HA NOI UNIVERSITY OF SCIENCE AND TECHNOLOGY
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Computer Vision

Chapter 3: Image Processing

1

Computer Vision

Chapter 3. Image Processing

2

Chapter 3 – Content (cont)

- Remind: Digital image representation
- Point processing
- Convolution and linear filtering
- More neighborhood operators
  - Median/max/min filters
  - Arithmetical/ Logical operations
  - Binary image and morphological operations
- Image transforms

3

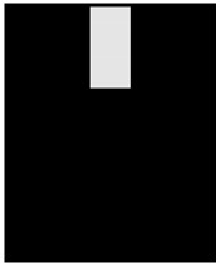
Arithmetical Operations Logical Operations

- AND operation
- OR operation
- Image subtraction
- Image addition

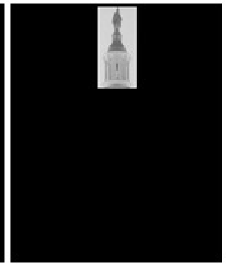
4

AND operation

Original



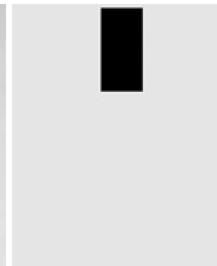
And mask



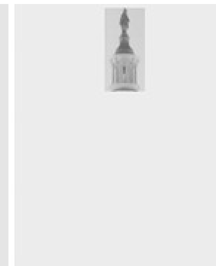
Output image

OR operation

Original



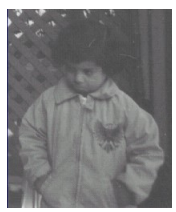
OR mask



Output image

Image Addition

- If f and g are two images, the pixelwise addition R is defined as:
  - R(x,y) = Min(f(x,y)+g(x,y) ; 255)
- Image addition is used to
  - lower the noise in a serie of images
  - increase the luminance by adding the image to itself



Source : Eric Favier. L'analyse et le traitement des images. ENISE.

Average Images

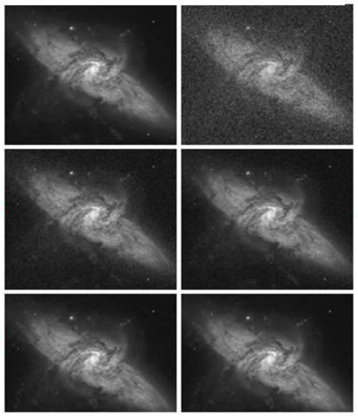
- g(x,y) is the addition of f(x,y) and noise $\eta(x,y)$

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

- If we have several images {g(x,y)}, we can compute the average one

$$\bar{g}(x, y) = \frac{1}{K} \sum_{i=1}^{K} g_i(x, y)$$

Average Images



K = 8

K = 16

K=64

K = 128

Image subtraction

- The pixelwise substraction of two images f and g is:
  S(x,y) = Max(f(x,y)-g(x,y) ; 0)
- Image substraction is used to
  - detect defaults, detect difference between images
  - detect motion in images

Source : Eric Favier. L'analyse et le traitement des images. ENISE.

Image subtraction

a b
c d

FIGURE 3.28 (a) Original fractal image. (b) Result of setting the four lower-order bit planes to zero. (c) Difference between (a) and (b). (d) Histogram-equalized difference image. (Original image courtesy of Ms. Melissa D. Binde, Swarthmore College, Swarthmore, PA).

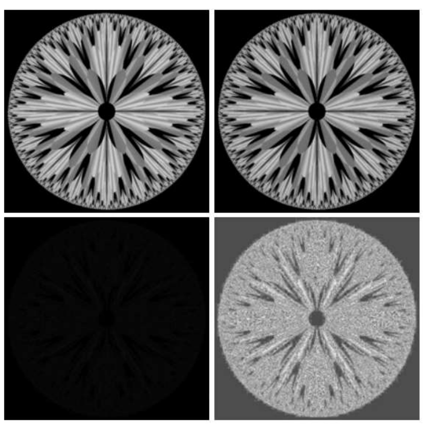


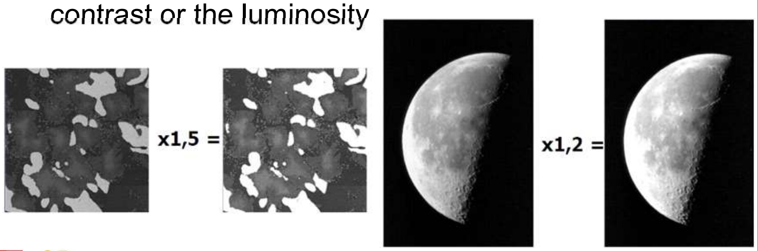
Image subtraction

Image *Background image* *Detection*

After detection, we still have some noise, that we can clean to keep only the object of interest

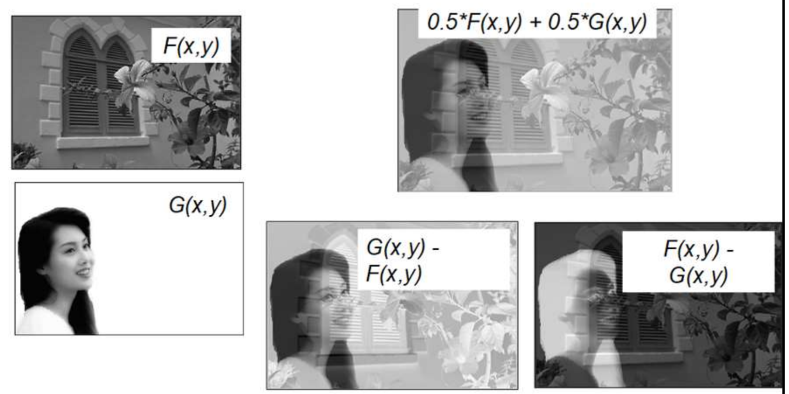
Image multiplication

- The multiplication S of an image f by a ratio (factor) is defined as:
  - S(x,y) = Max(f(x,y)*ratio ; 255)
- Image multiplication can be used to increase the *contrast or* the luminosity



Source : Eric Favier. L'analyse et le traitement des images. ENISE.

Operations on images



Source : www.nte.montaigne.u-bordeaux.fr/SuppCours/5314/Dai/TraitImage01-02.ppt

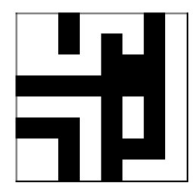
Example code

- I1 = imread('images/ball1.png');
- >> I1 = rgb2gray(I1);
- >> I1 = 255 - I1;
- >> I2 = imread('images/ball2.png');
- >> I2 = rgb2gray(I2);
- >> I2 = 255 - I2;
- >> I3 = I1 + I2;
- >> I4 = image_substract(I1, I2);
- >> I5 = I4;
- >> I5(find(I4==-255))=255;
- >> I5 = uint8(I5);
- I6 = I3 - I5;
- I7 = I1 – I6;
- I8 = I2 – I6;

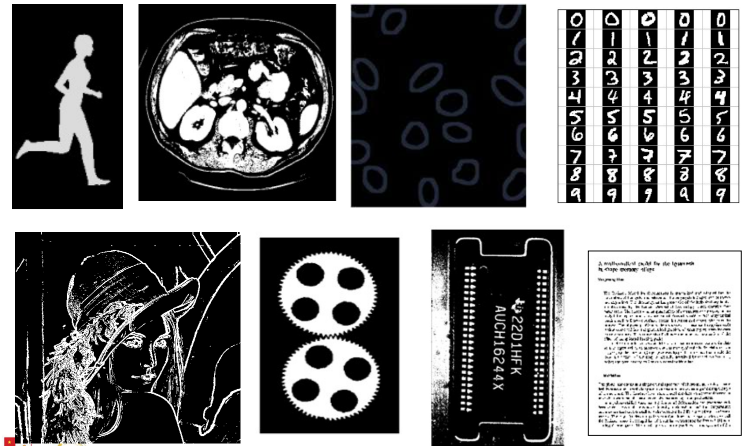
Binary images

- Two types of pixels: foreground pixel (object, note 1) and background pixel (note 0)
- Be used
  - To mark region(s) of interest
  - As results of thresholding method

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



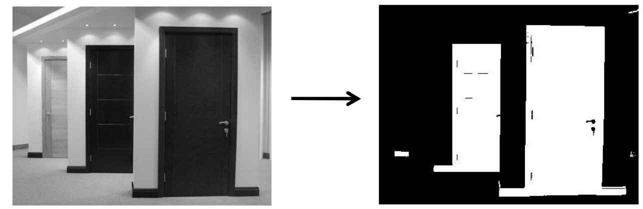
Binary images



Binarization image: Thresholding

- Given a grayscale image or an intermediate matrix → threshold to create a binary output.

Example: Intensity-based detection



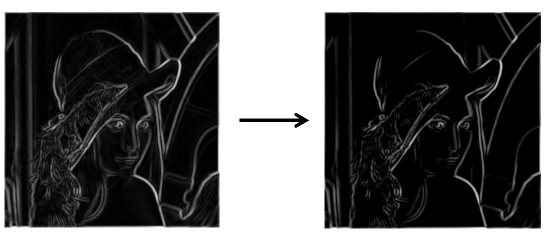
`fg_pix = find(im < 65);`

Looking for dark pixels

Binarization image: Thresholding

- Given a grayscale image or an intermediate matrix → threshold to create a binary output.

Example: Effect of edge detection



Gradient magnitude

`fg_pix = find(gradient_mag > t);`

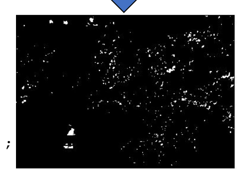
Looking for pixels where gradient is strong.

Binarization image: Thresholding

- Given a grayscale image or an intermediate matrix → threshold to create a binary output.

Example: background subtraction

Looking for pixels that differ significantly from the “empty” background.

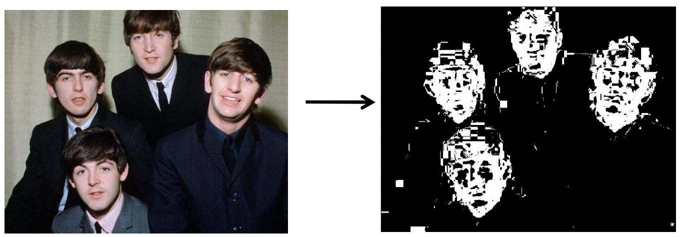


`fg_pix = find(diff > t);`

Binarization image: Thresholding

- Given a grayscale image or an intermediate matrix → threshold to create a binary output.

Example: color-based detection



```
fg_pix = find(hue > t1 & hue < t2);
```

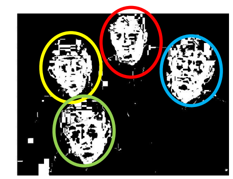
Looking for pixels within a certain hue range.

Slide credit: Kristen Grauman

Issues

- What to do with “noisy” binary outputs?
  - Holes
  - Extra small fragments

- How to demarcate multiple regions of interest?
  - Count objects
  - Compute further features per object



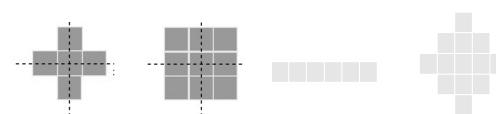
Slide credit: Kristen Grauman

Morphological operators

- Change the shape of the foreground regions via intersection/union operations between a scanning structuring element and binary image.
- Useful to clean up result from thresholding
- Main components
  - Structuring element
  - Operators:
    - Basic operators: Dilation, Erosion
    - Others: Opening, Closing, …

Structuring elements

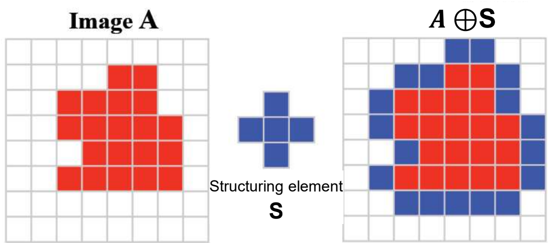
- **Masks** of varying shapes and sizes used to perform morphology, for example:



- Scan mask (structuring element) over the object (foreground) borders (inside and outside) and transform the binary image

Dilation

- Moving S on each pixel of A
  - check if the intersection (pixels belonging to object) is not empty
  - If yes, the center of B belongs to the result image
- If a pixel of S is onto object pixels (A), then the central pixel belongs to object
  - Otherwise (i.e. all pixels of are background), set to background (no change)

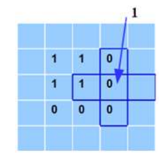


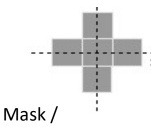
SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

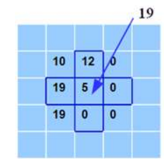
25

Dilation

- As max filter
- Can be applied both on
  - binary images
  - or grayscale images





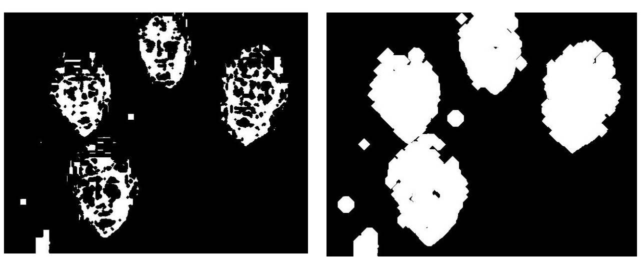


SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

26

Example: Dilation

- Expands connected components
- Grow features
- Fill holes



Before dilation **After dilation**

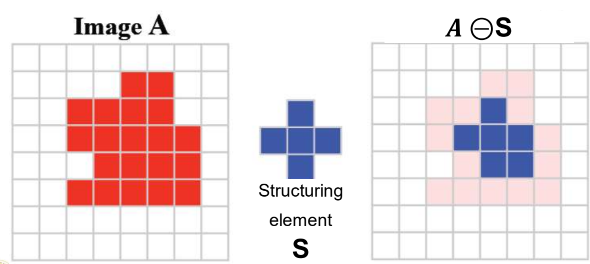
Slide credit: Kristen Grauman

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

27

Erosion

- We put the element S on each pixel x of A
  - like convolution
- If all pixels of S are onto object pixels (A), then the central pixel belongs to object
  - Otherwise (i.e. a mask pixel is background), set to background

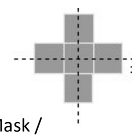


SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

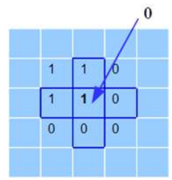
28

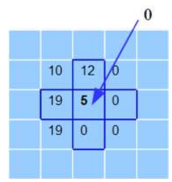
Erosion

- As min filter
- Can be applied both on
  - binary images
  - or grayscale images



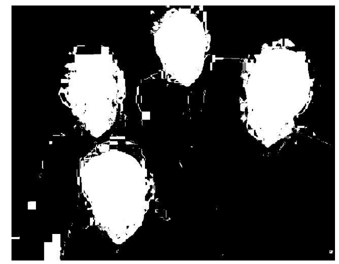
Mask /
Structuring element





Example: Erosion

- Erode connected components
- Shrink features
- Remove bridges, branches, noise

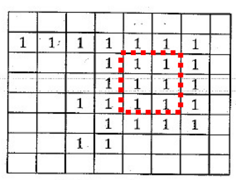


Before erosion

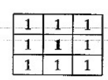
After erosion

Slide credit: Kristen Grauman

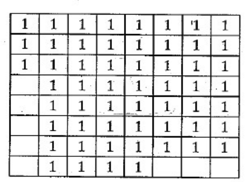
2D example: Dilation, Erosion



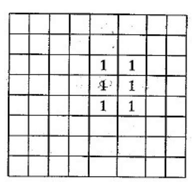
(a) Binary image **B**



(b) Structuring element **S**



(c) Dilation $\mathbf{B} \oplus \mathbf{S}$

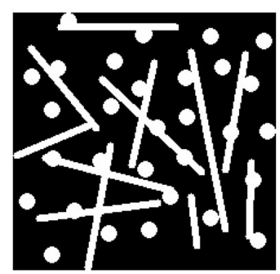


(d) Erosion $\mathbf{B} \ominus \mathbf{S}$

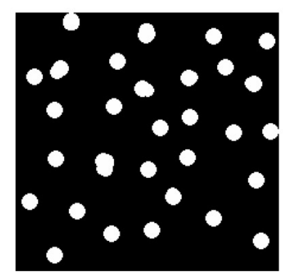
Shapiro & Stockman

Opening

- Erode, then dilate
- Remove small objects, keep original shape



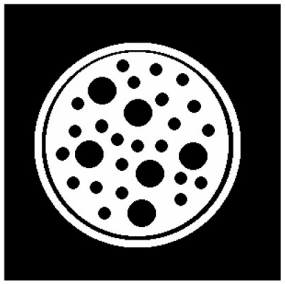
Before opening



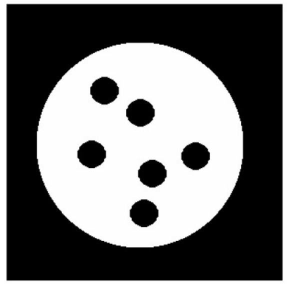
After opening

Closing

- Dilate, then erode
- Fill holes, but keep original shape



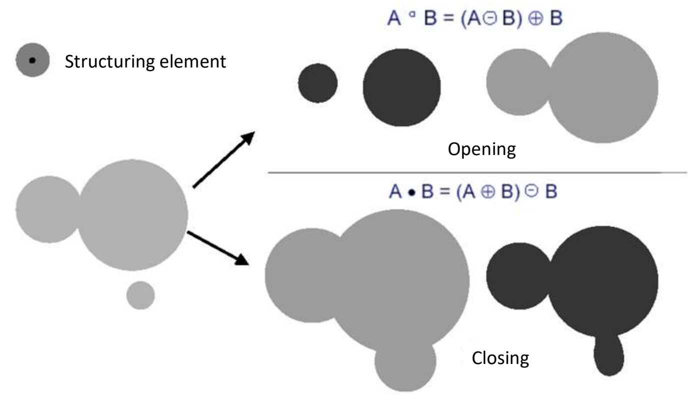
Before closing



After closing

demo: http://bigwww.epfl.ch/demo/jmorpho/start.php

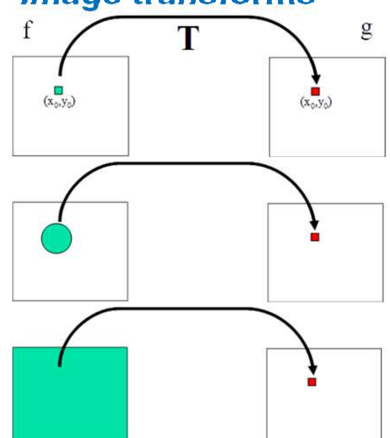
Opening vs Closing



Content

- Remind: digital image representation
- Point processing
- Convolution and linear filtering
- More neighborhood operators
- **Image transforms**
  - Frequency domain
    - Frequencies in images: Spectral image
    - Fourier transform
    - Frequential Processing (frequential filters)
  - KL transform, PCA

Global Pixel transformation Image transforms



Isolated: $g(x_0,y_0)=T[f(x_0,y_0)]$

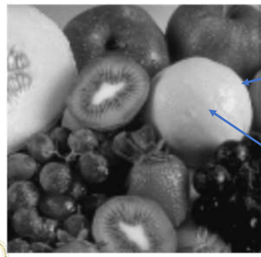
Local: $g(x_0,y_0)=T[f(V)]$
V:neighbors of (x_0,y_0)

Global: $g(x_0,y_0)=T[f(x,y)]$
example: FFT

Source : Caroline Rougier. Traitement d'images (IFT2730). Univ. de Montréal.

Frequency domain
Frequencies in images

- What are the (low/high) frequencies in an image?
  - Frequency **= intensity change**
  - Slow changes (homogeneous /blur regions): low frequency
  - fast/abrupt changes (egde, contour, noise): high frequency

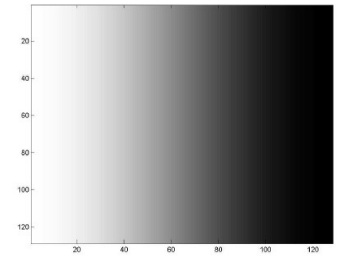


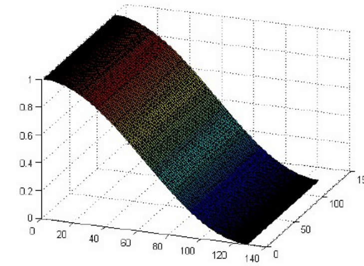
High frequency

Low frequency

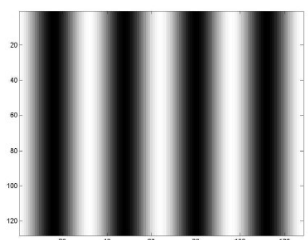
Most of energy concentrated in low frequencies

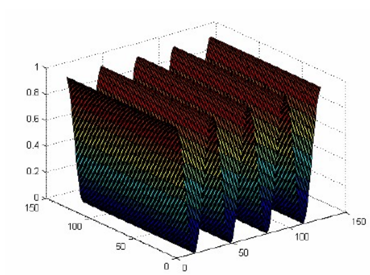
Low frequencies





High frequencies





Fourier Transforms (FT)
2D FT of Continuous signal

- Continuous FFT:

$$F(u,v) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x,y)\, e^{-2\pi i(xu+yv)}\, dx\, dy$$

- Inverse FFT:

$$f(x,y) = \int^{\infty}\int^{\infty} F(u,v)\, e^{2\pi i(xu+yv)}\, du\, dv$$

- FFT: Fast Fourier Transform - Fast FT algorithm

Discrete Fourier Transforms (DFT)

- **Discrete two-dimensional Fourier Transform (DFT) of an image array is defined in series form as follows**;
  - Direct Fourier transform:
$$X(u,v)=\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}X(m,n)\exp\left(-j2\pi\left(\frac{um}{M}+\frac{vn}{N}\right)\right)$$
  - Invers Fourier transform:
$$X(m,n)=\frac{1}{MN}\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}X(u,v)\exp\left(j2\pi\left(\frac{um}{M}+\frac{vn}{N}\right)\right)$$

$\mathbf{X(m,n)}$: Original image MxN; $\mathbf{X(u,v)}$: Spectral image

- The indices (u,v) are called the spatial frequencies
- $|X(u,v)|$: magnitude spectral function, Φ(u,v): phase functions

Both the magnitude and the phase functions are necessary for the complete reconstruction of an image from its Fourier transform

- **FFT**: an efficient computational algorithm for the discrete Fourier Transform

2D FFT - discrete

Direct transform

$$F(u,v)=\frac{1}{MN}\sum_{m=0}^{M-1}\sum_{n=0}^{N-1}f(m,n)\exp\left[-2\pi i\left(\frac{mu}{M}+\frac{nv}{N}\right)\right],$$

$$u=0,1,\dots,M-1, \qquad v=0,1,\dots,N-1,$$

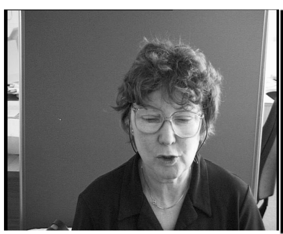
Inverse transform

$$f(m,n)=\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)\exp\left[2\pi i\left(\frac{mu}{M}+\frac{nv}{N}\right)\right],$$

$$m=0,1,\dots,M-1, \qquad n=0,1,\dots,N-1.$$

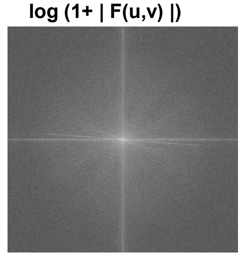
Image Fourier transform

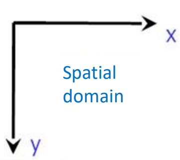
Original image

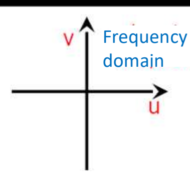


Spectra | F(u,v) |

Enhanced Spectra log (1+ | F(u,v) |)





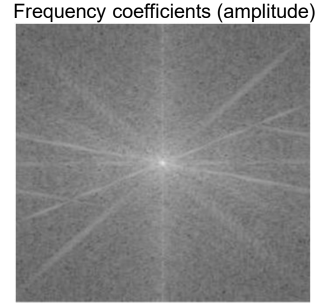


Example: FFT of an image

Natural image

f(x,y)

Fourier decomposition
Frequency coefficients (amplitude)



|F(ω)|

What does it mean to be at pixel x,y?
What does it mean to be more or less bright in the Fourier decomposition image?

Slide by Steve Seitz

Example: FFT of images

- Original image
- Magnitude (amplitude) spectra

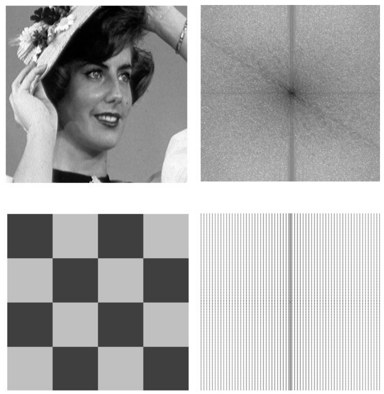
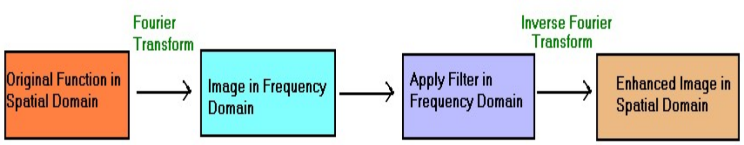


Image spectral analysis

- An image is a visual signal
  - We can analysis the frequencies of the signal
- How?
  - we will create a new « image » which will contains all frequencies of the image
    - Like a 2D frequency graphic
  - The basic tool for it is the **Fourier Transform**
- We talk about the **frequency domain**, opposing to the **spatial domain** (image)
- Image filtering in the frequency domain

Image filtering in the frequency domain

- We will be dealing only with functions (images) of finite duration so we will be interested only in Fourier Transform

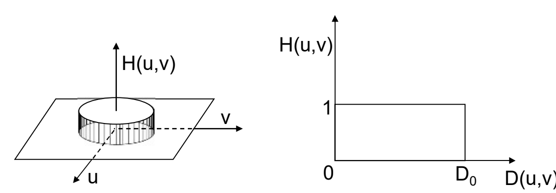


- Linear filters in frequency domain:
  Low Pass Filter, High Pass Filter, Band Pass Filter

H(u,v) - Ideal low pass filter

$$H(u,v) = \begin{cases} 1 & D(u,v) \le D_0 \\ 0 & D(u,v) > D_0 \end{cases} \qquad D(u,v) = \sqrt{u^2 + v^2}$$

D_0 = cut off frequency



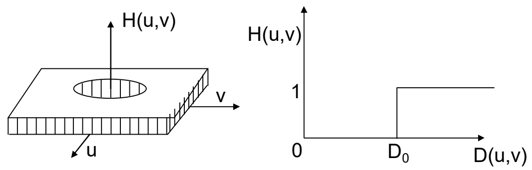
H(u,v) – Ideal high pass filter

H(u,v) - Ideal Filter

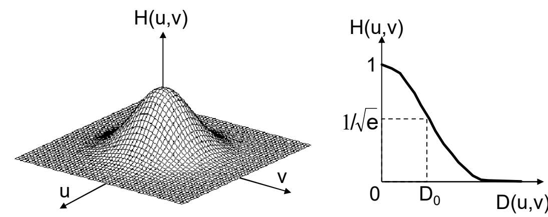
$$H(u,v) = \begin{cases} 0 & D(u,v) \le D_0 \\ 1 & D(u,v) > D_0 \end{cases}$$

$D(u,v) = \sqrt{u^2 + v^2}$

D_0 = cut off frequency



H(u,v) - Gaussian filter

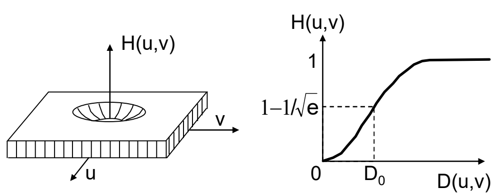


$H(u,v) = e^{-D^2(u,v)/(2D^2_0)}$

$D(u,v) = \sqrt{u_2 + v_2}$

Softer Blurring + no Ringing

High-pass gaussian filter



$H(u,v) = 1 - e^{-D^2(u,v)/(2D^2_0)}$

$D(u,v) = \sqrt{u_2 + v_2}$

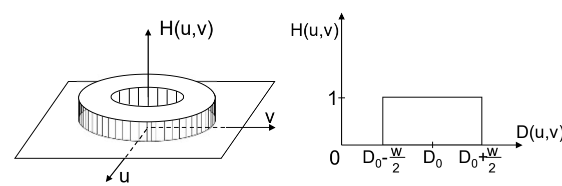
H(u,v) – Band pass filter

$$H(u,v) = \begin{cases} 0 & D(u,v) \le D_0 - \frac{w}{2} \\ 1 & D_0 - \frac{w}{2} \le D(u,v) \le D_0 + \frac{w}{2} \\ 0 & D(u,v) > D_0 + \frac{w}{2} \end{cases}$$

$D(u,v) = \sqrt{u^2 + v^2}$

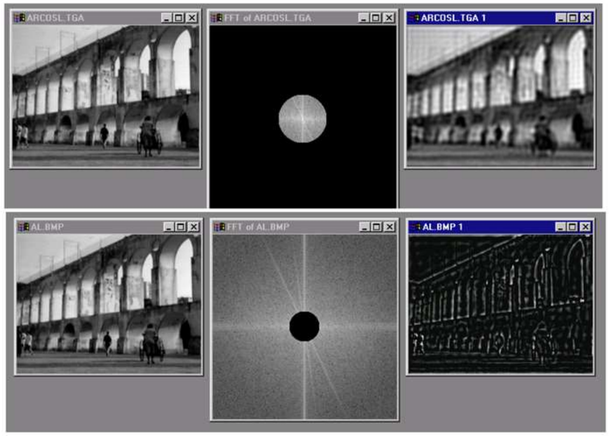
D_0 = cut off frequency

w = band-width



Can be obtained by multiplying the filter functions of **a low-pass and of a high-pass** in the frequency domain

Low-pass and high-pass filtering



Slide by Alexei Efros

Low pass filtering

spatial domain → frequency domain

f(x,y) → F(u,v)

G(u,v) = F(u,v).H(u,v)

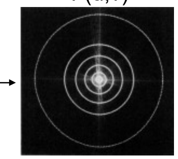
filter

g(x,y) ← G(u,v)

Purpose: smoothing noise

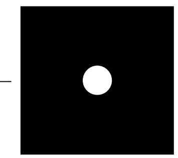
f(x,y)

F(u,v)



•

g(x,y)

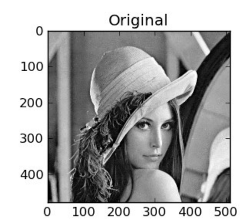


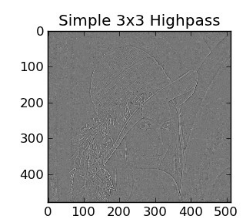
H(u,v)

Blurring - Ideal low-pass filters

99.7% 99.37% 98.65%

High pass filtering





$$h_1(3x3) = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

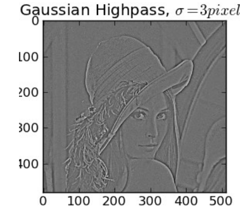


Image filtering: Hybrid Images

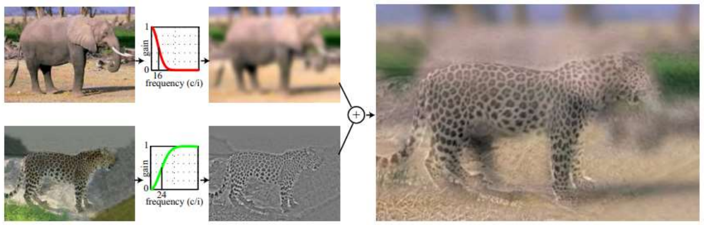


Figure 2: hybrid images are generated by superimposing two images at two different spatial scales: the low-spatial scale is obtained by filtering one image with a low-pass filter, and the high spatial scale is obtained by filtering a second image with a high-pass filter. The final hybrid image is composed by adding these two filtered images.

A. Oliva, A. Torralba, P.G. Schyns, SIGGRAPH 2006

Principle Component Analysis (PCA) (Karhunen-Loeve transform)

- **PCA** transforms the original input space into a lower dimensional space
  - By constructing dimensions that are linear combinations of the given features
- The objective: consider independent dimensions along which data have largest variance (i.e., greatest variability - uncorrelation)

Slide by Jana Kosecka

Principle Component Analysis (cont.)

- **PCA** enables transform a number of possibly correlated variables into a smaller number of uncorrelated variables called principal components
- The first principal component accounts for as much of the variability in the data as possible
- Each succeeding component (orthogonal to the previous ones) accounts for as much of the remaining variability as possible

Slide by Jana Kosecka

Principal Component Analysis (cont.)

- PCA is the most commonly used dimension reduction technique.
- Data samples $x_1, \cdots, x_N$
- Compute the mean $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$
- Computer the covariance matrix:

$$\Sigma_x = \frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})(x_i - \bar{x})^T$$

Slide by Jana Kosecka

Principal Component Analysis (cont.)

- Compute the eigenvalues λ and eigenvectors e of the matrix Σ_x
- Solve $\Sigma_x x = \lambda x$
- Order them by magnitude:

$$\lambda_1 \geq \lambda_2 \geq .\lambda_N.$$

- PCA reduces the dimension by keeping direction e such that $\lambda < T$.

Slide by Jana Kosecka

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Principal Component Analysis (cont.)

- How to get uncorrelated components which Capture most of the variance
- Project the data onto the selected eigenvectors:

$$y_i = e_i^T (x_i - \overline{x})$$

- If we consider first M eigenvectors we get new lower dimensional representation

$$[y_1, \cdots, y_M]$$

- Proportion covered by first M eigenvalues

$$\frac{\sum_{i=1}^{M} \lambda_i}{\sum_{i=1}^{N} \lambda_i}$$

Slide by Jana Kosecka

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

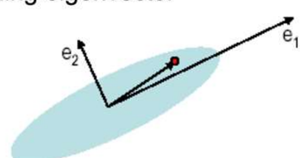
Principal Component Analysis (cont.)

- For many datasets, most of the eigenvalues are negligible and can be discarded.

The eigenvalue λ measures the variation In the direction of corresponding eigenvector

Example:

$$\lambda_1 \neq 0, \lambda_2 = 0.$$



Slide by Jana Kosecka

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Determining the number of components

- Plot the eigenvalues
  - each eigenvalue is related to the amount of variation explained by the corresponding axis (eigenvector)
  - If the points on the graph tend to level out (show an “elbow” shape), these eigenvalues are **usually close enough to zero that they can be ignored**

Slide by Jana Kosecka

SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY

Phép biến đổi KL – PCA (Phân tích thành các thành phần chính - Principle Component Analysis)

- Không gian vector dữ liệu n chiều: $\chi = \{X\}, X=[x_i]\ \forall\ x_i \in R, i \in [1, n]$.
- **PCA** là phép biến đổi đơn vị theo công thức: $\mathbf{Y} = \Phi\,\mathbf{X}$ sao cho thỏa mãn:

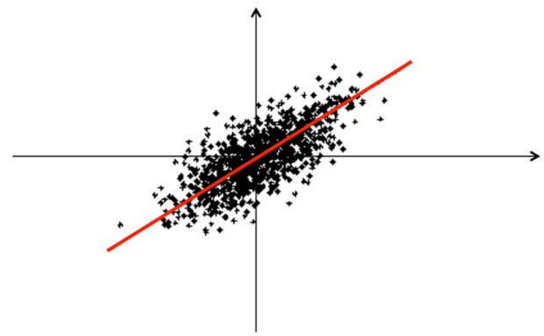
$$Y = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{pmatrix}$$

Vector Y bao gồm các thành phần không tương quan, nghĩa là mọi thành phần y_i, y_k, $\forall$i, k $\in$[1,n] thỏa mãn hiệp phương sai bằng 0:

$$C_{ij} = E[y_i y_k] - E[y_i].E[y_k] = 0$$

Φ là ma trận của phép biến đổi, còn gọi ma trận chiếu, được định nghĩa như sau: $\Phi = [e_i^T]$ với $i \in [1, n]$.
e_i : là các vectơ riêng của **ma trận hiệp phương sai** của các vector X
e_i : là các vectơ cơ sở trực giao của không gian đặc trưng
KL-PCA giảm thứ nguyên: chọn ***m<n*** tương ứng với ***m* giá trị riêng lớn** nhất của ma trận hiệp phương sai. Kết quả "*m* - dimensional subspace"

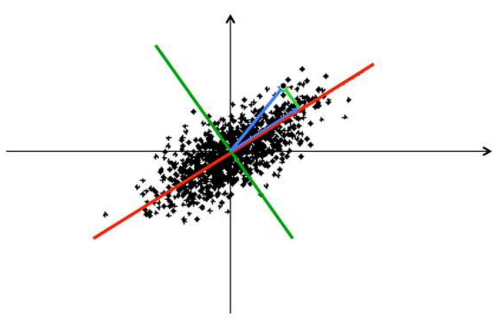
Illustration of PCA



First principal component of a two-dimensional data set.

Slide by Jana Kosecka

Illustration of PCA

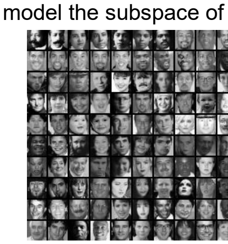


Second principal component of a two-dimensional data set.

Slide by Jana Kosecka

PCA application: Using PCA for face recognition

- When viewed as vectors of pixel values, face images are extremely high-dimensional
  - 100x100 image = 10,000 dimensions
- However, relatively few 10,000-dimensional vectors correspond to valid face images
- We want to effectively model the subspace of face images



Slide by Jana Kosecka

Eigenfaces: Key idea

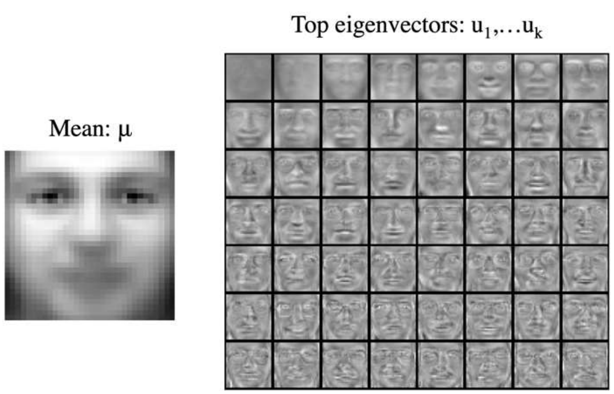
- Assume that most face images lie on a low-dimensional subspace determined by the first k ($k<d$) directions of maximum variance
- Use PCA to determine the vectors or "eigenfaces" $\mathbf{u}_1,\ldots,\mathbf{u}_k$ that span that subspace
- Represent all face images in the dataset as linear combinations of eigenfaces

Slide by Jana Kosecka

Eigenfaces example- Traning images

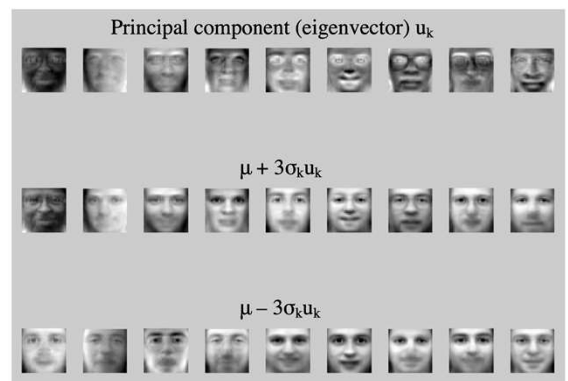
Slide by Jana Kosecka

Eigenfaces example



Slide by Jana Kosecka

Eigenfaces example



Slide by Jana Kosecka

Trích chọn đặc trưng dùng PCA

Các bước thực hiện PCA

B1: Tính vector trung bình $\mu = E\{X_k\}$ và tính các vector :

$$X_k = X_k - \mu$$

B2: Tính ma trận hiệp phương sai của các vector X trong bộ dữ liệu quan sát (ảnh ban đầu): $C_X = E\{XX^T\} = \frac{1}{M}(XX^T)$

B3: Giải phương trình đặc tính tìm được các trị riêng của ma trận hiệp phương sai C_X, và kết hợp điều kiện trực giao được *n* vector riêng ứng *n* trị riêng. Các vector riêng trực giao ở trường hợp nhận dạng khuôn mặt được gọi là **Eigenfaces.**

B4: Trích chọn thành phần chính**:** Chọn ra *m* vector riêng ứng với *m* trị riêng lớn nhất ($m<n$) để thiết lập ma trận phép biến đổi hay ma trận chiếu Φ

B5 : Thực hiện phép biến đổi PCA theo quan hệ : $\mathbf{Y} = \Phi\,\mathbf{X}$

➢ **Kết quả đạt được là bộ dữ liệu mới với các vector Y là các vector đặc trưng của đối tượng.**

Recognition with eigenfaces

- Process labeled training images:
- Find mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$
- Find k principal components (eigenvectors of $\boldsymbol{\Sigma}$) $\mathbf{u}_1,\ldots\mathbf{u}_k$
- Project each training image $\mathbf{x}_i$ onto subspace spanned by principal components: $(w_{i1},\ldots,w_{ik}) = (\mathbf{u}_1^T(\mathbf{x}_i - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x}_i - \boldsymbol{\mu}))$

- Given novel image $\mathbf{x}$:
- Project onto subspace: $(w_1,\ldots,w_k) = (\mathbf{u}_1^T(\mathbf{x} - \boldsymbol{\mu}), \ldots, \mathbf{u}_k^T(\mathbf{x} - \boldsymbol{\mu}))$
- Optional: check reconstruction error $\mathbf{x} - \hat{\mathbf{x}}$ to determine whether image is really a face
- Classify as closest training face in k-dimensional subspace

Slide by Jana Kosecka

Exercise 1

- Given an 8-bit image – 8 x 8

$$\begin{bmatrix} 52 & 55 & 61 & 66 & 70 & 61 & 64 & 73 \\ 63 & 59 & 55 & 90 & 109 & 85 & 69 & 72 \\ 62 & 59 & 68 & 113 & 144 & 104 & 66 & 73 \\ 63 & 58 & 71 & 122 & 154 & 106 & 70 & 69 \\ 67 & 61 & 68 & 104 & 126 & 88 & 68 & 70 \\ 79 & 65 & 60 & 70 & 77 & 68 & 58 & 75 \\ 85 & 71 & 64 & 59 & 55 & 61 & 65 & 83 \\ 87 & 79 & 69 & 68 & 65 & 76 & 78 & 94 \end{bmatrix}$$

1) Compute and show the histogram:

2) Compute the Power-law Transformation using $s = c.r^{\gamma}$

with $c = 0.1$; $r = 1$, $\gamma = 1$

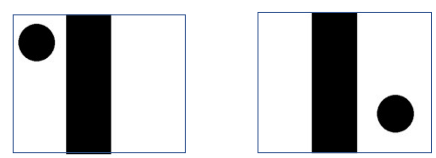
3) Comment about the contrast of the image and make a modification of contrast

4) Equalize the histogram for above image with 8-bins

5) Compute the filtered image by the filters such as: Mean filter, Laplacian filter, Median filter, Min/ Max filter

Exercise 2

- Given two images as bellow



1. Transform images to negative ones
2. Process to have an image which has only the "ball"