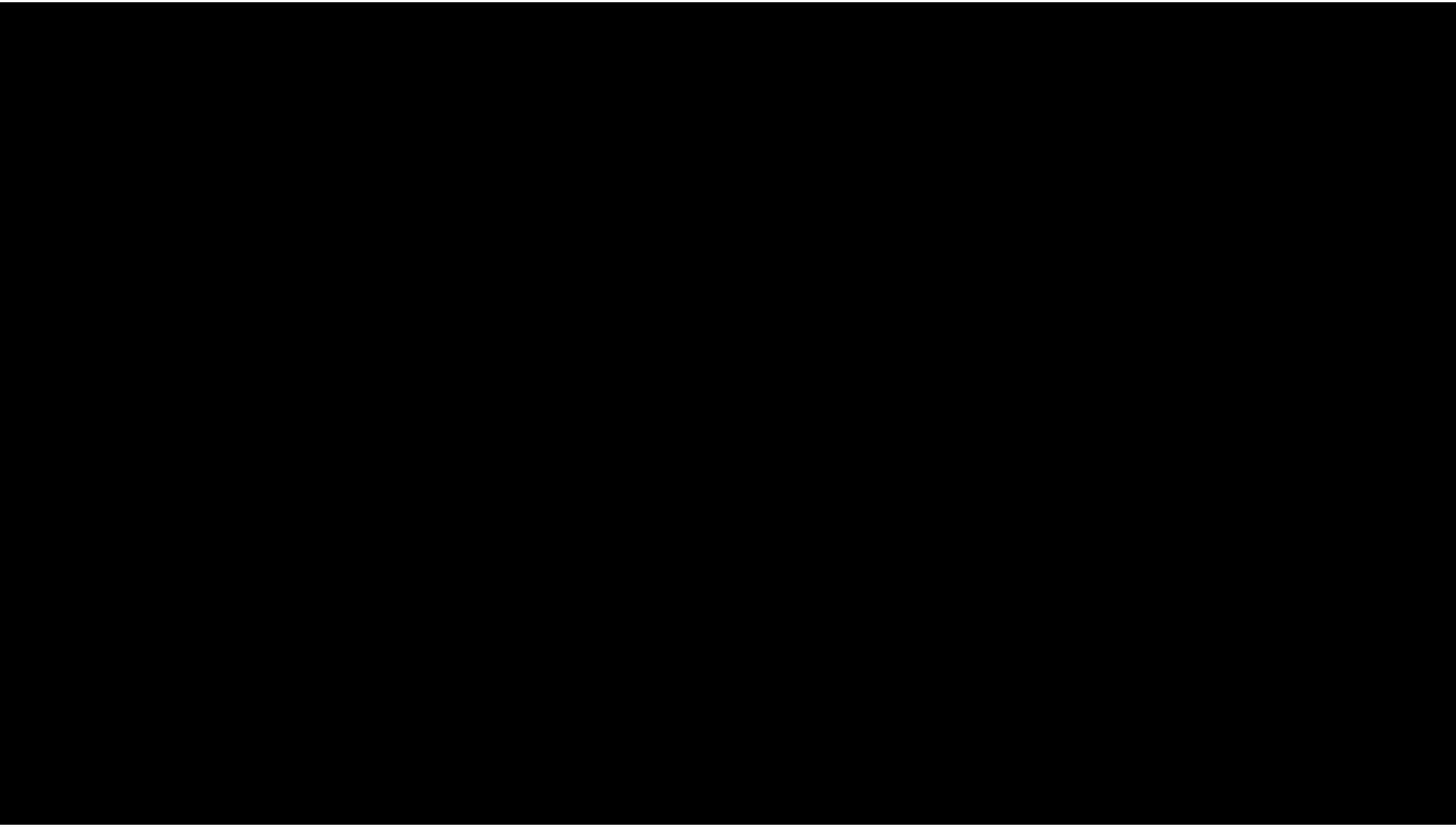
EE XXX
Design of Intelligent SoC Robot
http://ssl.kaist.ac.kr/class/18fall/socrobotdesign

Hoi-Jun Yoo
KAIST

Computer Vision for Robot



Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
2. *Basic of Image*
3. *Image Processing Algorithm*

-Practical Part-

1. *Practical Algorithms for "Robot War"*

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
   : Why does robot use CV?
2. *Basic of Image*
3. *Image Processing Algorithm*

-Practical Part-

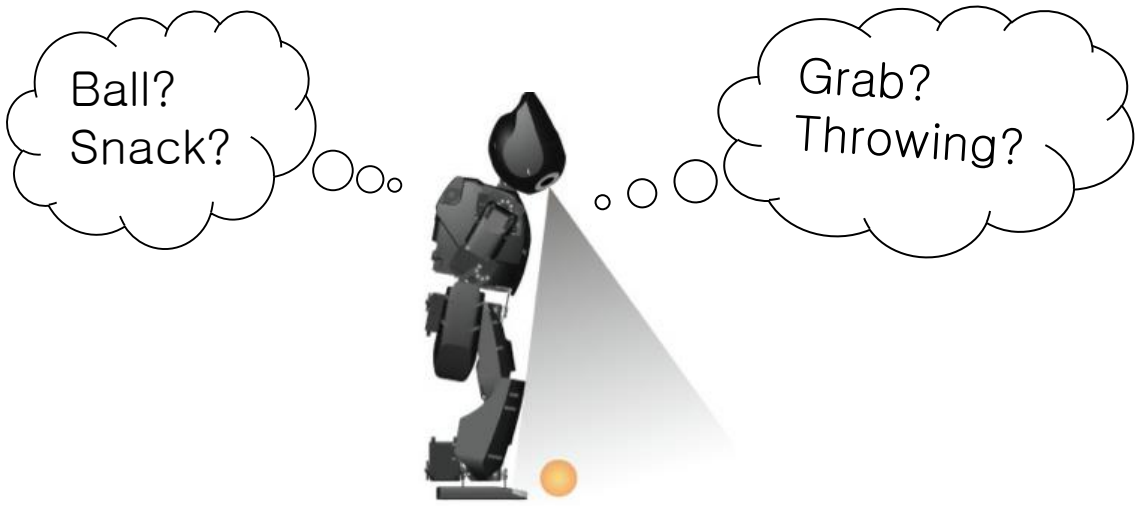
1. *Practical Algorithms for "Robot War"*

Why Computer Vision (CV) is required for Robot?

- For automatic understanding of images and video
- Types of CV algorithms for robot
  - **Image Processing** : Obtaining the image information from digital images
  - **Image Recognition** : Identifying the types of and numbers of objects from an image

Today's lecture

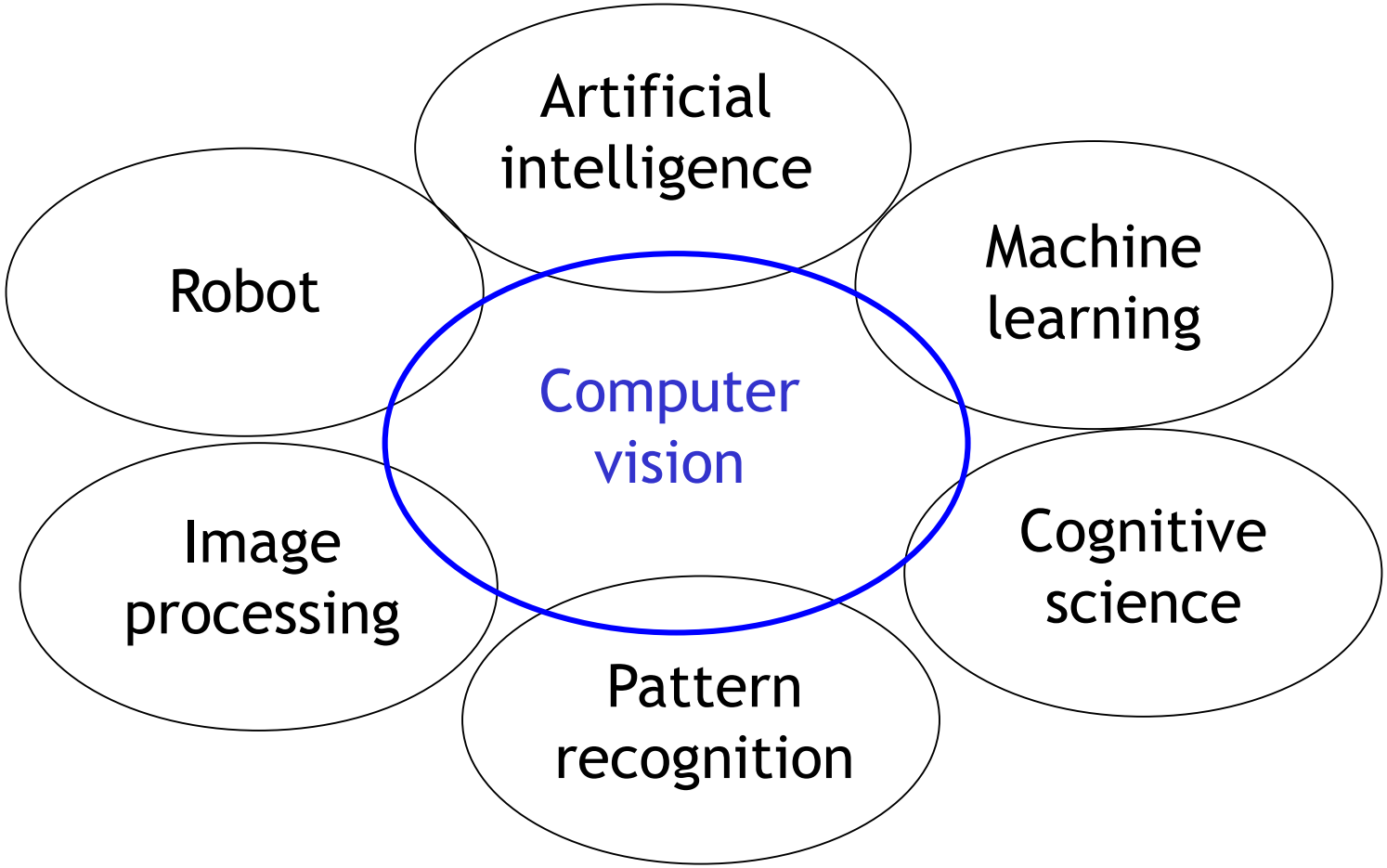
Next week's lecture



What is computer vision?

- Automatic understanding of images and video
  - Properties of the 3D world from visual data
    (measurement and modeling)
  - Algorithms and representations to recognize objects, people, scenes, and activities.
    (perception and interpretation)

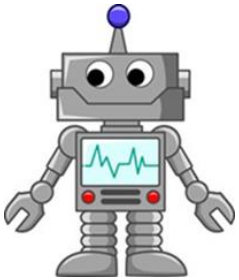
Related disciplines



Computer Vision and PR

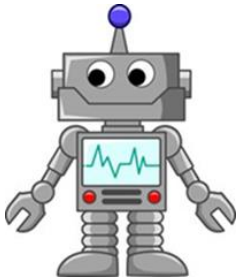
Slide by Fei-Fei, Fergus

Detection: are there people?



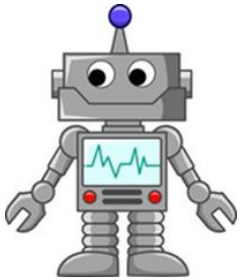
Slide by Fei-Fei, Fergus

Verification: is that a lamp?



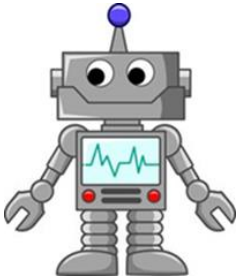
Slide by Fei-Fei, Fergus

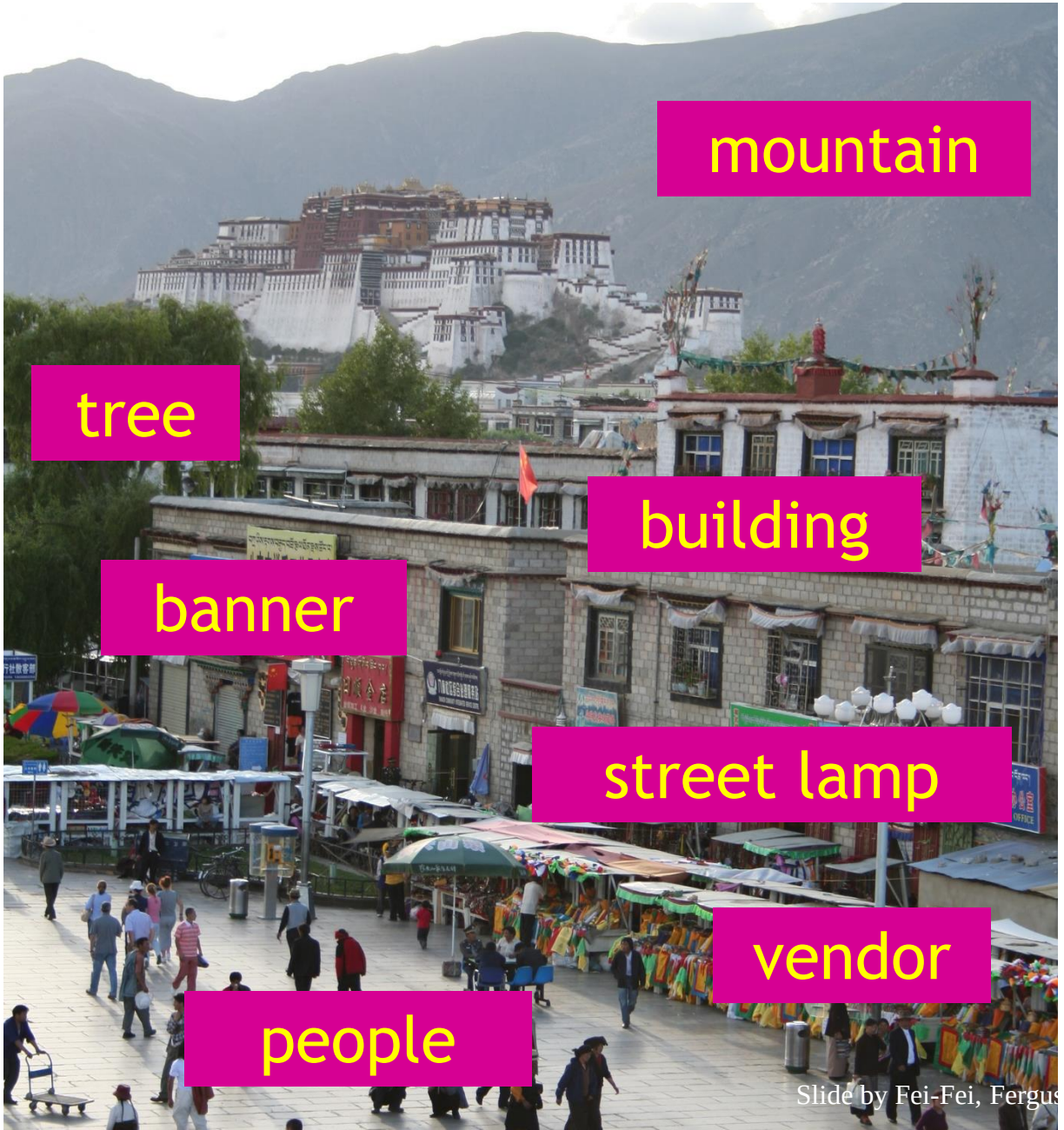
Identification: is that Potala Palace?



Slide by Fei-Fei, Fergus

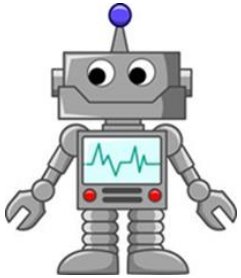
Object categorization

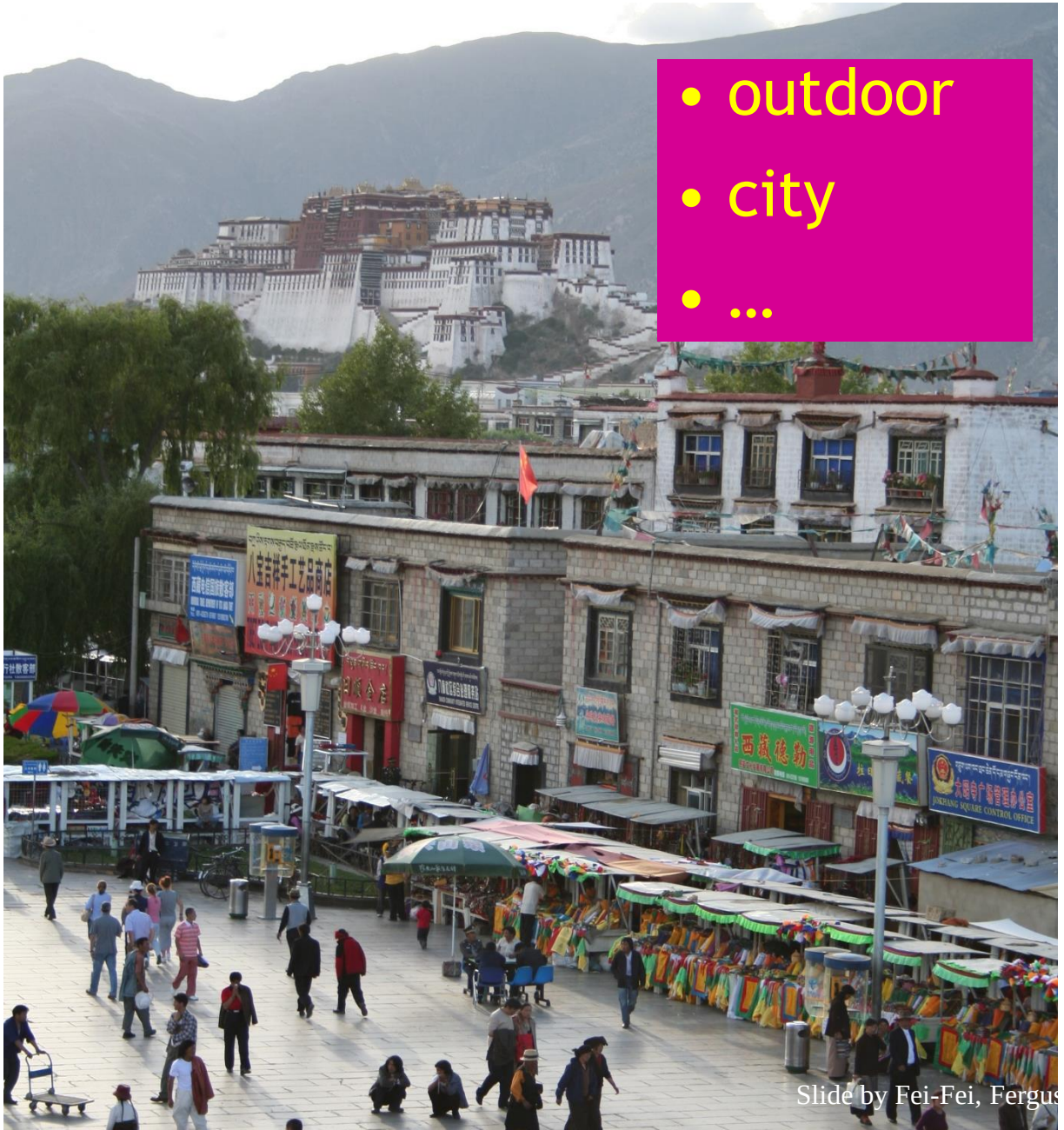




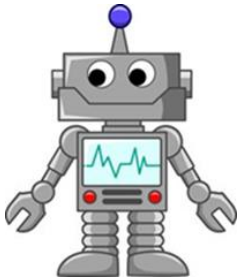
Slide by Fei-Fei, Fergus

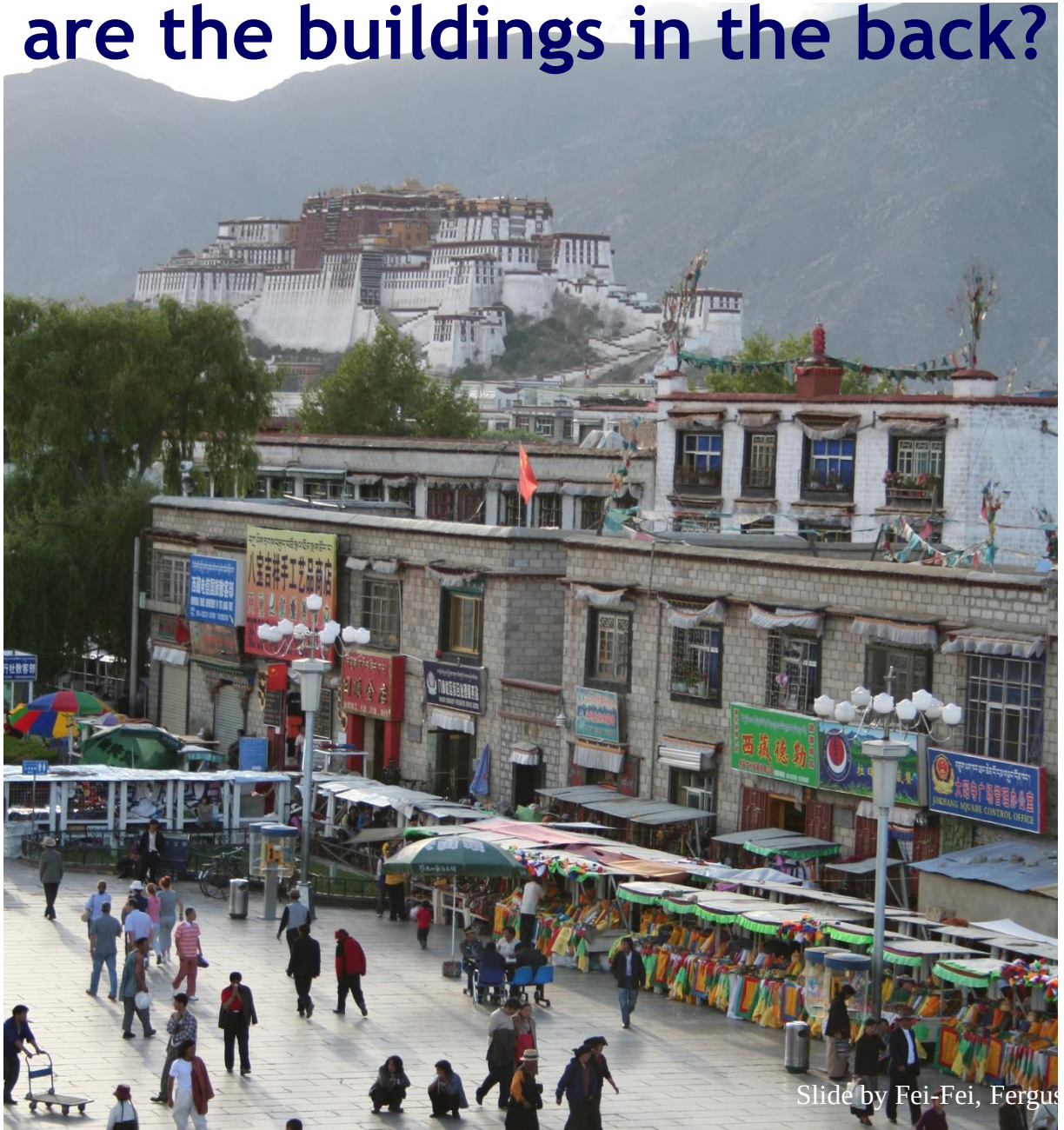
Scene and context categorization





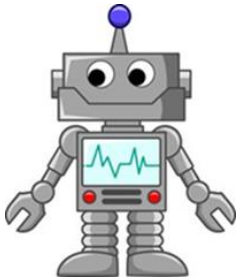
Is this space large or small?
How far are the buildings in the back?





Slide by Fei-Fei, Fergus

Activity



What is this person doing?

What are these two doing??

Slide by Fei-Fei, Fergus

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

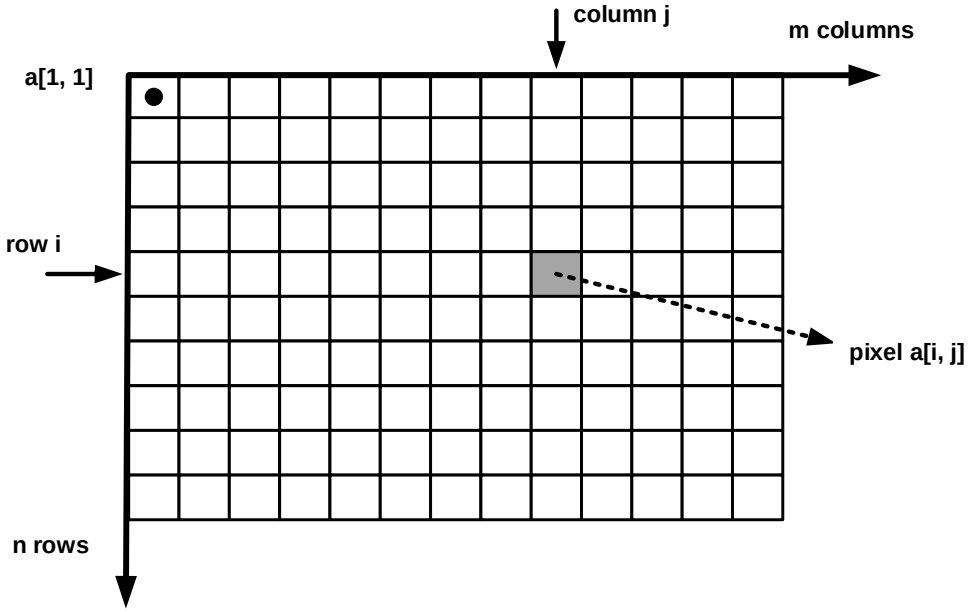
1. *Introduction of CV for robot*
2. ***Basic of Image***
   : What is a image? How can we get it?
3. *Image Processing Algorithm*

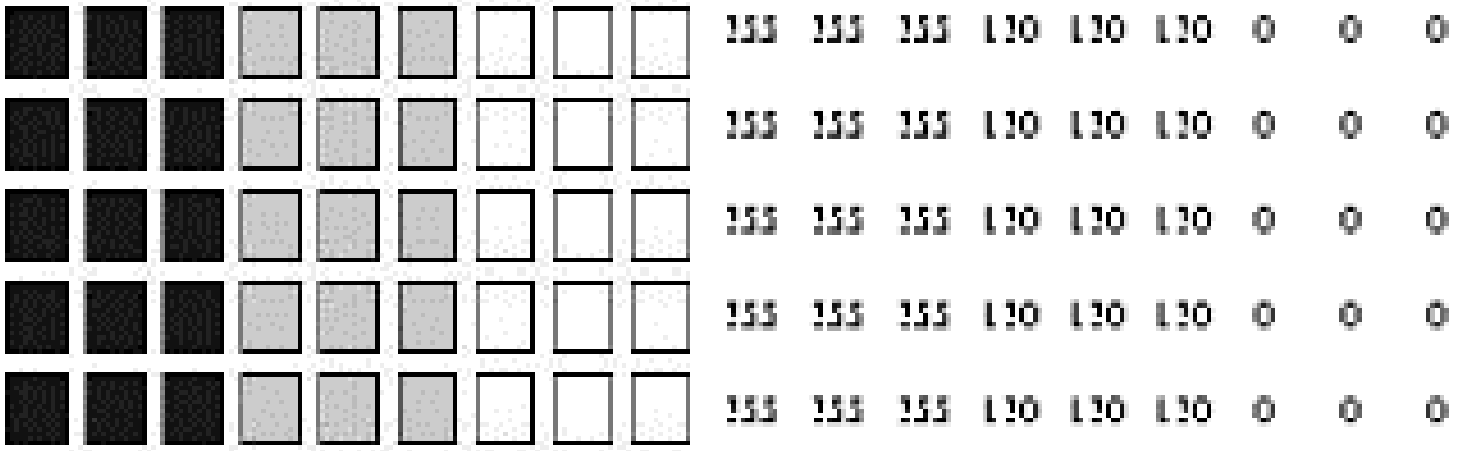
-Practical Part-

1. *Practical Algorithms for "Robot War"*

What is Image?

- Definition of “Image”
  - A two-dimensional array of pixels
  - The indices [i, j] of pixels : values that specify the rows and columns in pixel values

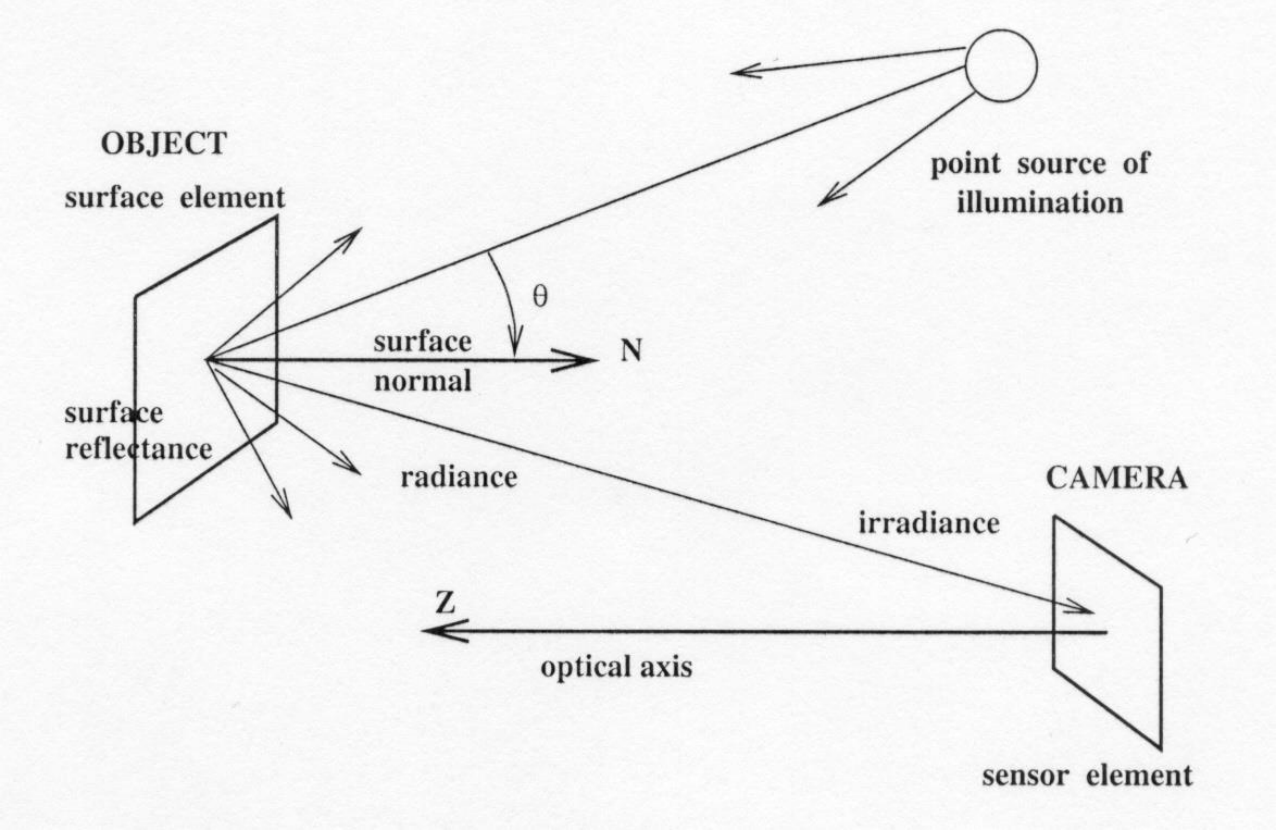




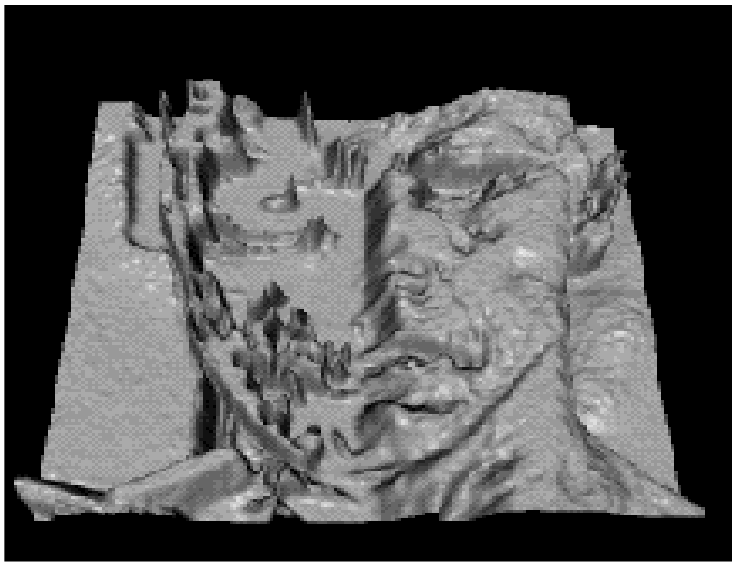
A Simple model of image formation

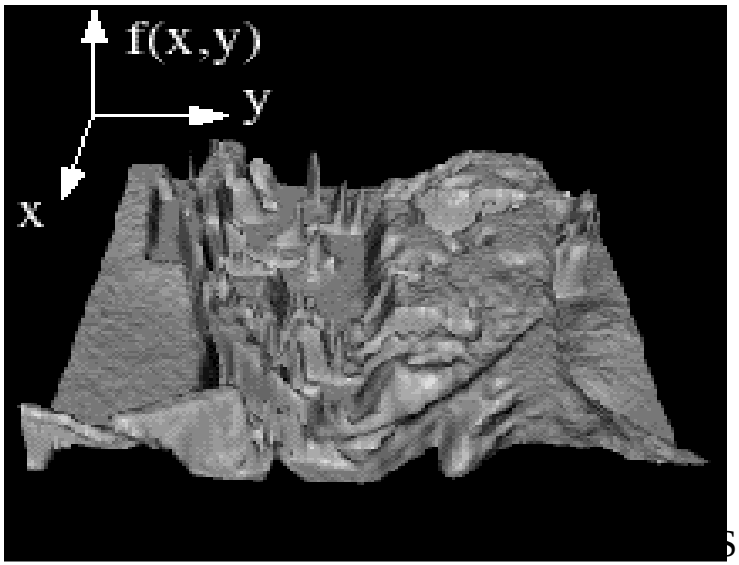
The **physics of light**, which determines the brightness of a point in the image plane as a function of illumination and surface properties.

The **geometry of image formation**, which determines where a point of a scene will be projected on the image plane.



Images as functions





Images as functions

- We can think of an image as a function, *f*, from R^2 to R:
  - $f(x, y)$ gives the intensity at position (x, y)
  - Realistically, we expect the image only to be defined over a rectangle, with a finite range:

$$f : [a,b] \times [c,d] \rightarrow [0, 1.0]$$

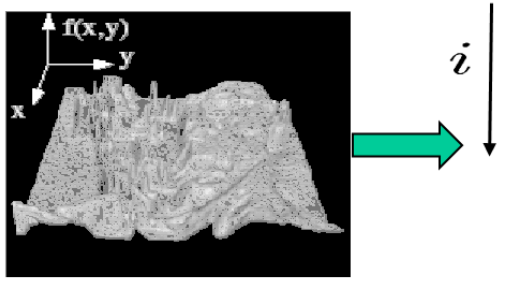
- A color image is just three functions pasted together. We can write this as a "vector-valued" function:

$$f(x,y) = \begin{bmatrix} r(x,y) \\ g(x,y) \\ b(x,y) \end{bmatrix}$$

Source: S. Seitz

How can we shift “light” to “digital image”

- Image digitization process
  - **1. Sampling**
    - measuring the value of an image at a finite number of points.
  - **2. Quantization**
    - Represent the measured value at the sampled point to the nearest integer.

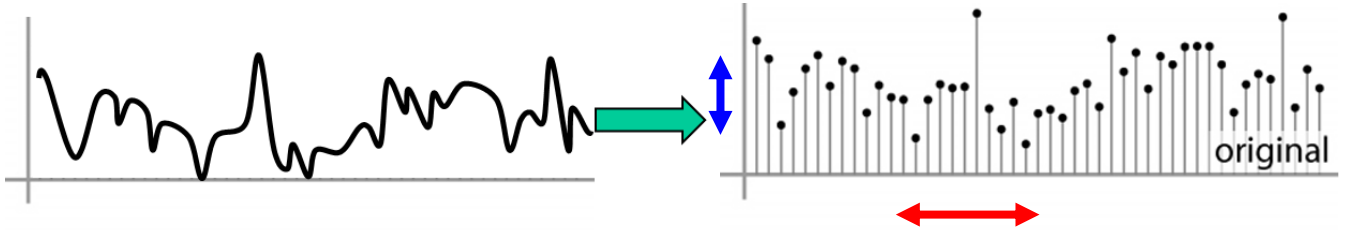


j

i

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

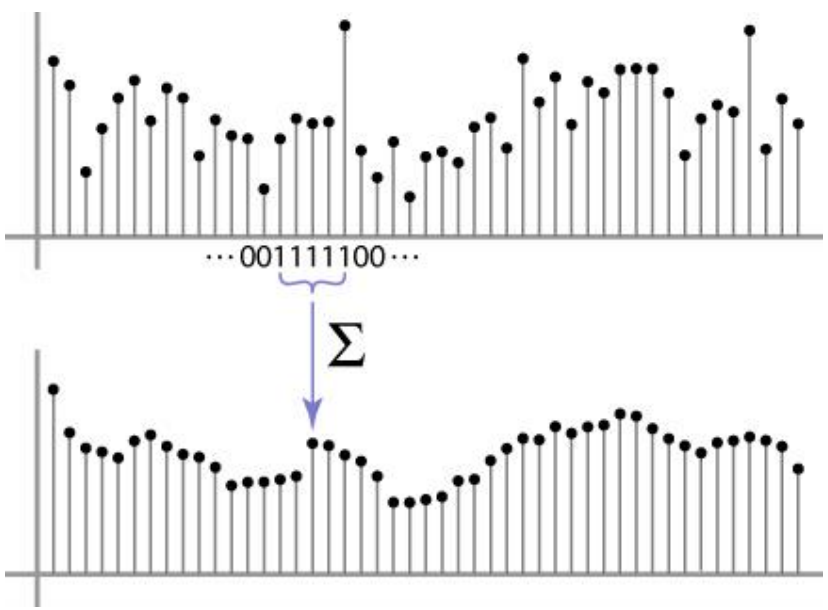
2D

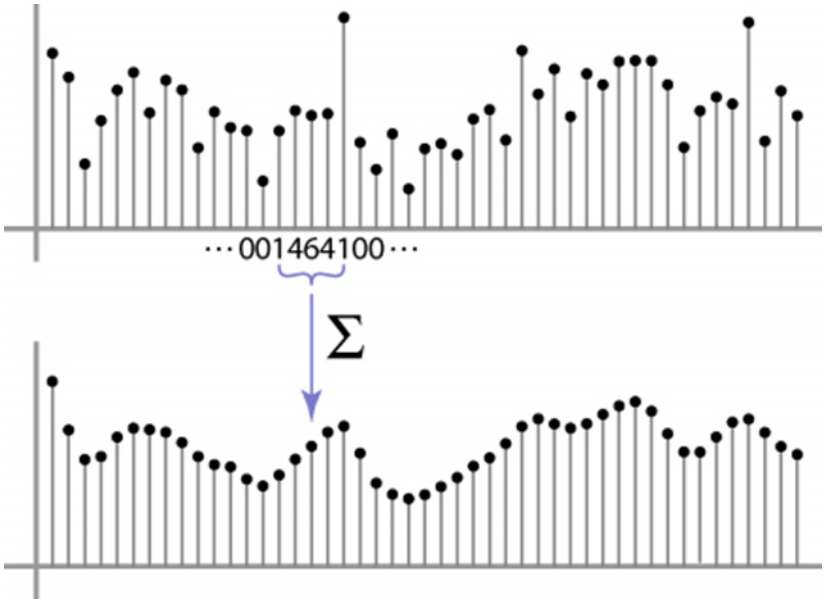


1D

Weighted Moving Average

- Let's replace each pixel with an average of all the values in its neighborhood
- Moving average in 1D:
- Can add weights to our moving average
- *Weights* [1, 1, 1, 1, 1] / 5, Non-uniform weights [1, 4, 6, 4, 1] / 16





Source: S. Marschner

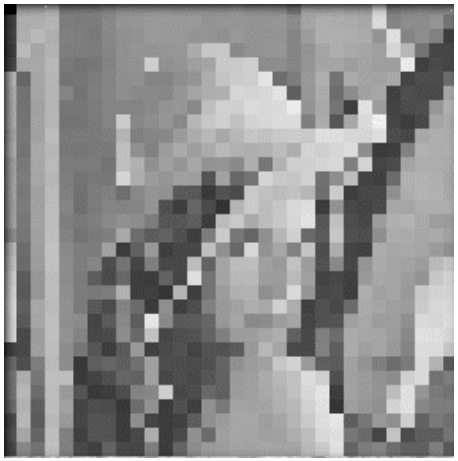
Example of Image Sampling

Original Image

Sampled by
a factor of 2

Sampled by
a factor of 4

Sampled by
a factor of 8



Example of Image Quantization

256 gray levels
(8bits/pixel)

32 gray levels
(5bits/pixel)

16 gray levels
(4bits/pixel)

8 gray levels
(3bits/pixel)

4 gray levels
(2bits/pixel)

2 gray levels
(1bits/pixel)

Colour or Monochrome?

- Monochrome images require 1 number – greyscale (0 - 255)

- Colour images require 3 numbers – red, green and blue values (0–255)

- Colour images require more work, but give more information.

- Examples are greyscale for simplicity. There is nothing harder about colour, it just requires more work.

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
2. *Basic of Image*
3. ***Image Processing Algorithm***
   - ***Elementary Processing***
   - ***Applications of Elementary Processing***

-Practical Part-

1. *Practical Algorithms for "Robot War"*

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
2. *Basic of Image*
3. ***Image Processing Algorithm***
   - ***Elementary Processing***
     : Histogram Equalization, Filtering
   - *Applications of Elementary Processing*

-Practical Part-

1. *Practical Algorithms for "Robot War"*

Histograms

- Histogram: plots how many times (frequency) each intensity value in image occurs
- Example:
  - Image (left) has 256 distinct gray levels (8 bits)
  - Histogram (right) shows frequency (how many times) each gray level occurs

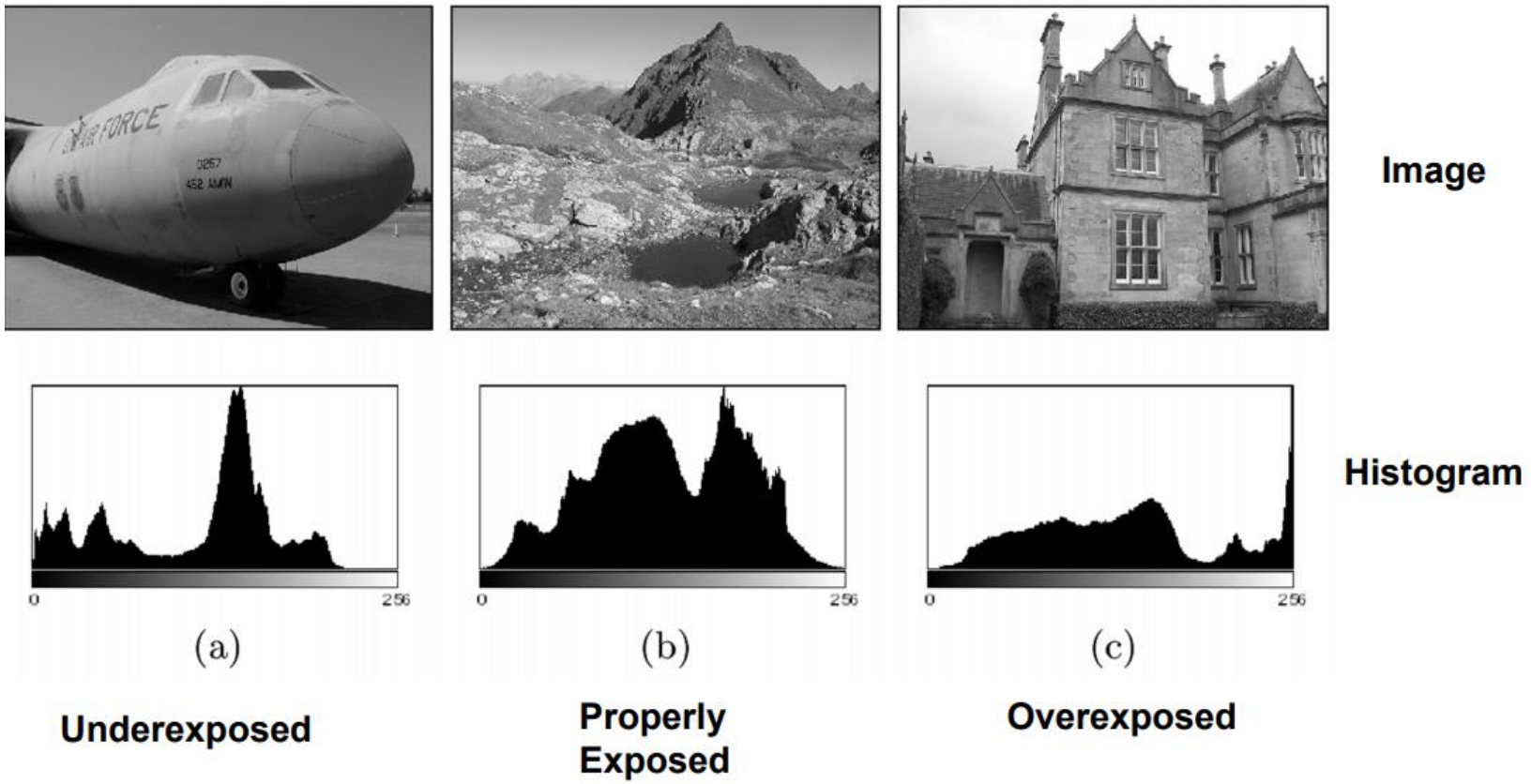
Histograms

- Different images can have the same histogram
- 3 images below have same histogram

- Half of pixels are gray, half are white
  - Same histogram = same statistics
  - Distribution of intensities could be made the same → Histogram Equalization
- Can we reconstruct image from histogram? No!

Application – Detecting Bad Exposure

- Exposures Are intensity values spread (good) out or bunched up (bad)



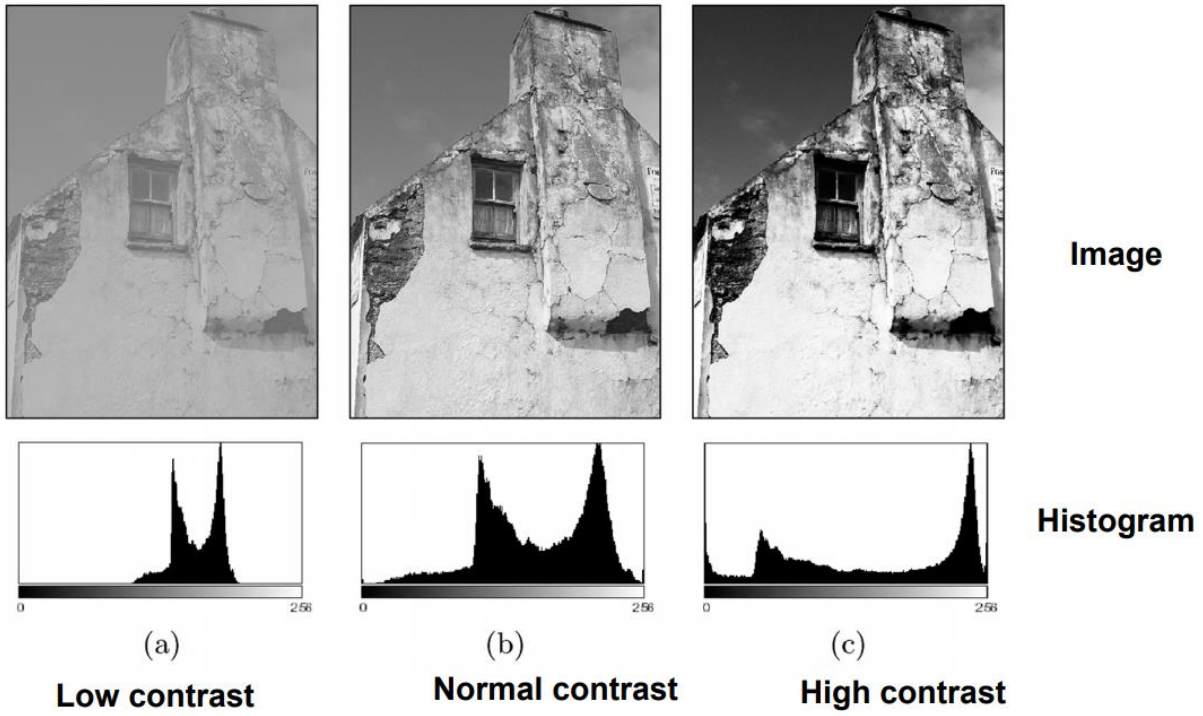
Application – Image Contrast

- The contrast of a grayscale image indicates how easily objects in the image can be distinguished
- High contrast image: many distinct intensity values
- Low contrast: image uses few intensity values

Good Contrast?

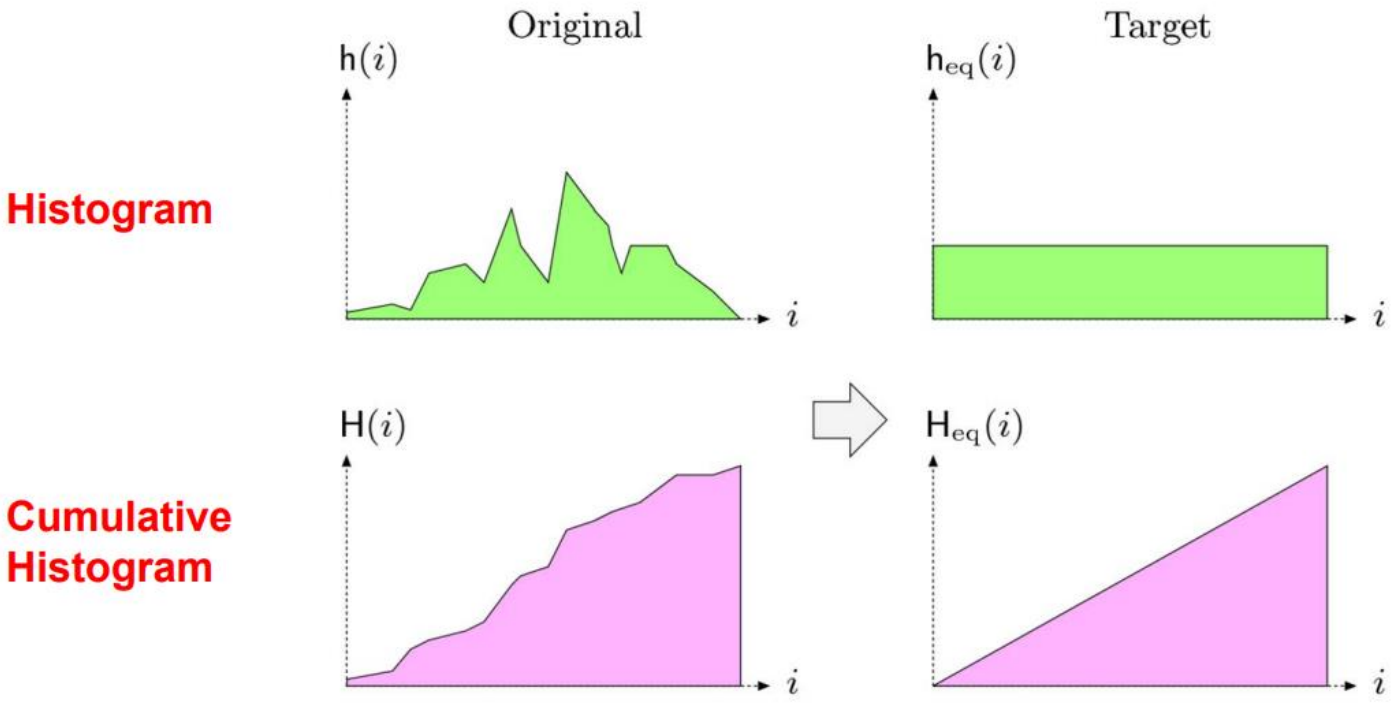
Widely spread intensity values

+ large difference between min and max intensity values

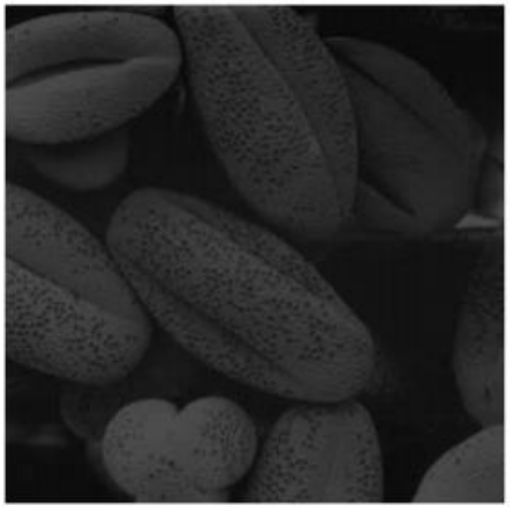


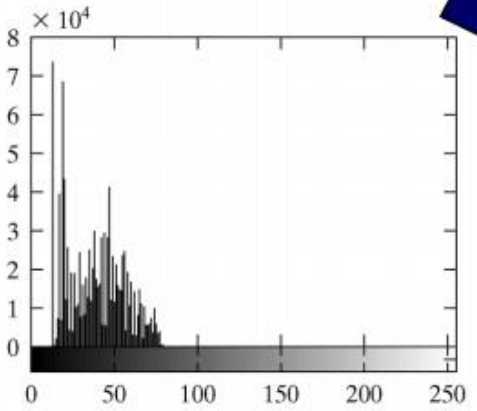
Histogram Equalization

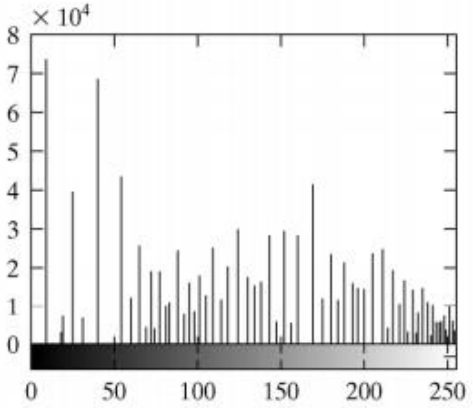
- Adjust 2 different images to make their histograms (intensity distributions) similar
- Apply a point operation that changes histogram of modified image into uniform distribution

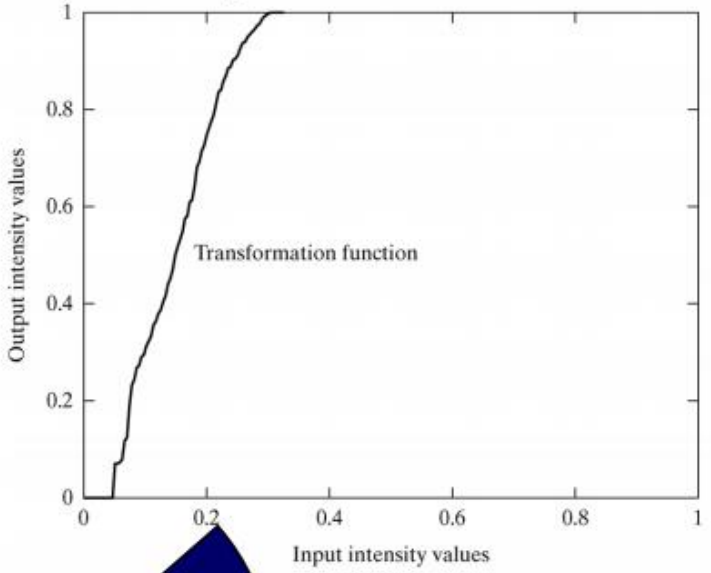


Histogram Equalization

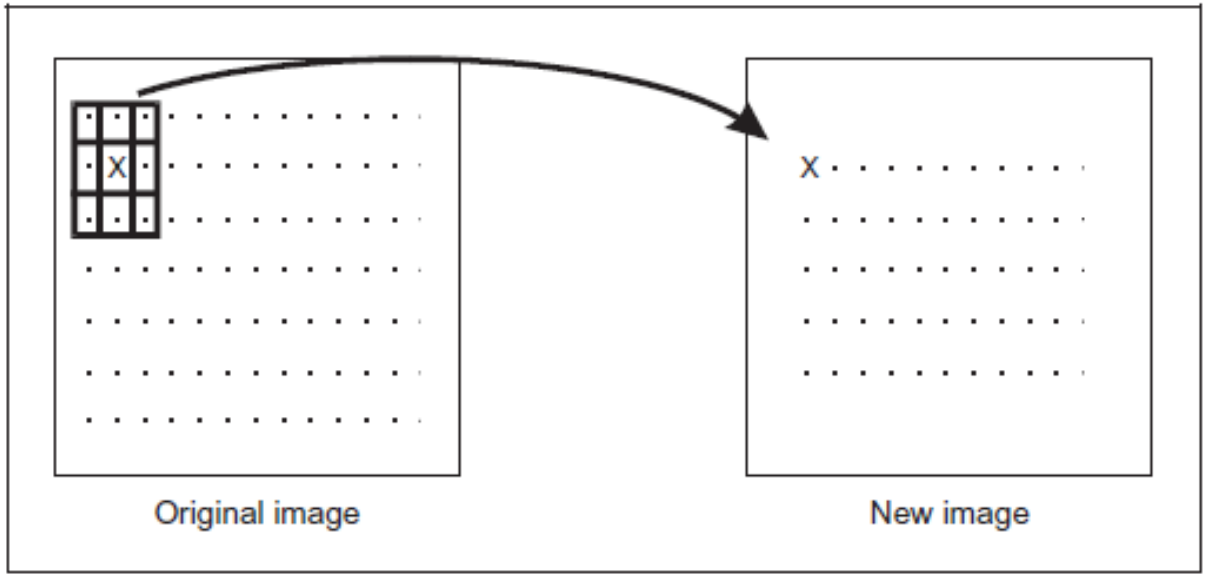








Filtering and Convolution Operation



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

Moving Average In 2D

$F[x, y]$

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								

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

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							

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

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						

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

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					

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

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				

Source: S. Seitz

Moving Average In 2D

$F[x, y]$

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	

Source: S. Seitz

Correlation filtering

Say the averaging window size is 2k+1 x 2k+1:

$$H(i,j) = \underbrace{\frac{1}{(2k+1)^2}}_{\text{Attribute uniform weight to each pixel}} \underbrace{\sum_{u=-k}^{k}\sum_{v=-k}^{k} F(i+u, j+v)}_{\text{Loop over all pixels in neighborhood around image pixel F[i,j]}}$$

Now generalize to allow different weights depending on neighboring pixel's relative position:

$$H(i,j) = \sum_{u=-k}^{k}\sum_{v=-k}^{k} \underbrace{G(u,v)}_{\text{Non-uniform weights}} \cdot F(i+u, j+v)$$

Correlation filtering

$$H(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} G(u,v) \cdot F(i+u, j+v)$$

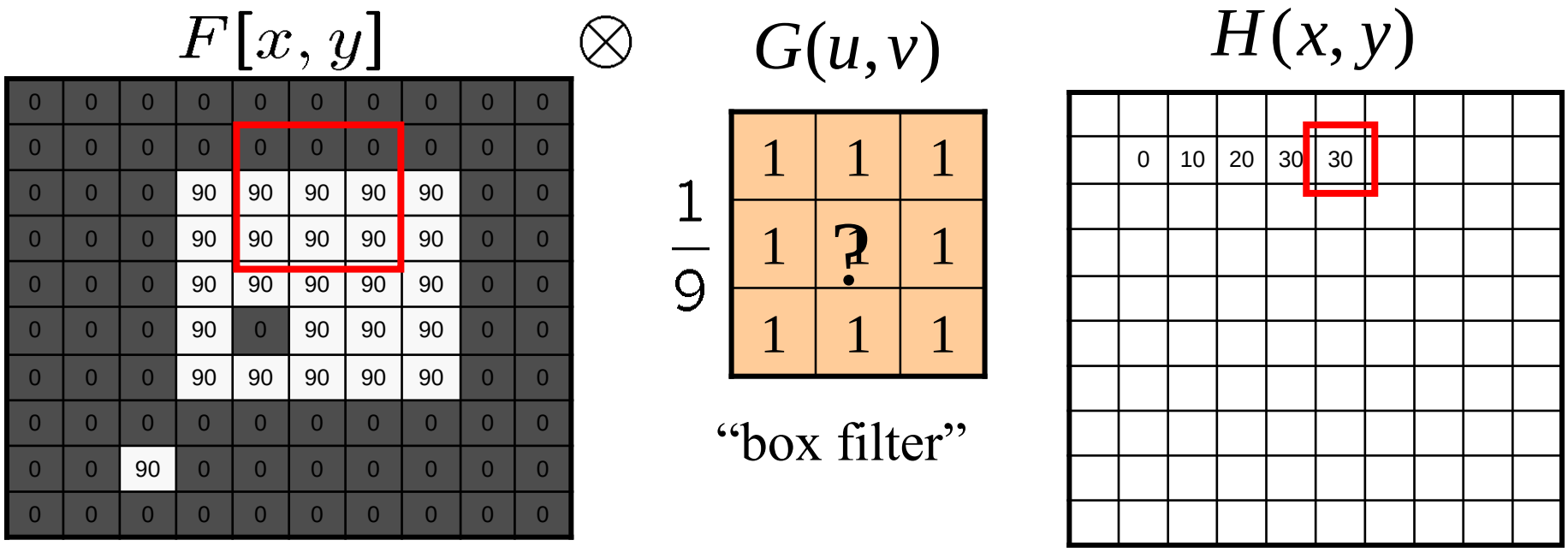
This is called cross-correlation, denoted $H = G \times F$

Filtering an image: replace each pixel with a linear combination of its neighbors.

The filter "kernel" or "mask" $G[u,v]$ is the prescription for the weights in the linear combination.

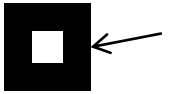
Averaging filter

- What values belong in the kernel *H* for the moving average example?



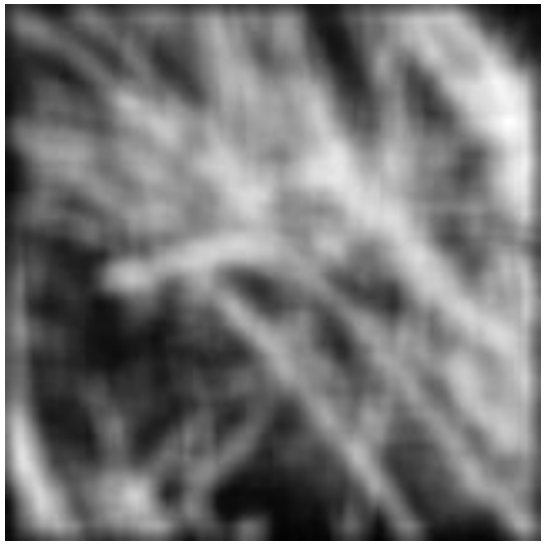
$$H = G \times F$$

Smoothing w/ Averaging Filter



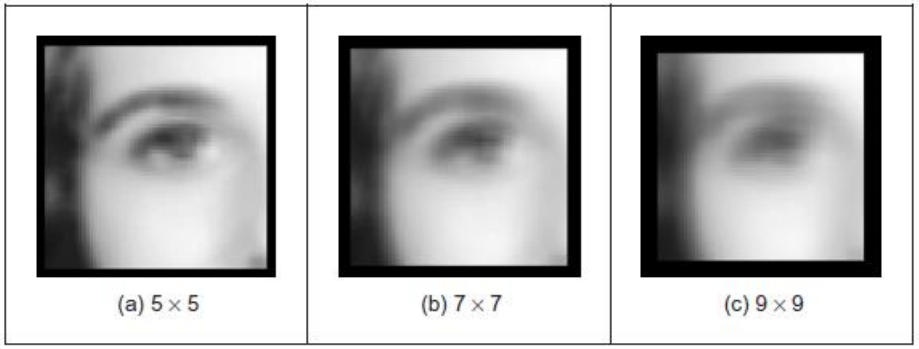
depicts box filter:
white = high value, black = low value

original



filtered

Averaging w/ Different Filter Size



Gaussian filter

- What if we want nearest neighboring pixels to have the most influence on the output?

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

$F[x, y]$

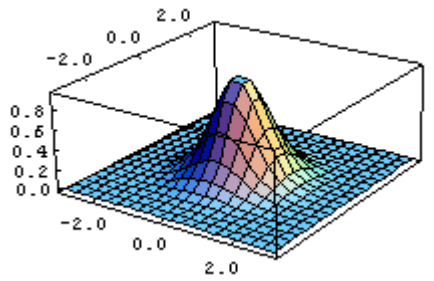
$$\frac{1}{16}$$

1	2	1
2	4	2
1	2	1

$G(u, v)$

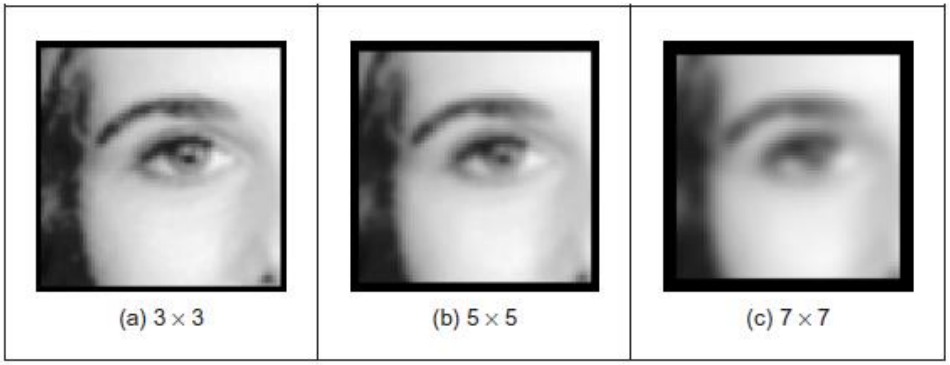
This kernel is an approximation of a Gaussian function:

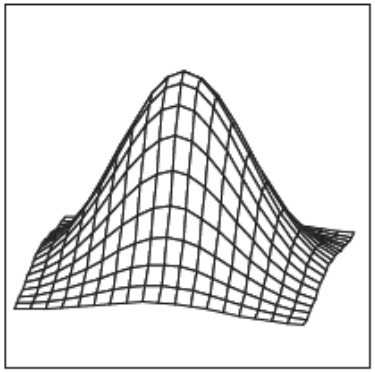
$$h(u, v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{\sigma^2}}$$



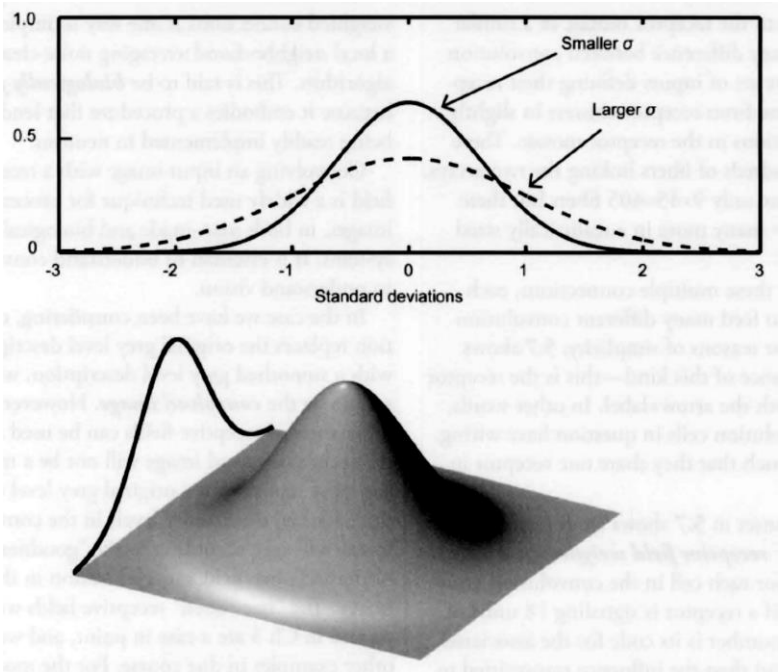
Source: S. Seitz

Gaussian Filter



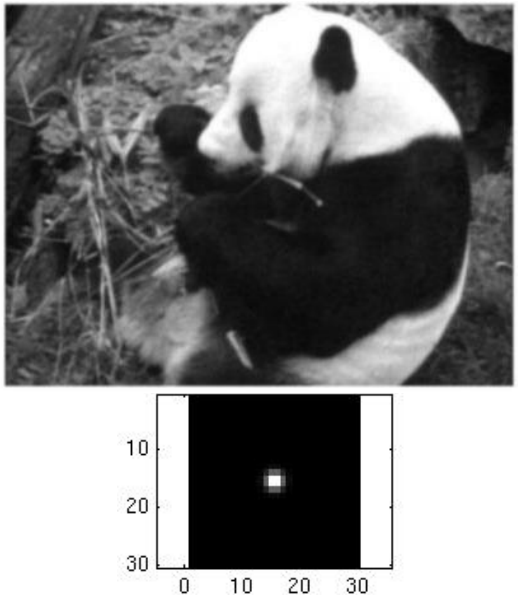


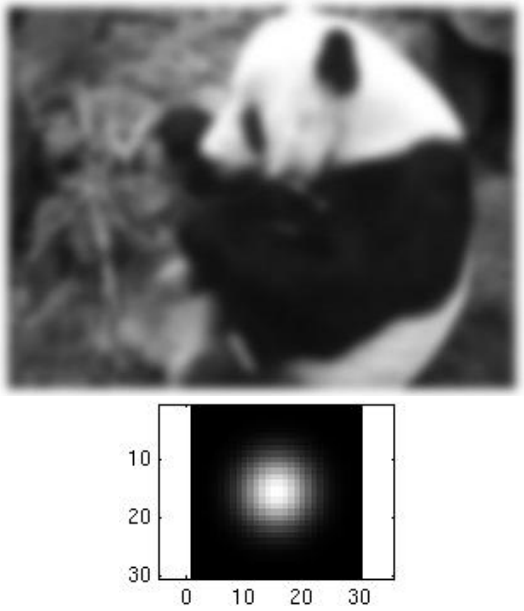
0.002	0.013	0.022	0.013	0.002
0.013	0.060	0.098	0.060	0.013
0.022	0.098	0.162	0.098	0.022
0.013	0.060	0.098	0.060	0.013
0.002	0.013	0.022	0.013	0.002



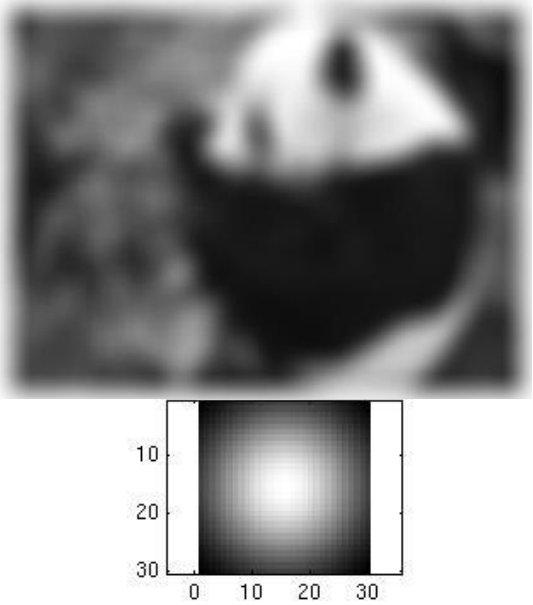
Smoothing w/ a Gaussian

Parameter σ is the “scale” / “width” / “spread” of the Gaussian kernel, and controls the amount of smoothing.



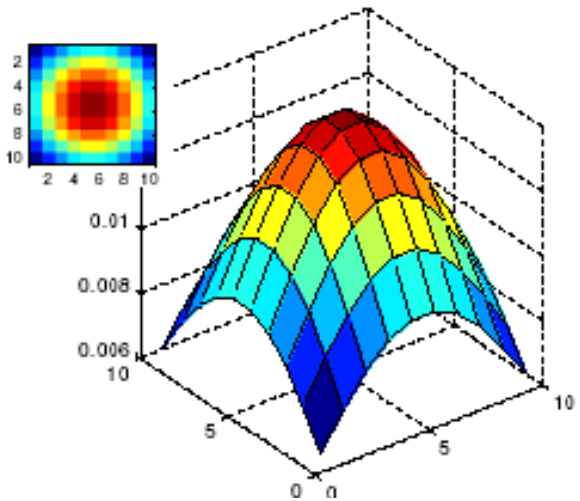


. . .

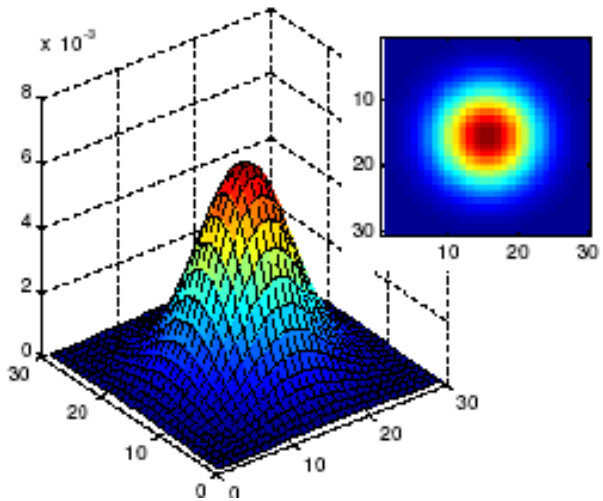


Gaussian filters

- What parameters matter here?
- **Size** of kernel or mask
  - Note, Gaussian function has infinite support, but discrete filters use finite kernels



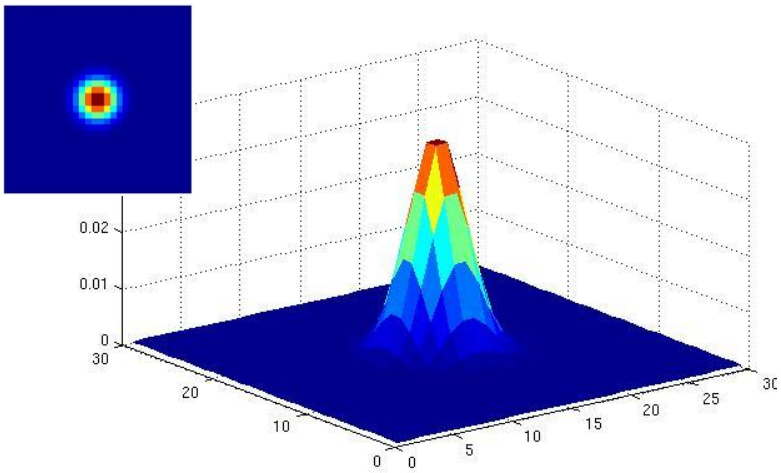
σ = 5 with 10 x 10 kernel



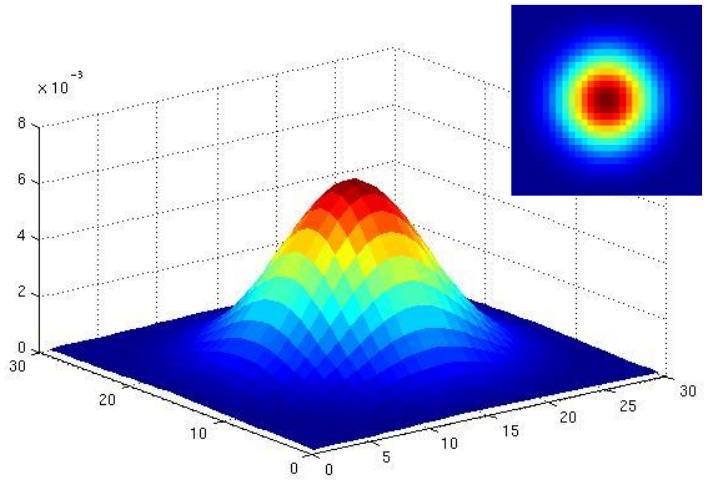
σ = 5 with 30 x 30 kernel

Gaussian filters

- What parameters matter here?
- **Variance** of Gaussian: determines extent of smoothing

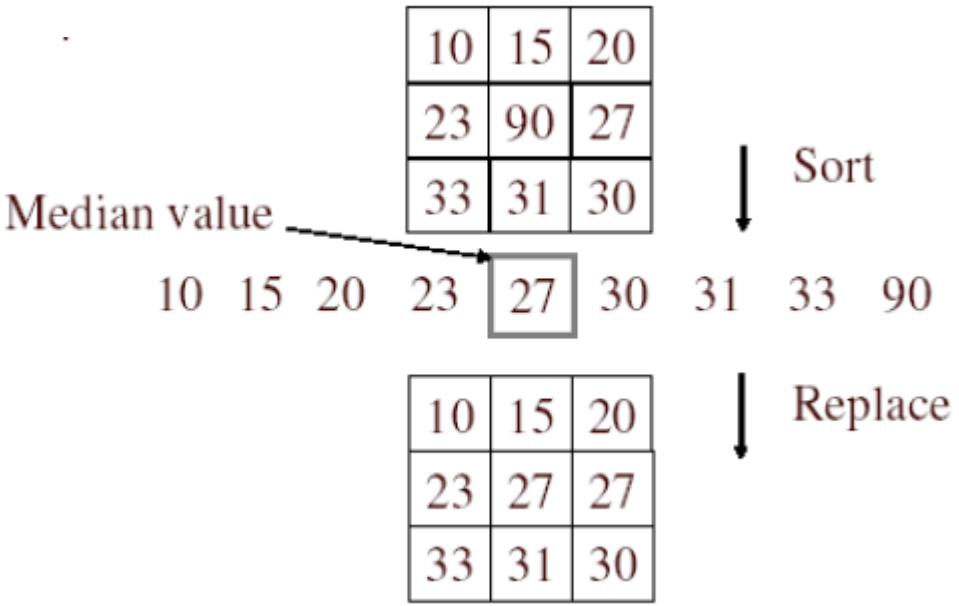


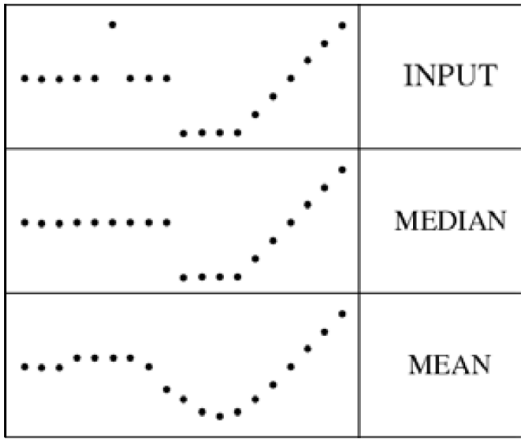
$\sigma = 2$ with 30 x 30 kernel



$\sigma = 5$ with 30 x 30 kernel

Median filter



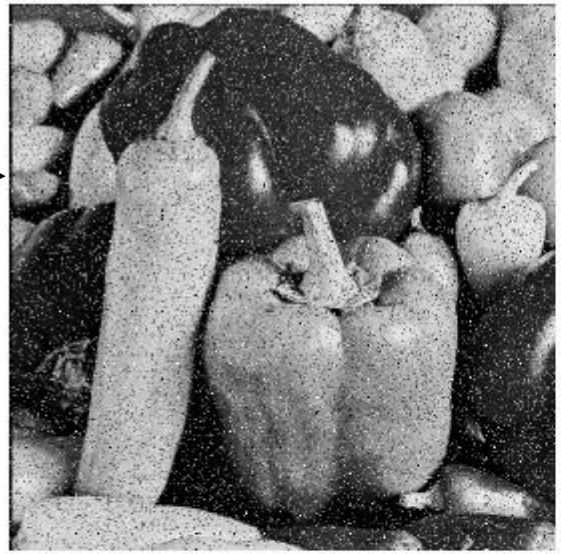


- No new pixel values introduced
- Removes spikes: good for impulse, salt & pepper noise

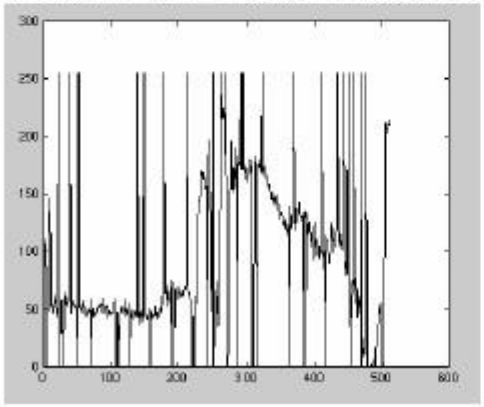
- Median filter is edge preserving

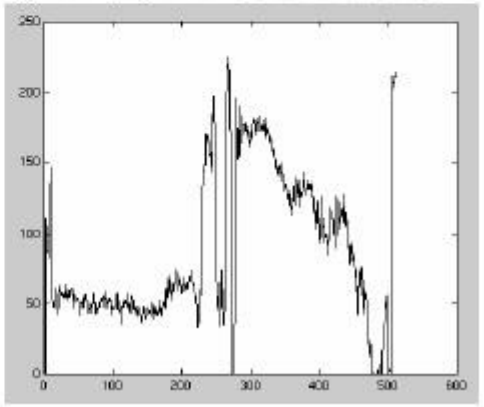
Smoothing w/ Median filter

Salt and pepper noise



Median filtered

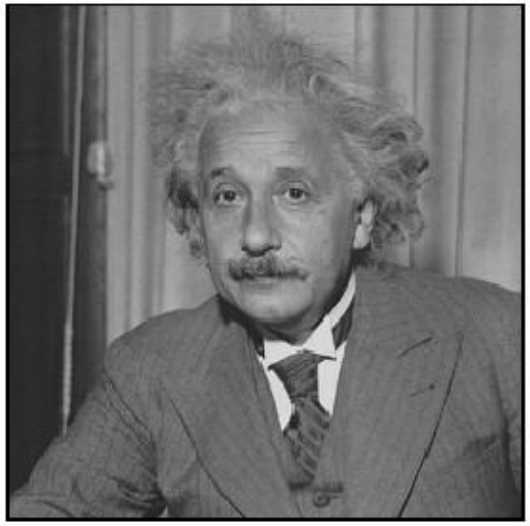


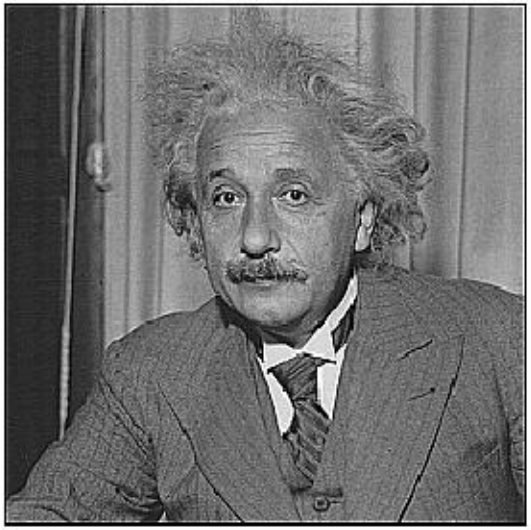


Plots of a row of the image

Source: M. Hebert

Example of Applications using Convolution





Smooth/Sharpen Images...

Find edges...

Find waldo…

Convolution and Cross-Correlation

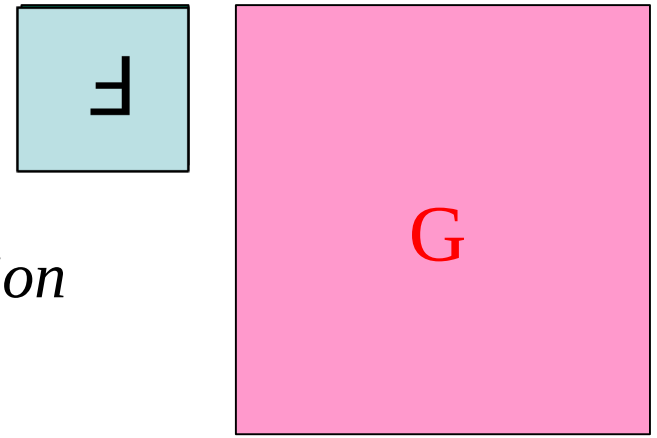
- **Convolution:**
  - Flip the filter (bottom to top, right to left)
  - Then apply cross-correlation

$$H(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} G(u,v) \cdot F(i-u, j-v)$$

$$H = G * F = G \times \exists$$

convolution operator

Cross-correlation operator



Convolution vs. Correlation

Convolution

$$H(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} G(u,v) \cdot F(i-u, j-v)$$

$$H = G * F$$

Cross-correlation

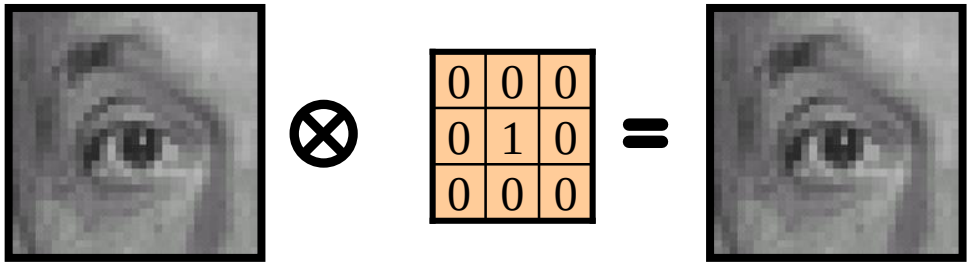
$$H(i,j) = \sum_{u=-k}^{k} \sum_{v=-k}^{k} G(u,v) \cdot F(i+u, j+v)$$

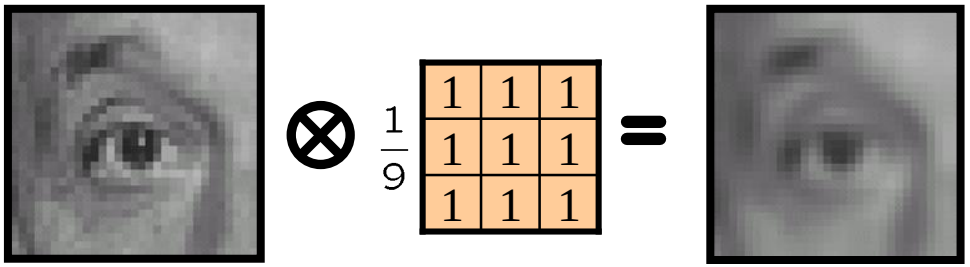
$$H = G \times F$$

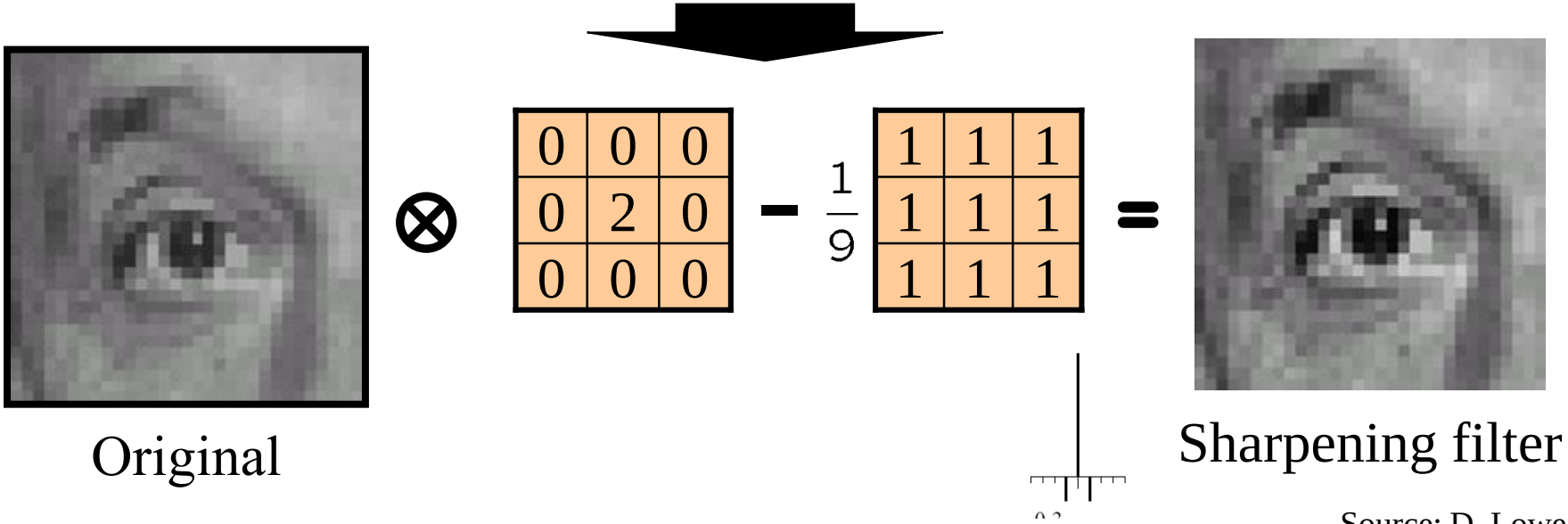
For a Gaussian or box filter, how will the outputs differ?

If the input is an impulse signal, how will the outputs differ?

Convolution Example using Linear Property







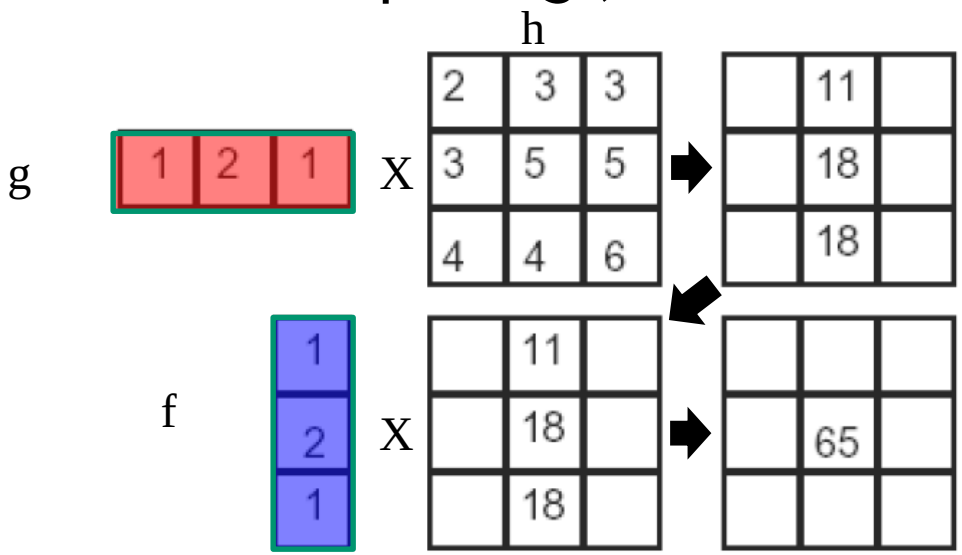
Original

Sharpening filter

Source: D. Lowe

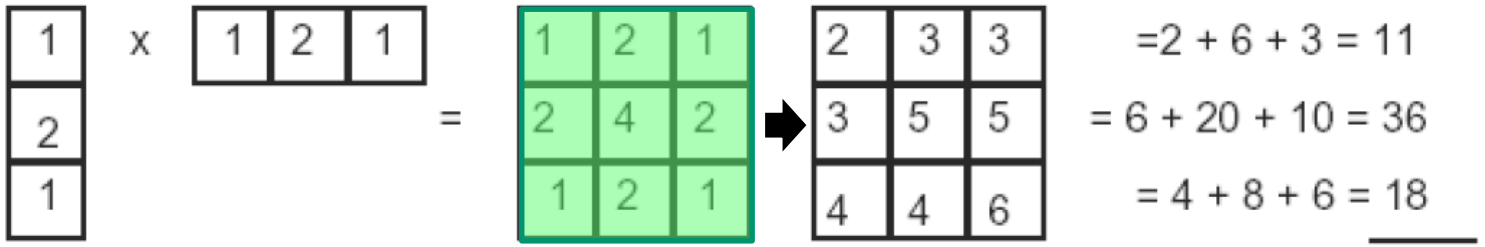
Separability

- In some cases, filter is separable, and we can factor into two steps: e.g.,



12+17+19=48

What is the computational complexity advantage for a separable filter of size k x k, in terms of number of operations per output pixel?



f * (g * h) = (f * g) * h

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
2. *Basic of Image*
3. ***Image Processing Algorithm***
   - *Elementary Processing*
   - ***Applications of Elementary Processing***
     : Edge, Distance, Optical Flow Detection

-Practical Part-

1. *Practical Algorithms for "Robot War"*

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

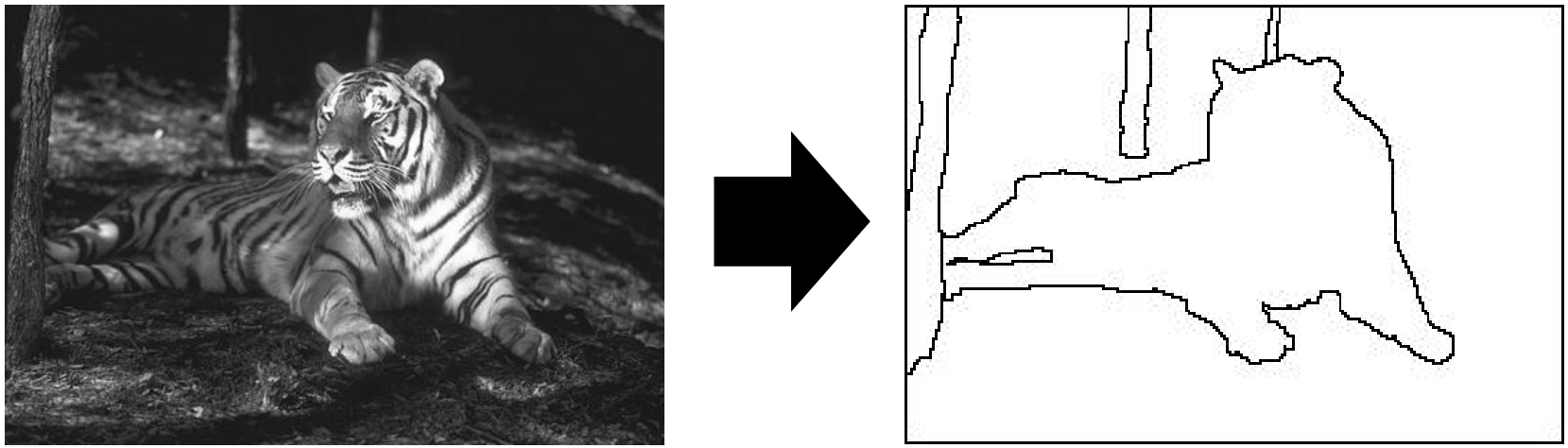
1. Introduction of CV for robot
2. Basic of Image
3. **Image Processing Algorithm**
   - Elementary Processing
   - **Applications of Elementary Processing**
     : Edge, Distance, Optical Flow **Detection**

-Practical Part-

1. Practical Algorithms for “Robot War”

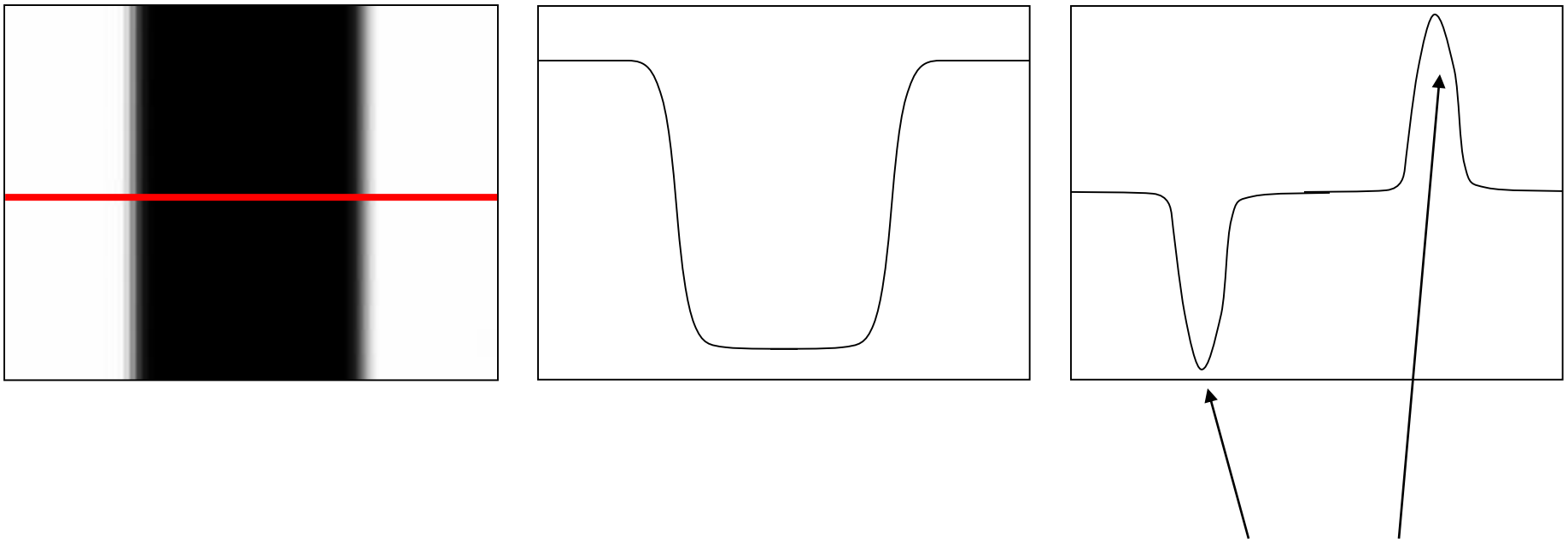
Advanced Image Processing - Edge Detection

- Convert a 2D image into a set of curves
  - Reduce dimensionality of data, but preserve useful information
    - Extracts salient features of the scene
  - More compact than pixels
  - Useful in applications such as:
    - object detection, structure from motion, tracking, etc …



Derivatives and edges

An edge is a place of rapid change in the image intensity function.



Source: L. Lazebnik

Differentiation and convolution

For 2D function, f(x,y), the partial derivative is:

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

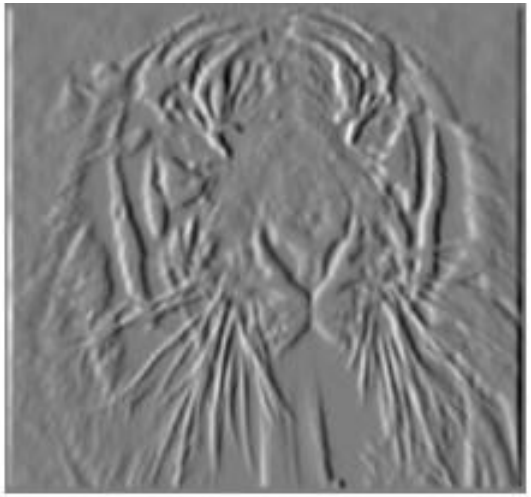
For discrete data, we can approximate using finite differences:

$$\frac{\partial f(x,y)}{\partial x} \approx \frac{f(x+1, y) - f(x,y)}{1}$$

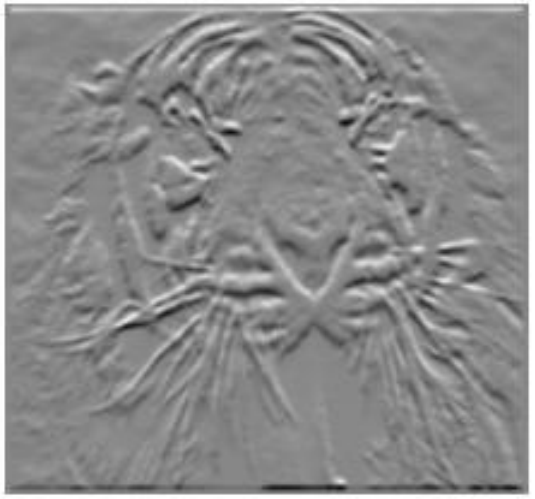
To implement above as convolution, what would be the associated filter?

Partial derivatives of an image

$$\frac{\partial f(x,y)}{\partial x}$$



$$\frac{\partial f(x,y)}{\partial y}$$



-1	1

-1
1

?

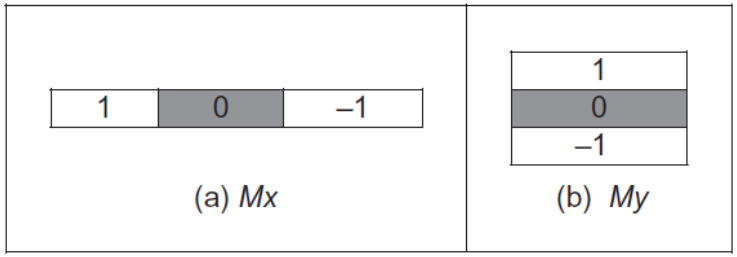
or

1
-1

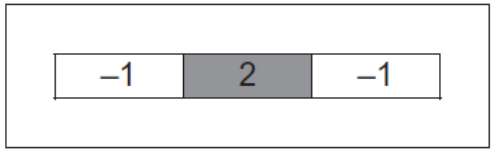
Which shows changes with respect to x?

(showing flipped filters)

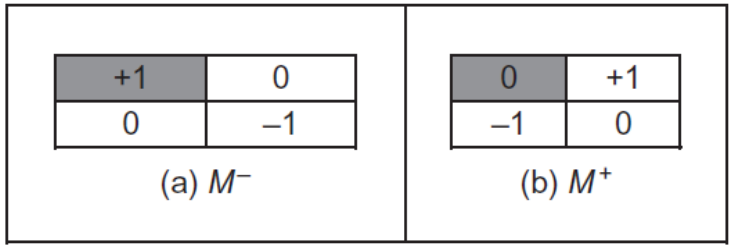
Edge Detection Filters



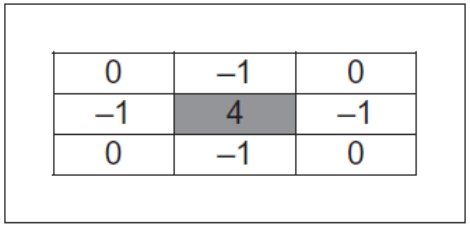
First order difference



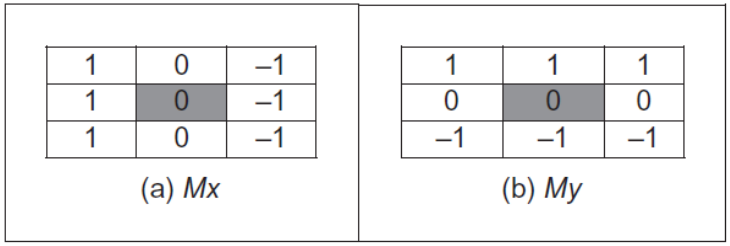
Second order difference



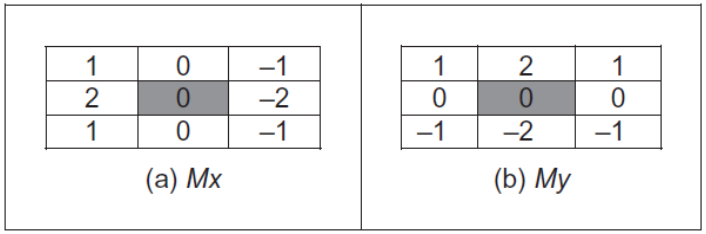
Roberts



Laplacian



Prewitt



Sobel

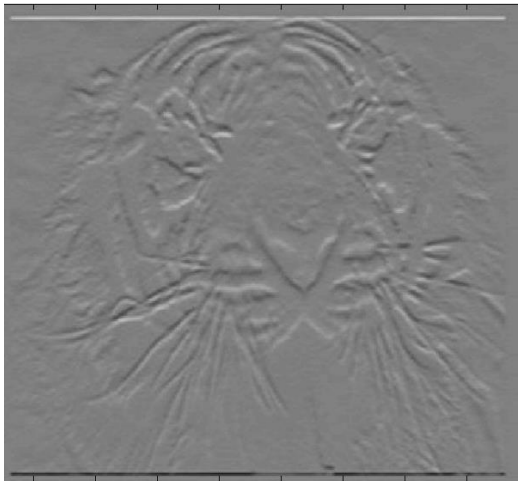
Assorted finite difference filters

Prewitt: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Sobel: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

```
>> My = fspecial('sobel');
>> outim = imfilter(double(im), My);
>> imagesc(outim);
>> colormap gray;
```



Detail of Edge Detection

- How can we estimate edge?
  - Estimated as edges if the brightness difference between neighboring pixels exceeds the certain threshold

- Different Edge Detection with Different methods

Original Image

Sobel

Prewitt

Laplacian

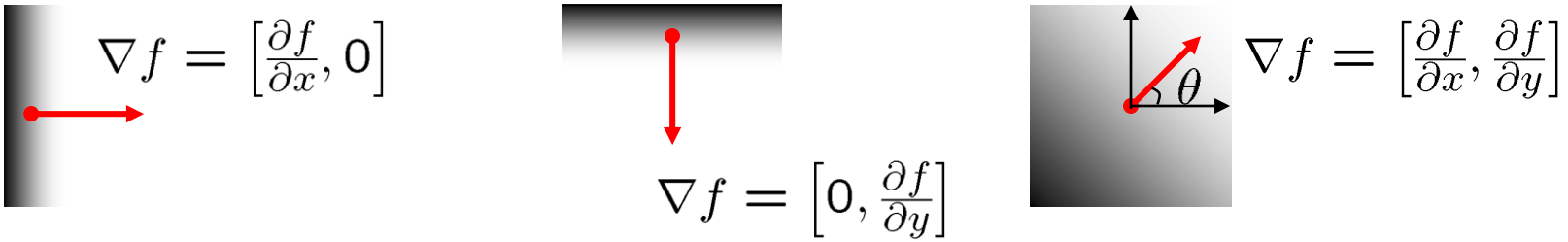
Canny

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



The gradient direction (orientation of edge normal) is given by:

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

The *edge strength* is given by the gradient magnitude

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

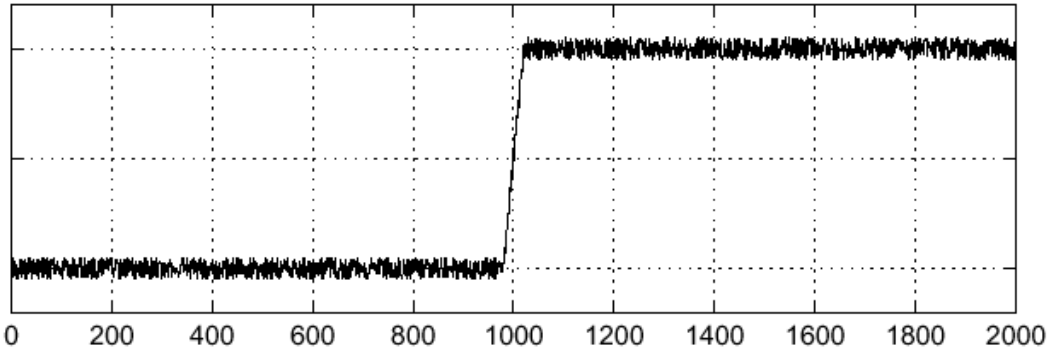
Slide credit S. Seitz

Effects of noise

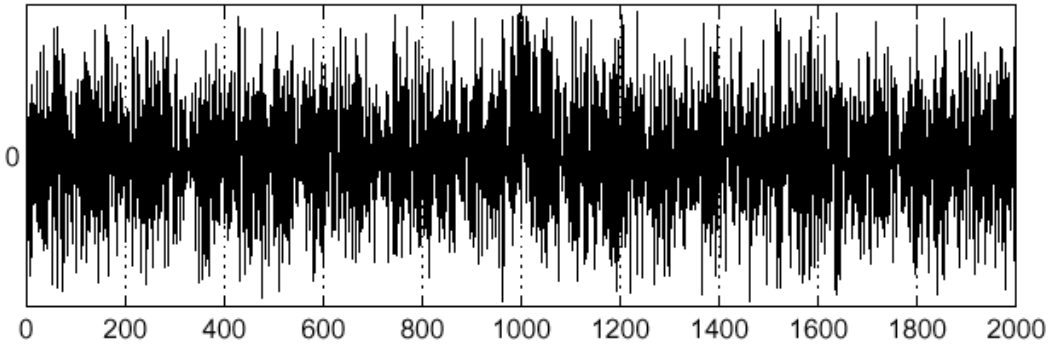
Consider a single row or column of the image

- Plotting intensity as a function of position gives a signal

$f(x)$



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



Where is the edge?

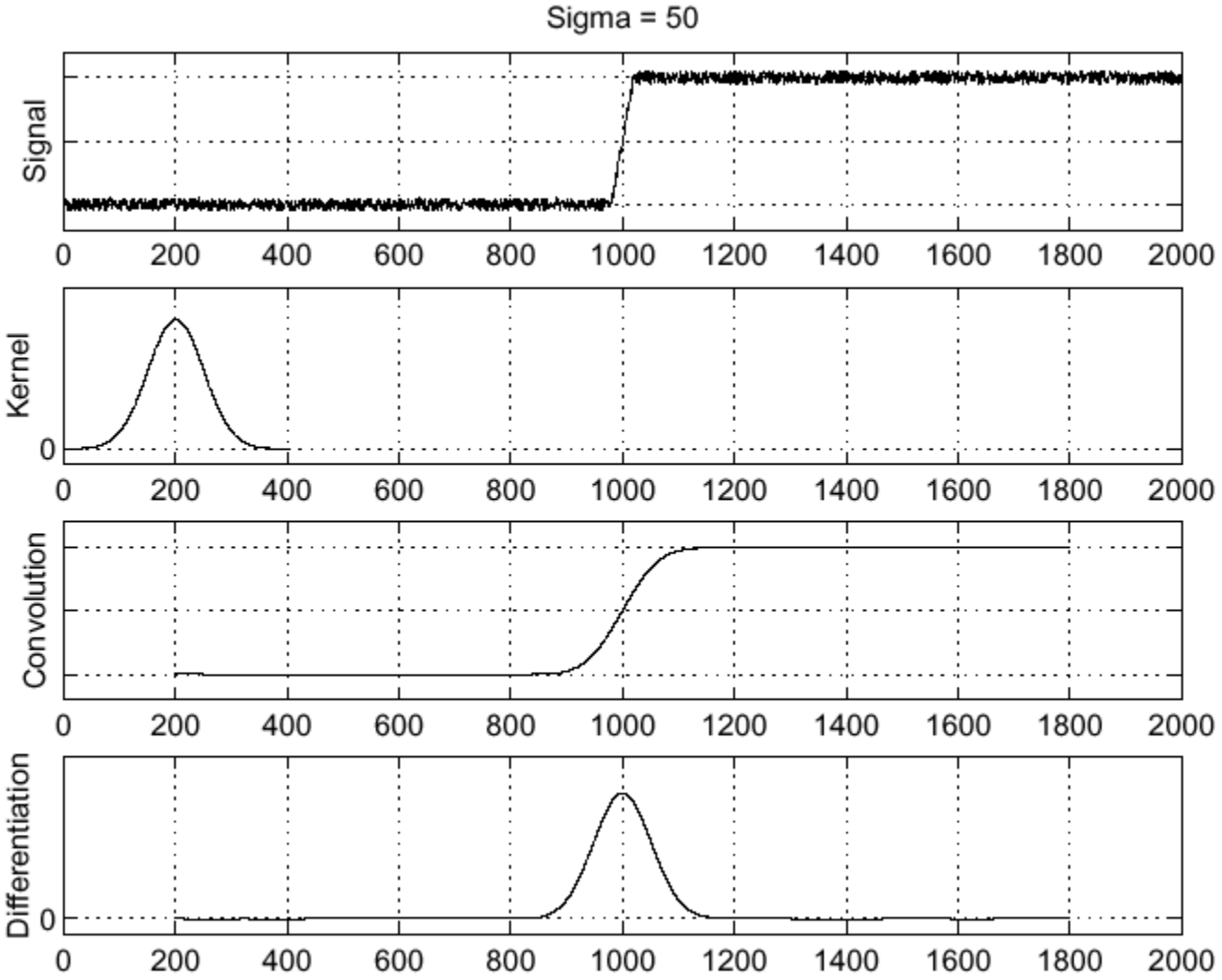
Solution: smooth first

f

h

$h \star f$

$\frac{\partial}{\partial x}(h \star f)$



Where is the edge? Look for peaks in $\frac{\partial}{\partial x}(h \star f)$

Derivative theorem of convolution

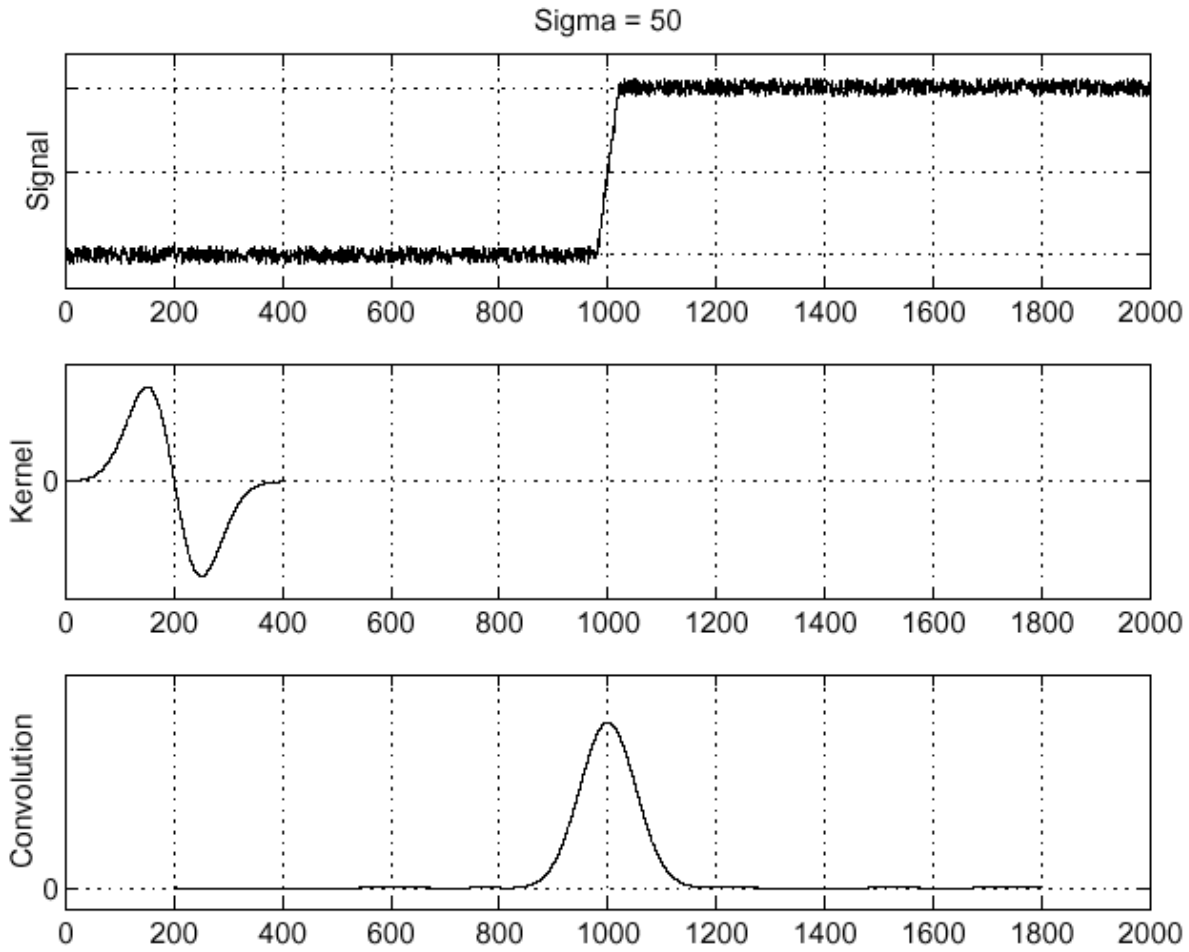
$$\frac{\partial}{\partial x}(h \star f) = (\frac{\partial}{\partial x}h) \star f$$

Differentiation property of convolution.

f

$\frac{\partial}{\partial x}h$

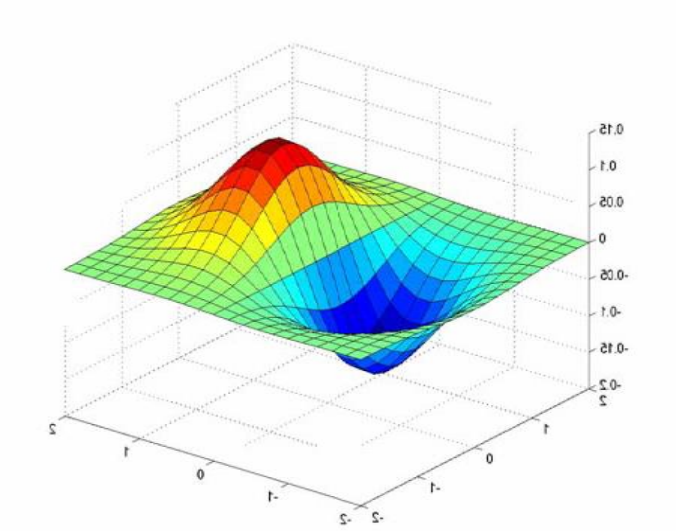
$(\frac{\partial}{\partial x}h) \star f$



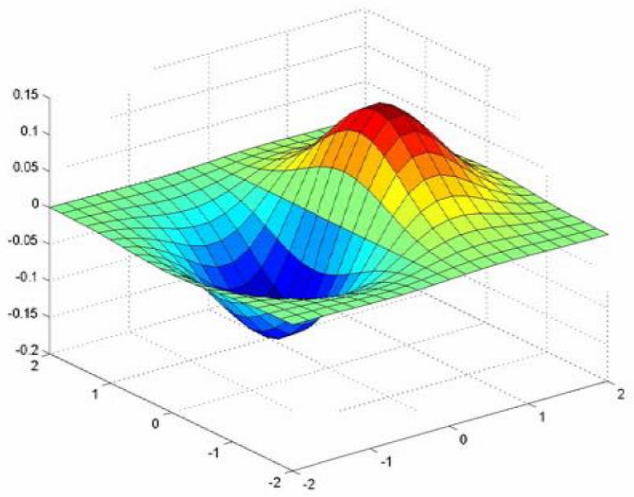
Derivative of Gaussian filter

$$(I \otimes g) \otimes h = I \otimes (g \otimes h)$$

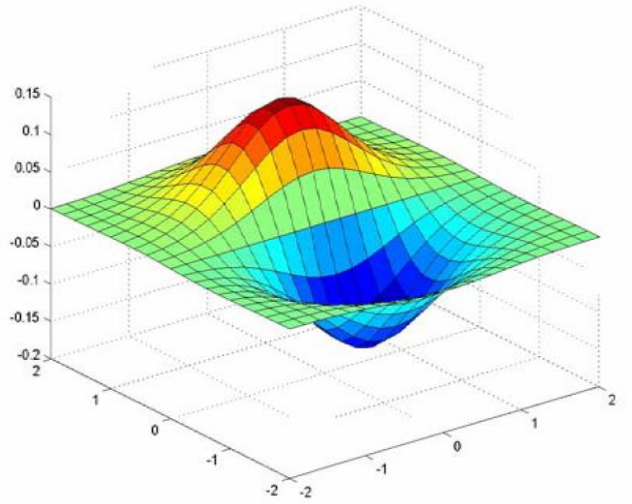
$$\begin{bmatrix} 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0219 & 0.0983 & 0.1621 & 0.0983 & 0.0219 \\ 0.0133 & 0.0596 & 0.0983 & 0.0596 & 0.0133 \\ 0.0030 & 0.0133 & 0.0219 & 0.0133 & 0.0030 \end{bmatrix} \otimes \begin{bmatrix} 1 & -1 \end{bmatrix}$$



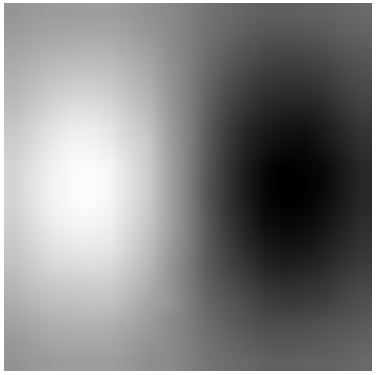
Derivative of Gaussian filters

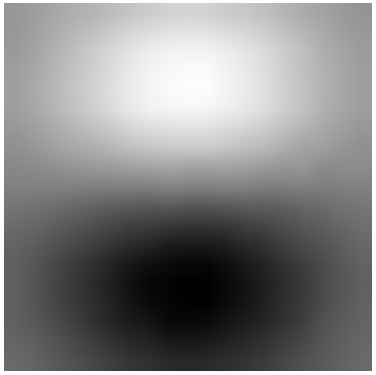


x-direction



y-direction





Source: L. Lazebnik

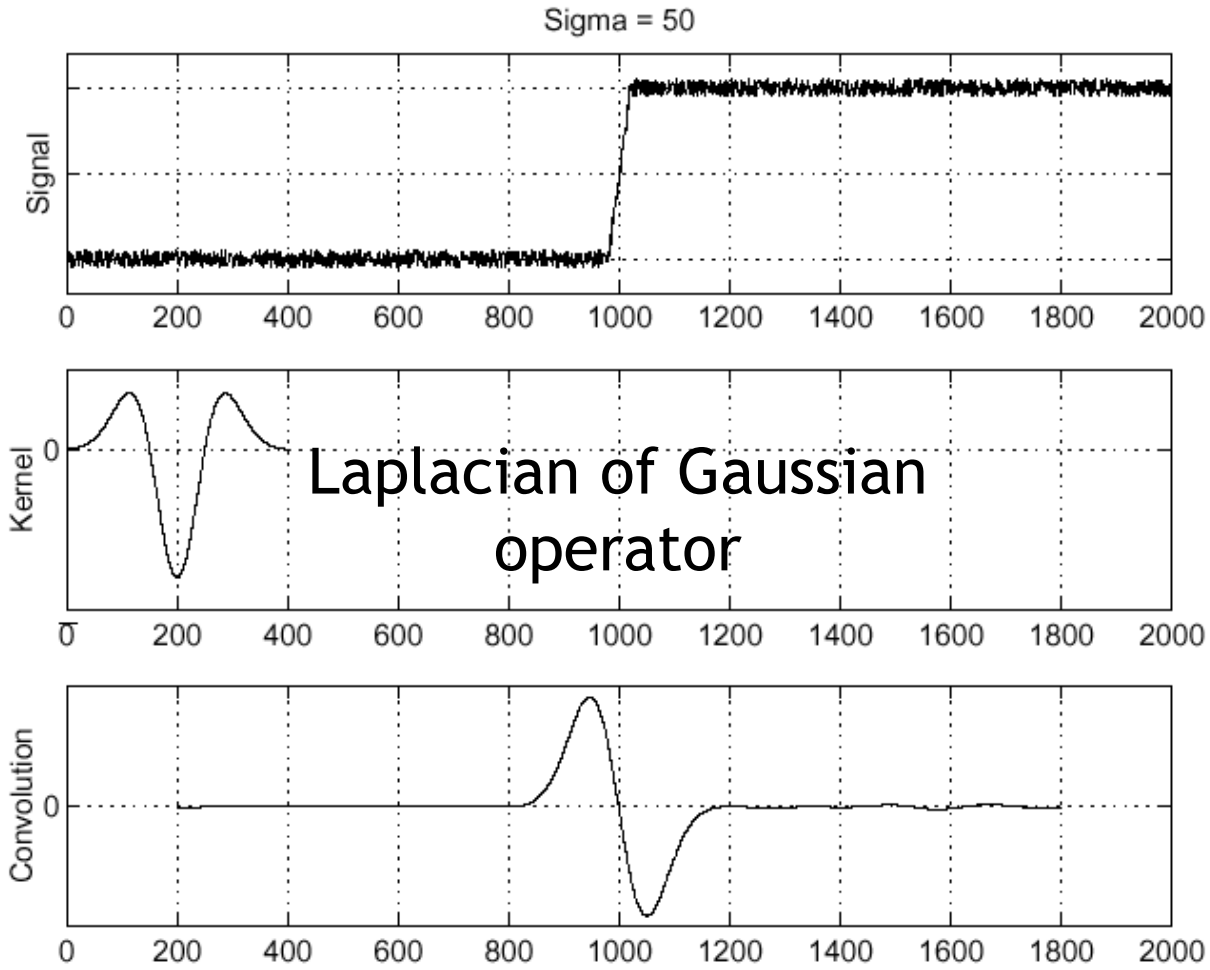
Laplacian of Gaussian

Consider $\frac{\partial^2}{\partial x^2}(h \star f)$

f

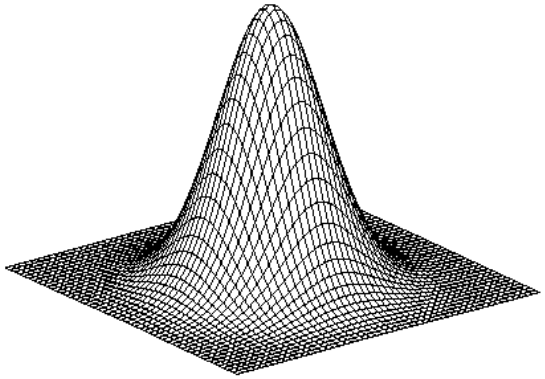
$\frac{\partial^2}{\partial x^2}h$

$(\frac{\partial^2}{\partial x^2}h) \star f$



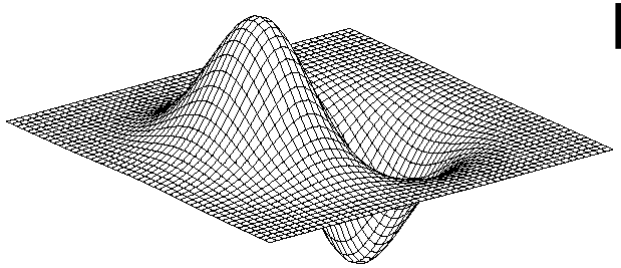
Where is the edge? Zero-crossings of bottom graph

2D edge detection filters



Gaussian

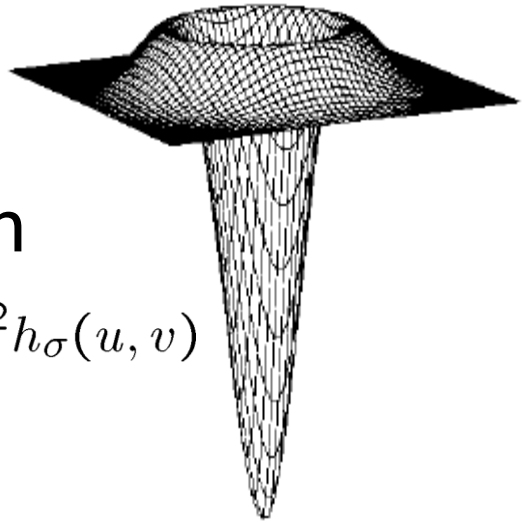
$$h_\sigma(u,v) = \frac{1}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$



Derivative of Gaussian

$$\frac{\partial}{\partial x} h_\sigma(u,v)$$

Laplacian of Gaussian



$$\nabla^2 h_\sigma(u,v)$$

- ∇^2 is the Laplacian operator:

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

Edge Detection w/ Laplacian Filter

0	1	0
1	−4	1
0	1	0

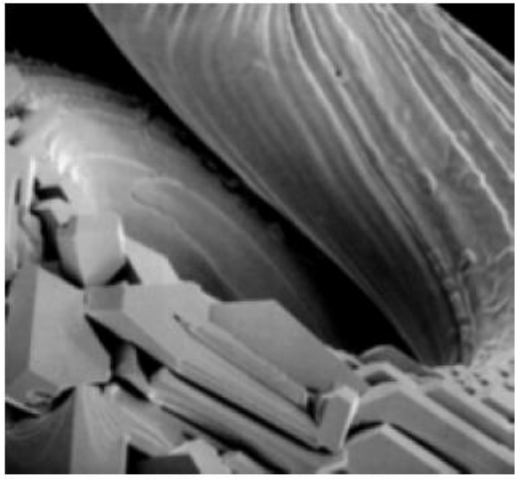
1	1	1
1	−8	1
1	1	1

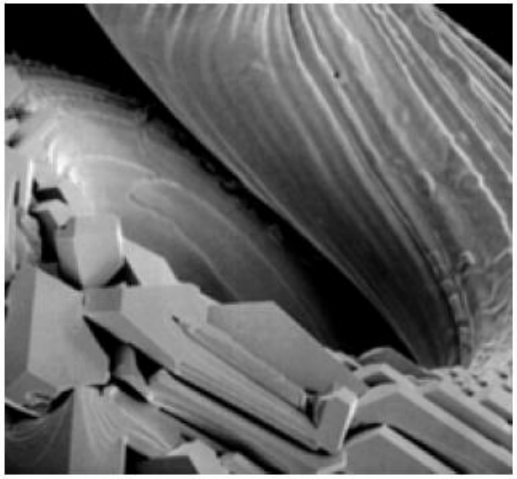
0	−1	0
−1	4	−1
0	−1	0

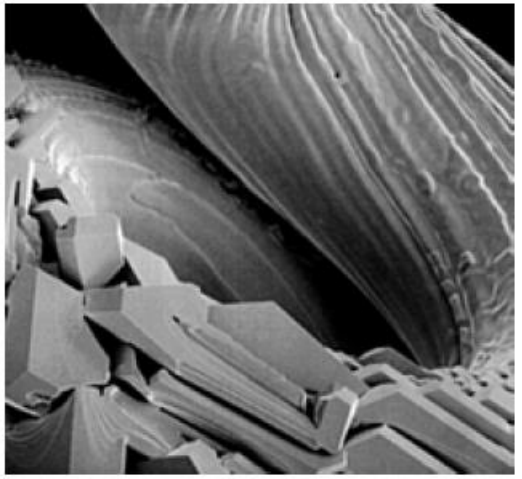
−1	−1	−1
−1	8	−1
−1	−1	−1

0	−1	0
−1	5	−1
0	−1	0

−1	−1	−1
−1	9	−1
−1	−1	−1





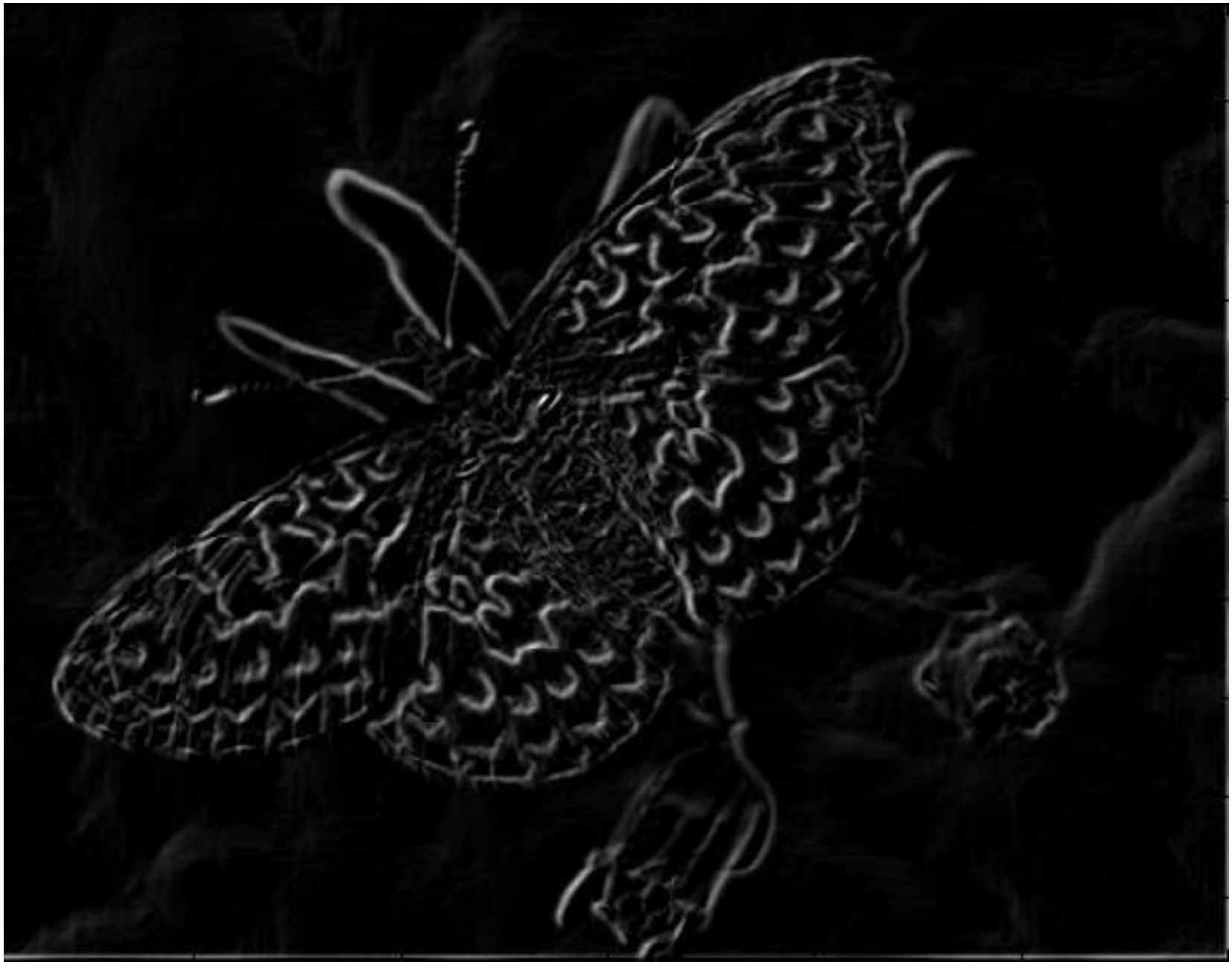


Primary Edge Detection Steps

1. Smoothing:
   - Suppress noise
2. Edge enhancement:
   - Edge detection filters
3. Edge localization
   - Determine which local maxima from filter output are actually edges vs. noise
   - Threshold: if pixel value < threshold, pixel value =0, otherwise pixel value=1

Original image

Gradient magnitude image



Thresholding gradient with a lower value

Thresholding gradient with a higher value

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
2. *Basic of Image*
3. ***Image Processing Algorithm***
   - *Elementary Processing*
   - ***Applications of Elementary Processing***
     : *Edge*, *Distance*, *Optical Flow* *Detection*

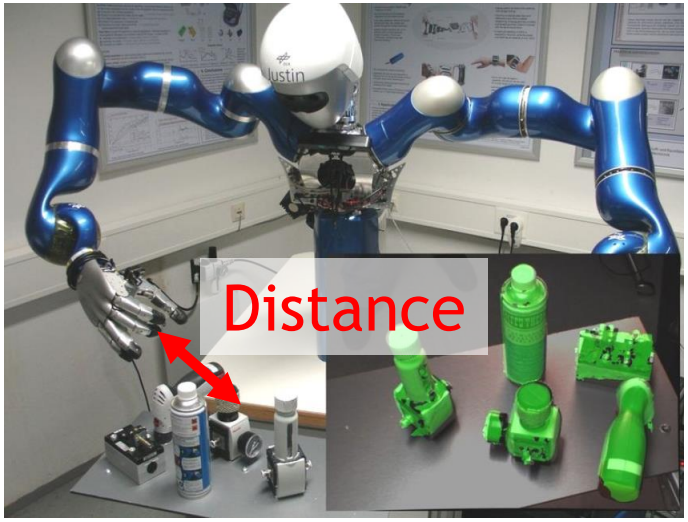
-Practical Part-

1. *Practical Algorithms for "Robot War"*

Advanced Image Processing – Distance Estimation

- Estimating distance Information is Essential!
  - Estimate the distance from the objects to move it
  - Estimate the distance from the obstacles to avoid it
- Time of flight (ToF) or stereo matching can be utilized

Distance information can be used for …

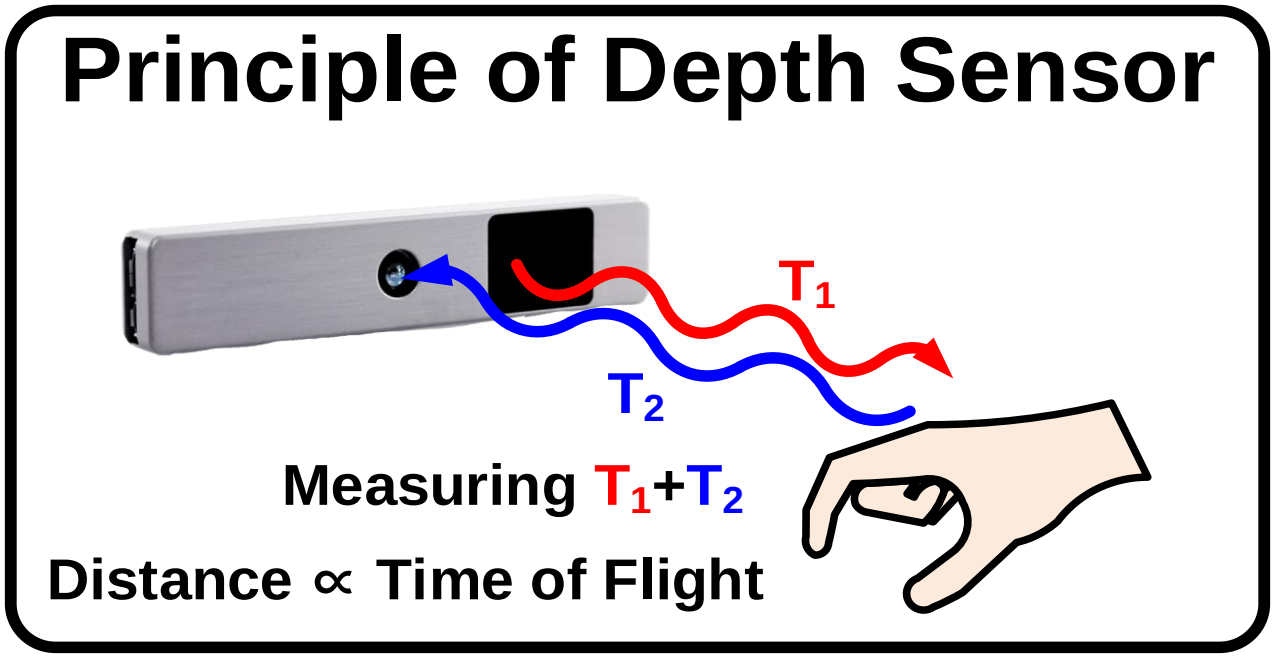


moving objects

avoid obstacles

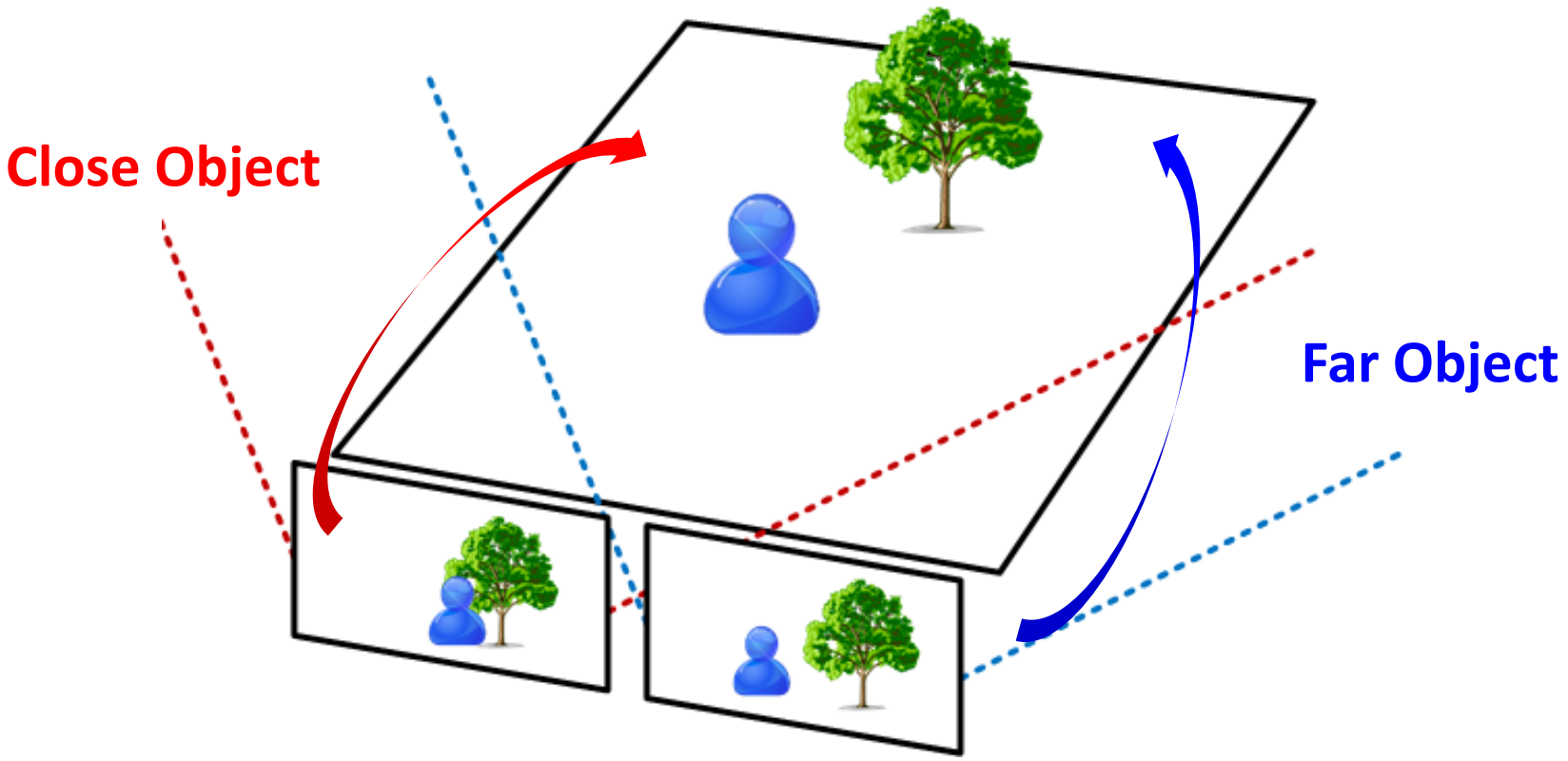
Time of Flight (ToF)

- Basic idea: send out pulse of light (usually laser), time how long it takes to return
- Accurate, but large power consumption



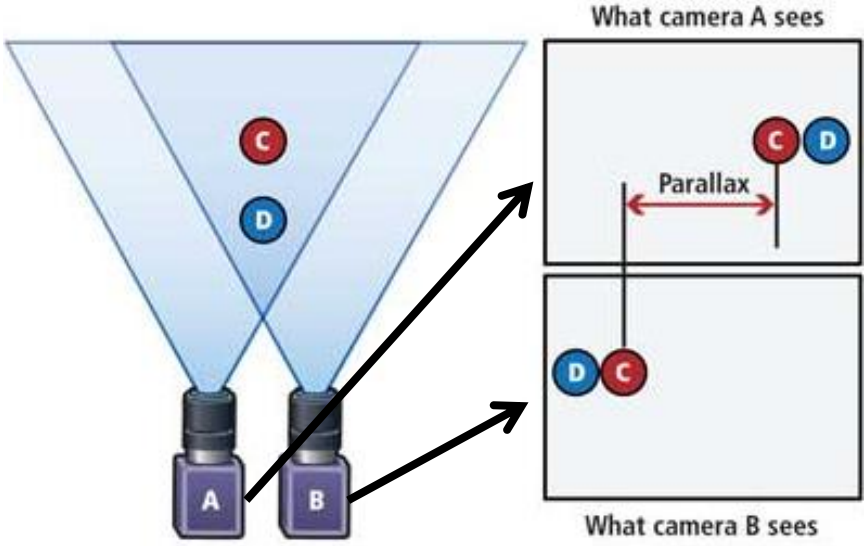
Stereo Matching

- Basic Idea : Inferencing depth from two different views
  - Close object ➔ Large displacement
  - Far object ➔ Small displacement



How can we do stereo matching?

- By Using Stereo Camera
  - Measure the pixels distance from a left and right camera scene and estimate the depth using the ratio of equivalent triangles

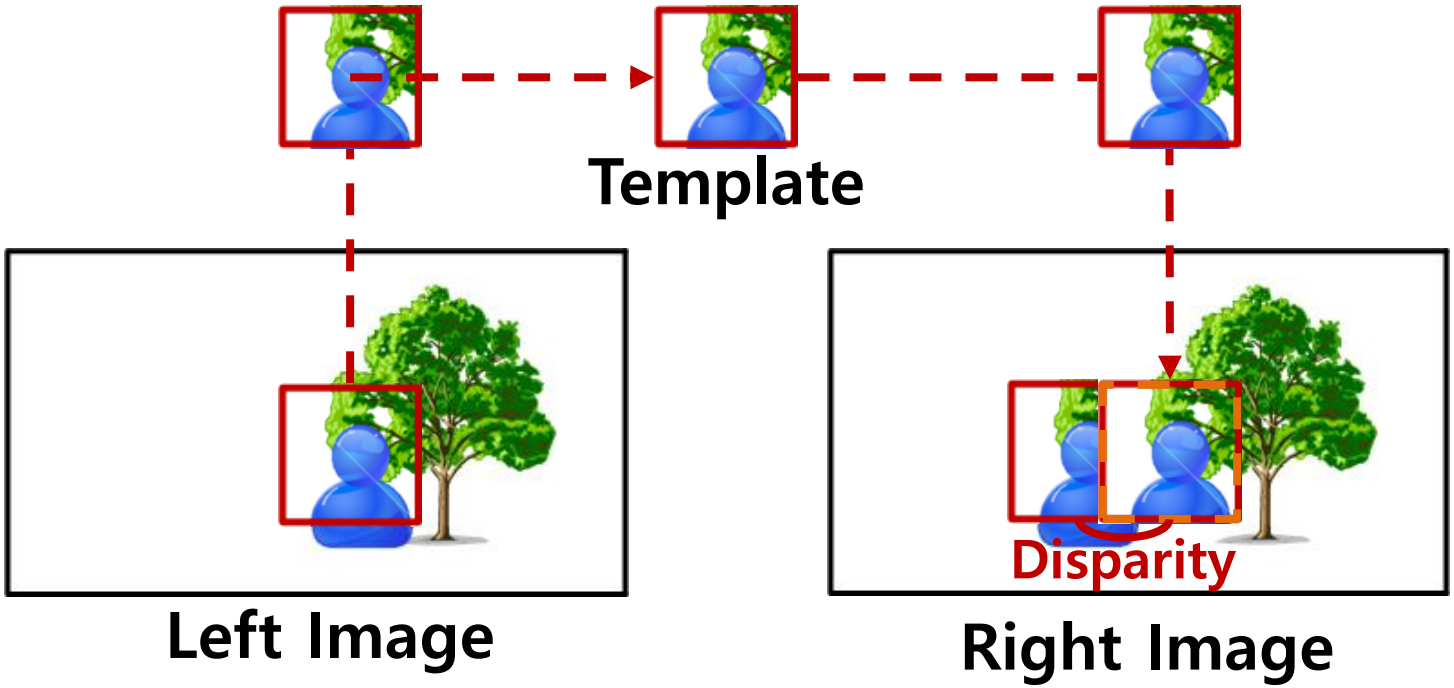


Left Camera Right Camera

Depth Map

Detail of Stereo Matching Process

- Sliding Window Processing
  - Find best matched disparity between L & R images



Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. Introduction of CV for robot
2. Basic of Image
3. **Image Processing Algorithm**
   - Elementary Processing
   - **Applications of Elementary Processing**
     : Edge, Distance, **Optical Flow Detection**

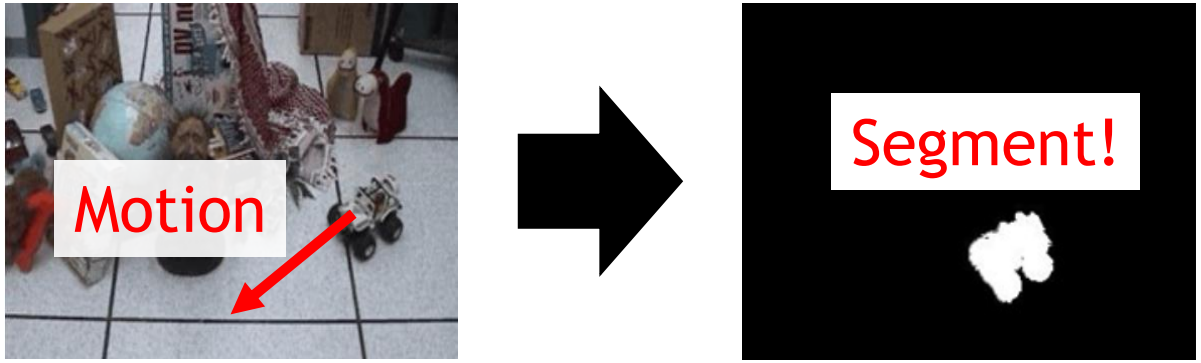
-Practical Part-

1. Practical Algorithms for “Robot War”

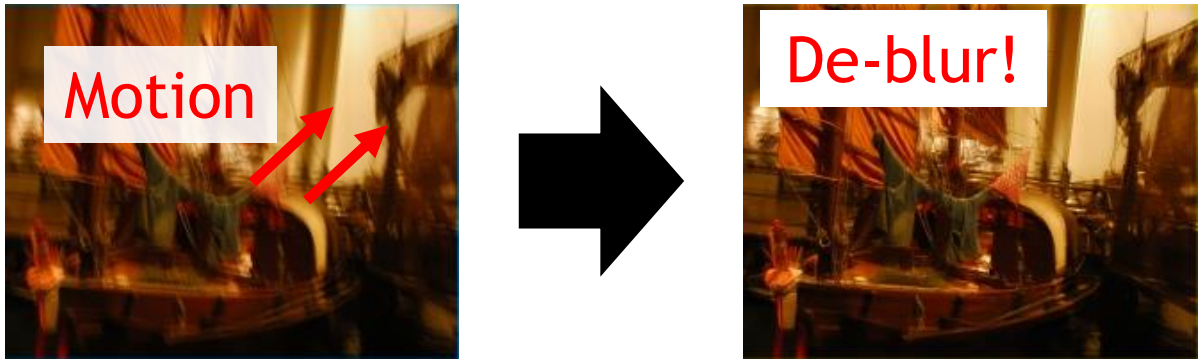
Advanced Image Processing - Motion Information

- Motion Information is also useful in robot!
  - Segment only moving object from input video streams
  - Human action recognition from input video streams
  - Motion de-blurring on moving camera environment

Moving Object Segmentation

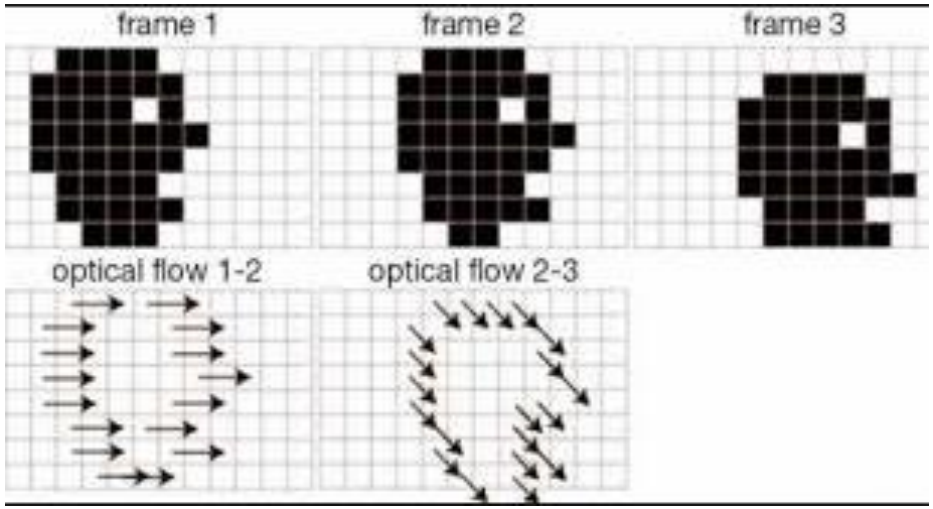


Motion De-blurring



How can we get motion information?

- By Estimating Optical Flow!
  - Trace the object movement
  - Calculate the different pixels intensity from the adjacent frames and estimate the gradient components .

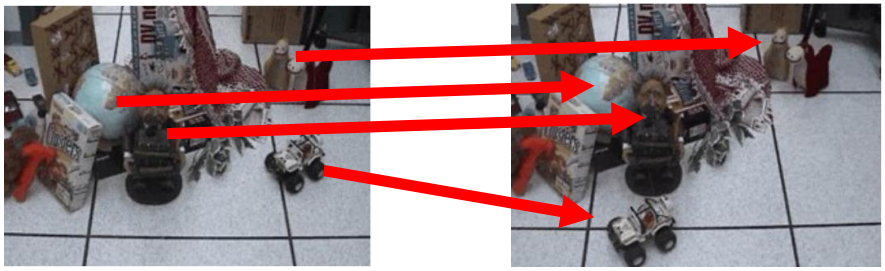


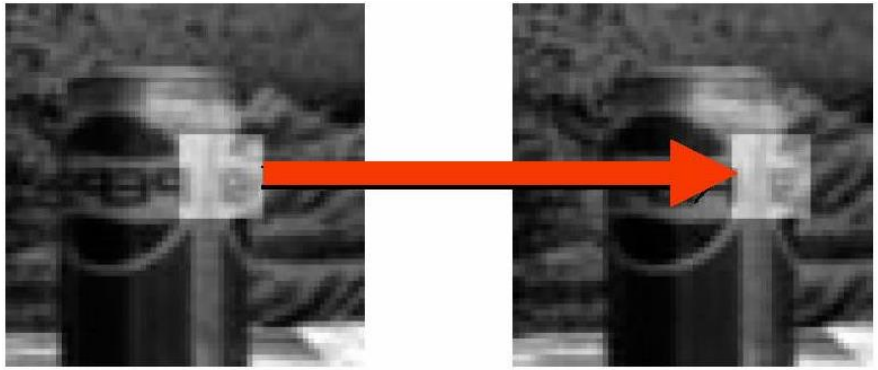
Original Image

Optical Flow Result

Detail of Optical Flow Estimation Process

- Basic idea : Choose a small patch in previous frame (t), and find similar one in next frame (t+1)
  - Assumption : The appearance of the image patches do not change (brightness constancy)





$$\frac{dI}{dt} = 0$$

I(x,y) = grey level value
Velocity vector is (v_1, v_2)

$$\frac{\partial I}{\partial x} v_1 + \frac{\partial I}{\partial y} v_2 + \frac{\partial I}{\partial t} = 0$$

Outline

-Computer Vision (CV) for Intelligent SoC Robot-

-Theoretical Part-

1. *Introduction of CV for robot*
2. *Basic of Image*
3. *Image Processing Algorithm*

-Practical Part-

1. *Practical Algorithms for "Robot War"*
   : Which CV techniques will help you?

Image Processing and Image Recognition

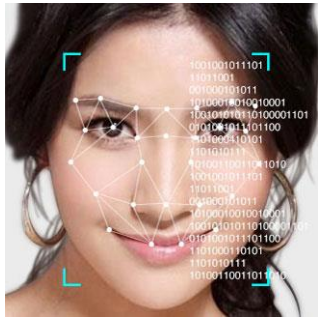
- **Image Processing**
  - Subcategory of digital signal processing
  - Computer algorithms to obtain the image information from images
    - Ex) Resolution enhancement, image restoration, compression, recognition, etc.

1920s	• Transfer the newspapers between America and Europe ➔ Beginning of image processing
1964	• The Jet Propulsion Laboratory ➔ Image restoration from satellite images
1980s	• Development of medical imaging like CT, MRI
1990s	•The fast computers ➔ The common for the computer vision and graphics broadcasting

Image Processing and Image Recognition

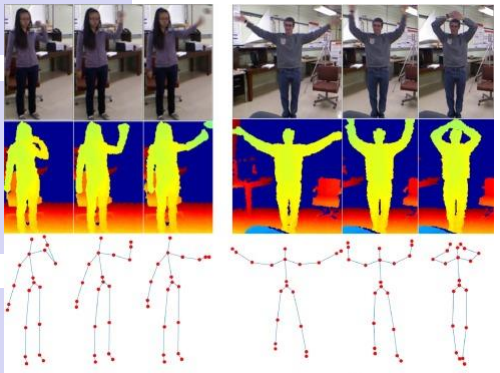
- **Image Recognition**
  - Identify the types of and numbers of objects from an image

• Face recognition
➔ Security, digital camera, smartphone

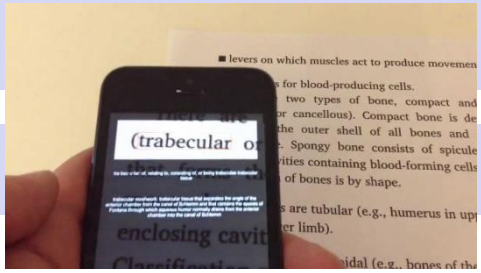


• Lane line recognition
➔ Autonomous navigation (Driving, Parking)

• Environment recognition
➔ Robot, forest fire



• Action recognition
➔ Game



• Text recognition

SoC Robot – Image Processing

Processing/ Recognition	Methods / Results	Remarks
Color Conversion	Because RGB format is vulnerable to illumination change, YCbCr or HIS is mainly used for image processing / recognition	**Brightness changes with place / time**
Noise Reduction	Remove noise from the input video from camera	**Camera noise**
Opponent Robot recognition	Because Color-based recognition is sensitive to environment variation, image recognition using feature extraction is required	**Environment changes with place / time**
Opponent Robot motion recognition	By recognizing the motion (movement) of the robot, it is possible to predict the next motion, thereby determining the attack / defense	**Attack / defense strategy**
Location recognition	Recognize the opponent robot's and my position for various strategies	**Attack / defense strategy**
Distance recognition	By measuring the distance between the robot and my position, it can estimate the distance that can be attacked or defended, and can take various attack or evasion actions	**Attack / defense strategy**

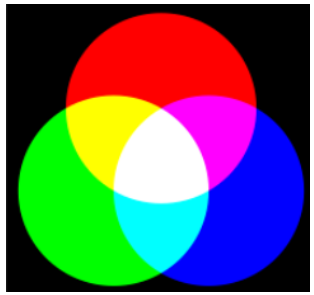
Program Analysis

Source Code Analysis for Intelligent SoC Robot (HDL: H / C language: C)

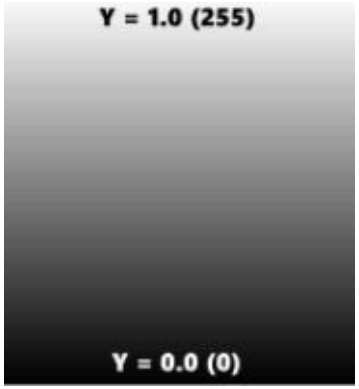
Technique	Details	Team								
		A	B	C	D	E	F	G	H	I
Filtering	Noise Filtering	H		C	C	C		C		
Color Conversion	Resolution Conversion		H			H			H	
	Format (RGB-YUV) Conversion		H	H	C	H	H		H	C
Color Recognition	Black line Recognition	H	C	C		C	H	H	H	
	Opponent Robot Recognition	H	C	C	C	C		C		C
Motion Recognition	Opponent Attack Motion Recognition									
	Opponent Defense Motion Recognition									
	Opponent Standing Motion Recognition							C		
	Opponent Sitting Motion Recognition							C		
Location Recognition	Opponent Location Recognition	H	C	C	C	C	H	C	C	C
Distance Recognition	Opponent Distance Recognition	H	C	C	C	C	H	C		
Object Tracking	Opponent Object Tracking	H	C	C	C					

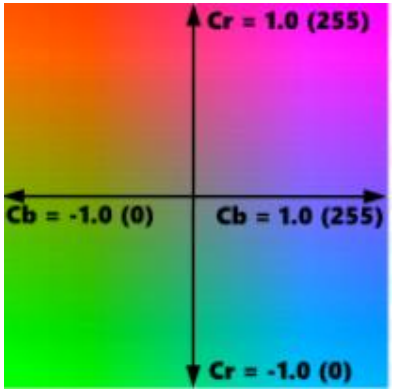
Image Processing - Color

- **RGB**
  - Mix colors of three primary colors, RED, GREEN, BLUE
  - Sensitive to brightness
  - Correlation of color elements is too large



- **YCbCr**
  - Y: Luminance, Cb, Cr: Difference chrominance components
  - Robust to brightness by using Cb and Cr for color recognition
  - Compression effect in color information compared to RGB

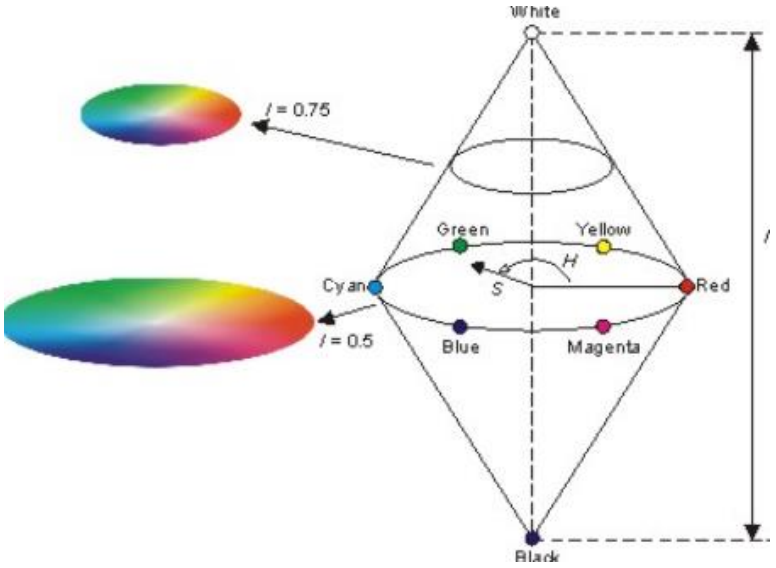




$$\begin{pmatrix} Y \\ U \\ V \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ -0.147 & -0.289 & 0.436 \\ 0.615 & -0.514 & -0.101 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix}$$

Color Processing

- **HIS**
  - Same way human feels color
  - Hue: The feeling of color. 6 basic colors (red, yellow, green, cyan, blue, magenta) are interpolated to make other colors
  - Saturation: Degree of color how much diluted with white color
  - Intensity: The intensity (the amount of light)



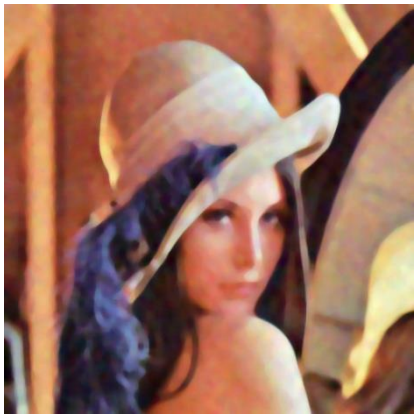
Spatial Filter

- **Uses one pixel of the input image and its surrounding pixels as the input**

Input Image

h_1	h_2	h_3
h_4	h_5	h_6
h_7	h_8	h_9

Mask



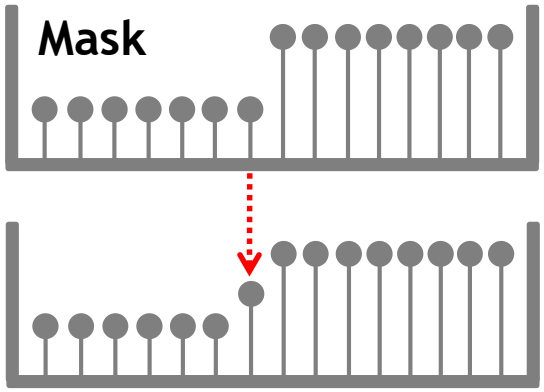
Filtering Result

- Since the image is a two-dimensional signal, the filter used in the image processing is also two-dimensional
- Constructing spatial filter is called as “Mask” or “Window”

Spatial Filter

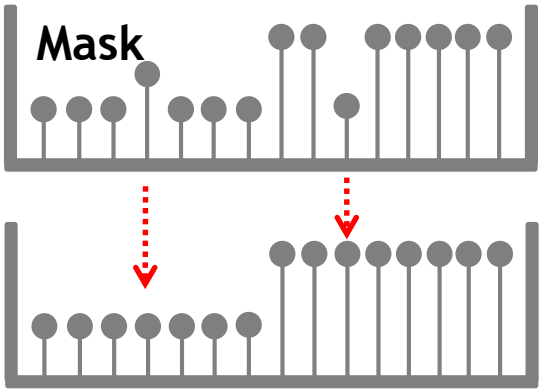
- **Image Smoothing**
  - A preprocessing that eliminates image noise and softens the image

- **Low-pass filter**
  - Eliminate high frequency components
  - Smoothing original Image



Low pass filter

- **Median Filter**
  - Sort by size and select the median value
  - Preserve the edge and remove noise

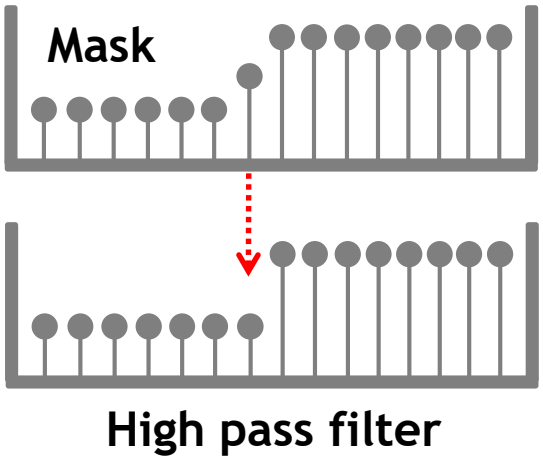


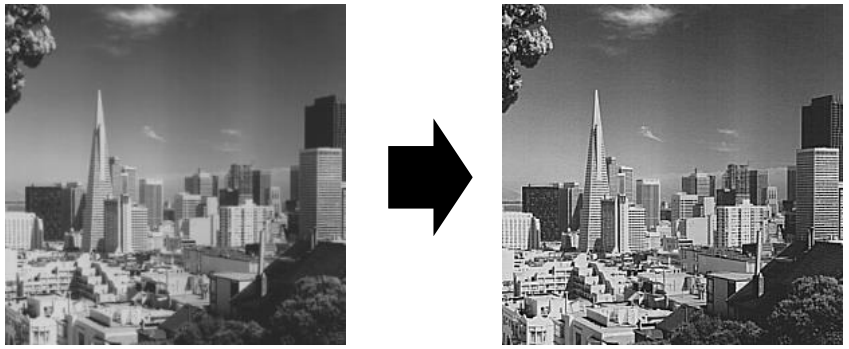
Median filter

Spatial Filter

- **Image Sharpening**
  - Improvement of the specific information of the image

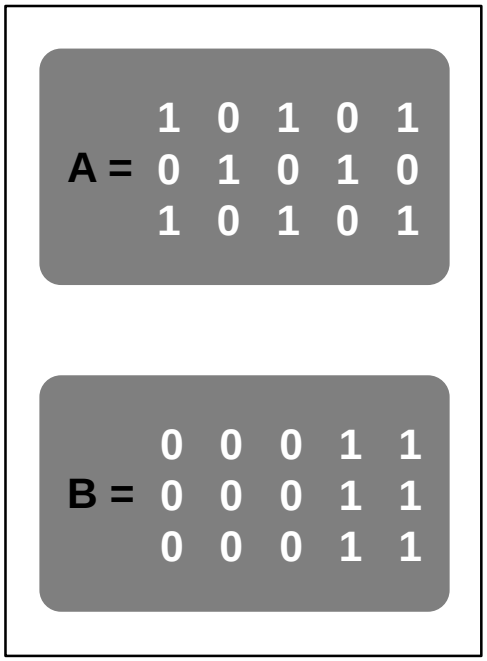
- **High-pass Filter**
  - Eliminate low frequency components
  - Emphasize the edges





Morphology

- **Process that changes the shape while maintaining the basic characteristics of the image**
  - Operation of Erosion and Dilation
  - Operation of Opening and Closing
  - Used for image segmentation and preprocessing
    - Ex) noise reduction, feature extraction



Union Operation
- A ⊕ B, A ∪ B

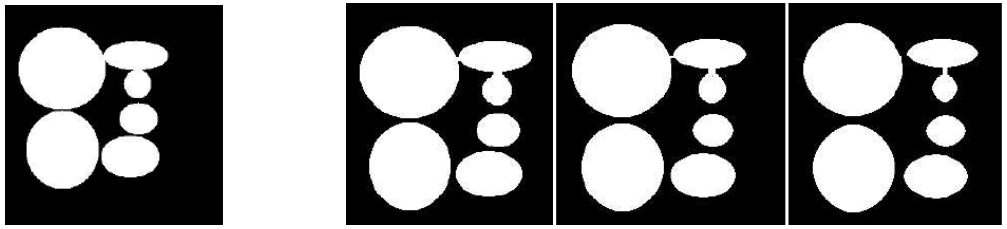
$$\triangleq \doteq \begin{matrix} 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \end{matrix}$$

Intersection Operation
- A ⊗ B, A ∩ B

$$\triangleq \doteq \begin{matrix} 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{matrix}$$

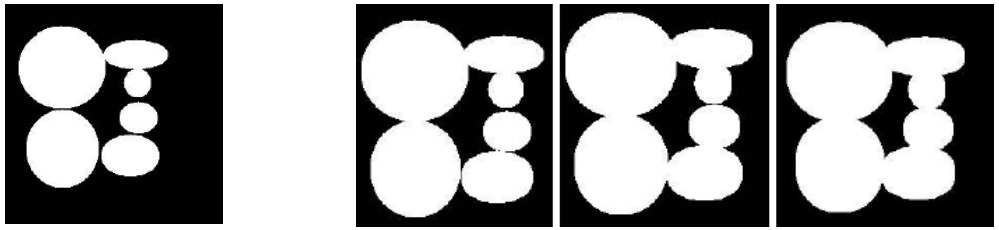
Morphology (cont.)

- **Operation of Erosion**
  - Intersection operation when template and binary image are matched
  - Remove the edge protrusion, and reduce the size of the object



Original Image **1 Erosion** **2 Erosion** **3 Erosion**

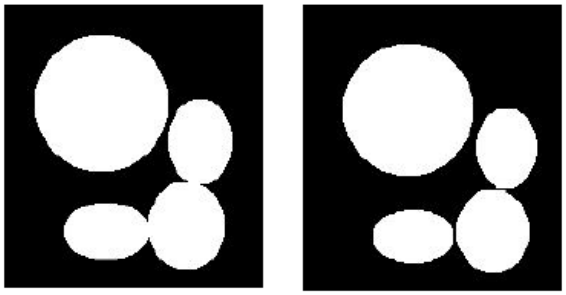
- **Operation of Dilation**
  - Union operation when template and binary image are matched
  - Remove the depth of the hole or edge, and enlarge the size of the object



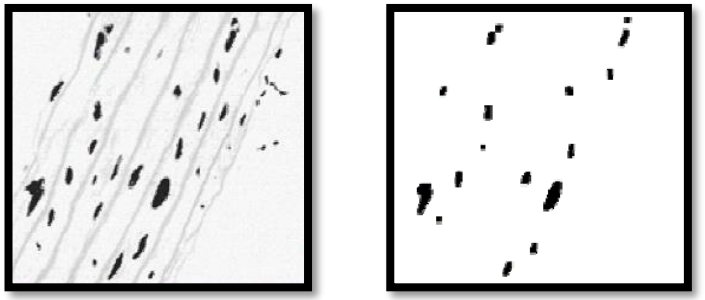
Original Image **1 Dilation** **2 Dilation** **3 Dilation**

Morphology (cont.)

- **Operation of Opening**
  - Apply the dilation operation after the erosion
  - Useful for separating the objects

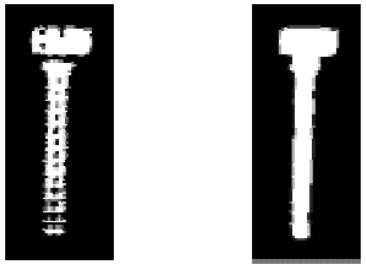


Original Image After Opening

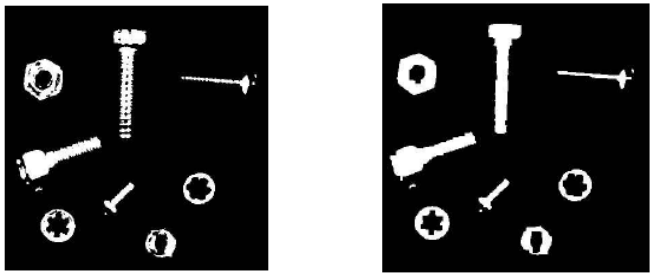


Original Image After Opening

- **Operation of Closing**
  - Apply the erosion operation after the dilation
  - Use for combining the objects



Original Image After Closing



Original Image After Closing

SoC Robot – Image Processing

- **Edge Detection**
  - Edge or contour
  - Estimated as an edges if the brightness difference between neighboring pixels exceeds the certain threshold
- **Condition for edge detection mask**
  - Width and height of mask have to be equal, and odd numbers
  - Vertical symmetry based on center coefficient
  - The values of the center coefficient are always positive or zero
  - The others of the center coefficient are negative
  - The sum in all coefficients is zero

(a) Original image (b) Robert operator (c) Sobel operator (d) Prewitt operator

(e) Laplacian operator (f) Kirsch operator (g) Robinson operator

SoC Robot – Image Processing

- **Sobel Mask**
  - Widely used for edge detection with weighting the pixel edges
  - Edge detection in all direction is possible and strong in noise

−1	0	1
−2	0	2
−1	0	1

Vertical Mask

1	2	1
0	0	0
−1	−2	−1

Horizontal Mask

- **Prewitt Edge Detection**
  - Fast responsibility
  - Less detecting edge due to smaller weighting than Sobel mask
  - More sensitive to vertical / horizontal edge than diagonal

−1	0	1
−1	0	1
−1	0	1

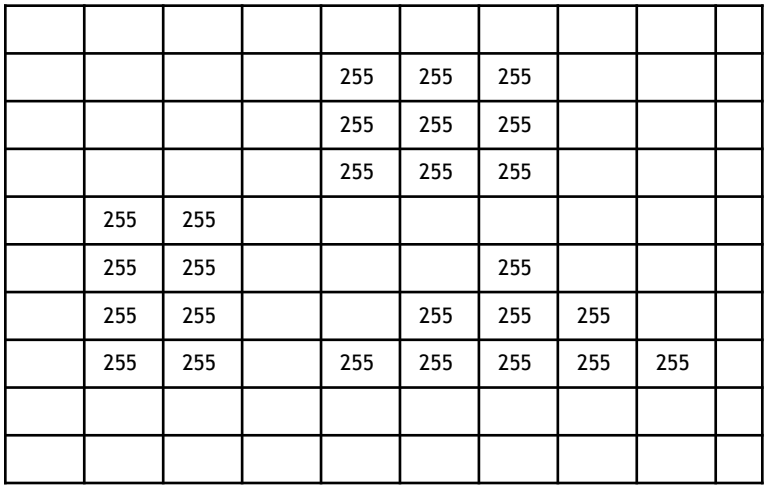
Vertical Mask

1	1	1
0	0	0
−1	−1	−1

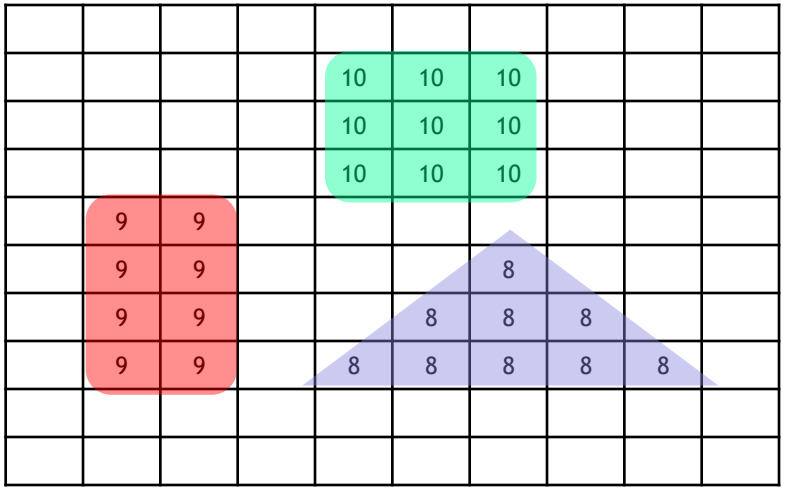
Horizontal Mask

SoC Robot – Image Processing

- **Labeling**
  - Labeling the same number to all adjacent pixels and another number to the other segments



Grayscale Image



Labeling Image

- **Grassfire Algorithm**
  - Algorithm for labeling pixels like spreading fire in the grass

SoC Robot – Image Processing

- **Feature: Geometric properties of the object for recognition and separation**
- **Features for recognizing objects**
  - **Average brightness**
  - **Max. and min. brightness**
  - **Area / Thinness / Shape Ratio**
  - **Circumference:** the total number of pixels around the region
  - **Diameter:** The farthest distance between two pixels that are at the edge of an object
  - **Center of gravity:** The average of x- and y-coordinate pixels in the region
  - **Moment:** Significant direction features in the center of gravity coordinate
  - **The degree of curvature** around the object
  - **Hole** in the object / Chain code

SoC Robot – Image Processing

- **Recognition**
  - Finding and identifying the target objects in the region
- **Template matching**
  - A recognizing method of comparing with the standard model which stored previously an image of an object or processed results
  - Recognition fails easily from different camera angle and distance
- **Bayesian theorem**
  - Impossible to obtain the same recognition results with same object features due to the method's stochastic characteristic.
- **Neural network**
  - **Training:** Adjust the connection weights by the pattern classes
  - **Inference:** Classify the nearest class by calculating the distance from the input vector and trained vectors
  - => Learning skills like human: Pattern Classification, Recognition, Optimization, Prediction

SoC Robot – Image Processing

- **Object Tracking**
  - Track the path of an arbitrarily moving object
  - Use position and size of the object in the frame of the image
- **Mean-Shift**
  - Find new center by comparing histogram in certain region with histogram in the next frame
- **CAMShift (Continuous Adaptive Mean Shift)**
  - Extend Mean-Shift Algorithm to the video image
  - Robust to rotation and zooming
  - Fast, Simple, Efficient