

# Basics of Image Processing

**Prof. Mingkui Tan**

SCUT Machine Intelligence Laboratory (SMIL)



SMIL内部资料 请勿外泄

# Contents

1 Introduction to Image Processing

2 Image Processing Applications

3 Image Filtering

SMIL内部资料 请勿外泄

# Contents

1 Introduction to Image Processing

2 Image Processing Applications

3 Image Filtering

SMIL内部资料 请勿外泄

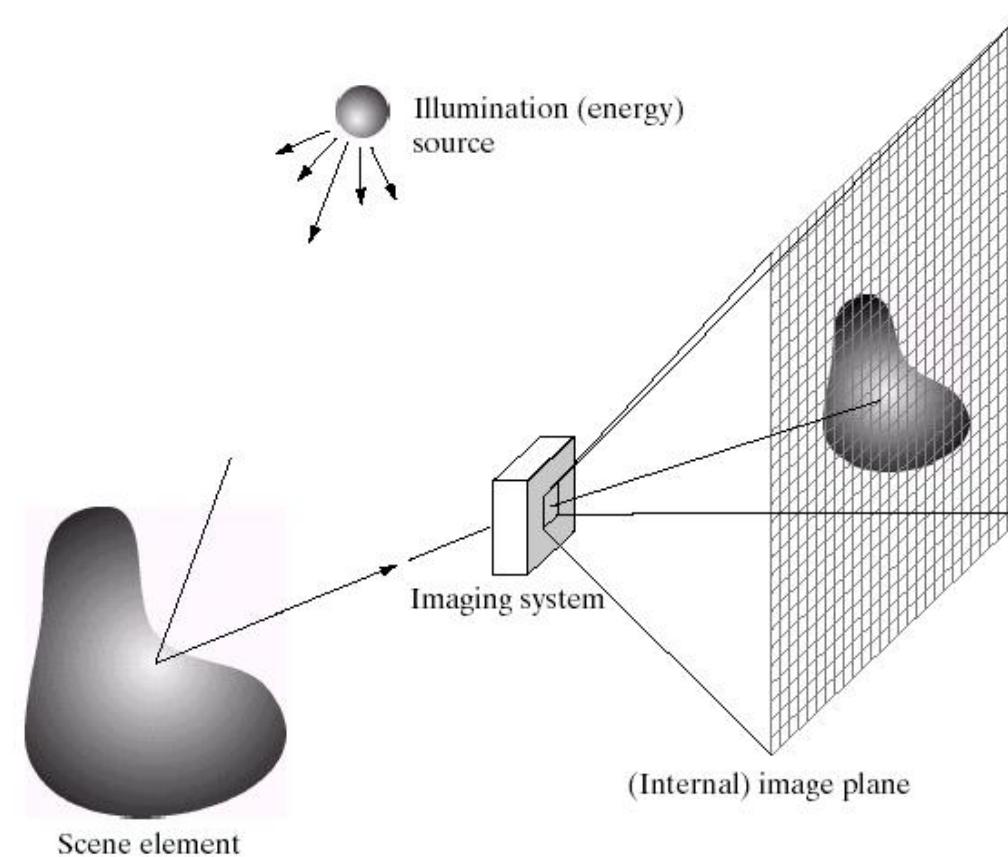
# Digital Camera



A digital camera replaces film with a sensor array

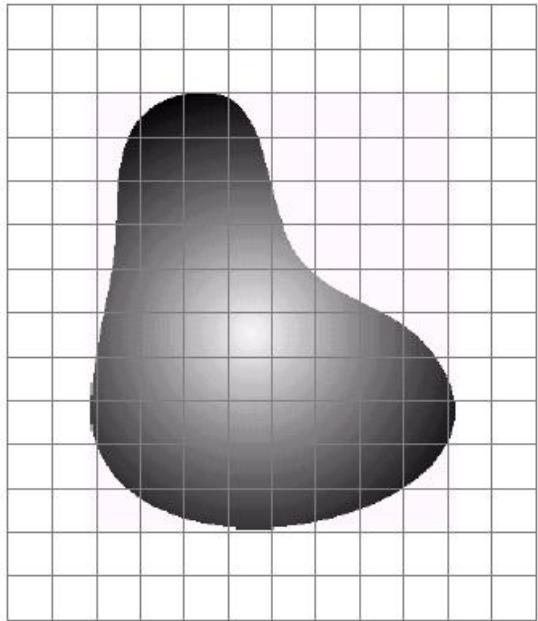
- Each cell in the array is light-sensitive diode that converts photons to electrons
- Two common types: Charge Coupled Device (CCD) and CMOS
- <http://electronics.howstuffworks.com/digital-camera.htm>

# How Light is Recorded

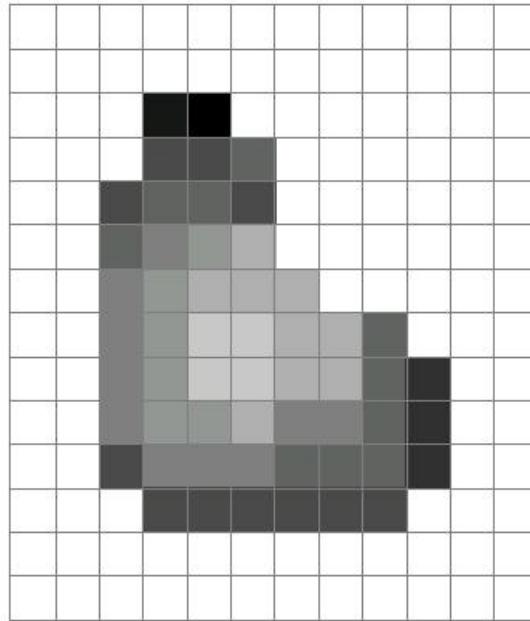


SMIL内部资料 请勿外泄

# Sensor Array

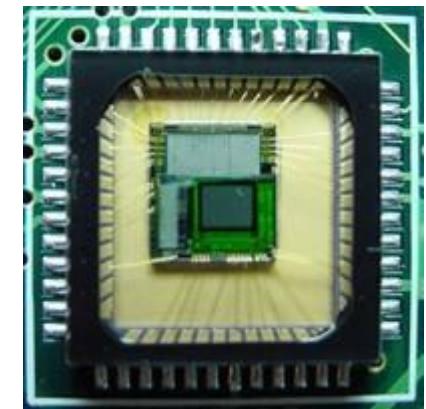


(a)



(b)

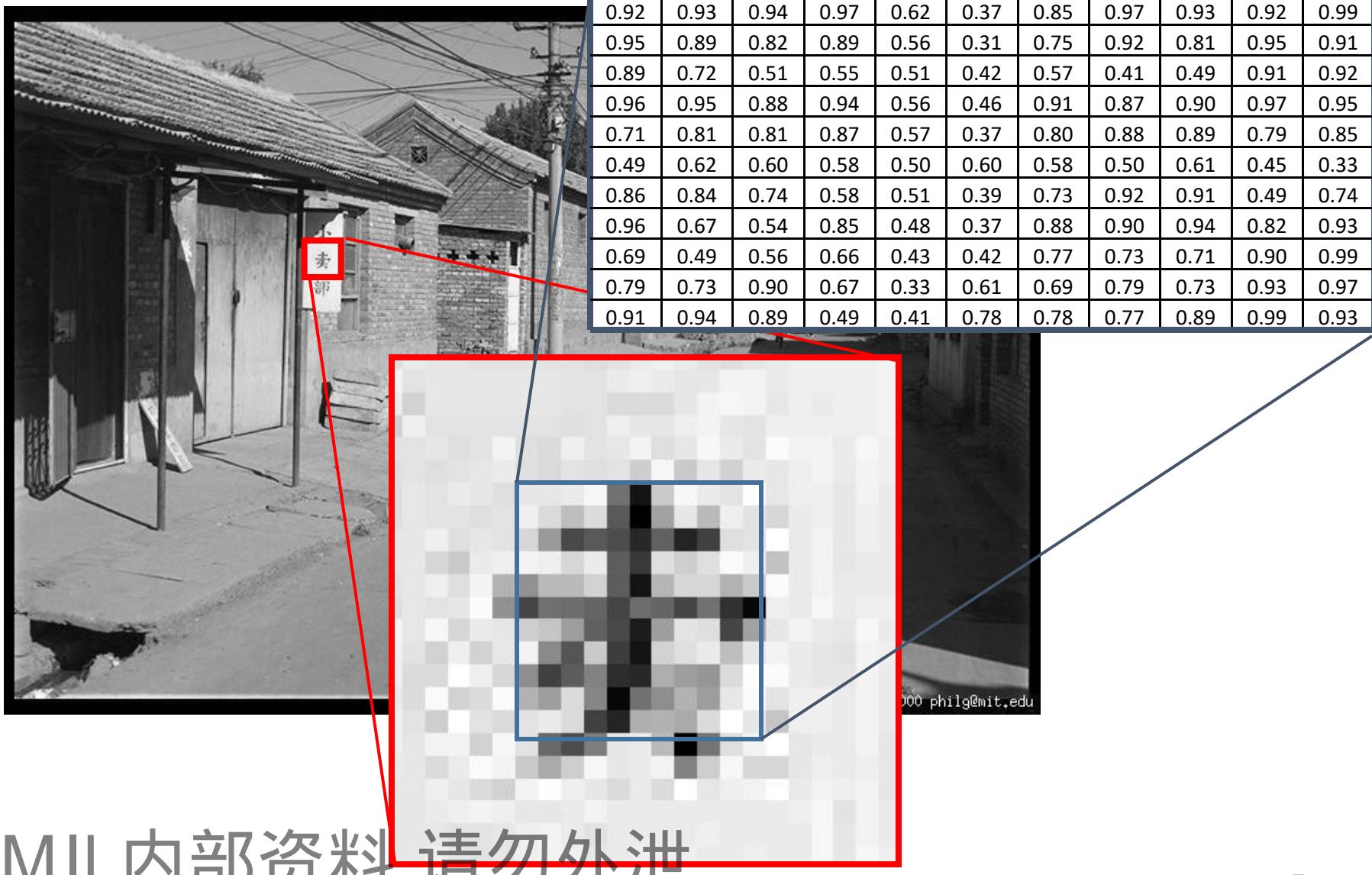
- (a) Continuous image projected onto a sensor array
- (b) Result of image sampling and quantization



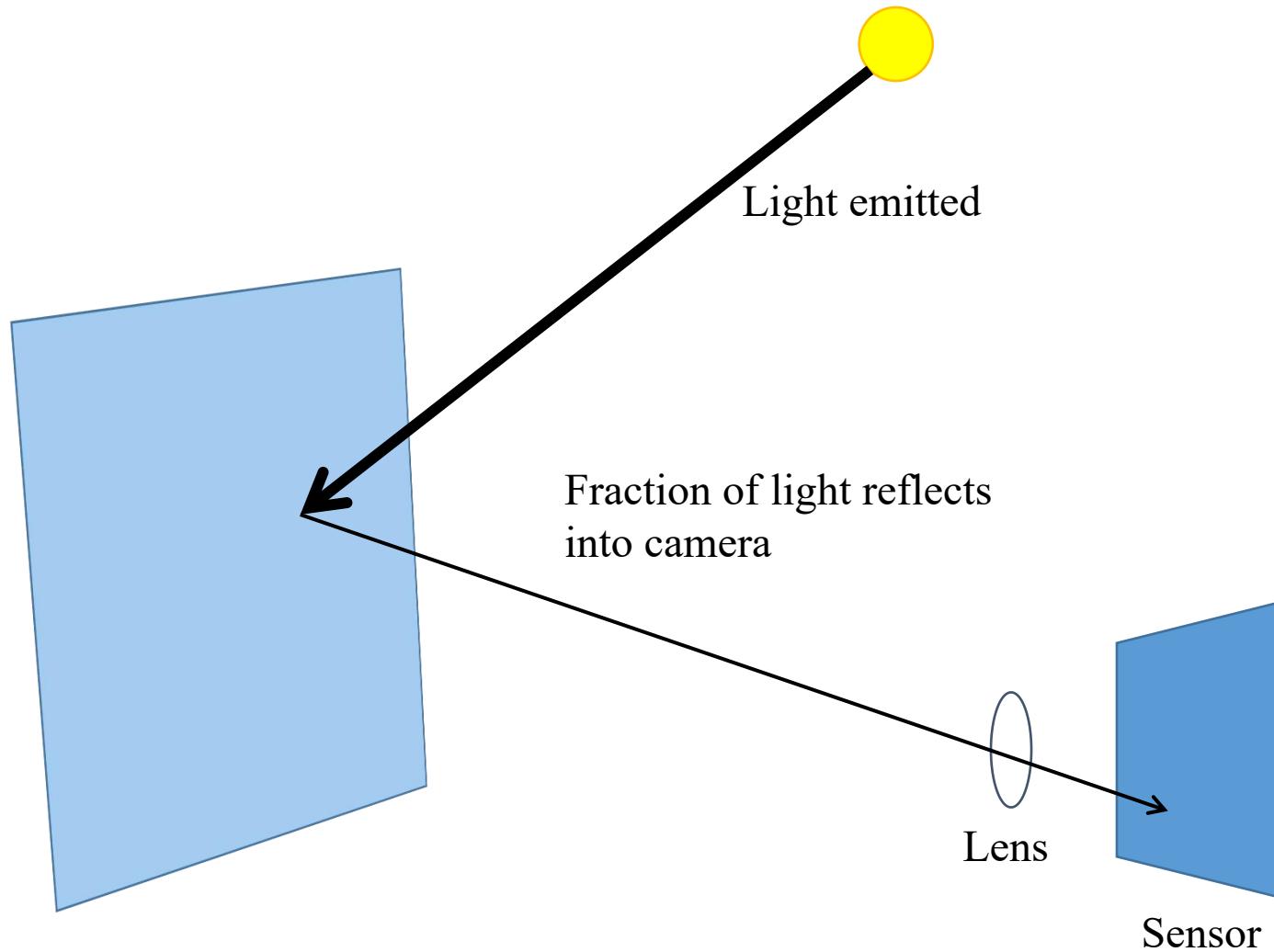
CMOS sensor

- Each sensor cell records amount of light coming in at a small range of orientations

# Raster Image (Pixel Matrix)



# How does a Pixel Get its Value?

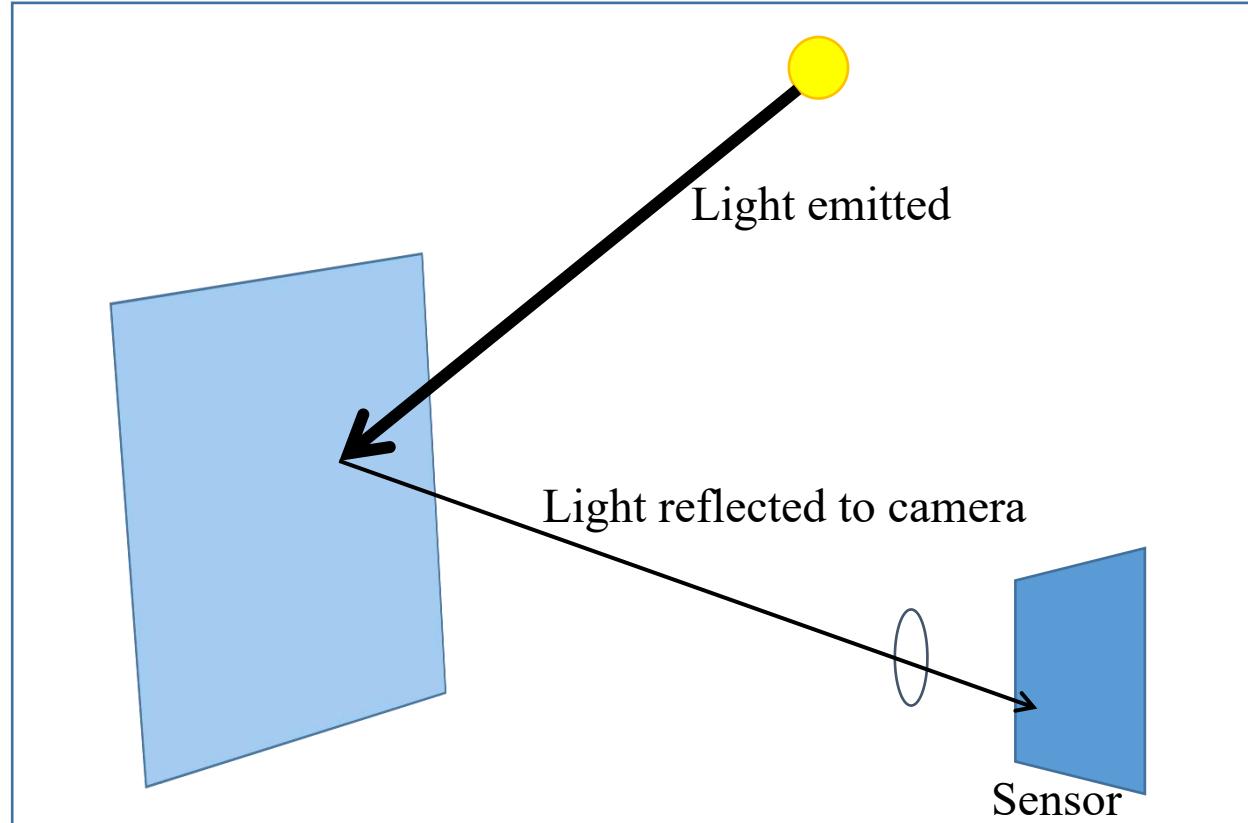


SMIL内部资料 请勿外泄

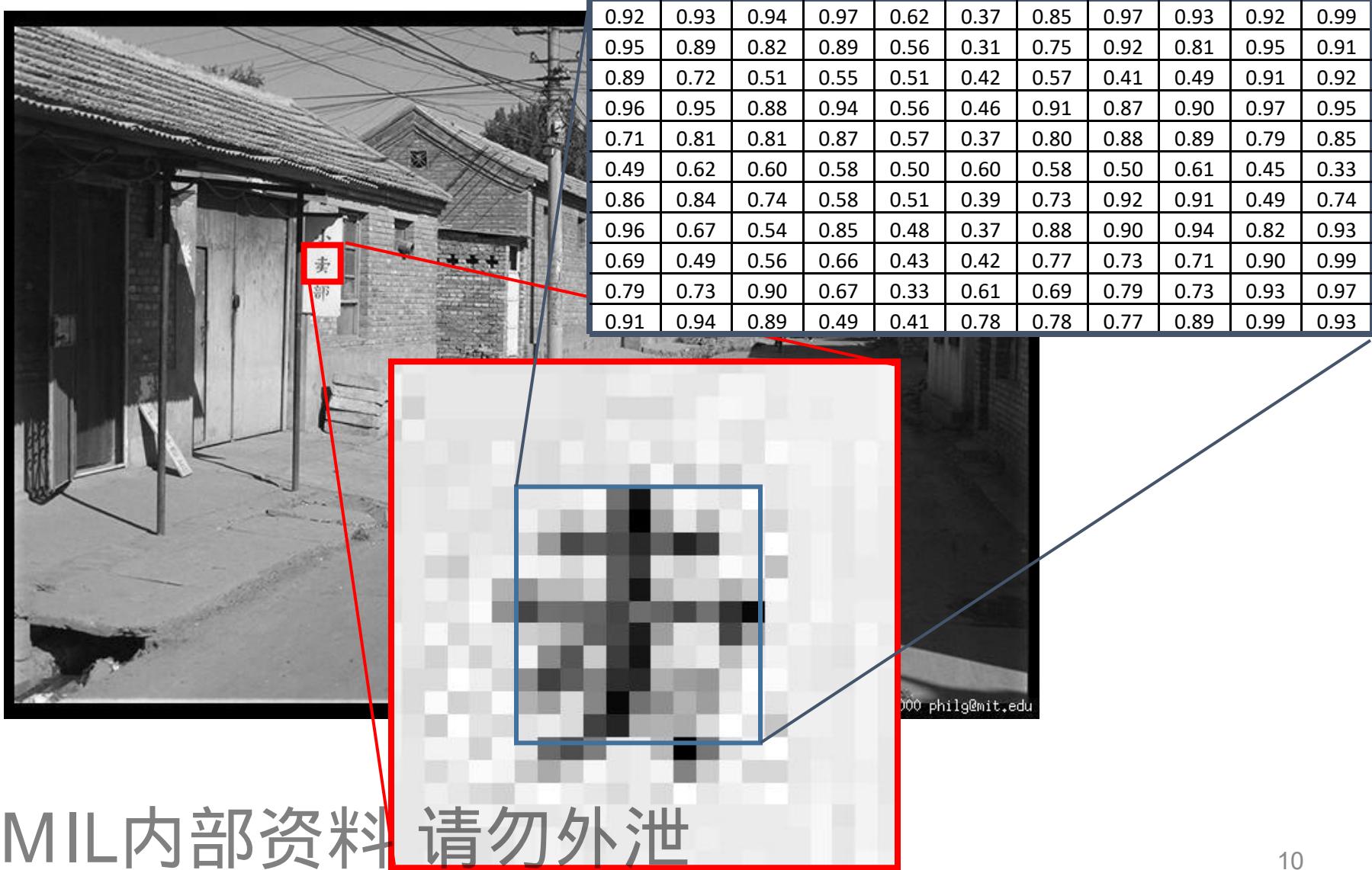
# How does a Pixel Get its Value?

## ■ Major factors

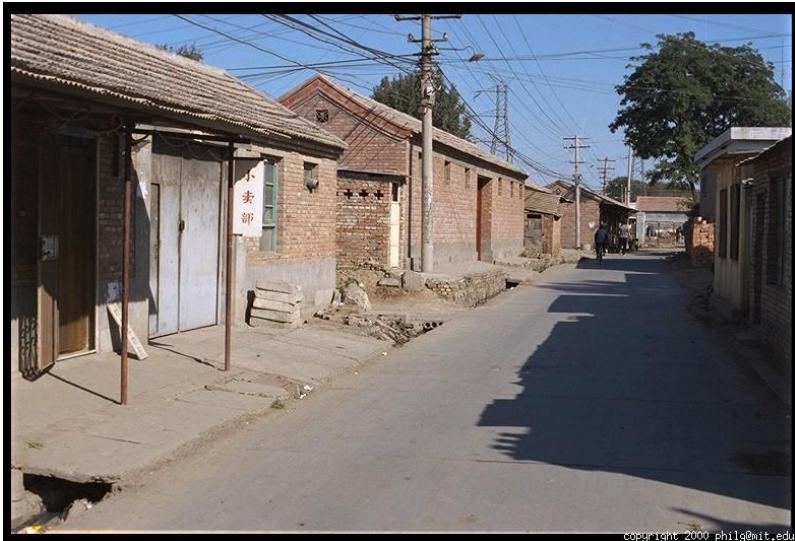
- Illumination strength and direction
- Surface geometry
- Surface material
- Nearby surfaces
- Camera gain/exposure



# Gray Image (One Single Matrix)



# Color Image



G



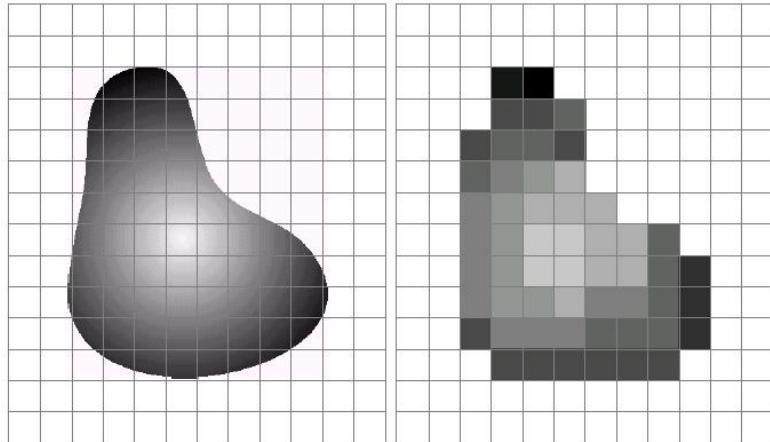
B



copyright 2000 philg@mit.edu

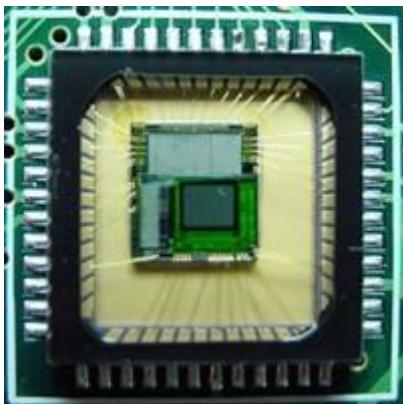
SMIL内部资料 请勿外泄

# Digital Color Images



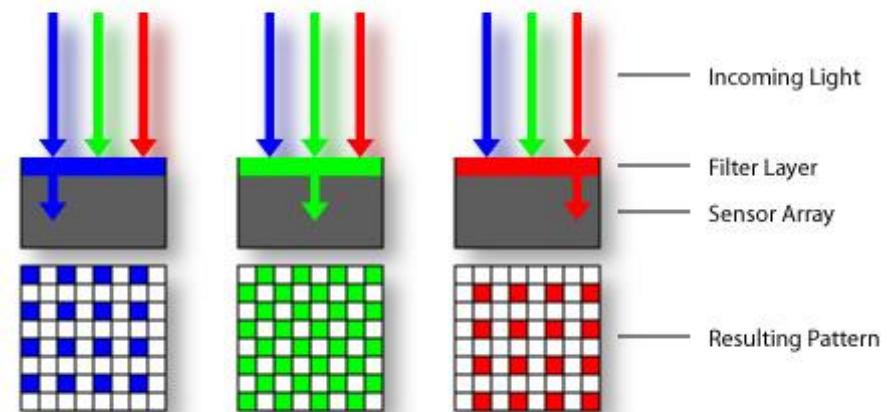
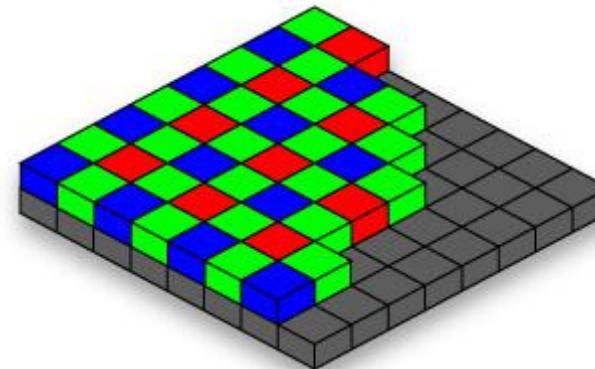
a b

**FIGURE 2.17** (a) Continuous image projected onto a sensor array. (b) Result of image sampling and quantization.



CMOS sensor

Bayer Filter



SMIL内部资料 请勿外泄

# Digital Image Storage

Stored in two parts

## ■ Header

- width, height ,... ,cookie
  - Cookie is an indicator of what type of image file

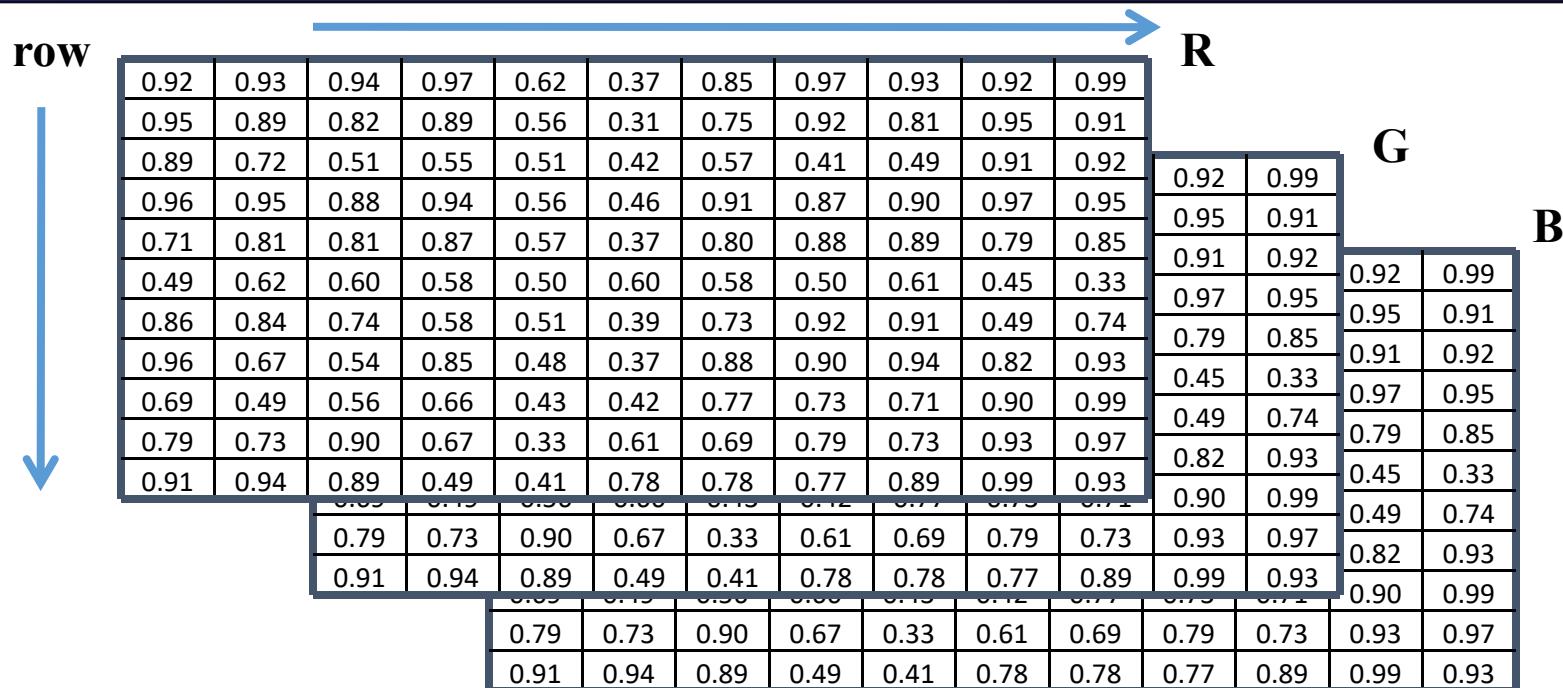
## ■ Data

- uncompressed, compressed, ASCII, binary.

File types

## ■ JPEG, BMP, PPM

# Images in Matlab



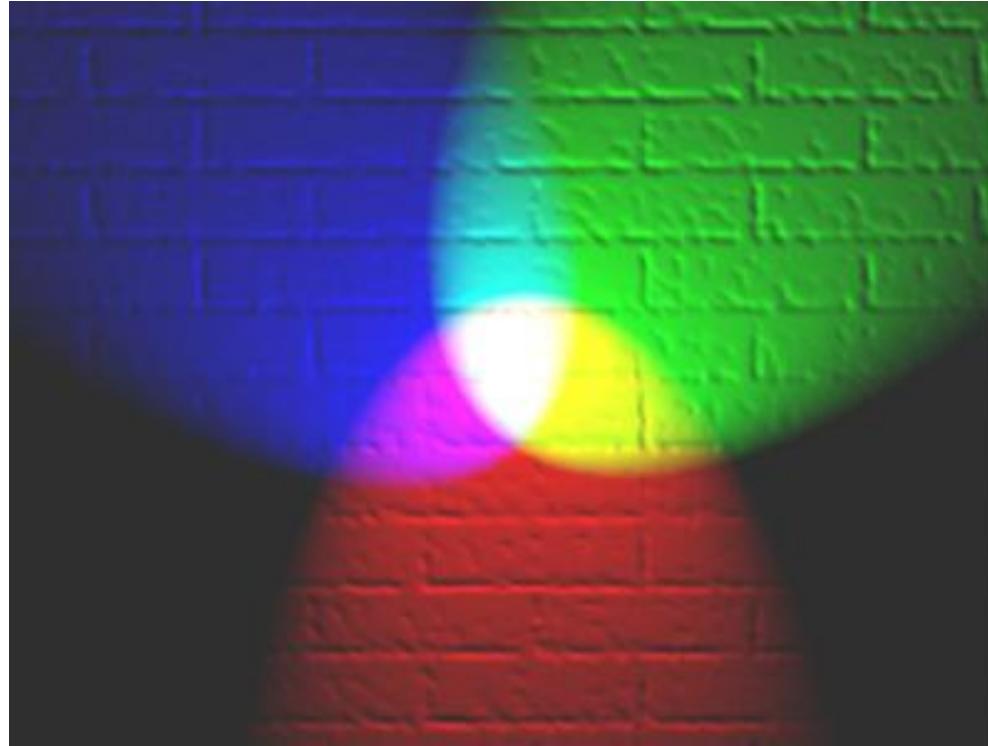
- Image represented as a matrix
  - Suppose we have a  $N \times M$  RGB image called “im”
    - $im(1,1,1)$  = top-left pixel value in R-channel
    - $im(y, x, b)$  = y pixels down, x pixels to right in the  $b^{th}$  channel
    - $im(N, M, 3)$  = bottom-right pixel in B-channel
  - `imread(filename)` returns a `uint8` image (values 0 to 255)

SMIL 内部资料 请勿外泄 • Convert to double format (values 0 to

Convert to double format (values 0 to 1) with `im2double`

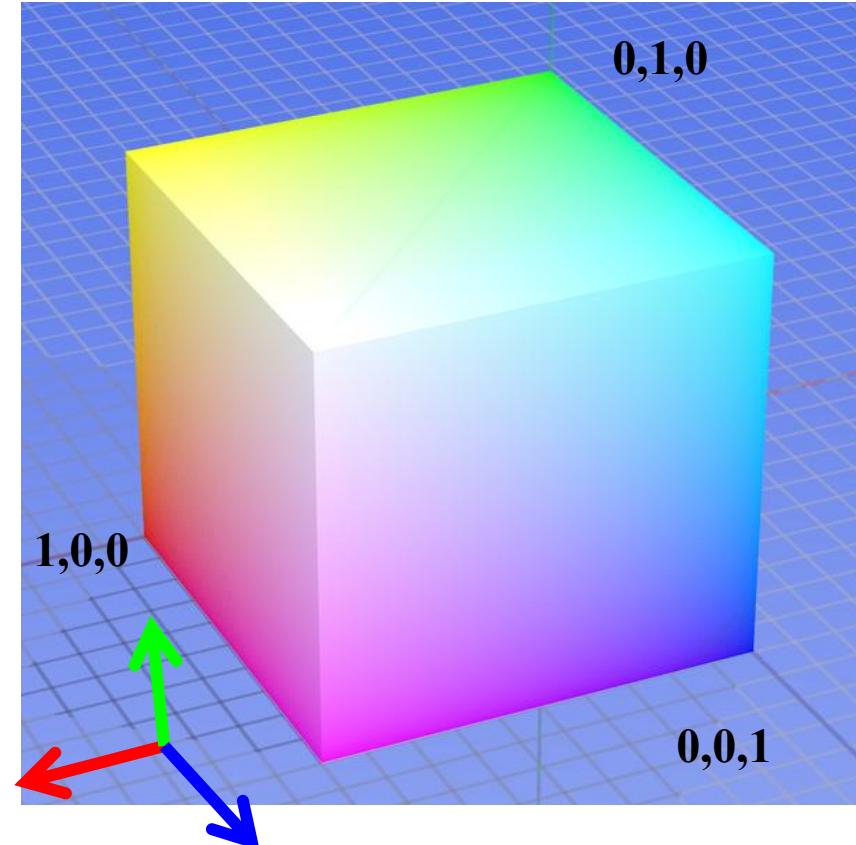
# Color Space

How can we represent color?



# Color Space: RGB

Default color space



**R**  
(G=0,B=0)



**G**  
(R=0,B=0)



**B**  
(R=0,G=0)

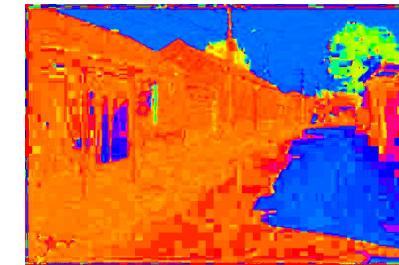
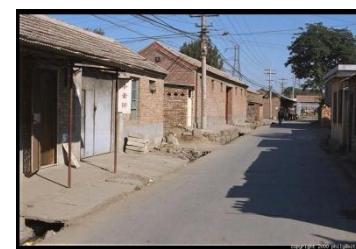
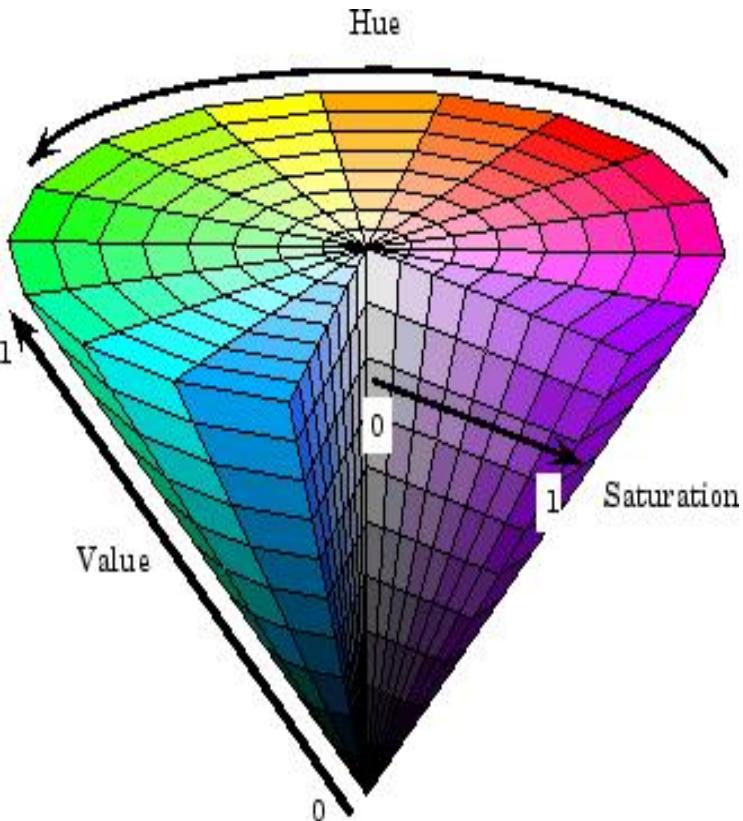
## ■ Some drawbacks

- Strongly correlated channels

SIMI 内部资料 请勿外泄

# Color Space: HSV

Intuitive color space



**H**  
( $S=1, V=1$ )



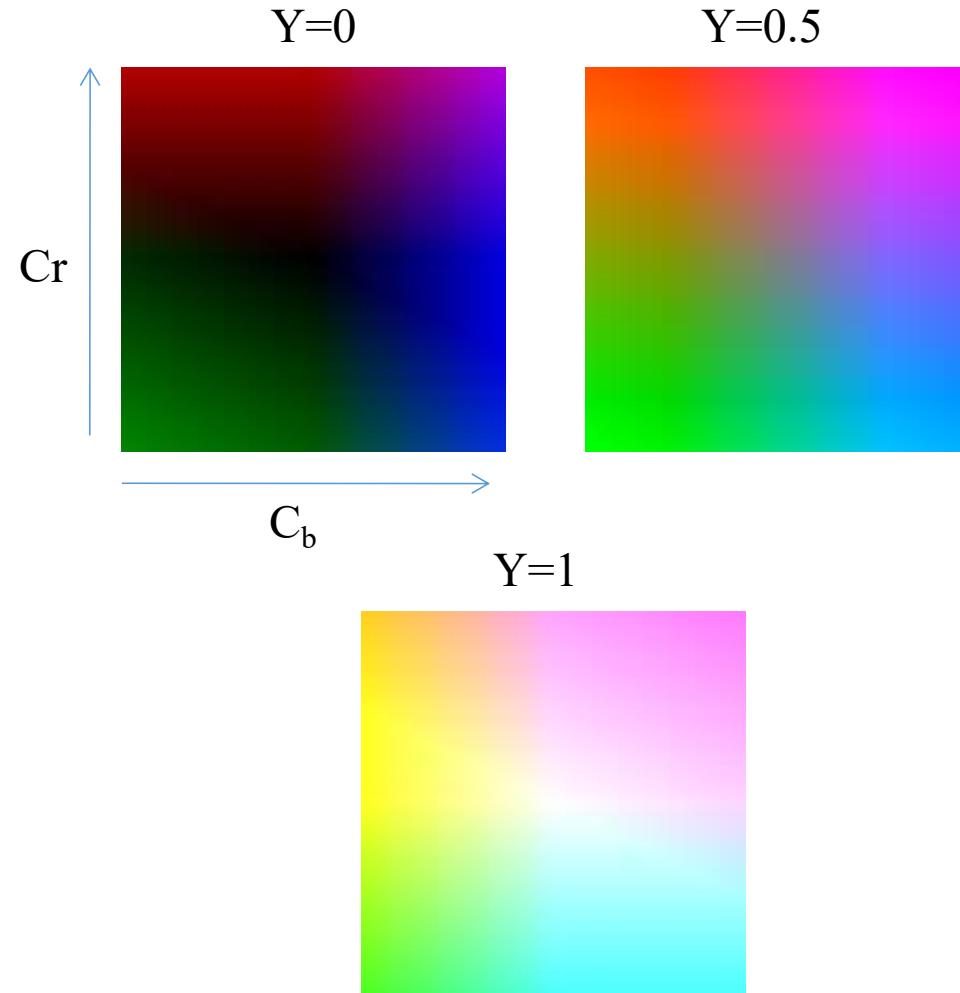
**S**  
( $H=1, V=1$ )



**V**  
( $H=1, S=0$ )

SMIL内部资料 请勿外泄

# Color Space: YCbCr



- Fast to compute, good for compression, used by TV

# Color Space: YCbCr

$$Y' = 16 + \frac{65.738 \cdot R'_D}{256} + \frac{144.384 \cdot G'_D}{256} + \frac{25.064 \cdot B'_D}{256}$$



**Y**  
(Cb=0.5,Cr=0.5)

$$C_B = 128 - \frac{37.945 \cdot R'_D}{256} - \frac{74.494 \cdot G'_D}{256} + \frac{112.439 \cdot B'_D}{256}$$



?

?

(Y=0.5,Cr=0.5)

$$C_R = 128 + \frac{112.439 \cdot R'_D}{256} - \frac{94.154 \cdot G'_D}{256} - \frac{18.285 \cdot B'_D}{256}$$



?

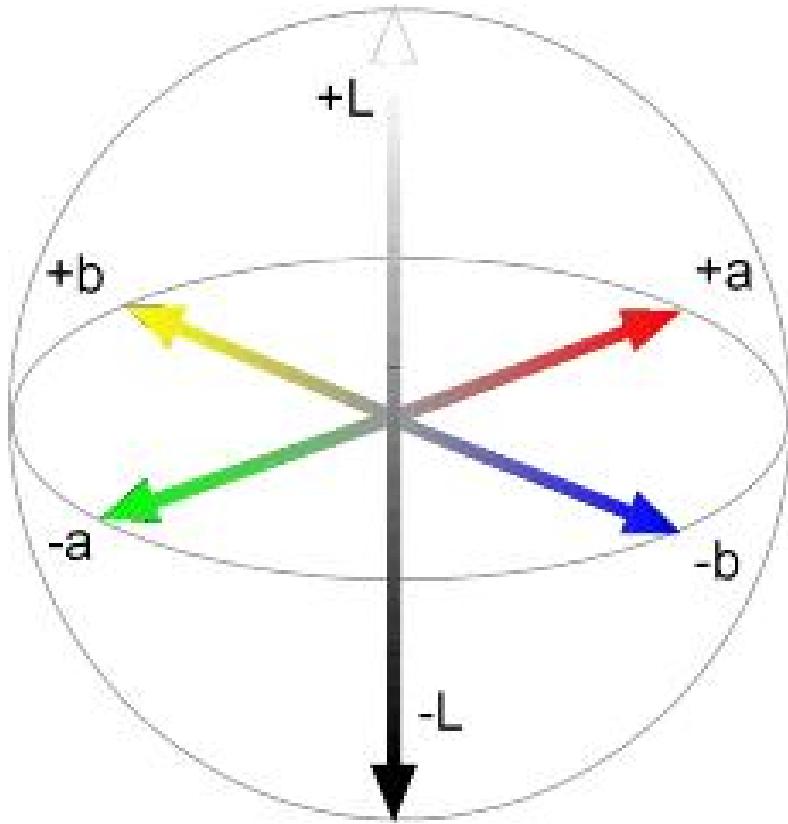
?

(Y=0.5,Cb=0.5)

SMIL内部资料 请勿外泄

# Color Space: CIE L\*a\*b\*

“Perceptually uniform” color space



Luminance = brightness

Chrominance = color

SMIL 内部资料 请勿外泄



**L**  
( $a=0, b=0$ )



**a**  
( $L=65, b=0$ )



**b**  
( $L=65, a=0$ )

# Questions

- **Which contains more information?**
  - intensity (1 channel)
  - chrominance (Chroma 2 channels)
  
- **If you had to choose, would you rather go without luminance or chrominance?**

# Only Color Shown – Constant Intensity



SMIL内部资料 请勿外泄

# Only Intensity Shown – Constant Color



copyright 2000 philg@mit.edu

SMIL内部资料 请勿外泄

# What is Image Processing?

- Improving the image qualities
- Improvement of pictorial information for human interpretation
- The processing helps in maximizing clarity, sharpness and details of features of interest towards information extraction

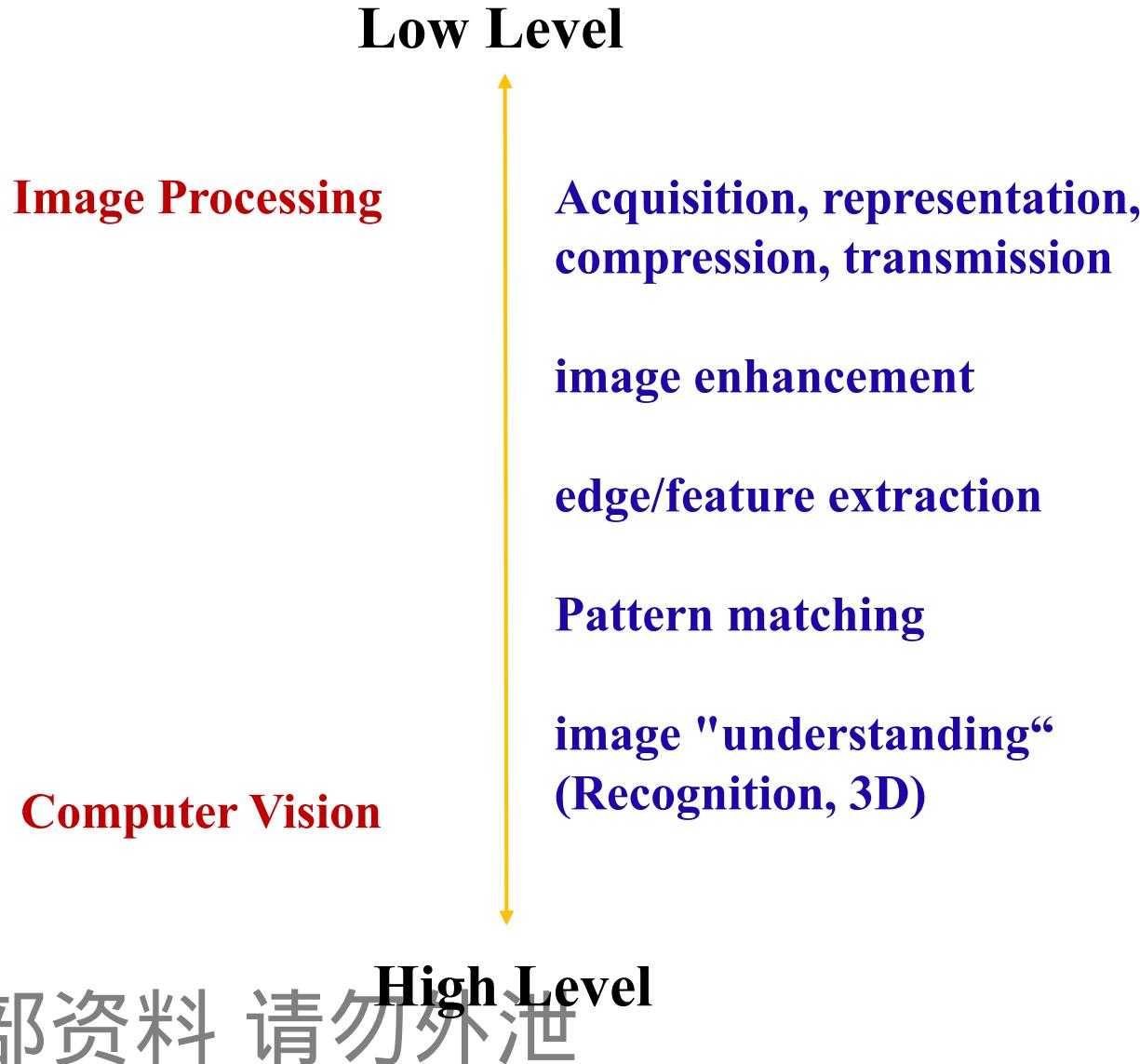
# Contents

1 Introduction to Image Processing

2 Image Processing Applications

3 Image Filtering

# Image Processing vs Computer Vision





SMIL内部资料 请勿外泄

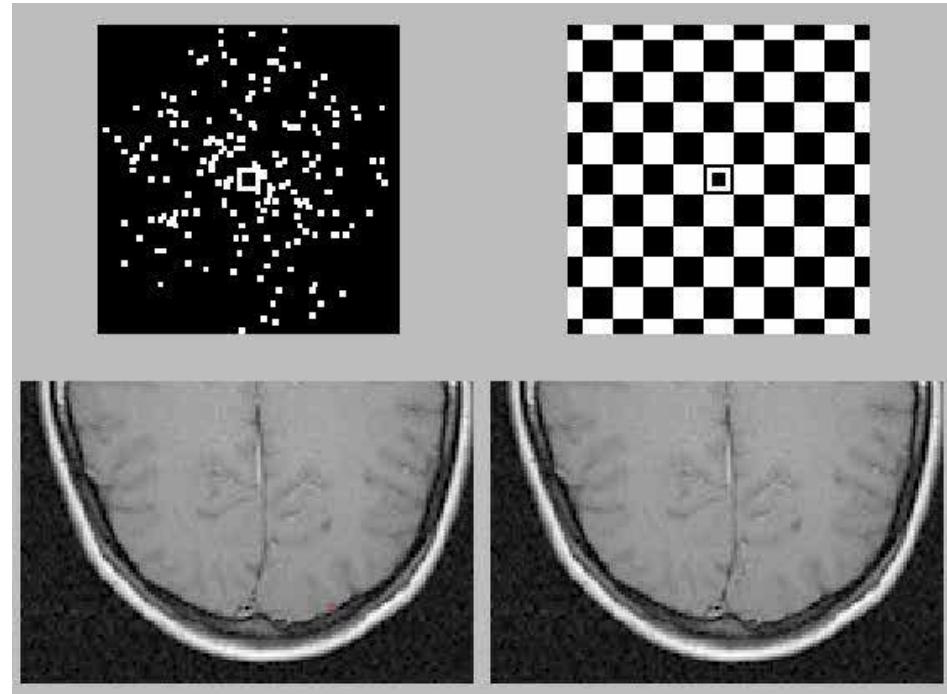
# IP and CV are Interdisciplinary Fields

- Mathematical Models (CS, EE, Math)

- Eye Research (Biology)

- Brain Research:

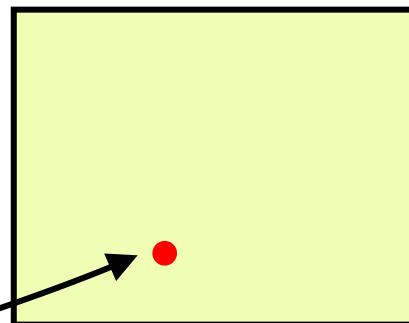
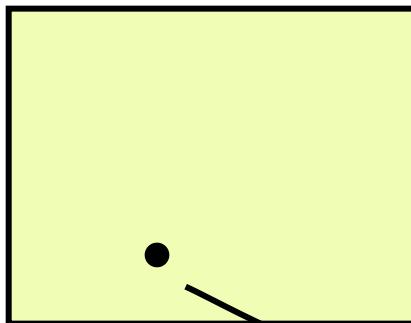
- Psychophysics (Psychologists)
- Electro-physiology (Biologists)
- Functional MRI (Biologists)



# Image Operations

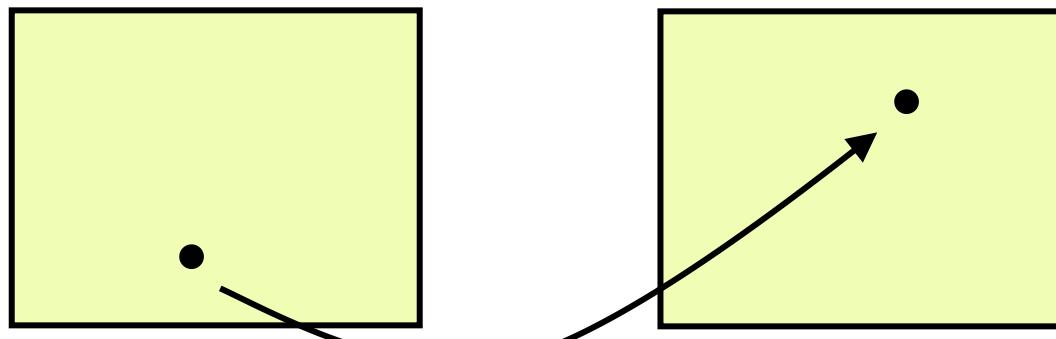
- Point Operations
- Geometric Operations
- Spatial Operations
- Global Operations (Freq. domain)
- Multi-Resolution Operations
- The Fourier Transform
- TMulti-Resolution

# Point Operations



SMIL内部资料 请勿外泄

# Geometric Operations



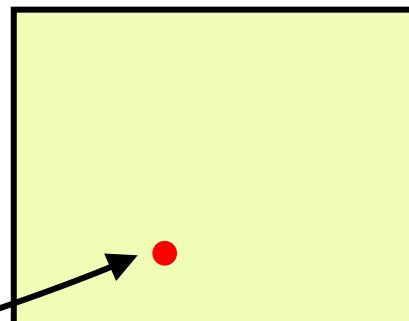
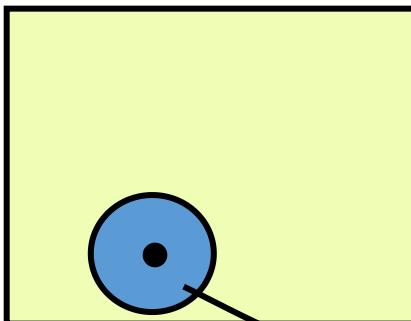
SMIL内部资料 请勿外泄

# Geometric and Point Operations



SMIL内部资料 请勿外泄

# Spatial Operations



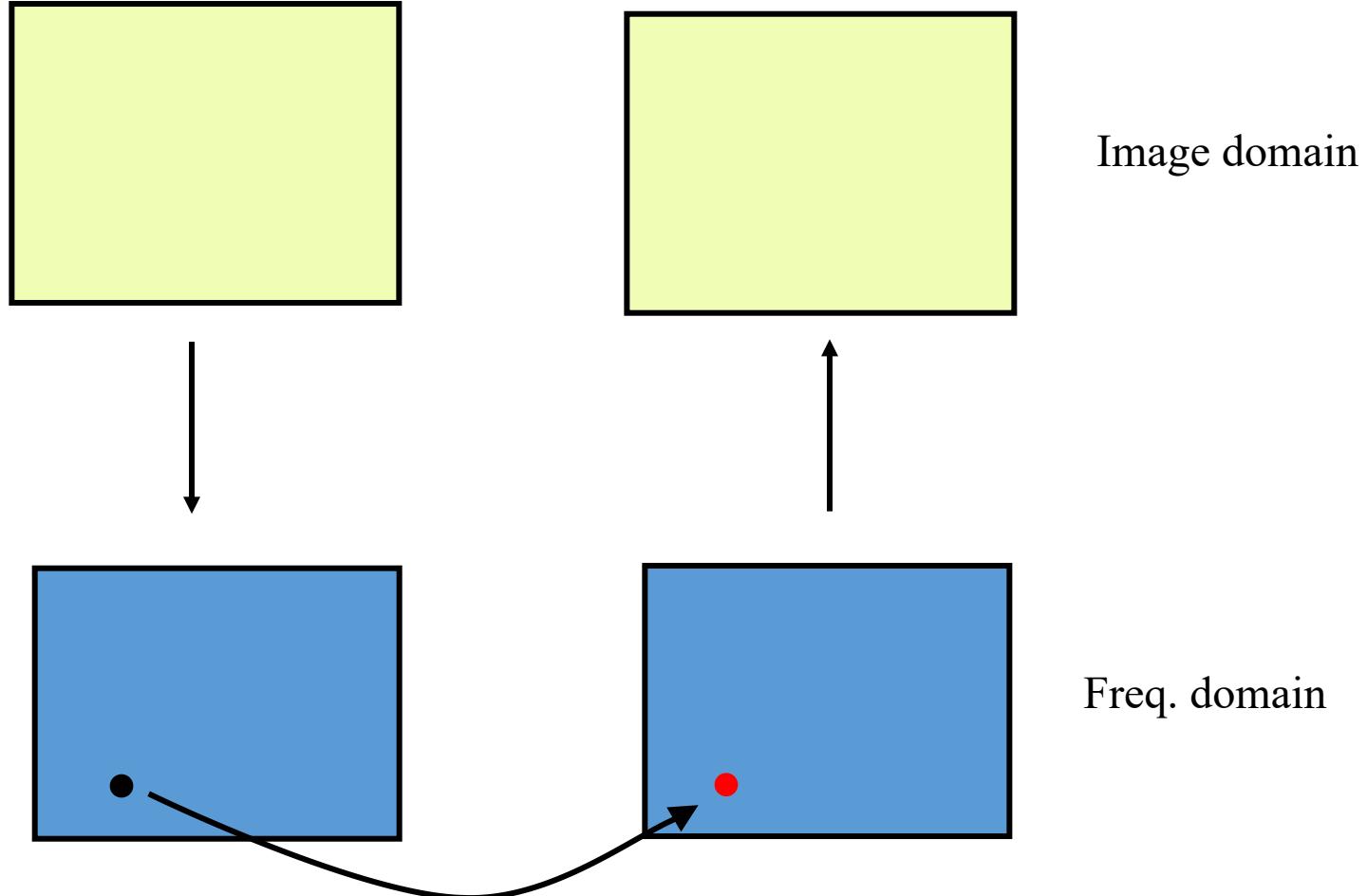
SMIL内部资料 请勿外泄

# Global Operations



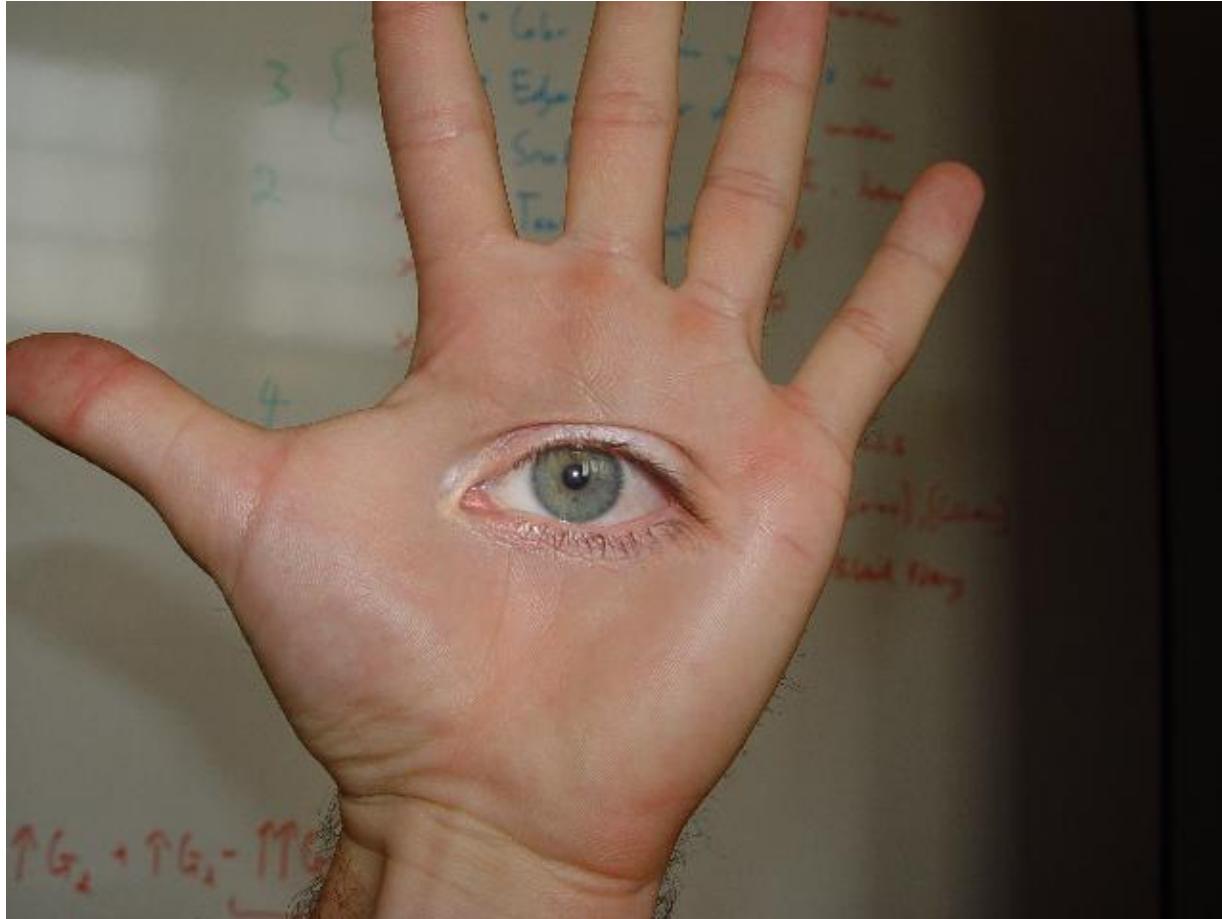
SMIL内部资料 请勿外泄

# Global Operations



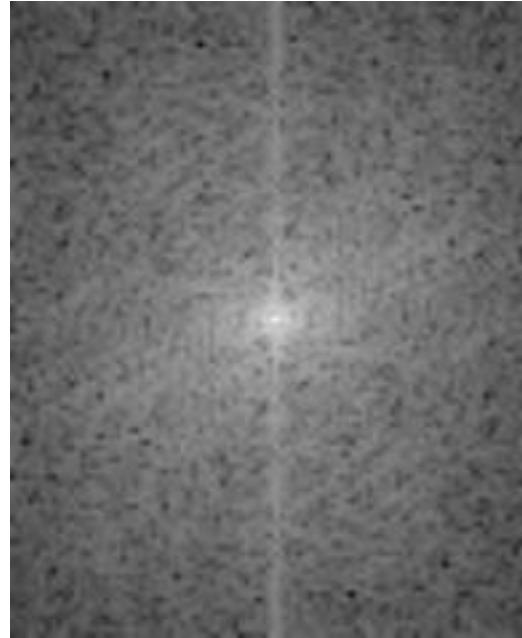
SMIL内部资料 请勿外泄

# Multi-resolution Operations



SMIL内部资料 请勿外泄

# Fourier Transform



Jean Baptiste Joseph Fourier 1768-1830

SMIL内部资料 请勿外泄

# TMulti-Resolution

Low resolution



High resolution



SMIL内部资料 请勿外泄

# Image Denoising

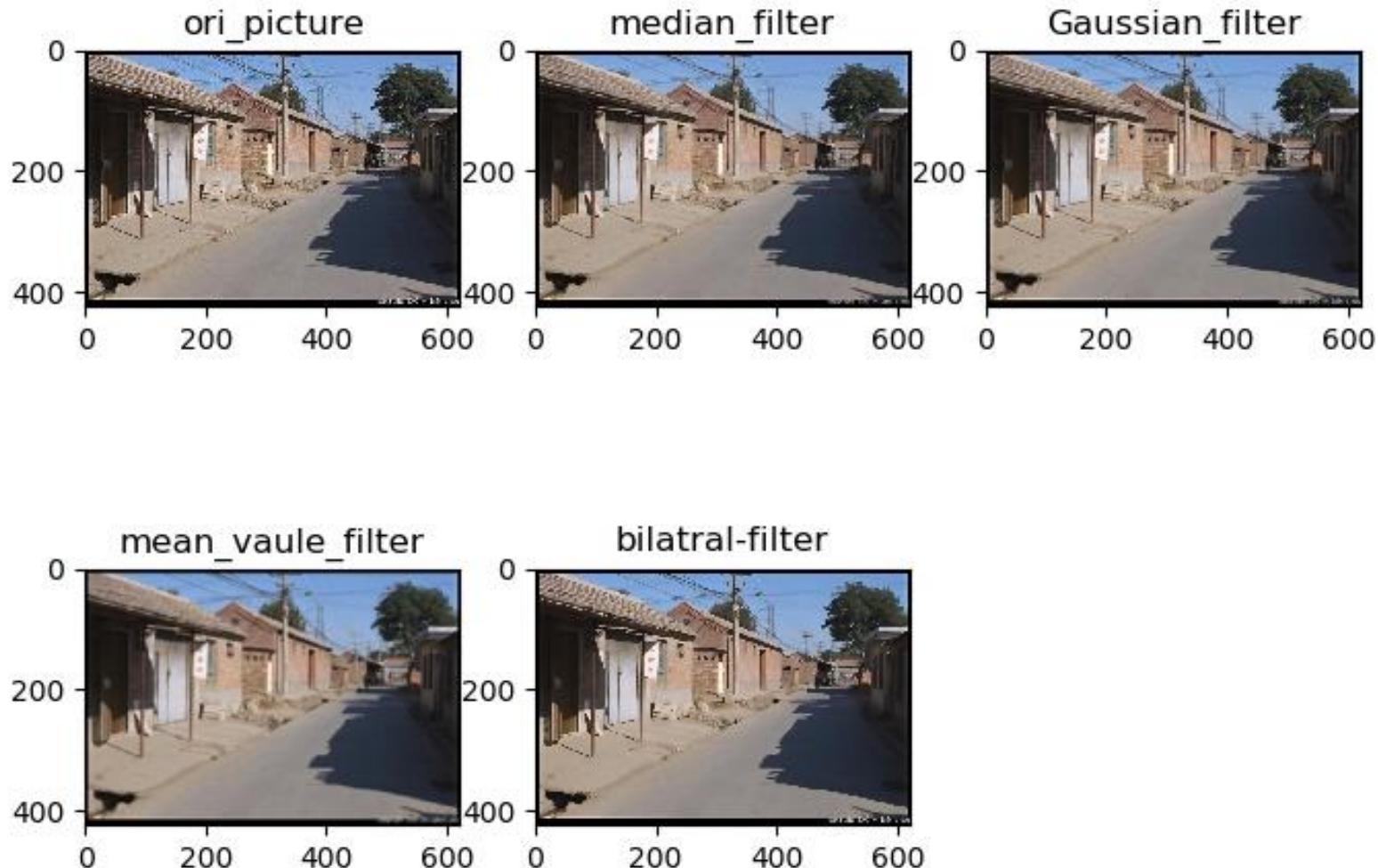
## Definition

- Image Denoising refers to the process of reducing noise in digital images called image denoising

## Denoising method

- Median filtering
- Gaussian filtering
- Mean filtering
- Wiener filtering
- Fourier filtering

# Image Denoising



SMIL内部资料 请勿外泄

# Image Enhancement



SMIL内部资料 请勿外泄

# Image Enhancement

## Definition

- Enhancing useful information in an image
- It can be a distorted process
- It improve the visual effect of the image, for a given image application

## Image enhancement method

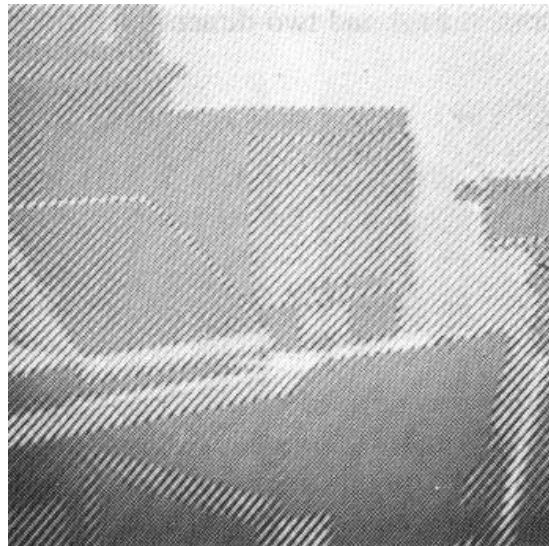
- Airspace based algorithm
- Point algorithm
- Neighborhood enhancement algorithm
- Frequency domain based algorithm

# Image Deblurring

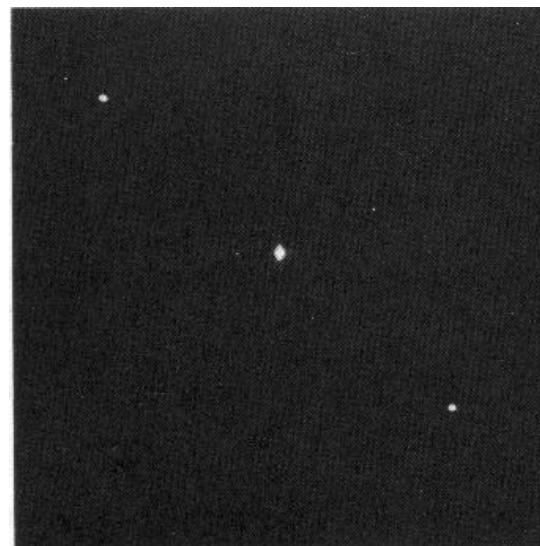
- There are many reasons for image blur, including optical factors, atmospheric factors, artificial factors, technical factors, etc.
- To achieve better processing results, blurring caused by different reasons often requires different processing methods
- From a technical point of view, fuzzy image processing methods are mainly divided into three categories:
  - Image enhancement
  - Image restoration
  - Super resolution reconstruction

# Operations in Frequency Domain

Original Noisy image



Fourier Spectrum



Filtered image



# Image Inpainting

## Definition

- Image restoration refers to the process of reconstructing images and missing or damaged parts of the video
- Image restoration, also known as image interpolation or video interpolation
- Using sophisticated algorithms to replace missing, damaged the image data, mainly to replace some small areas and blemishes

# Image Inpainting



SMIL内部资料 请勿外泄

# Video Inpainting



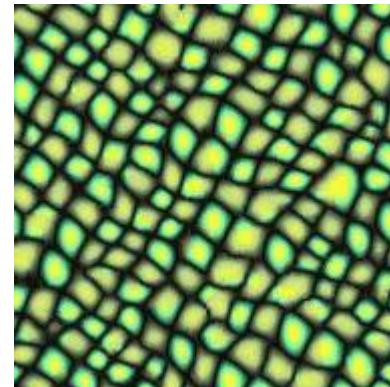
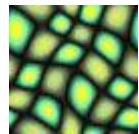
SMIL内部资料 请勿外泄

# Texture Synthesis

## Definition

- Texture synthesis is proposed to solve the problem of seam aliasing and other problems in texture mapping
- At present, texture synthesis methods can be divided into two categories:
  - One is Process Texture Synthesis (PTS)
    - It generates texture directly on the surface by simulating the physics generation process
  - the other is Sample Based Texture Synthesis (SBTS)
    - Given a small texture, generate a large similar texture

# Texture Synthesis



SMIL内部资料 请勿外泄

# Industry and Applications

## ■ Automobile driver assistant

- Lane departure warning
- Adaptive cruise control
- Obstacle warning

## ■ Digital Photography

- Image Enhancement
- Compression
- Color manipulation
- Image editing
- Digital cameras



MobilEye system

# Industry and Applications

## ■ Film and Video

- Editing
- Special effects

## ■ Image Database

- Content based image retrieval
- visual search of products
- Face recognition

## ■ Industrial Automation and Inspection

- vision-guided robotics
- Inspection systems



# Industry and Applications

## ■ Sports analysis

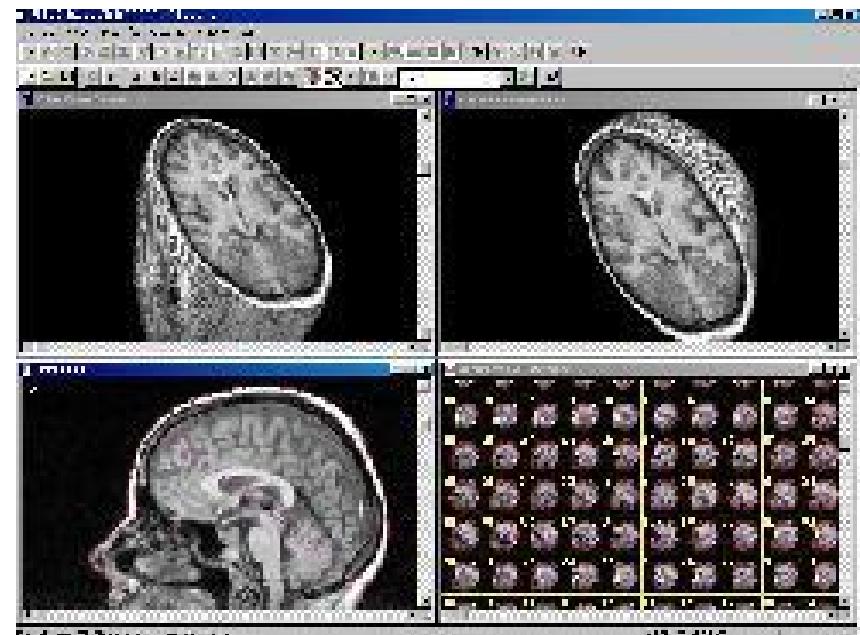
- sports refereeing and commentary
- 3D visualization and tracking sports actions

## ■ Medical and Biomedical

- Surgical assistance
- Sensor fusion
- Vision based diagnosis

## ■ Astronomy

- Astronomical Image Enhancement
- Chemical/Spectral Analysis



# Industry and Applications

## ■ Arial Photography

- Image Enhancement
- Missile Guidance
- Geological Mapping

## ■ Robotics

- Autonomous Vehicles

## ■ Security and Safety

- Biometry verification (face, iris)
- Surveillance (fences, swimming pools)



# Industry and Applications

## ■ Military

- Tracking and localizing
- Detection
- Missile guidance

## ■ Traffic and Road Monitoring

- Traffic monitoring
- Adaptive traffic lights



Cruise Missiles

## Why Computer Vision is Hard?

# Why Computer Vision is Hard?

- Inverse problems
- Priori-knowledge is required
- Complexity extensive
  - Top-Down vs Bottom-Up paradigm
  - Parallelism
- Non-local operations
  - Propagation of Information

# Contents

1 Introduction to Image Processing

2 Image Processing Applications

3 Image Filtering

SMIL内部资料 请勿外泄

# Basis for Interpreting Intensity Images



- Key idea: for nearby scene points, most factors do not change much
- The information is mainly contained **in local differences** of

brightness

SMIL内部资料 请勿外泄

# Basis for Interpreting Intensity Images

- Darkness = Large Difference in Neighboring Pixels



# Three Views of Filtering

## Image filters in spatial domain

- Filter is a mathematical operation on values of each patch
- Smoothing, sharpening, measuring texture

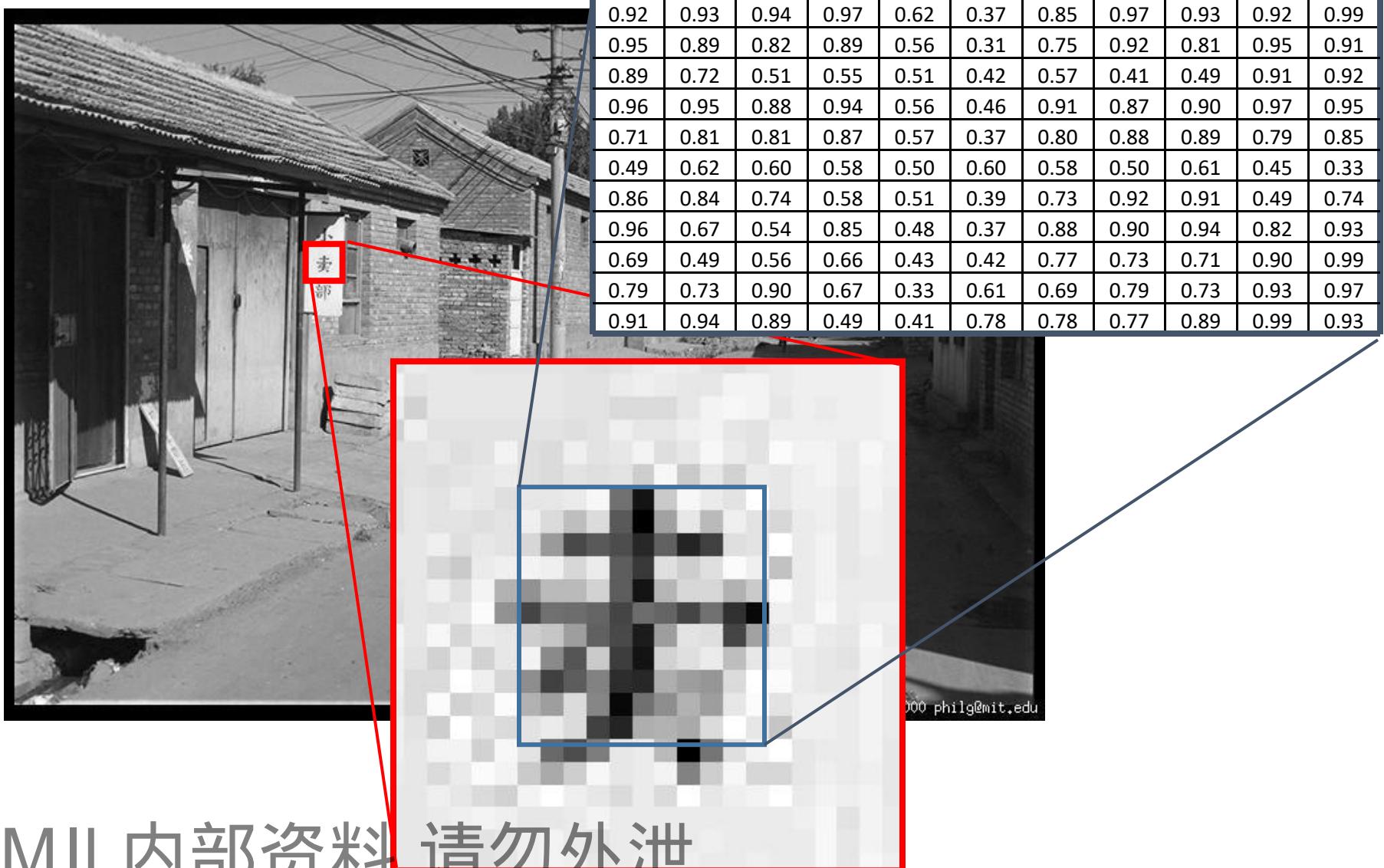
## Image filters in the frequency domain

- Filtering is a way to modify the frequencies of images
- Denoising, sampling, image compression

## Templates and Image Pyramids

- Filtering is a way to match a template to the image
- Detection, coarse-to-fine registration

# Gray Image (Pixel Matrix)



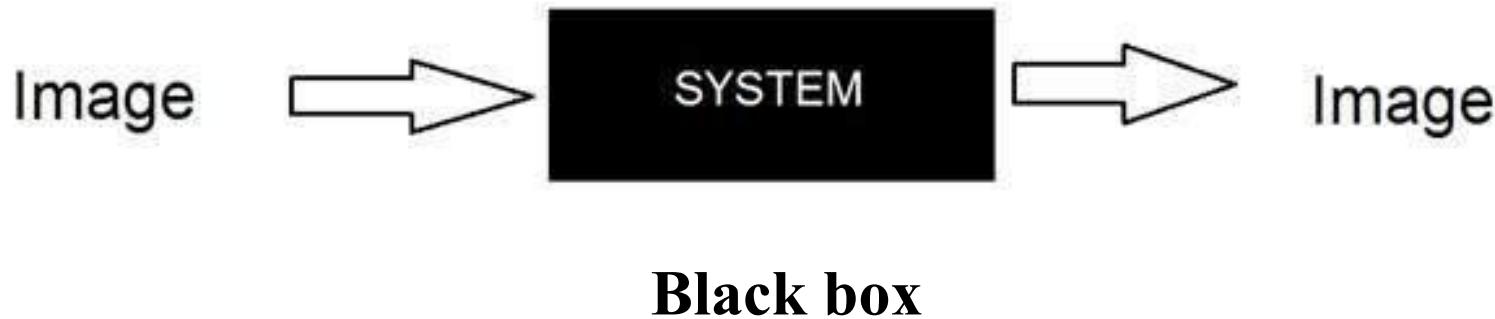
# Image Filtering

- Image filtering: for each pixel, compute function of local neighborhood and output a new value
- Same function applied at each position
- Output and input image are typically the same size

# Image Filtering or Convolution

- Linear filtering: a weighted sum/difference of pixel values
- Really important!
  - Enhance images
    - Denoising, smooth, increase contrast, blurring, noise reduction, image sharpening etc.
  - Extract information from images
    - Texture, edges, distinctive points, etc.
  - Detect patterns
    - Template matching
    - Detect patterns
  - **Deep Learning**
    - Convolutional Neural Networks

# Concept of Convolution



SMIL内部资料 请勿外泄

# Concept of Convolution



- $g(x, y) = h(x, y) * f(x, y)$  ; mask convolved with an image
- $g(x, y) = f(x, y) * h(x, y)$  ; image convolved with mask
- The convolution operator (\*) is commutative,  $h(x, y)$  is the mask or filter.

# Convolution Mask

- Mask is a matrix, usually of size  $1 \times 1$ ,  $3 \times 3$ ,  $5 \times 5$  or  $7 \times 7$  (odd number).
- Convolution steps:
  - Flip the mask horizontally and vertically
  - Slide the mask onto the image
  - Multiply the corresponding elements and add them
  - Repeat this until all image values are calculated

# Convolution Example

## ■ Image matrix

2	4	6
8	10	12
14	16	18

- Place the mask over each image element. Multiply the corresponding elements and add them

9	8	7				
6	5	2	4	4		6
3	2	8	1	10		12
	14		16		18	

# Convolution Example

- Red – mask, blue – image

9	8	7				
6	5	2	4	4		6
3	2	8	1	10		12
	14		16		18	

- First pixel:  $= (5*2) + (4*4) + (2*8) + (1*10)$

$$= 10 + 16 + 16 + 10$$

$$= 52$$

- Convolution can achieve blurring, edge detection, sharpening, noise reduction, etc.

# Convolution Example

## ■ Example: box filter

$$\frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

$$g[\cdot, \cdot] \frac{1}{9}$$

1	1	1
1	1	1
1	1	1

			0							

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

$$g[\cdot, \cdot] \frac{1}{9}$$

1	1	1
1	1	1
1	1	1

	0	10								

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

1	1	1
1	1	1
1	1	1

$$g[\cdot, \cdot] \frac{1}{9}$$

0	10	20								

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

1	1	1
1	1	1
1	1	1

$g[\cdot, \cdot] \frac{1}{9}$


$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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	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

 $h[\cdot, \cdot]$ 

$$g[\cdot, \cdot] \frac{1}{9}$$

1	1	1
1	1	1
1	1	1

	0	10	20	30	30				

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

$$g[\cdot, \cdot] \frac{1}{9}$$

1	1	1
1	1	1
1	1	1

	0	10	20	30	30					

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

$$g[\cdot, \cdot] \frac{1}{9}$$

1	1	1
1	1	1
1	1	1

	0	10	20	30	30					

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Image Filtering

 $f[\cdot, \cdot]$ 

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

 $h[\cdot, \cdot]$ 

1	1	1
1	1	1
1	1	1

$g[\cdot, \cdot] \frac{1}{9}$

	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	

$$h[m, n] = \sum_{k,l} g[k, l] f[m + k, n + l]$$

SMIL内部资料 请勿外泄

# Box Filter

- What does it do?
  - Replaces each pixel with an average of its neighborhood
  - Achieve smoothing effect (remove sharp features)

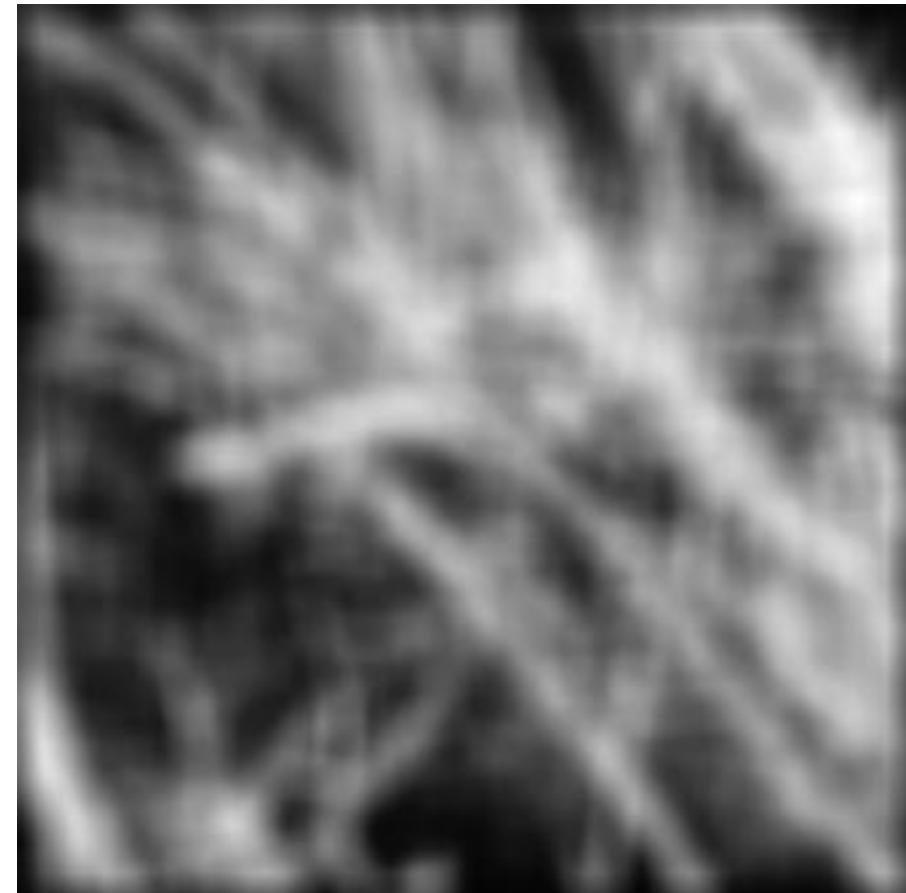
$$g[\cdot, \cdot]$$

$$\frac{1}{9}$$

1	1	1
1	1	1
1	1	1

# Box Filter

- Smoothing with box filter



# Practice with Linear Filters



0	0	0
0	1	0
0	0	0

?

Original

# Practice with Linear Filters



Original

0	0	0
0	1	0
0	0	0



Filtered  
**(no change)**

# Practice with Linear Filters



0	0	0
0	0	1
0	0	0

?

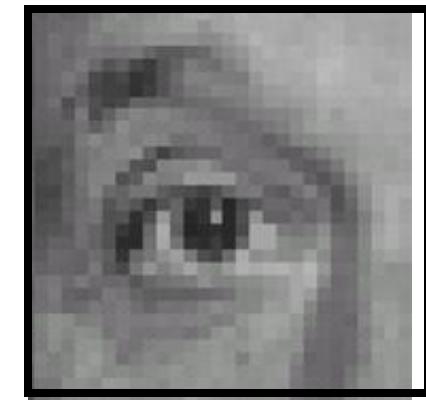
Original

# Practice with Linear Filters



Original

0	0	0
0	0	1
0	0	0



Shifted left  
By 1 pixel

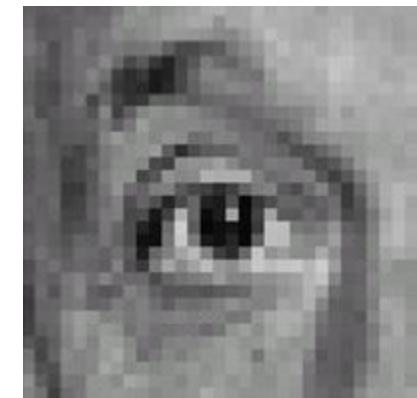
# Practice with Linear Filters



$$\begin{array}{|c|c|c|} \hline 0 & 0 & 0 \\ \hline 0 & 2 & 0 \\ \hline 0 & 0 & 0 \\ \hline \end{array}$$

-

$$\frac{1}{9} \begin{array}{|c|c|c|} \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline 1 & 1 & 1 \\ \hline \end{array}$$

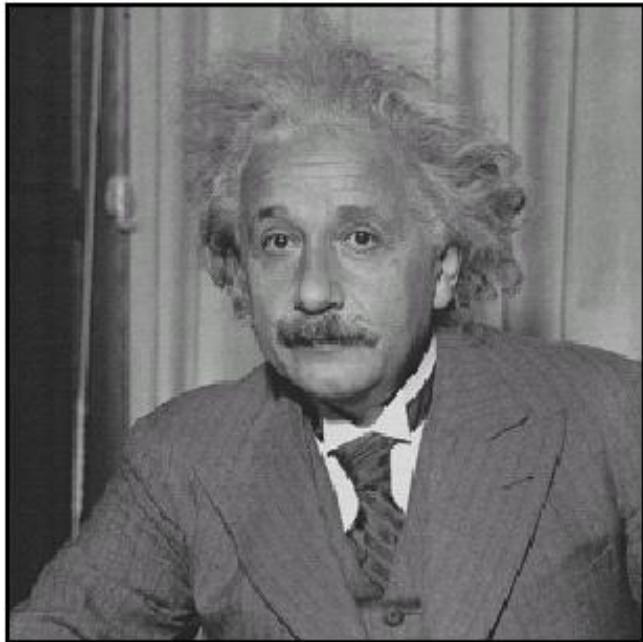


Original

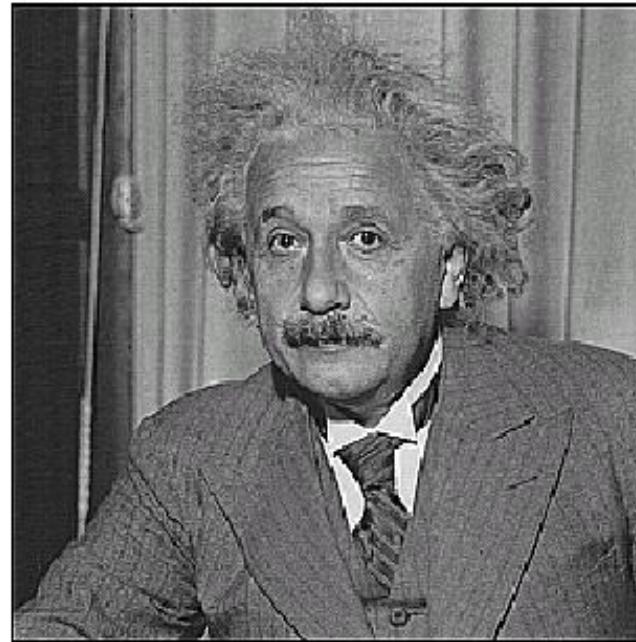
## Sharpening filter

- Accentuates differences with local average

# Sharpening

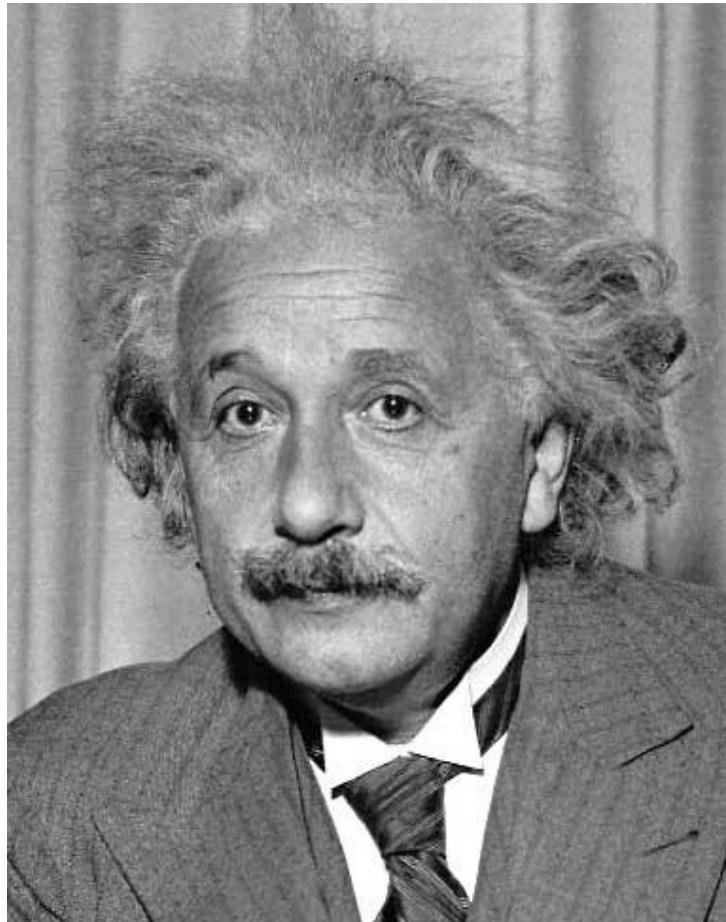


**before**



**after**

# Other Filters



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

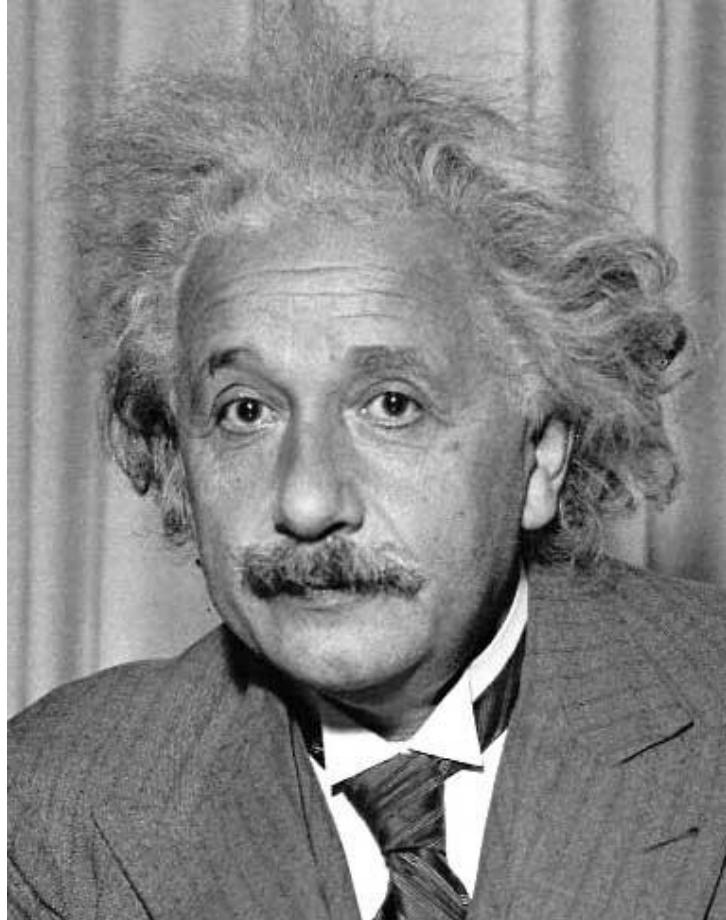
Sobel

Vertical Edge (absolute value)



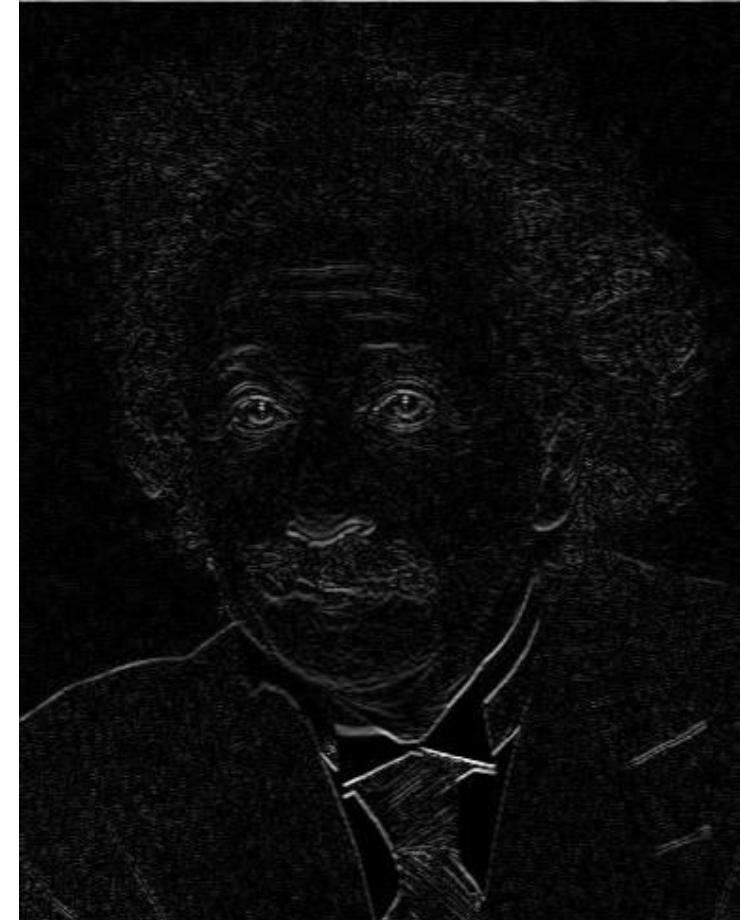
# Other Filters

Horizontal Edge  
(absolute value)



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

Sobel



# Basic Gradient Filters

Horizontal Gradient

0	0	0
-1	0	1
0	0	0

or

-1	0	1
----	---	---

Vertical Gradient

0	-1	0
0	0	0
0	1	0

or

-1
0
1

# Filtering vs Convolution

## ■ 2d filtering

- $h = \text{filter2}(g, f);$     or     $h = \text{imfilter}(f, g);$

$g = \text{filter}$

$f = \text{image}$

$$h[m, n] = \sum_{k,l} g[k, l]f[m + k, n + l]$$

## ■ 2d convolution

- $h = \text{conv2}(g, f);$

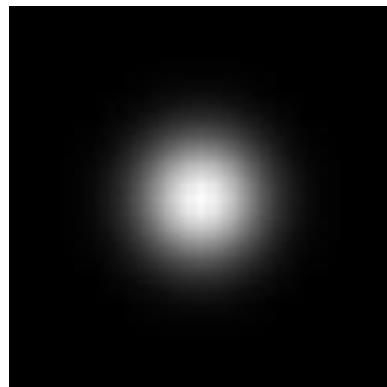
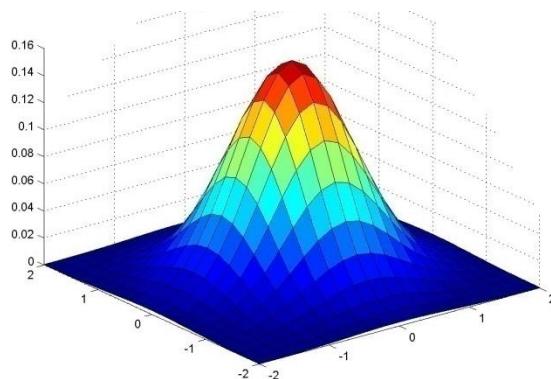
$$h[m, n] = \sum_{k,l} g[k, l]f[m - k, n - l]$$

# Key Properties of Linear Filters

- **Linearity:**  $\text{filter}(f_1 + f_2) = \text{filter}(f_1) + \text{filter}(f_2)$
- **Shift invariance:** same behavior regardless of pixel location  
 $\text{filter}(\text{shift}(f)) = \text{shift}(\text{filter}(f))$
- Any linear, shift-invariant operator can be represented as a convolution

# Important Filter: Gaussian

- Spatially-weighted average

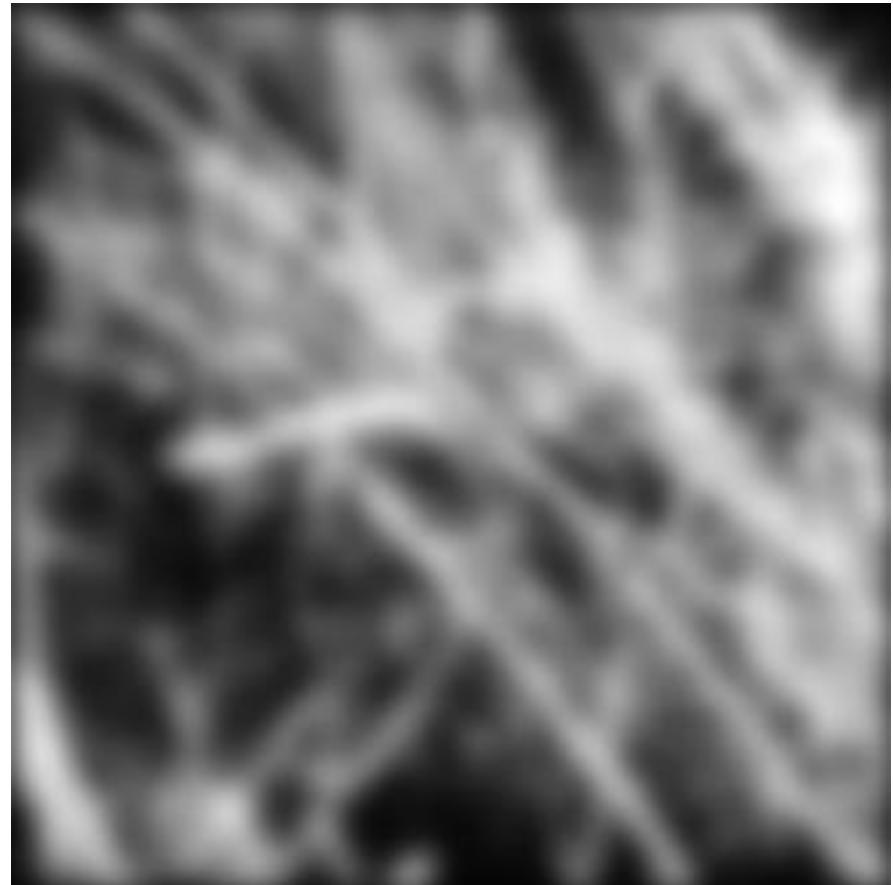


0.003	0.013	0.022	0.013	0.003
0.013	0.059	0.097	0.059	0.013
0.022	0.097	0.159	0.097	0.022
0.013	0.059	0.097	0.059	0.013
0.003	0.013	0.022	0.013	0.003

$5 \times 5, \sigma = 1$

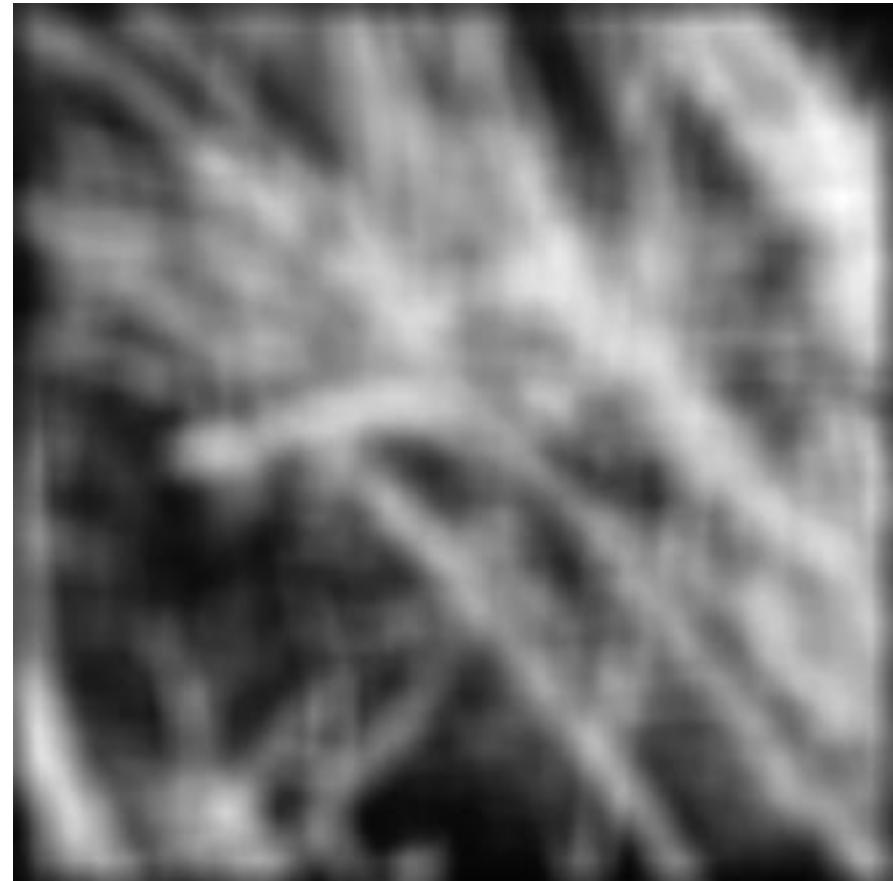
$$G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}}$$

# Smoothing with Gaussian Filter



SMIL内部资料 请勿外泄

# Smoothing with Box Filter



SMIL内部资料 请勿外泄

# Gaussian Filter

- Remove "high-frequency" components from the image (low-pass filter)
  - Images become more smooth
- Convolution with self is another Gaussian
  - So can smooth with small-width kernel, repeat, and get same result as larger-width kernel would have
  - Convolving two times with Gaussian kernel of width  $\sigma$  is same as convolving once with kernel of width  $\sigma\sqrt{2}$  ?
- Separable kernel
  - Factors into product of two 1D Gaussians

# Separability of 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

# Separability Example

2D filtering  
(center location only)

$$\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix} * \begin{matrix} 2 & 3 & 3 \\ 3 & 5 & 5 \\ 4 & 4 & 6 \end{matrix}$$

The filter factors  
into a product of 1D  
filters:

$$\begin{matrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{matrix} = \begin{matrix} 1 \\ 2 \\ 1 \end{matrix} \times \begin{matrix} 1 & 2 & 1 \end{matrix}$$

Perform filtering  
along rows:

$$\begin{matrix} 1 & 2 & 1 \end{matrix} * \begin{matrix} 2 & 3 & 3 \\ 3 & 5 & 5 \\ 4 & 4 & 6 \end{matrix} = \begin{matrix} 11 \\ 18 \\ 18 \end{matrix}$$

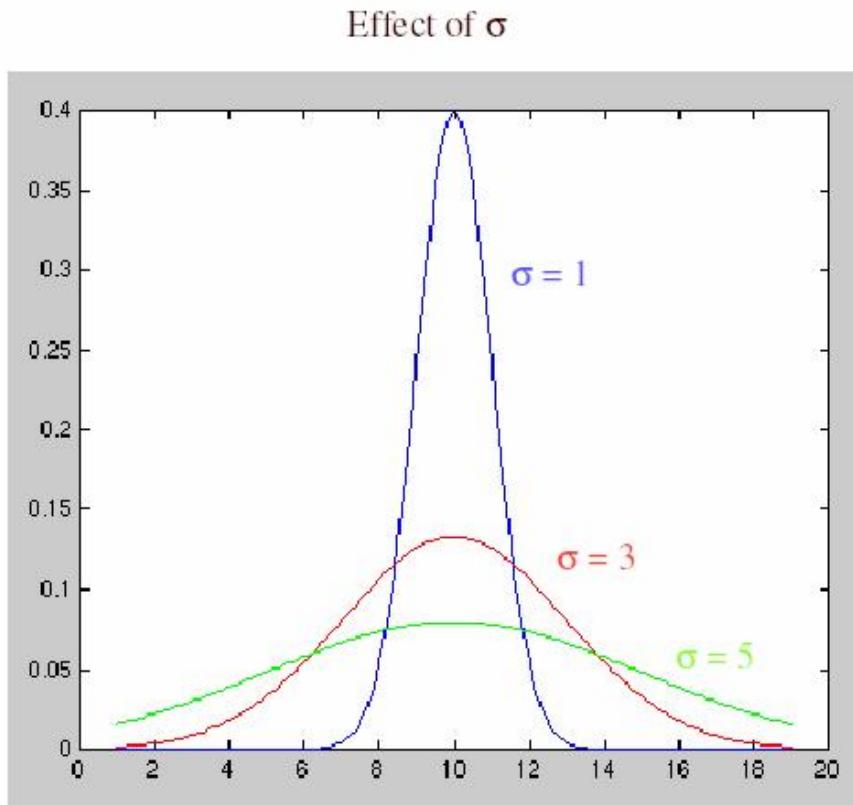
Followed by filtering  
along the remaining column:

$$\begin{matrix} 1 \\ 2 \\ 1 \end{matrix} * \begin{matrix} 11 \\ 18 \\ 18 \end{matrix} = \begin{matrix} 65 \end{matrix}$$

Why is separability useful in practice?

# Practical filters

- How big should the filter be?
- Values at edges should be near zero ← important!
- Rule of thumb for Gaussian: set filter half-width to about  $3 \sigma$



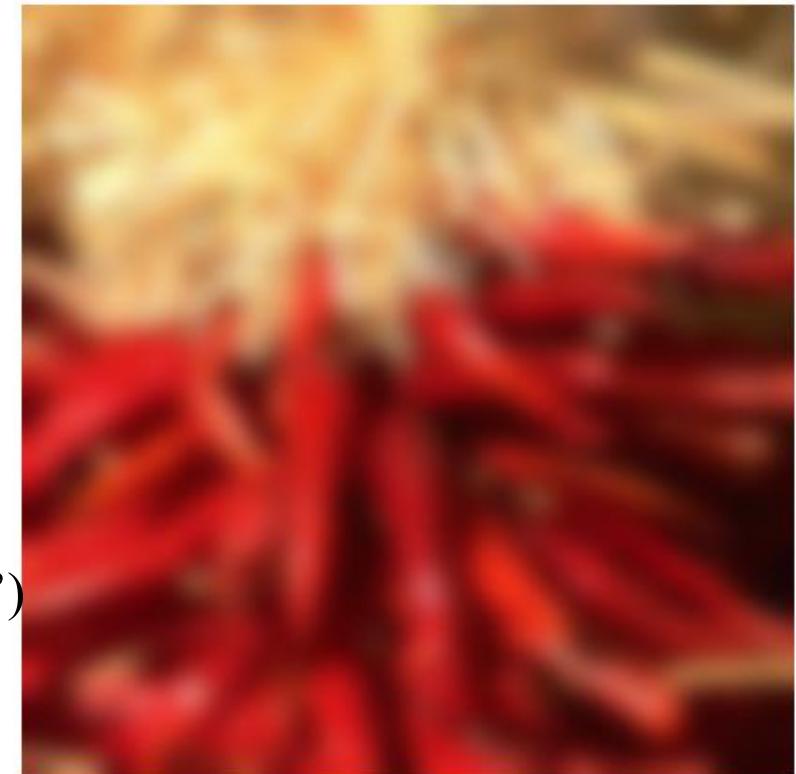
# Practical Matters

## ■ What about near the edge?

- the filter window falls off the edge of the image
- need to extrapolate
- methods:
  - clip filter (black)
  - wrap around
  - copy edge
  - reflect across edge

## ■ methods (MATLAB):

- clip filter (black): `imfilter(f, g, 0)`
- wrap around: `imfilter(f, g, 'circular')`
- copy edge:  
`imfilter(f, g, 'replicate')`
- reflect across edge:



# Thank You