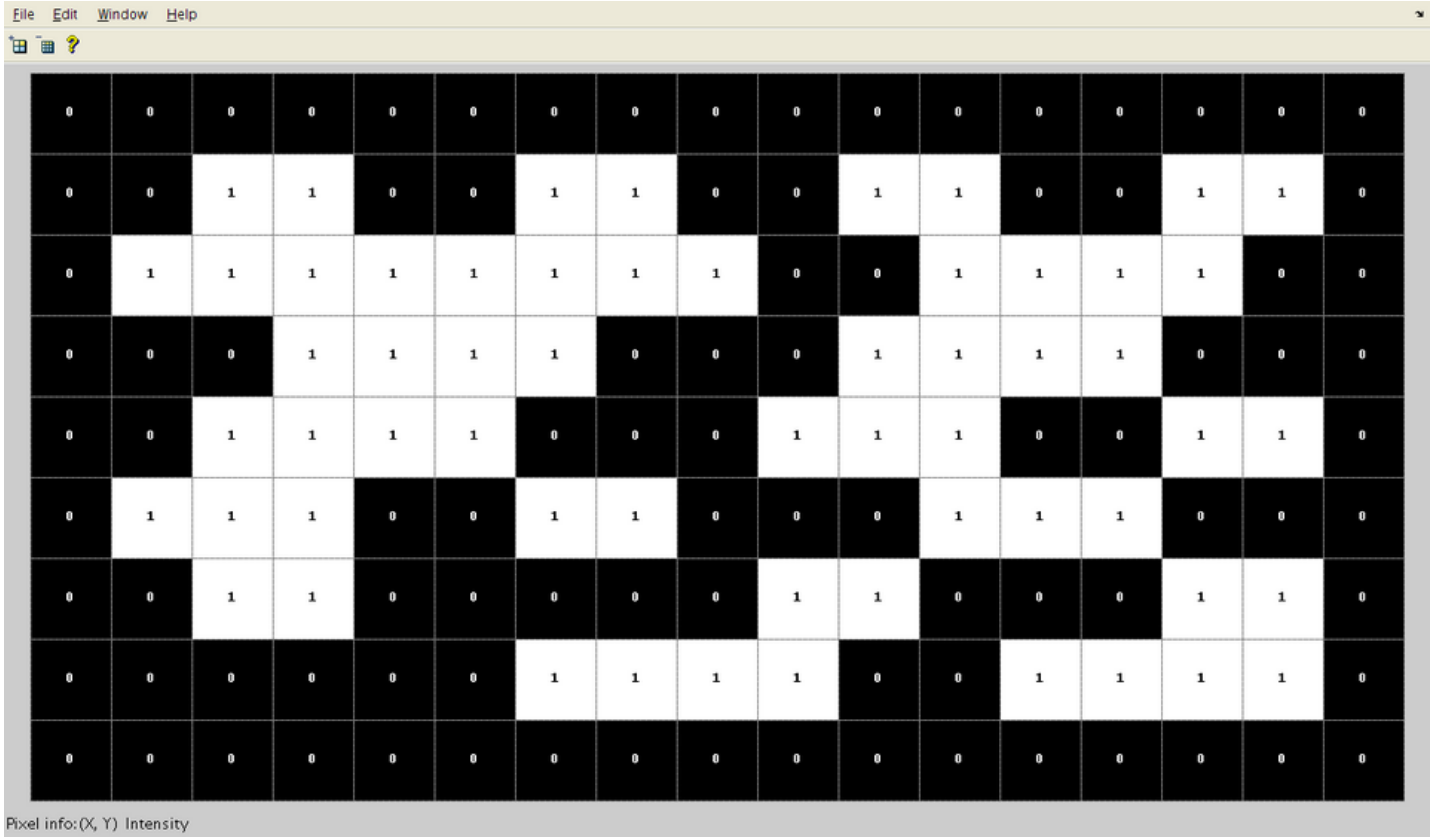
1. Iterate through each element of the data by column, then by row (Raster Scanning)
2. If the element is not the background
    1. Get the neighboring elements of the current element
    2. If there are no neighbors, uniquely label the current element and continue
    3. Otherwise, find the neighbor with the smallest label and assign it to the current element
    4. Store the equivalence between neighboring labels

Second Pass:

1. Iterate through each element of the data by column, then by row
2. If the element is not the background
    1. Relabel the element with the lowest equivalent label

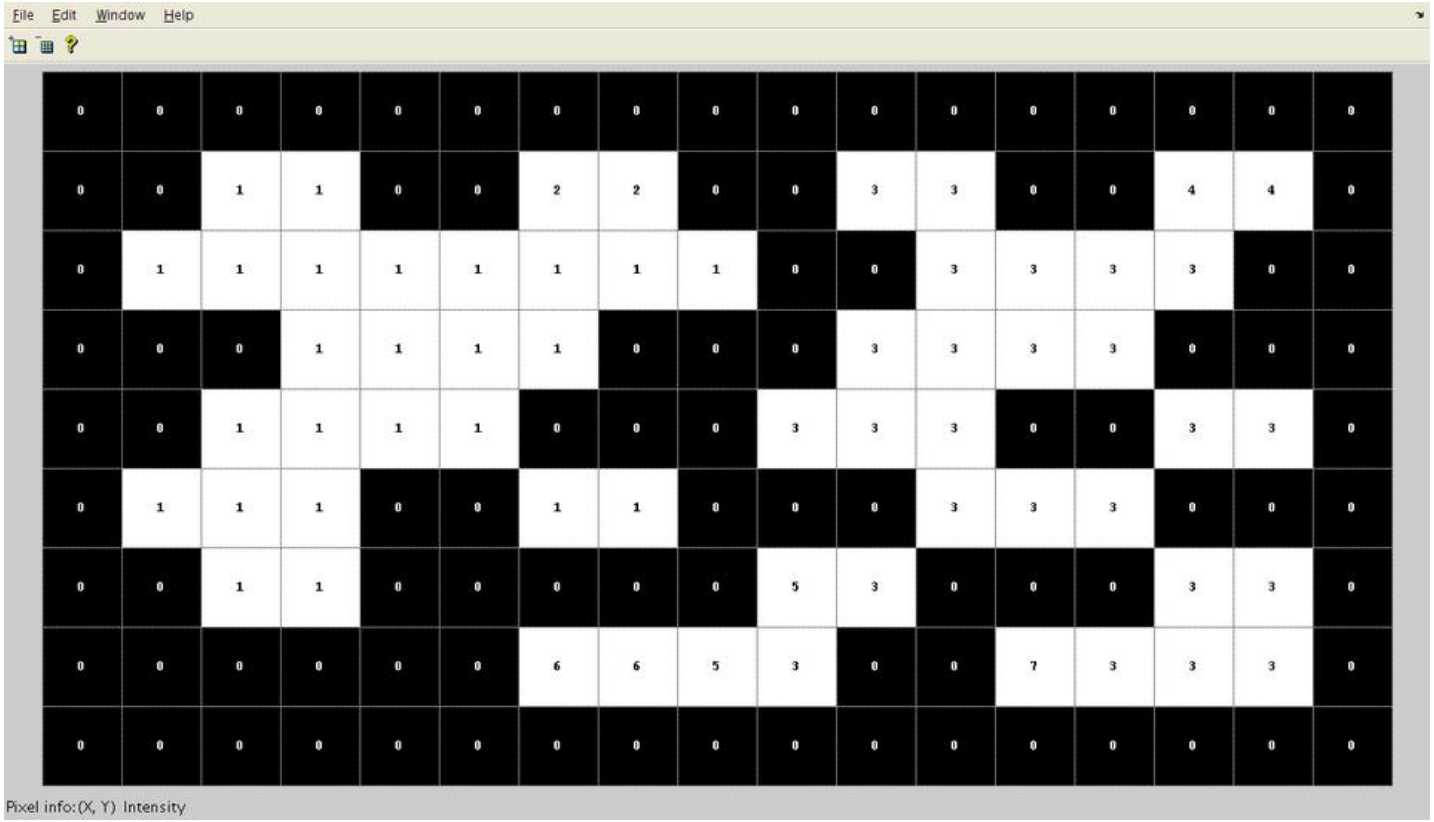
Region property computation

Input



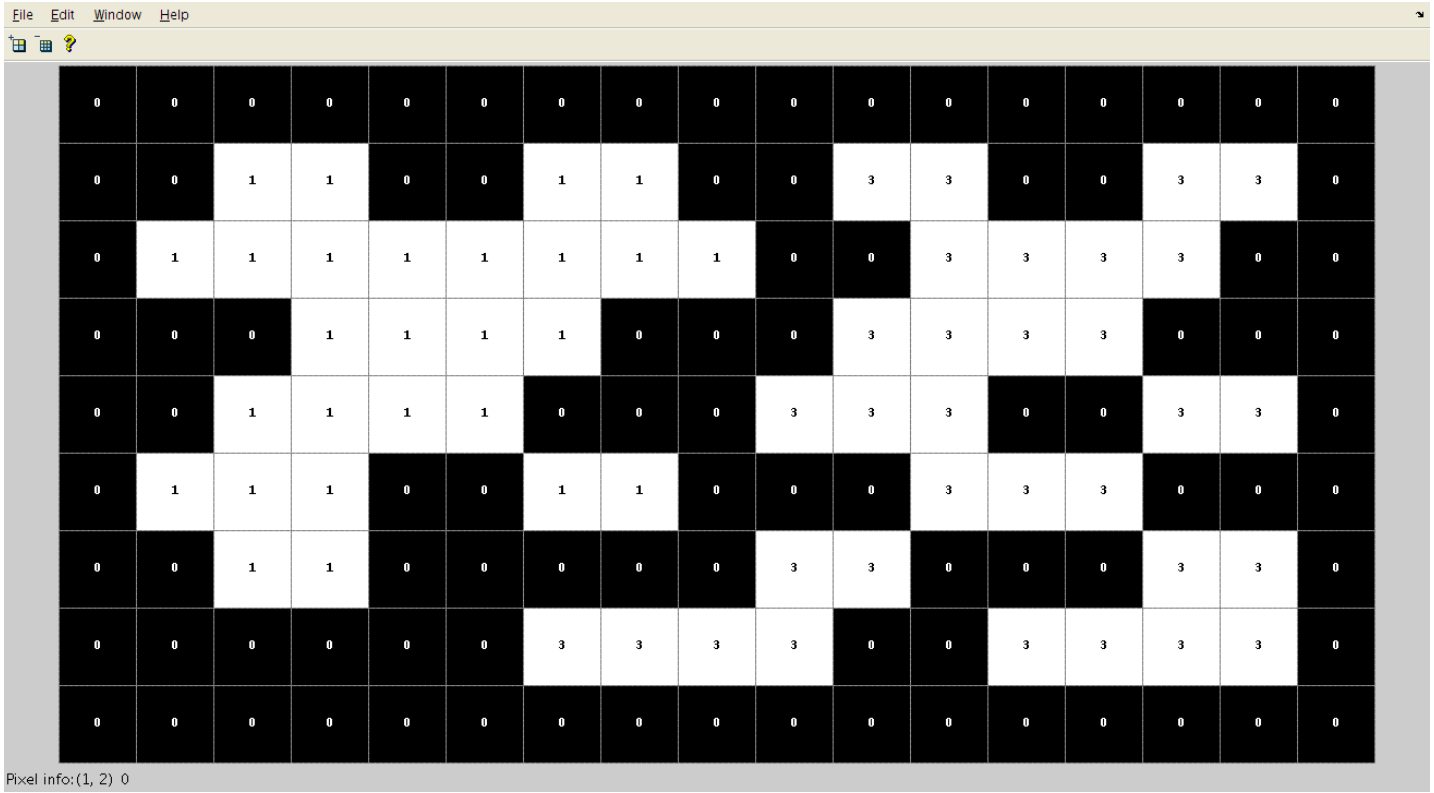
Region property computation

After First Pass



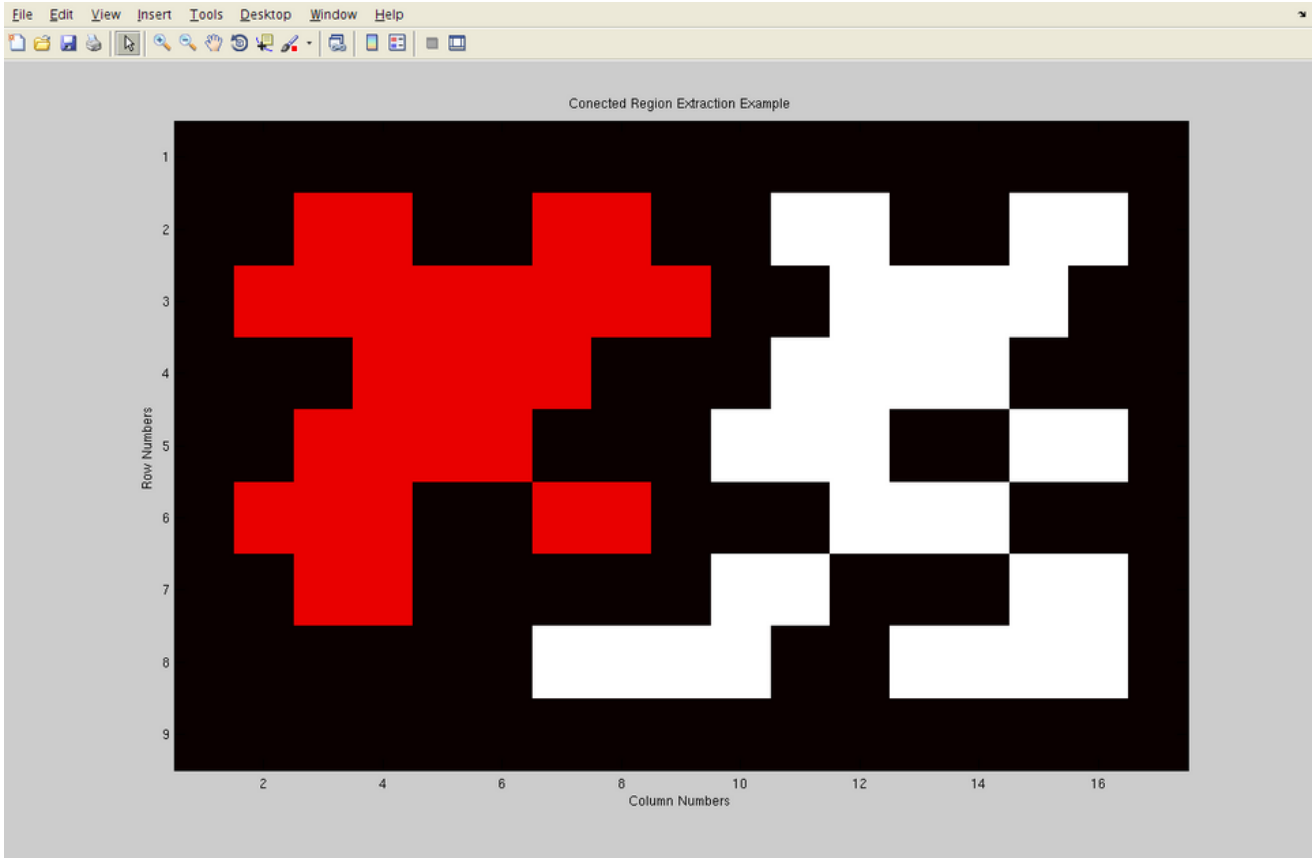
Region property computation

After Label Merging



Region property computation

Final Labeling



Outline

- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

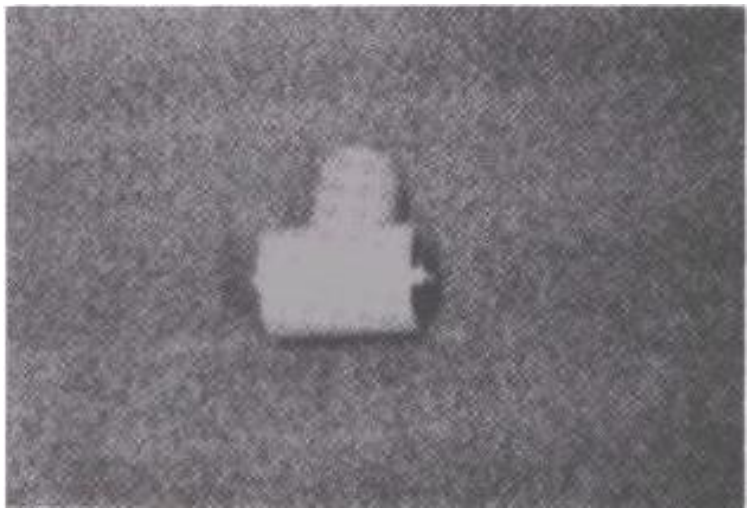
Mathematical Morphology

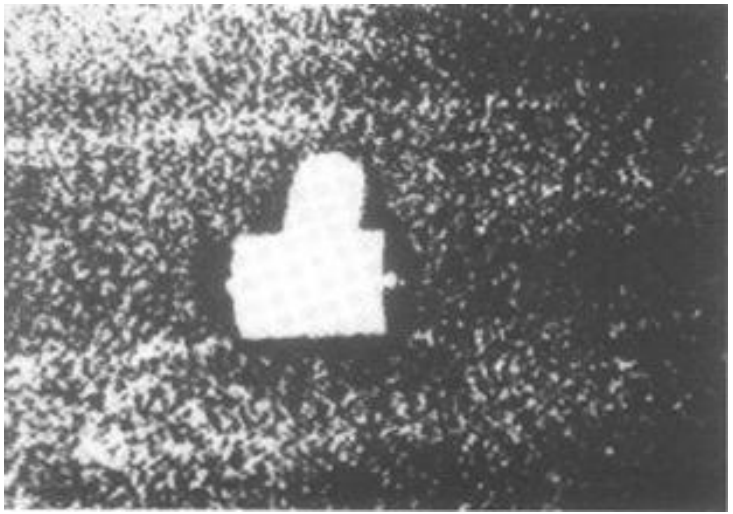
- **Morphology:** originally, a branch of biology science that studies the form and structure of animals and plants

- Mathematical Morphology in image processing is used to extract image components for representation and description of region shape, such as boundaries, skeletons, and the convex hull.

Motivation: remove small noise regions

- Example result by a binarization algorithm





Mathematical Morphology

In binary image processing, mathematical morphology consists of two basic operations,

dilation , erosion

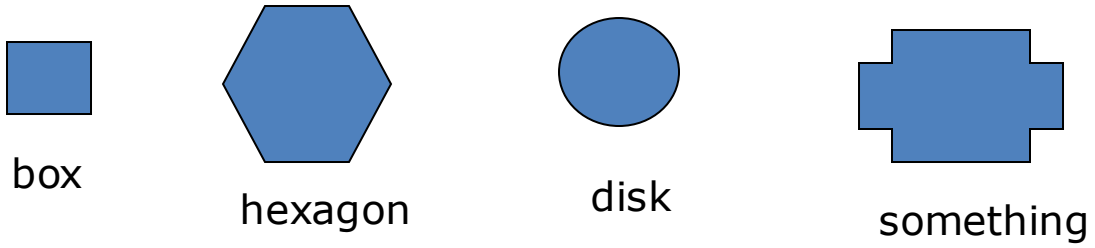
and several composite operations

closing , opening, . . .

Structuring Elements

A structuring element is a shape mask used in the basic morphological operations.

They can be any shape and size that is digitally representable, and each has an origin.



box(length, width)

disk(diameter)

Dilation

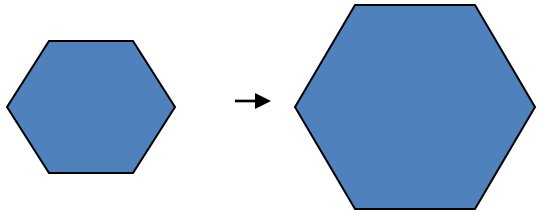
dilate(B,S) takes binary image B, places the origin of structuring element S over each 1-pixel, and **OR**s the structuring element S into the output image at the corresponding position.

Dilation

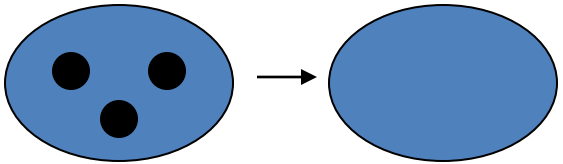
Dilation expands the connected sets of 1s of a binary image.

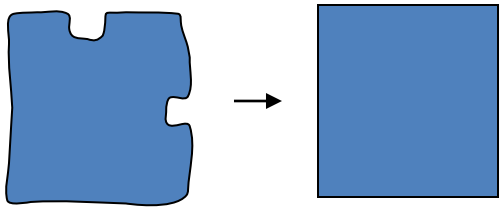
It can be used for

1. growing features

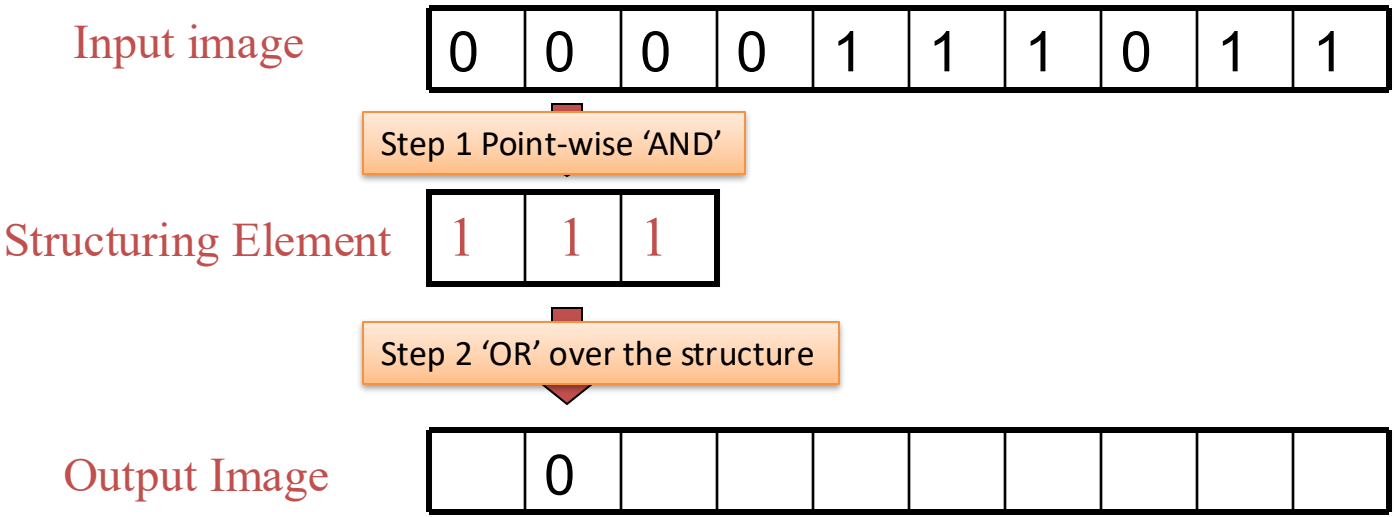


2. filling holes and gaps

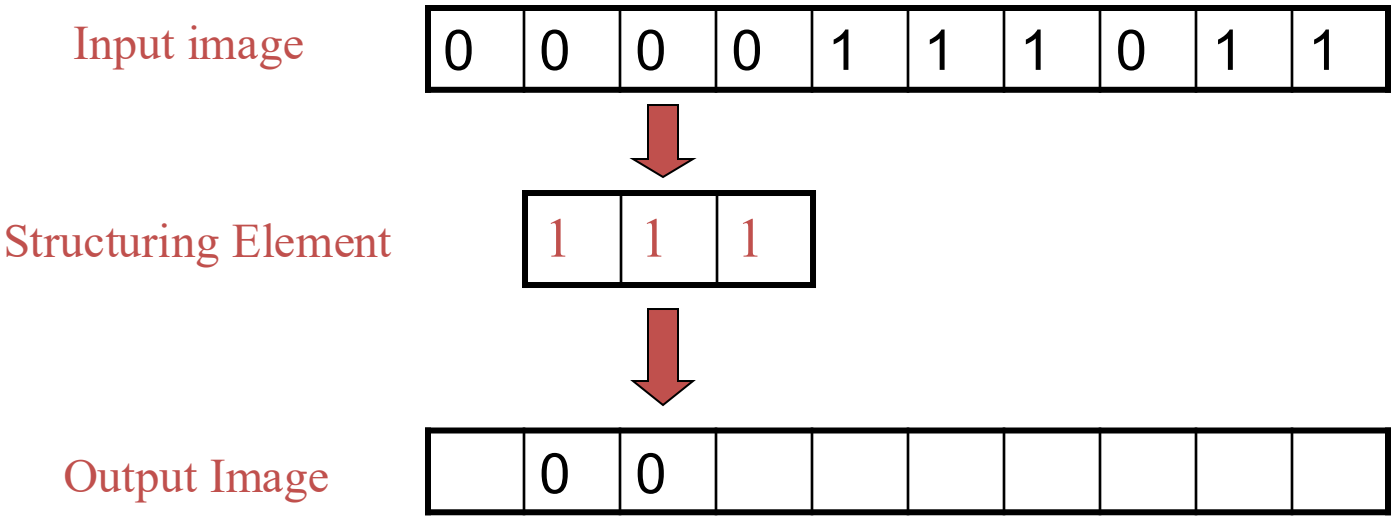




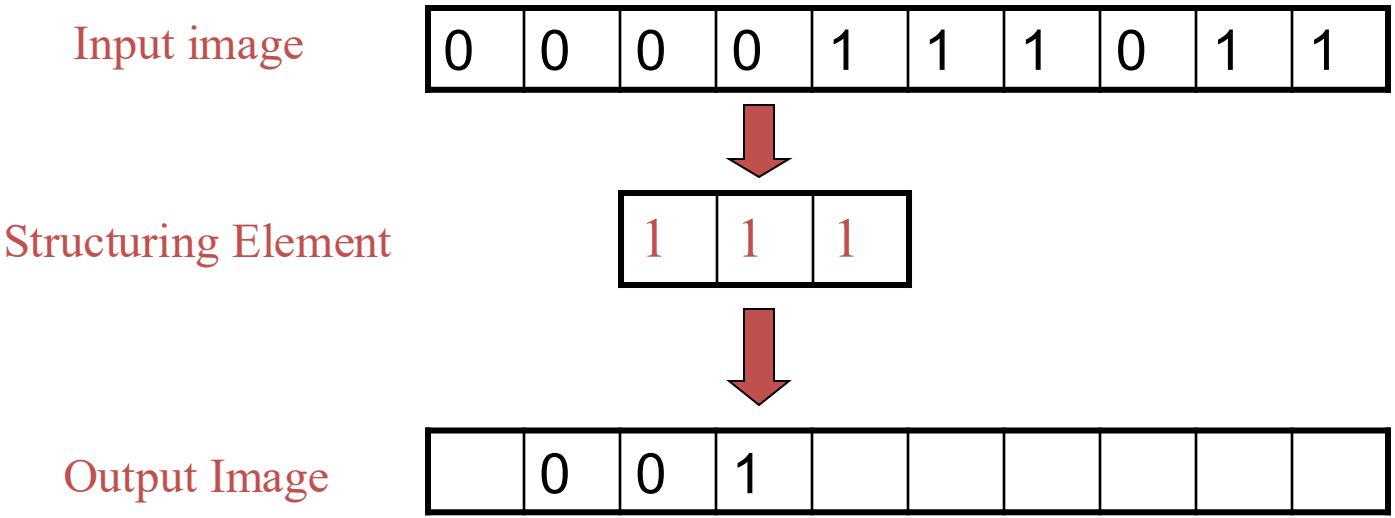
Example for Dilation



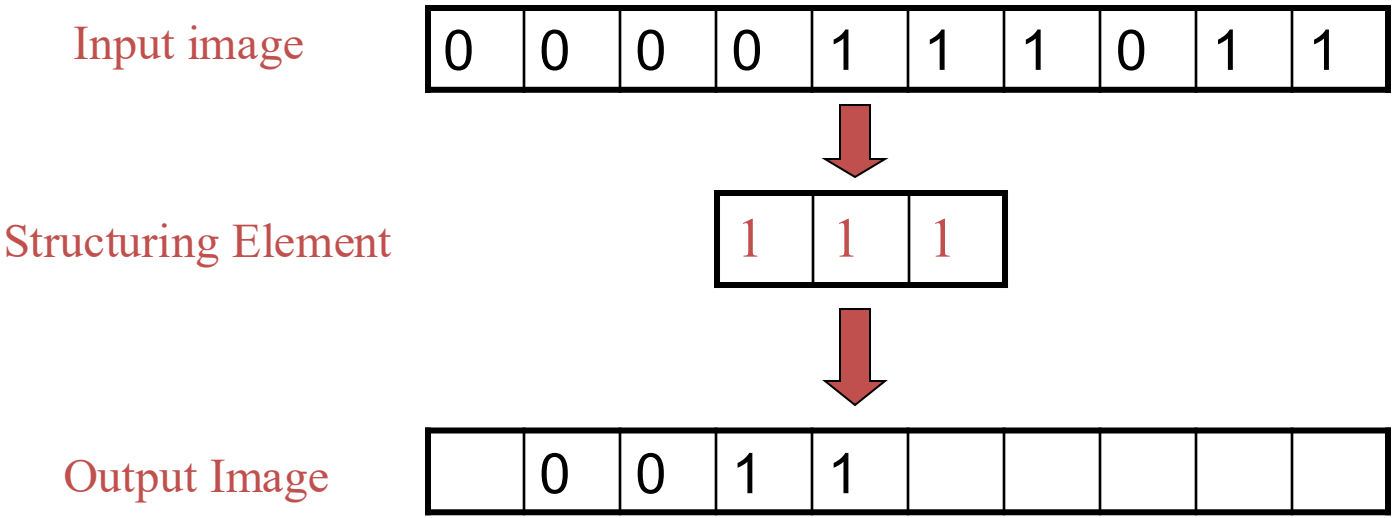
Example for Dilation



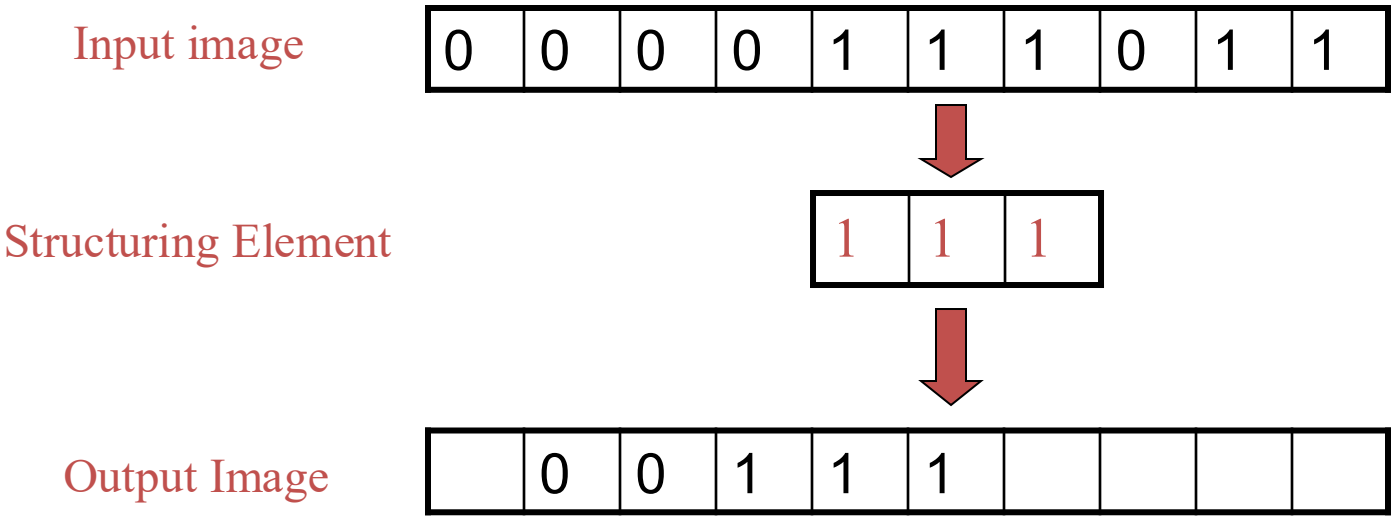
Example for Dilation



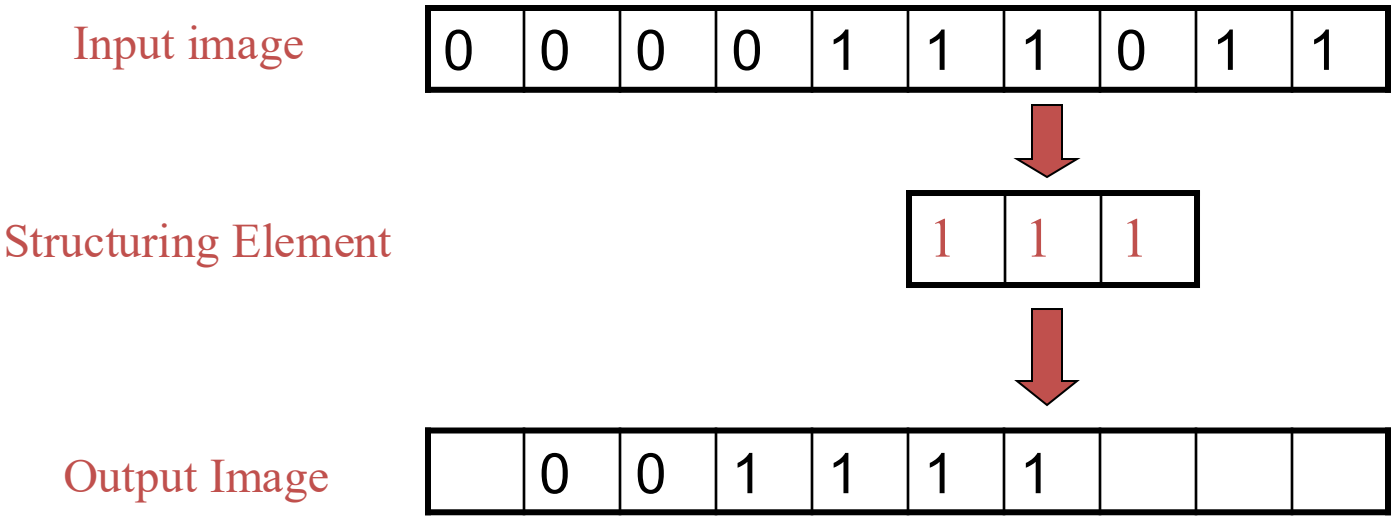
Example for Dilation



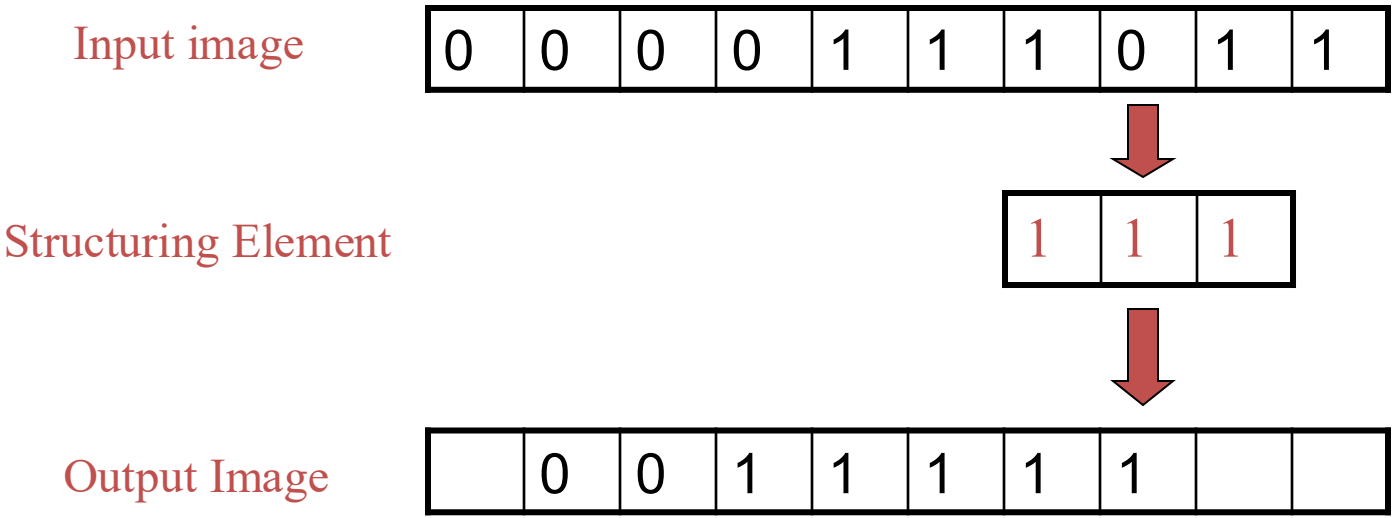
Example for Dilation



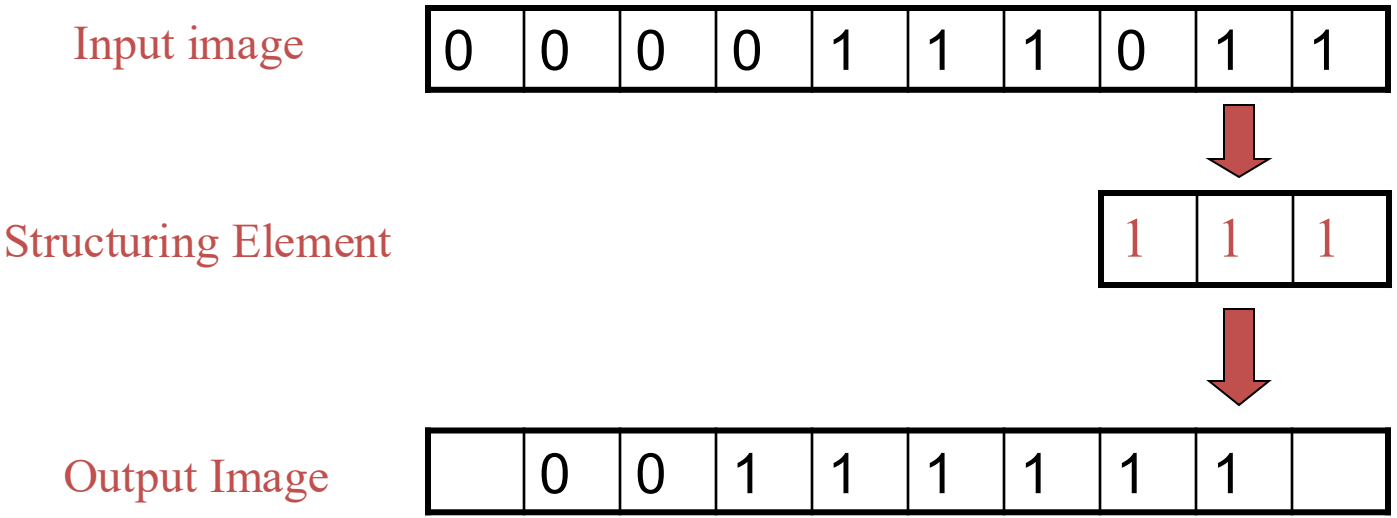
Example for Dilation



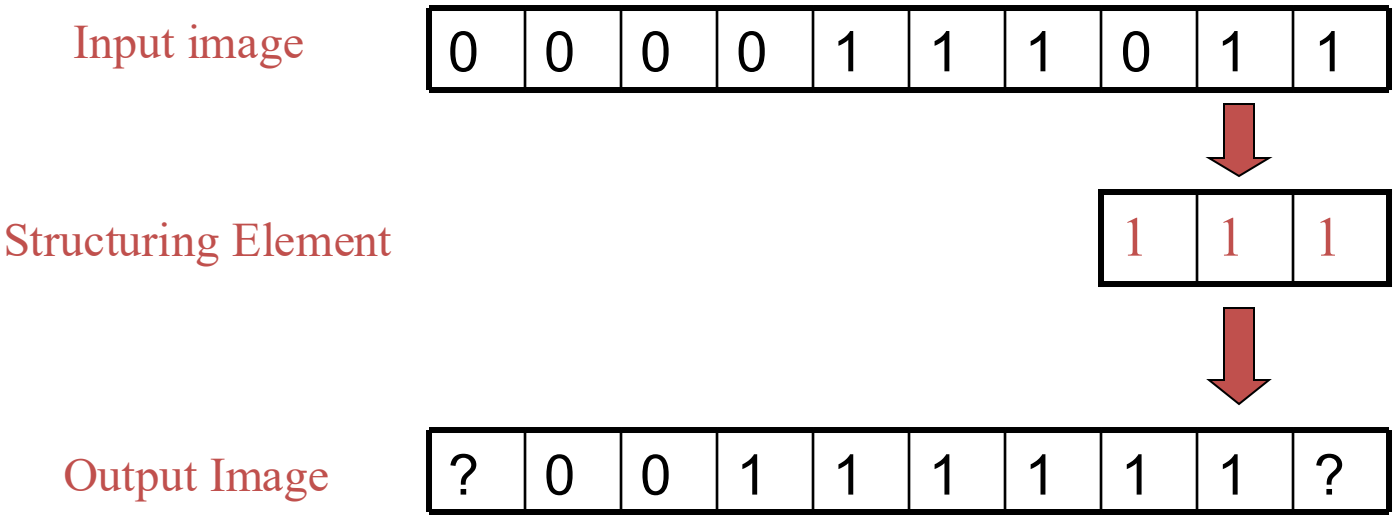
Example for Dilation



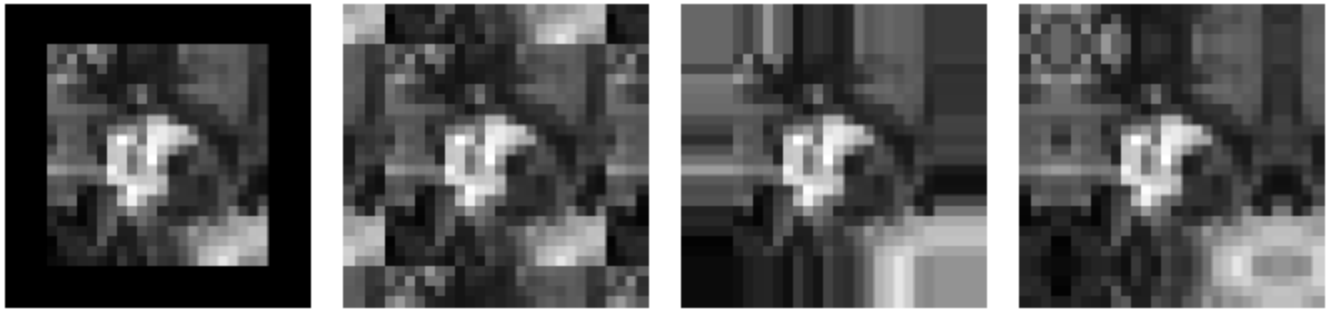
Example for Dilation



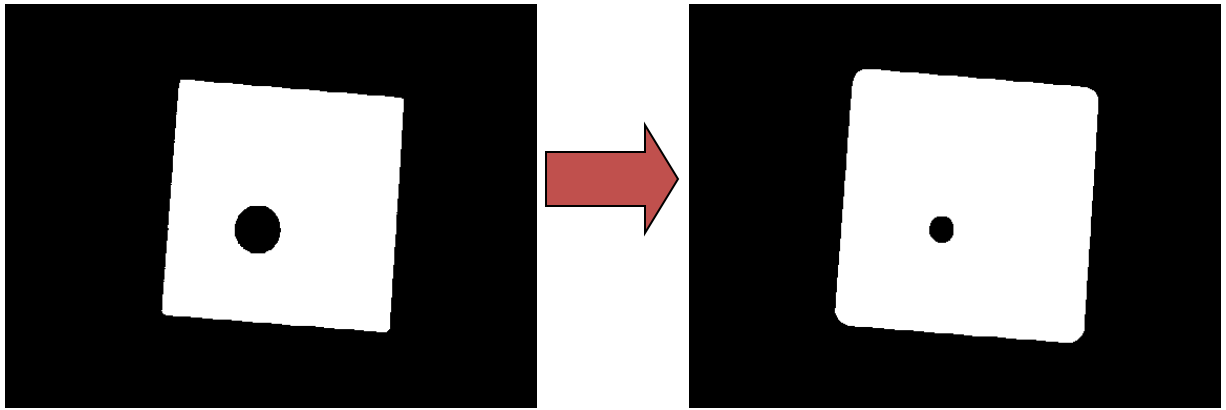
Example for Dilation



How to handle borders?



Another Dilation Example

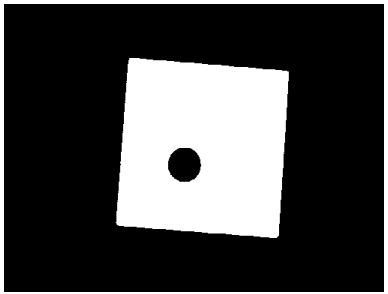


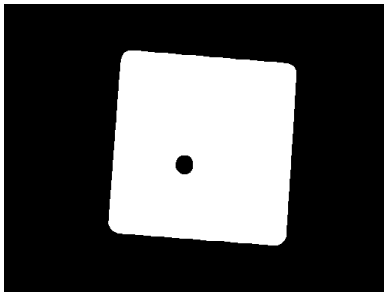
- Image get lighter, more uniform intensity

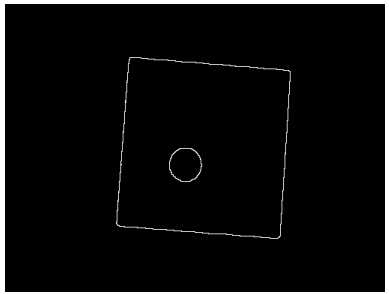
Edge detection example

- Edge Detection

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







Erosion

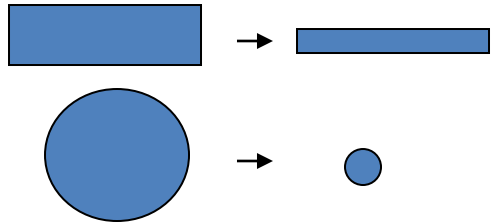
erode(B,S) takes a binary image B, places the origin of structuring element S over every pixel position, and **AND** a binary 1 into that position of the output image, if and only if every position of S (with a 1) covers a 1 in B.

Erosion

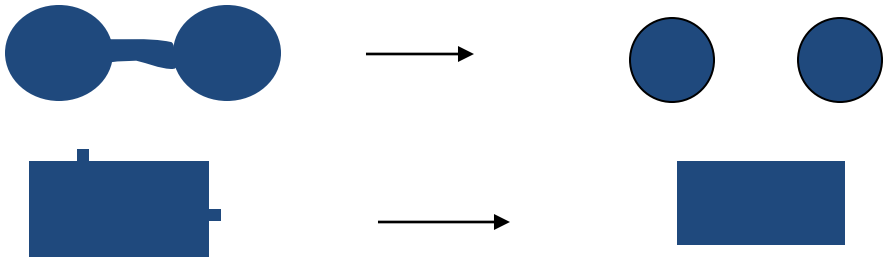
Erosion shrinks the connected sets of 1s of a binary image.

It can be used for

1. shrinking features

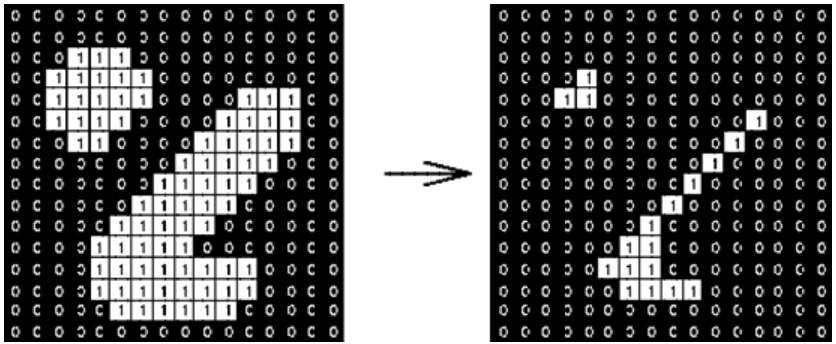


2. Removing bridges, branches and small protrusions



Erosion

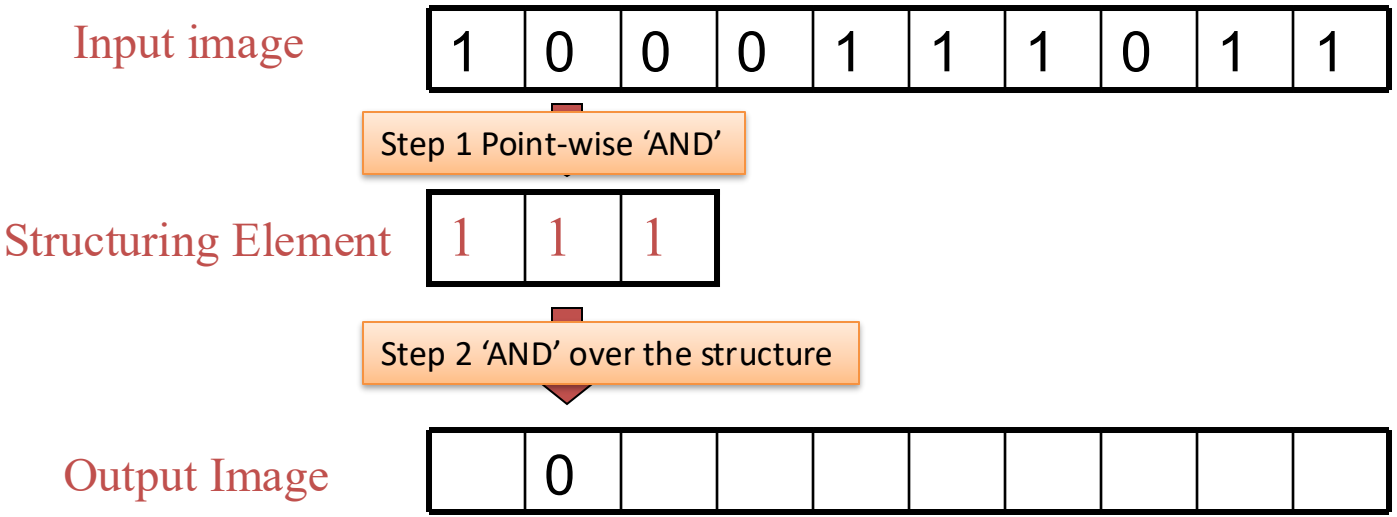
- **Erosion** is a basic morphological operation



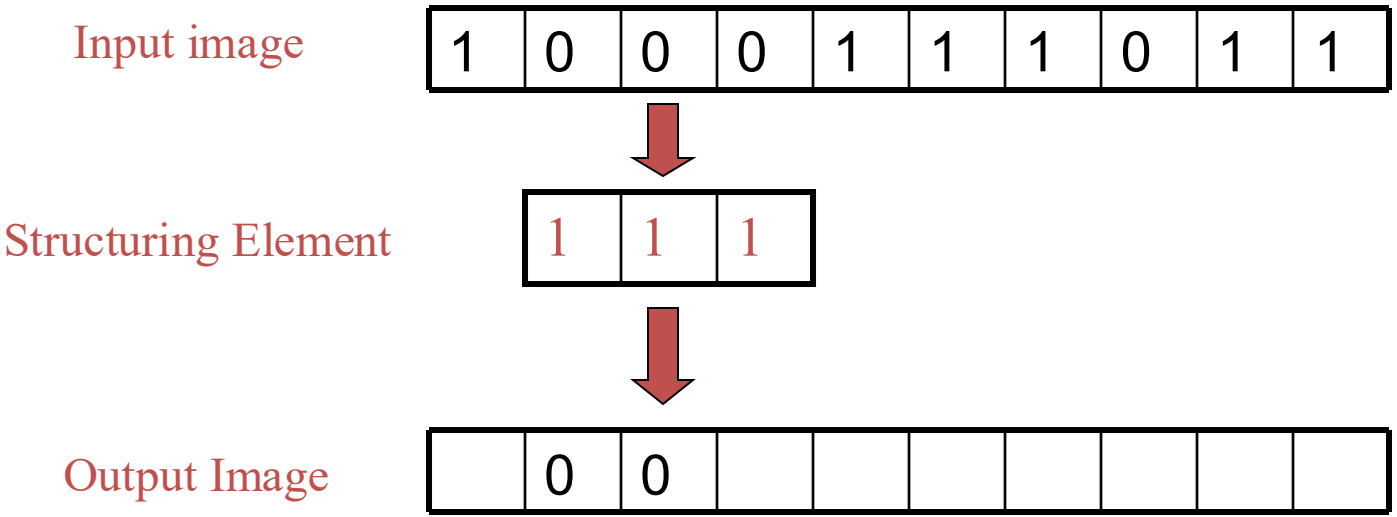
- Applied **Structuring Element**:

1	1	1
1	1	1
1	1	1

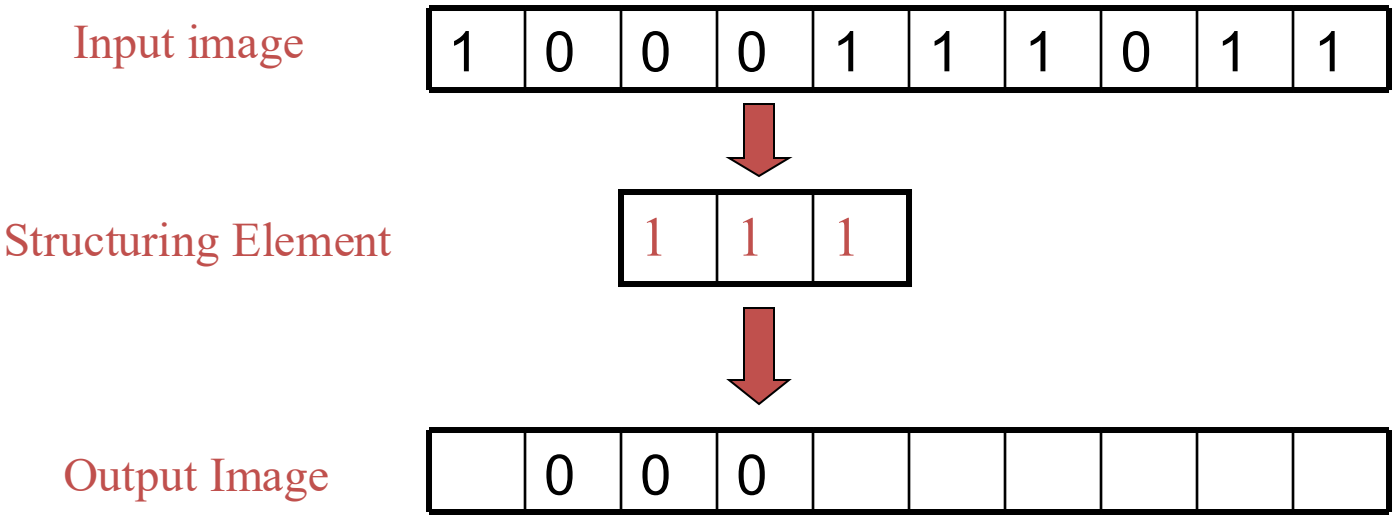
Example for Erosion



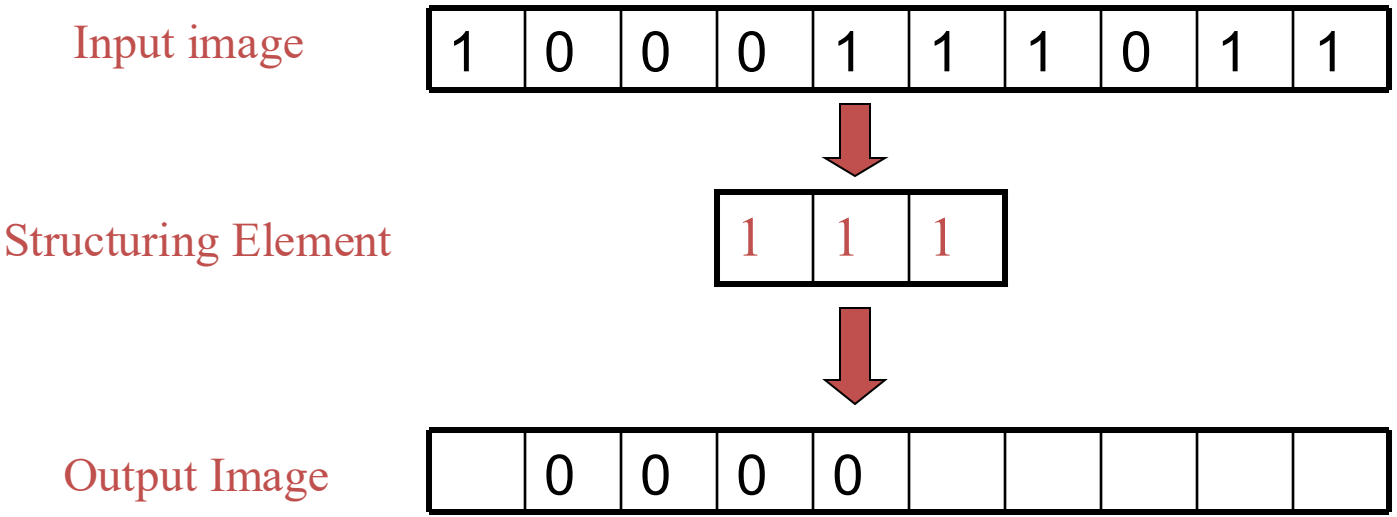
Example for Erosion



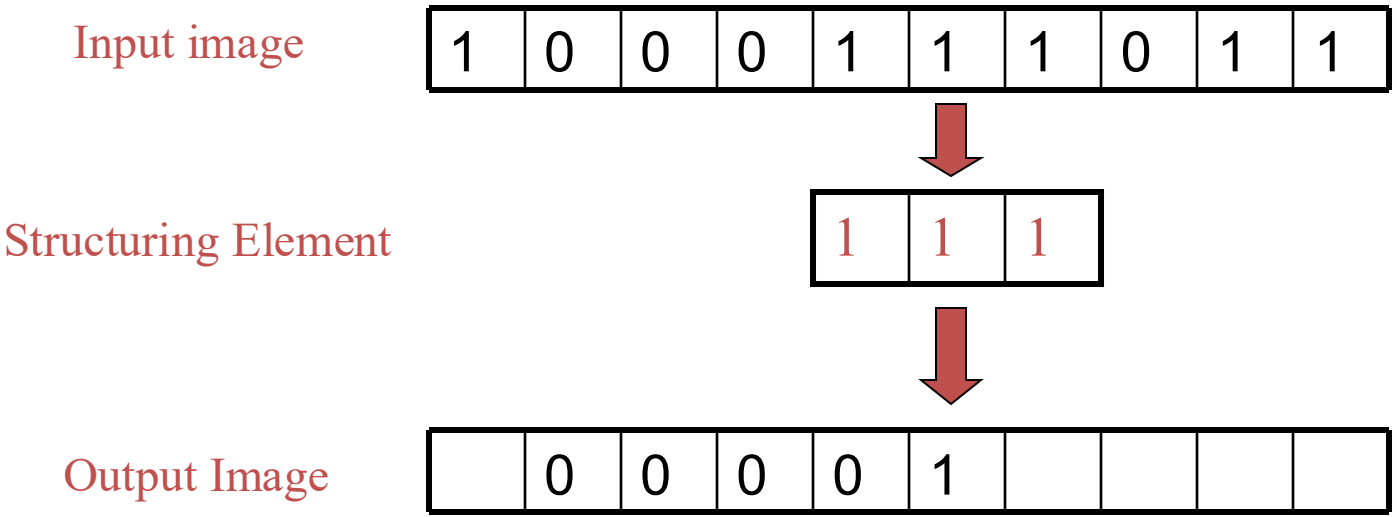
Example for Erosion



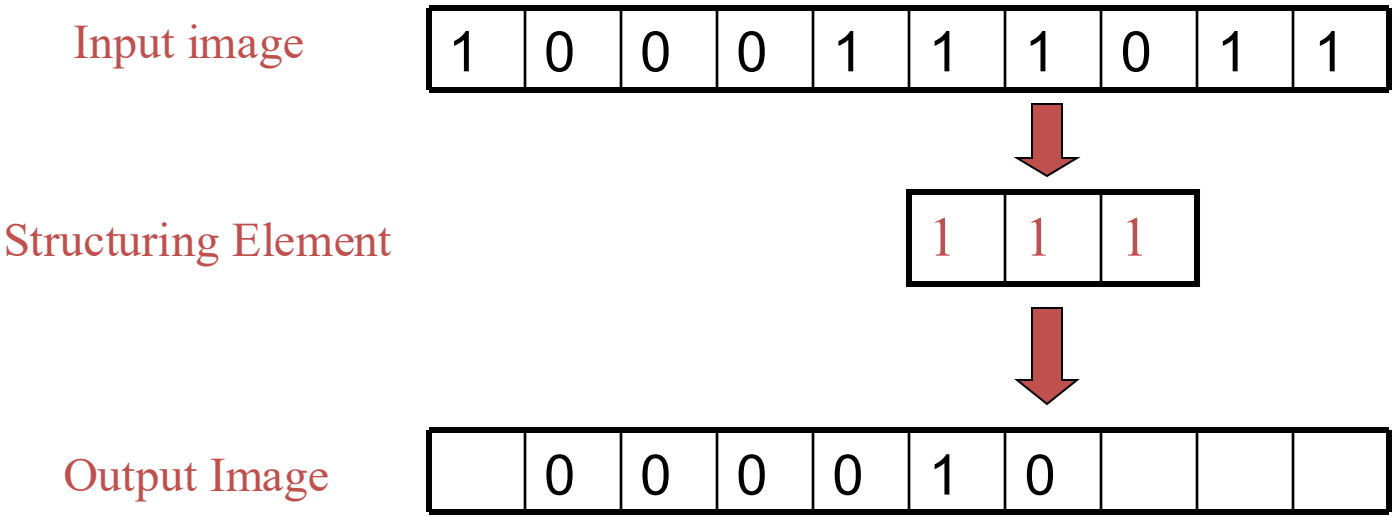
Example for Erosion



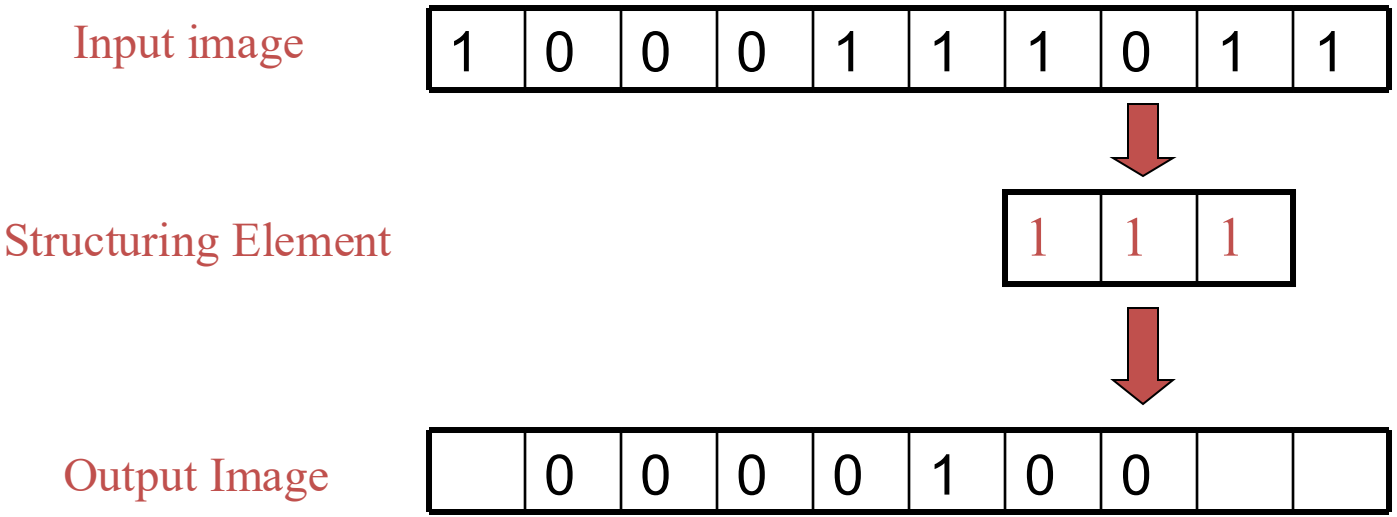
Example for Erosion



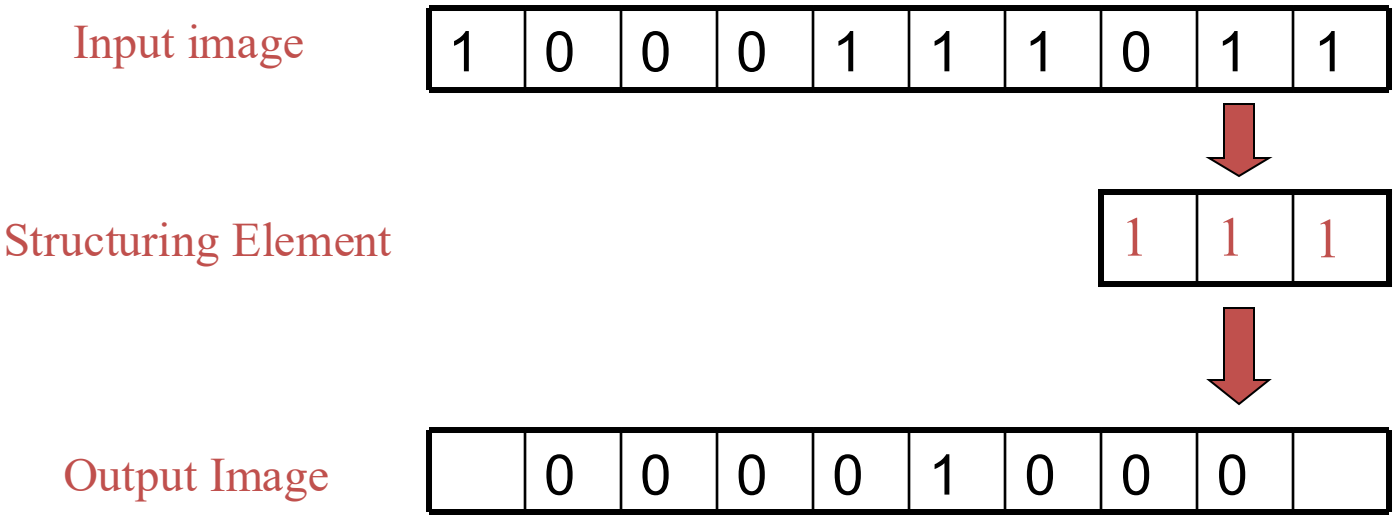
Example for Erosion



Example for Erosion

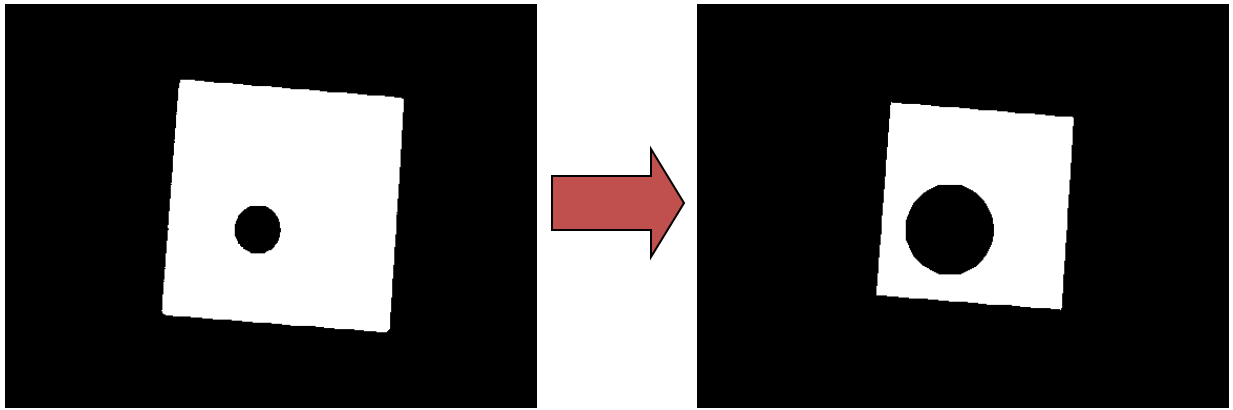


Example for Erosion



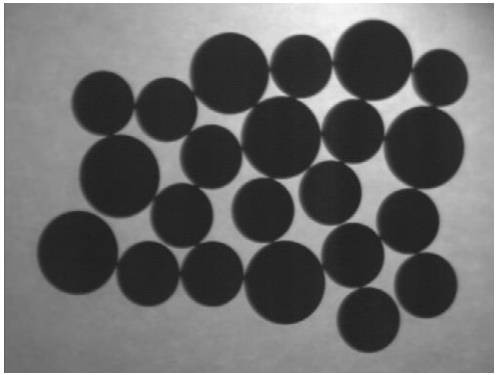
An example of erosion

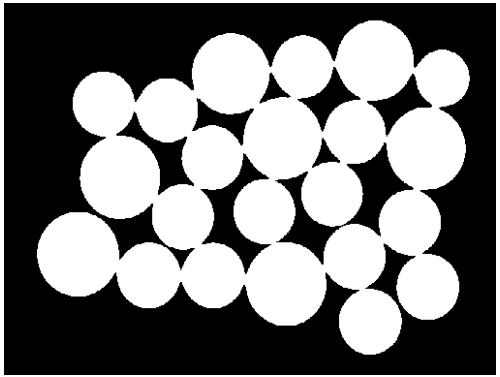
- White = 0, black = 1, dual property, image as a result of erosion gets darker

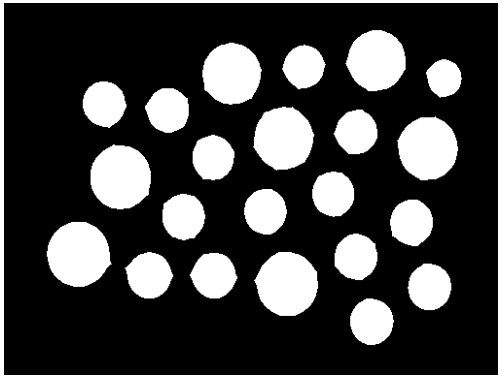


Counting Coins

- Counting coins is difficult if they touch each other!
- Solution: Binarization and Erosion separates them!





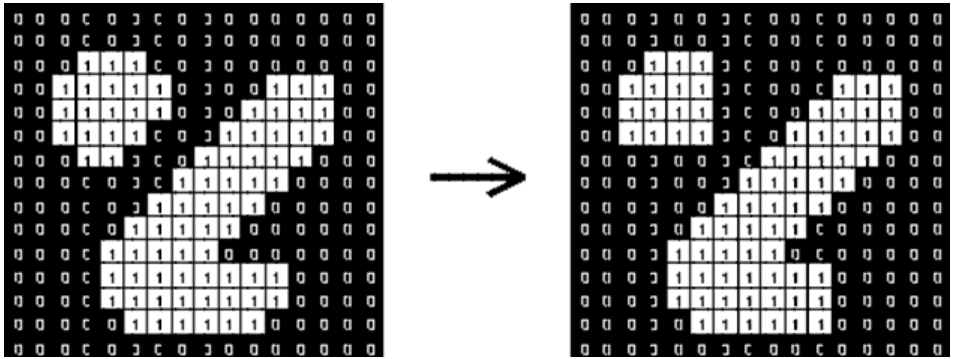


Opening

- Similar to Erosion
  - Spot and noise removal
  - Less destructive
- Erosion next dilation
- the s*ame structuring element for both operations.*

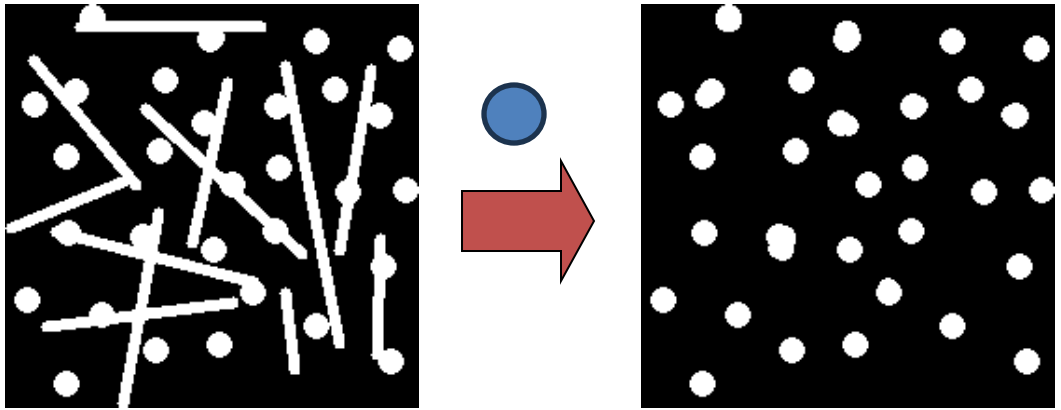
Opening

- Structuring element: 3x3 square

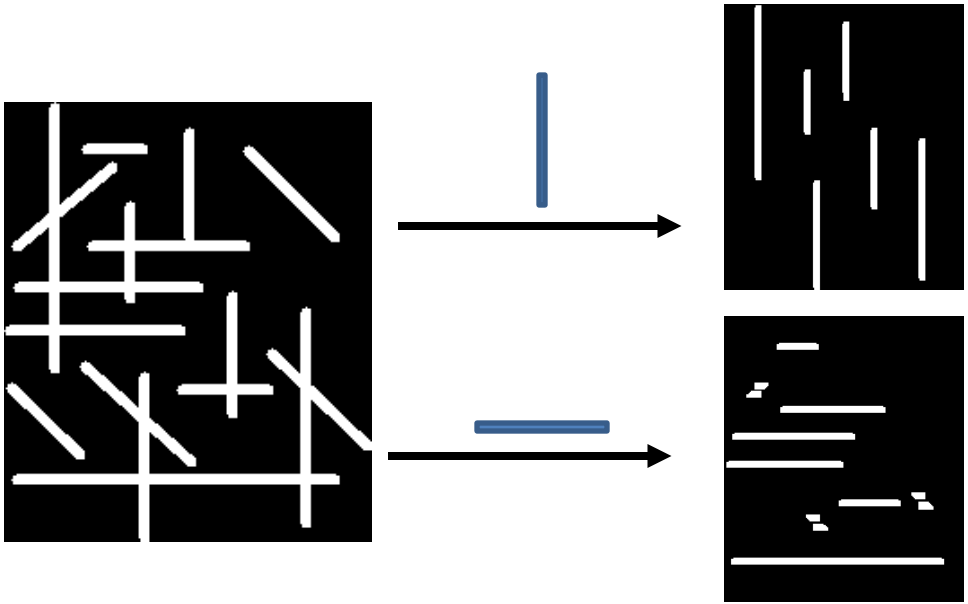


Opening Example

- Opening with a 11 pixel diameter disc
- Erosion plus dilation

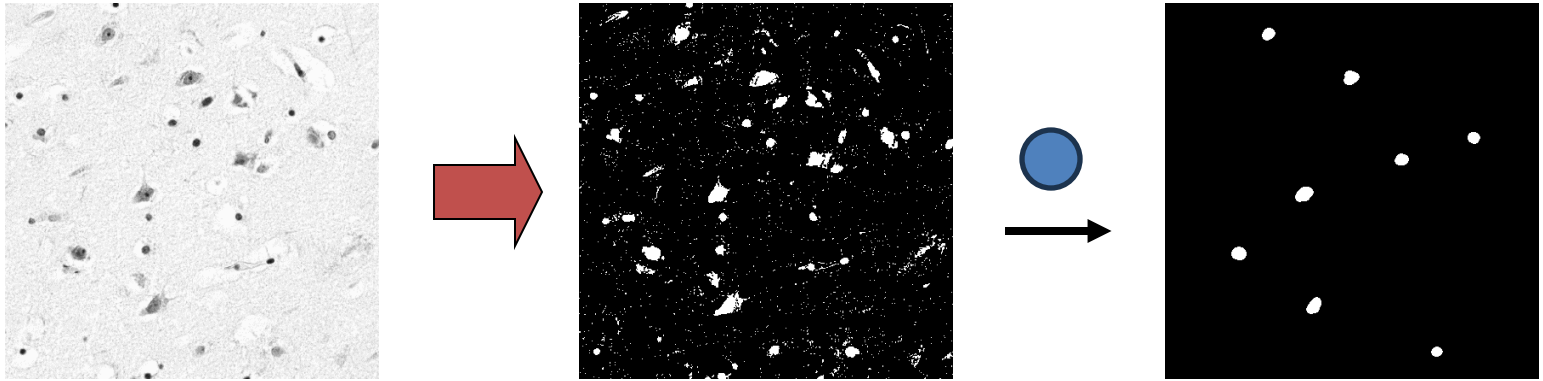


Opening Example



Use Opening for Separating Blobs

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

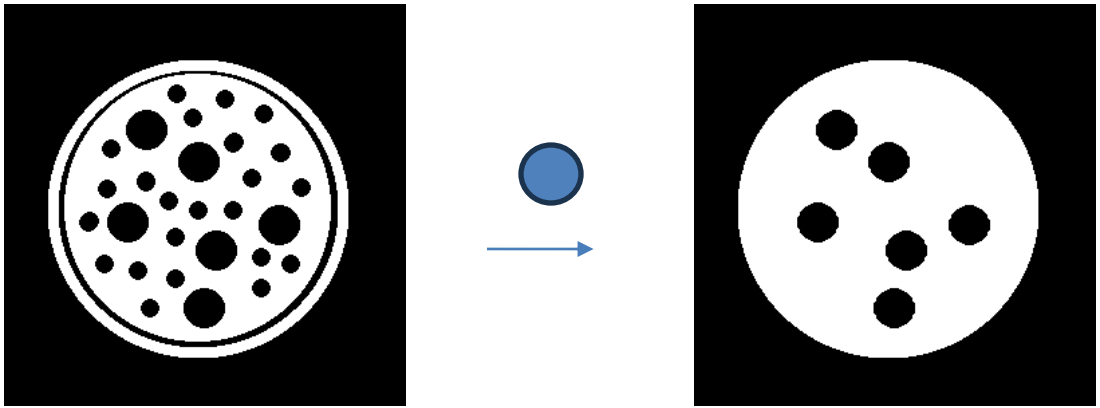


Closing

- Similar to Dilation
  - Removal of holes
  - Tends to enlarge regions, shrink background
- Dilation next erosion.

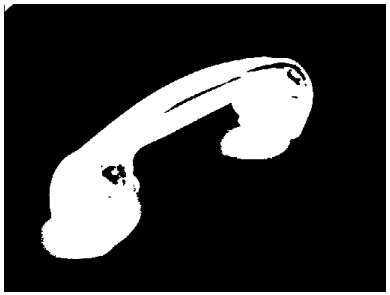
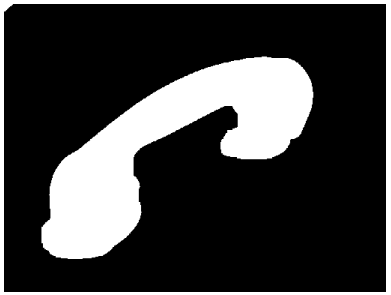
Closing Example

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



Closing Example

1. Threshold
2. Closing with disc of size 20

Opening & Closing

- Derived from the two basic operations
  - Dilation
  - Erosion
- Opening and closing are dual operations
  - i.e. opening the foreground pixels with a particular structuring element
  - is equivalent to closing the background pixels with the same element.

Outline

- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Filtering

Goal: Remove unwanted parts of a signal, and keep only task-relevant information.

For images, new pixel values are typically computed based on some function of the local neighborhood of the pixels.

[iStock]

Linear Filtering

A linear filter in can be written as $g(i,j) = \sum_{k,l} f(i+k, j+l) \cdot h(k,l)$

$f(x,y)$ → H → $g(x,y)$

Linear, if and only if: $\mathrm{H}\big(a(i,j) + b(i,j)\big) = \mathrm{H}\big(a(i,j)\big) + \mathrm{H}\big(b(i,j)\big)$

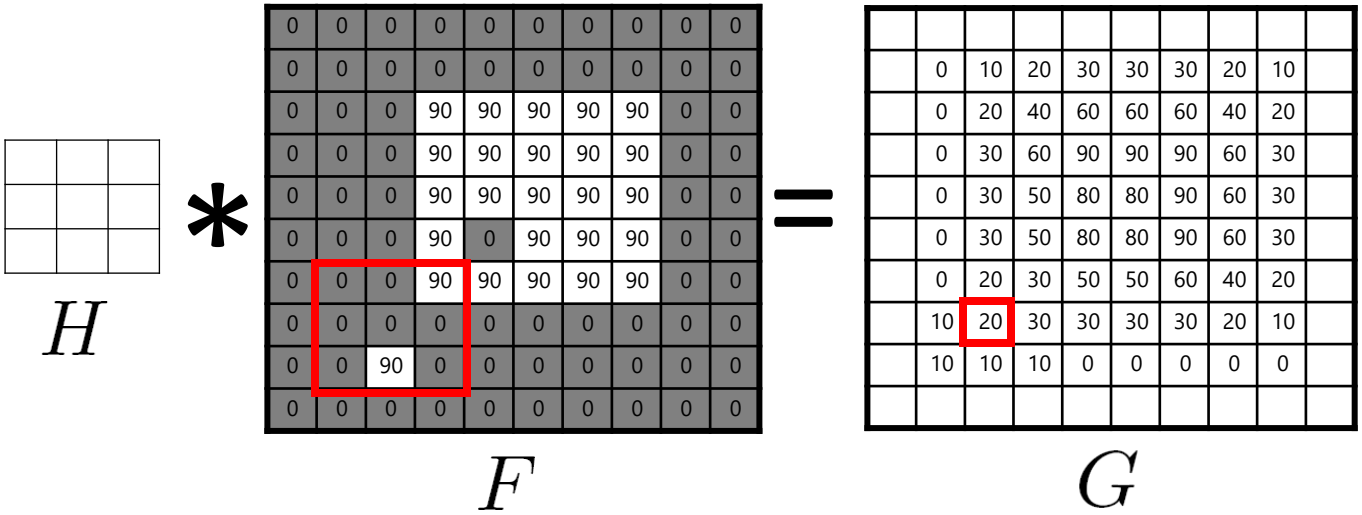
$$\mathrm{H}\big(C \cdot a(i,j)\big) = \mathrm{C} \cdot \mathrm{H}\big(a(i,j)\big)$$

Motivation: Noise reduction

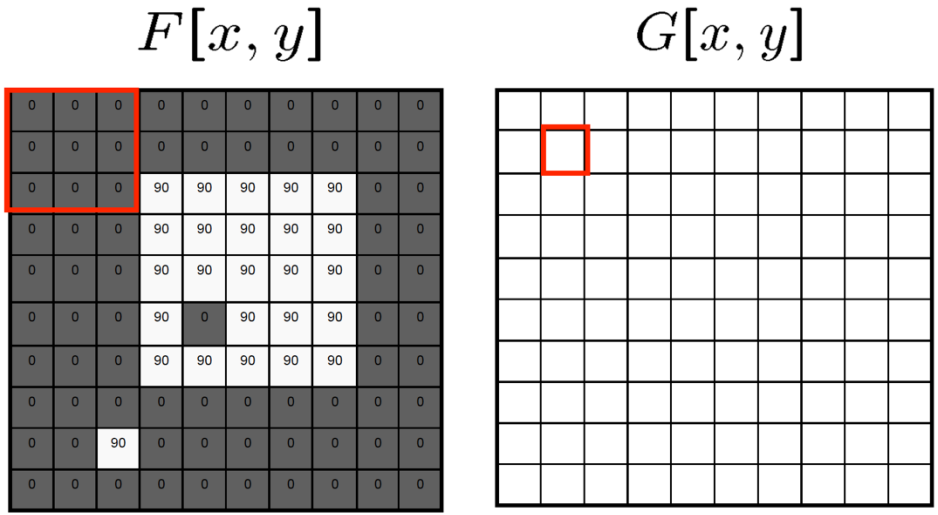
- Given a camera and a still scene, how can you reduce noise?

Source: S. Seitz

Mean filtering

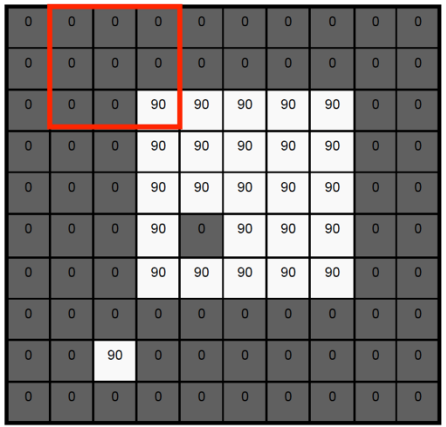


Mean filtering / Moving average

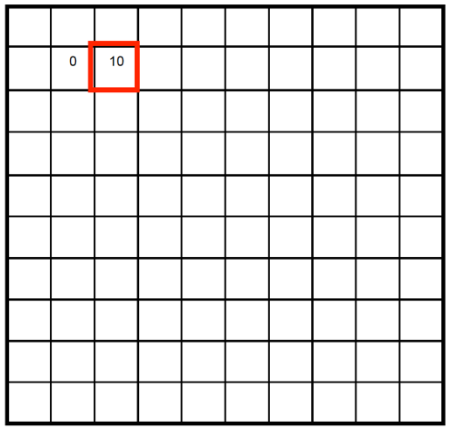


Mean filtering / Moving average

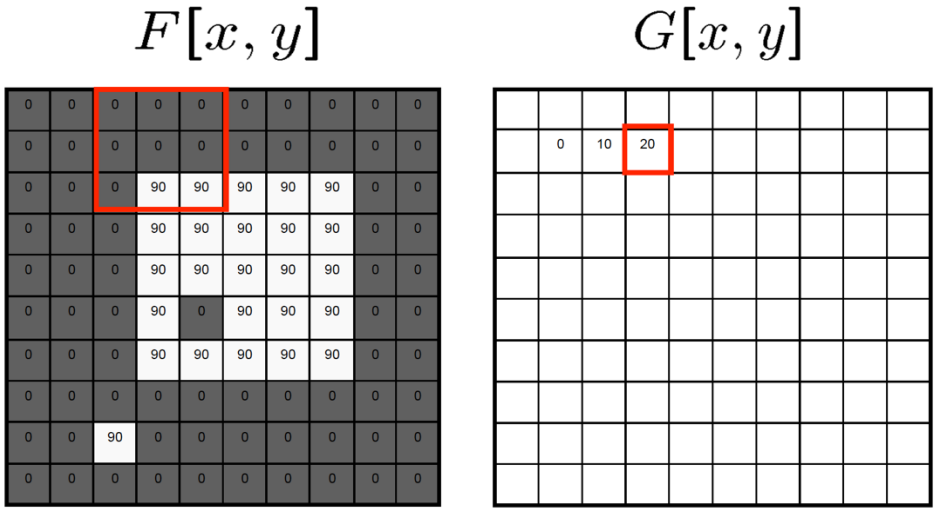
$F[x, y]$



$G[x, y]$

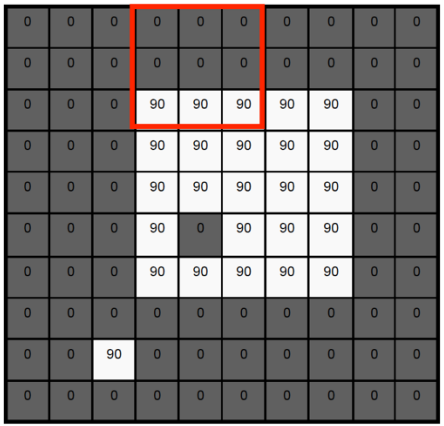


Mean filtering / Moving average

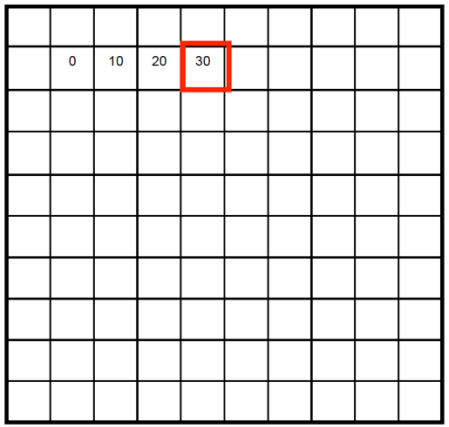


Mean filtering / Moving average

$F[x, y]$

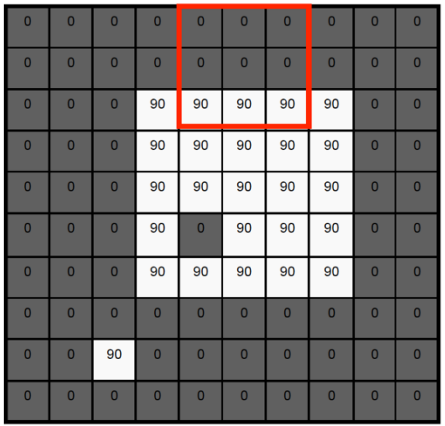


$G[x, y]$

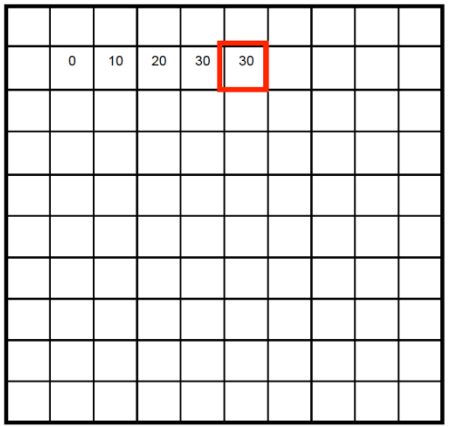


Mean filtering / Moving average

$F[x, y]$



$G[x, y]$



Mean filtering / Moving average

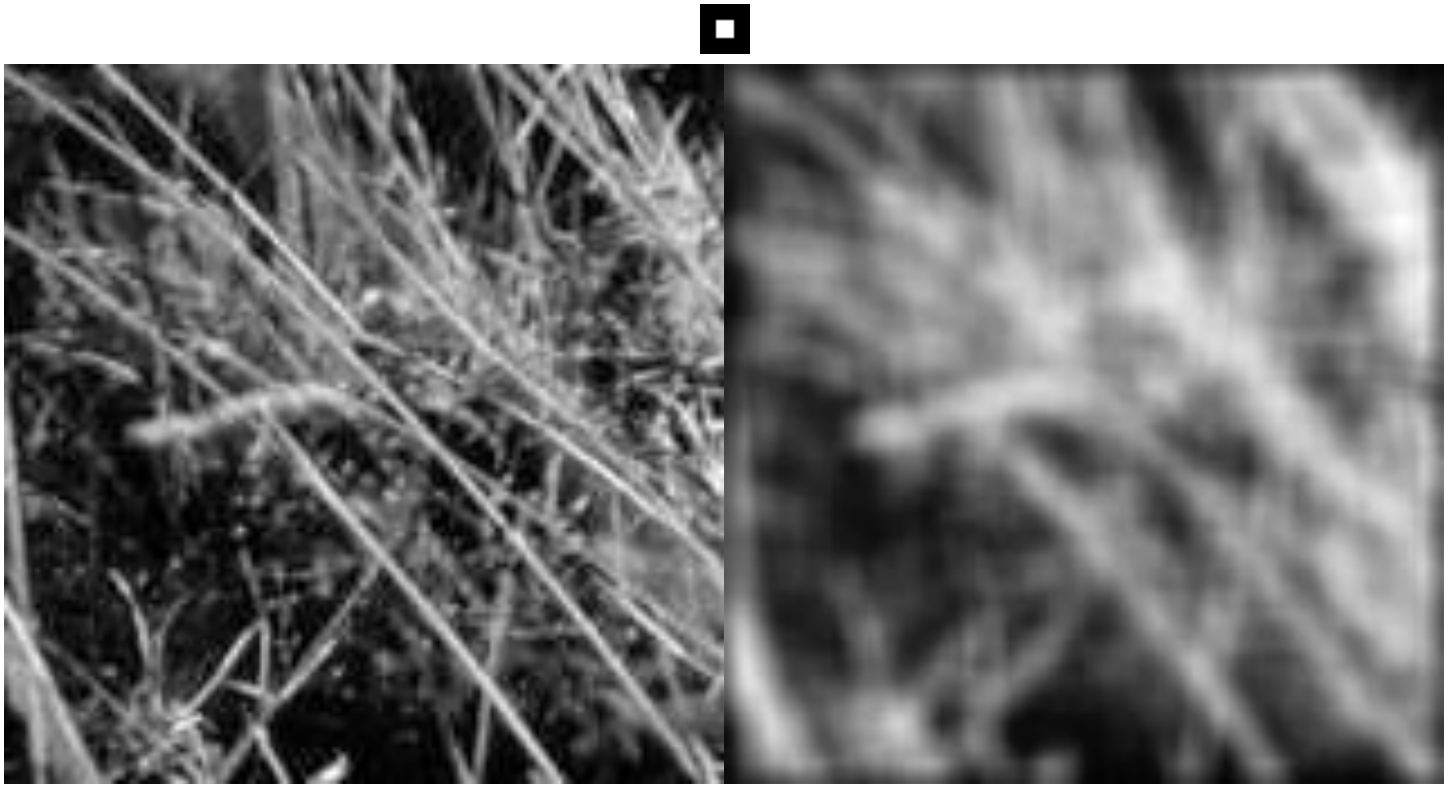
$F[x, y]$

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

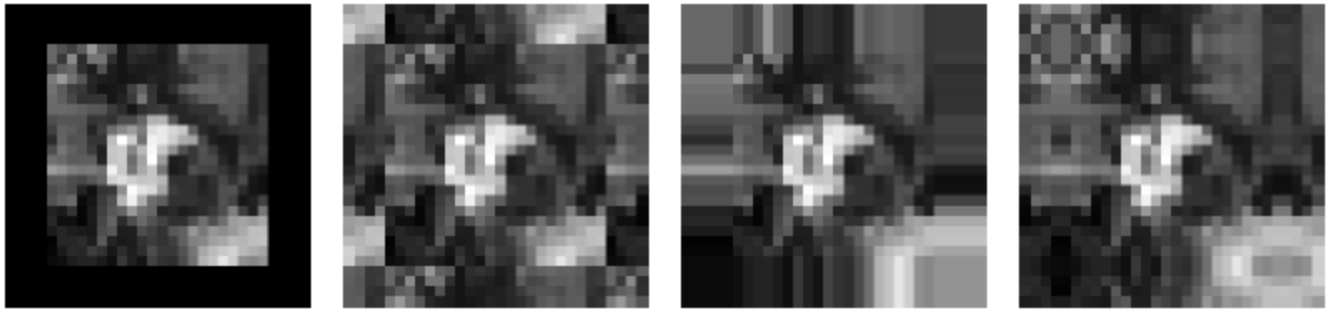
$G[x, y]$

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

Mean filtering / Moving average



How to handle borders?



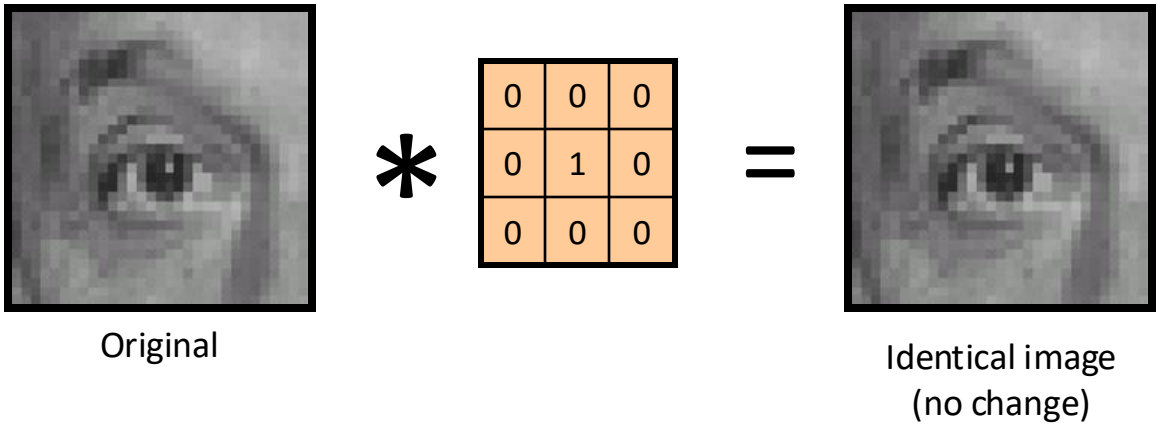
Linear filters: examples

Original

$$*\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \; ?$$

[Slide credit: David Forsyth]

Linear filters: examples



[Slide credit: David Forsyth]

Source: D. Lowe

Linear filters: examples

Original

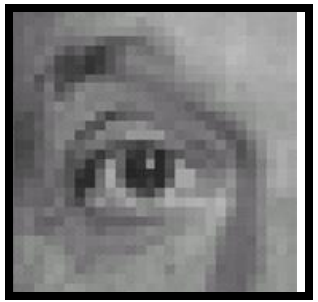
$$*\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \; ?$$

[Slide credit: David Forsyth]

Linear filters: examples

Original

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



Shifted left by 1 pixel

[Slide credit: David Forsyth]

Linear filters: examples

Original

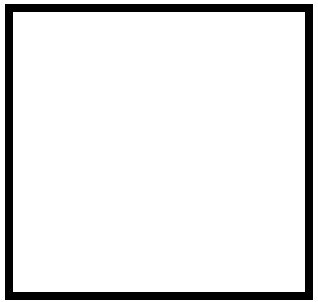
$$*\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} = \; ?$$

[Slide credit: David Forsyth]

Linear filters: examples

Original

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



[Slide credit: David Forsyth]

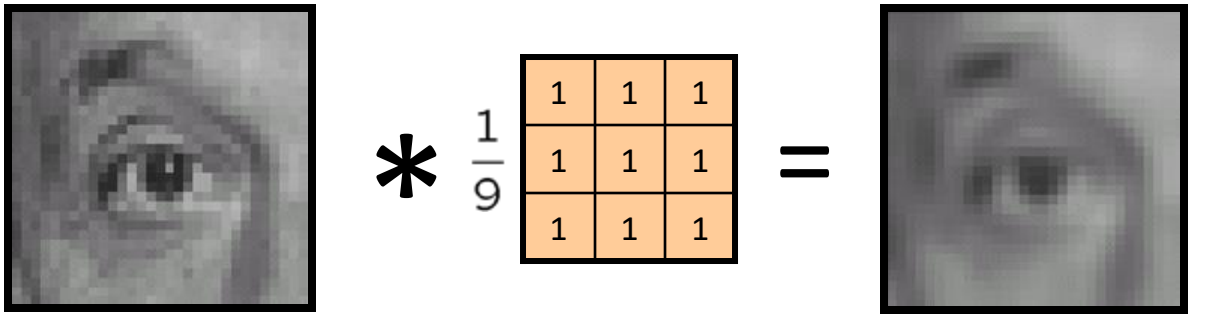
Linear filters: examples

Original

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

[Slide credit: David Forsyth]

Linear filters: examples

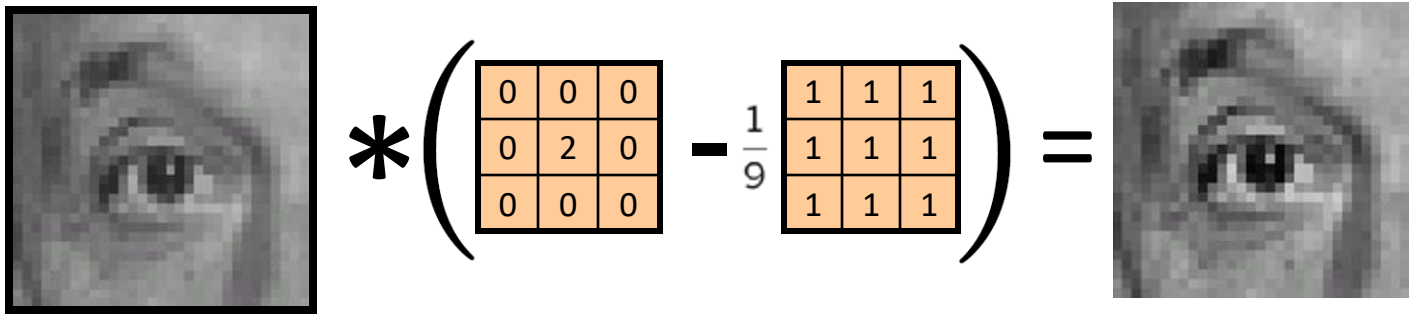


Original

Blur (with a mean filter)

[Slide credit: David Forsyth]

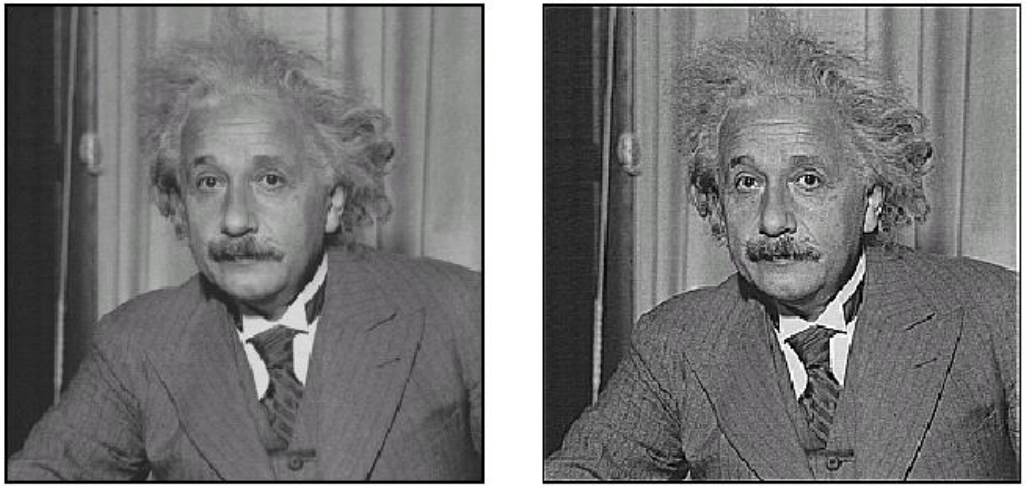
Linear filters: examples



Original

Sharpening filter
(accentuates edges)

Sharpening



before

after

[Slide credit: David Forsyth]

Edge filter

[-1 1]

$$\frac{\partial \mathbf{I}}{\partial x} \simeq \mathbf{I}(x,y) - \mathbf{I}(x-1,y)$$

g[m,n]

$\otimes$

[-1, 1]

h[m,n]

=

f[m,n]

Edge filter

$[-1\ 1]^T$

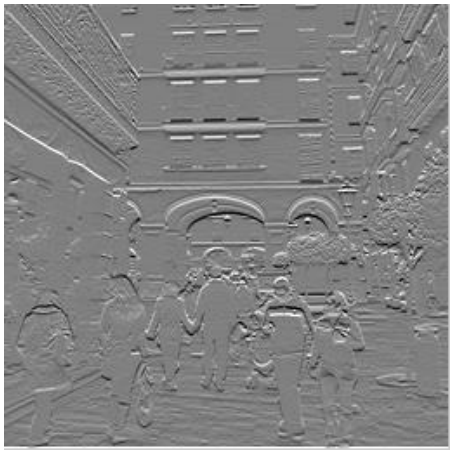
g[m,n]

$\otimes$

$[-1, 1]^T$

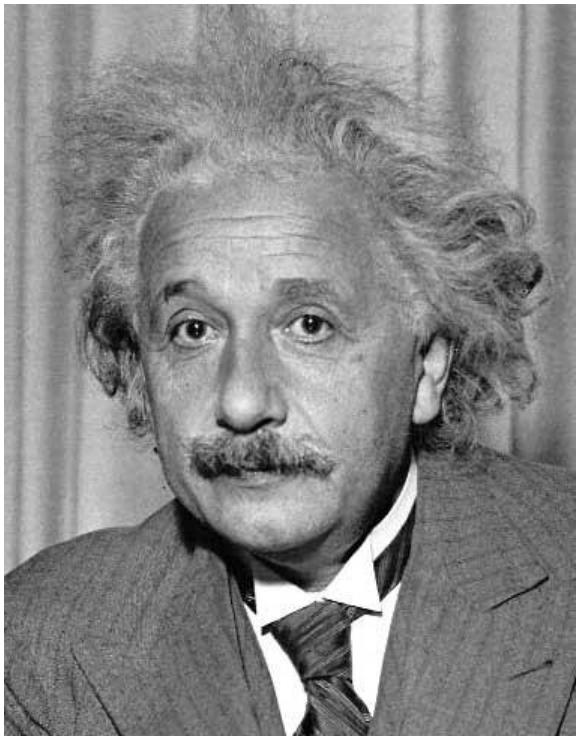
h[m,n]

=



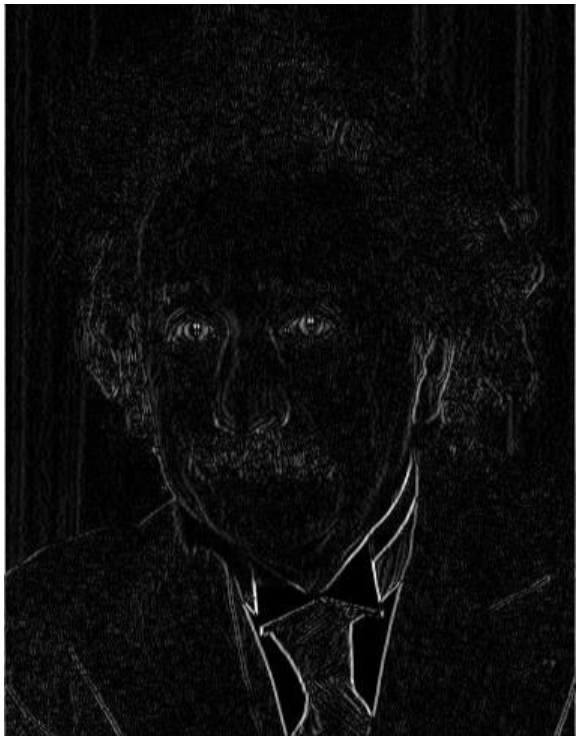
f[m,n]

Derivative filters



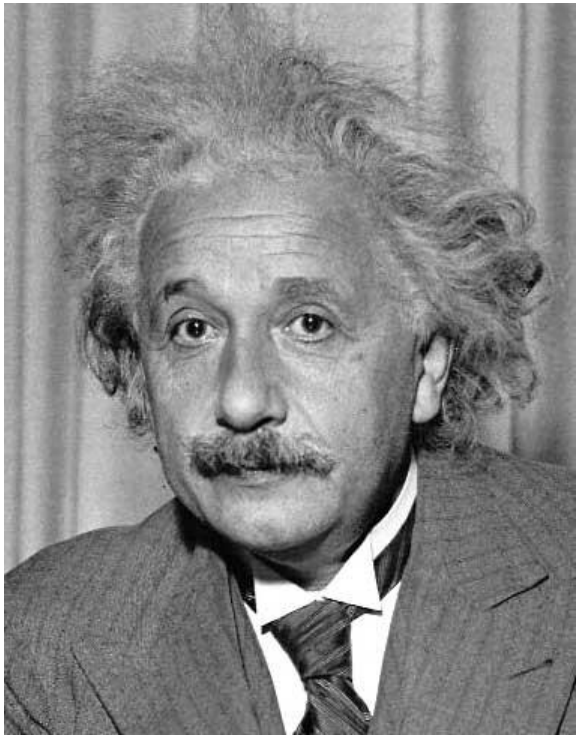
1	0	-1
2	0	-2
1	0	-1

Sobel



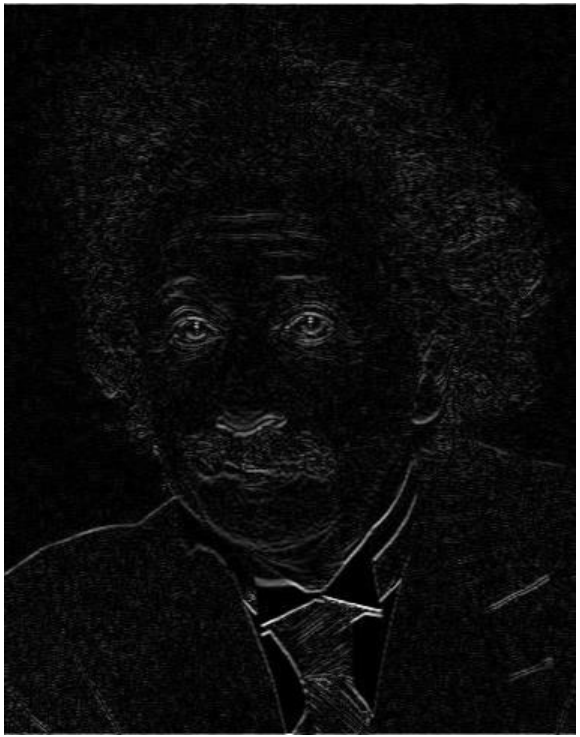
Vertical Edge
(absolute value)

Derivative filters



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

Sobel



Horizontal Edge
(absolute value)

Motion blur filter

g[m,n]

h[m,n]

=

f[m,n]

Motion blur filter

g[m,n]

h[m,n]

=

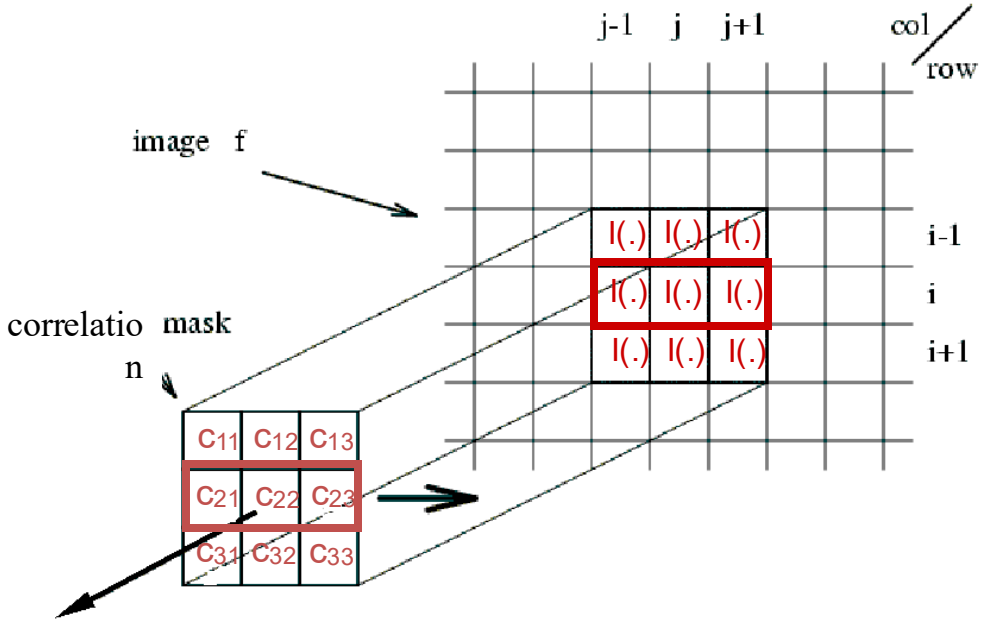
f[m,n]

Outline

- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Correlation

Motivation: Template Matching



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

Correlation

- Linear operation of *correlation*:

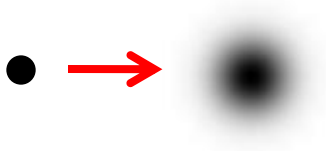
$$I' = K \otimes I$$

$$I'(x,y) = \sum_{(i,j) \in N(x,y)} K(i,j) I(x+i, y+j)$$

- Represent the linear weights as an image, ***K***

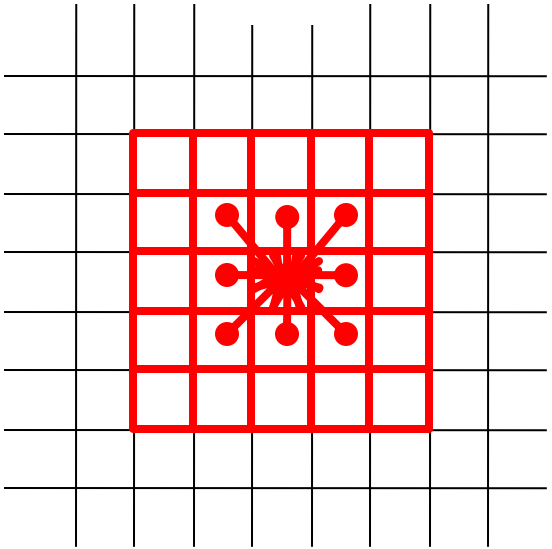
Convolution

Motivation: point spread function



Kernel

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



$$I'(x,y) = K(1,1)I(x-1,y-1) + K(0,1)I(x,y-1) + K(-1,1)I(x+1,y-1)$$
$$+ K(1,0)I(x-1,y) + K(0,0)I(x,y) + K(-1,0)I(x+1,y)$$
$$+ K(1,-1)I(x-1,y+1) + K(0,-1)I(x-1,y) + K(-1,-1)I(x+1,y+1)$$

Convolution

- Linear operation of *convolution*:

$$I' = K * I$$

$$I'(x, y) = \sum_{(i,j) \in N(x,y)} K(i, j) I(x - i, y - j)$$

- Represent the linear weights as an image, ***K***
- Same as correlation, but with kernel reversed

Correlation vs. Convolution

Correlation

$$I'(x, y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} K(i, j) I(x + i, y + j)$$

Convolution

$$I'(x, y) = \sum_{j=-k}^{k} \sum_{i=-k}^{k} K(i, j) I(x - i, y - j)$$

$$= \sum_{j=-k}^{k} \sum_{i=-k}^{k} K(-i, -j) I(x + i, y + j)$$

So if *K(i,j)* = *K(-i, -j)*, then Correlation == Convolution

Mathematical definition of image convolution

- Image $f(x, y)$ and kernel $k(x, y)$
- Continuous form, $(x, y) \in \mathbb{R}^2$

$$k(x, y) * f(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} k(u, v) f(x - u, y - v) du dv$$

- Discrete form, kernel size (W_k, H_k), Image size (W_f, H_f),

$$k(x, y) * f(x, y) = \sum_{i=0}^{W_k - 1} \sum_{j=0}^{H_k - 1} k(i, j) f(x - i, y - j)$$

- Commutative:

$$k(x, y) * f(x, y) = f(x, y) * k(x, y)$$

Mathematical definition of cross correlation

- Image $f(x, y)$ and kernel $k(x, y)$
- Continuous form, $(x, y) \in \mathbb{R}^2$

$$k(x,y) \otimes f(x,y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} k(u,v) f(x+u, y+v) du dv$$

- Discrete form, kernel size (W_k, H_k), Image size (W_f, H_f),

$$k(x,y) \otimes f(x,y) = \sum_{i=0}^{W_k - 1} \sum_{j=0}^{H_k - 1} k(i,j) f(x+i, y+j)$$

- If $k = f$, the operation is called auto-correlation
- Relation with convolution

$$k(x,y) \otimes f(x,y) = k(-x,-y) * f(x,y)$$

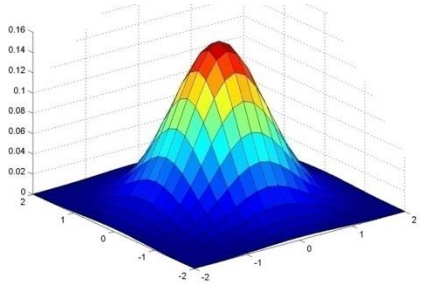
Image filtering

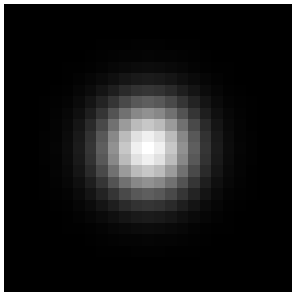
Useful for…

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

Important linear filter: Gaussian

- Weight contributions of neighboring pixels by nearness





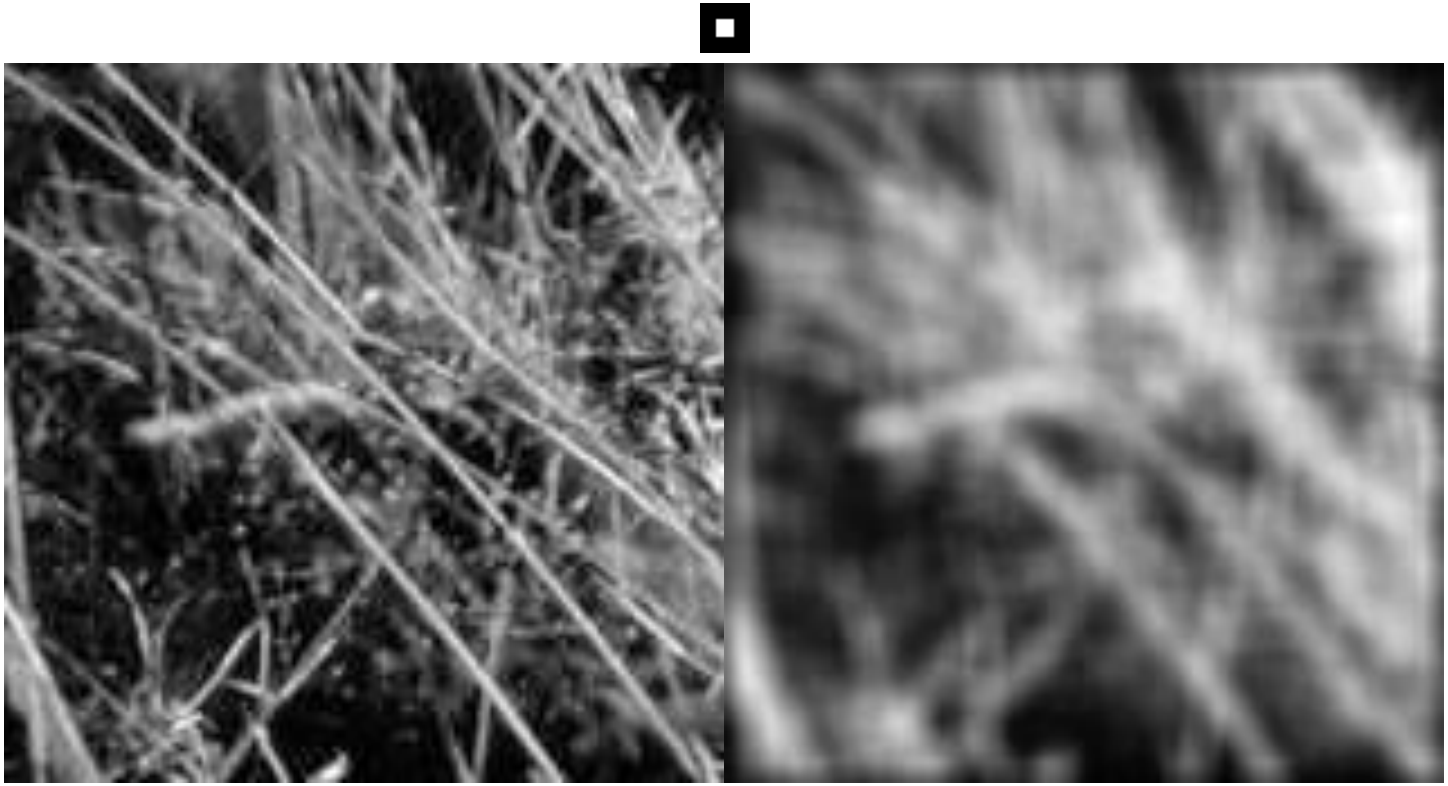
0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

5 x 5, $\sigma = 1$

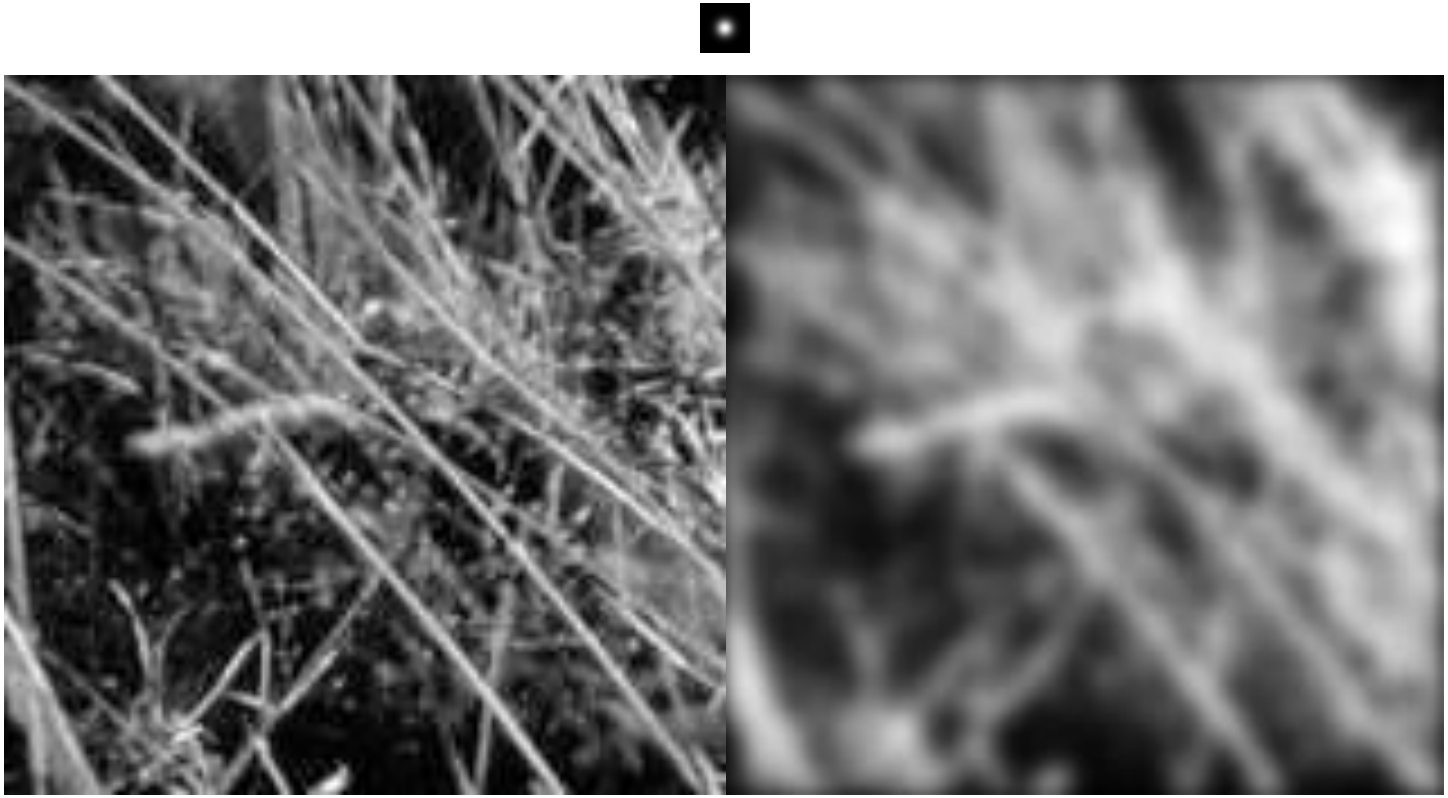
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

Slide credit: Christopher Rasmussen

Smoothing with box filter

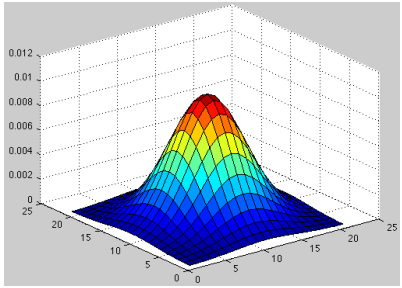


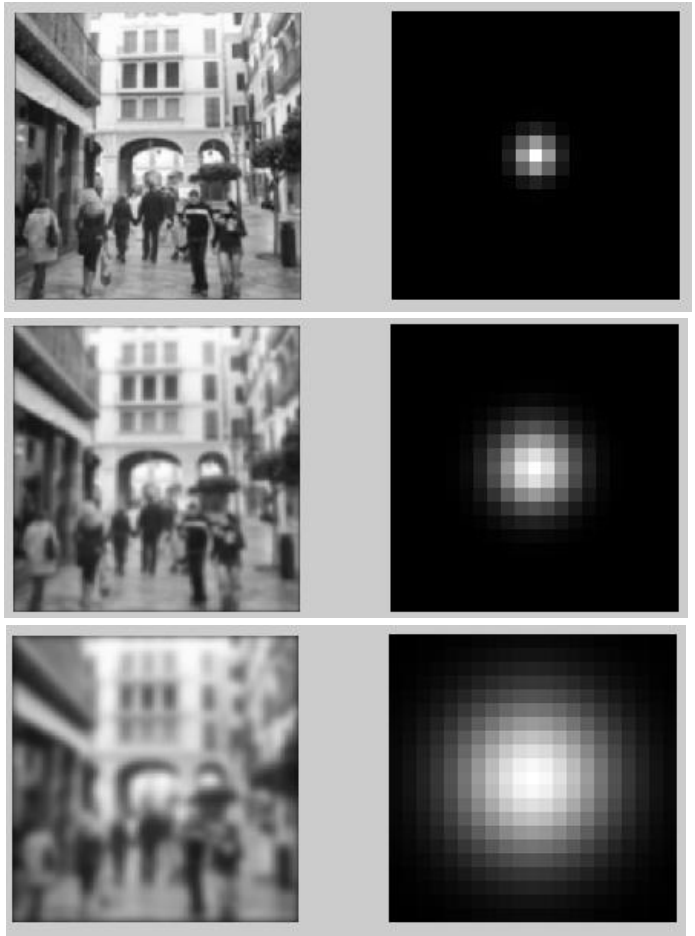
Smoothing with Gaussian filter



Gaussian filter: change scales

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





σ=2

σ=4

Properties of Gaussian filter

- Rotational symmetry treats features of all orientations equally (isotropy).

- Efficient: Rule of thumb is kernel width $\geq 5\sigma$
  - Separable
  - Cascadable: Approach to large σ comes from identity

$$G_{\sigma_1} * G_{\sigma_2} = G_{\sqrt{\sigma_1^2+\sigma_2^2}}$$

Separability of the Gaussian filter

$$
\begin{aligned}
G_\sigma(x,y) &= \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{2\sigma^2}} \\
&= \left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{x^2}{2\sigma^2}}\right)\left(\frac{1}{\sqrt{2\pi}\sigma}\exp^{-\frac{y^2}{2\sigma^2}}\right)
\end{aligned}
$$

The 2D Gaussian can be expressed as the product of two functions, one a function of x and the other a function of y

In this case, the two functions are the (identical) 1D Gaussian

Source: D. Lowe

Separability example

2D convolution
(center location only)

1	2	1
2	4	2
1	2	1

$*$

2	3	3
3	5	5
4	4	6

The filter factors
into a product of 1D
filters:

1	2	1
2	4	2
1	2	1

=

1
2
1

x

1	2	1

Perform convolution
along rows:

1	2	1

$*$

2	3	3
3	5	5
4	4	6

=

	11	
	18	
	18	

Followed by convolution
along the remaining column:

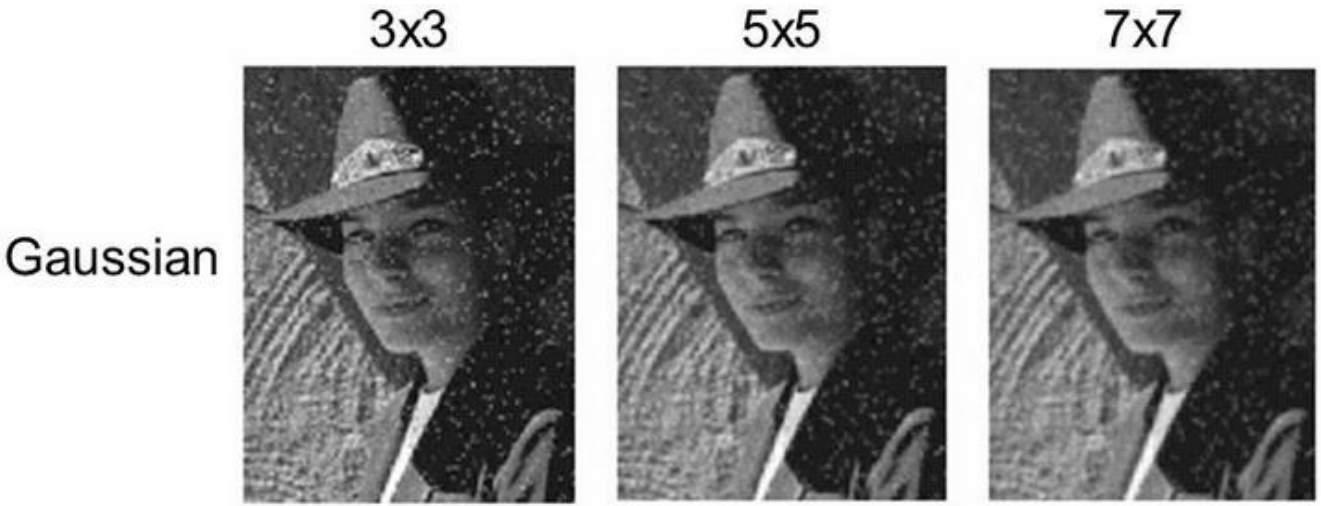
Source: K. Grauman

Outline

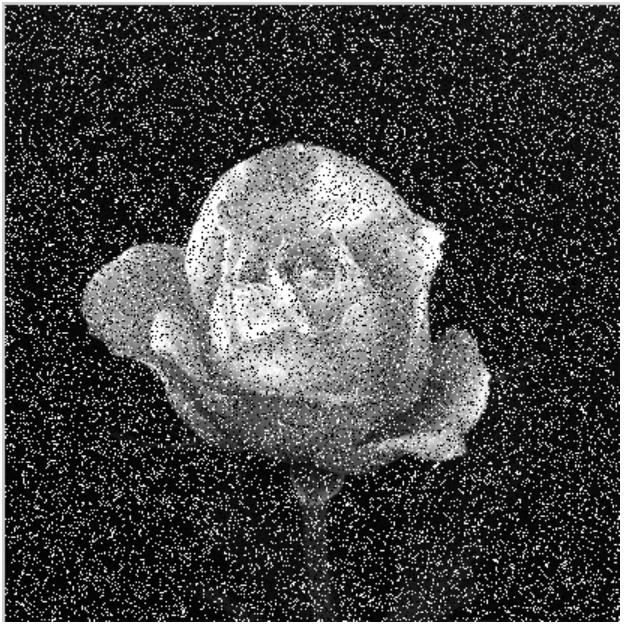
- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Salt-and-Pepper Noise

- Noise with random occurrence of white and black pixels (signal independent).
  Can be due to transmission errors, corrupted pixels, faulted memory locations, etc.



How to better preserve image edges?



Use Median Instead of Average

- Toy Example (1D):

1, 5, 2, 80, 7

1, 2, 5, 7, 80

Mean = 19

Median = 5

Median Filter

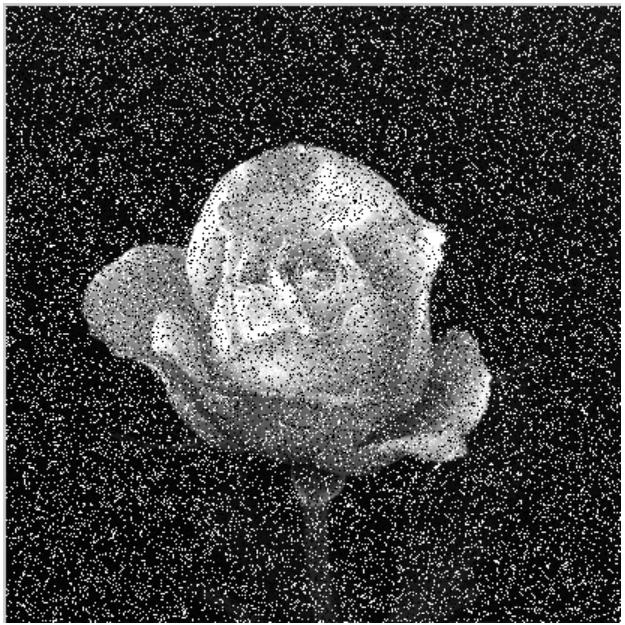
What is the value of the yellow box after 3x3 median filtering?

1	2	2	2	3
2	5	2	4	3
2	2	2	2	3
4	3	7	5	3
3	3	3	1	1

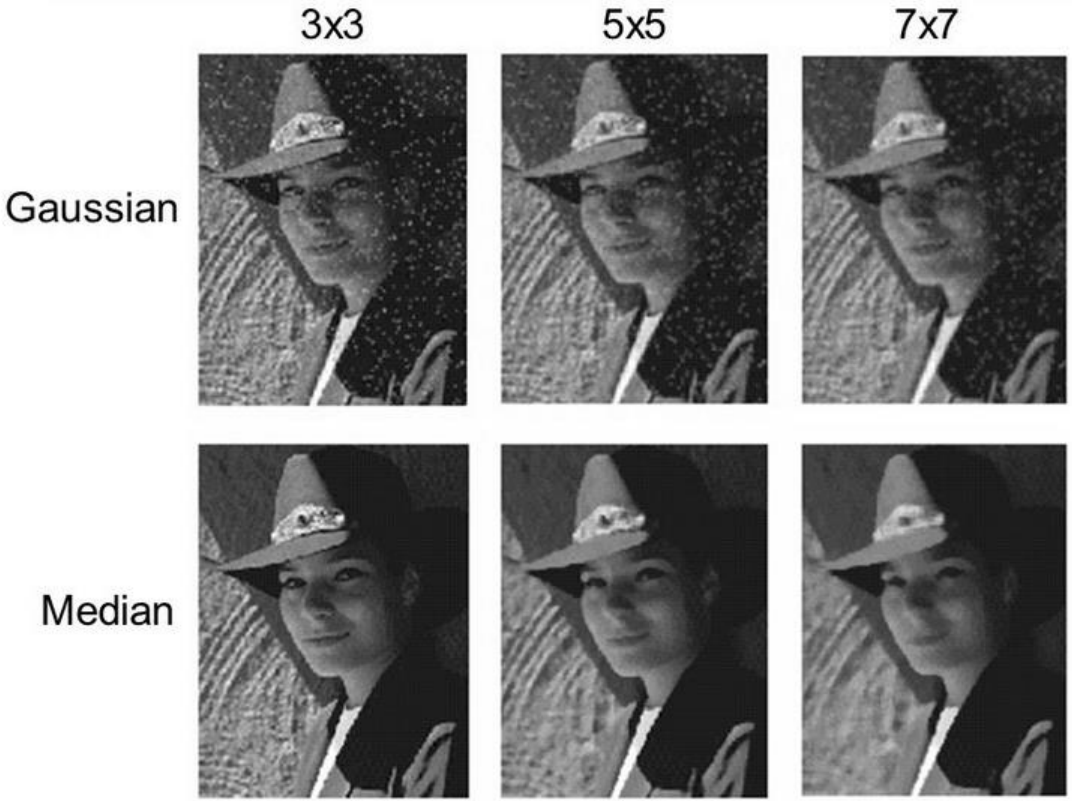
Sorted:

2, 2, 2,
2, 3, 4,
5, 5, 7

Median Filter Result



Gaussian vs. Median Filtering



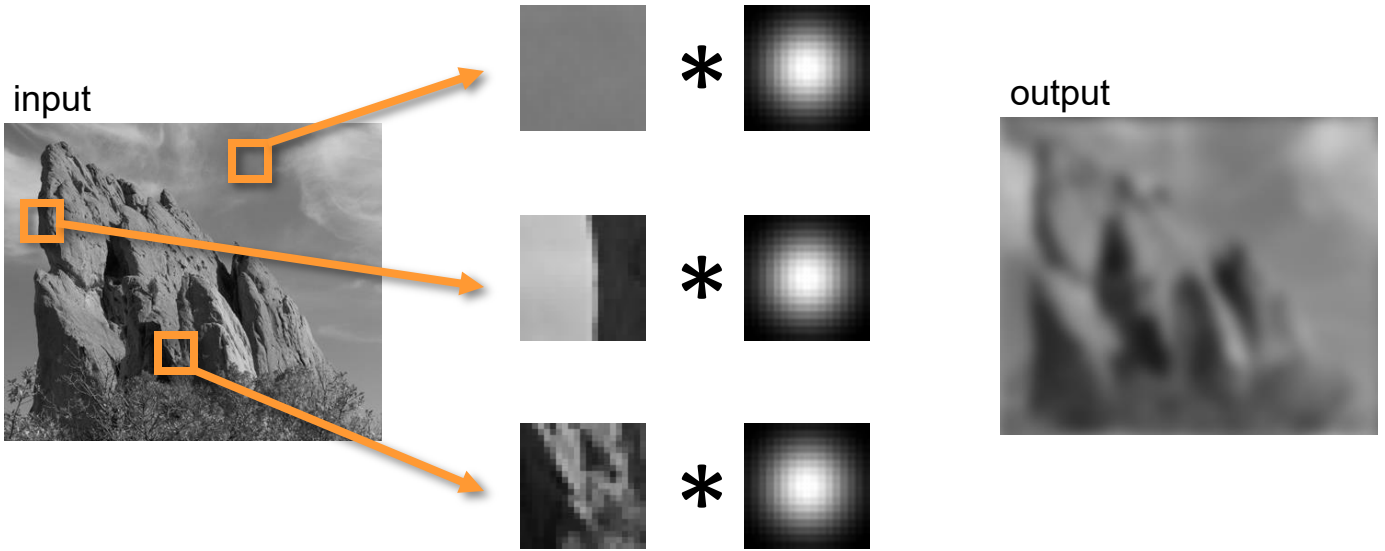
Bilateral Filter

Bilateral Filter (Tomasi et al.1998)

- Goal: Smooths regions while preserving edges

Bilateral Filter

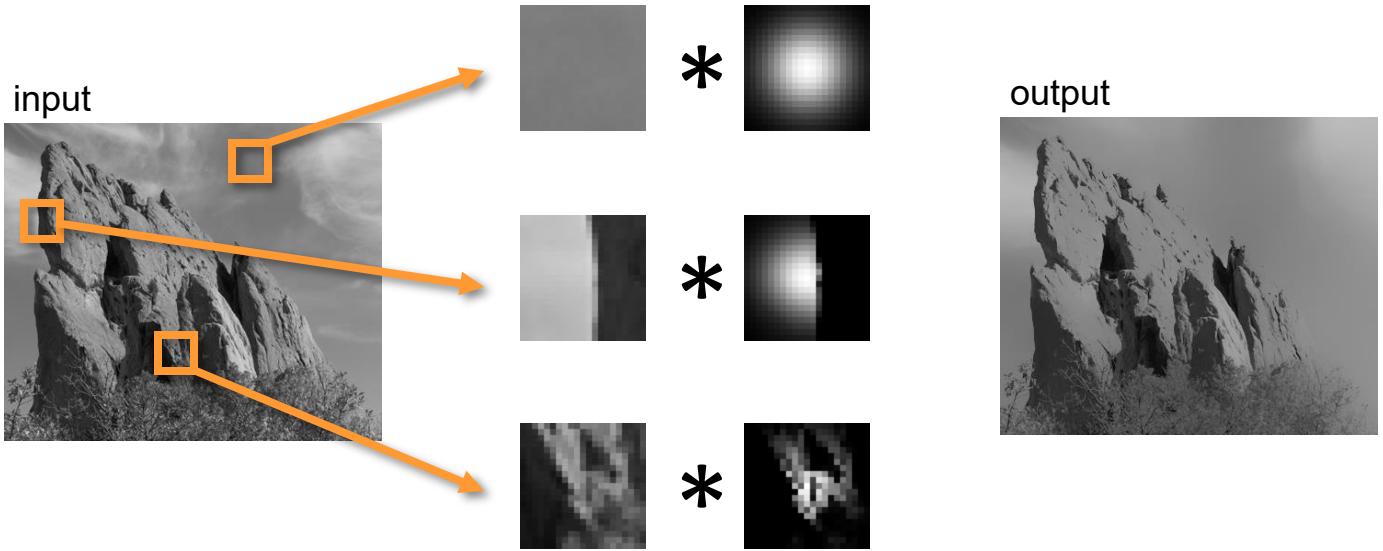
Motivation: In Gaussian filtering the kernel should blur less across image edges



Same Gaussian kernel everywhere.

Bilateral Filter

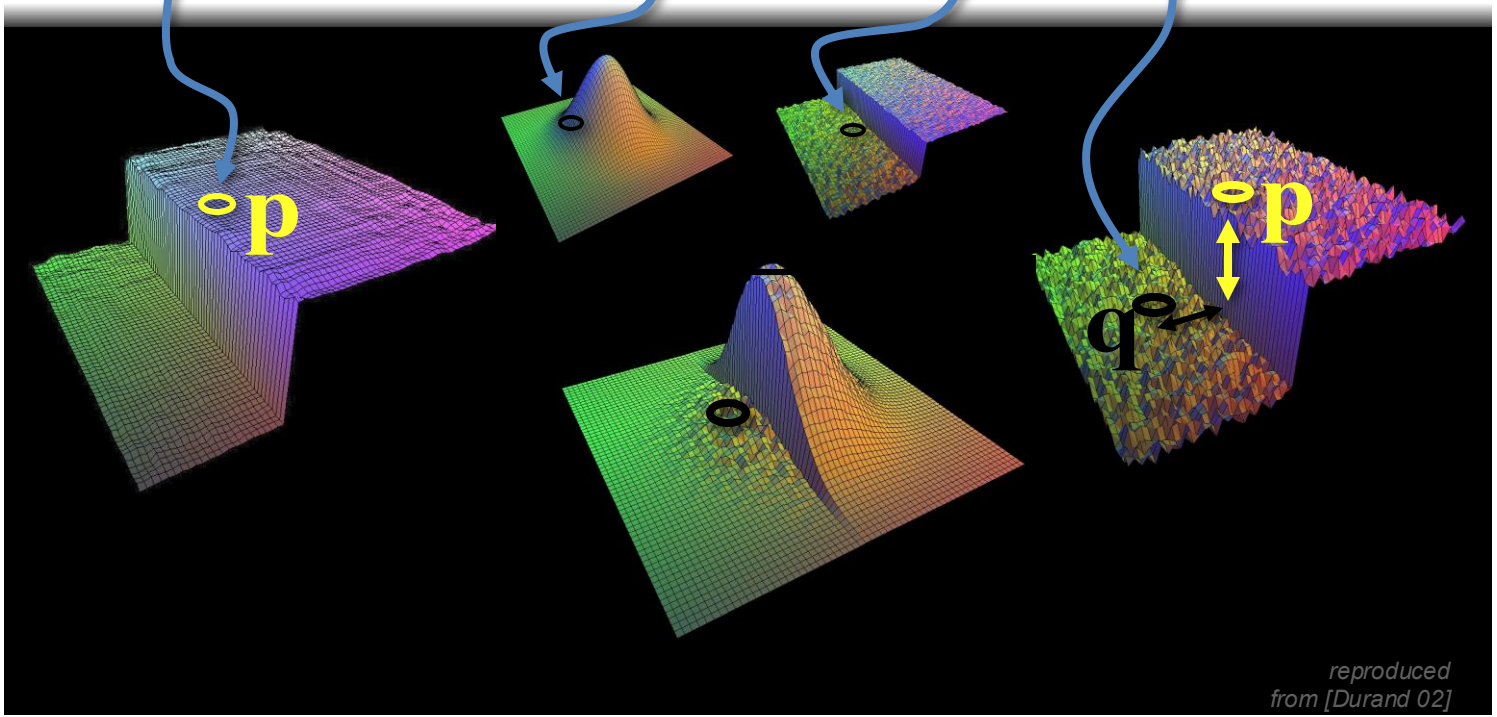
Motivation: No averaging across edges.



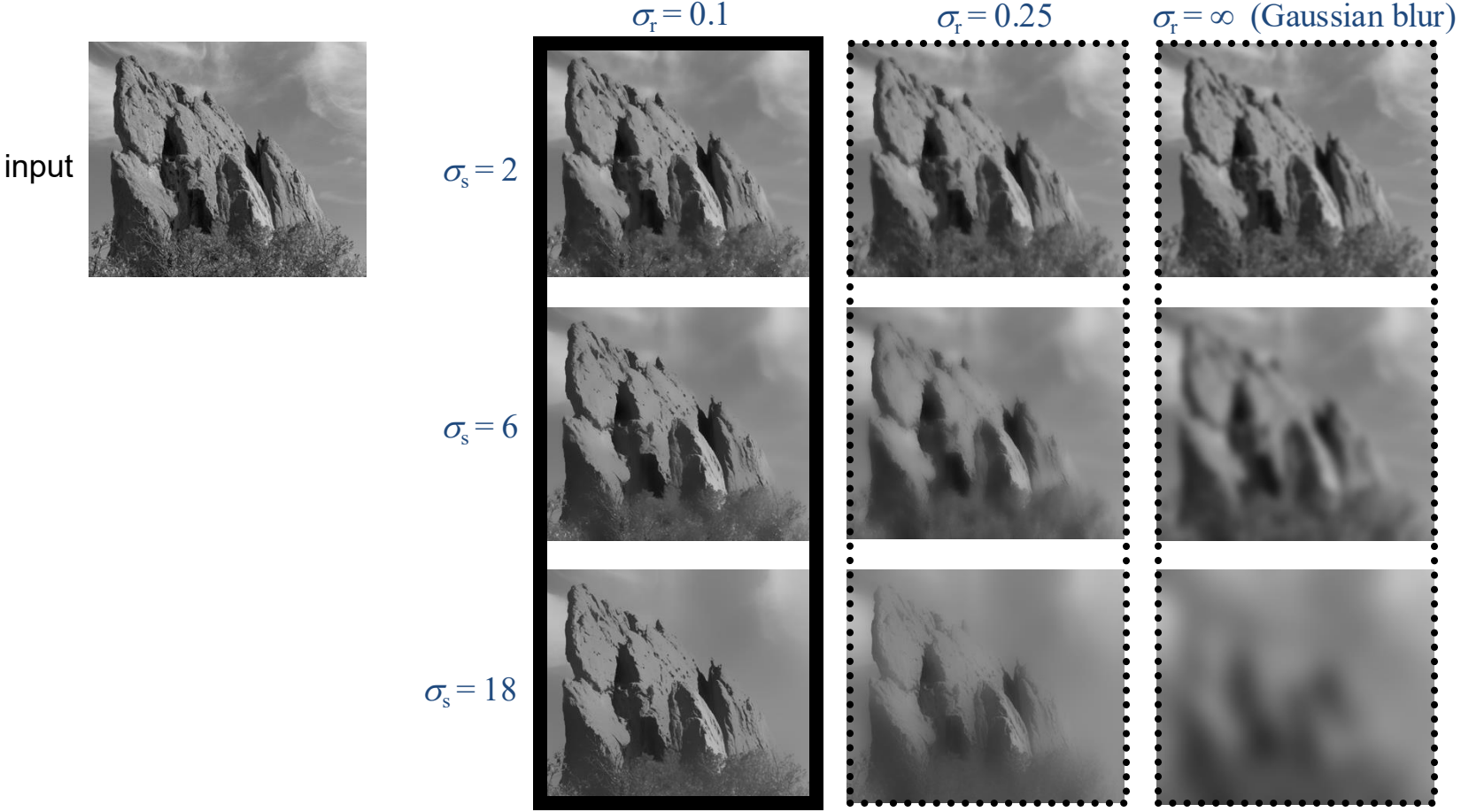
The kernel shape depends on the image content.

Bilateral Filter on a Height Field

$$BF[I]_{\mathbf{p}} = \frac{1}{W_{\mathbf{p}}} \sum_{\mathbf{q} \in S} G_{\sigma_s}(\|\mathbf{p}-\mathbf{q}\|)\, G_{\sigma_r}(|I_{\mathbf{p}}-I_{\mathbf{q}}|)\, I_{\mathbf{q}}$$



Varying the Space Parameter



Outline

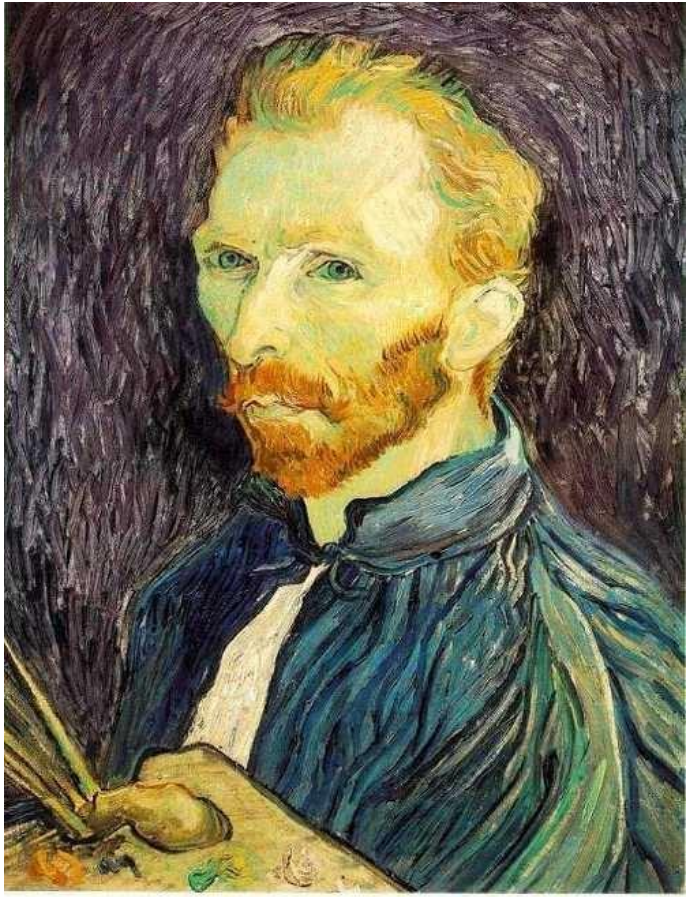
- **Neighborhood image processing**
  - Connected Component Analysis
  - Mathematical Morphology (Dilation, Erosion, Opening, Closing)
- **Linear Image Filtering**
  - Correlation and Convolution
  - Frequently used Filters
  - Gaussian Filter
- **Nonlinear Image Filtering**
  - Median Filter
  - Bilateral Filter
- **Frequency Domain Analysis**
  - Fourier Transform
  - Sampling and Aliasing

Let’s Enhance that Image!

[**Let's Enhance HD** - YouTube]

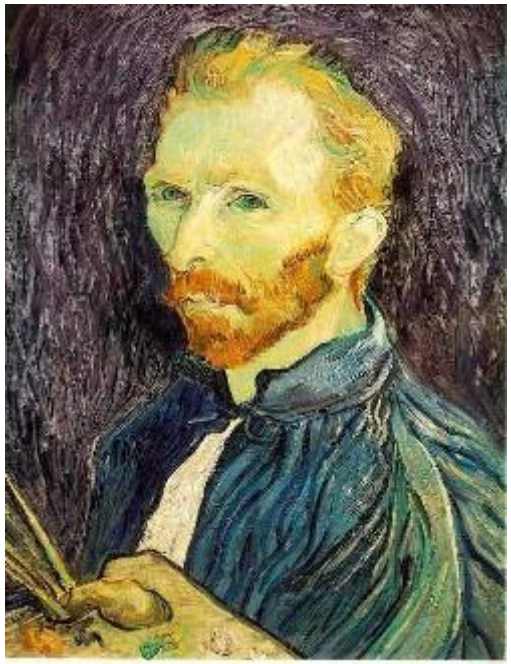
Image Scaling

This image is too big, can we generate a half-sized version?



[Slide credit: Steve Seitz]

Image sub-sampling



1/4

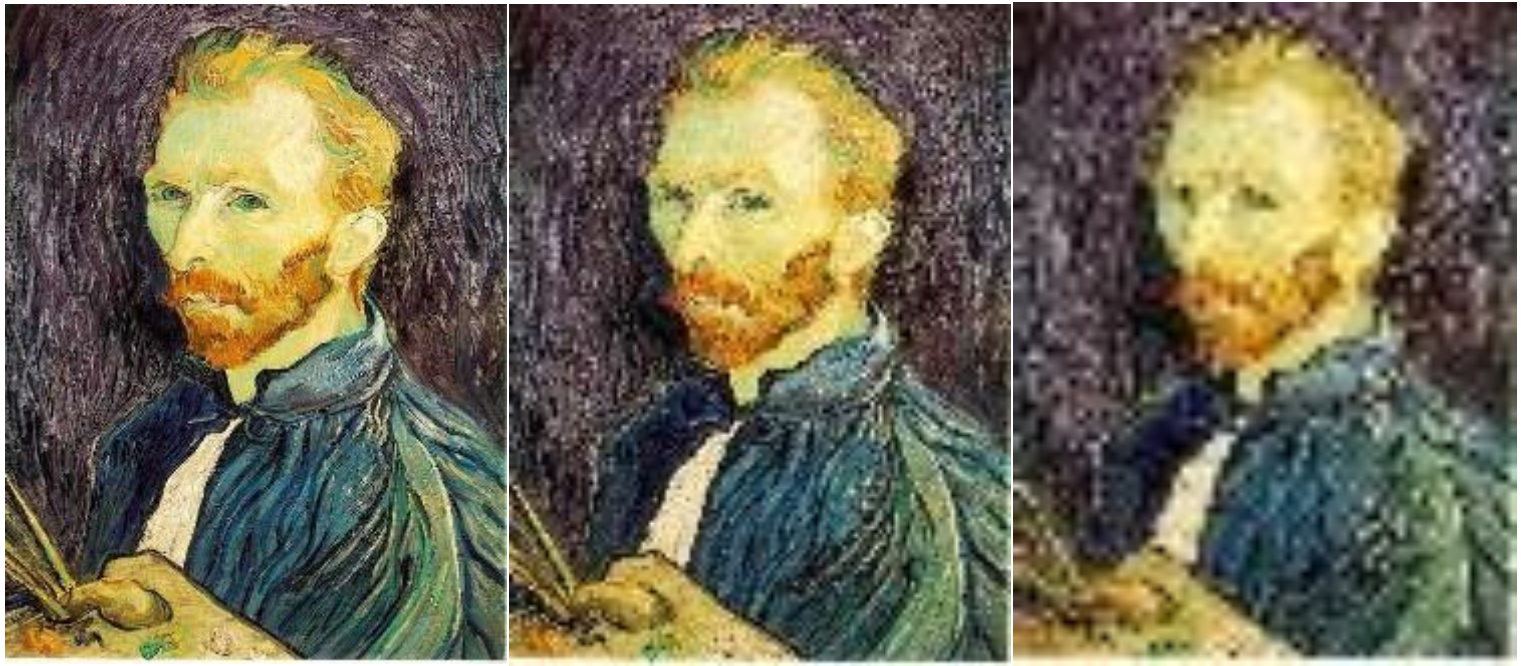
1/8

Throw away every other row and column to create a 1/2 size image

- called *image sub-sampling*

[Slide credit: Steve Seitz]

Image sub-sampling



1/2 1/4 (2x zoom) 1/8 (4x zoom)

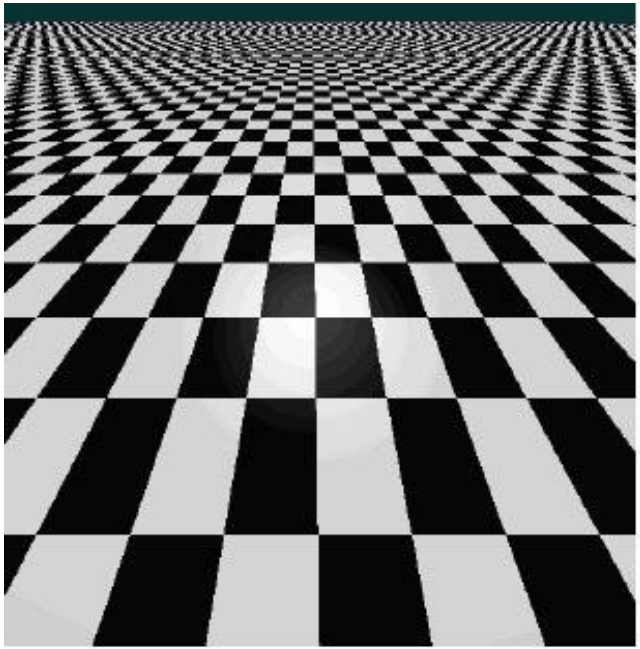
Why does this look so crufty?

[Slide credit: Steve Seitz]

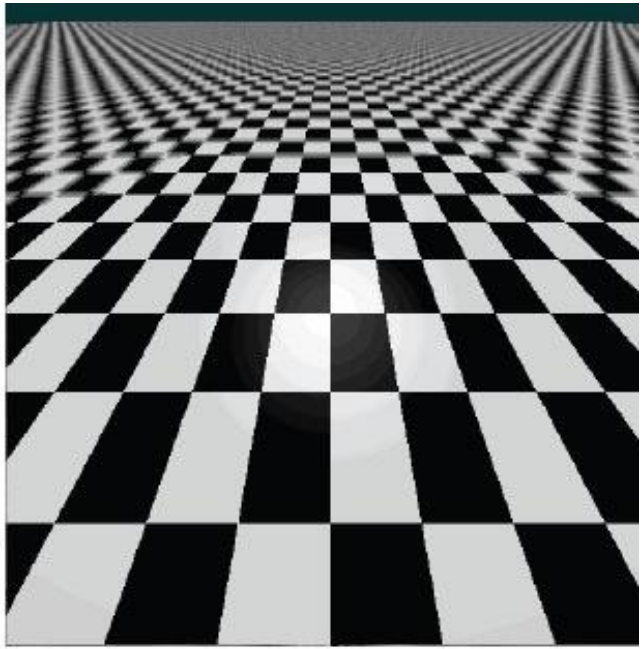
Image sub-sampling – another example

[Slide credit: Fredo Durand]

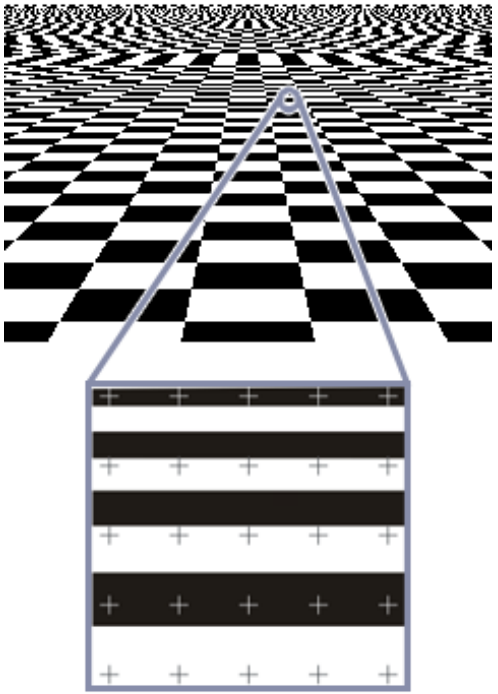
Image sub-sampling



aliasing effects

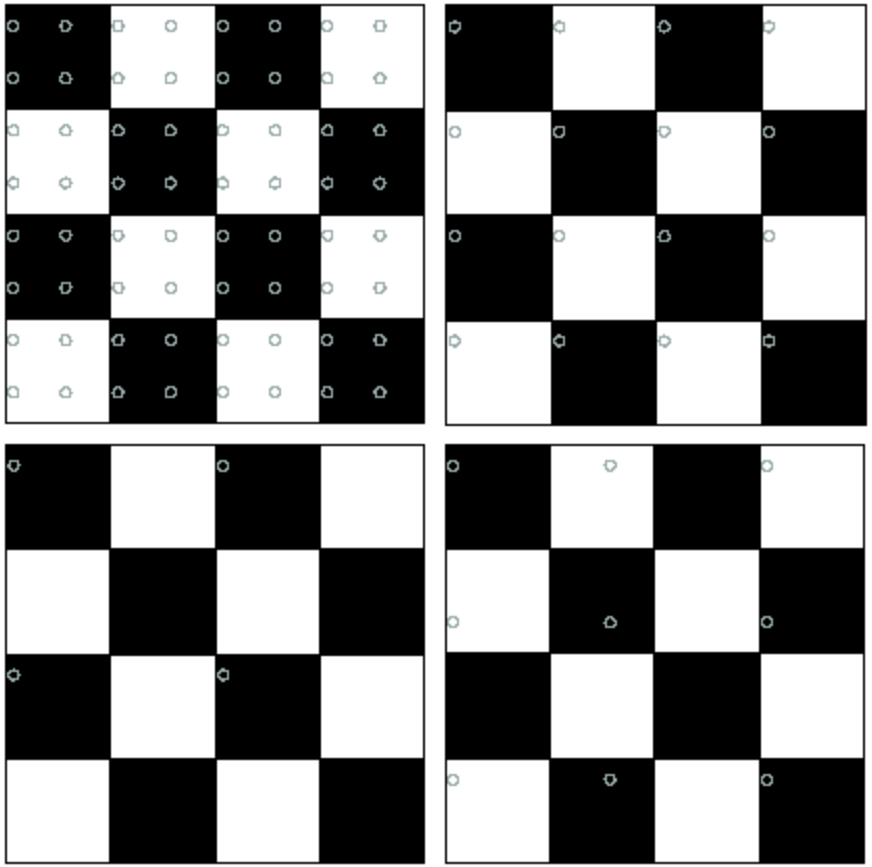


anti-aliasing by over-sampling



[Slide credit: Steve Seitz]

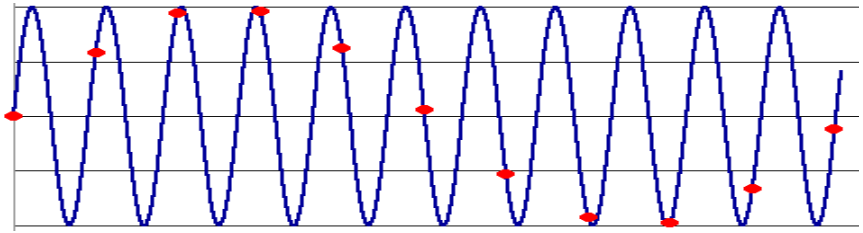
Nyquist limit – 2D example



Good sampling

Bad sampling

Aliasing



- Occurs when your sampling rate is not high enough to capture the amount of detail in your image
- Can give you the wrong signal/image—an *alias*
- To do sampling right, need to understand the structure of your signal/image
- To avoid aliasing:
  - sampling rate ≥ 2 * max frequency in the image
    - said another way: ≥ two samples per cycle
  - This minimum sampling rate is called the **Nyquist rate**

[Slide credit: L. Zhang]

Aliasing in Practice

Fourier Transform

Spatial domain x | **Frequency domain** u

Fourier Transform →

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$$

Fourier Analysis

← Inverse Fourier Transform

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{-i2\pi ux} du$$

Fourier Synthesis

$$e^{i\theta} = cos\theta + i\, sin\theta \qquad i = \sqrt{-1}$$

Note: This is the continuous case for 1D.

Jean-Baptiste Joseph Fourier
(1768-1830)

Fourier Transform

Fourier Transform is expressed with **complex** numbers.

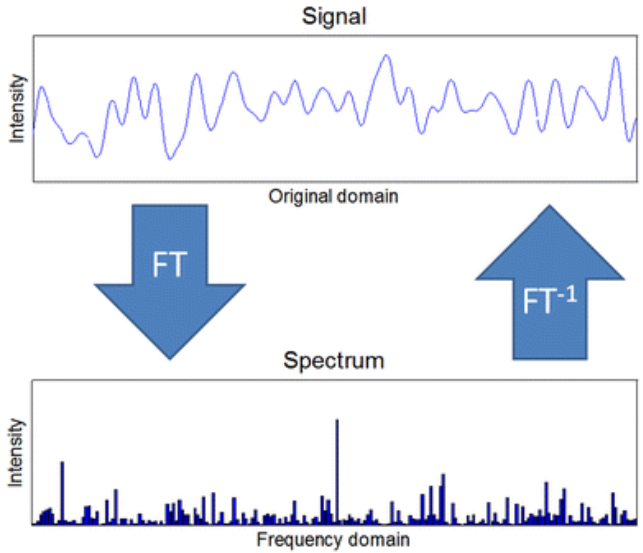
Here, $F(u)$ holds both the Amplitude and Phase of the sinusiod of frequency u.

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-i2\pi ux} dx$$

$$F(u) = \Re\{F(u)\} + i \cdot \Im\{F(u)\}$$

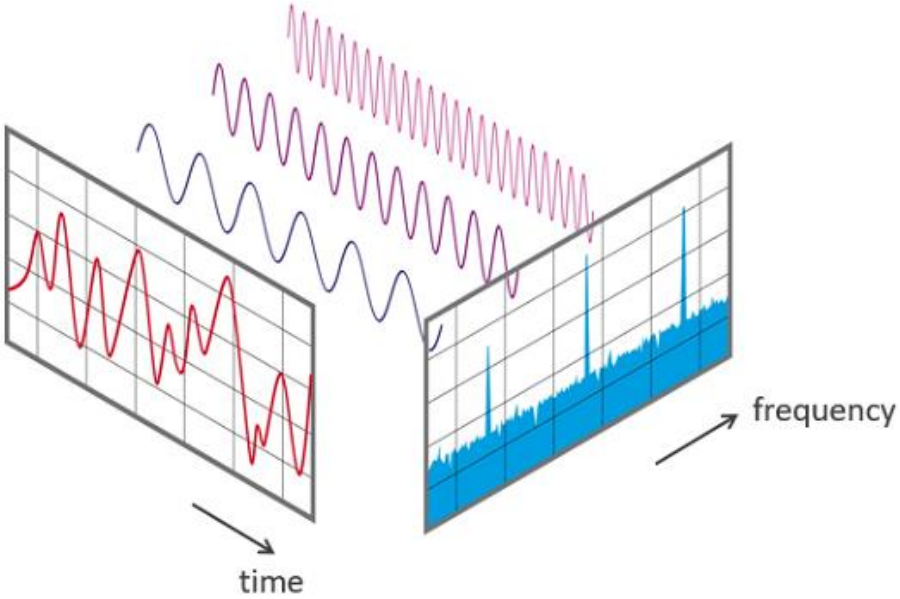
Amplitude: $A(u) = \sqrt{\Re\{F(u)\}^2 + \Im\{F(u)\}^2}$

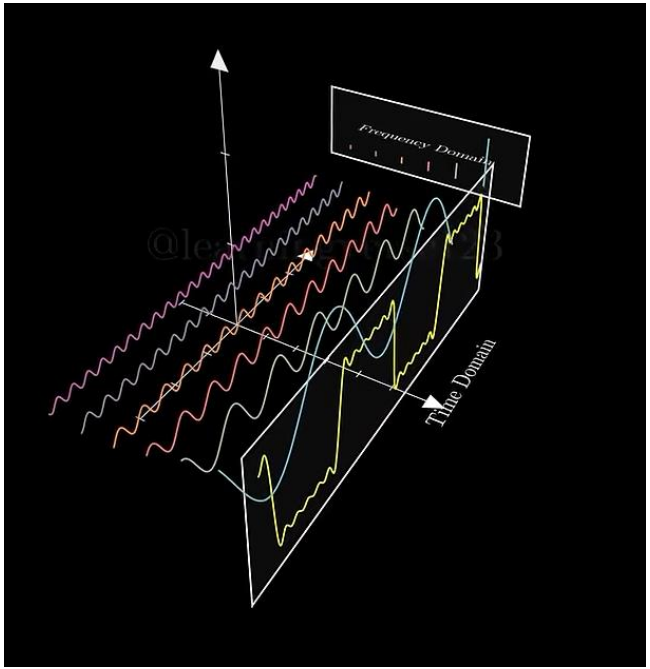
Phase: $\phi(u) = atan2(\Re\{F(u)\}, \Im\{F(u)\})$



Fourier Analysis

Decompose a given function into its frequencies and phases



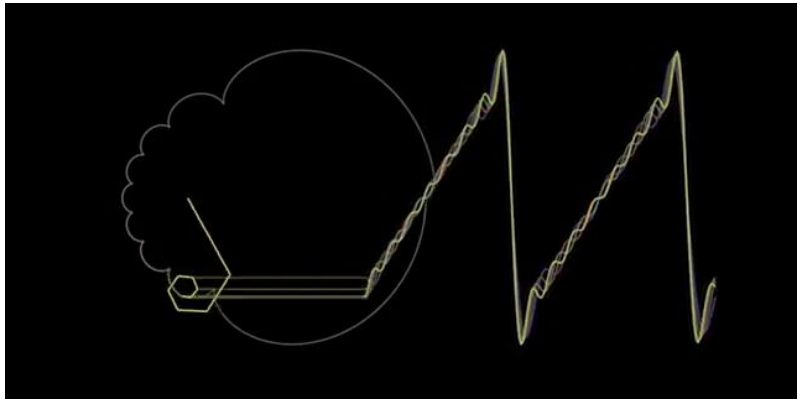


[learningverse123 - YouTube]

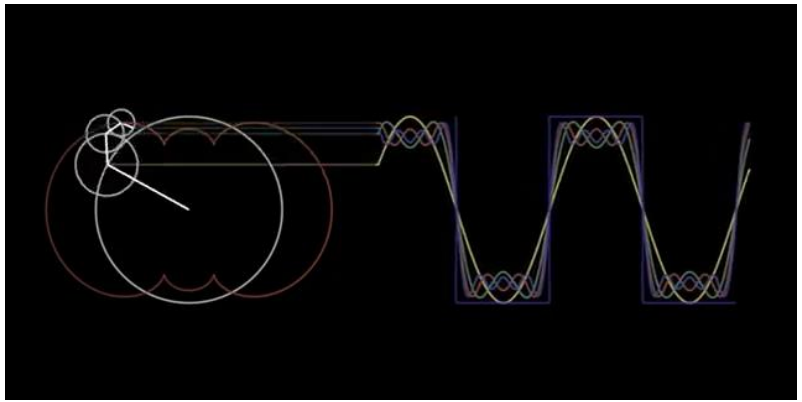
Fourier Synthesis

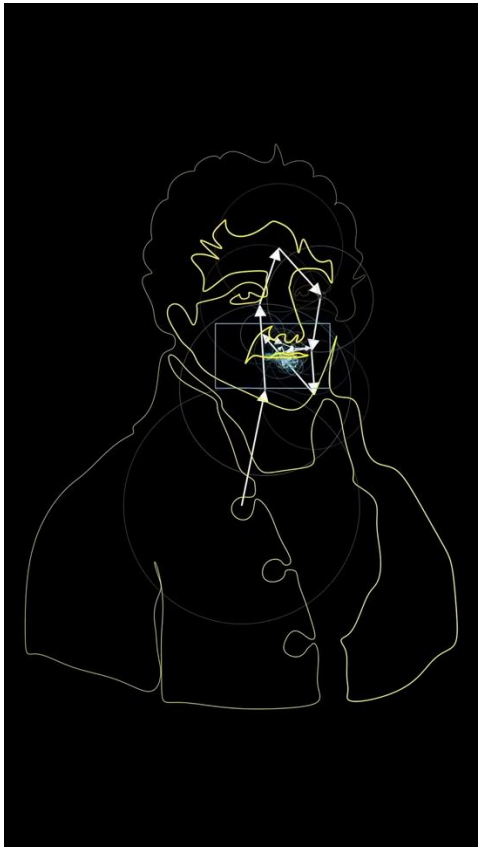
Compose a function from chosen frequencies and phases.

- Sawtooth:



- Square wave:





[3Blue1Brown - YouTube]

Fourier Transform

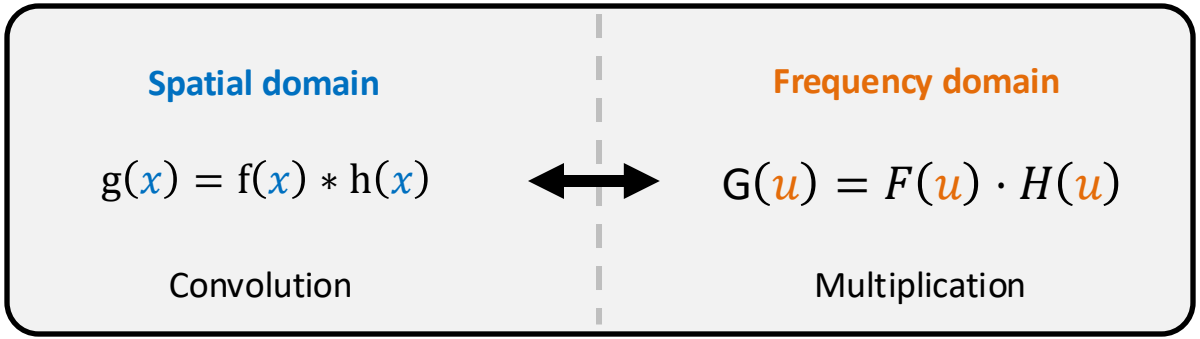
Important Facts

- The Fourier transform (FT) is linear
- There is an inverse FT
- The FT of a Gaussian is a Gaussian

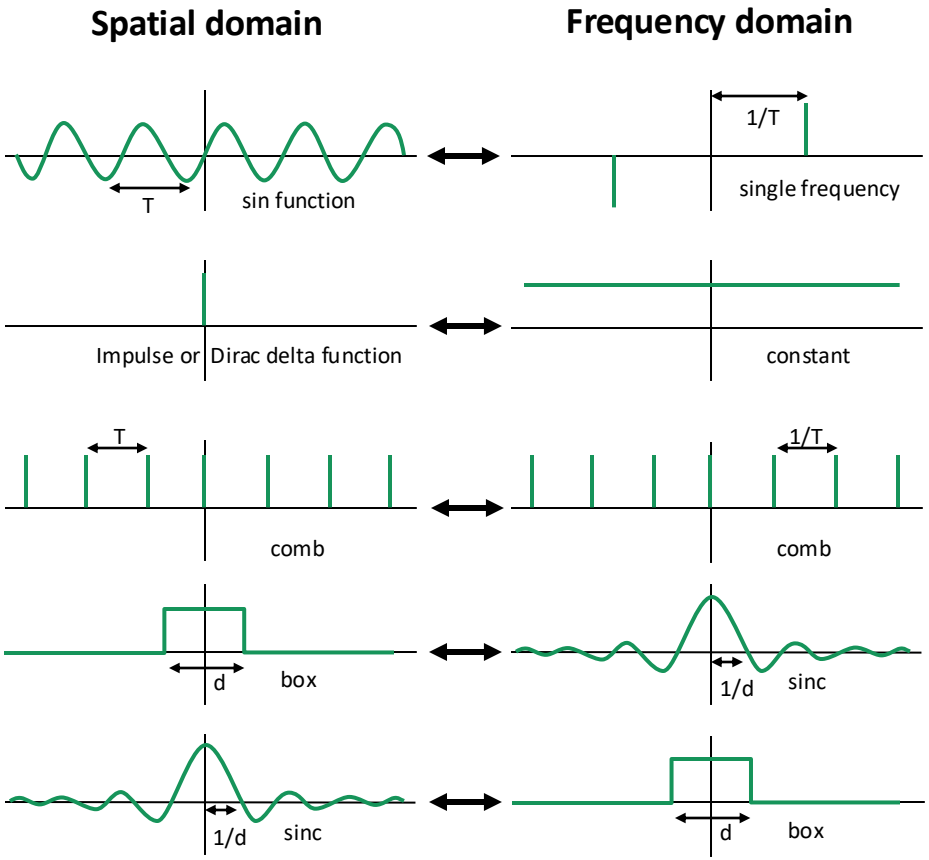
Convolution Theorem

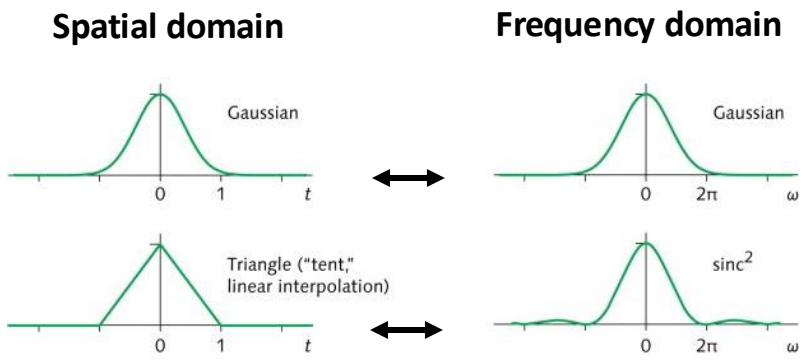
The convolution theorem

1. The Fourier transform of the convolution of two functions is the product of their Fourier transforms
2. The Fourier transform of the product of two functions is the convolution of the Fourier transforms



Fourier Transform of Important Functions





Dirac delta function:

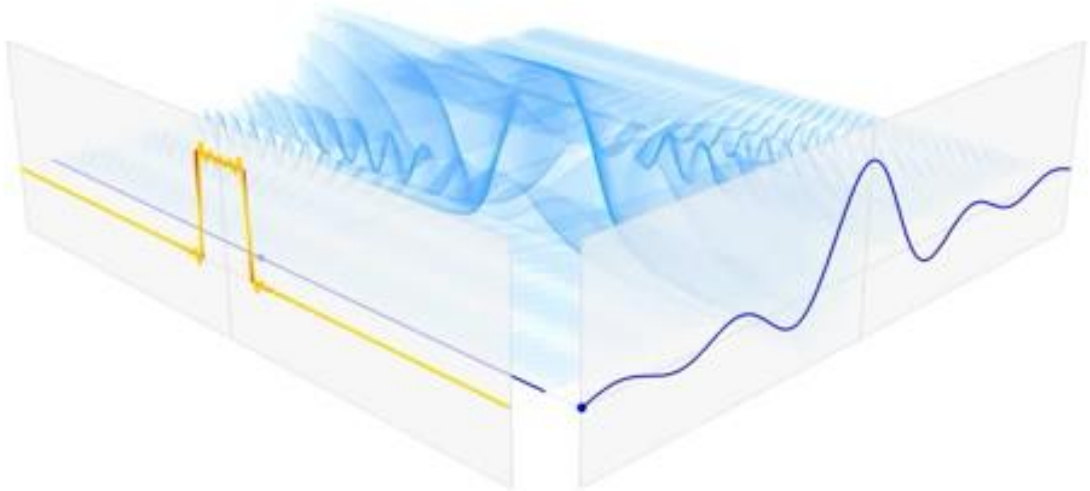
$$\delta(x) = \begin{cases} 0 & \text{if } x \neq 0 \\ \infty & \text{otherwise} \end{cases} \quad \text{with} \quad \int_{-\infty}^{\infty} \delta(x)\,dx = 1$$

Sinc function:

$$\text{sinc}(x) = \frac{\sin(x)}{x}$$

Fourier Transforms of sinc and rect Functions

$$\int_{-\infty}^{\infty} f(x)\, e^{-2\pi i x \xi}\, dx$$



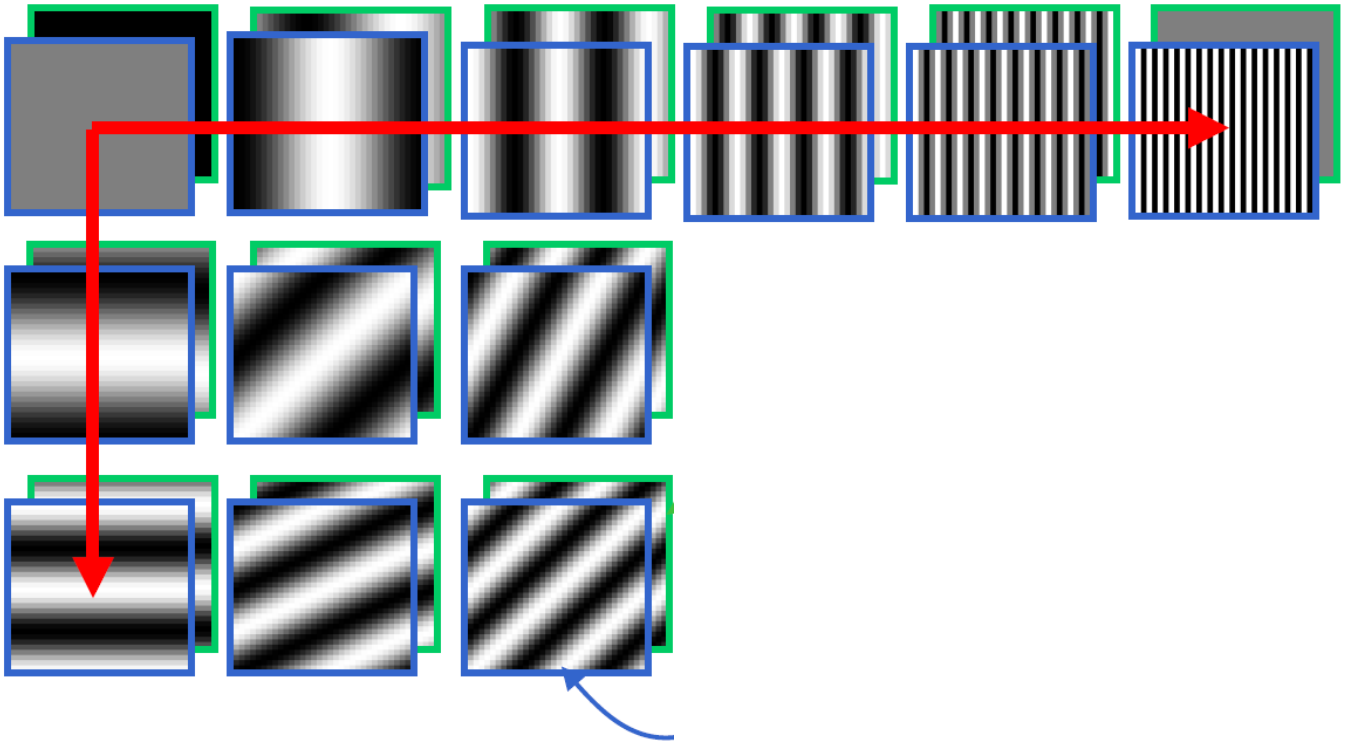
The Discrete Fourier Transform (2D)

The *Discrete Fourier Transform* of $f(x, y)$, for x = 0, 1, 2…M-1 and y = 0,1,2…N-1, denoted by $F(u, v)$, is given by the equation:

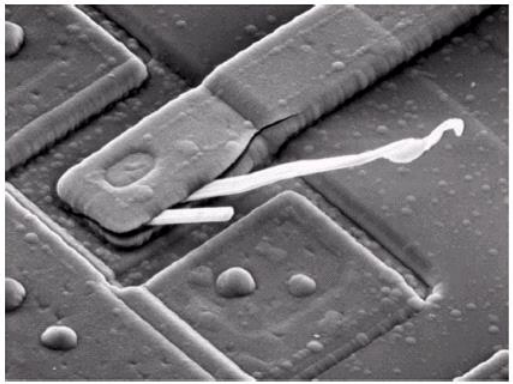
$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-j2\pi(ux/M + vy/N)}$$

for u = 0, 1, 2…M-1 and v = 0, 1, 2…N-1.

Represent Images by a set of Fourier basis functions

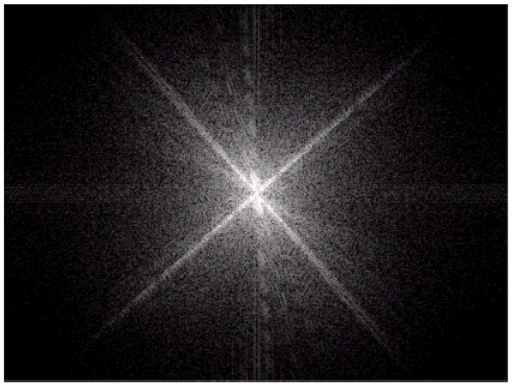


DFT & Images (cont…)



Scanning electron microscope image of an integrated circuit magnified ~2500 times

DFT

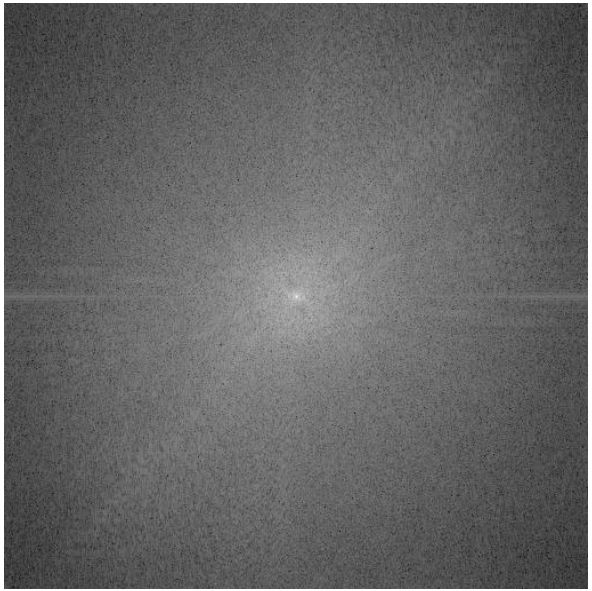


Fourier spectrum of the image

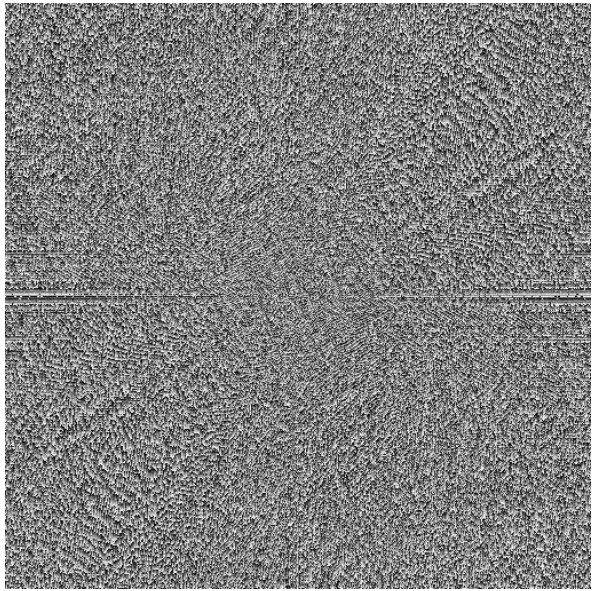
Fourier Space Visualization

Input

Magnitude



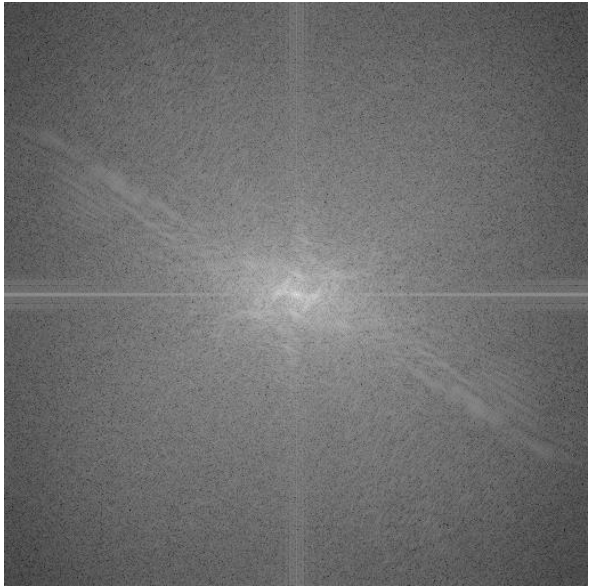
Phase



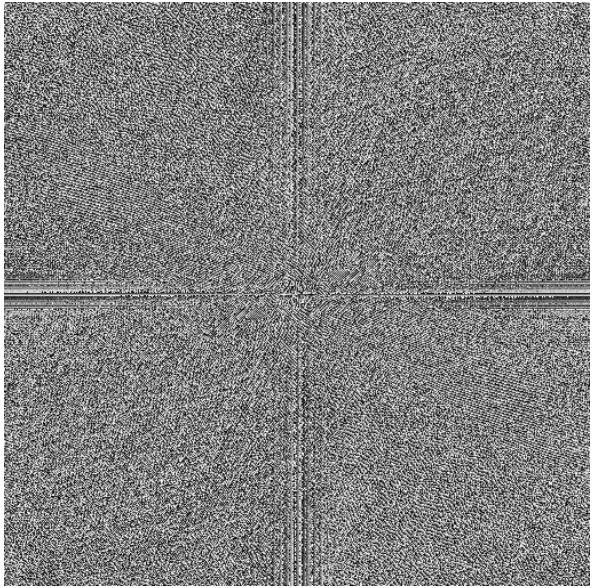
Fourier Space Visualization

Input

Magnitude



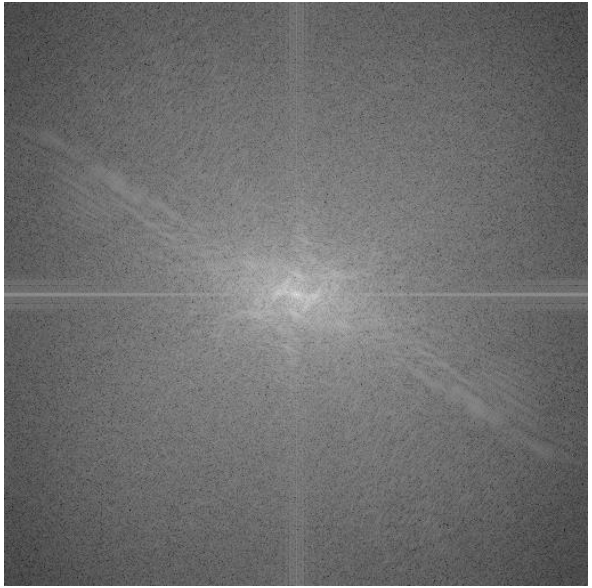
Phase



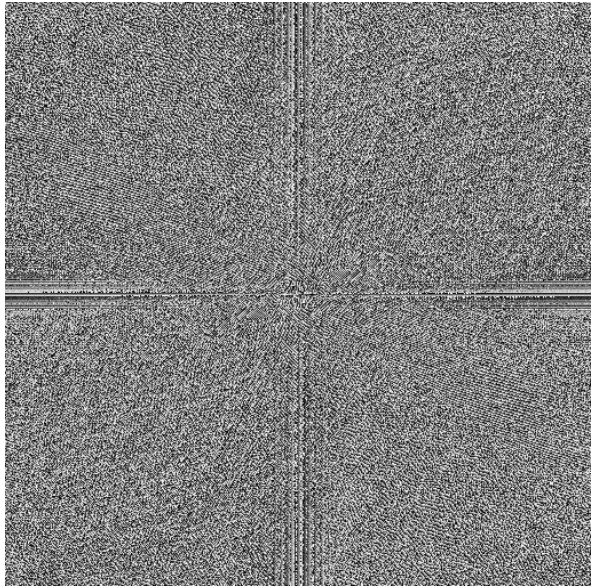
Fourier Space Visualization

Input

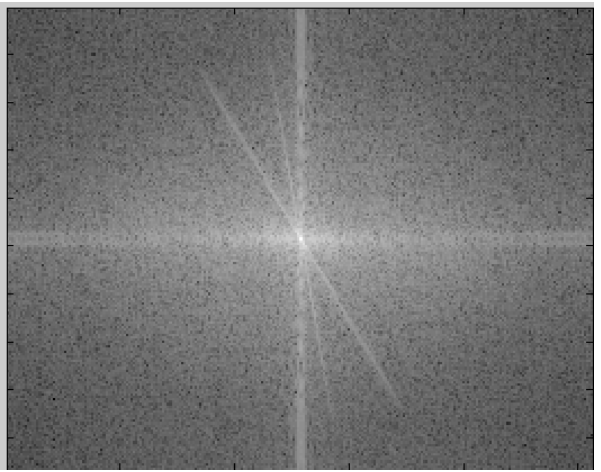
Magnitude



Phase



Example



Low Pass Filter

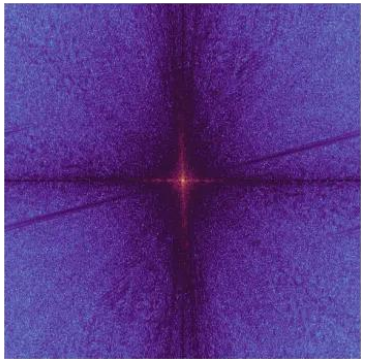
spatial domain frequency domain

$f(x,y) \longrightarrow F(u,v)$

filter

$G(u,v) = F(u,v) \bullet H(u,v)$

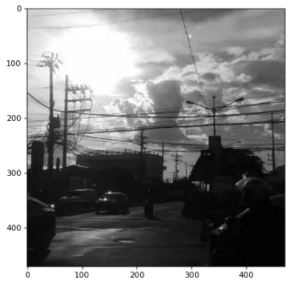
$g(x,y) \longleftarrow G(u,v)$



Low Pass Filter

spatial domain

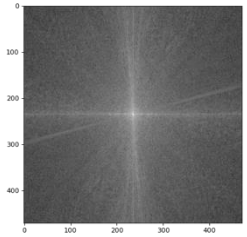
f(x,y)



Dark horizontal lines

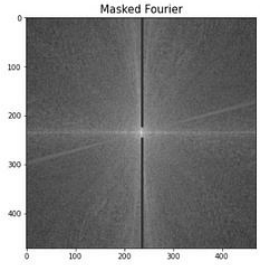
Frequency domain

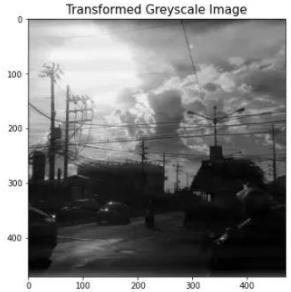
F(u,v)



filter

$G(u,v) = F(u,v) \bullet H(u,v)$





Thinner horizontal lines

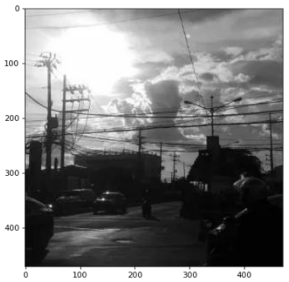
g(x,y)

G(u,v)

Low Pass Filter

spatial domain

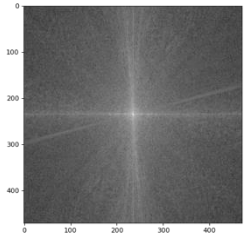
f(x,y)



Dark vertical poles

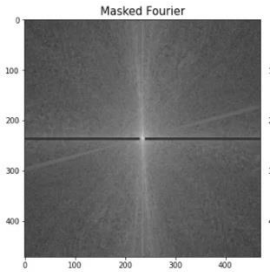
Frequency domain

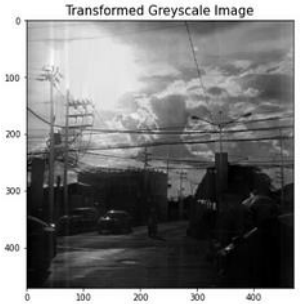
F(u,v)



filter

G(u,v) = F(u,v) • H(u,v)



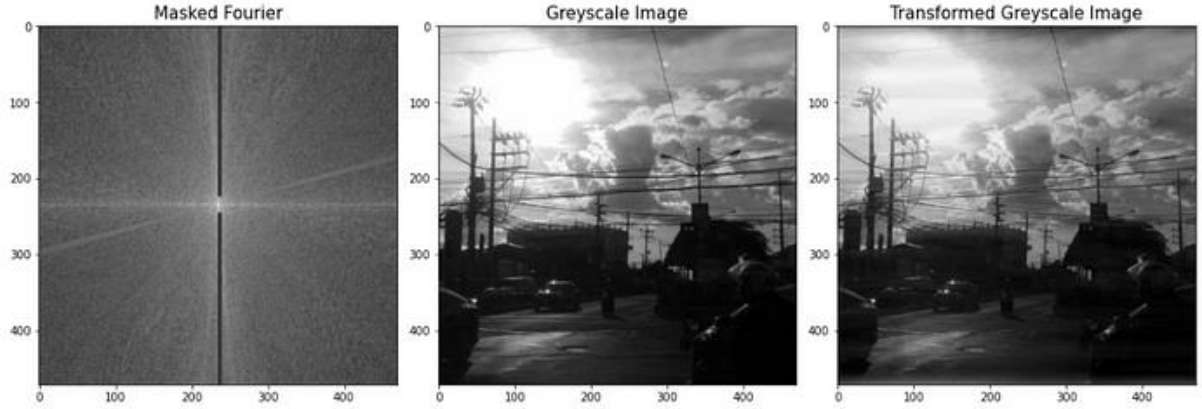


Lighter vertical poles

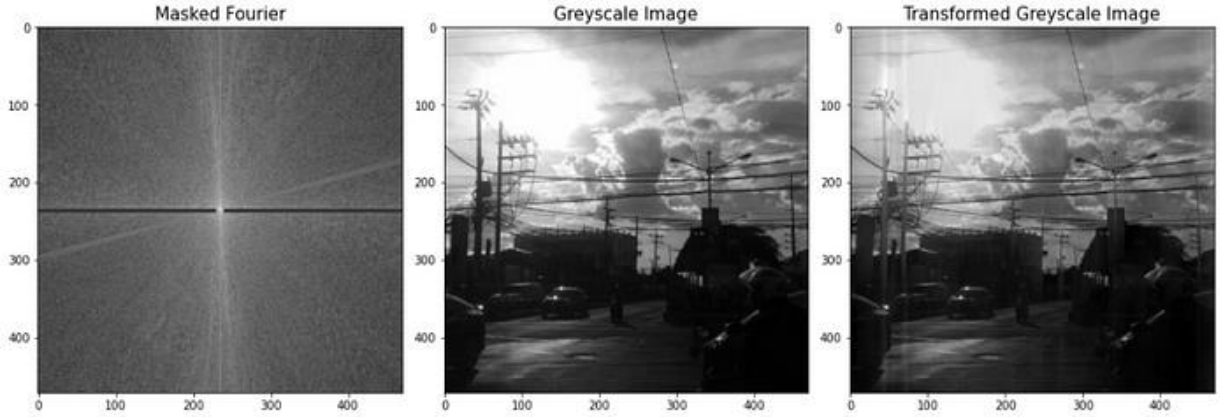
g(x,y)

G(u,v)

Low Pass Filter

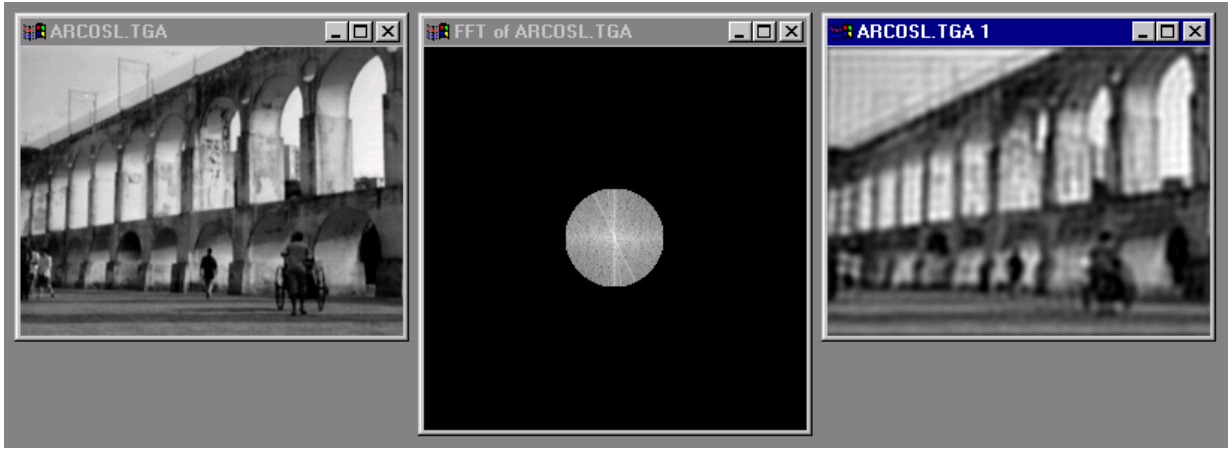


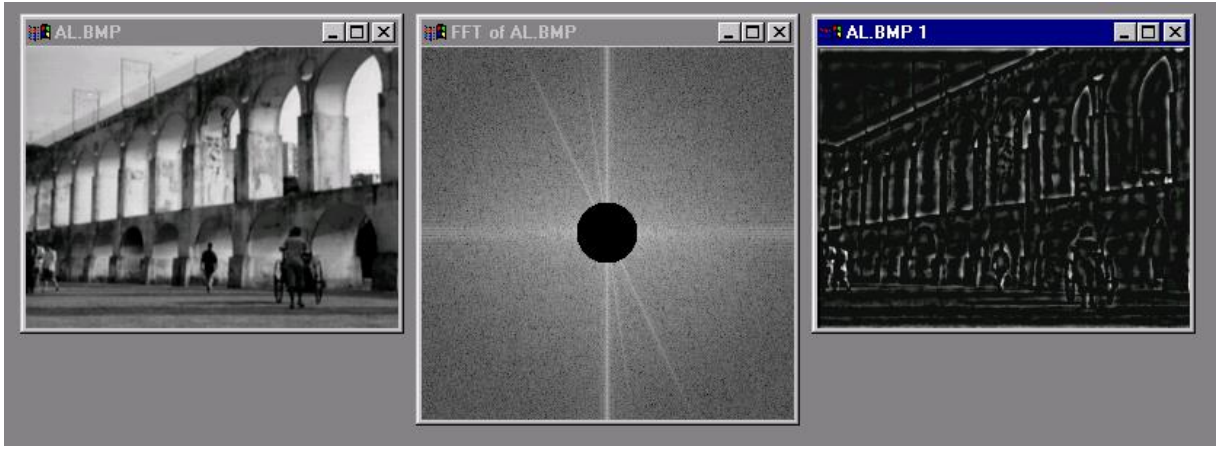
Thinner horizontal lines



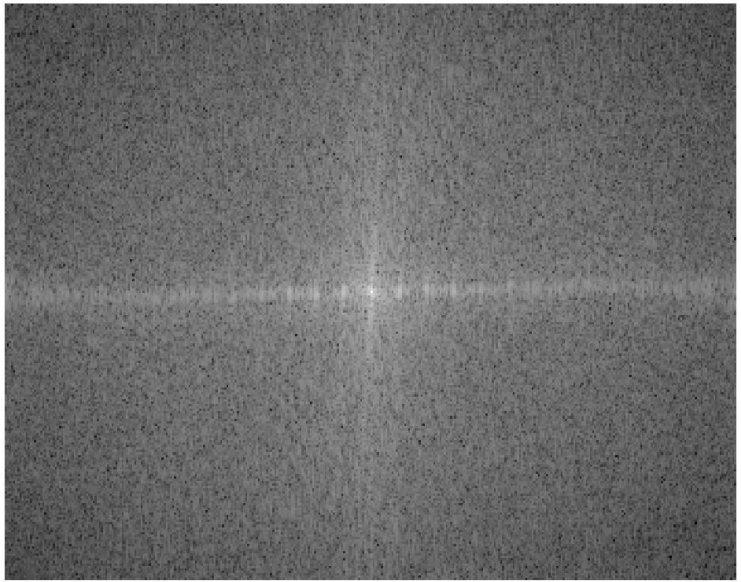
Lighter vertical poles

Low and High Pass filtering

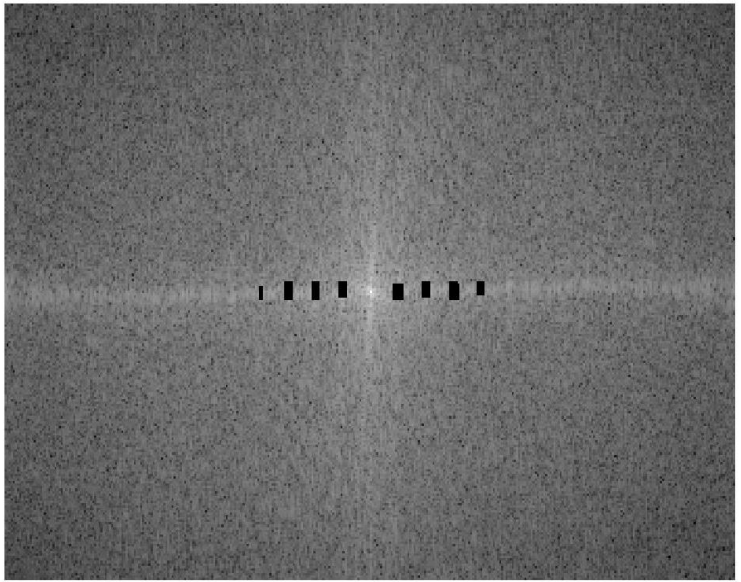




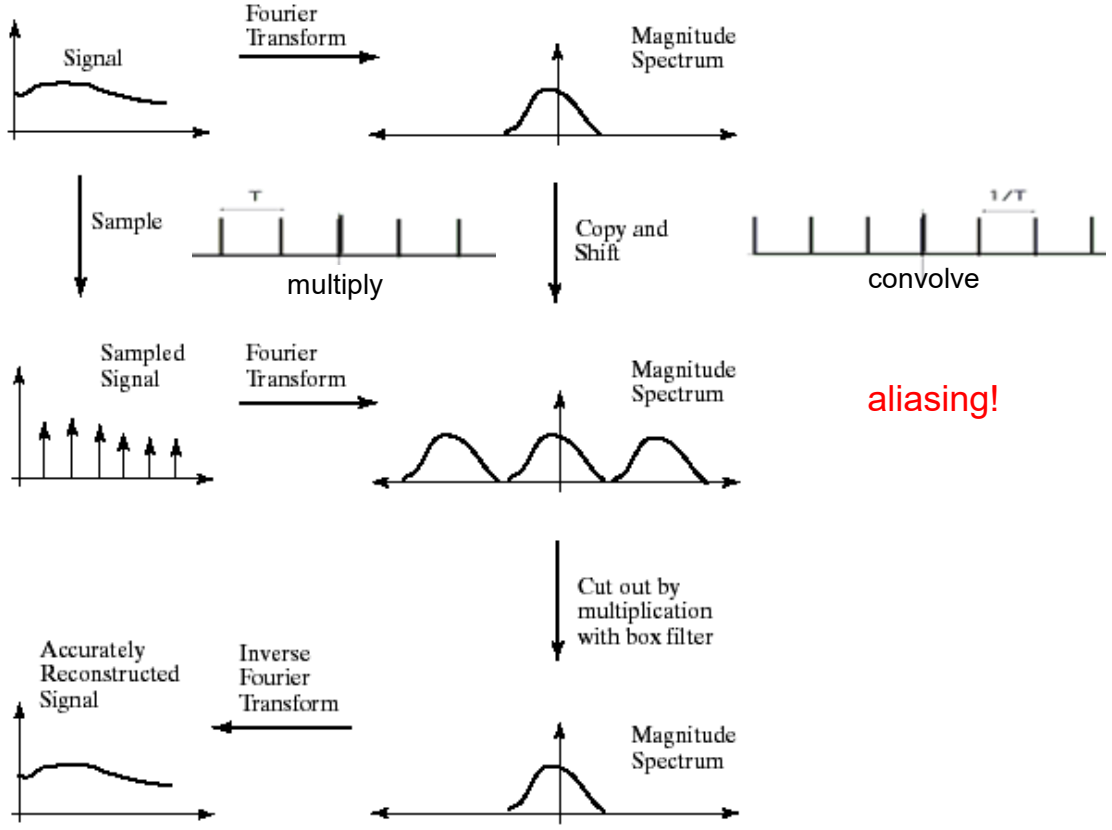
Frequency Filtering



Frequency Filtering

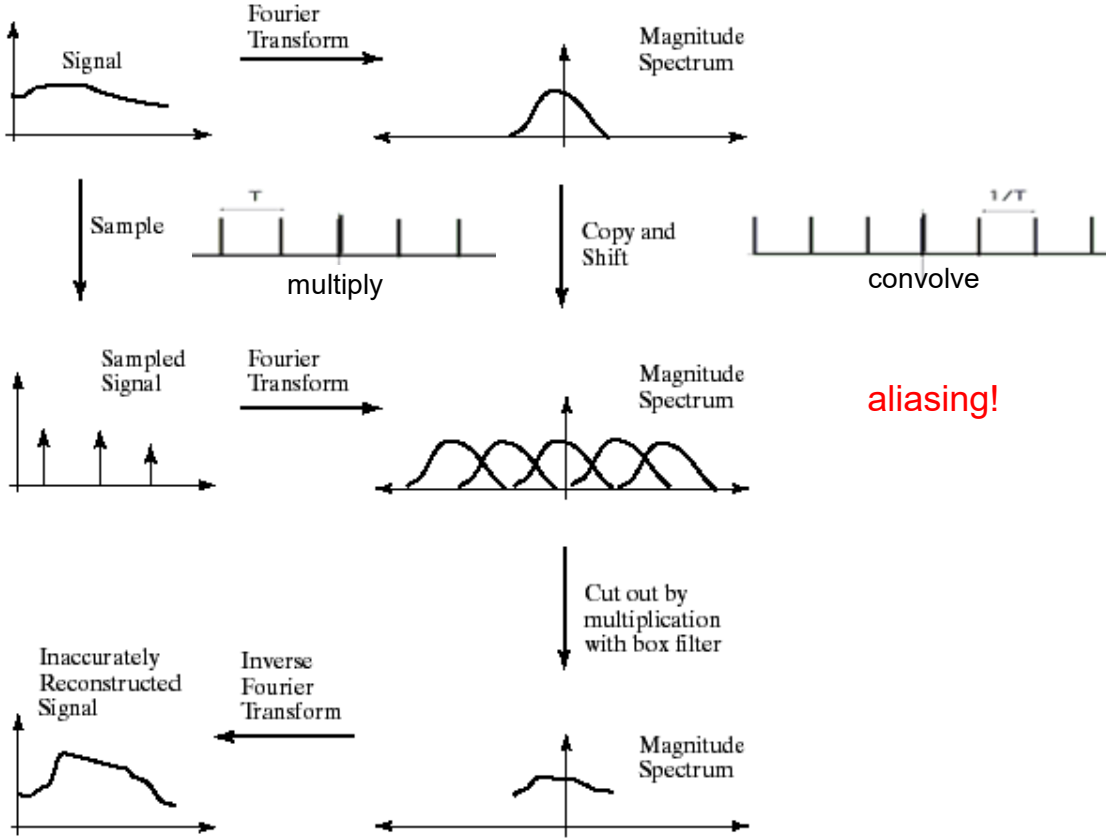


Aliasing



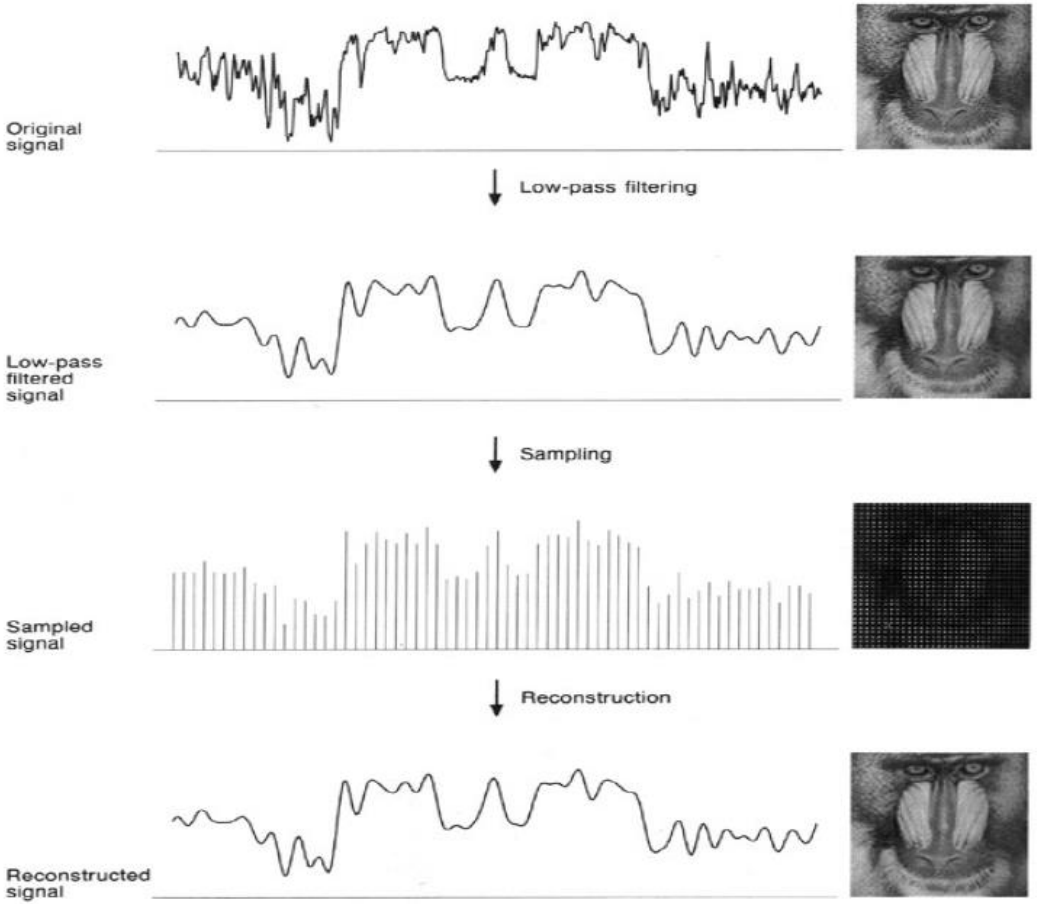
154

Aliasing



155

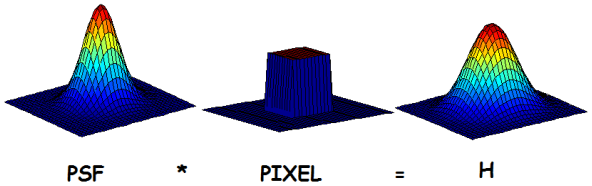
Proper sampling



[Slide credit: Dani Lischinski]

Application: Spatial super-resolution

- lens+pixel=low-pass filter (desired to avoid aliasing)



- Low-res images = D*H*G*(desired high-res image)
  - D: decimate, H:lens+pixel, G: Geometric warp
- Simplified case for translation: LR=(D*G)*(H*HR)
  - G is shift-invariant and commutes with H
  - First compute H*HR, then deconvolve HR with H
- Super-resolution needs to restore attenuated frequencies
  - Many images improve S/N ratio (~sqrt(n)), which helps
  - Eventually Gaussian's double exponential always dominates

Additional Resources

- Convolution
  - 3Blue1Brown: YouTube
- Fourier Transform
  - 3Blue1Brown: YouTube
  - Shree Nayar: YouTube, YouTube

Disclaimer

Many of the slides used here are obtained from online resources (including many open lecture materials) without appropriate acknowledgement. They are used here for the sole purpose of classroom teaching. All the credit and all the copyrights belong to the original authors. You should not copy it, redistribute it, put it online, or use it for any other purposes than for this course.