

Digital Image Processing and Applications

Dr. Xigun Lu

College of Computer Science

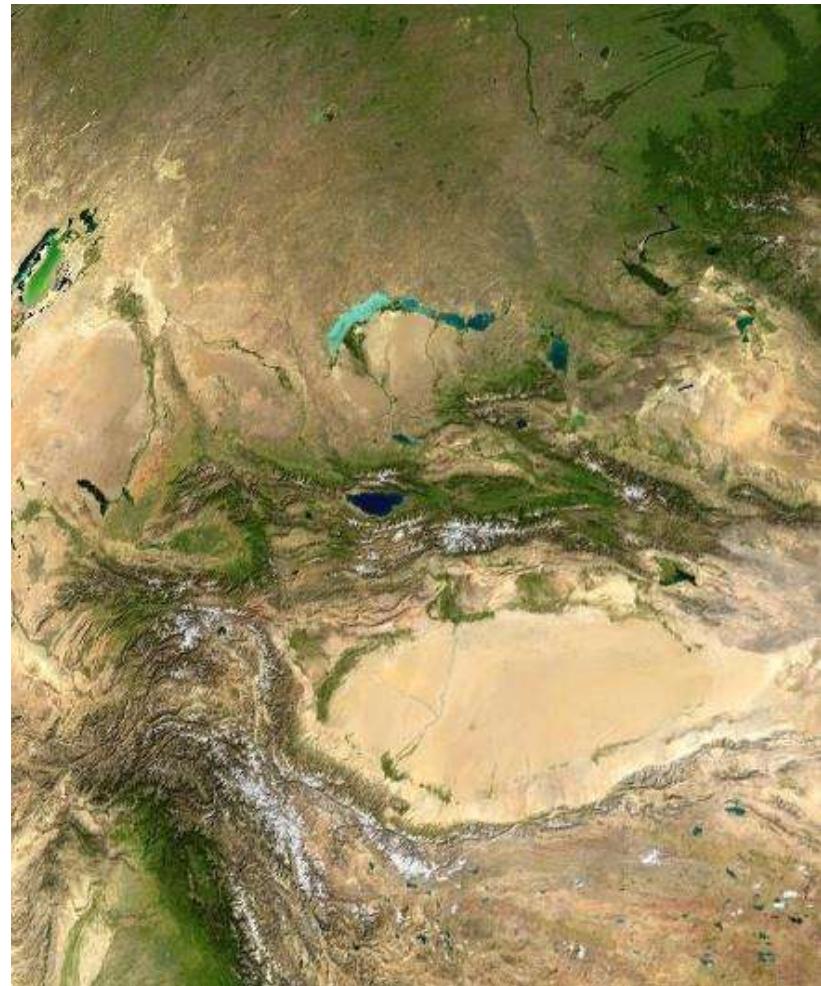
Zhejiang University

Course Outline

- Image Representation
- Motion Estimation (including Optical Flow Techniques)
- Image & Video Compression
- Human Visual System
- Image Halftone Techniques
- Image Filtering Techniques
- Image Interpolation & Super-resolution
- Edge Detection

A picture is worth a thousand words.
A video is worth a thousand sentences.

- Rich information from visual data
- Examples of images around us
 - Natural photographic images
 - Artistic and engineering drawings
 - Scientific images (satellite, medical, etc.)
- Motion picture – video
 - Movies, TV programs, news
 - Family video
 - Surveillance and highway camera



Why do we process images & videos?

- Enhancement and restoration
 - To remove artifacts and scratches from an old photo/movie
 - To improve contrast and correct blurred images/videos
- Transmission and storage
 - To transmit images from oversea via Internet, or from a remote planet
- Information analysis and automated recognition
 - To provide “human vision” to machines ~ computer vision
- Security and rights protection
 - encryption and watermarking

Why Digital?

- “Exactness”
 - Perfect reproduction without degradation
 - Perfect duplication of processing result
- Convenient & powerful computer-aided processing
 - Can perform rather sophisticated processing through hardware or software
 - Even kindergartners can do it!
- Easy storage and transmission
 - Paperless transmission of high quality photos/videos through network within seconds

Compression

- **Color image of 600×800 pixels**
 - Without compression
 - $600 \times 800 \times 24 \text{ bits/pixel}$
 $= 11.52\text{K bytes} = 1.44\text{M bytes}$
 - After JPEG compression
(popularly used on web)
 - only 89K bytes
 - compression ratio $\sim 16:1$
- **Movie**
 - 720 x 480 per frame, 30 frames/sec, 24 bits/pixel
 - Raw video $\sim 243\text{M bits/sec}$
 - DVD \sim about 5M bits/sec
 - Compression ratio $\sim 48:1$



“Library of Congress” by Dr. M.Wu
(@Maryland University)

Image Halftone



Original Image



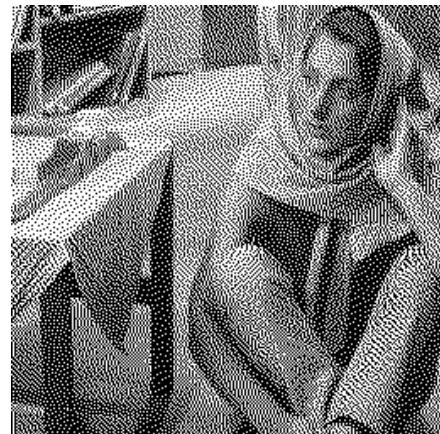
Threshold at Mid-Gray



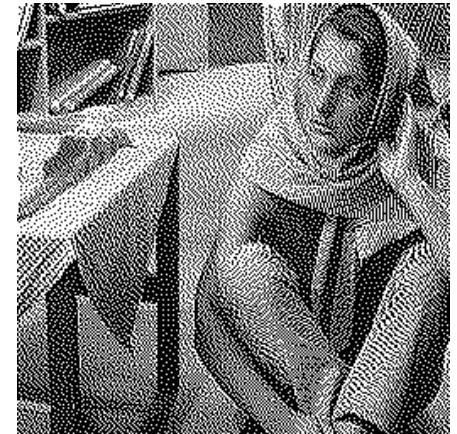
Dispersed Dot Screening



Clustered Dot Screening



Floyd Steinberg Error Diffusion



Stucki Error Diffusion

Image Denoising



Median Filtering



Bilateral Filtering [1]

Image Interpolation & Superresolution



clear



nearest

bilinear

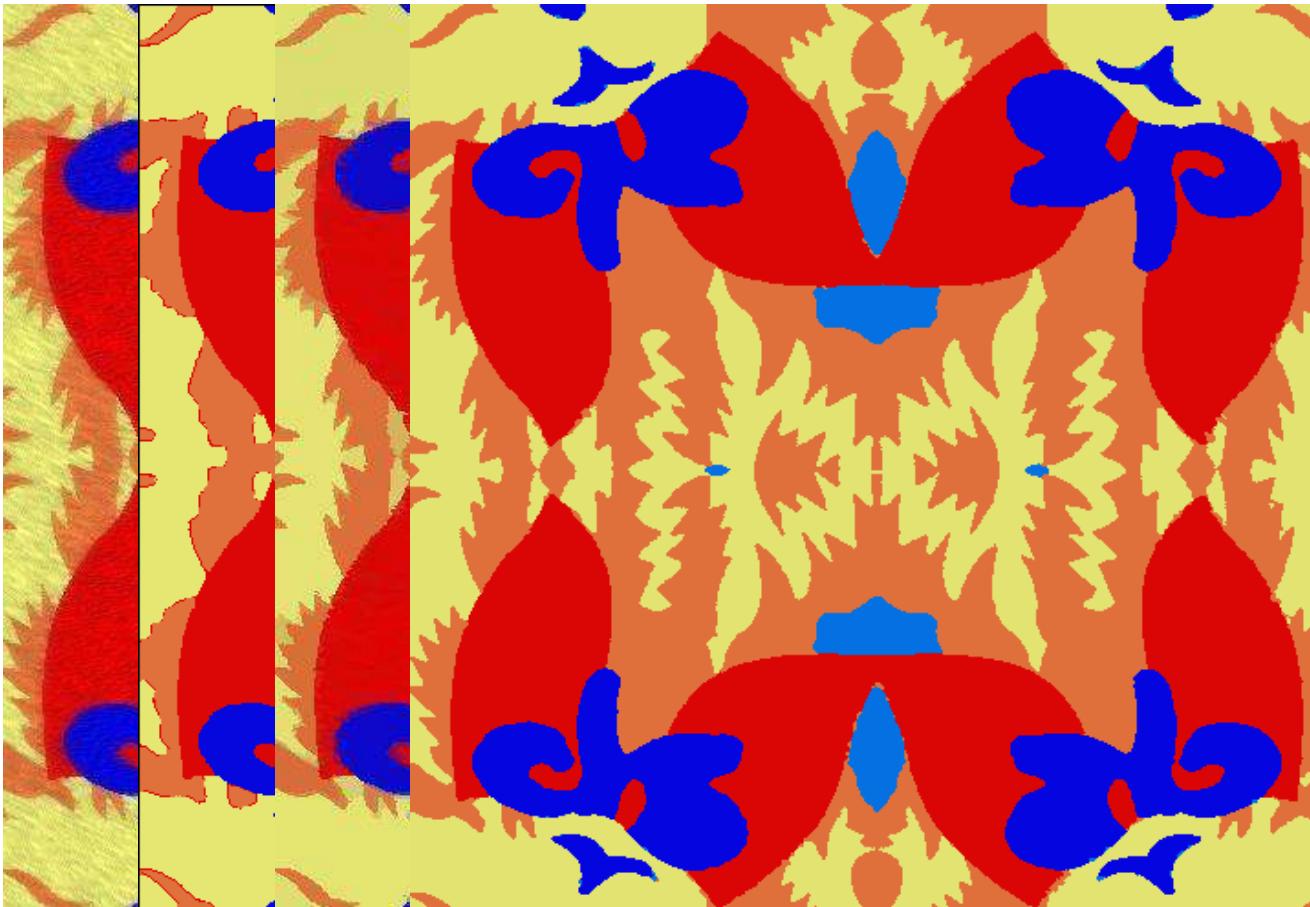
bicubic

in scale BP

ICCV2009 [2]



Image Segmentation



Multiscale Context Model [5]

References

- [1] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in Proc. of IEEE International Conference on Computer Vision (ICCV), 1998.
- [2] D. Glasner, S. Bagon, and M. Irani, “Super-resolution from a single image,” in Proc. of IEEE International Conference on Computer Vision (ICCV), pp.349-356, 2009.
- [3] P. Felzenszwalb and P. Huttenlocher, “Efficient belief propagation for early vision,” International Journal of Computer Vision, vol.70, no.1, 2006.
- [4] D. Comaniciu and P. Meer, “Mean shift: a robust approach toward feature space analysis,” IEEE Trans. On Pattern Analysis and Machine Intelligence, vol.24, no.5, pp. 603-619, 2002.
- [5] X. Lu, “Color textile image segmentation based multiscale probabilistic reasoning,” Optical Engineering, vol.46, no.8, 087002, 2007.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Spatial Domain Representations of Natural Images

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Image Representation

SPATIAL DOMAIN REPRESENTATIONS

Different Image Types

- Binary images (0 or 1)
- Gray images ([0, 255] or 8 bits/pixel)
- Color images
 - Indexed color images (based on palette)
 - Full color images (24 bits/pixel, 8 bits for red, 8 bits for green, and 8 bits for blue)

Is this a binary image or a gray image?



This is a halftoned image. It looks like a gray image, but it is a binary image.

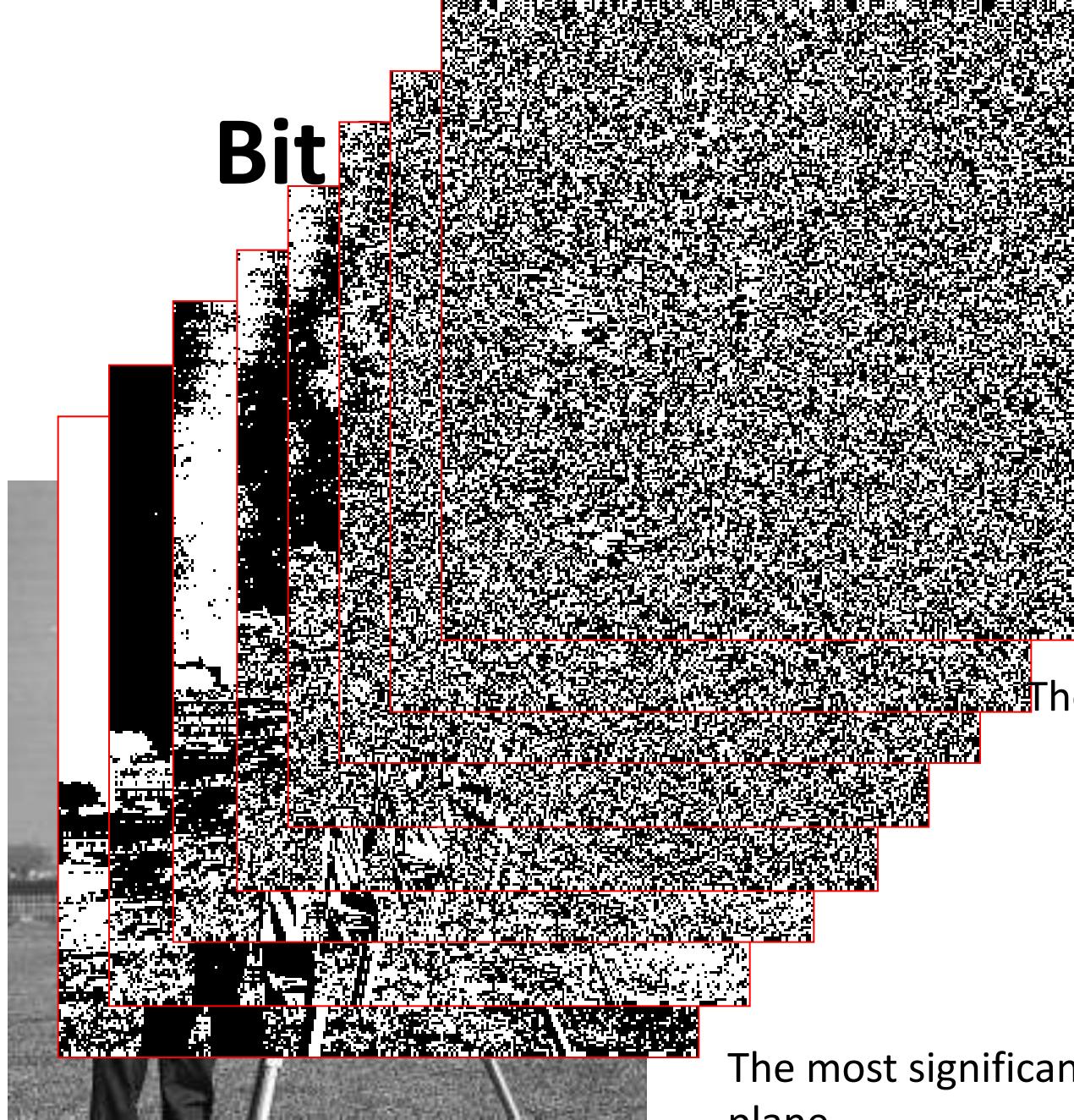
Perceived Gray Tone



Lena



Halftoned Lena



A standard gray image

image

The least significant bit plane

The most significant bit
plane

Bit Depth (# of Gray Levels)

256



32



16



8



4



2



Full Color Images

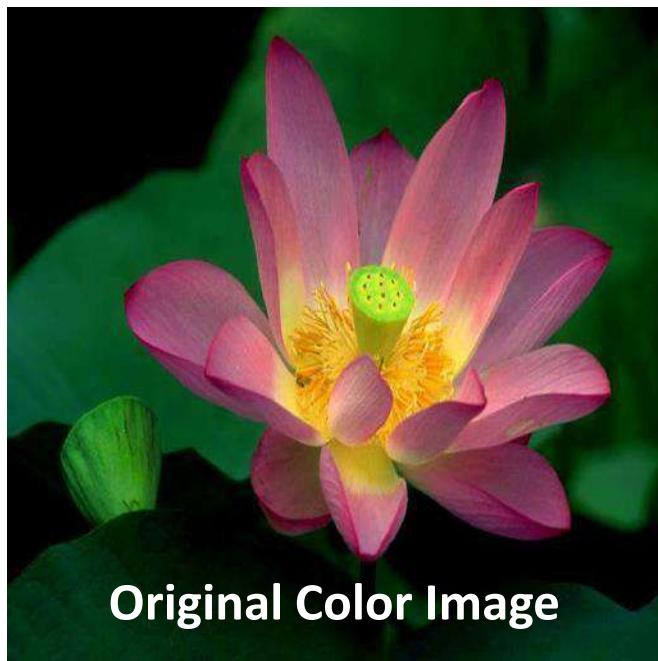
- 24 bits per pixel, and the three channels R G B are three gray images respectively.

```
class Image {  
    unsigned int width;  
    unsigned int height;  
    unsigned char *data; →  
}
```

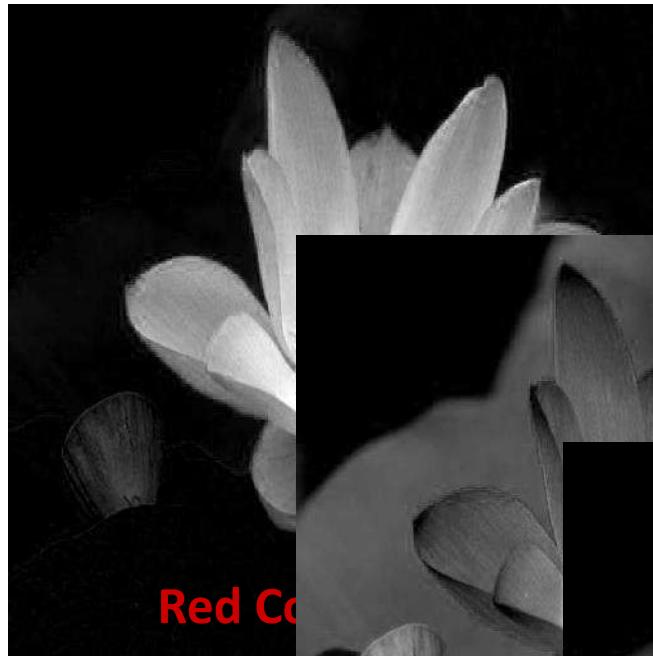
0 _r	0 _g	0 _b	1 _r	1 _g	1 _b	2 _r	2 _g	2 _b	3 _r	3 _g
----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------	----------------

0 _{r,g,b}	1 _{r,g,b}	2 _{r,g,b}
3 _{r,g,b}	4 _{r,g,b}	5 _{r,g,b}
6 _{r,g,b}	7 _{r,g,b}	8 _{r,g,b}

Color Components of a Color Image



Original Color Image



Red Component



Green Component



Blue Component

An Indexed Color Image

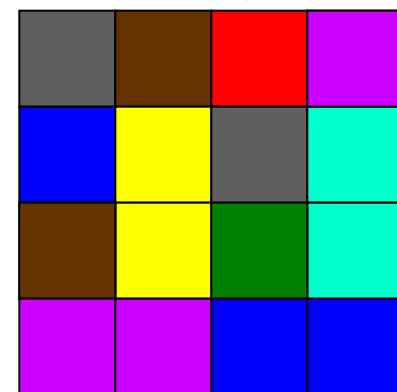
Color Table

0	Red
1	Blue
2	Magenta
3	Brown
4	Grey
5	Cyan
6	Green
7	Yellow

Pixel Data

4	3	0	2
1	7	4	5
3	7	6	5
2	2	1	1

Image



A GIF Image



Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Fourier Transform (I)

Dr. Xiqun Lu

College of Computer Science

Zhejiang University

Image Representation

PRIORI BASIS FOR NATURAL IMAGES

Basic Exponential Signals (I)

- Basic building signals $e^{j\omega t}$ and $e^{j\Omega n}$
 - ω and Ω denote the radian frequencies in the continuous-time and discrete-time domains, respectively.
- The properties of continuous-time exponential signals $e^{j\omega t}$
 - This is a **periodic** signal
 - The fundamental period is $T=2\pi/\omega$
 - the larger the magnitude of ω , the higher rate of oscillation of the signal.
 - $e^{j\omega_1 t}$ and $e^{j\omega_2 t}$ are **orthogonal** to each other whenever $|\omega_1| \neq |\omega_2|$

$$e^{j\omega t} = e^{j\omega(t+kT)} = e^{j\omega(t+k\frac{2\pi}{\omega})}$$

To prove $e^{j\omega_1 t}$ and $e^{j\omega_2 t}$ are **orthogonal** to each other whenever $|\omega_1| \neq |\omega_2|$.
 Because they are periodic signals, we only need to prove they are orthogonal to each other within a common period.

Now assume T is the least common multiple of $T_1 = 2\pi/\omega_1$ and $T_2 = 2\pi/\omega_2$.

$$\text{Assume } T = k_1 T_1 = k_1 \frac{2\pi}{\omega_1} \quad \text{and} \quad T = k_2 T_2 = k_2 \frac{2\pi}{\omega_2} \quad (k_1, k_2 \in \mathbb{Z}^+)$$

$$\begin{aligned} \int_0^T e^{j\omega_1 t} \overline{e^{j\omega_2 t}} dt &= \int_0^T e^{j(\omega_1 - \omega_2)t} dt = \frac{1}{j(\omega_1 - \omega_2)} e^{j(\omega_1 - \omega_2)t} \Big|_0^T \\ &= \frac{1}{j(\omega_1 - \omega_2)} (e^{j(k_1 - k_2)2\pi} - 1) = 0 \end{aligned}$$

Basic Exponential Signals (II)

- The properties of discrete-time exponential signals $e^{j\Omega n}$

$$e^{j\Omega n} = e^{j\Omega(n+N)} = e^{j(\Omega n + 2\pi m)}$$

- is **NOT** periodic for **arbitrary** values of Ω

- Only when $\Omega/2\pi = m/N$ (m and N are integers, i.e. only when $\Omega/2\pi$ is a rational number)
 - But $e^{j\Omega n}$ is a **periodic** signal w.r.t. Ω . $e^{j\Omega n} = e^{j(\Omega+2\pi)n}$

- $e^{j\Omega n}$ are **NOT distinct**, as the signal with frequency Ω_0 is identical to the signals with frequencies $(\Omega_0 \pm 2\pi)$, $(\Omega_0 \pm 4\pi)$, ... and so on $(\Omega_0 \pm 2\pi k)$.

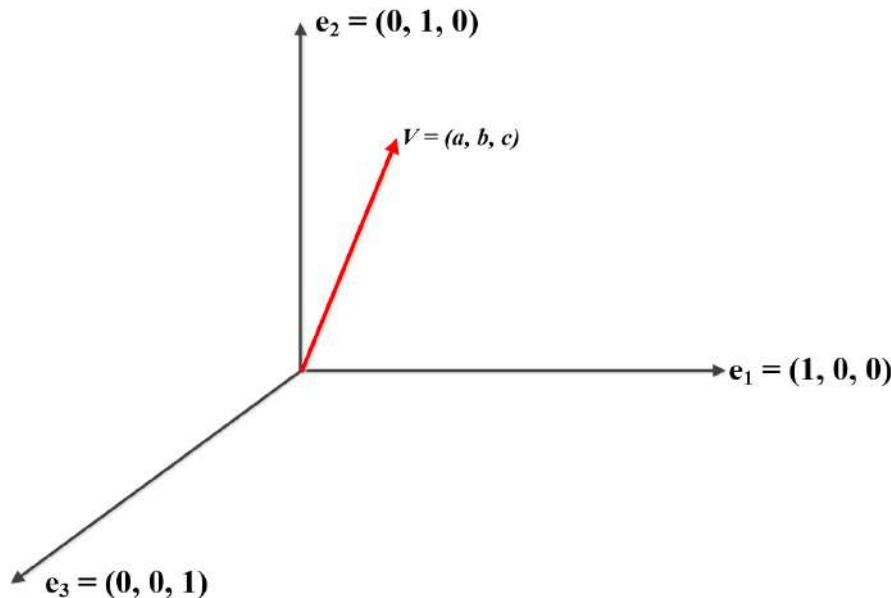
$$e^{j\omega t} \quad \text{vs.} \quad e^{j\Omega n}$$

$e^{j\omega t}$	$e^{j\Omega n}$
Distinct signals for distinct <i>magnitudes</i> of ω	Identical signals for exponential at frequencies separated by 2π
Periodic for any choice of ω	Periodic only if $\Omega_0=2\pi(m/N)$ for some integers $N > 0$ and m .
Fundamental frequency ω_0	Fundamental frequency if it is periodic: Ω_0/m
Fundamental period $\omega_0=0$: undefined $\omega_0\neq 0$: $2\pi/\omega_0$	Fundamental period if it is periodic $\Omega_0=0$: undefined $\Omega_0\neq 0$: $m(2\pi/\Omega_0)$

$e^{j\omega_1 t}$ and $e^{j\omega_2 t}$ are **orthogonal** to each other whenever $|\omega_1| \neq |\omega_2|$

Euclidean Geometric Space

- 3D Euclidean Geometric Space
 - 3 basic vectors



– Analysis

$$a = \mathbf{v} \cdot \mathbf{e}_1 \quad b = \mathbf{v} \cdot \mathbf{e}_2 \quad c = \mathbf{v} \cdot \mathbf{e}_3$$

– Synthesis

$$\mathbf{v} = a\mathbf{e}_1 + b\mathbf{e}_2 + c\mathbf{e}_3$$

Signal Space

- The number of basic signals is infinite.
- This is a complex space

– Analysis

$$X(\omega) = \int_{-\infty}^{+\infty} x(t) \overline{e^{j\omega t}} dt = \int_{-\infty}^{+\infty} x(t) e^{-j\omega t} dt$$

– Synthesis

$$x(t) = \int_{-\infty}^{+\infty} X(\omega) e^{j\omega t} d\omega$$

Continuous-Time Fourier Transform

- A representation of continuous-time **aperiodic** signals

- Analysis

$$X(f) = \int_{-\infty}^{+\infty} x(t)e^{-j2\pi ft} dt \quad or \quad X(\omega) = \int_{-\infty}^{+\infty} x(t)e^{-j\omega t} dt$$

- Synthesize

$$x(t) = \int_{-\infty}^{+\infty} X(f)e^{j2\pi ft} df \quad or \quad x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} X(\omega)e^{j\omega t} d\omega$$

Dirichlet Conditions

- A continuous-time **aperiodic** signal $x(t)$ has a Fourier transform only if it satisfies the following conditions:

- 1) $x(t)$ is absolutely integrable, namely

$$\int_{-\infty}^{+\infty} |x(t)| dt < \infty$$

- 2) $x(t)$ has only a finite number of maximum and minima within any finite interval.
 - 3) $x(t)$ has only a finite number of discontinuities within any finite interval.

References

- [1] A. V. Oppenheim, A. S. Willsky and I. T. Young, Signals and Systems, Prentice-Hall, 1983.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Fourier Transform (II)

Dr. Xiqun Lu

College of Computer Science

Zhejiang University

Image Representation

PRIORI BASIS FOR NATURAL IMAGES

Continuous-time Fourier Series

- Suppose $x(t)$ is a continuous-time **periodic** signal: $x(t) = x(t + kT_0)$
 - The basic signals are $e^{jk\omega_0 t}$ ($k = 0, \pm 1, \pm 2, \dots$) ($\omega_0 = 2\pi/T_0$)

- Analysis

$$a_k = \frac{1}{T_0} \int_{T_0} x(t) e^{-jk\omega_0 t} dt$$

- The coefficients $\{a_k\}$ are often called the Fourier series coefficients or the spectral coefficients of $x(t)$
- Synthesis

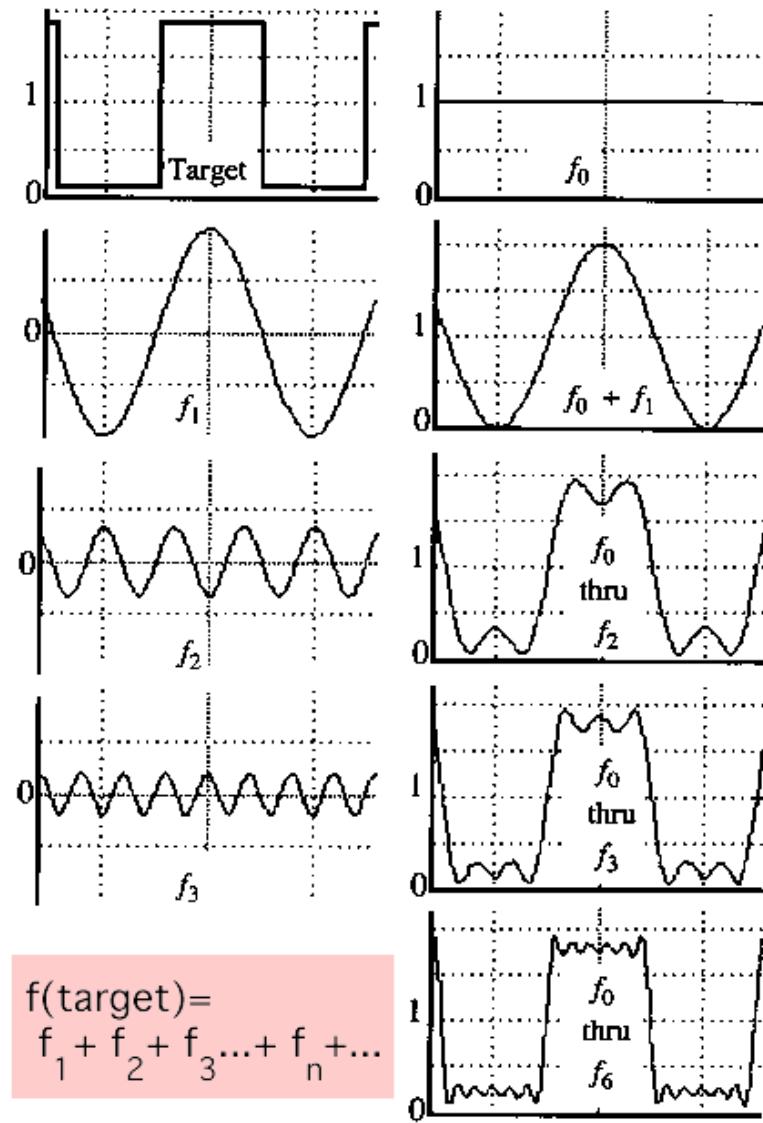
$$x(t) = \sum_{k=-\infty}^{+\infty} a_k e^{jk\omega_0 t}$$

- If $x(t)$ is a real signal, then $a_k = a_{-k}$.

Dirichlet Conditions

- A **periodic** signal $x(t)$, has a Fourier series only if it satisfies the following conditions:
 - 1) $x(t)$ is absolute integrable over any period, namely
$$\int_{\alpha}^{\alpha+T} |x(t)| dt < \infty \quad \alpha \in R$$
 - 2) $x(t)$ has only a finite number of maximum and minima over any period
 - 3) $x(t)$ has only a finite number of discontinuities over any period

An Periodic Square Wave



Discrete-time Fourier Series

- How about $x[n]$ is a *discrete-time periodic* signal?
 - $x[n] = x[n + N]$ or $x[n] = x[n + kN]$ ($k \in \mathbb{Z}$)
 - Now if we divide the circle 2π into N points, we will get N different discrete frequencies $e^{jk\frac{2\pi}{N}n}$, $k \in \{0, 1, \dots, N-1\}$ or $k \in \{1, 2, \dots, N\}$, and so on. ($k = \langle N \rangle$)
 - $e^{jk_1\frac{2\pi}{N}n}$ and $e^{jk_2\frac{2\pi}{N}n}$ are orthogonal to each other whenever $k_1 \neq k_2$ and $k_1, k_2 \in \langle N \rangle$ (the set of N consecutive integer numbers)
 - **An important distinction** between the set of harmonically related signals in discrete-time and continuous-time is
 - There are **only N** different signals $e^{jk\frac{2\pi}{N}n}$ in the set $k = 0, \pm 1, \pm 2, \dots$
 - Whereas all of the $e^{jk\omega_0 t}$ ($k = 0, \pm 1, \pm 2, \dots$) are **distinct**.

Discrete-time Fourier Series

- Suppose $x[n]$ is a periodic signal in discrete-time domain
 - Remember that we only have N different signals in the set $k = 0, \pm 1, \pm 2, \dots$
- Analysis
$$a_k = \frac{1}{N} \sum_{n=<N>} x[n] e^{-jk\frac{2\pi}{N}n}$$
- Synthesis
$$x[n] = \sum_{k=<N>} a_k e^{jk\frac{2\pi}{N}n}$$

Discrete-time Fourier Transform

- Now suppose $x[n]$ is an **aperiodic** signal in *discrete-time* domain

- Analysis

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$$

- Synthesis

$$x[n] = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega) e^{j\Omega n} d\Omega$$

- The continuous periodic in the frequency domain

$$X(\Omega + 2\pi) = X(\Omega)$$

Summary of Basis Signals

- **Continuous-time**
 - Fourier Series for periodic signals — $e^{jk\omega_0 t}$
 - Fourier Transform for aperiodic Signals — $e^{j\omega t}$
- **Discrete-time**
 - Discrete-time Fourier series for periodic signals — $e^{jk\frac{2\pi}{N}n}$
 - Discrete-time Fourier Transform for aperiodic Signals
 - $e^{j\Omega n}$

References

- [1] A. V. Oppenheim, A. S. Willsky and I. T. Young, Signals and Systems, Prentice-Hall, 1983.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Fourier Transform (III)

— DFT

Dr. Xiqun Lu

*College of Computer Science
Zhejiang University*

Image Representation

PRIORI BASIS FOR NATURAL IMAGES

Discrete Fourier Transform (DFT)

- A **finite duration aperiodic signal** $x[n]$, $x[n] = 0$ outside of the interval $0 \leq n \leq N_1$.
- Construct a periodic signal $\tilde{x}[n]$ with period of N ($N \geq N_1$), over one period $\tilde{x}[n] = x[n]$ ($0 \leq n < N$).
 - According to the discrete-time Fourier series, the DFT of $x[n]$ is usually written as

$$\tilde{X}(k) = a_k = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-jk\frac{2\pi}{N}n}, \quad k = 0, 1, \dots, N-1$$

- The synthesis equation

$$x[n] = \sum_{k=0}^{N-1} \tilde{X}(k) e^{jk\frac{2\pi}{N}n}, \quad n = 0, 1, \dots, N-1$$

The DFT Analysis in Matrix-Vector Form

$$\tilde{X}(k) = a_k = \frac{1}{N} \sum_{n=0}^{N-1} x[n] e^{-jk\frac{2\pi}{N}n}, \quad k = 0, 1, \dots, N-1$$

$$\begin{pmatrix} \tilde{X}(0) \\ \vdots \\ \tilde{X}(N-1) \end{pmatrix} = \begin{pmatrix} W_{0,0} & \cdots & W_{0,N-1} \\ \vdots & \ddots & \vdots \\ W_{N-1,0} & \cdots & W_{N-1,N-1} \end{pmatrix} \begin{pmatrix} x[0] \\ \vdots \\ x[N-1] \end{pmatrix}$$

where $W_{k,n} = \frac{1}{N} e^{-jk\frac{2\pi}{N}n}$

DFT Basis

- DFT Matrix

$$W = \frac{1}{N} \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{-j2\pi/N} & \cdots & e^{-j2\pi(N-1)/N} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{-j2\pi(N-1)/N} & \cdots & e^{-j2\pi(N-1)(N-1)/N} \end{pmatrix}$$

The DFT Analysis in Matrix-Vector Form

$$\begin{pmatrix} \tilde{X}(0) \\ \vdots \\ \tilde{X}(N-1) \end{pmatrix} = \begin{pmatrix} (W_{0,0} & \cdots & W_{0,N-1}) & \begin{pmatrix} x[0] \\ \vdots \\ x[N-1] \end{pmatrix} \\ & \vdots & \\ (W_{N-1,0} & \cdots & W_{N-1,N-1}) & \begin{pmatrix} x[0] \\ \vdots \\ x[N-1] \end{pmatrix} \end{pmatrix}$$

Inverse DFT Basis

$$W^H = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & e^{j2\pi/N} & \cdots & e^{j2\pi(N-1)/N} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & e^{j2\pi(N-1)/N} & \cdots & e^{j2\pi(N-1)(N-1)/N} \end{pmatrix}$$

The DFT **Synthesize** in Matrix-Vector Form

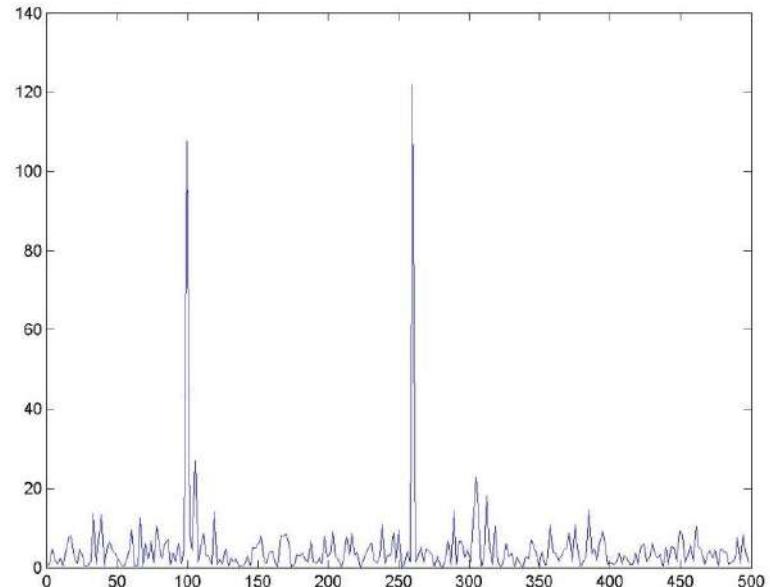
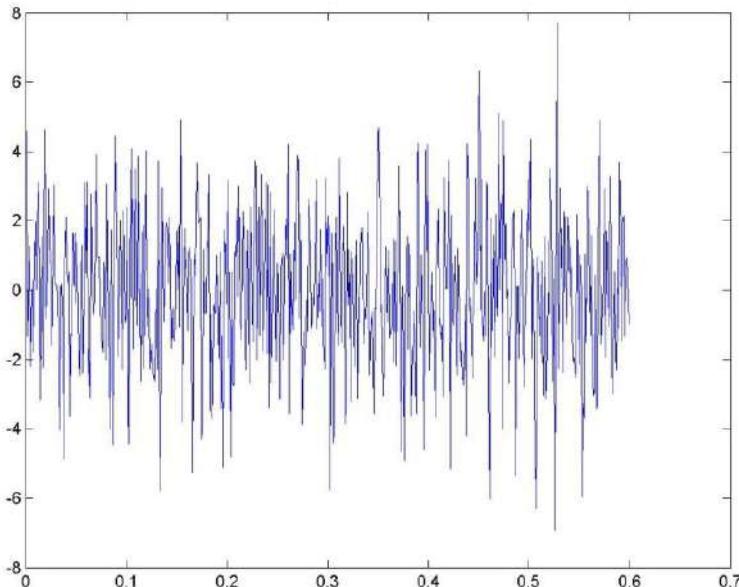
$$x[n] = \sum_{k=0}^{N-1} \tilde{X}(k) e^{jk\frac{2\pi}{N}n}, \quad n = 0, 1, \dots, N-1$$

$$\begin{pmatrix} x[0] \\ \vdots \\ x[N-1] \end{pmatrix} = \tilde{X}(0) \begin{pmatrix} W_{0,0}^* \\ \vdots \\ W_{0,N-1}^* \end{pmatrix} + \cdots + \tilde{X}(N-1) \begin{pmatrix} W_{N-1,0}^* \\ \vdots \\ W_{N-1,N-1}^* \end{pmatrix}$$

where

$$W_{k,n}^* = e^{jk\frac{2\pi}{N}n}$$

An Fourier Analysis Example



```
t = 0:0.001:0.6;      f = 1000*(0:255)/512;
```

```
x = cos (2*pi*100*t)+sin(2*pi*260*t);
```

```
y = x + 2*randn(size(t)); Y = fft(y, 512);
```

```
P = Y.*conj(Y)/512;
```

```
figure(1); plot(t, y); figure(2); plot(f, P(1:256));
```

References

- [1] A. V. Oppenheim, A. S. Willsky and I. T. Young, Signals and Systems, Prentice-Hall, 1983.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Fourier Transform (IV)

— 2D DFT

Dr. Xiqun Lu

*College of Computer Science
Zhejiang University*

Image Representation

PRIORI BASIS FOR NATURAL IMAGES

Discrete Fourier Transform

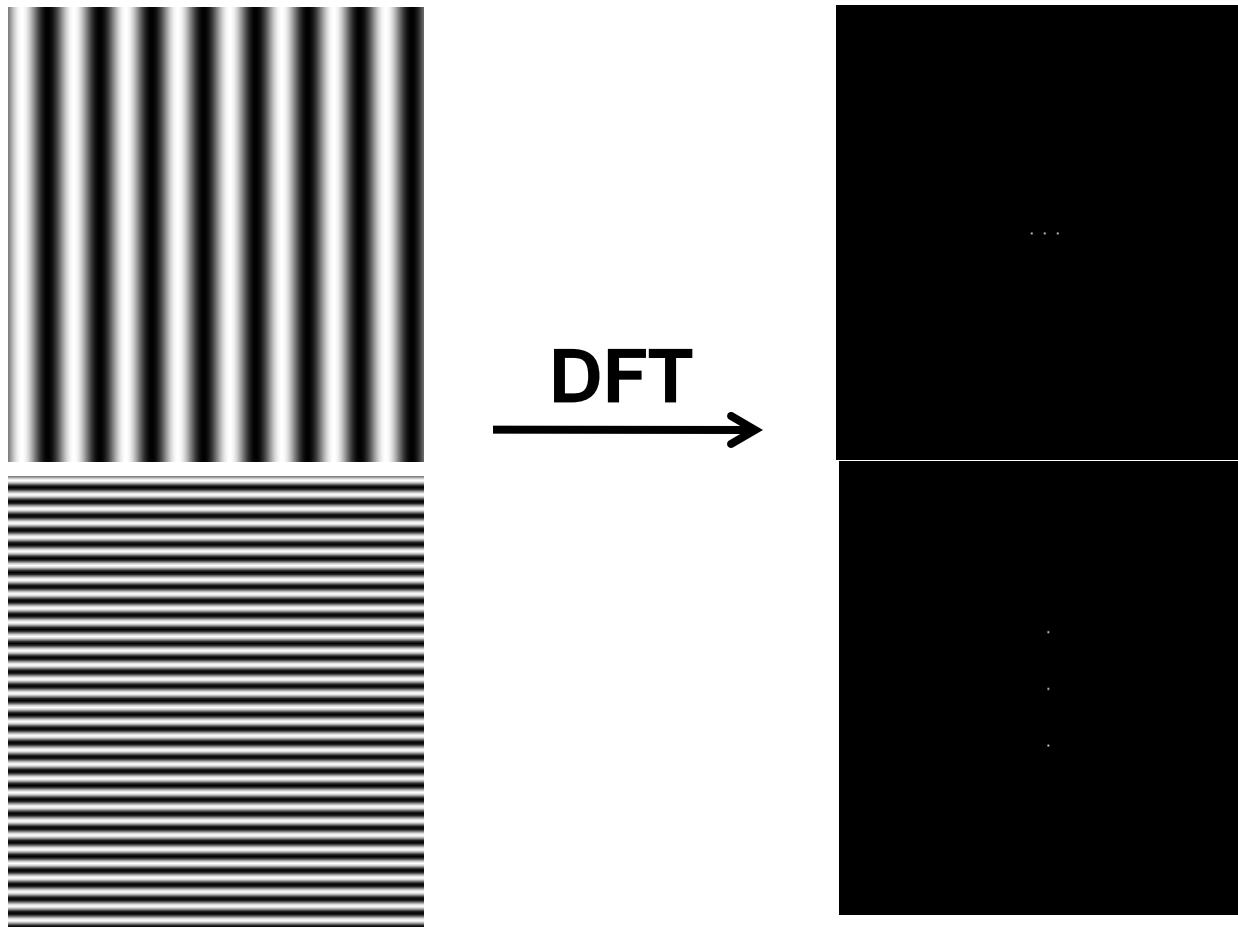
- Forward transform (Analysis)

$$X(k_1, k_2) = \sum_{n_1=0}^{N_1-1} \sum_{n_2=0}^{N_2-1} x(n_1, n_2) \exp\left(-jk_1 \frac{2\pi}{N_1} n_1 - jk_2 \frac{2\pi}{N_2} n_2\right)$$

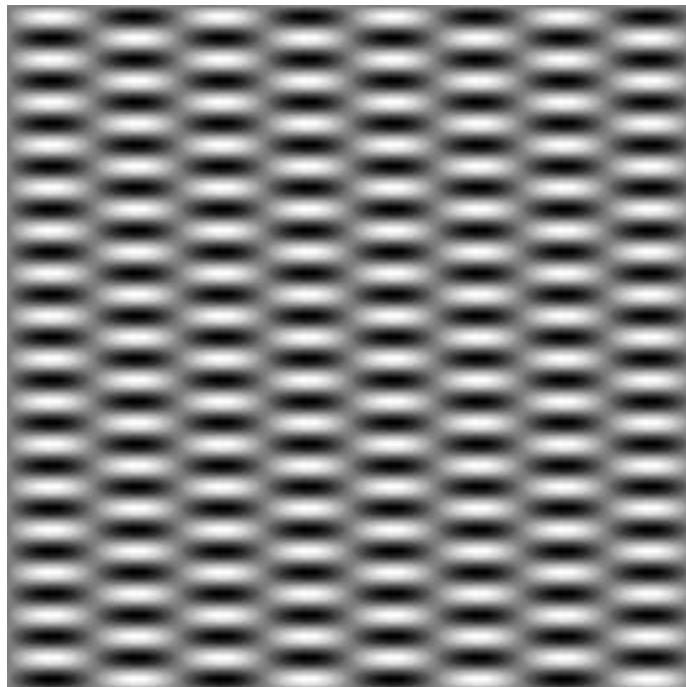
- Inverse transform (Synthesis)

$$x(n_1, n_2) = \frac{1}{N_1 N_2} \sum_{k_1=0}^{N_1-1} \sum_{k_2=0}^{N_2-1} X(k_1, k_2) \exp\left(jk_1 \frac{2\pi}{N_1} n_1 + jk_2 \frac{2\pi}{N_2} n_2\right)$$

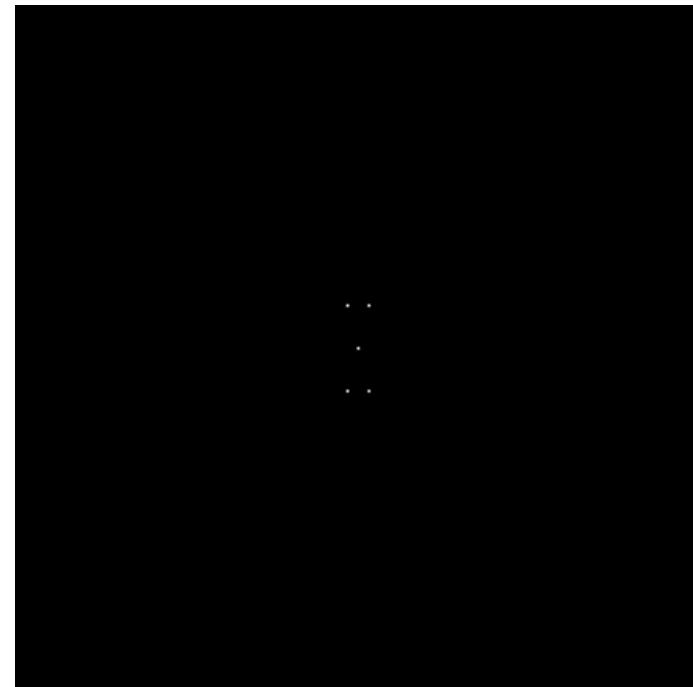
Synthesized 2D images and their magnitude spectrums



Synthesized 2D image and its magnitude spectrum (II)

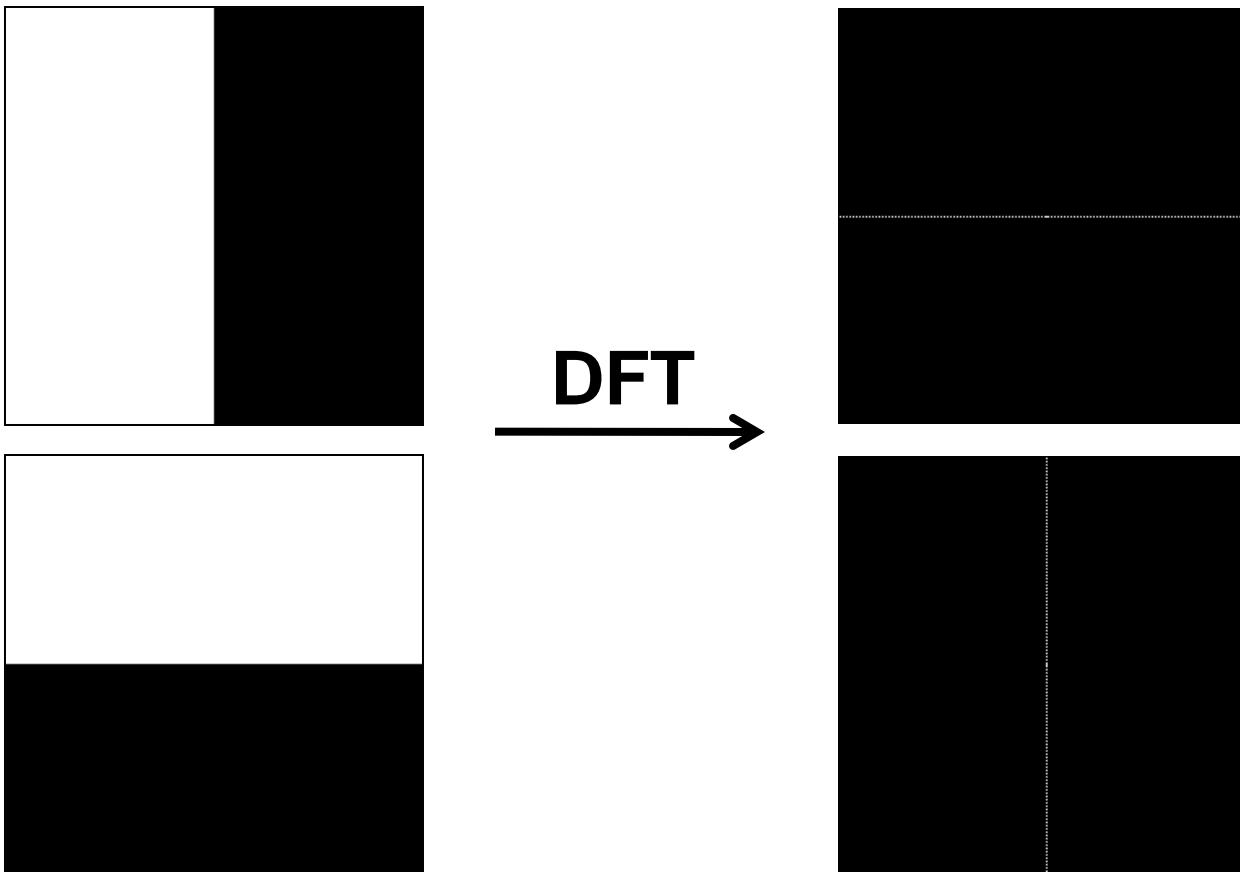


(a) This image exclusively has 4 cycles horizontally and 16 cycles vertically.

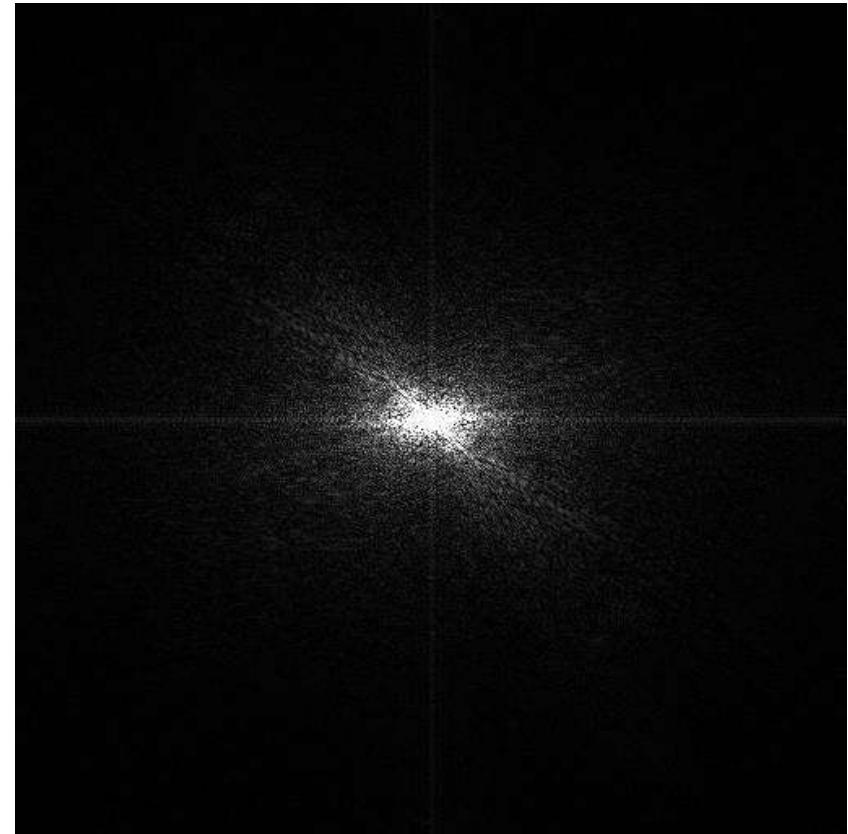


(b) Magnitude spectrum of the image

Two step images and their magnitude spectrums



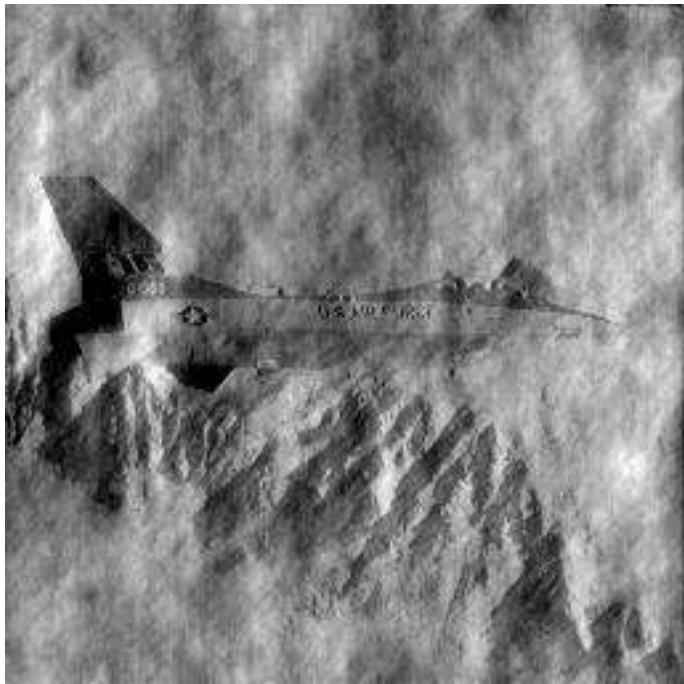
The Magnitude (DFT of Lena)



To Combine the **Magnitude** of One Image and the **Phase** of the Other Image



Resulting



Hint: **fft2**, **abs** and **angle** in the MatlabTM.

References

- [1] A. V. Oppenheim, A. S. Willsky and I. T. Young, Signals and Systems, Prentice-Hall, 1983.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Discrete Cosine Transform (DCT)

Dr. Xiqun Lu

College of Computer Science

Zhejiang University

Image Representation

PRIORI BASIS FOR NATURAL IMAGES

DCT

- A priori basis for natural images (not data-driven)
- DCT Basis Functions is defined as:

$$t_{k+1} = \left\{ \sqrt{\frac{2}{N}} a_k \cos \frac{(2n+1)k\pi}{2N} \right\}_{n=0,1,\dots,N-1}$$

where $k = 0, 1, \dots, N - 1$

$$a_k = \begin{cases} \frac{1}{\sqrt{2}} & k = 0 \\ 1 & k \neq 0 \end{cases}$$

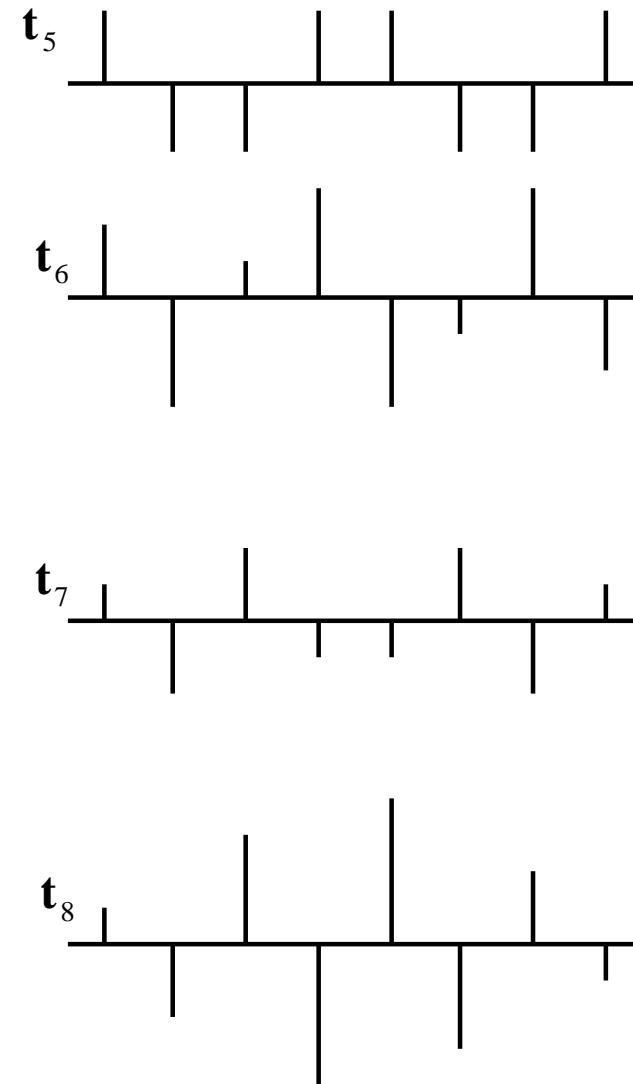
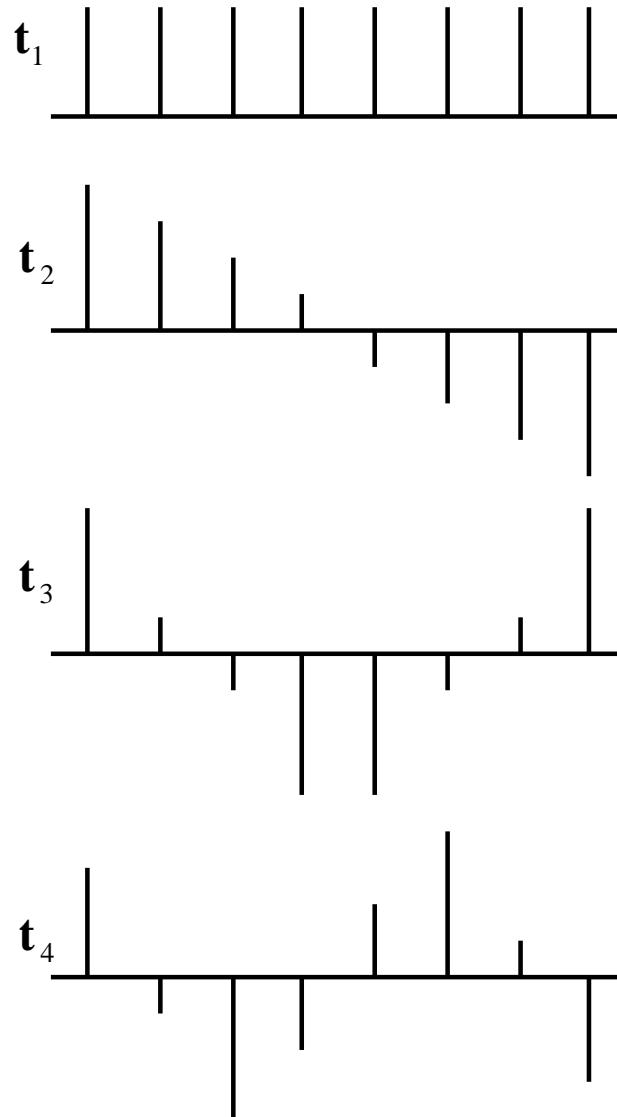
8-Point DCT Example

- Suppose we have a 8-point signal, that is $n = 0, 1, \dots, 7$, then the 8-point DCT transform matrix will be:

$$\mathbf{T}^t = \begin{bmatrix} 0.354 & 0.49 & 0.462 & 0.416 & 0.354 & 0.278 & 0.191 & 0.098 \\ 0.354 & 0.416 & 0.191 & -0.098 & -0.354 & -0.49 & -0.462 & -0.278 \\ 0.354 & 0.278 & -0.191 & -0.49 & -0.354 & 0.098 & 0.462 & 0.416 \\ 0.354 & 0.098 & -0.462 & -0.278 & 0.354 & 0.416 & -0.191 & -0.49 \\ 0.354 & -0.098 & -0.462 & 0.278 & 0.354 & -0.416 & -0.191 & 0.49 \\ 0.354 & -0.278 & -0.191 & 0.49 & -0.354 & -0.098 & 0.462 & -0.416 \\ 0.354 & -0.416 & 0.191 & 0.098 & -0.354 & 0.49 & -0.462 & 0.278 \\ 0.354 & -0.49 & 0.462 & -0.416 & 0.354 & -0.278 & 0.191 & -0.098 \end{bmatrix}$$

- Each column in \mathbf{T}^t is computed with a fixed “ k ”, for example the 1st column is computed by $k = 0$, the last column is computed by $k = 7$.
- You can consider each column of \mathbf{T}^t as a basic vector.

8 Basis Vectors of the 8-Point DCT



The DCT in Matrix-Vector Form

$$X(k) = \sum_{n=0}^{N-1} t_{k+1} x[n]$$

- Example: 8-Point ($N = 8$) DCT

$$\begin{pmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \end{pmatrix} = \begin{pmatrix} 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 & 0.354 \\ 0.49 & 0.416 & 0.278 & 0.098 & -0.098 & -0.278 & -0.416 & -0.49 \\ 0.462 & 0.191 & -0.191 & -0.462 & -0.462 & -0.191 & 0.191 & 0.462 \\ 0.416 & -0.098 & -0.49 & -0.278 & 0.278 & 0.49 & 0.098 & -0.416 \\ 0.354 & -0.354 & -0.354 & 0.354 & 0.354 & -0.354 & -0.354 & 0.354 \\ 0.278 & -0.49 & 0.098 & 0.416 & -0.416 & -0.098 & 0.49 & -0.278 \\ 0.191 & -0.462 & 0.462 & -0.191 & -0.191 & 0.462 & -0.462 & 0.191 \\ 0.098 & -0.278 & 0.416 & -0.49 & 0.49 & -0.416 & 0.278 & -0.098 \end{pmatrix} \begin{pmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{pmatrix}$$

2-D DCT Transform

- Forward DCT

$$F(u, v) = \frac{2}{N} C(u) C(v) \left[\sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos \frac{(2x+1)u\pi}{2N} \cos \frac{(2y+1)v\pi}{2N} \right]$$

- Backward DCT

$$f(x, y) = \frac{2}{N} \left[\sum_{u=0}^{N-1} \sum_{v=0}^{N-1} C(u) C(v) F(u, v) \cos \frac{(2x+1)u\pi}{2N} \cos \frac{(2y+1)v\pi}{2N} \right]$$

where $C(u), C(v) = \begin{cases} \frac{1}{\sqrt{2}} & u, v = 0 \\ 1 & otherwise \end{cases}$

$$T = \begin{pmatrix} 0.5000 & 0.6533 & 0.5000 & 0.2706 \\ 0.5000 & 0.2706 & -0.5000 & -0.6533 \\ 0.5000 & -0.2706 & -0.5000 & 0.6533 \\ 0.5000 & -0.6533 & 0.5000 & -0.2706 \end{pmatrix} \quad 4 \times 4 \text{ DCT Transform}$$

$$f_1 = \begin{pmatrix} 0.5000 \\ 0.5000 \\ 0.5000 \\ 0.5000 \end{pmatrix} \quad f_2 = \begin{pmatrix} 0.6533 \\ 0.2706 \\ -0.2706 \\ -0.6533 \end{pmatrix} \quad f_3 = \begin{pmatrix} 0.5000 \\ -0.5000 \\ -0.5000 \\ 0.5000 \end{pmatrix} \quad f_4 = \begin{pmatrix} 0.2706 \\ -0.6533 \\ 0.6533 \\ -0.2706 \end{pmatrix}$$

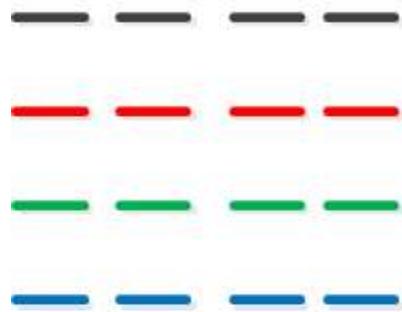
$$f_1 f_1^T = \begin{pmatrix} 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \\ 0.25 & 0.25 & 0.25 & 0.25 \end{pmatrix} \quad f_1 f_2^T = \begin{pmatrix} 0.3266 & 0.1383 & -0.1383 & -0.3266 \\ 0.3266 & 0.1383 & -0.1383 & -0.3266 \\ 0.3266 & 0.1383 & -0.1383 & -0.3266 \\ 0.3266 & 0.1383 & -0.1383 & -0.3266 \end{pmatrix}$$

$$f_1 f_3^T = \begin{pmatrix} 0.25 & -0.25 & -0.25 & 0.25 \\ 0.25 & -0.25 & -0.25 & 0.25 \\ 0.25 & -0.25 & -0.25 & 0.25 \\ 0.25 & -0.25 & -0.25 & 0.25 \end{pmatrix} \quad f_1 f_4^T = \begin{pmatrix} 0.1353 & -0.3266 & 0.3266 & -0.1353 \\ 0.1353 & -0.3266 & 0.3266 & -0.1353 \\ 0.1353 & -0.3266 & 0.3266 & -0.1353 \\ 0.1353 & -0.3266 & 0.3266 & -0.1353 \end{pmatrix}$$

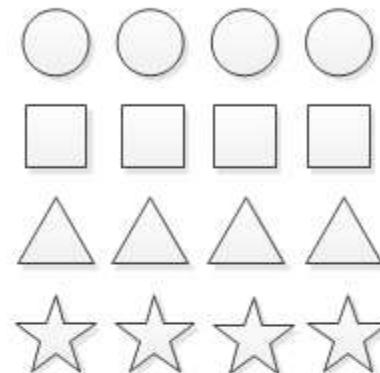
⋮

$$f_4 f_4^T = \begin{pmatrix} 0.0732 & -0.1768 & 0.1768 & -0.0732 \\ -0.1768 & 0.4268 & -0.4268 & 0.1768 \\ 0.1768 & -0.4268 & 0.4268 & -0.1768 \\ -0.0732 & 0.1768 & -0.1768 & 0.0732 \end{pmatrix}$$

An Example of 4-by-4 DCT

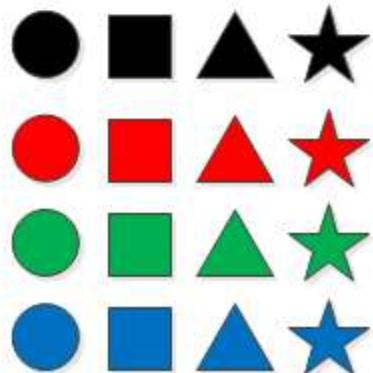


a 4-by-4 block

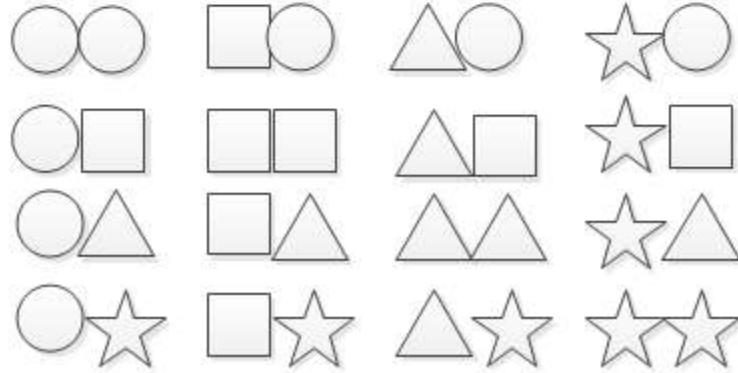


4 Filters

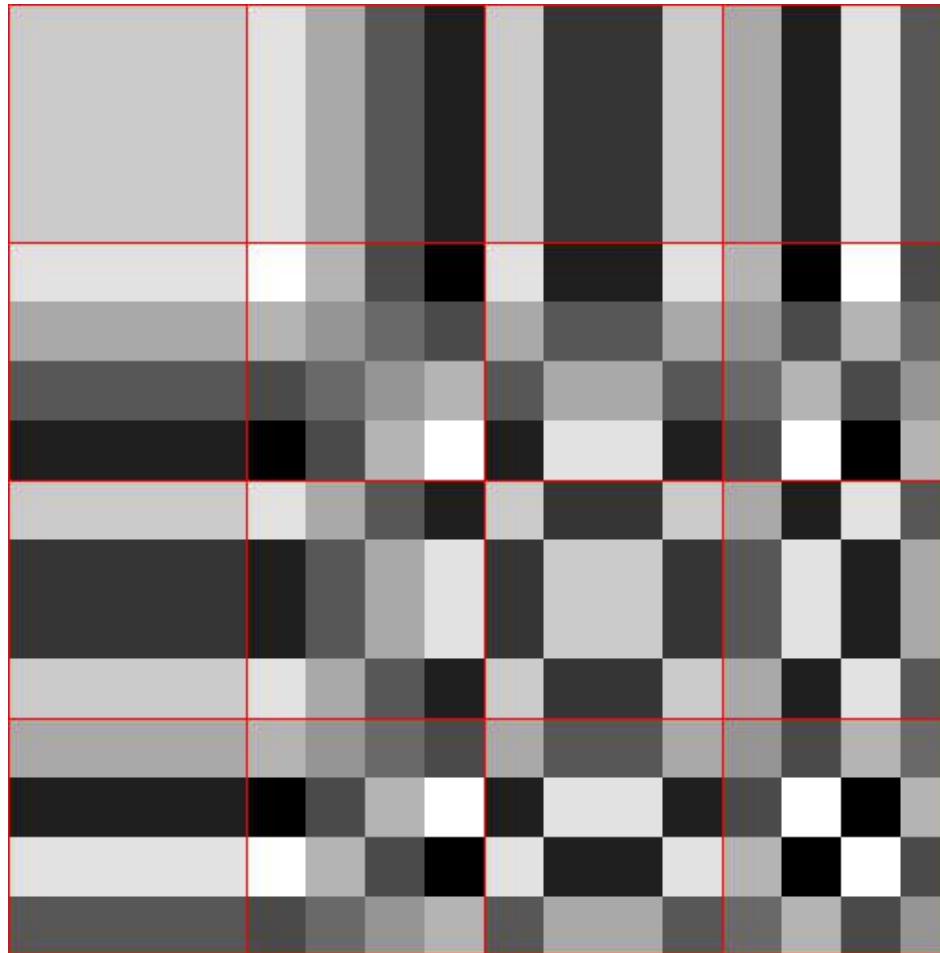
1) Row Filtering



2) Column Filtering



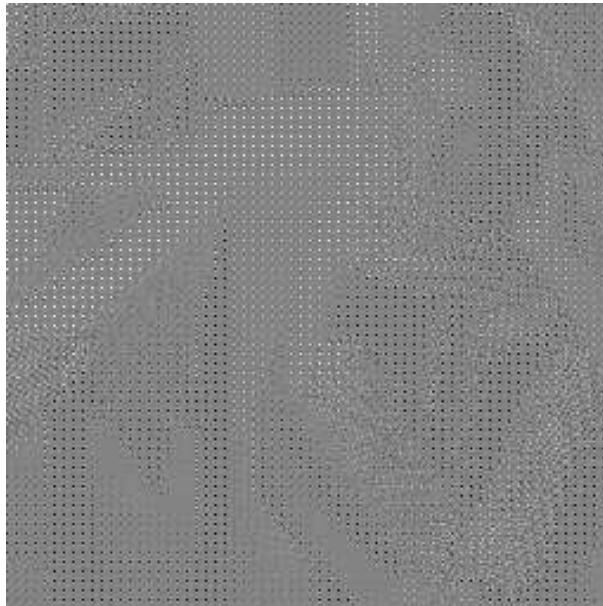
16 Basic Images of 4x4 DCT



Experimental Results



“Barbara”



DCT Transform of 4×4 blocks



256×256 DCT Transform
of Whole Image

Spatial and Frequency Localization Uncertainty Principle

Matlab Code

- `im = imread('barbara.bmp','bmp');`
- `[H, W, dim] = size(im);`
- `if dim ~= 1`
- `im = rgb2gray(im);`
- `end`
- `N = 4;`
- `im = double(im) - 127; % center the input image around zero`
- `im_blocks = im2col(im, [N N], 'distinct');`
- `num_blocks = size(im_blocks, 2);`
- `for i = 1:num_blocks`
- `DCT_coef(:,i) = dct(im_blocks(:,i), N*N);`
- `end`
-
- `im_DCT = col2im(DCT_coef, [N N], [H W], 'distinct');`
- `subbands = col2im(DCT_coef', [H/N W/N], [H W], 'distinct');`

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Principal Component Analysis (PCA)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Image Representation

POSTERIOR BASIS FOR NATURAL IMAGES

Basis for Natural Images

- **Priori Basis**
 - Fourier transforms
 - DCT
 - ...
- **Posterior Basis (data-driven)**
 - To estimate the linear transform from the data itself, and the transform could be ideally adapted to the kind of data that is being processed.
 - PCA
 - Dictionary based on machine learning

Two Properties of Image Transforms

- Variable decoupling
 - The coefficients of these bases are **less correlated** or become **independent** in ideal cases.
- Dimension reduction
 - The number of bases for approximately reconstructing an image is often much smaller than the number of pixels.

PCA [1]

- Either DFT or DCT has “energy compaction” property.
- What is the **optimal** transform in terms of energy compaction?
- Two basic concepts in statistical analysis: *variance* and *covariance* of random vectors

Variance and Covariance

- Consider two random vectors with **zero means** $\mathbf{a} = [a_1, a_2, \dots, a_n]^T$, $\mathbf{b} = [b_1, b_2, \dots, b_n]^T$.
- The **variance** of \mathbf{a} and \mathbf{b} are defined as

$$\sigma_{\mathbf{a}}^2 = \langle a_i - \bar{a}_i \rangle_i = \sum_{i=1}^n a_i^2$$

$$\sigma_{\mathbf{b}}^2 = \langle b_i - \bar{b}_i \rangle_i$$

- where the expectation $\langle \rangle_i$ is the average over n variables.
- The **covariance** between \mathbf{a} and \mathbf{b} is a straight-forward generalization:

$$\sigma_{\mathbf{ab}}^2 = \langle a_i - \bar{a}_i \rangle_i \langle b_i - \bar{b}_i \rangle_i$$

Sample Covariance Matrix

- We generalize two random vectors to an arbitrary number. Suppose we have m random vectors, and each vector has n variables.
- The $n \times m$ observation matrix \mathbf{X} is defined as

$$\mathbf{X} = [\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_m] = \begin{bmatrix} x_{11} & \cdots & x_{m1} \\ \vdots & \ddots & \vdots \\ x_{1n} & \cdots & x_{mn} \end{bmatrix}$$

- The **sample** covariance matrix is estimated

$$C_{\mathbf{X}} = \frac{1}{m-1} \mathbf{X} \mathbf{X}^T$$

Goal of PCA

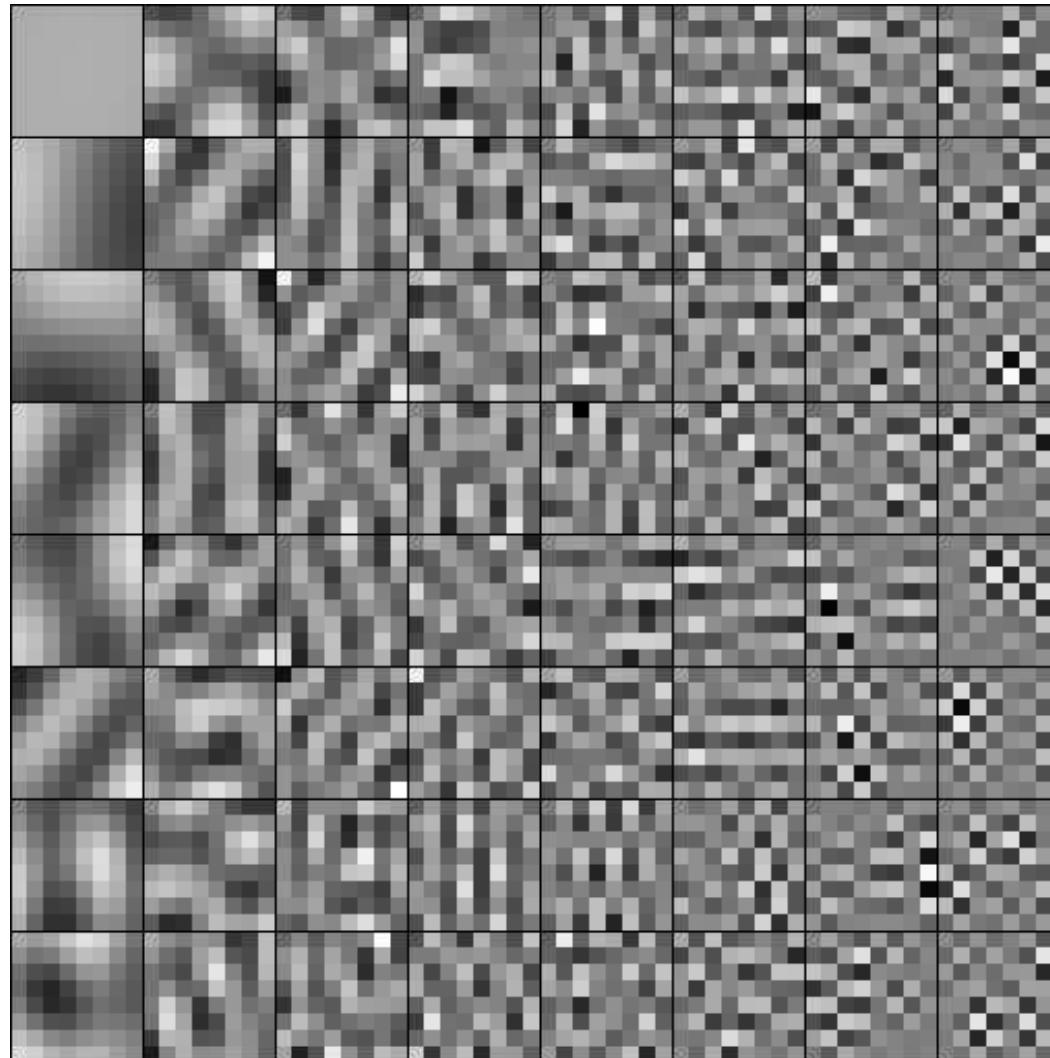
- The goal of PCA is to find *a set of basis* in which the observation matrix \mathbf{X} is transformed to a new observation matrix \mathbf{Y} , and *the covariance matrix* of \mathbf{Y} will be a **diagonal** matrix — to remove correlation among data whereas to preserve the energy as much as possible.
- Assumptions of PCA:
 - Firstly all basis vectors $\{\mathbf{p}_1, \dots, \mathbf{p}_n\}$ are **orthonormal** (i.e. $\mathbf{p}_i \bullet \mathbf{p}_j = \delta_{ij}$), that is all basis vectors construct an orthonormal matrix \mathbf{P} .
 - Secondly, *the directions with the largest variances* are more important than those with low variances, so in the transformed space the energy of signals is kept as compact as possible.

Eigen-Value Decomposition

- In the linear algebra, the covariance matrix C_x can be *diagonalized* by an orthogonal matrix of its eigen-vectors.
- Eigen-value decomposition: $C_x = EDE^T$
- If the orthonormal transform matrix P is selected as $P = E^T$,
 $\mathbf{Y} = \mathbf{P}\mathbf{X} = \mathbf{E}^T\mathbf{X}$

$$\begin{aligned}C_Y &= \frac{1}{m-1} \mathbf{Y} \mathbf{Y}^T \\&= \frac{1}{m-1} (\mathbf{E}^T \mathbf{X}) (\mathbf{E}^T \mathbf{X})^T \\&= \frac{1}{m-1} \mathbf{E}^T \mathbf{X} \mathbf{X}^T \mathbf{E} \\&= \mathbf{E}^T C_x \mathbf{E} \\&= \mathbf{E}^T E D E^T \mathbf{E} \\&= \mathbf{D}\end{aligned}$$

64 8×8 principal components (or basic images) learned from the image “Lena”



A toy example of PCA

1) Divide an image with the size of 512*512 into 8*8 blocks

2) Observation Matrix \mathbf{X} : 64 * 4096

3) The Sampling Covariance Matrix \mathbf{C}_X : 64*64

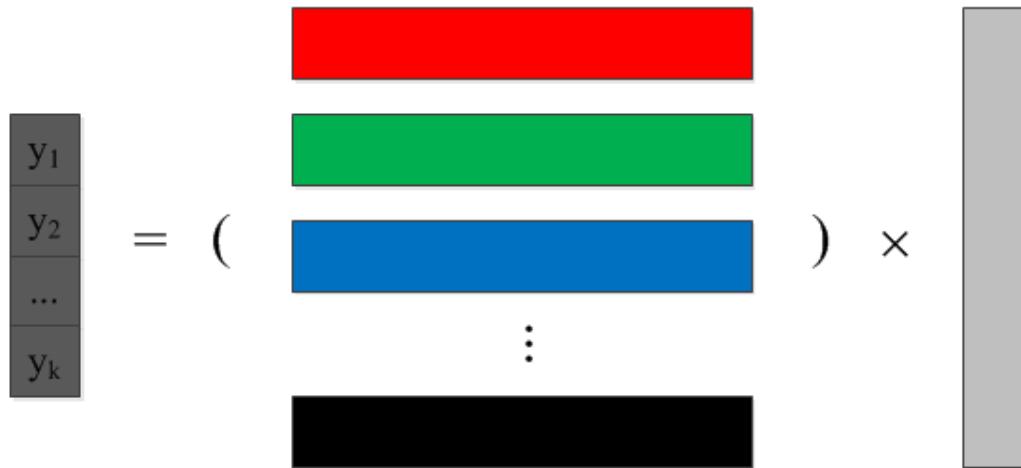
4) Eigenvalue Decomposition of \mathbf{C}_X : $\mathbf{C}_X = \mathbf{E}\mathbf{D}\mathbf{E}^T$

5) We select the eigenvectors with the first k largest eigenvalues, and form a PCA transform matrix \mathbf{P} (\mathbf{P} : 64 * k , $k << 64$)

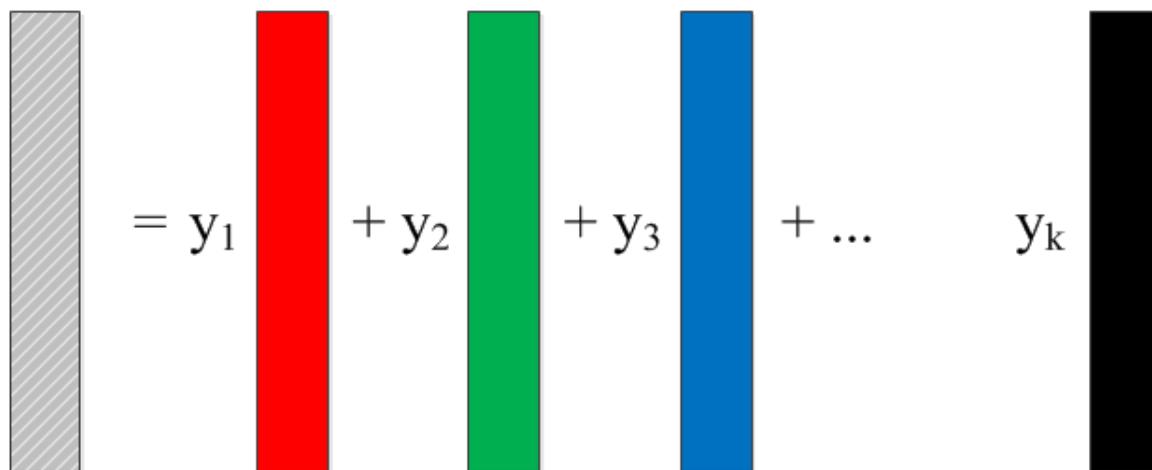
6) Now we can transform the observation matrix \mathbf{X} into another space by the PCA matrix \mathbf{P} ($\mathbf{Y} = \mathbf{P}^T \mathbf{X}$ and \mathbf{Y} : $k * 4096$)

7) Reconstruction: $\mathbf{X}_R = \mathbf{PY}$ (Note $\mathbf{X}_R \approx \mathbf{X}$)

PCA Forward Transform: $y = \mathbf{P}^T x$



PCA Inverse Transform: $x = \mathbf{P}y$



Reconstructed images with different number of PCs

20



Original Image



PSNR= 36.0315 dB

5



PSNR= 32.2466 dB

10



PSNR= 29.2603 dB

References

- [1] M. J. T. Smith and A. Docef, A Study Guide for Digital Image Processing, Scientific Publishers, Inc. Riverdale, Georgia, 1999.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Motion Estimation

Dr. Xigun Lu

College of Computer Science

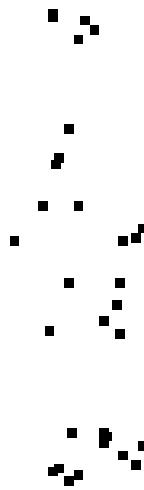
Zhejiang University

Motion Estimation

- Optical Flow in Computer Vision
 - Related topics: feature correspondence, image alignment, image registration
 - Main consideration: **Accuracy**
- Block Matching Algorithms (BMA) in Signal Processing
 - Related topics: video compression
 - Main consideration: **Efficiency**

We are living a moving world

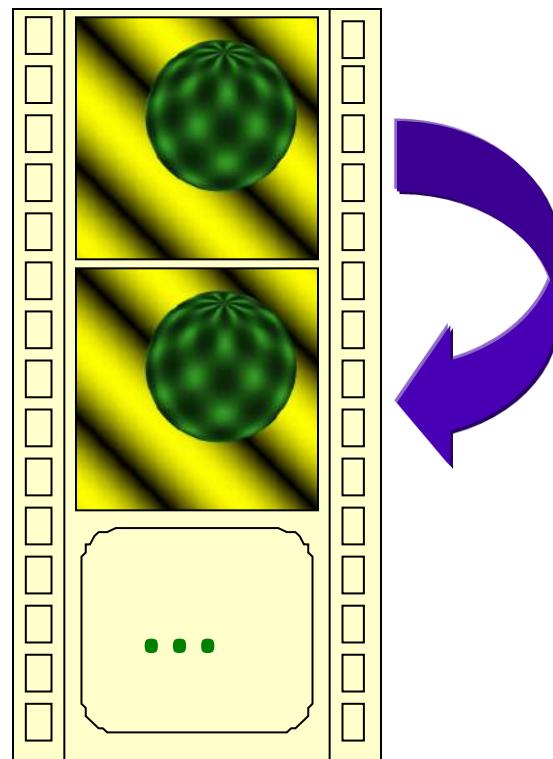
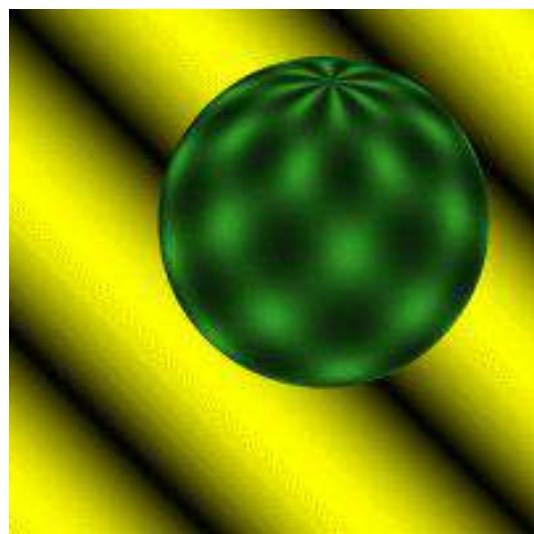
- Perceiving, understanding and predicting motion is an important part of our daily lives
- Even “impoverished” motion data can evoke a strong percept [1]



Outline

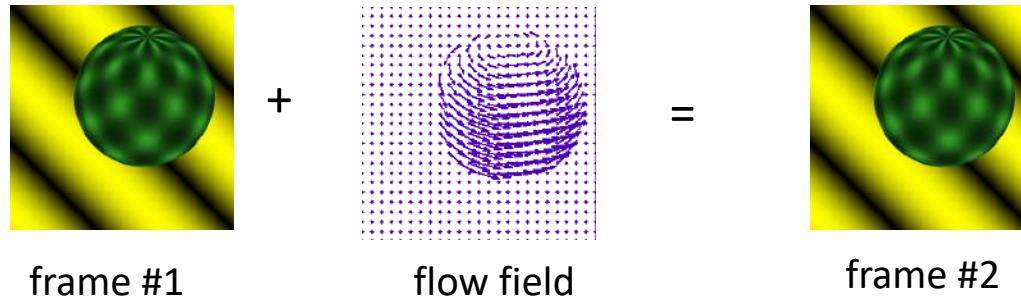
- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade (LK) algorithm [2]
 - Horn-Schunck (HS) algorithm [3]
- Block Matching Algorithm (BMA) (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

What is Optical Flow?



Definitions

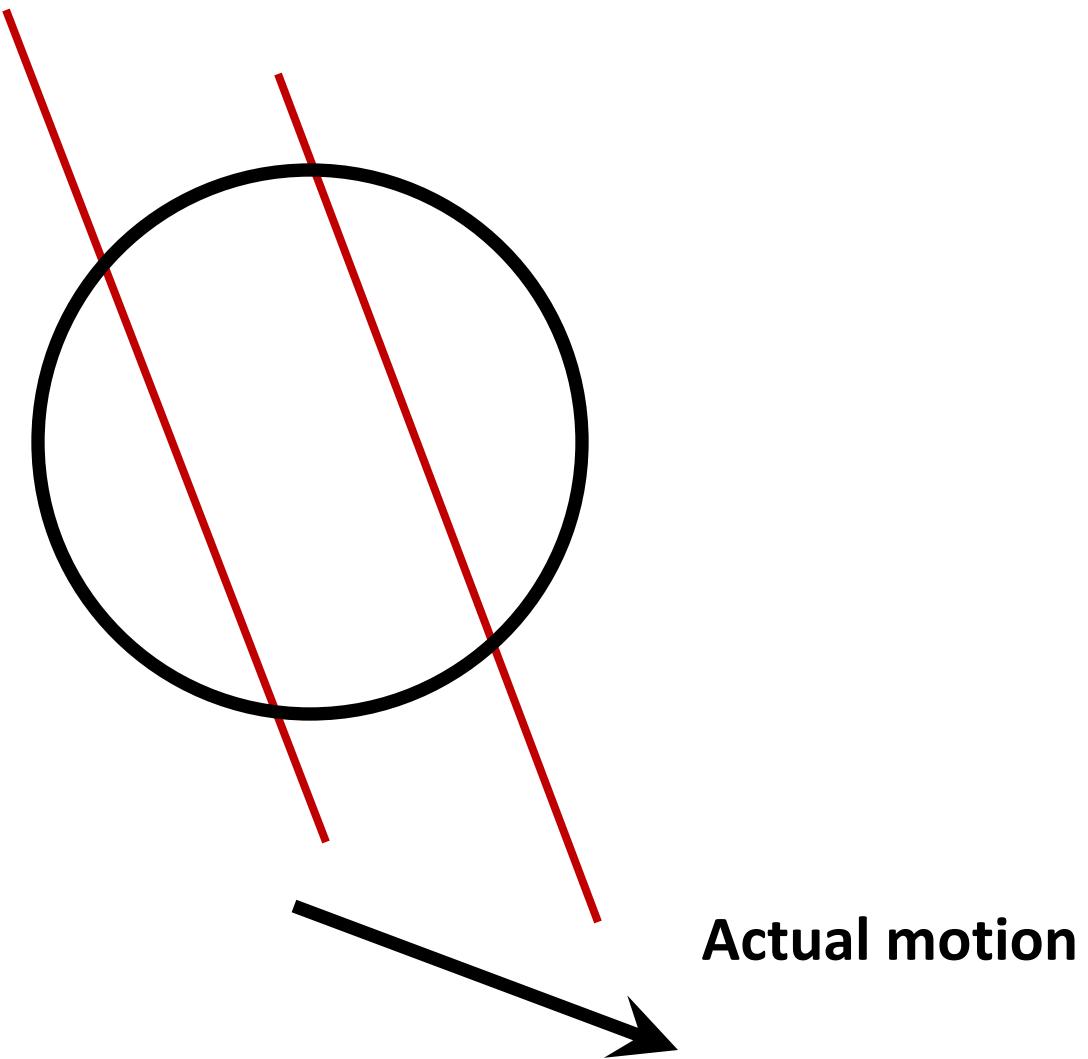
The **optical flow** is a velocity field in the image which transforms one image into the next image in a sequence [3]



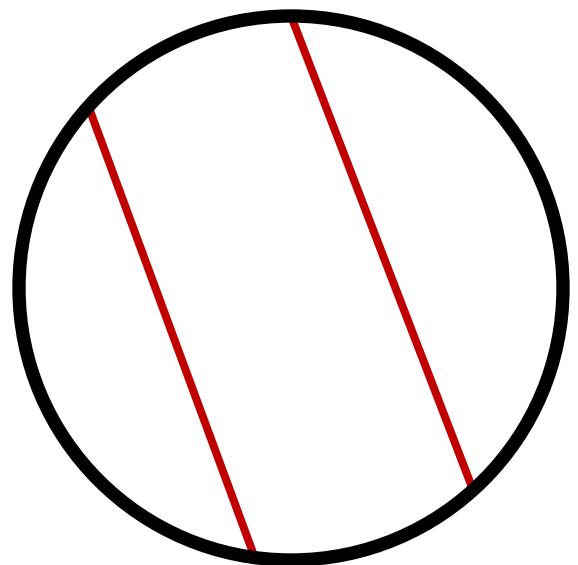
The **motion field** ... is the projection into the image of three-dimensional motion vectors [3]

The major limitation of optical flow is that it treats the 3D motion field as a 2D structure.

Aperture Problem



Aperture Problem



Perceived motion

References

- [1] G. Johansson, “Visual perception of biological motion and a model for its analysis”, *Perception and Psychophysics*, vol.14, 201-211, 1973.
- [2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. of *International Joint Conf. On Artificial Intelligence*, pp.674-679, 1981.
- [3] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, 17:185-203, 1981.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *Proceedings of national Telecommunications conference*, New Orleans, LA, pp.G5.3.1–G5.3.5, Dec. 1981.
- [5] R. Li, B. Zeng, and M. L. Liou, “A new three-step algorithm for block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 4(4): 438-442, 1994.
- [6] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 6(3): 313-317, 1996.
- [7] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. On Image Processing*, 9(2): 287-290, 2000.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Motion Estimation

— Optical Flow (I)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

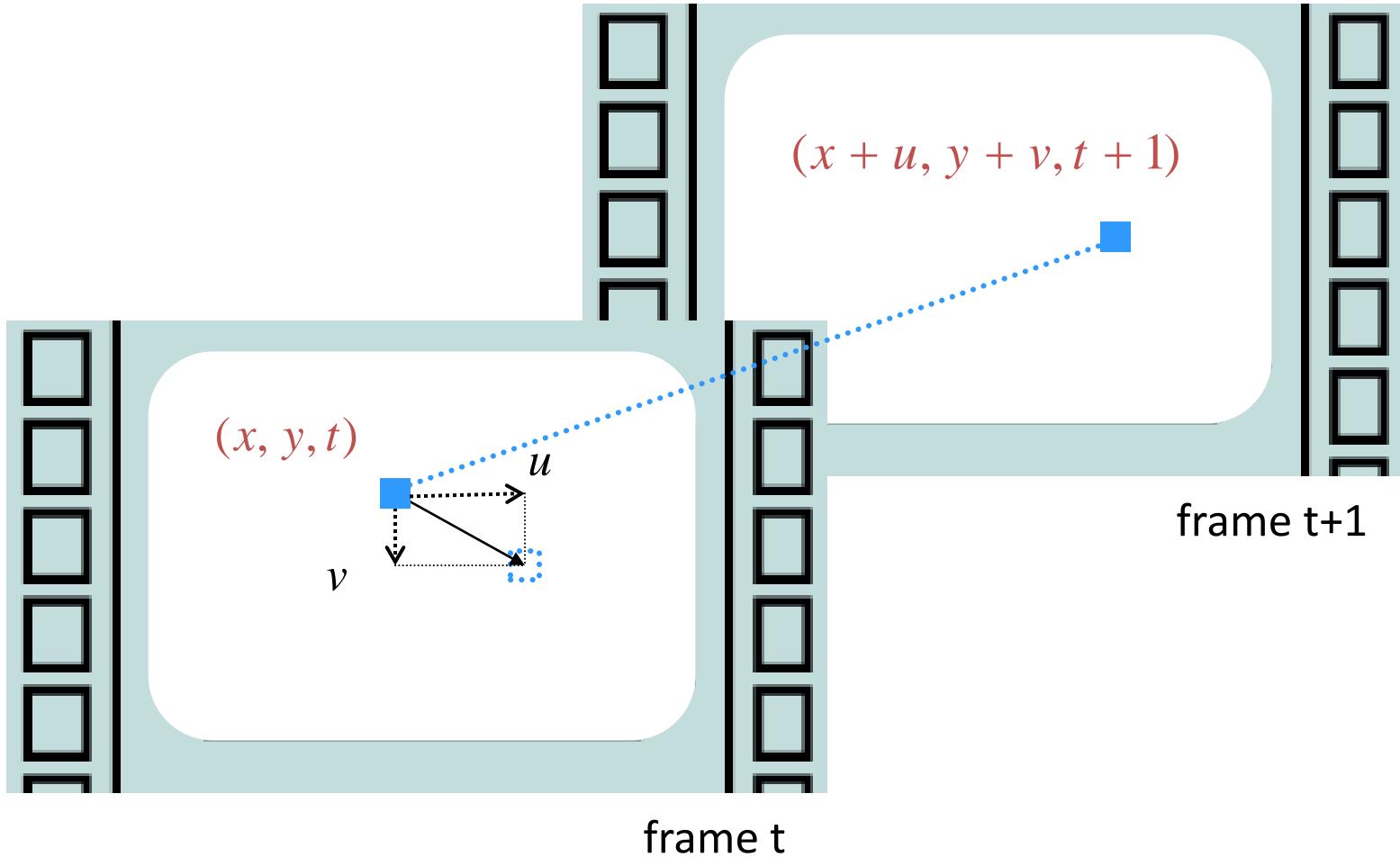
Key Assumptions of Lucas-Kanade [2]

- **Brightness constancy:** projection of the same point looks the same in every frame
- **Small motion:** points do not move very fast
- **Spatial coherence:** points move like their neighbors

[2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” In *Proceedings of the International Joint Conference on Artificial Intelligence*, pp. 674–679, 1981.

Brightness Constancy

$$I(x + u, y + v, t + 1) = I(x, y, t)$$



Brightness Constant

- The color or intensity values of image objects in subsequent frames do no change over time.

$$I(x + \delta_x, y + \delta_y, t + \delta_t) = I(x, y, t)$$

- Assume that δ_t is enough small, the above eq. can be linearized by a first order Taylor series expansion (we omit the high-order terms here)

$$\begin{aligned} I(x + \delta_x, y + \delta_y, t + \delta_t) &= I(x, y, t) + \frac{\partial I(x, y, t)}{\partial x} \delta_x + \frac{\partial I(x, y, t)}{\partial y} \delta_y + \frac{\partial I(x, y, t)}{\partial t} \delta_t \\ \frac{\partial I(x, y, t)}{\partial x} \delta_x + \frac{\partial I(x, y, t)}{\partial y} \delta_y + \frac{\partial I(x, y, t)}{\partial t} \delta_t &= 0 \end{aligned}$$

- The **optical flow constraint** (for gray images) is

$$I_x u + I_y v + I_t = 0$$

where $I_x = \frac{\partial I(x, y, t)}{\partial x}, I_y = \frac{\partial I(x, y, t)}{\partial y}, I_t = \frac{\partial I(x, y, t)}{\partial t}, u = \frac{\delta_x}{\delta_t}, v = \frac{\delta_y}{\delta_t}$

How to get more equations for a pixel?

- Spatial Coherence Constraint
 - Assume the pixel's neighbors have the same (u, v)
 - For example, if we use a 5×5 window, that gives us 25 equations (each pixel in the window has one equation)

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix}$$

Lucas-Kanade Algorithm (LK)^[1]

- Overdetermined linear system

$$\begin{bmatrix} I_x(p_1) & I_y(p_1) \\ I_x(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_{25}) & I_y(p_{25}) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} I_t(p_1) \\ I_t(p_2) \\ \vdots \\ I_t(p_{25}) \end{bmatrix}$$

$$\mathbf{A}\mathbf{d} = \mathbf{b}$$

(25×2) (2×1) (25×1)

- Least-squares solution of \mathbf{d} —
$$\min_{\mathbf{d}} \|\mathbf{A}\mathbf{d} - \mathbf{b}\|_2^2$$

Lucas-Kanade Algorithm (LK)^[1]

- Least-square solution of \mathbf{d}

$$\min_{\mathbf{d}} \|\mathbf{A}\mathbf{d} - \mathbf{b}\|_2^2 \rightarrow (\mathbf{A}\mathbf{d} - \mathbf{b})^T (\mathbf{A}\mathbf{d} - \mathbf{b})$$

$$(\mathbf{A}^T \mathbf{A})\mathbf{d} = \mathbf{A}^T \mathbf{b}$$

$$\begin{pmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{pmatrix}_{\mathbf{A}^T \mathbf{A}} \begin{pmatrix} u \\ v \end{pmatrix}_{\mathbf{d}} = - \begin{pmatrix} \sum I_x I_t \\ \sum I_y I_t \end{pmatrix}_{\mathbf{A}^T \mathbf{b}}$$

The summations are over all pixels in the local window.

Conditions for solvability

$$\left(\begin{array}{cc} \sum I_x I_x & \sum I_x I_y \\ \sum I_y I_x & \sum I_y I_y \end{array} \right) \begin{pmatrix} u \\ v \end{pmatrix} = - \left(\begin{array}{c} \sum I_x I_t \\ \sum I_y I_t \end{array} \right)$$

$\mathbf{A}^T \mathbf{A}$ \mathbf{d} $\mathbf{A}^T \mathbf{b}$

- When is this solvable?
 - $\mathbf{A}^T \mathbf{A}$ should be invertible
 - $\mathbf{A}^T \mathbf{A}$ should not be too small due to noise
 - eigenvalues λ_1 and λ_2 of $\mathbf{A}^T \mathbf{A}$ should not be too small
 - $\mathbf{A}^T \mathbf{A}$ should be well-conditioned
 - λ_1 / λ_2 should not be too large (λ_1 = the larger eigenvalue)

Flat region



$$\mathbf{A}^T \mathbf{A}$$

- gradients have small magnitude
- small λ_1 , small λ_2

Edge



$$\mathbf{A}^T \mathbf{A}$$

- gradients very large or very small
- large λ_1 , small λ_2

High-texture region



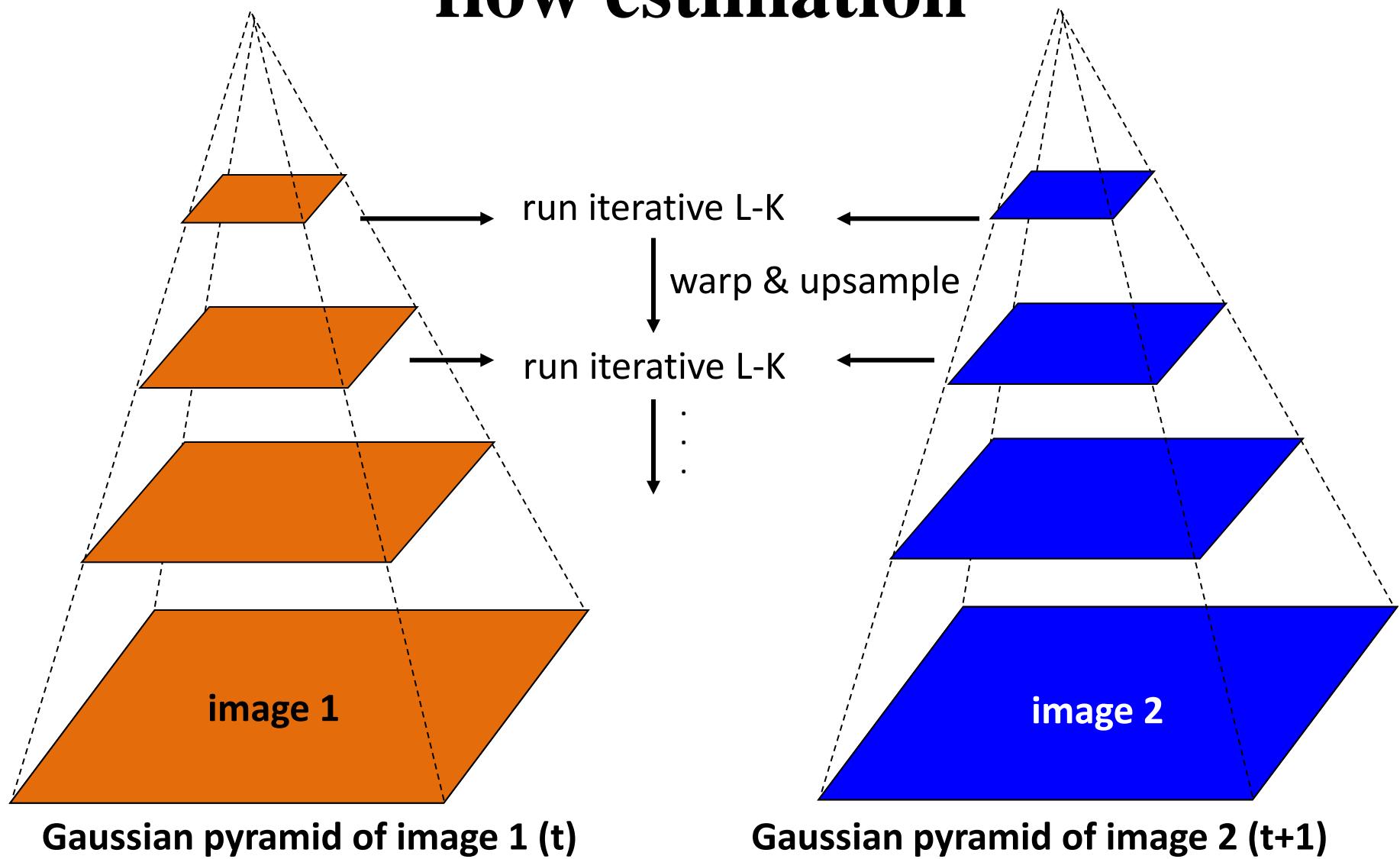
$$\mathbf{A}^T \mathbf{A}$$

- gradients are different, large magnitudes
- large λ_1 , large λ_2

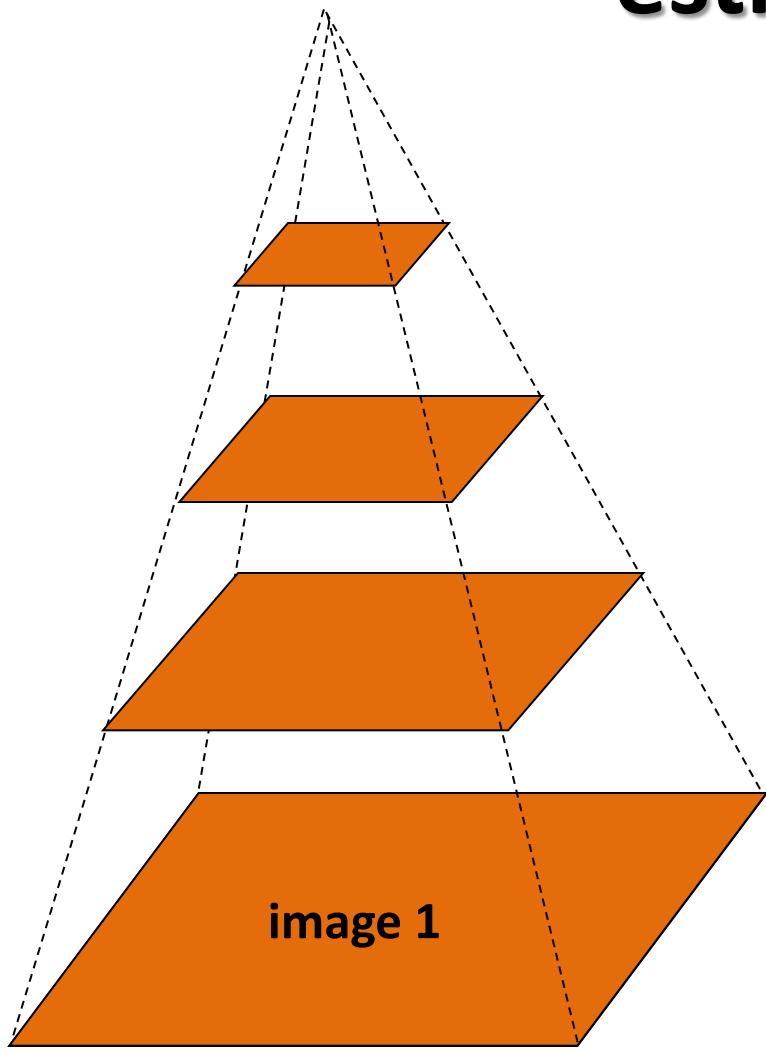
Errors in Lucas-Kanade Algorithm

- When the assumptions are violated
 - Brightness constancy is **not** satisfied
 - Gradient Constancy
 - The motion is **not** small
 - To estimate optical flow in a coarse-to-fine hierarchical way
 - A point does **not** move like its neighbors
 - What's the ideal size of local analysis window?

Coarse-to-fine Hierarchical optical flow estimation



Coarse-to-fine optical flow estimation



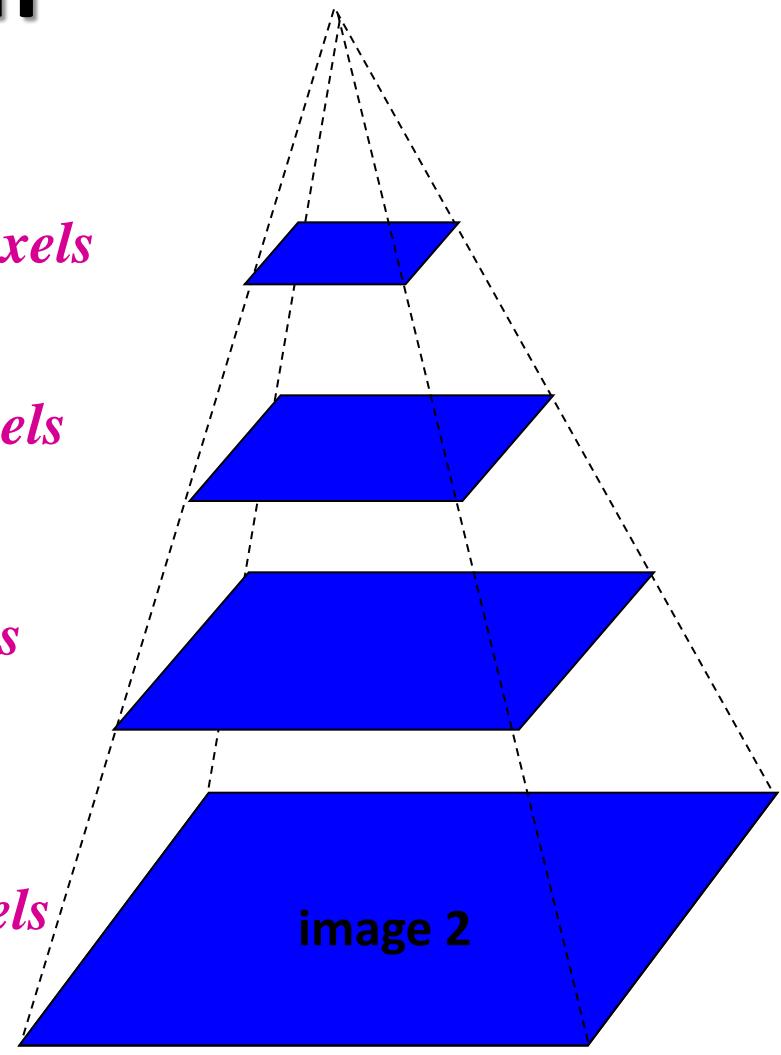
Gaussian pyramid of image 1

$u=1.25 \text{ pixels}$

$u=2.5 \text{ pixels}$

$u=5 \text{ pixels}$

$u=10 \text{ pixels}$



Gaussian pyramid of image 2

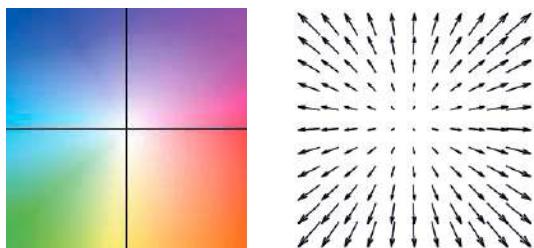
Example



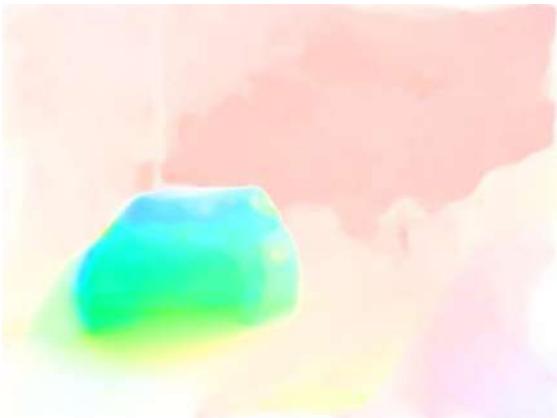
Input two frames



Coarse-to-fine LK



Flow visualization



Coarse-to-fine LK with median filtering



References

- [1] G. Johansson, “Visual perception of biological motion and a model for its analysis”, *Perception and Psychophysics*, vol.14, 201-211, 1973.
- [2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. of *International Joint Conf. On Artificial Intelligence*, pp.674-679, 1981.
- [3] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, 17:185-203, 1981.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *Proceedings of national Telecommunications conference*, New Orleans, LA, pp.G5.3.1–G5.3.5, Dec. 1981.
- [5] R. Li, B. Zeng, and M. L. Liou, “A new three-step algorithm for block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 4(4): 438-442, 1994.
- [6] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 6(3): 313-317, 1996.
- [7] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. On Image Processing*, 9(2): 287-290, 2000.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Motion Estimation — Optical Flow (II)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Horn-Schunck Algorithm (HS) [3]

- The basic assumption of HS algorithm — piece-wise smooth flow field
 - But flow field has *discontinuities* at the boundaries of objects in the scene
- Embed the optical flow constraint into a *regularization* framework

$$E_{HS}(u, v) = \int_{\Omega} \{(I_x u + I_y v + I_t)^2 + \alpha(|\nabla u|^2 + |\nabla v|^2)\} dx dy$$

↑
Brightness Constancy

↑
Spatial Coherency

$$|\nabla u|^2 = \left(\frac{\partial u}{\partial x} \right)^2 + \left(\frac{\partial u}{\partial y} \right)^2 = u_x^2 + u_y^2, \quad |\nabla v|^2 = \left(\frac{\partial v}{\partial x} \right)^2 + \left(\frac{\partial v}{\partial y} \right)^2 = v_x^2 + v_y^2$$

2D Euler Lagrange

- 2D Euler Lagrange: the functional

$$S = \iint_{\Omega} L(x, y, f, f_x, f_y) dx dy$$

is minimized only if f satisfies the partial differential equation

$$\frac{\partial L}{\partial f} - \frac{\partial}{\partial x} \frac{\partial L}{\partial f_x} - \frac{\partial}{\partial y} \frac{\partial L}{\partial f_y} = 0$$

$$L(u, v, u_x, u_y, v_x, v_y) = (I_x u + I_y v + I_t)^2 + \alpha(u_x^2 + u_y^2 + v_x^2 + v_y^2)$$

- In Horn-Schunck

$$\frac{\partial L}{\partial u} = 2(I_x u + I_y v + I_t) I_x$$

$$\frac{\partial L}{\partial u_x} = 2\alpha u_x$$

$$\frac{\partial}{\partial x} \frac{\partial L}{\partial u_x} = 2\alpha u_{xx}$$

$$\frac{\partial L}{\partial u_y} = 2\alpha u_y$$

$$\frac{\partial}{\partial y} \frac{\partial L}{\partial u_y} = 2\alpha u_{yy}$$

Linear PDE

- The Euler-Lagrange PDE for Horn-Schunck is

$$(I_x u + I_y v + I_t) I_x - \alpha(u_{xx} + u_{yy}) = (I_x u + I_y v + I_t) I_x - \alpha \Delta u = 0$$

$$(I_x u + I_y v + I_t) I_y - \alpha(v_{xx} + v_{yy}) = (I_x u + I_y v + I_t) I_y - \alpha \Delta v = 0$$

- u_{xx} and u_{yy} can be obtained by a Laplacian operator:

$$\Lambda = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

- In the end, we solve a large linear equation

$$\begin{pmatrix} I_x^2 + \alpha \Lambda & I_x I_y \\ I_y I_x & I_y^2 + \alpha \Lambda \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = - \begin{pmatrix} I_x I_t \\ I_y I_t \end{pmatrix}$$

How to solve a large linear system $\mathbf{Ax}=\mathbf{b}$?

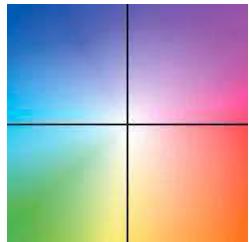
$$\begin{pmatrix} I_x^2 + \alpha\Lambda & I_x I_y \\ I_y I_x & I_y^2 + \alpha\Lambda \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = - \begin{pmatrix} I_x I_t \\ I_y I_t \end{pmatrix}$$

- With $\alpha > 0$, this system is positive definite!
- You can solve with your favorite iterative solver
 - Gauss-Seidel, successive over-relaxation (SOR)

Example



Input two frames



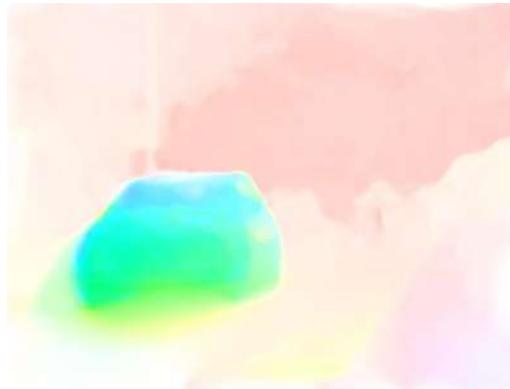
Flow visualization



Horn-Schunck



Coarse-to-fine LK



Coarse-to-fine LK with median filtering

References

- [1] G. Johansson, “Visual perception of biological motion and a model for its analysis”, *Perception and Psychophysics*, vol.14, 201-211, 1973.
- [2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. of *International Joint Conf. On Artificial Intelligence*, pp.674-679, 1981.
- [3] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, 17:185-203, 1981.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *Proceedings of national Telecommunications conference*, New Orleans, LA, pp.G5.3.1–G5.3.5, Dec. 1981.
- [5] R. Li, B. Zeng, and M. L. Liou, “A new three-step algorithm for block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 4(4): 438-442, 1994.
- [6] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 6(3): 313-317, 1996.
- [7] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. On Image Processing*, 9(2): 287-290, 2000.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Motion Estimation — BMA

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

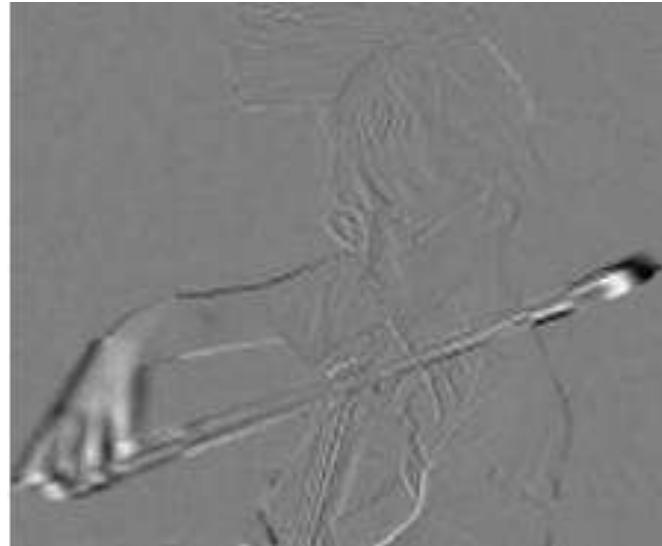
Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Correlation in Video Frames

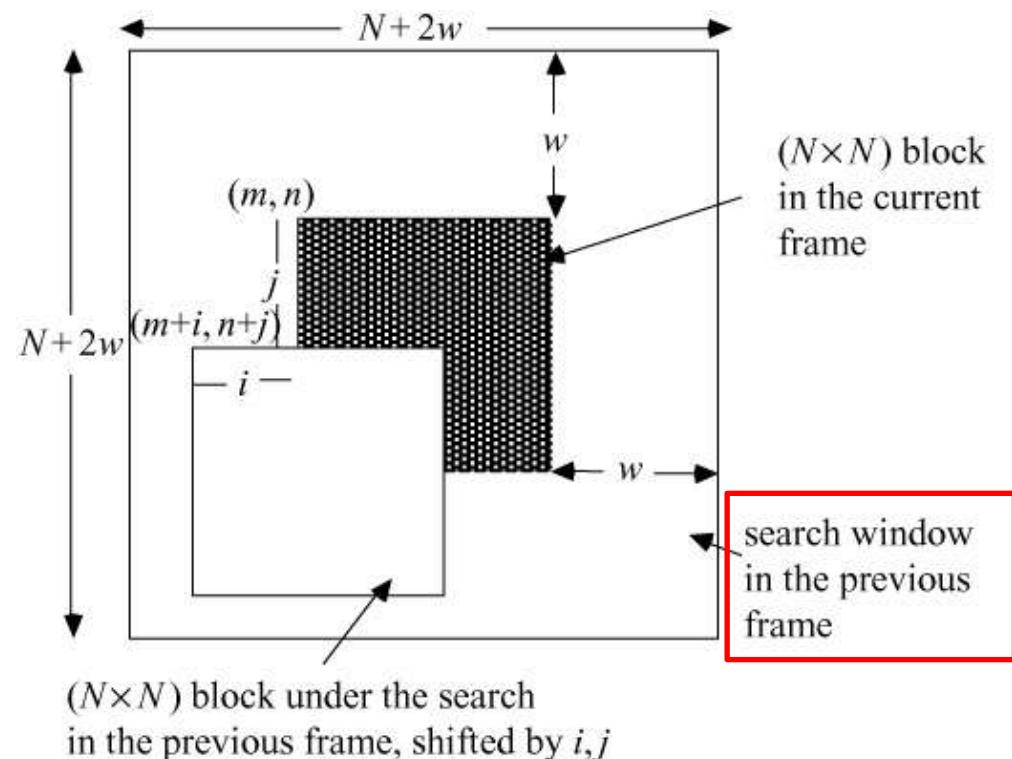


To remove the
**temporal
redundancy**
between successive
frames.



The Principle of BMA

- For each image-block, find the best matching block in the previously **encoded** frame.
 - Motion vector
 - The DCT transform coefficients of the **residual** signal will be transmitted.



Block-matching: Error Measure

- Mean Square Error (MSE)

$$MSE = \frac{1}{I*J} \sum_{i=1}^I \sum_{j=1}^J [f(i, j) - g(i-dx, j-dy)]^2$$

- Mean of the Absolute frame Difference (MAD)

$$MAD = \frac{1}{I*J} \sum_{i=1}^I \sum_{j=1}^J |f(i, j) - g(i-dx, j-dy)|$$

- Sum of Absolute Difference (SAD)

$$SAD = \sum_{i=1}^I \sum_{j=1}^J |f(i, j) - g(i-dx, j-dy)|$$

Generally, $I = J = 16$

Full Search Method

- Search every point in the searching window
 - Matching: $(2w+1)^2$
 - For each matching, the error computation: $N \times N$
 - If $w=16$ (the default search range in MPEG-1/2/4), the total number of search points will be 1089.

References

- [1] G. Johansson, “Visual perception of biological motion and a model for its analysis”, *Perception and Psychophysics*, vol.14, 201-211, 1973.
- [2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. of *International Joint Conf. On Artificial Intelligence*, pp.674-679, 1981.
- [3] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, 17:185-203, 1981.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *Proceedings of national Telecommunications conference*, New Orleans, LA, pp.G5.3.1–G5.3.5, Dec. 1981.
- [5] R. Li, B. Zeng, and M. L. Liou, “A new three-step algorithm for block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 4(4): 438-442, 1994.
- [6] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 6(3): 313-317, 1996.
- [7] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. On Image Processing*, 9(2): 287-290, 2000.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Motion Estimation — BMA

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Outline

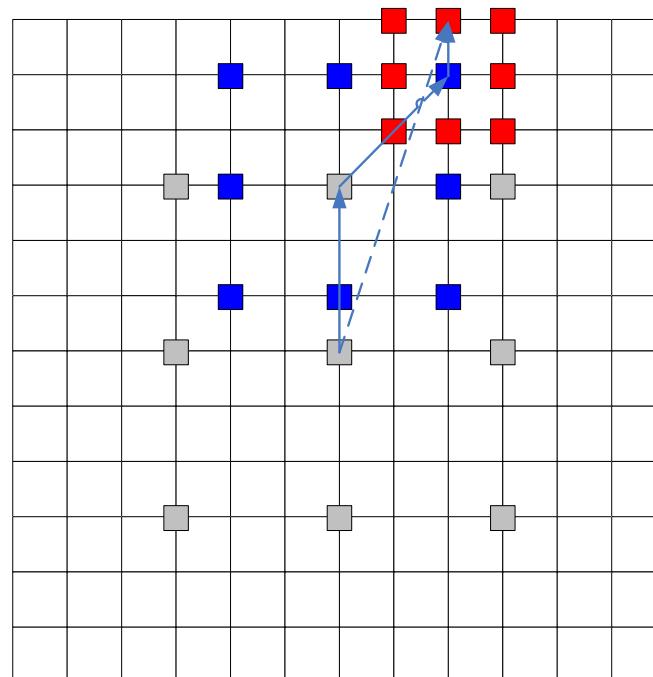
- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Three Step Search: TSS [4]

- For $w=7$, the matching points of TSS are 25, however the FS needs 225 comparisons.
- However, TSS uses a *uniformly* allocated search pattern in its first step, which is not very efficient to catch *small* motions appearing in stationary or quasi-stationary blocks.

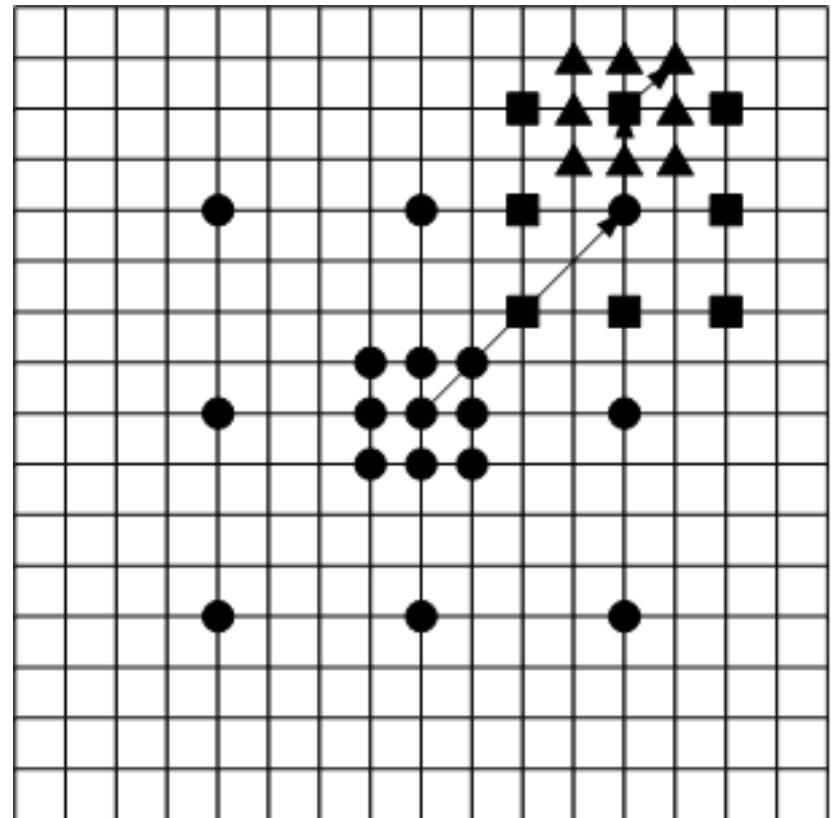


New TSS: NTSS [5]

- The TSS uses **a uniformly** allocated checking point pattern in its first step, which becomes *inefficient* for the estimation of **small** motions.
- The block motion field of a real world sequence is usually *gentle*, *smooth*, and *varies slowly*.
 - The global minimum motion distribution is **center-biased**, rather *uniformly* distributed.
- NTSS differs from TSS by
 - Assuming a **center-biased** checking point pattern
 - Incorporating a **halfway-stop** technique for stationary and quasi-stationary blocks.

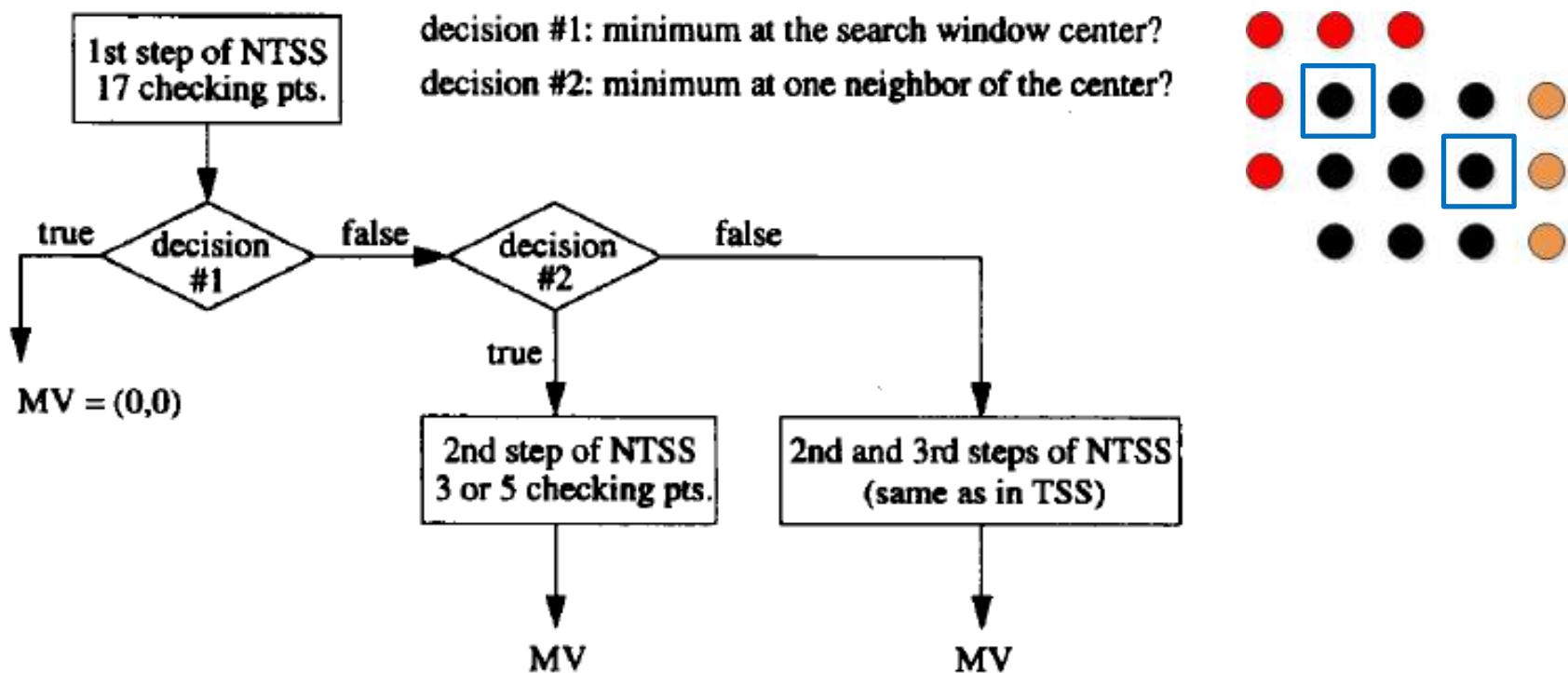
New TSS: NTSS [5]

- In the first step, in addition to the original checking points used in TSS, **eight extra points** are added, which are the eight neighbors of the search window center.
- A halfway-stop technique is used:
 - **The first-step-stop:** if the minimum Block Distortion Measure (BDM) in the first step occurs at the search window center, stop the search.
 - **The second-step-stop:** if the minimum BDM point in the first step is one of the eight neighbors of the window center, the search in the second step will be performed only for eight neighbors of the minimum and then stop the search.



New TSS: NTSS [5]

- For the maximum motion displacement ± 7 , the NTSS **in the worst case** requires **33** block matches while TSS needs only **25** block matches.



References

- [1] G. Johansson, “Visual perception of biological motion and a model for its analysis”, *Perception and Psychophysics*, vol.14, 201-211, 1973.
- [2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. of *International Joint Conf. On Artificial Intelligence*, pp.674-679, 1981.
- [3] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, 17:185-203, 1981.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *Proceedings of national Telecommunications conference*, New Orleans, LA, pp.G5.3.1–G5.3.5, Dec. 1981.
- [5] R. Li, B. Zeng, and M. L. Liou, “A new three-step algorithm for block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 4(4): 438-442, 1994.
- [6] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 6(3): 313-317, 1996.
- [7] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. On Image Processing*, 9(2): 287-290, 2000.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Motion Estimation — BMA

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Outline

- Optical Flow (Pixel-level)
 - What is optical flow?
 - Lucas-Kanade algorithm (LK) [2]
 - Horn-Schunck algorithm (HS) [3]
- BMA (Block-level)
 - The principle of BMA
 - Full search scheme
 - Three step search [4]
 - New three step search [5]
 - Four step search [6]
 - Diamond search scheme [7]

Four-Step Search: FSS [6]

- 1) For the maximum motion displacement of ± 7 , the proposed FSS algorithm utilizes a center-biased search pattern with 9 checking points on a 5×5 window in the first step.
 - If the minimum BDM point is found at the center of the search window, go to step 4; otherwise go to step 2.
- 2) The search window is kept as 5×5 . But the search pattern will depend on the position of the previous minimum BDM point.
 - If the previous minimum BDM point is located at the **corner** of the previous search window, 5 additional checking points are used.
 - If the previous minimum BDM point is located at the **edge** of the previous search window, 3 additional checking points are used.
- 3) The searching pattern strategy is the same as step 2, but finally it will go to step 4.
- 4) The search window is reduced to 3×3 .

Four-Step Search: FSS [6]

- The total number of checking points is varied from $(9+8) = 17$ to $(9+5+5+8) = 27$. The worst case computational requirement of the FSS is 27 block matches.

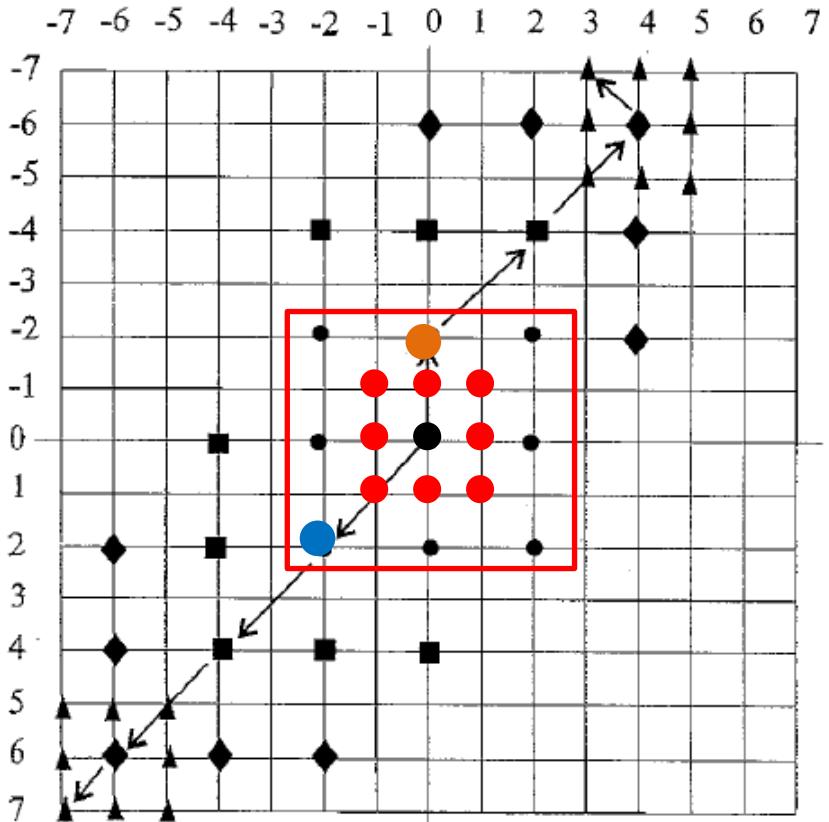


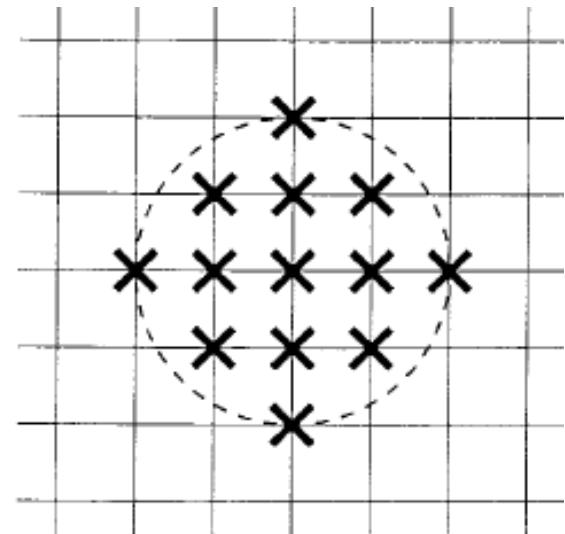
Fig. 3. Two different search paths of 4SS.

Corner Point

Edge Point

Diamond Search Algorithm: DS [7]

- **The error surface is usually not monotonic:**
 - Small search pattern → quite likely to be trapped into local minimum for those video sequences with large motion.
 - Large search pattern → most likely to mislead the search path to a wrong direction.
- **Center-biased:** the distribution of the global minimum points is centered at the position of zero motion.



Diamond Search Algorithm: DS

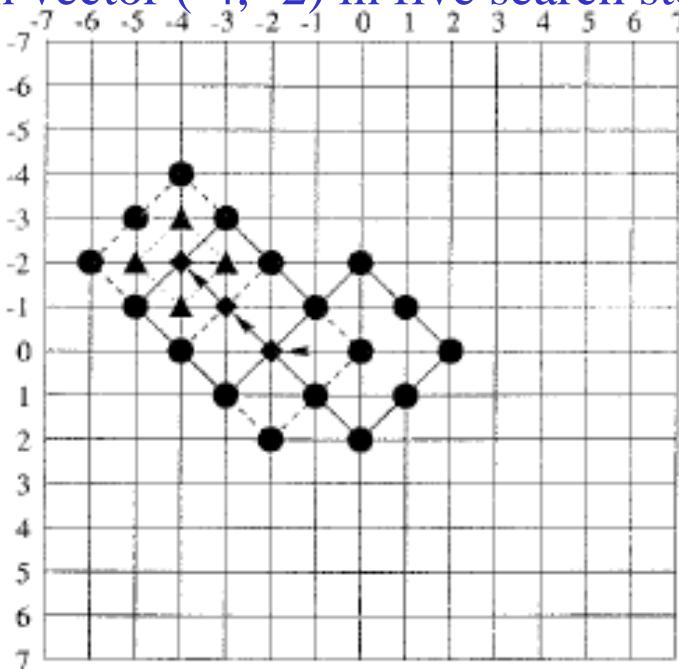
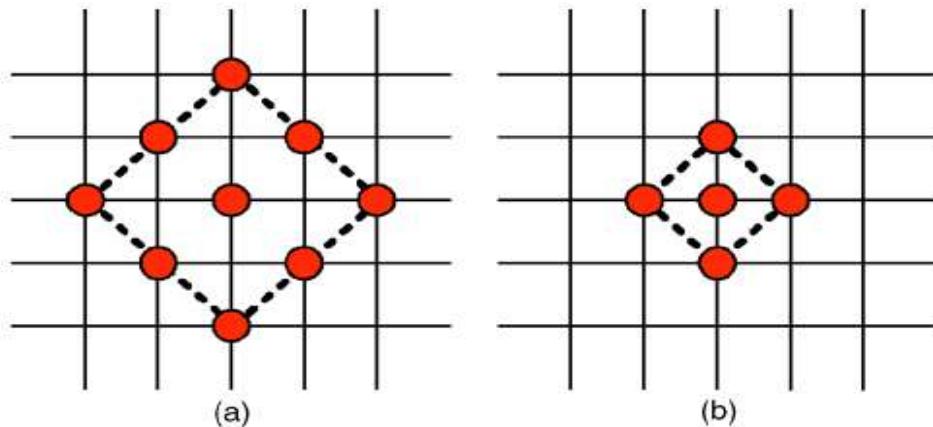
- The motion vector distribution probabilities within certain distances from the search window center.
 - About **52.76% to 98.70%** of the motion vectors are enclosed **in a circular support with a radium of 2 pixels** and centered on the position of zero motion.
 - The block displacement mainly in horizontal and vertical directions.

Radium (pel)	Tennis	Football	Caltrain	Susie	Salesman	Claire
0	0.2622	0.6196	0.0416	0.0938	0.6562	0.9076
1	0.3751	0.7297	0.5373	0.3592	0.9452	0.9702
2	0.5276	0.7983	0.8523	0.5950	0.9609	0.9870
3	0.7178	0.8641	0.9168	0.7622	0.9741	0.9932
4	0.8402	0.9042	0.9380	0.8225	0.9795	0.9950
5	0.8930	0.9329	0.9561	0.8779	0.9853	0.9957
6	0.9200	0.9483	0.9720	0.9038	0.9957	0.9964
7	0.9599	0.9658	0.9894	0.9365	0.9975	0.9973

Diamond Search Algorithm: DS

- Large Diamond Search Pattern (LDSP) and Small Diamond Search Pattern (SDSP)
 - LDSP is repeatedly used until the step in which the minimum block distortion (MBD) occurs at *the center point*.
 - switched from LDSP to SDSP

Search path example which leads to the motion vector (-4, -2) in five search steps.



References

- [1] G. Johansson, “Visual perception of biological motion and a model for its analysis”, *Perception and Psychophysics*, vol.14, 201-211, 1973.
- [2] B. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in Proc. of *International Joint Conf. On Artificial Intelligence*, pp.674-679, 1981.
- [3] B. Horn and B. Schunck, “Determining optical flow,” *Artificial Intelligence*, 17:185-203, 1981.
- [4] T. Koga, K. Iinuma, A. Hirano, Y. Iijima, and T. Ishiguro, “Motion compensated interframe coding for video conferencing,” *Proceedings of national Telecommunications conference*, New Orleans, LA, pp.G5.3.1–G5.3.5, Dec. 1981.
- [5] R. Li, B. Zeng, and M. L. Liou, “A new three-step algorithm for block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 4(4): 438-442, 1994.
- [6] L.-M. Po and W.-C. Ma, “A novel four-step search algorithm for fast block motion estimation,” *IEEE Trans. On Circuits and Systems for Video Technology*, 6(3): 313-317, 1996.
- [7] S. Zhu and K.-K. Ma, “A new diamond search algorithm for fast block-matching motion estimation,” *IEEE Trans. On Image Processing*, 9(2): 287-290, 2000.

Thank You!

Dr. Xiqun Lu

xqlu@zju.edu.cn

Image Compression

— JPEG

Dr. Xigun Lu

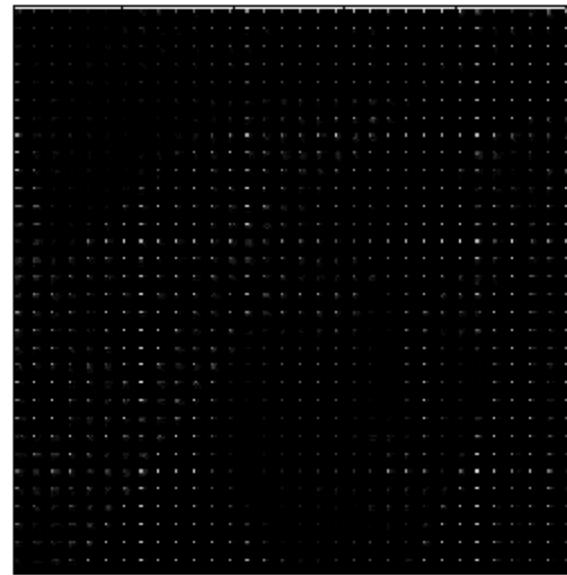
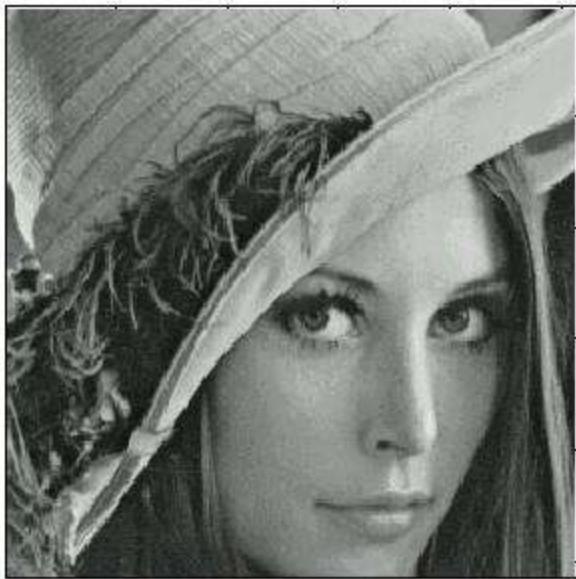
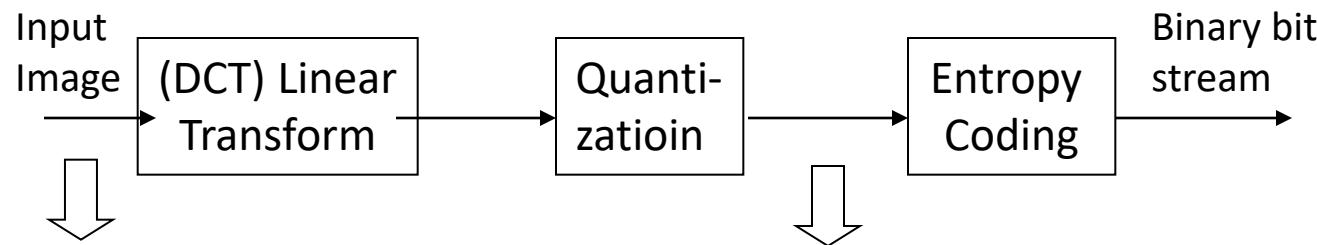
College of Computer Science

Zhejiang University

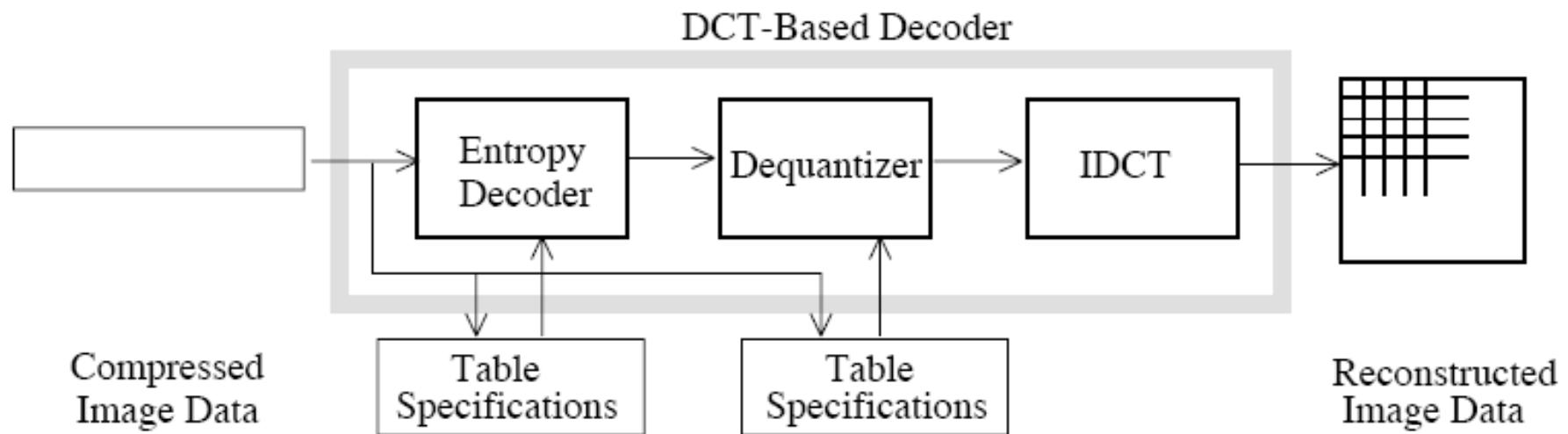
Overview of JPEG [1]

- JPEG is a standardized compression method for full-color and gray-scale images.
- JPEG is intended for compressing "**real-world**" scenes, but **NOT** for
 - line drawings
 - Cartoons
 - other non-realistic images
- JPEG can be **lossy or lossless**
 - Lossy: DCT-based method — the output image is not exactly identical to the input image.
 - Lossless: Predictive-based method

Encoder of Sequential Mode

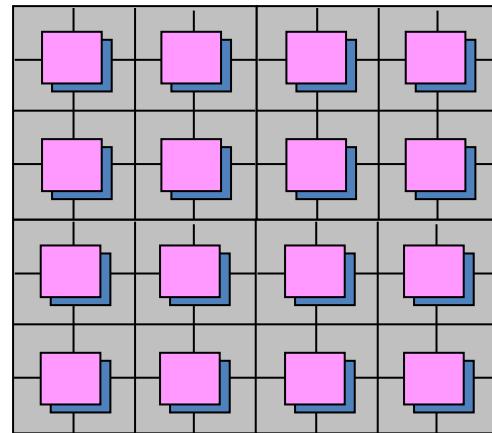


Decoder of Sequential Mode



Pre-Processing

- Color sub-sampling
 - A color image is converted from RGB to YUV color space.
 - Sub-sample U-V planes: 4:1:1 scheme.
 - For every 16 by 16 block of a color image, six 8 by 8 blocks are encoded.
- Level shifting: Each pixel value is subtracted by 128, so it ranges (-128, 127).



Four 8×8 blocks of luminance pixels, plus two 8×8 sub-sampled chrominance components makes a 16 by 16 **macro-block**

2D DCT

- 8×8 DCT

$$F(u, v) = \frac{1}{4} C(u)C(v) \left[\sum_{x=0}^7 \sum_{y=0}^7 f(x, y) \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16} \right]$$

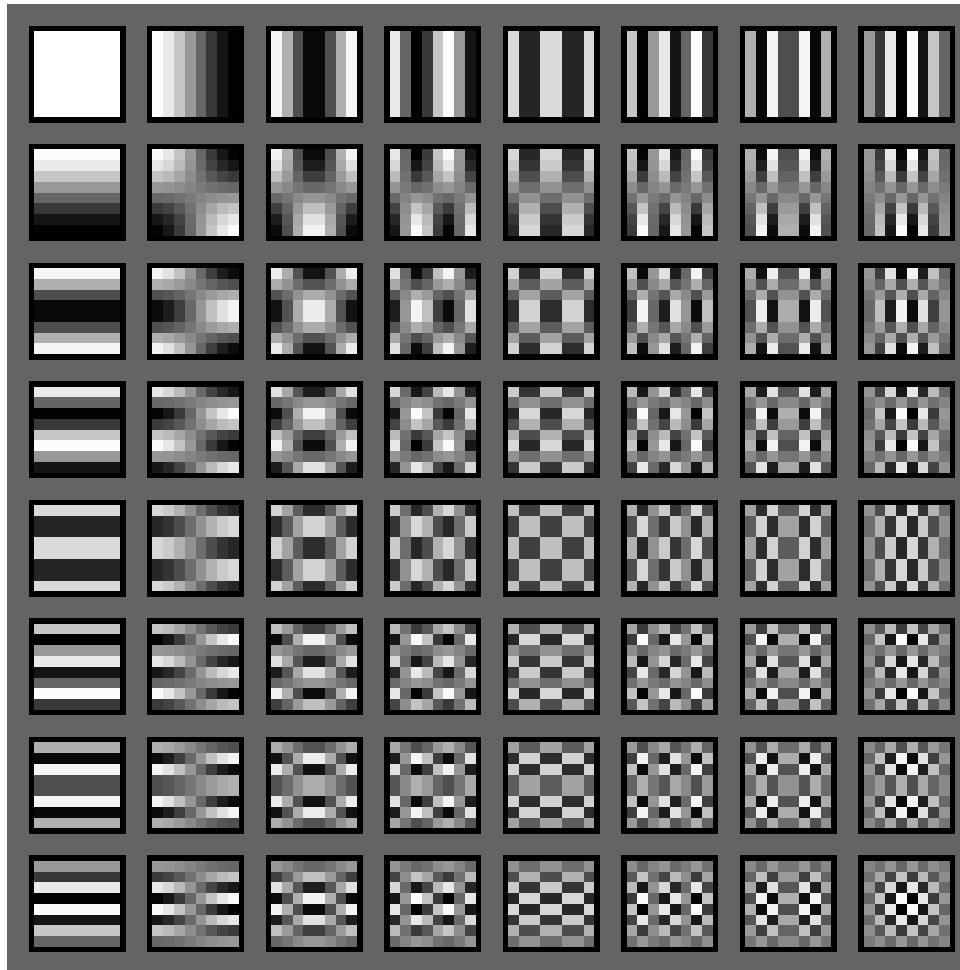
$$f(x, y) = \frac{1}{4} \left[\sum_{x=0}^7 \sum_{y=0}^7 C(u)C(v) F(u, v) \cos \frac{(2x+1)u\pi}{16} \cos \frac{(2y+1)v\pi}{16} \right]$$

Where $C(u), C(v) = 1/\sqrt{2}$ for $u, v = 0$

$C(u), C(v) = 1$ otherwise

The FDCT takes each 8×8 as its input and decomposes it into 64 orthogonal basis signals. Each contains one of the 64 unique 2D “spatial frequencies” which comprise the input signal’s “**spectrum**”.

Basic Images of 8×8 DCT

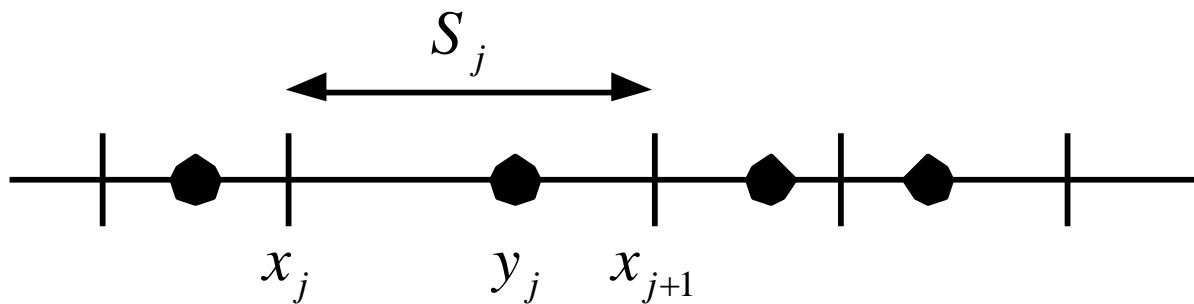


The Idea Behind the DCT

- Because sample values typically **vary slowly** from point to point across an image, the FDCT processing step lays the foundation for achieving data compression by **concentrating most of the signal energy in the lower spatial frequencies**.
- In principle, the DCT introduces **no loss** to the source image samples if the FDCT and IDCT could be computed with perfect accuracy.
 - However, the equations to compute the coefficient of DCT contain “cosine” functions, so no physical implementation can compute them with perfect accuracy.

Quantization

- **Lossy:** to discard information not visually significant!
- Many-to-one mapping



Quantization will introduce
quantization **error (or noise)**

DPCM of DC Coefficients

- DC coding: All DC coefficients of each 8 by 8 blocks of the entire image are combined to make a sequence of DC coefficients.
- Next, DPCM is applied:
$$\text{DiffDC}(\text{block}_i) = \text{DC}(\text{block}_i) - \text{DC}(\text{block}_{i-1})$$
- Then DiffDCs will be encoded using Huffman entropy

1216	1232	1224
1248	1248	1208

Example:

- Original:

1216 → 1232 → 1224 →
1248 → 1248 → 1208

- After DPCM:

1216 → +16 → -8 → +24
→ 0 → -40

AC Coefficients

- AC coefficients are first weighted with a *quantization matrix*:

$$C(i,j)/q(i,j) = C_q(i,j)$$

Then quantized.

- Then they are scanned in a zig-zag order into a 1D sequence to be subject to AC Huffman encoding.
- This ordering helps to facilitate entropy coding by placing low-frequency coefficients before high-frequency coefficients.
- Question: Given a 8 by 8 array, how to convert it into a vector according to the zig-zag scan order? What is the algorithm?

Zig-Zag scan order

1	2	6	7	15	16	28	29
3	5	8	14	17	27	30	43
4	9	13	18	26	31	42	44
10	12	19	25	32	41	45	54
11	20	24	33	40	46	53	55
21	23	34	39	47	52	56	61
22	35	38	48	51	57	60	62
36	37	49	50	58	59	63	64

Baseline Encoding Example

139	144	149	153	155	155	155	235.6	-1.0	-12.1	-5.2	2.1	-1.7	-2.7	1.3	16	11	10	16	24	40	51	61
144	151	153	156	159	156	156	-22.6	-17.5	-6.2	-3.2	-2.9	-0.1	0.4	-1.2	12	12	14	19	26	58	60	55
150	155	160	163	158	156	156	-10.9	-9.3	-1.6	1.5	0.2	-0.9	-0.6	-0.1	14	13	16	24	40	57	69	56
159	161	162	160	160	159	159	-7.1	-1.9	0.2	1.5	0.9	-0.1	0.0	0.3	14	17	22	29	51	87	80	62
159	160	161	162	162	155	155	-0.6	-0.8	1.5	1.6	-0.1	-0.7	0.6	1.3	18	22	37	56	68	109	103	77
161	161	161	161	160	157	157	1.8	-0.2	1.6	-0.3	-0.8	1.5	1.0	-1.0	24	35	55	64	81	104	113	92
162	162	161	163	162	157	157	-1.3	-0.4	-0.3	-1.5	-0.5	1.7	1.1	-0.8	49	64	78	87	103	121	120	101
162	162	161	161	163	158	158	-2.6	1.6	-3.8	-1.8	1.9	1.2	-0.6	-0.4	72	92	95	98	112	100	103	99

(a) source image samples

(b) forward DCT coefficients

(c) quantization table

	15	0	-1	0	0	0	0	240	0	-10	0	0	0	0	0	144	146	149	152	154	156	156	156
(1, 2) (-2)	-2	-1	0	0	0	0	0	-24	-12	0	0	0	0	0	0	148	150	152	154	156	156	156	156
(0, 1) (-1)	-1	-1	0	0	0	0	0	-14	-13	0	0	0	0	0	0	155	156	157	158	158	157	156	155
(0, 1) (-1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160	161	161	162	161	159	157	155
(0, 1) (-1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	163	163	164	163	162	160	158	156
(2, 1) (-1)	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	163	164	164	164	162	160	158	157
(0, 0) %	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	163	164	164	164	162	160	158	157
EOB	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	160	161	162	162	162	161	159	158
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	158	159	161	161	162	161	159	158

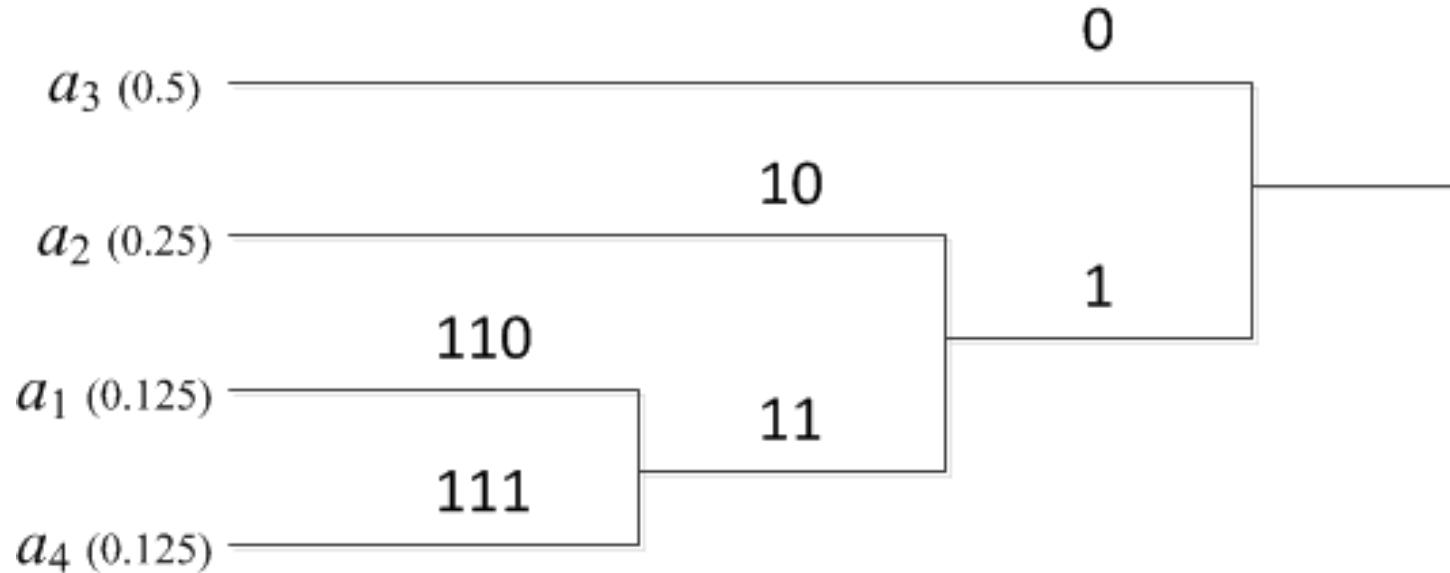
(d) normalized quantized coefficients

(e) denormalized quantized coefficients

(f) reconstructed image samples

Huffman Coding

- Suppose we have a discrete letter set $\{a_1, a_2, a_3, a_4\}$, and the occurrence probability of each letter is $P_1 = 0.125$, $P_2 = 0.25$, $P_3 = 0.5$, $P_4 = 0.125$.



No code is the prefix of other codes.

- Entropy

$$-\sum_i P_i \log_2(P_i)$$

References

- [1] G. K. Wallace "The JPEG still picture compression standard", Communications of the ACM, 34(4):30-44, April 1991.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Video Compression

— MPEG

Dr. Xigun Lu

College of Computer Science

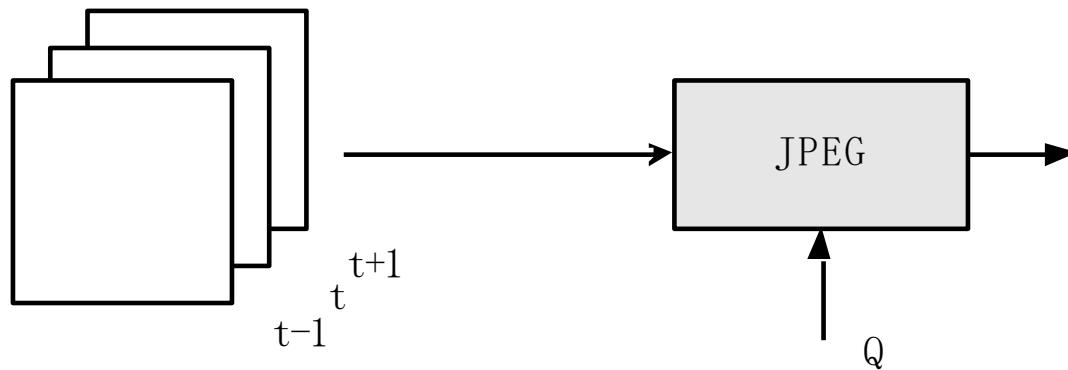
Zhejiang University

Still Image Coding vs. Video Sequence Coding

- Within a frame, it is the same as still image coding.
- Make compression in temporal dimension different from spatial ones

Intra-frame Coding

- Encode frame by frame, disregarding all temporal information.
- Easy bit allocation per frame.
- Random access is possible.
- Robust to decompression / transmission problems.
- Moderate compression capabilities.
- Example: Motion-JPEG (AVI compressed)



Temporal Redundancy

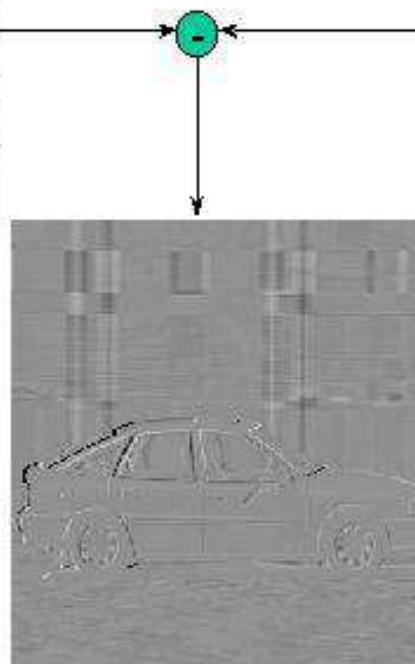
Example



Frame $t-1$



Frame t

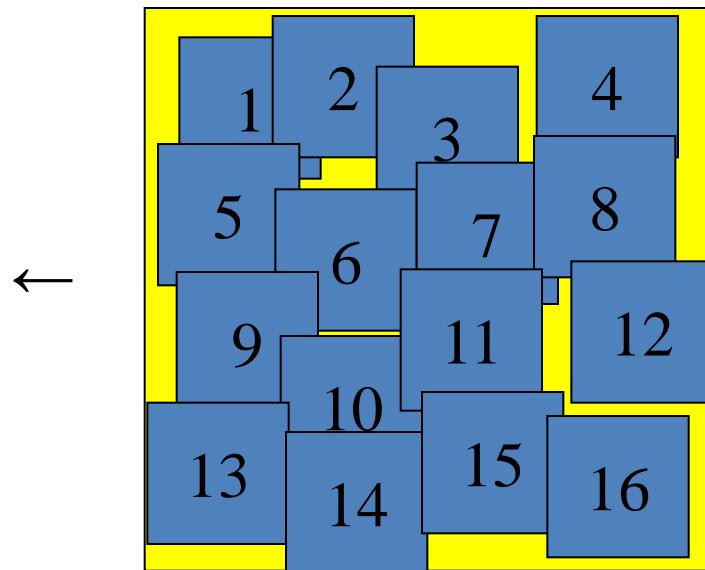


Frame difference
between t and $t-1$

Forward Motion Compensation

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

Predicted frame $\hat{x}(t)$
constructed from different
parts of reference frame



Reference (decoded **NOT** raw) frame
 $\tilde{x}(t-1)$

Frame 1



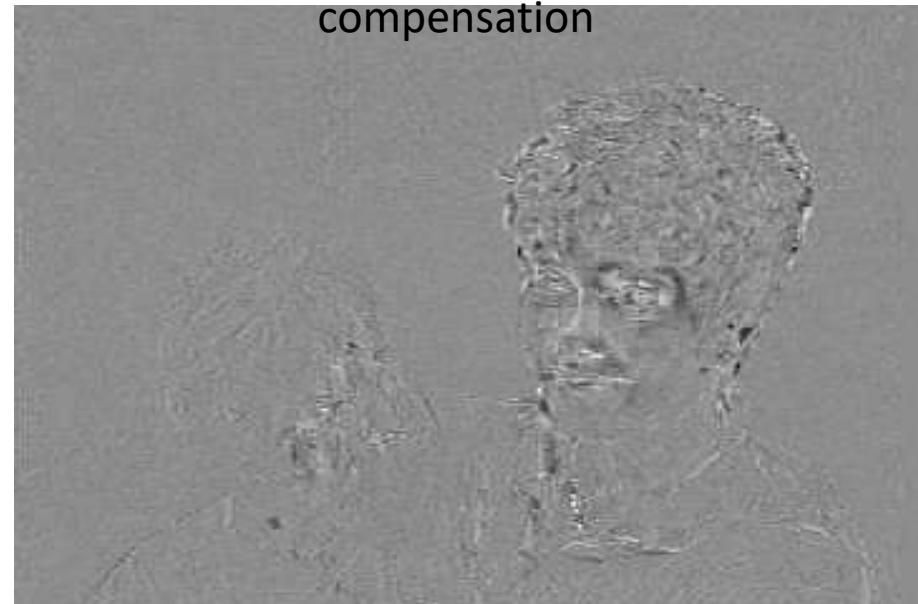
Frame 4



Absolute difference **without** motion compensation



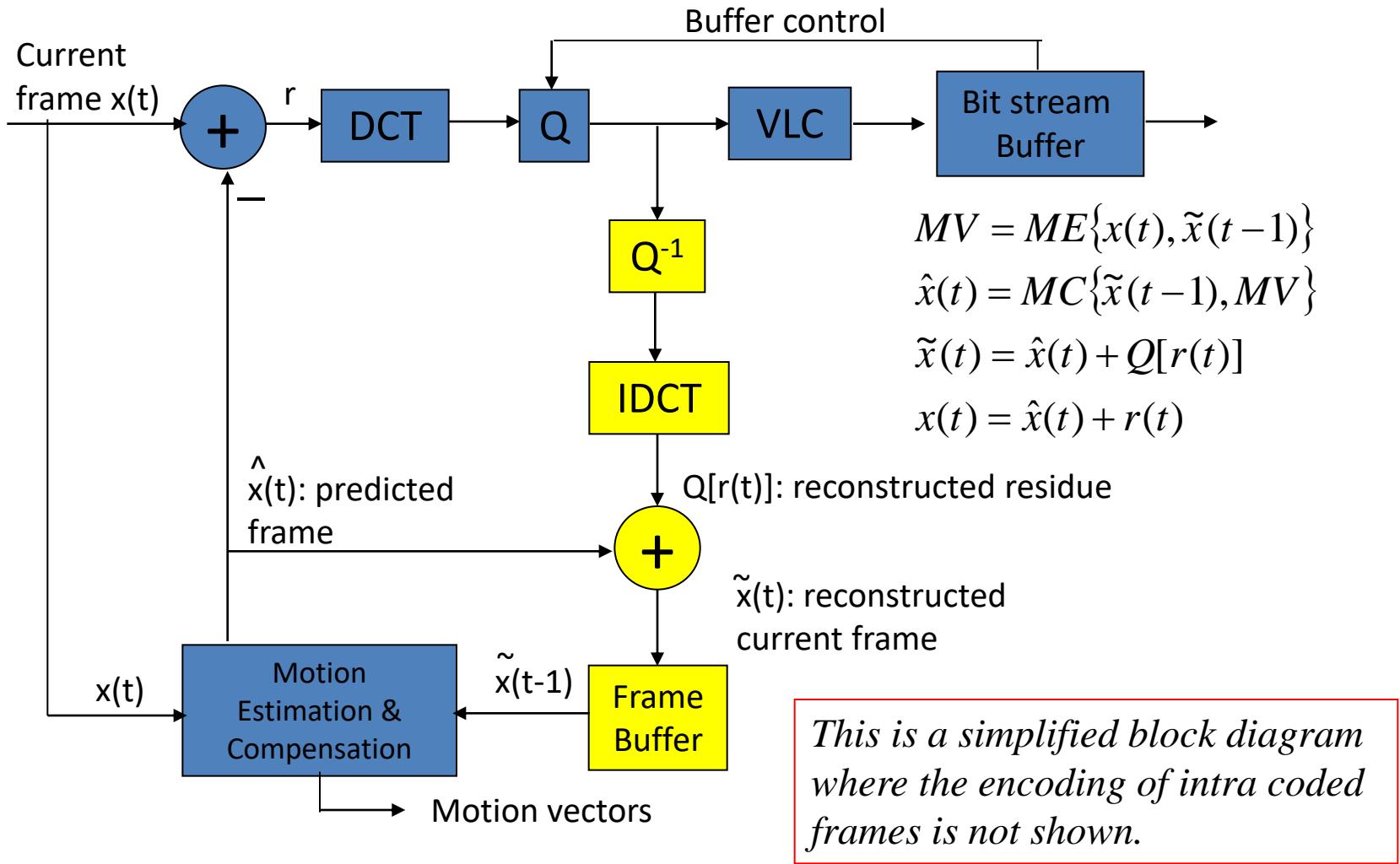
Absolute difference **with** motion compensation



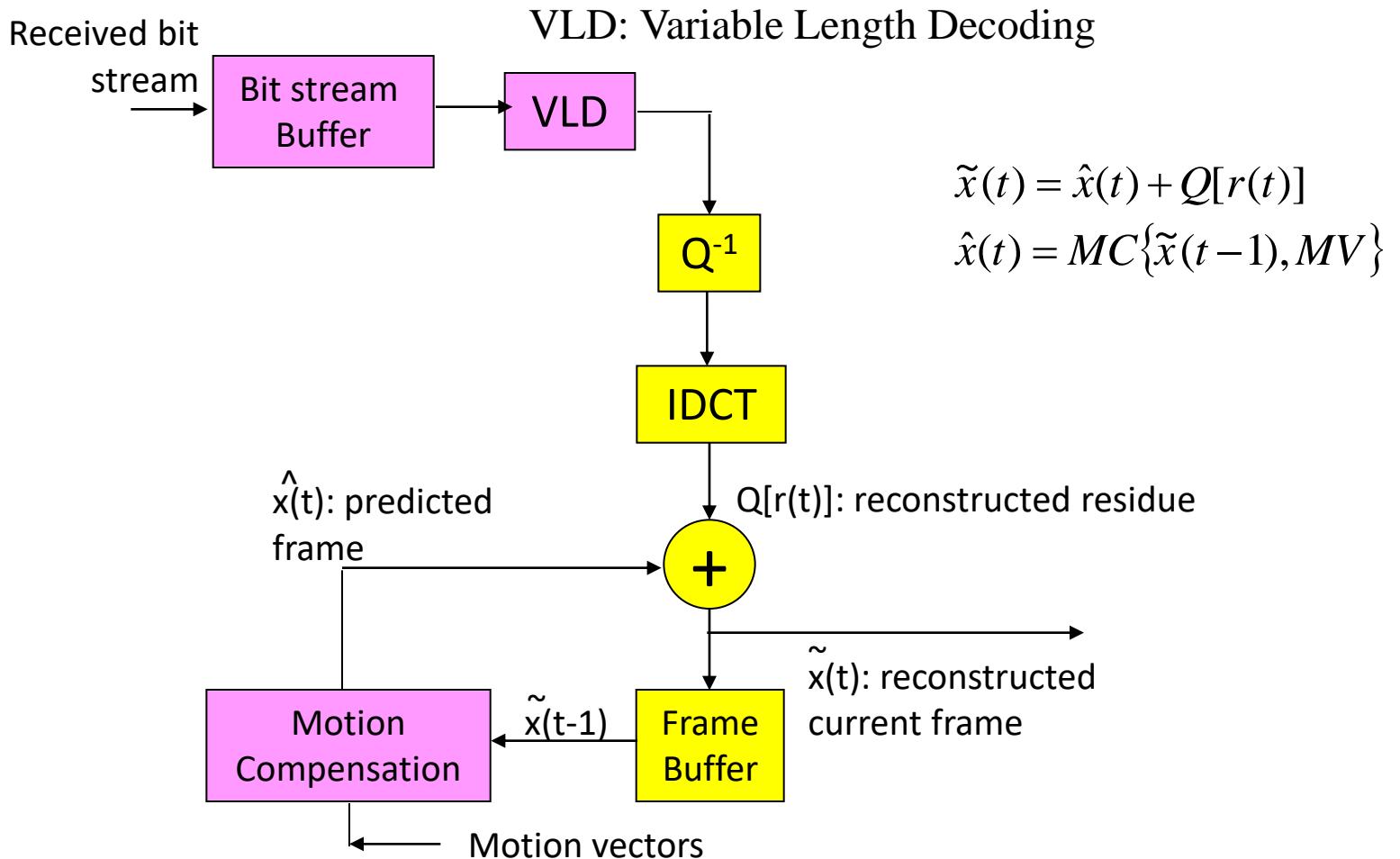
Motion Estimation Methods

- The basic motion estimation methods used in video compression:
 - **Block-Matching Algorithm**
 - Full Search
 - Three Step Search
 - New TSS
 - Four-Step Search
 - Diamond Search Algorithm

MPEG Encoding Framework



MPEG Decoding Framework



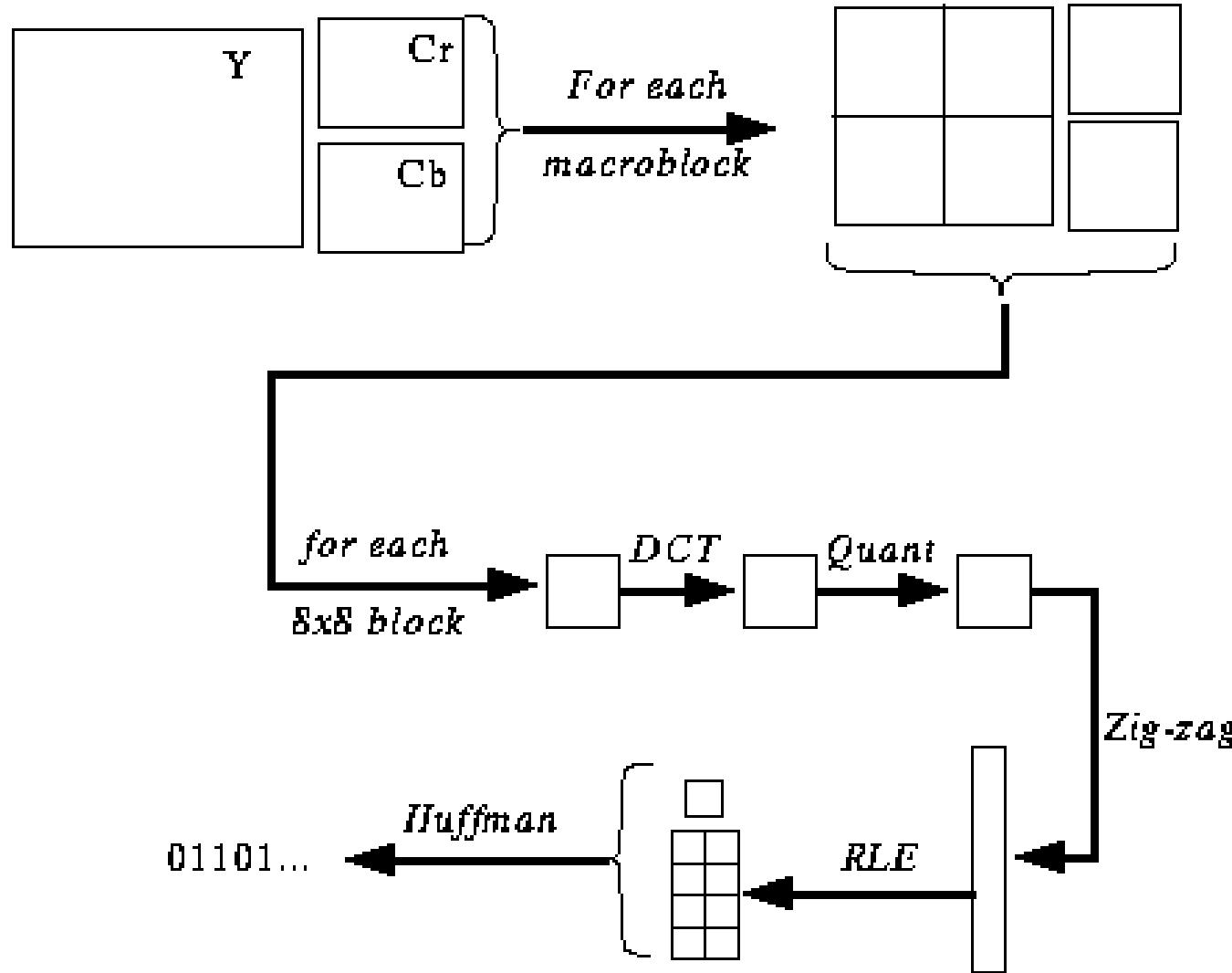
Motion Estimation

- Three types of frames:
 - Intra (**I**): the frame is coded as if it is an image
 - Predicted (**P**): predicted from an **I** frame
 - Bi-directional (**B**): forward and backward predicted from a pair of **I** or **P** frames.
- A typical frame arrangement is (subscripts are used to distinguish them):

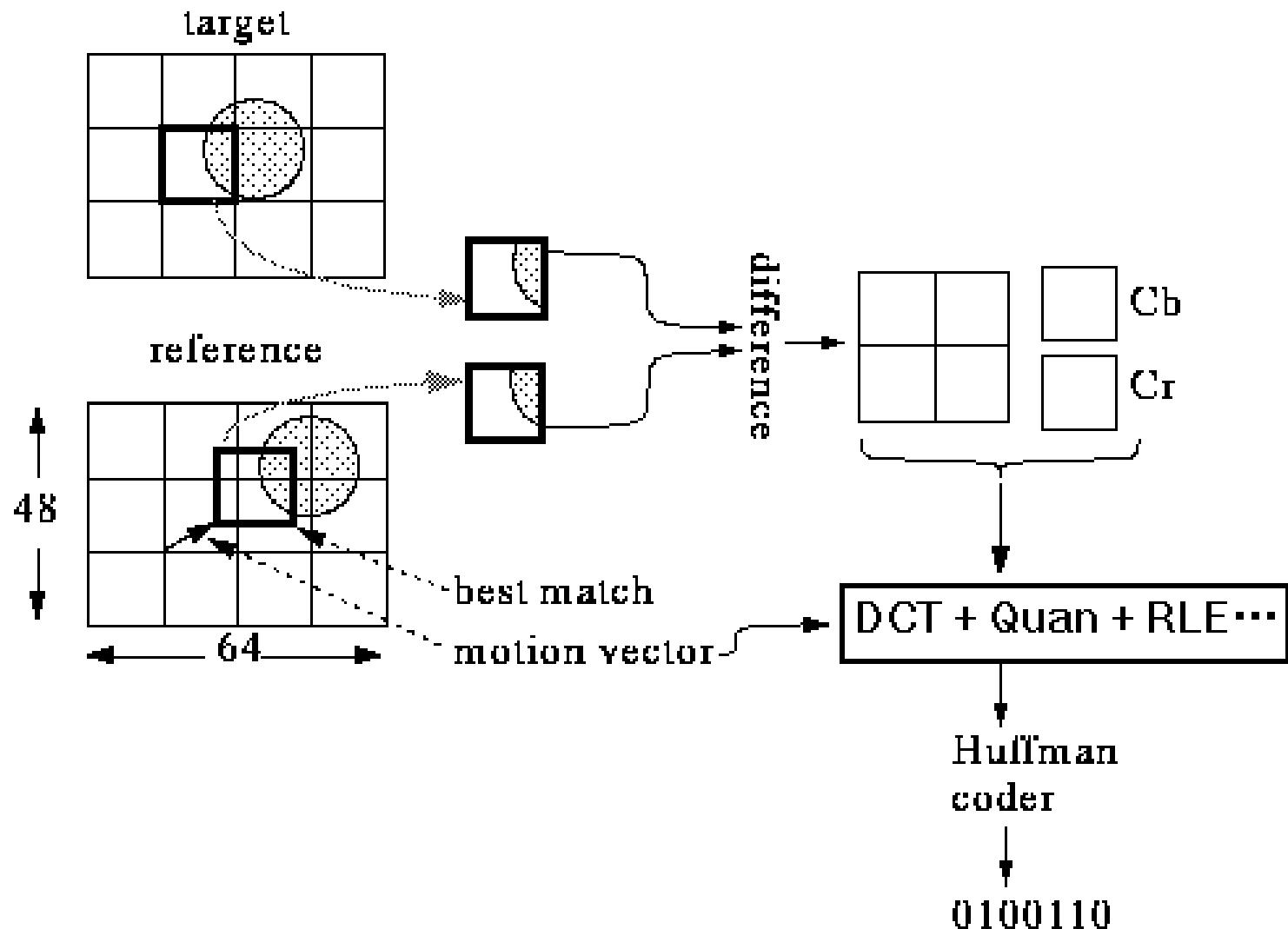
$I_1 \ B_1 \ B_2 \ P_1 \ B_3 \ B_4 \ P_2 \ B_5 \ B_6 \ I_2$

- P_1, P_2 are both forward-predicted from I_1 . B_1, B_2 are interpolated from I_1 and P_1 , B_3, B_4 are interpolated from P_1, P_2 , and B_5, B_6 are interpolated from P_2, I_2 .

Intra-frame Coding (I-frame)

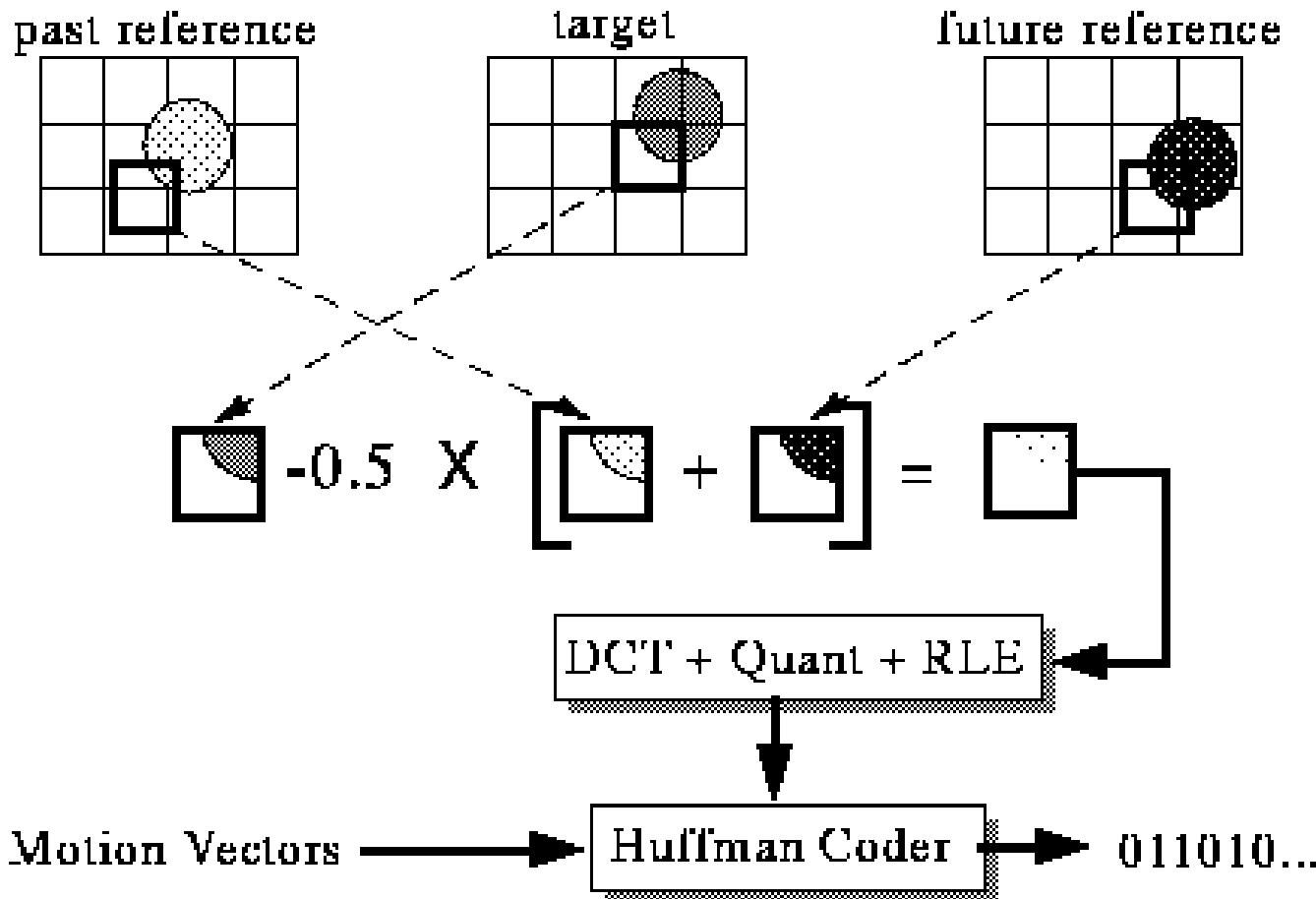


Inter-frame Coding (P-frame)



B-frame

- B frame macroblocks can specify *two* motion vectors (one to past and one to future), indicating result is to be averaged.



Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Introduction to Human Vision System (HVS)

Dr. Xiqun Lu

College of Computer Science

Zhejiang University

Understanding HVS, Why?

- Image or video is to be seen.
- Perceptual Based Image Processing
 - Focus on perceptually significant information
 - Discard perceptually insignificant information
- Issues:
 - Biological
 - Psychophysical

Anatomy of Human Eye

choroid 脉络膜

retina 视网膜

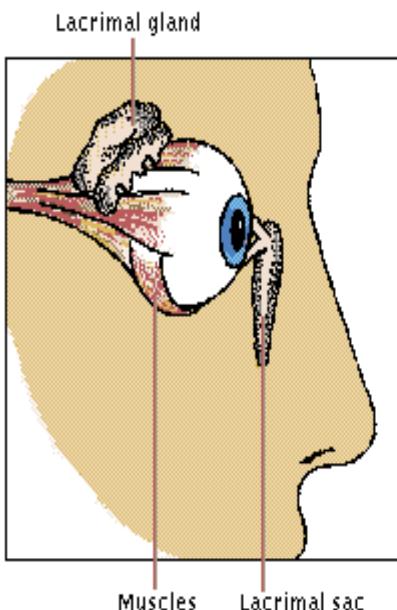
Cornea 角膜

Aqueous humor 眼房水

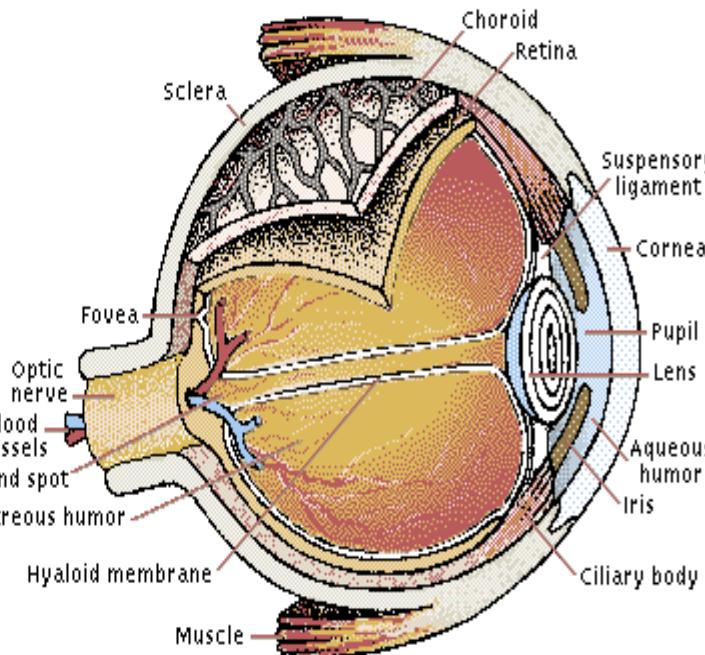
Fovea 黄斑

Vitreous humor 玻璃体

ciliary body 睫状体



Microsoft Illustration



The amount of light entering the eye is controlled by the pupil, which dilates and contracts accordingly. The cornea and lens, whose shape is adjusted by the ciliary body, focus the light on the retina, where receptors convert it into nerve signals that pass to the brain. A mesh of blood vessels, the choroid, supplies the retina with oxygen and sugar.

<http://psych.athabascau.ca/html/Psych402/Biotutorials/22/intro.shtml>

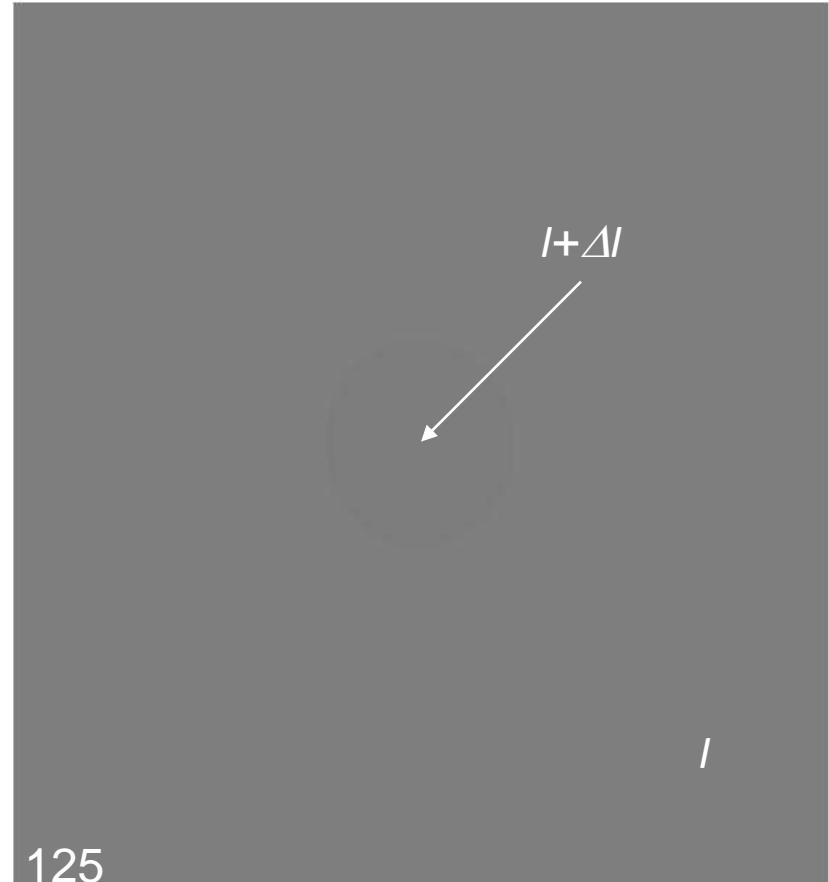
Visual Psychophysics

- Model vision "system" as an input-output system
 - visual stimuli: input
 - prescribed sensations: output.
- Visual psychophysics:
 - Characterize the response of HVS to different stimuli

Brightness Discrimination

- Can you see the circle?
- ΔI is the just noticeable difference (JND) [1].
- Weber's Law:

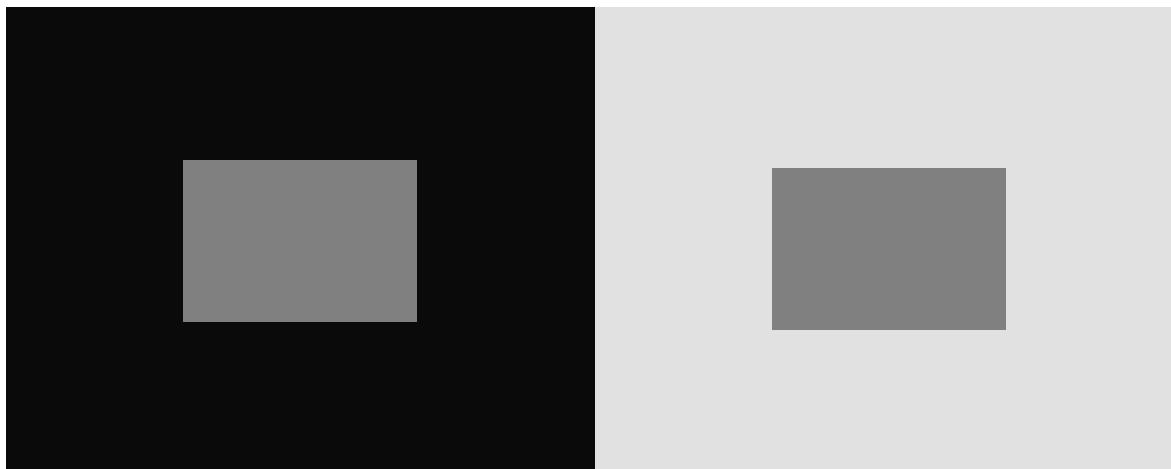
$$\frac{\Delta I}{I} \approx const$$



This rule is first discovered by Ernst Heinrich Weber (1795-1878), an anatomist and physiologist.

Luminance and Brightness

- **Luminance** measures physical light intensity.
- The perceived intensity, or **brightness**, depends upon the luminance of both the object and the background.



The illusion of simultaneous contrast

The luminance of both center objects is the same

Contrast

- The contrast is the ratio between the brightest intensity level and the dimmest.

$$C = \frac{I_{max} - I_{min}}{I_{max} + I_{min}}$$

$$0 \leq C \leq 1$$

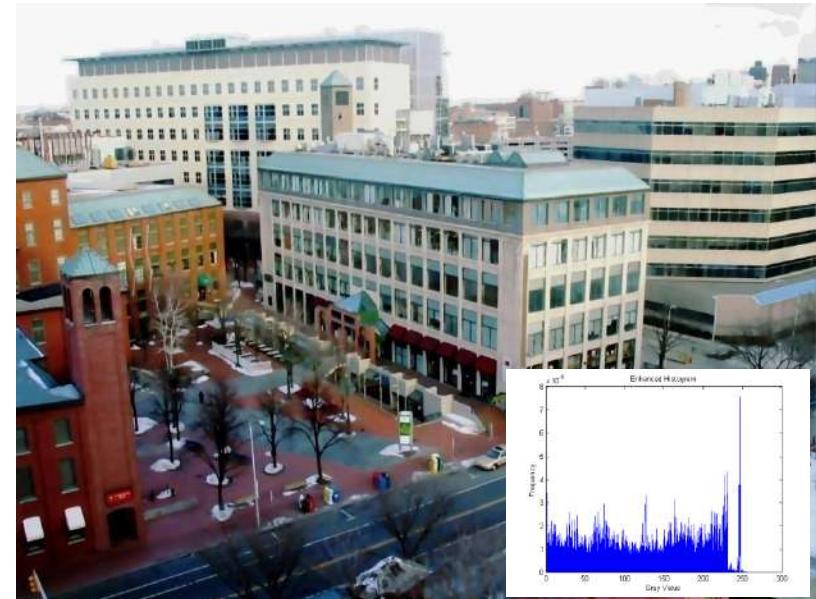
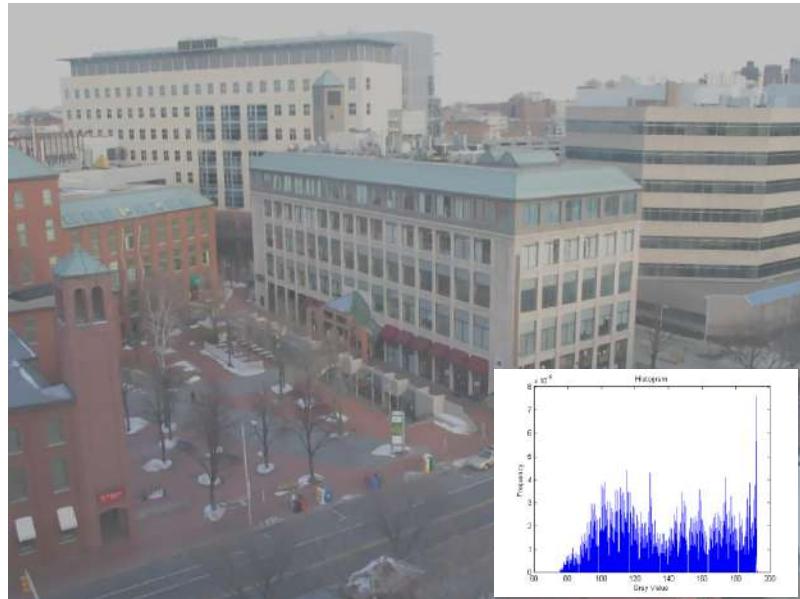
- Low contrast image has “closely spaced” distribution of intensity values (i.e., contrast close to 0)
- High contrast image has “widely spaced” distribution of intensity values (i.e., contrast close to 1)

Low Contrast vs. High Contrast



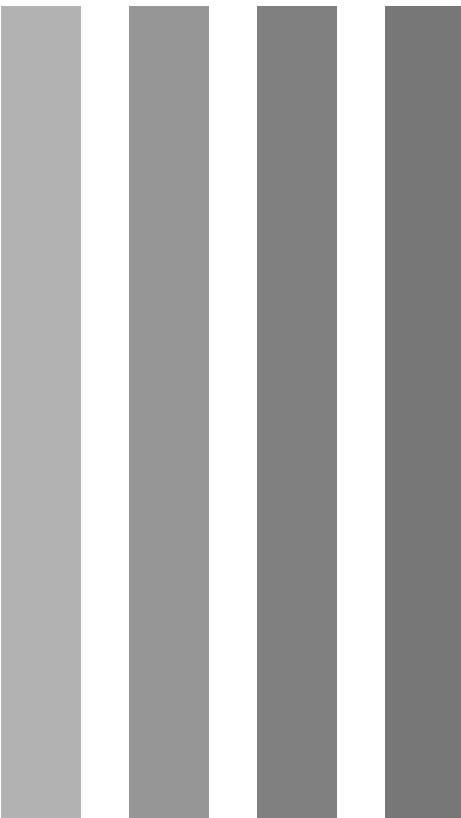
Enhance Contrast Using Histogram Equalization

Low Contrast vs. High Contrast

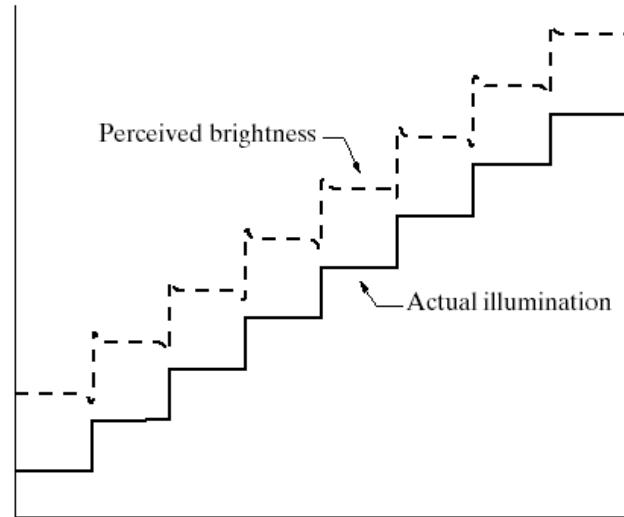
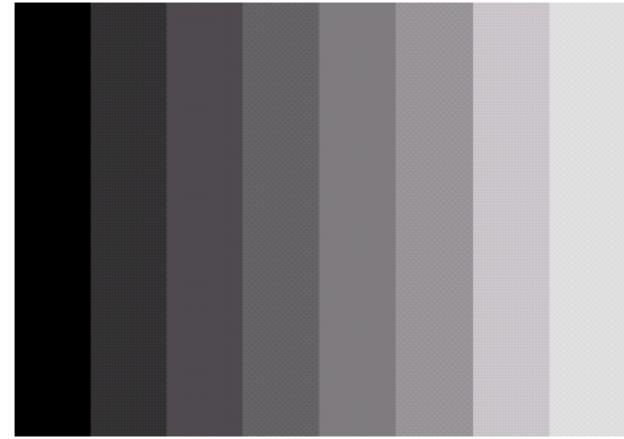


Applying Histogram Equalization to RGB color channels independently

Mach Band Effect [2]



- Note that each bar on the left appears to have color variation across it
 - Left edge appears darker than right
- The effect is entirely due to Mach banding



Why 8-bit/pixel for standard gray image?

256 levels



128 levels



64 levels



32 levels

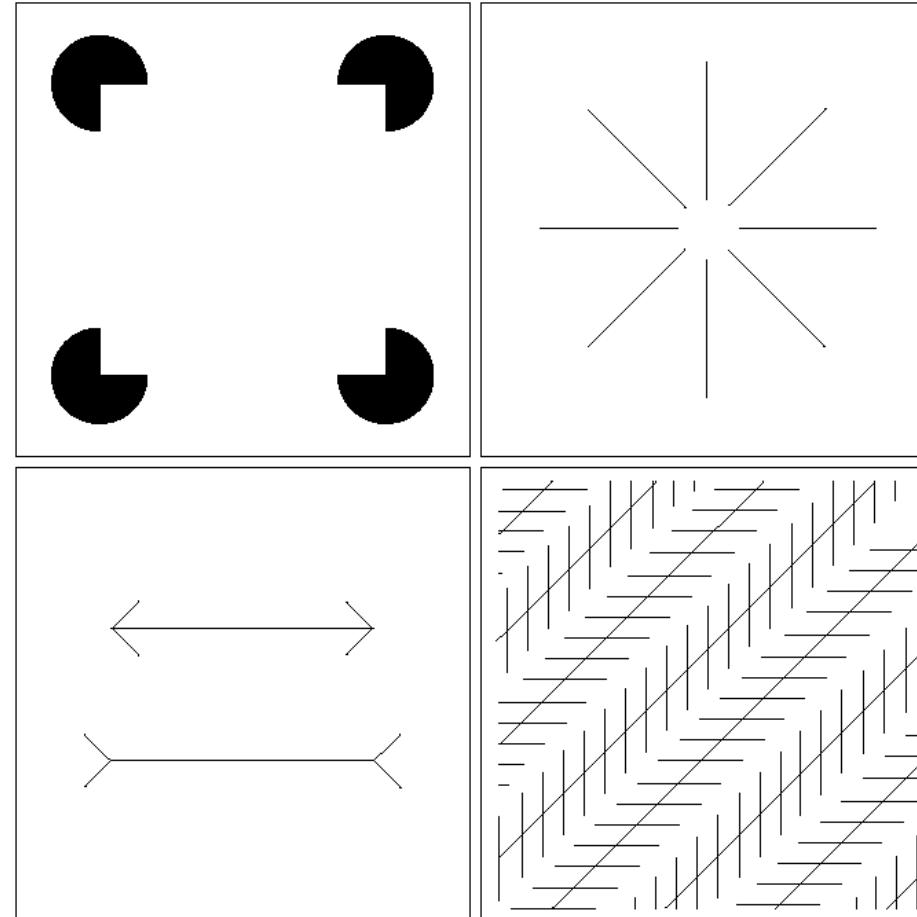


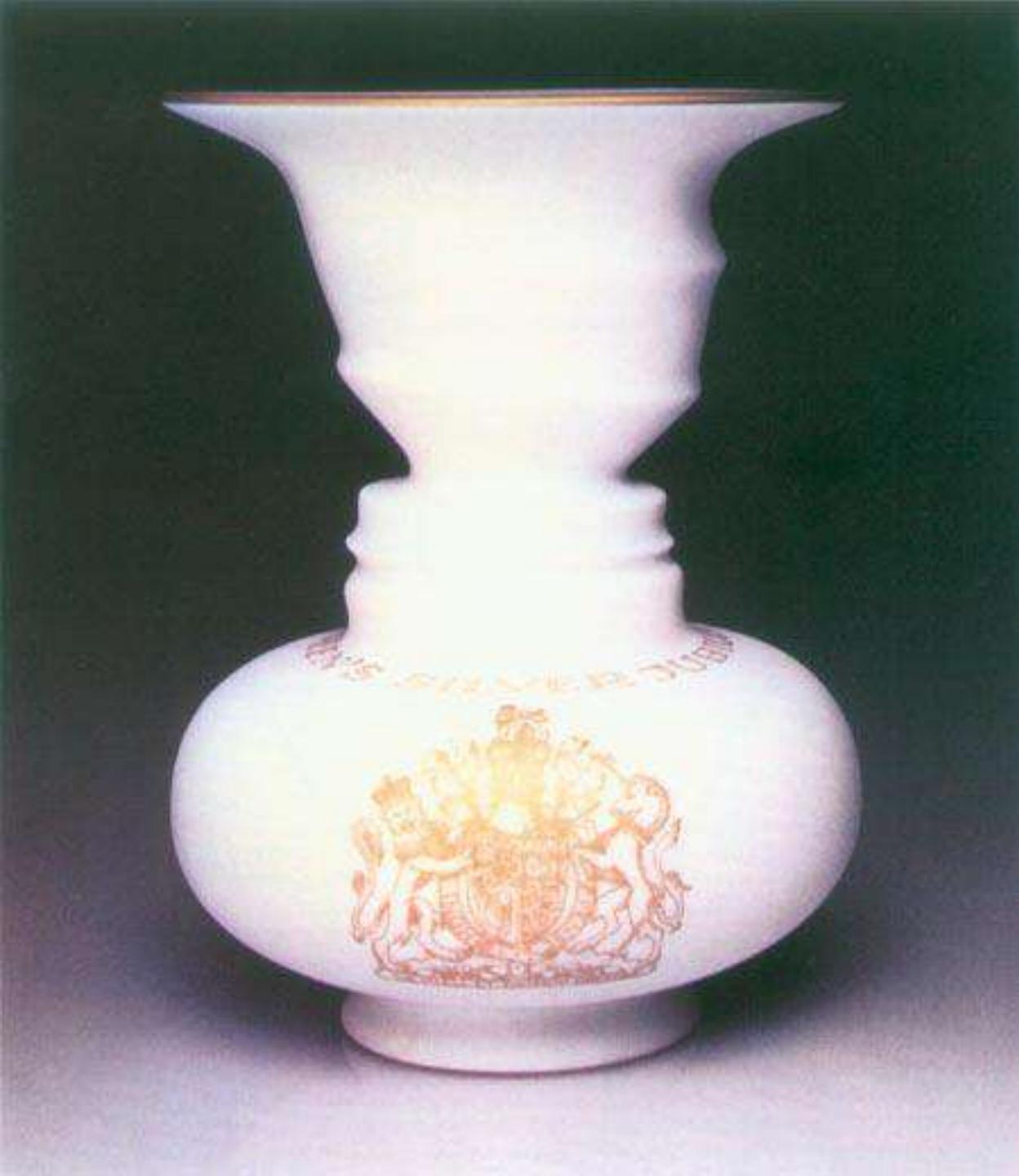
- Digital images are typically quantized to 256 gray levels.

Optical Illusions

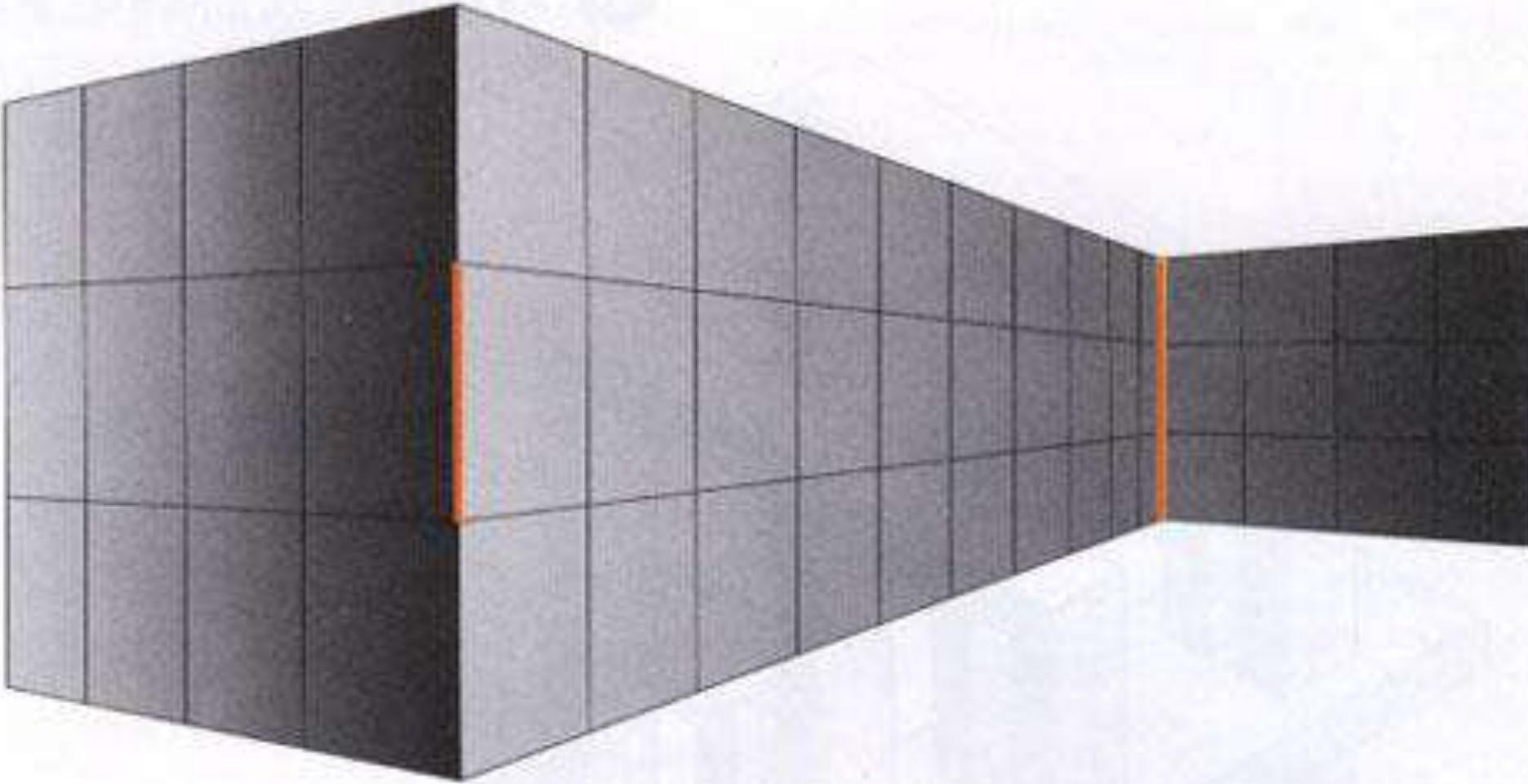
a
b
c
d

FIGURE 2.9 Some well-known optical illusions.



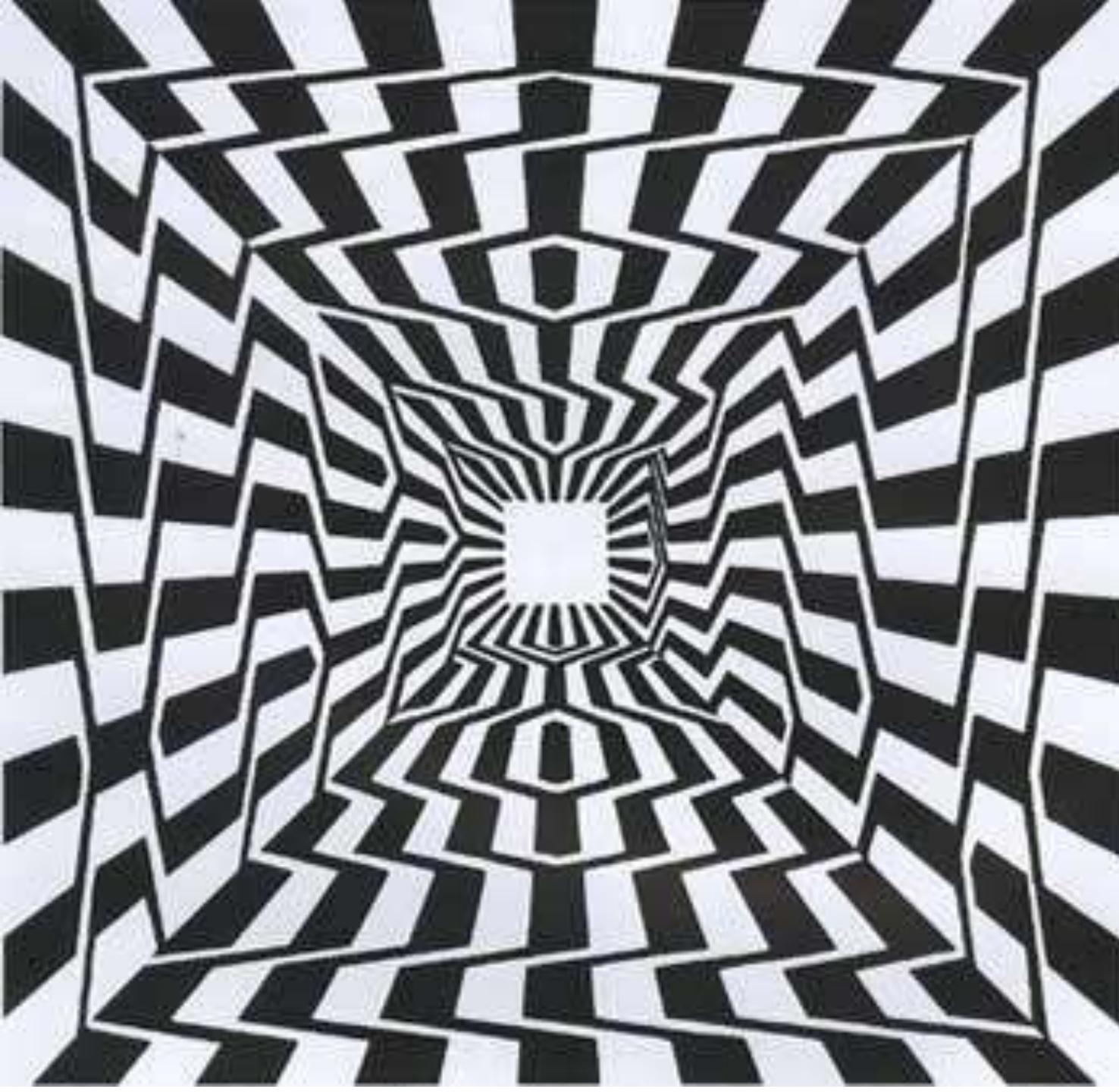
The image shows a white vase with a flared top and a textured base, set against a dark background. The vase features a decorative band with a repeating pattern near its base. The pattern consists of orange and yellow shapes that can be interpreted as either a floral design on a light background or a dark, swirling pattern on a light background, depending on whether one sees the vase or the background figures.

Rubin’s Vase Illusion or Figure- Ground Vase

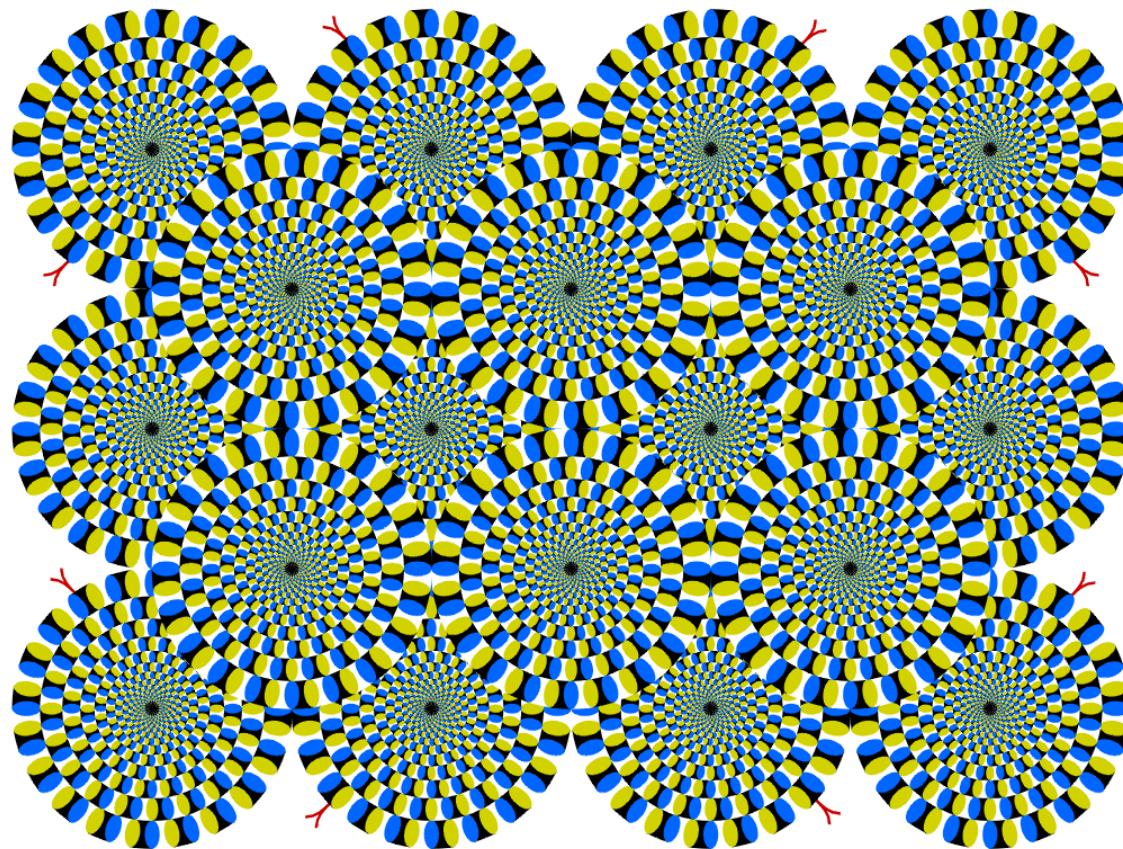


Müller-Lyer Illusion: Which red line is longer?
Actually these two red lines are with equal length

Are these perfect squares?



Anomalous Motion Illusion [3]



<http://www.ritsumei.ac.jp/~akitaoka/index-e.html>

References

- [1] <http://apps.usd.edu/coglab/WebersLaw.html>
- [2] http://en.wikipedia.org/wiki/Mach_bands
- [3] <http://www.ritsumei.ac.jp/~akitaoka/index-e.html>

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Color Spaces and Evaluation Metrics

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Color Spectrum

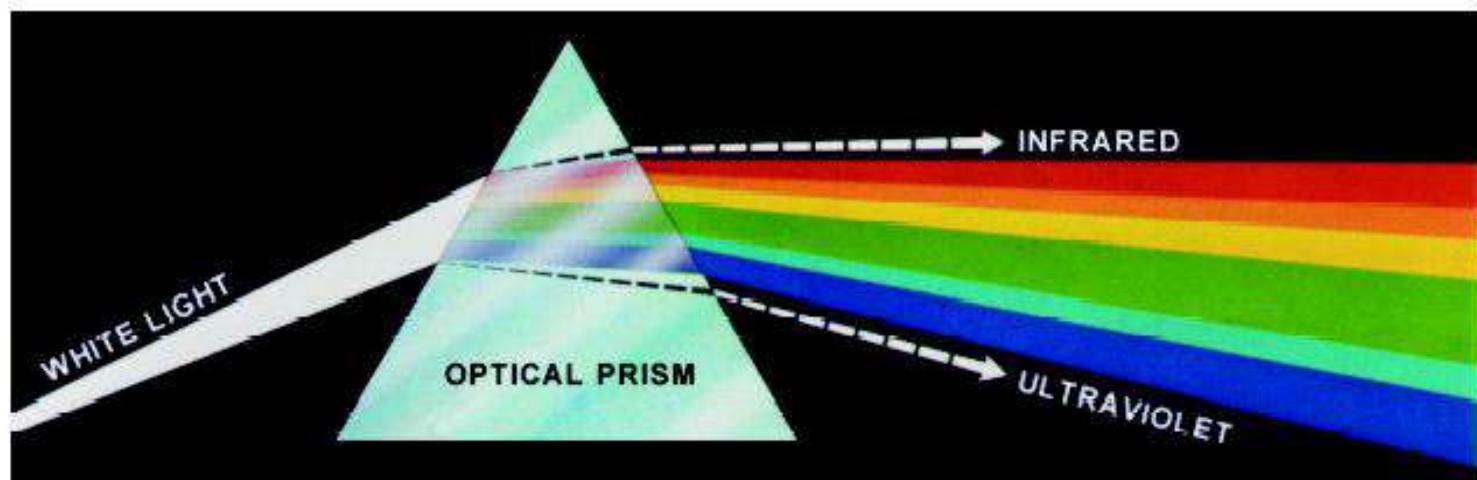


FIGURE 6.1 Color spectrum seen by passing white light through a prism. (Courtesy of the General Electric Co., Lamp Business Division.)

Wavelengths of the Electromagnetic Spectrum

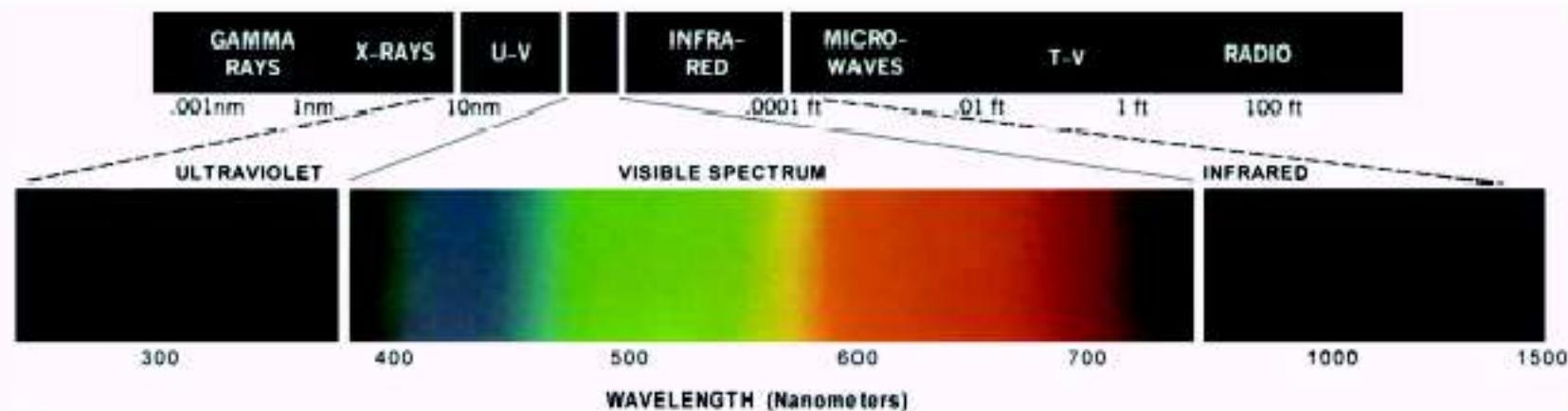


FIGURE 6.2 Wavelengths comprising the visible range of the electromagnetic spectrum. (Courtesy of the General Electric Co., Lamp Business Division.)

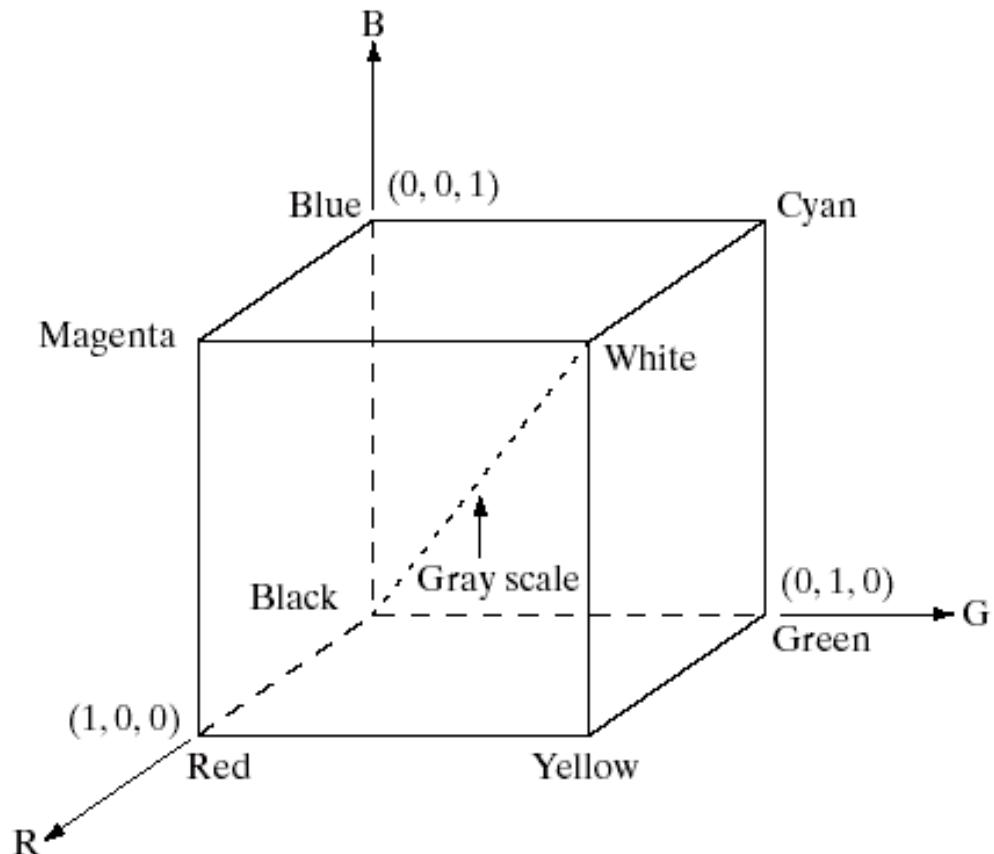
Grassman's First Law of Additive Color Mixture

- Any color can be matched by a linear combination of three other colors (primaries, e.g. RGB), provided that none of those three can be matched by a combination of the other two.
- $C = R_c(R) + G_c(G) + B_c(B)$

RGB Color Space

FIGURE 6.7

Schematic of the RGB color cube. Points along the main diagonal have gray values, from black at the origin to white at point $(1, 1, 1)$.



CMY and CMYB Space

- $[C\ M\ Y] = [1\ 1\ 1] - [R\ G\ B]$
- Cyan, Magenta and Yellow
- But mixing these three colors can not produce black!

The Luminance of a Color

- The luminance of a color with coordinates (R,G,B) in the CIE-RGB system is given

$$L(C)=0.176R+0.81G+0.011B$$

Problems with RGB

- Can only a small range of all the colors humans are capable of perceiving (particularly for monitor RGB)
 - Have you ever seen magenta on a monitor?
- It isn't easy for humans to say how much of RGB to use to make a given color
 - How much R, G and B is there in “brown”?
(Answer: .64,.16, .16)
- **Perceptually non-linear**
 - Two points a certain distance apart in one part of the space may be perceptually different
 - Two other points, the same distance apart in another part of the space, may be perceptually the same

Image Fidelity Criteria

- Subjective measures
 - Examination by human subjects
 - Goodness scale: excellent, good, poor, unsatisfactory
 - Impairment scale: unnoticeable, just noticeable, ...
 - Comparative measures
 - With another image or among a group of images
- Objective measures
 - Mean square error and variations
 - Advantage: simple, less dependent on human subjects, & easy to handle mathematically
 - Disadvantage: not always reflect human perception.

Mean-square Criterion

- Average (or sum) of squared difference of pixel luminance between two images

$$\varepsilon_1 = E\{|u - u'|^2\} \quad (\text{mean square error})$$

$$\varepsilon_2 = \frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N |u(m,n) - u'(m,n)|^2 \quad (\text{average square error})$$

$$\varepsilon_3 = \frac{1}{MN} \sum_{m=1}^M \sum_{n=1}^N E\{|u(m,n) - u'(m,n)|^2\} \quad (\text{average mean square error})$$

- Signal-to-noise ratio (SNR)
 - $\text{SNR} = 10 \log_{10}(\sigma_s^2 / \sigma_e^2)$ in unit of decibel (dB)
 - σ_s^2 is image variance, σ_e^2 variance of error
 - $\text{PSNR} = 10 \log_{10}(A^2 / \sigma_e^2)$ with A being peak-to-peak value

Structure Similarity Index Measure (SSIM) [2]

- For image quality assessment, it is useful to apply the SSIM index *locally* rather than globally.
 - Image statistical features are usually highly spatially **non-stationary**.
 - Image distortions may also be **space-variant**.
 - Because of the fovea feature of the HVS, at typical viewing distances, only **a local area in the image can be perceived with high resolution** by the human observer at one time instance.
 - Localized quality measurement can provide a spatially varying quality map of the image, which delivers more information about the quality degradation of the image and may be useful in some applications.

Structure Similarity Index Measure (SSIM) [2]

- In this paper, the authors use an 11×11 circular-symmetric Gaussian weighting function $\mathbf{w} = \{w_i | i = 1, 2, \dots, N\}$, with standard deviation of 1.5 samples, normalized to unit sum ($\sum_{i=1}^N w_i = 1$)

$$\mu_x = \sum_{i=1}^N w_i x_i$$

$$\sigma_x = \left(\sum_{i=1}^N w_i (x_i - \mu_x)^2 \right)^{\frac{1}{2}}$$

$$\sigma_{xy} = \sum_{i=1}^N w_i (x_i - \mu_x)(y_i - \mu_y)$$

Structure Similarity Index Measure (SSIM) [2]

- The basic idea of SSIM is to separate the task of similarity measurement into three comparisons: *luminance*, *contrast* and *structure*

- The luminance comparison function

$$l(x, y) = \frac{2\mu_x\mu_y + k_1}{\mu_x^2 + \mu_y^2 + k_1}$$

- The contrast comparison function

$$c(x, y) = \frac{2\sigma_x\sigma_y + k_2}{\sigma_x^2 + \sigma_y^2 + k_2}$$

- The structure similarity

$$s(x, y) = \frac{\sigma_{xy} + k_3}{\sigma_x\sigma_y + k_3}$$

- The estimation is using a local weighted window, e.g., a Gaussian window, $\text{SSIM}(x, y) = l(x, y) \cdot c(x, y) \cdot s(x, y)$

References

- [1] <http://web.mit.edu/abyrne/www/ColorRealism.html>
- [2] Z. Wang, A.C. Bovik, H.R. Sheikh, and E. P. Simoncelli, “Image quality assessment: from error visibility to structural similarity,” IEEE Trans. On Image Processing, 13(4):600-612, 2004.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Halftone (I)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

What is digital halftoning?

- Halftoning is a method for creating the *illusion* of continuous tone output with a binary device or a low resolution device.
- Effective digital halftoning can substantially improve the quality of rendered images at minimal cost.

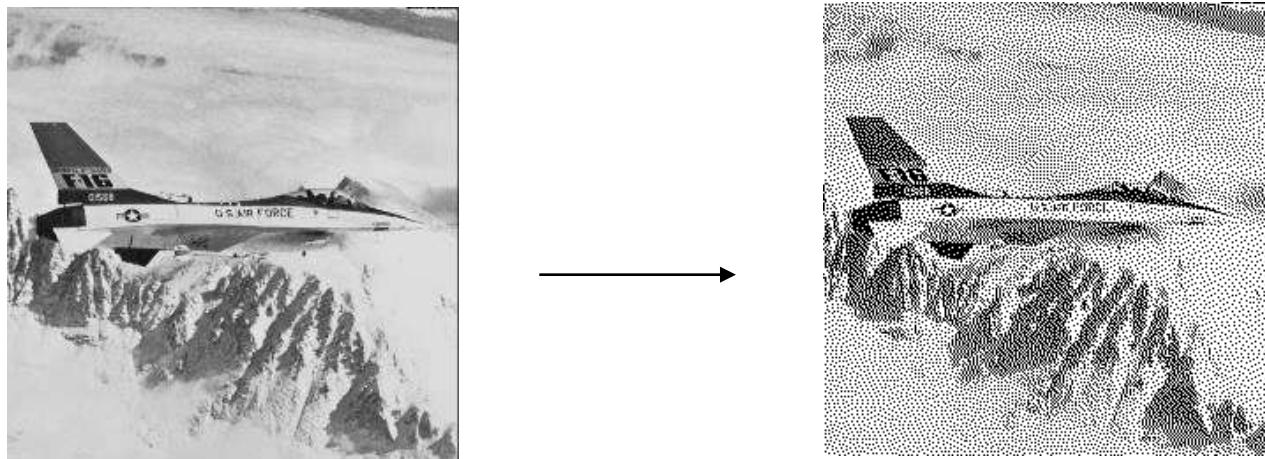


Fig.1 An example for halftoning process

Need for Digital Image Halftoning

- Examples of reduced grayscale/color resolution
 - Laser and inkjet printers
 - Facsimile machines
 - Low-cost liquid crystal displays
- Halftoning is word-length reduction for images
 - Grayscale: 8-bit to 1-bit (**binary**)
 - Color displays: 24-bit RGB to 12-bit RGB (**e.g. PDA/cell**)
 - Color displays: 24-bit RGB to 8-bit RGB (**e.g. cell phones**)
 - Color printers: 24-bit RGB to CMYK (**each color binarized**)
- Halftoning tries to reproduce full range of gray/color while preserving quality & spatial resolution

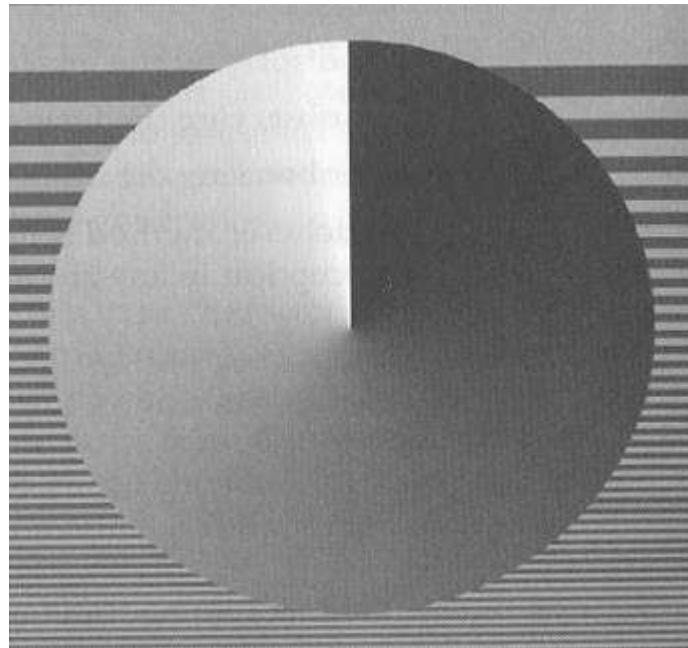
Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

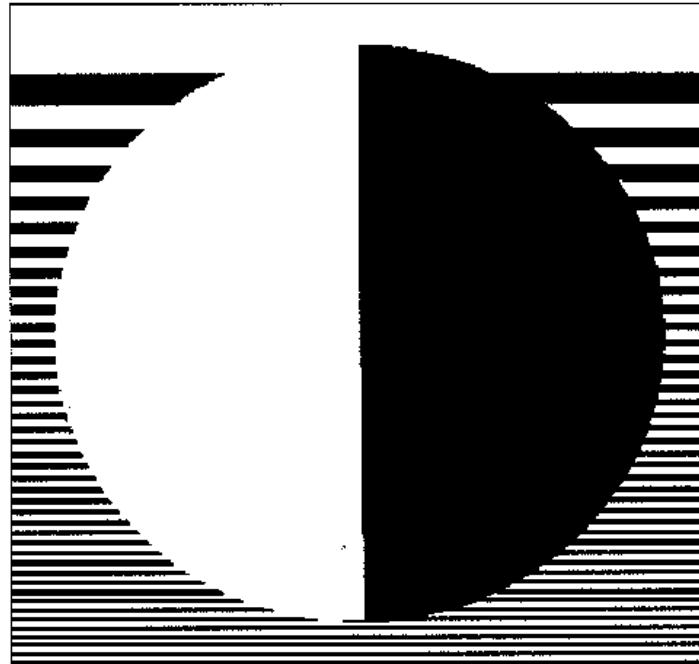
Sample Images

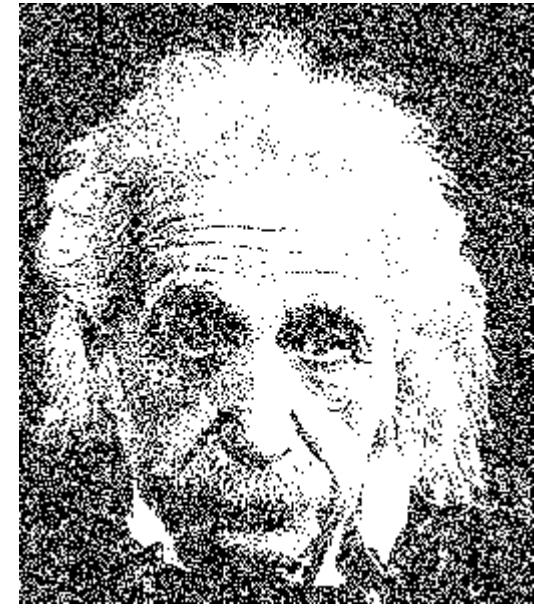
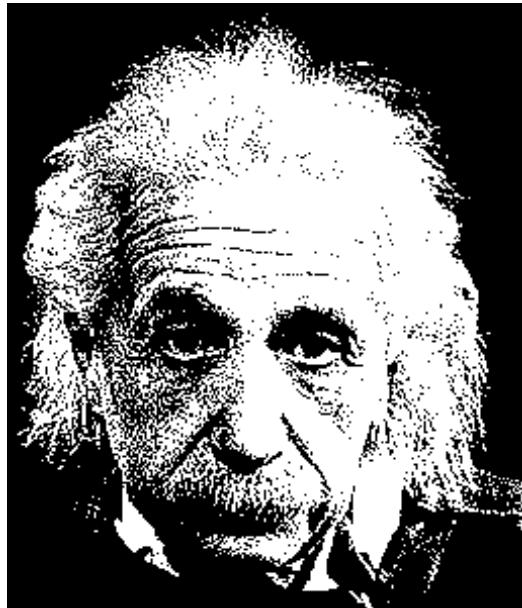
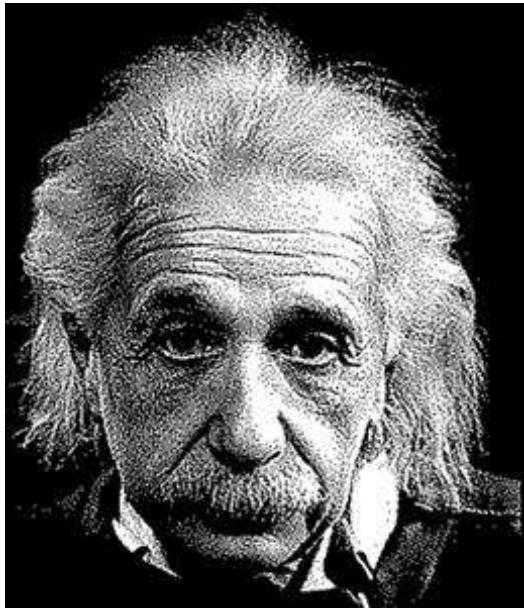


Threshold Dithering

- For every pixel: If the intensity < 0.5 , replace with black, otherwise replace with white
 - 0.5 is the threshold
 - This is the naïve version of the algorithm
- To keep the overall image brightness the same, you should:
 - Compute the average intensity over the image
 - Use a threshold that gives that average
 - For example, if the average intensity is 0.6, use a threshold that is higher than 40% of the pixels, and lower than the remaining 60%. (For a halftone of a constant gray image, the gray level value specifies the proportional of black dots over a white background no matter how the dots are to be arranged.)
- For all dithering we will assume that the image is gray and that intensities are represented as a value in $[0, 1.0]$
 - If you have a 0-255 image, you can scale all the thresholds (multiply by 255)

Naïve Threshold Algorithm





Original Gray Images

Threshold = average gray
value

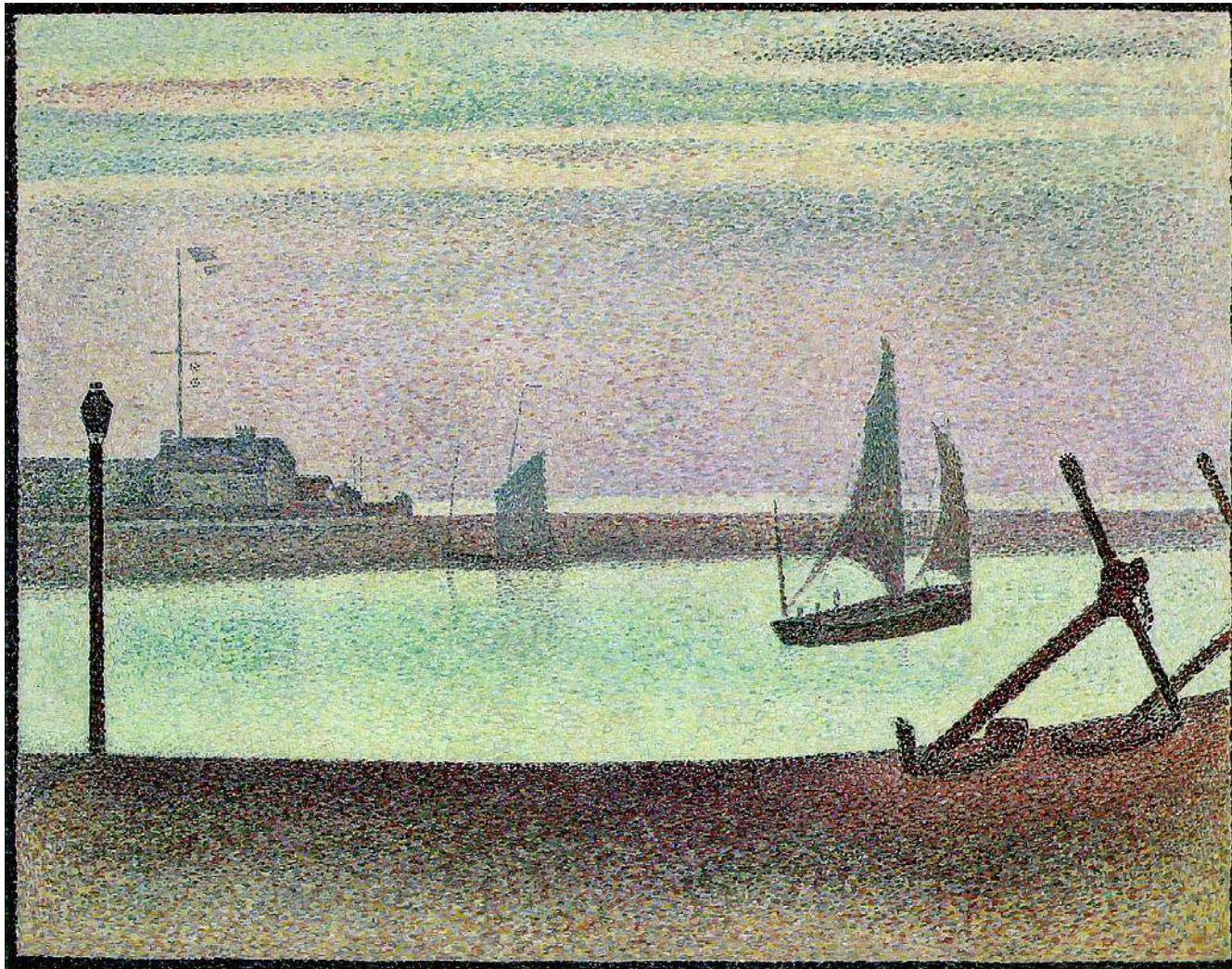
Tone-preserving halftoned

Pointillism (点画法)

- Several artists over the last hundred years have utilized halftone in paintings by using **stippling**.
- In these paintings, artists put dots of primary or complimentary colors on the canvas to achieve a certain color blend when viewing the painting at a distance.
- This approach, which is known as *pointillism*, contrasts with the conventional approach of mixing colors on a palette before applying the paint using **sweeping and fine strokes**.
- Pointillism was a French movement of the late 1800s that was an offshoot of impressionism.

“The Channel of Gravelines, Evening”

(1890) by Georges-Pierre Seurat



Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

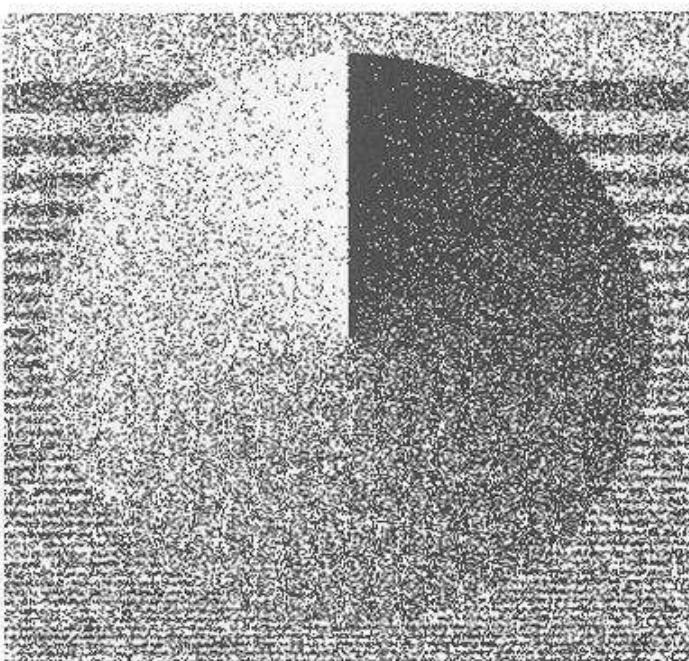
Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

Random Modulation

- Add a random amount to each pixel *before* thresholding
 - Typically add *uniformly* random amount from $[-a, a]$
- Pure addition of noise to the image
 - For better results, add better quality noise
 - For instance, use Gaussian noise (random values sampled from a normal distribution)
- Should use same procedure as before for choosing threshold
- Not good for black and white, but **OK** for more colors
 - Add a small random color to each pixel before finding the closest color in the table

Random Modulation



References

- [1] B. E. Bayer, “An optimum method for two-level rendition of continuous-tone pictures,” in Proceedings of the IEEE International Conference on Communication, pp. 11-26, 1973.
- [2] R. Floyd and L. Steinberg, “An adaptive algorithm for spatial grey scale,” Society for Information Display Symposium, Digest of Technical Papers, pp.36-37, 1975.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Halftone (II)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

Ordered Dithering [1]

- Ordered dithering is *a point process* that produces output by comparing a single continuous-tone input value to a deterministic **periodic array of thresholds (dither matrix)** [4].
- For example, a 2×2 dither matrix D_2 :

$$D_2 = \begin{bmatrix} 0 & 2 \\ 3 & 1 \end{bmatrix}$$

- The dither matrix can be converted to a “*threshold matrix*” or “screen” using the following operation.

$$T(i, j) = 255 \times \frac{D(i, j) + 0.5}{N^2}$$

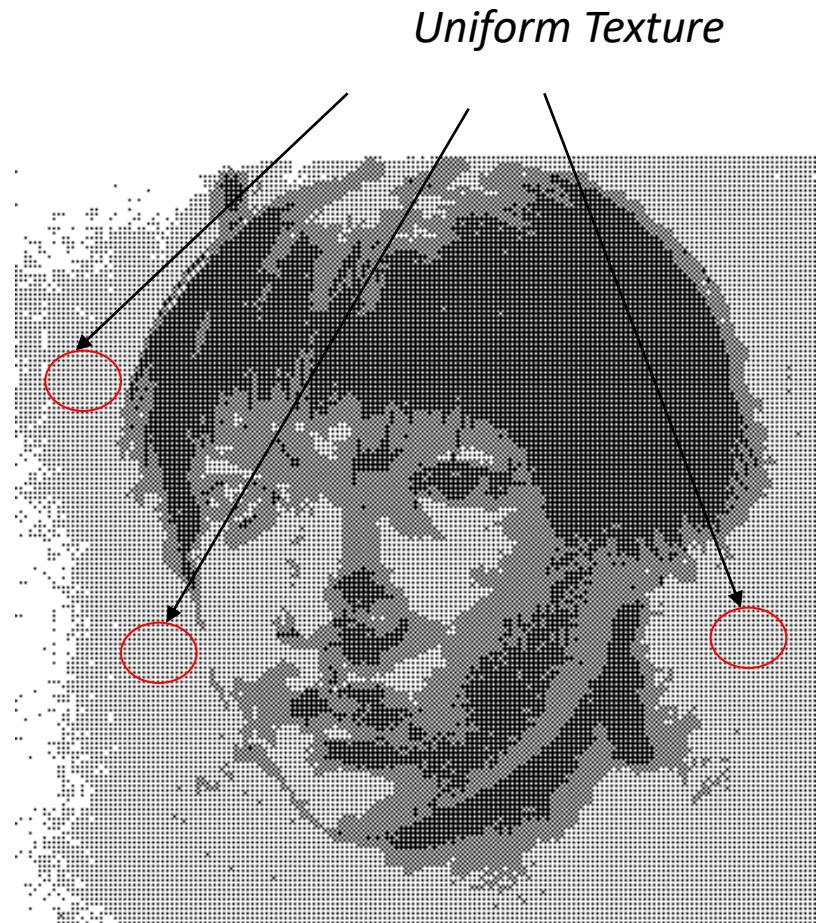
- The ordered dithering algorithm is applied via thresholding. X is an original grayscale image.

$$b(i, j) = \begin{cases} 255 & \text{if } X(i, j) > T(i, j) \\ 0 & \text{otherwise} \end{cases}$$

Example for Ordered Dithering



Fig.1 (a) Original grayscale image



(b) Halftone image dithered with D_2

Dither Matrix

- The size of the matrix and the arrangement of the values have an important effect on the dither process.
- Two common dither matrix patterns: **clustered pattern** and **dispersed pattern**.
 - **Clustered** pattern: if the consecutive thresholds are located in spatial proximity, then it is called a “clustered pattern”.
 - **Dispersed** pattern: the thresholds are uniformly distributed in the matrix.

Two 8×8 Dither Matrix Patterns

62	57	48	36	37	49	58	63
56	47	35	21	22	38	50	59
46	34	20	10	11	23	39	51
33	19	9	3	0	4	12	24
32	18	8	2	1	5	13	25
45	31	17	7	6	14	26	40
55	44	30	16	15	27	41	52
61	54	43	29	28	42	53	60

(a)

0	32	8	40	2	34	10	42
48	16	56	24	50	18	58	26
12	44	4	36	14	46	6	38
60	28	52	20	62	30	54	22
3	35	11	43	1	33	9	41
51	19	59	27	49	17	57	25
15	47	7	39	13	45	5	37
63	31	55	23	61	29	53	21

(b)

Fig.2 Examples for two dither matrix patterns (a)
Clustered pattern (b) **Dispersed** pattern

Properties of Clustered Pattern

- Relatively visible texture
- Relatively poor detail rendition
- Uniform texture across entire grayscale.
- Robust performance with non-ideal output devices

Dispersed Dithering

- *Bayer's optimal dither matrix* [1, Bayer, 1973]

$$D_{2n} = \begin{bmatrix} 4 \times D_n(i, j) & 4 \times D_n(i, j) + 2 \\ 4 \times D_n(i, j) + 3 & 4 \times D_n(i, j) + 1 \end{bmatrix}$$

- 4×4 and 8×8 *Bayer's optimal dither matrixes*

0	8	2	10
12	4	14	6
3	11	1	9
15	7	13	5

0	32	8	40	2	34	10	42
48	16	56	24	50	18	58	26
12	44	4	36	14	46	6	38
60	28	52	20	62	30	54	22
3	35	11	43	1	33	9	41
51	19	59	27	49	17	57	25
15	47	7	39	13	45	5	37
63	31	55	23	61	29	53	21

Fig.3 Bayer' optimal dither matrix

Example for Dispersed Dithering with 8×8 *Bayer's Optimal Dither Matrix*



(a)



(b)

Fig. 4(a) Original grayscale image (b) Halftone image dithered with
8 × 8 Bayer's optimal dither matrix

Properties of Dispersed Dithering

- Within any region containing K dots, the K thresholds should be distributed as uniformly as possible.
- Textures used to represent individual gray levels have low visibility.
- Improved detail rendition.
- Transitions between textures corresponding to different gray levels may be more visible.
- Not robust to non-ideal output devices

Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

Halftone Methods

- Threshold Dithering
- Random Modulation
- Ordered Dithering [1, Bayer, 1973]
 - Cluster dot screen
 - Disperse dot screen
- Error Diffusion [2, Floyd and Steinberg, 1975]

Error Diffusion Halftoning [2]

- Quantizes each pixel using a **neighborhood** operation, rather than a simple pointwise operation.
- Moves through image in a scan curve, quantizing the result, and “*pushing*” the error forward.
- Can produce better quality images than is possible with screens.
- Variations: which neighbor pixels are affected?

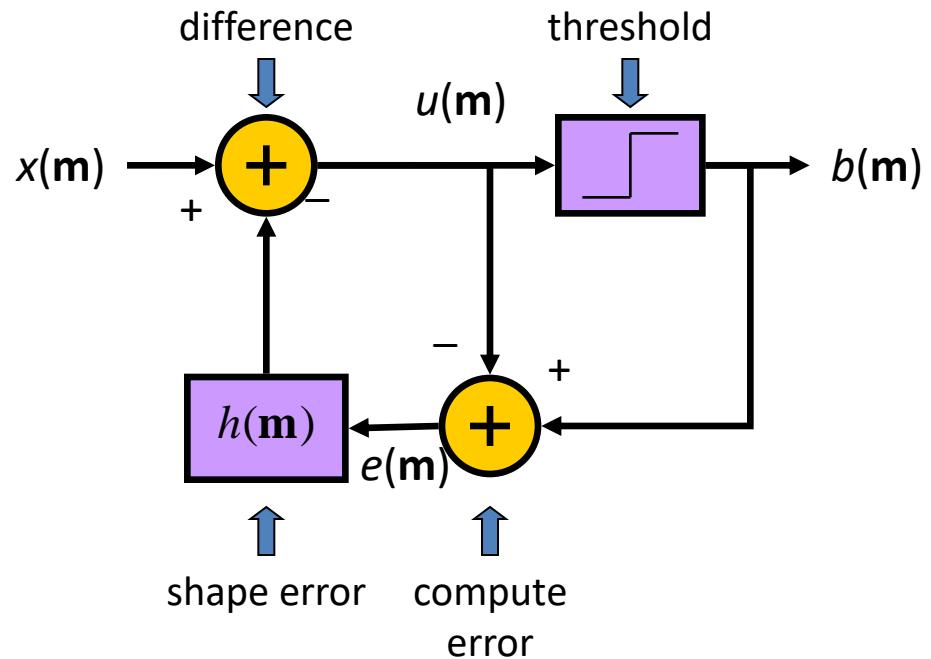
Filter View of Error Diffusion

Equations:

$$u_{m,n} = x_{m,n} - \sum_{(k,l) \in R} h_{k,l} e_{m-k,n-l}$$

$$b_{m,n} = Q(u_{m,n}) = \begin{cases} \lambda & u_{m,n} \geq \lambda / 2 \\ 0 & u_{m,n} < \lambda / 2 \end{cases}$$

$$e_{m,n} = b_{m,n} - u_{m,n} = Q(u_{m,n}) - u_{m,n}$$



Parameters:

- ♠ Threshold is typically $\lambda/2 = 127$.
- ♠ $h_{k,l}$ are typically chosen to be positive and sum to 1.

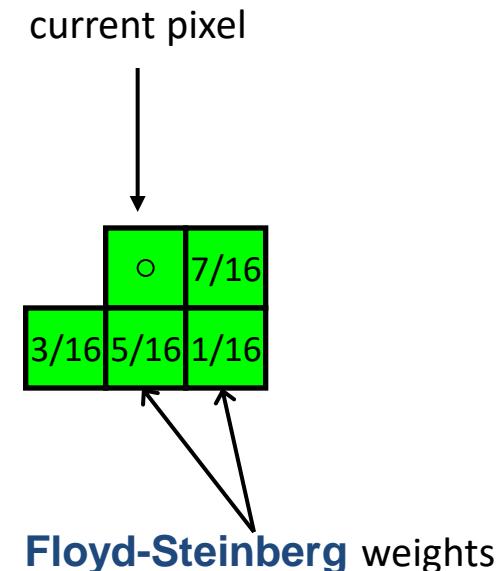
Error Diffusion Algorithm

- 1. Initialize $u_{m,n}$ with $x_{m,n}$, $e_{m,n}$ and $b_{m,n}$ with zeros.
- 2. For each pixel in the image (in scan curve)
 - (a) Compute

$$b_{m,n} = Q(u_{m,n}) = \begin{cases} \lambda & u_{m,n} \geq \lambda/2 \\ 0 & u_{m,n} < \lambda/2 \end{cases}$$

- (b) **Diffuse** $e_{m,n}$ forward, such as the following scheme
(Floyd-Steinberg filter)

- 3. Display the binary image $b_{m,n}$.



Variation: Filter

		*	7	5
3	5	7	5	3
1	3	5	3	1

(a)

		*	8	4
2	4	8	4	2
1	2	4	2	1

(b)

Fig.5 Variation: filter (a) Jarvis filter (48) (b) Stucki filter (42)

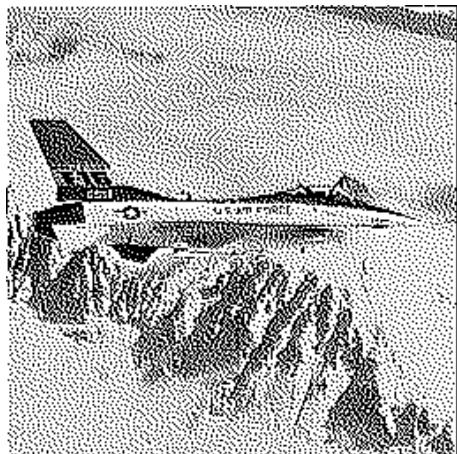
Examples for Different Error Diffusion Filters



(a)



(b)



(c)



(d)

Fig.6 (a) Original grayscale image “Plane” (b) (c) (d) halftone images generated by Floyd-Steinberg, Jarvis and Stucki filters.

References

- [1] B. E. Bayer, “An optimum method for two-level rendition of continuous-tone pictures,” in Proceedings of the IEEE International Conference on Communication, pp. 11-26, 1973.
- [2] R. Floyd and L. Steinberg, “An adaptive algorithm for spatial grey scale,” Society for Information Display Symposium, Digest of Technical Papers, pp.36-37, 1975.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Noise and Filtering (I)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Noise

- Impulse noise
- Gaussian white noise (thermal noise)
- Texture noise (spatial correlated noise)

Impulse Noise

- **Salt-Pepper Impulsive Noise** — the noise value will be either the **maximum** value or the **minimum** value of the image gray level, each with the equal probability.
- **Uniform Impulsive Noise** — the noise value is an **uniform random variable** between the maximum value and the minimum value of the image gray level.

$$x_{ij} = \begin{cases} o_{ij}; & \text{with probability } 1 - p_n \\ n_{ij}; & \text{with probability } p_n \end{cases}$$

The only input parameter is p_n .

Impulse Noise Examples



a) 3% Salt-Pepper impulse noise

```
J = imnoise(I, 'salt & pepper', 0.03);
```



b) 3% Uniform impulse noise

Gaussian Noise Examples



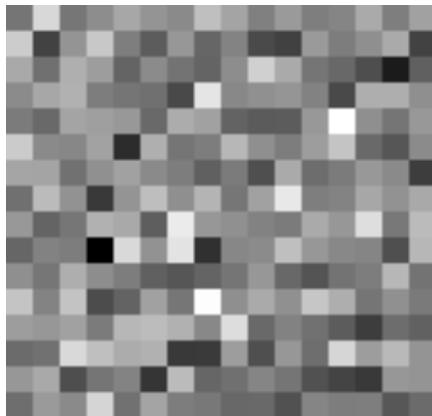
a) Original image



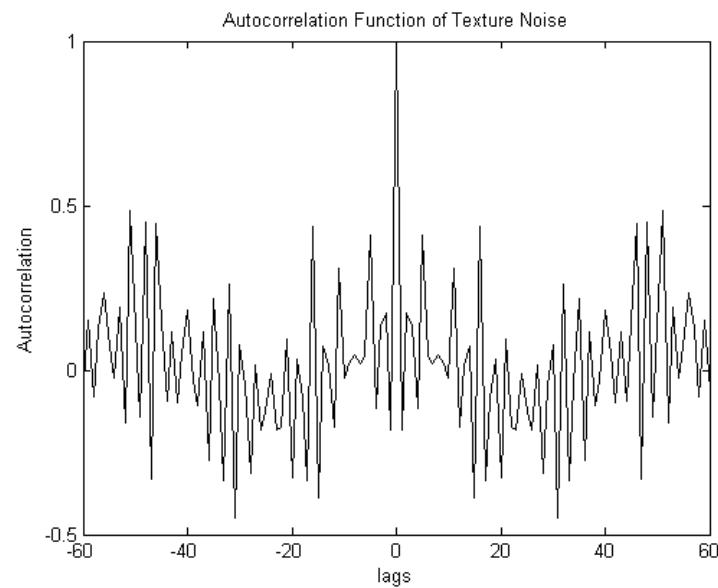
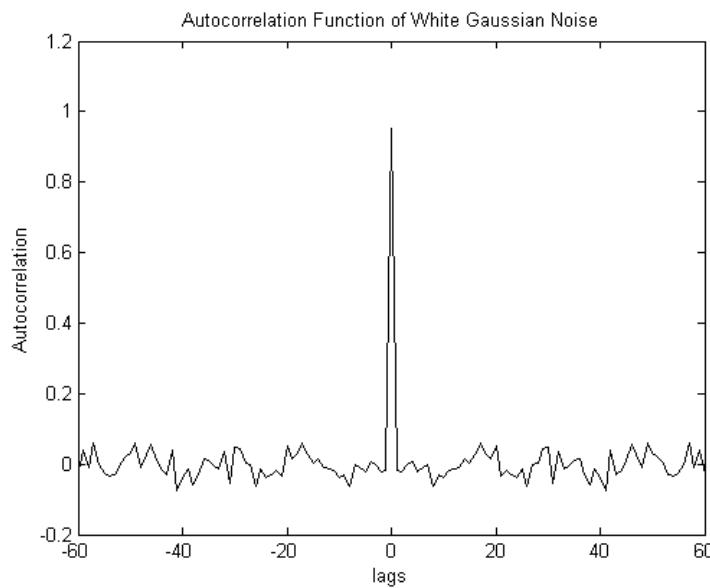
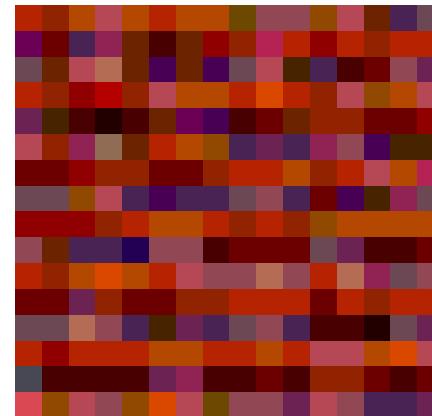
b) Gaussian white noise (mean = 0, std = 10)

Texture Noise [1]

White Gaussian Noise



Color Texture Noise



ISNR (the Improvement in SNR)

- We usually use the ISNR (the Improvement in SNR) to determine the quality of the filtered image.

$$ISNR = 10 \cdot \log_{10} \left\{ \frac{\sum_{i,j} [f(i,j) - y(i,j)]^2}{\sum_{i,j} [f(i,j) - \hat{f}(i,j)]^2} \right\}$$

where $f(i, j)$ and $y(i, j)$ denote the original and the noise image, respectively, and $\hat{f}(i, j)$ denotes the filtered image.

Impulse Noise Examples



a) 3% Salt-Pepper impulse noise



b) 3% Uniform impulse noise

Median Filtering Results

- 3×3 Median filter on the two above image corrupted by impulse noises.



a) Median filtering of 3% Salt-pepper
impulse noise



b) Median filtering of 3% Uniform
impulse noise

Median Filtering Results (Cont.)

- 3×3 Threshold Median filter on the two above image corrupted by impulse noises ($T=35$)



a) Median filtering of 3% Salt-pepper
impulse noise



b) Median filtering of 3% Uniform
impulse noise

References

- [1] X.-Q. Lu and H. Sakaino, “A spatial adaptive filter for smoothing of non-Gaussian texture noise” in Proc. of ICASSP, 2009.
- [2] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in Proc. Of the IEEE International Conference on Computer Vision, pp.839-846, 1998

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Noise and Filtering (II)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

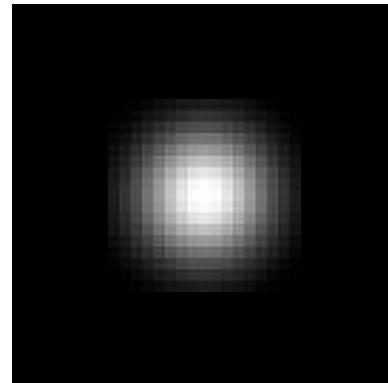
Bilateral Filter [2]

- Bilateral filtering smoothes images while preserving edges, by means of a **nonlinear combination** of nearby image values.

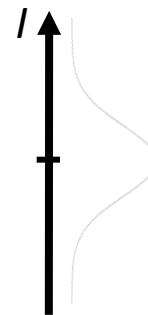
$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\| p - q \|) G_{\sigma_r}(|I_p - I_q|) I_q$$

normalization
factor

space weight

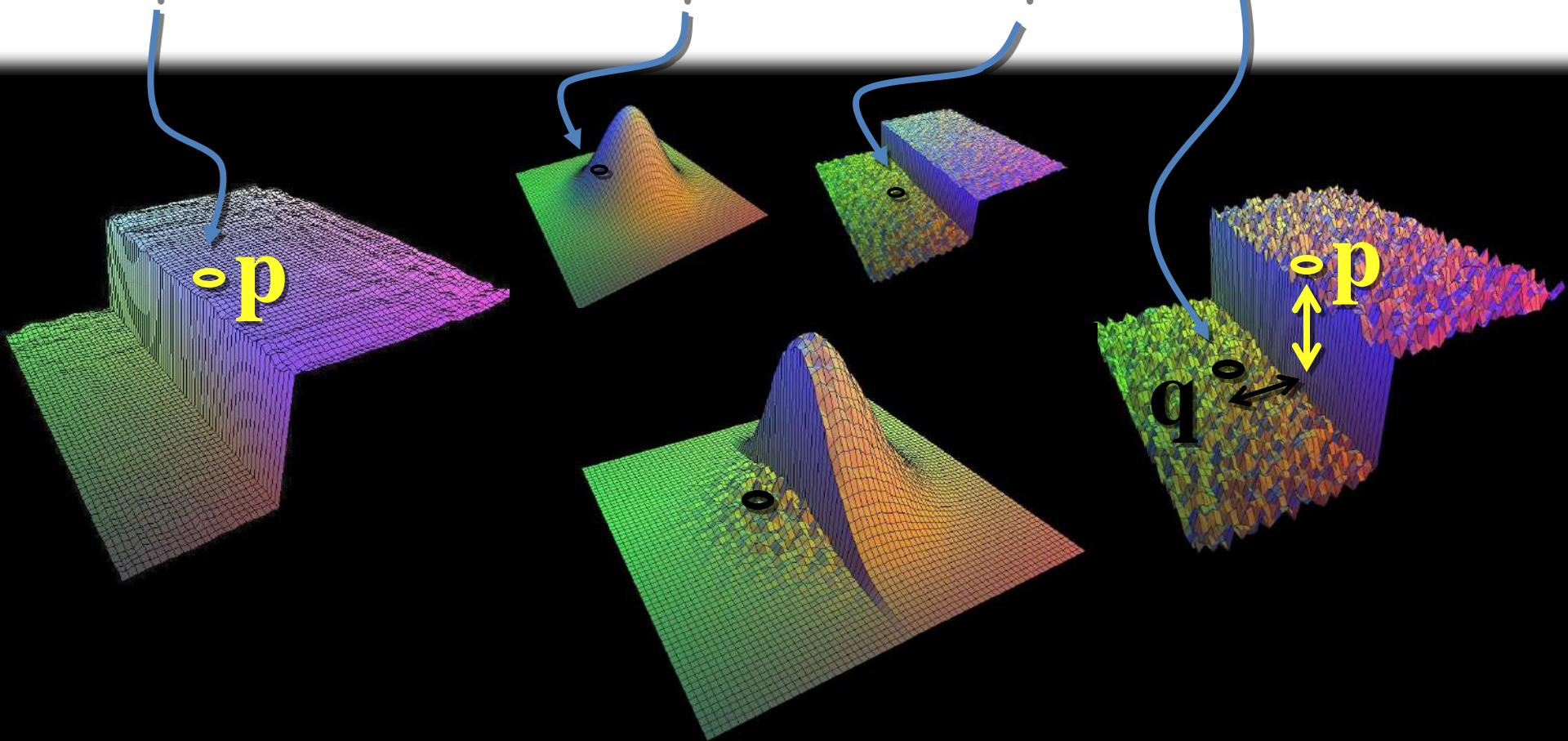


range weight



Bilateral Filter on a Height Field

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(|I_p - I_q|) I_q$$



Bilateral Filter for Gray Image



Noise image (Gaussian Noise, mean =0, std = 5)



Bilateral Filtering (ISNR= 1.4211 dB, 9×9, $\sigma_s = 5$, $\sigma_r = 10$, Time = 1.1909s)

Failed Example



Noise image (Gaussian Noise, mean = 0, std = 20)



Bilateral Filtering (ISNR= 1.0637 dB, 9×9, $\sigma_s = 5$, $\sigma_r = 10$, Time = 1.2673s)

Failed Example



Noise image (Gaussian Noise, mean = 0, std = 20)



Bilateral Filtering (ISNR= 4.0883 dB, 9×9, $\sigma_s = 5$, $\sigma_r = 50$, Time = 1.0722s)

Bilateral Filter for Color Image



Noise image (Gaussian Noise, mean =0, std = 10)



Bilateral Filtering (ISNR= 3.8447 dB, 9×9, $\sigma_s = 5$, $\sigma_r = 10$, Time = 3.2966s)

More Examples for Color Images



(a) Input Image

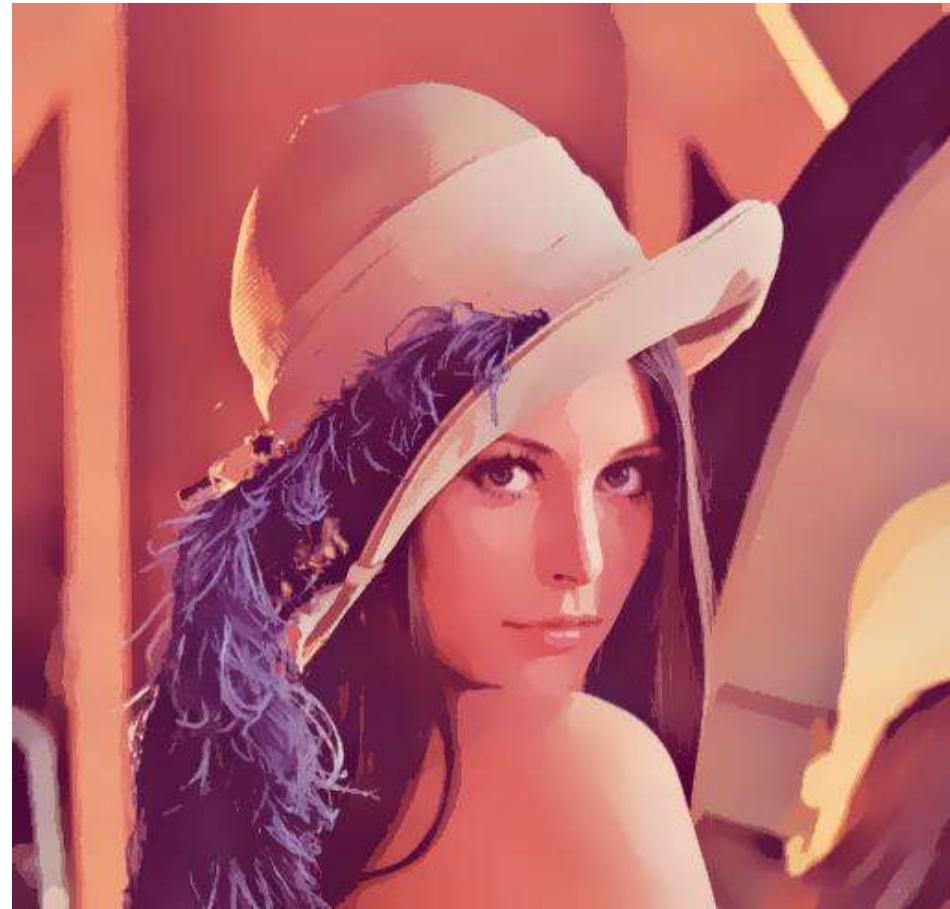


(b) Bilateral Filtered Image (iter = 5, w = 5, $\sigma_s = 5$, $\sigma_r = 3$)

More Examples for Color Images



(b) Bilateral Filtered Image (iter = 5, w
 $= 5, \sigma_s = 5, \sigma_r = 3$)



(c) Bilateral Filtered Image (iter = 5, w
 $= 5, \sigma_s = 5, \sigma_r = 10$)

More Examples for Color Images



(c) Bilateral Filtered Image (iter = 5, w = 5, σ_s = 5, σ_r = 10)



(d) Bilateral Filtered Image (iter = 10, w = 5, σ_s = 5, σ_r = 3)

Failed Example for Texture Noise



(a) Input Image



(b) Bilateral Filtered Image (iter = 5, w
 $= 5, \sigma_s = 5, \sigma_r = 3$)

References

- [1] X.-Q. Lu and H. Sakaino, “A spatial adaptive filter for smoothing of non-Gaussian texture noise” in Proc. of ICASSP, 2009.
- [2] C. Tomasi and R. Manduchi, “Bilateral filtering for gray and color images,” in Proc. Of the IEEE International Conference on Computer Vision, pp.839-846, 1998.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Noise and Filtering (III)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Nonlinear Total Variation Filtering [1]

- The constrained Minimization Problem

$$\min \int_{\Omega} \sqrt{u_x^2 + u_y^2} dx dy$$

- Subject to constraints involving u

Conservation of Energy:

$$u_0 = u + n$$

$$\int_{\Omega} u dx dy = \int_{\Omega} u_0 dx dy$$

$$\int_{\Omega} \frac{1}{2} (u - u_0)^2 dx dy = \sigma^2$$

Euler-Lagrange Equation

$$\int_{\Omega} \{(u_x^2 + u_y^2)^{1/2} + \lambda_1(u - u_0) + \lambda_2 \frac{1}{2}((u - u_0)^2 - \sigma^2)\} dx dy$$

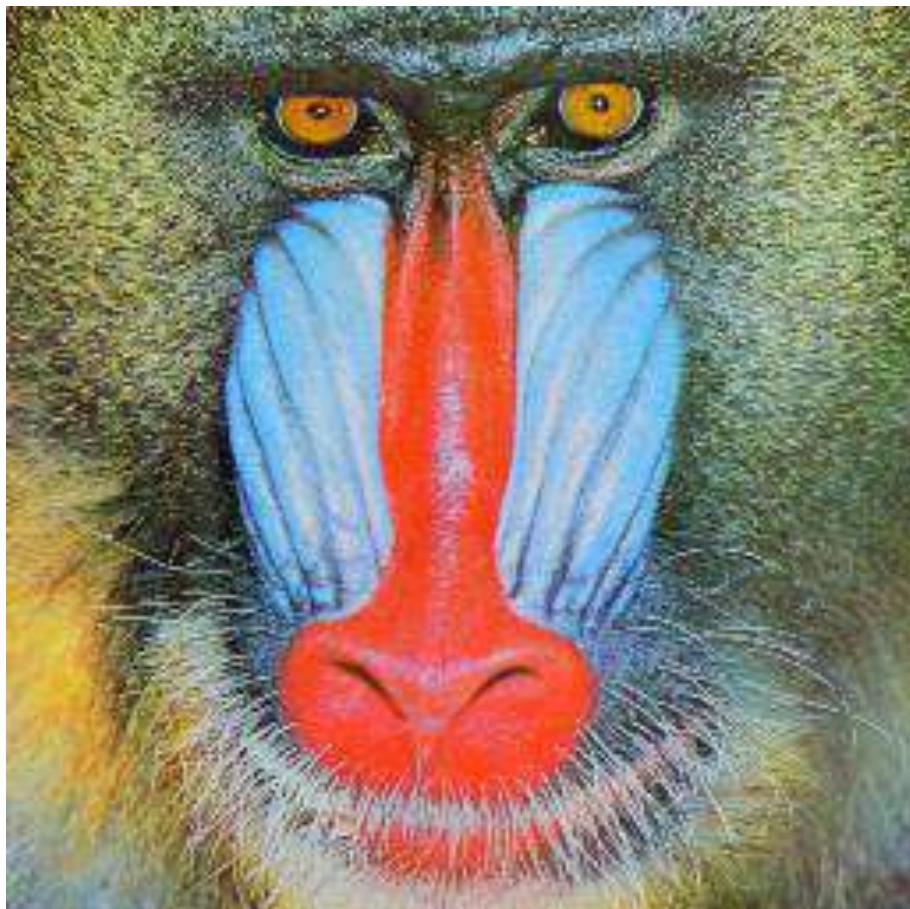
$$\frac{\partial}{\partial x} \left(\frac{u_x}{\sqrt{u_x^2 + u_y^2}} \right) + \frac{\partial}{\partial y} \left(\frac{u_y}{\sqrt{u_x^2 + u_y^2}} \right) - \lambda_1 - \lambda_2 (u - u_0) = 0$$

- The gradient descent algorithm

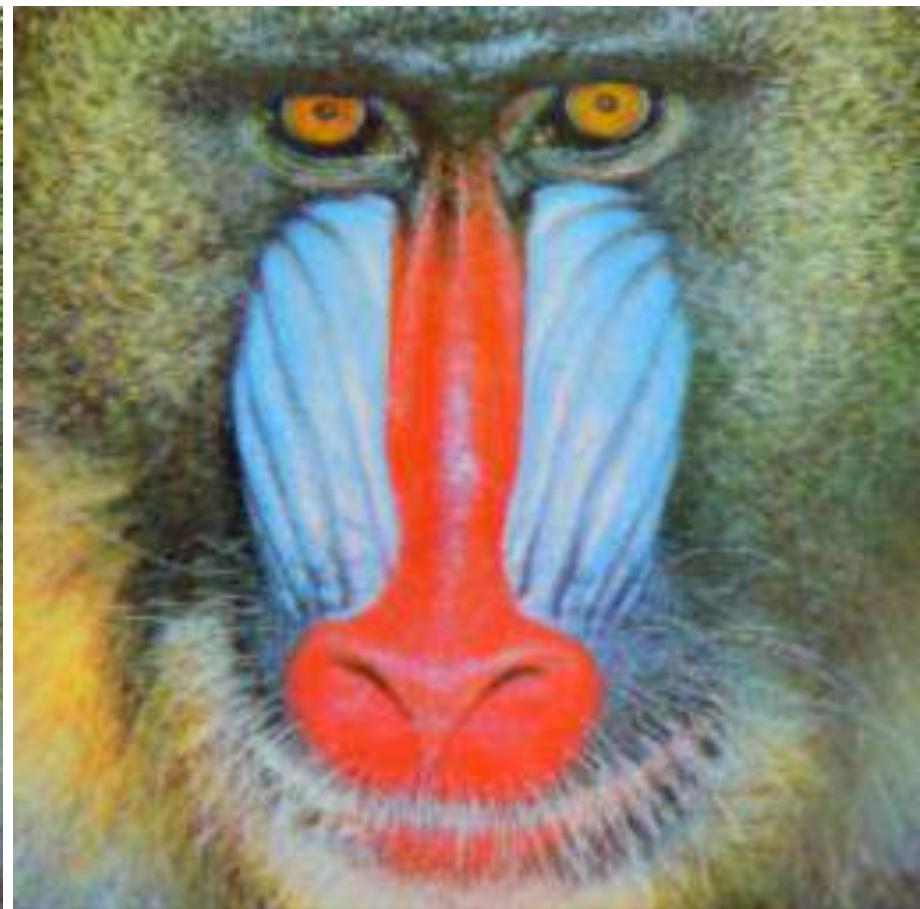
$$u^{(i+1)} = u^{(i)} + dt \left(\frac{u_{xx}}{\sqrt{u_x^2 + u_y^2}} + \frac{u_{yy}}{\sqrt{u_x^2 + u_y^2}} + \lambda(u^{(0)} - u^{(i)}) \right)$$

Started with $u^{(0)} = u_0$

Experimental Results (I)

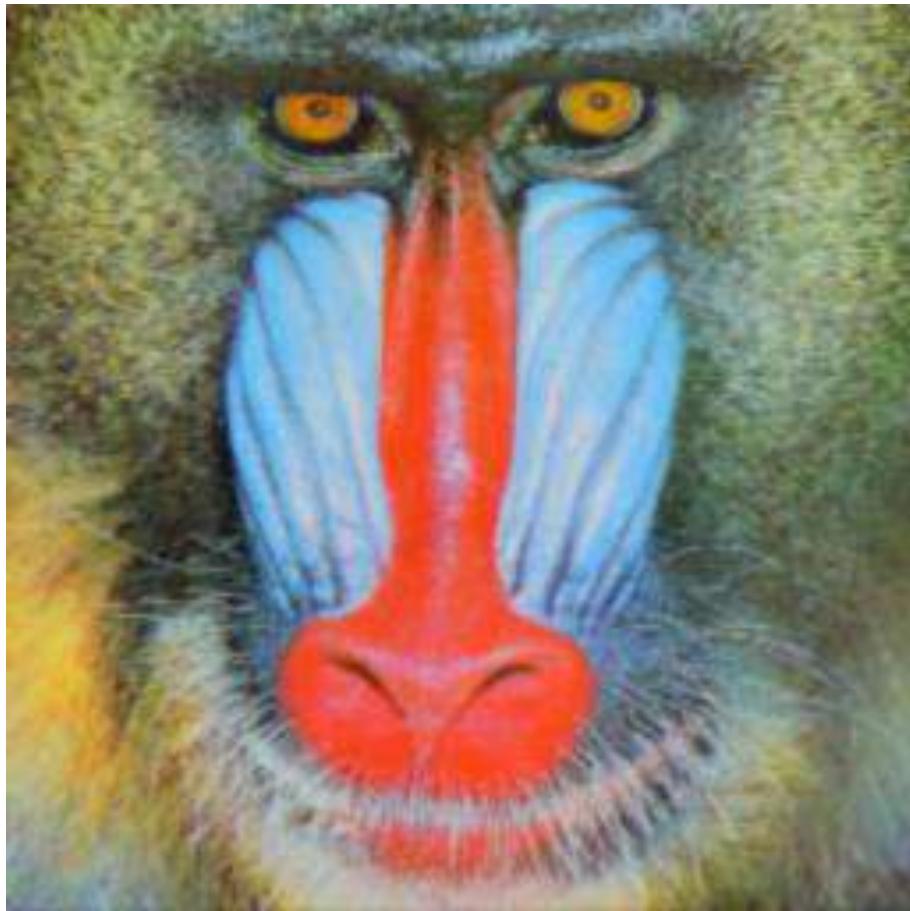


(a) The input image



(b) The filtered image (iters = 20, dt = 0.01, λ = 0.5)

Experimental Results (II)



(b) The filtered image (iters = 20, dt = 0.01, λ = 0.5)



(c) The filtered image (iters = 500, dt = 0.01, λ = 0.5)

Nonlinear Total Variation [4]



(a) Lena (Gaussian noise mean = 0 std = 10)
Iteration number = 100, ISNR = 3.1698 dB



(b) Lena (Gaussian noise mean = 0 std = 20)
ISNR = 7.9477 dB

References

- [1] L. Rudin, S. Osher and E. Fatemi, “Nonlinear total variation based noise removal,” Physical D, 60:259-268, 1992.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Noise and Filtering (IV)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Bilateral Texture Filtering [1]

- This is a modification of the original bilateral filter (Tomasi & Manduchi, 1998) , it performs *local patch-based* analysis of texture features and incorporates its results into the range filter kernel.
 - It incorporates texture information (instead of color information) into the range filter kernel.
- The central idea to ensure proper texture/structure separation is based on **patch shift** that captures the texture information from the most representative texture patch clear of prominent structure edges.
 - Texture often contains strong enough contrast to get confused with structure.

Patch Shift

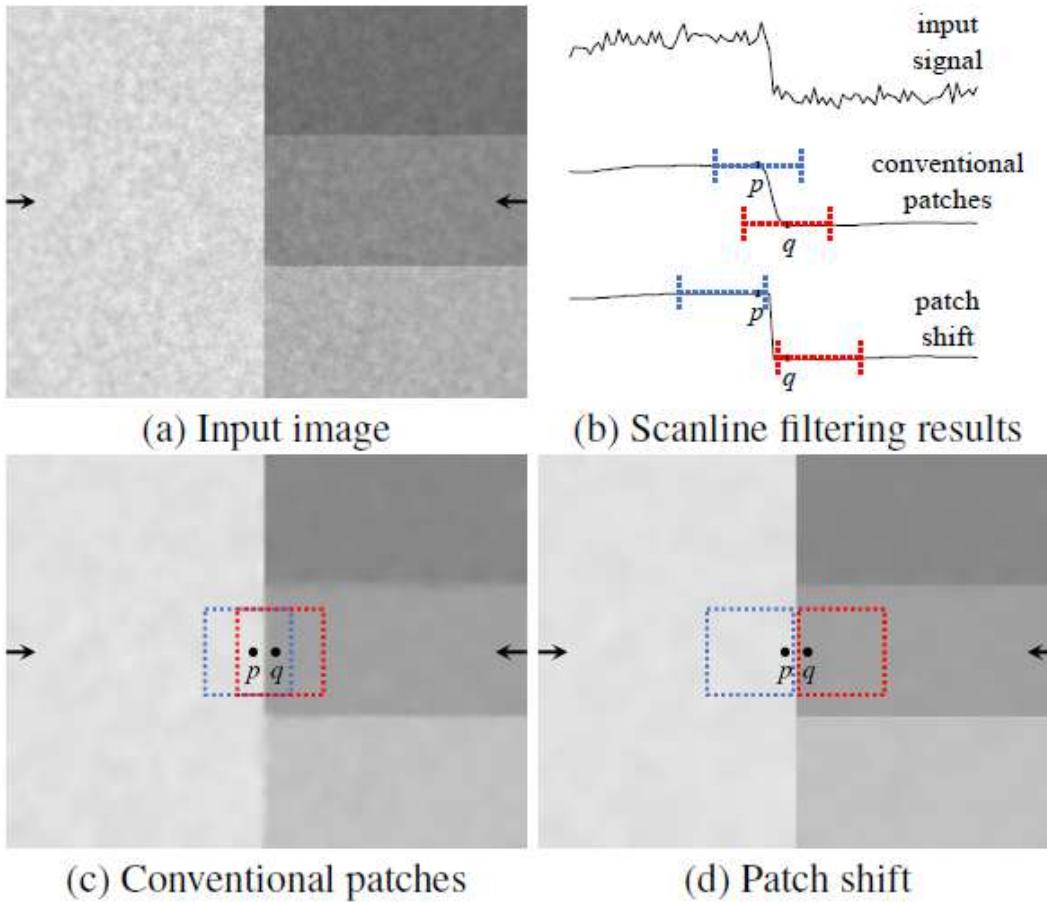
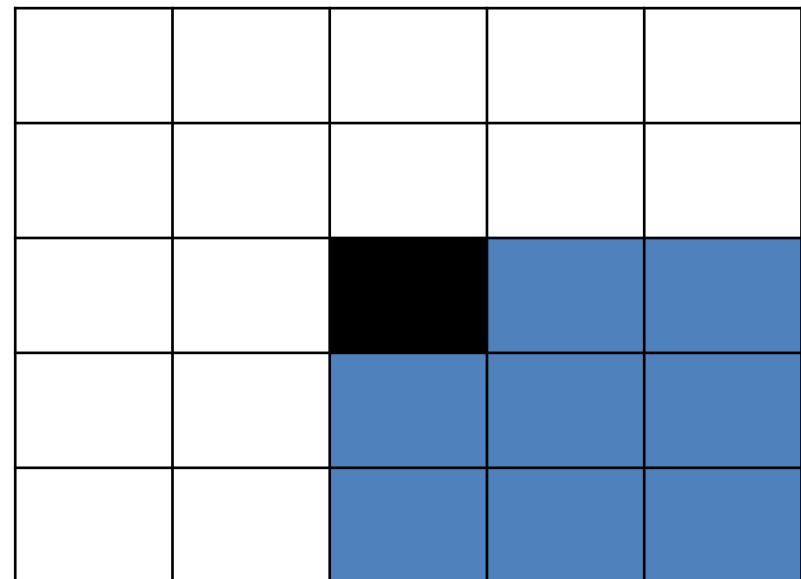
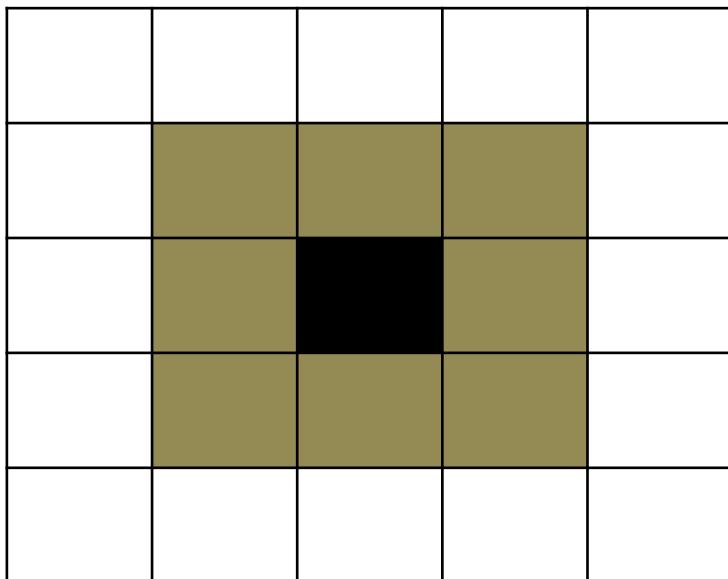


Figure 2: *Patch shift.* Conventionally, texture feature is computed in a patch centered at each pixel, in which case the patches for two adjacent pixels should have a large overlap, reducing the feature discriminability. In contrast, patch shift finds a nearby patch that stays clear of a prominent structure edge. (b) Filtering of the scanline marked by arrows. (c) Filtered by [Karacan et al. 2013]. (d) Filtered with patch shift. The results in (b) show that our approach preserves structure edges, unlike the conventional approach.

Patch Shift

- Assuming a $k \times k$ box representing a patch, each pixel p has a total of k^2 patches in I and contains p .



Bilateral Texture Filter

- The bilateral filter of Tomasi and Manduchi (1998)

$$BF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(\|I_p - I_q\|) I_q$$

- The bilateral texture filter

$$BTF[I]_p = \frac{1}{W_p} \sum_{q \in S} G_{\sigma_s}(\|p - q\|) G_{\sigma_r}(\|T_p - T_q\|) I_q$$



T: the guidance image based on
local texture measure

Salient Structure Measure

- Assuming a $k \times k$ box representing a patch, each pixel p has a total of k^2 patches in the input image that contains p .
- Let us *assume* for the moment that texture signal has **smaller amplitude** than the neighboring structure edge.
- We define texture as **fine-scale spatial oscillations** of signals.
- To measure the likelihood of containing **structure edge** for a patch Ω_q via its **tonal range** $\Delta(\Omega_q)$:

$$\Delta(\Omega_q) = I_{\max}(\Omega_q) - I_{\min}(\Omega_q)$$

where $I_{\max}(\Omega_q)$ and $I_{\min}(\Omega_q)$ denote the maximum and the minimum image intensities in patch Ω_q , respectively.

The Guidance Image

- 1) Given an input image I , we first apply $k \times k$ box kernel to compute the average image B .
- 2) For each pixel \mathbf{p} , we compute the tonal range $\Delta(\Omega_q)$. We then obtain **the guidance image** T via patch shift on each pixel. That is, we find the patch Ω_q whose $\Delta(\Omega_q)$ is *the minimum among k^2 candidates*, then copy $B_{\mathbf{q}}$ to $T_{\mathbf{p}}$.
- 3) Finally we obtain the output image J by applying joint bilateral filter on I , using T as the guidance image.

Modification of the Guidance Image (I)

- The tonal range suggests that patch shift *may not* work properly if the tonal range within a **pure texture region** is **as large as the nearby structure edge**.
- Modified Relative Total Variation ($mRTV$)

$$mRTV(\Omega_q) = \Delta(\Omega_q) \frac{\max_{\mathbf{r} \in \Omega_q} |(\partial I)_{\mathbf{r}}|}{\sum_{\mathbf{r} \in \Omega_q} |(\partial I)_{\mathbf{r}}| + \varepsilon} \quad (4)$$

- The $mRTV$ value is relatively **large** in a **structure** patch containing only a few edges, and relatively **small** in a **texture** patch having frequent oscillations.

Modification of the Guidance Image (II)

- The $mRTV$ values in a **smooth or flat image region** tend to be very small and thus may become sensitive to image noise.
 - Small noisy peaks can be mistaken for edges
- When copying B_q to T_p , if the two $mRTV$ values of Ω_p and Ω_q are similar, B_p is preferred over B_q as the value of T_p ; if and only if $mRTV(\Omega_q)$ is *considerably smaller* than $mRTV(\Omega_p)$, B_q is used for T_p .

$$T'_p = \alpha_p T_p + (1 - \alpha_p) B_p \quad (5)$$

$$\alpha_p = 2 \left(\frac{1}{1 + \exp(-\sigma_\alpha (mRTV(\Omega_p) - mRTV(\Omega_q)))} - 0.5 \right) \quad (6)$$

Bilateral Texture Filtering

Algorithm 1 Bilateral texture filtering

Input: image I

Output: texture filtered image J

```
for  $iter = 1 : n_{itr}$  do
     $B \leftarrow$  Uniform blurring of  $I$ 
    mRTV  $\leftarrow$  Compute Eq. (4) for each pixel  $p$ 
    for all  $p \in I$  do
        Find  $q \in \Omega_p$  with minimum  $mRTV_q$             $\triangleright$  patch shift
         $G_p \leftarrow B_q$ 
    end for
     $\alpha \leftarrow$  Compute Eq. (6) for each pixel  $p$ 
     $G' \leftarrow \alpha G + (1 - \alpha)B$                     $\triangleright$  Eq. (5)
     $J \leftarrow$  joint bilateral filtering of  $I$  using  $G'$  as guidance
     $I \leftarrow J$                                       $\triangleright$  input for the next iteration
end for
```

Experimental Results (I)



Input Images

Bilateral Texture Filtered Images

Experimental Results (II)



Input Images



Bilateral Texture Filtered Images



References

- [1] H. Cho, H. Lee, H. Kang, and S. Lee, "Bilateral texture filtering," ACM Transactions on Graphics, 33(4):1-8, 2014.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Interpolation & Superresolution (I)

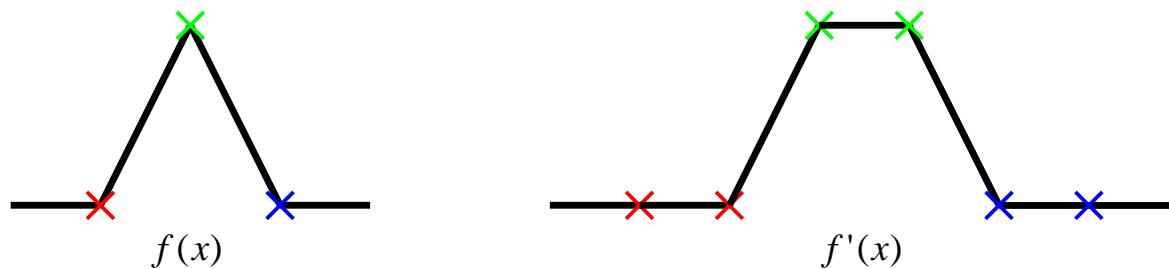
Dr. Xigun Lu

College of Computer Science

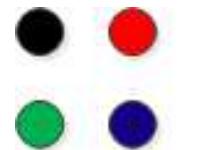
Zhejiang University

Pixel Replication

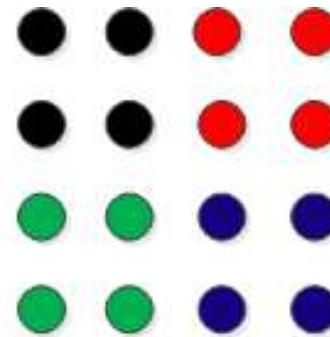
- To increase the number of pixels in an image, but without adding any data or detail.
- 1D signal



- 2D image



input



output

Interpolated Results (I)

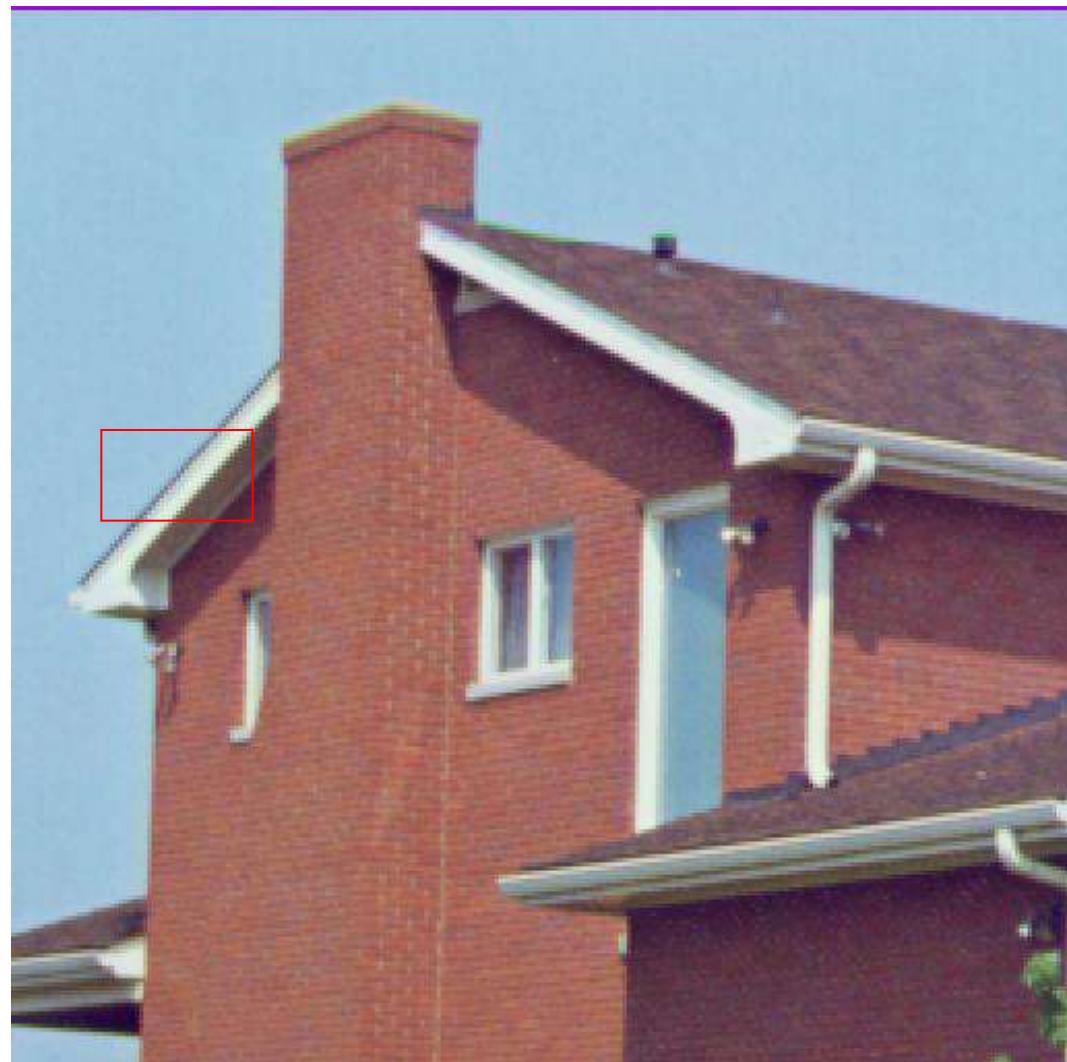


Input image 256×256



Output high-resolution image 512×512

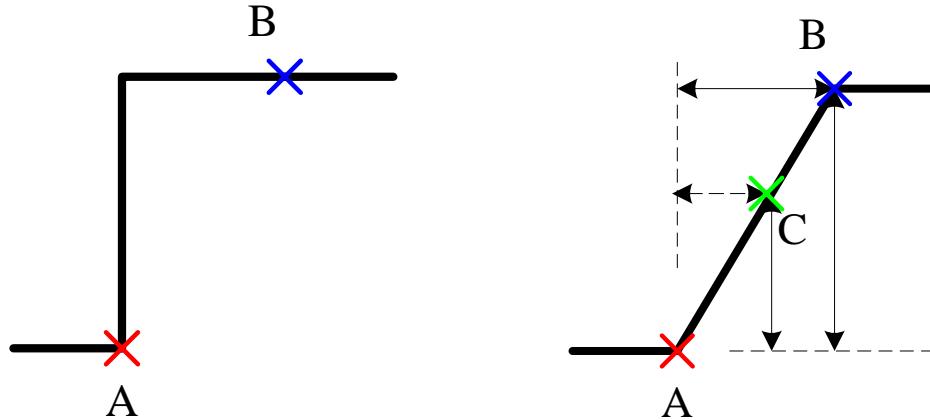
Interpolated Results (II)



Output high-resolution image 512×512

Bilinear Interpolation

- To use distance-weighted average of the some nearest pixel values to estimate a new pixel value.



The interpolated value at C is

$$f(x_0 + d) = \frac{f(x_1) - f(x_0)}{x_1 - x_0} \cdot d + f(x_0)$$

Interpolated Results (I)



Input image 256×256

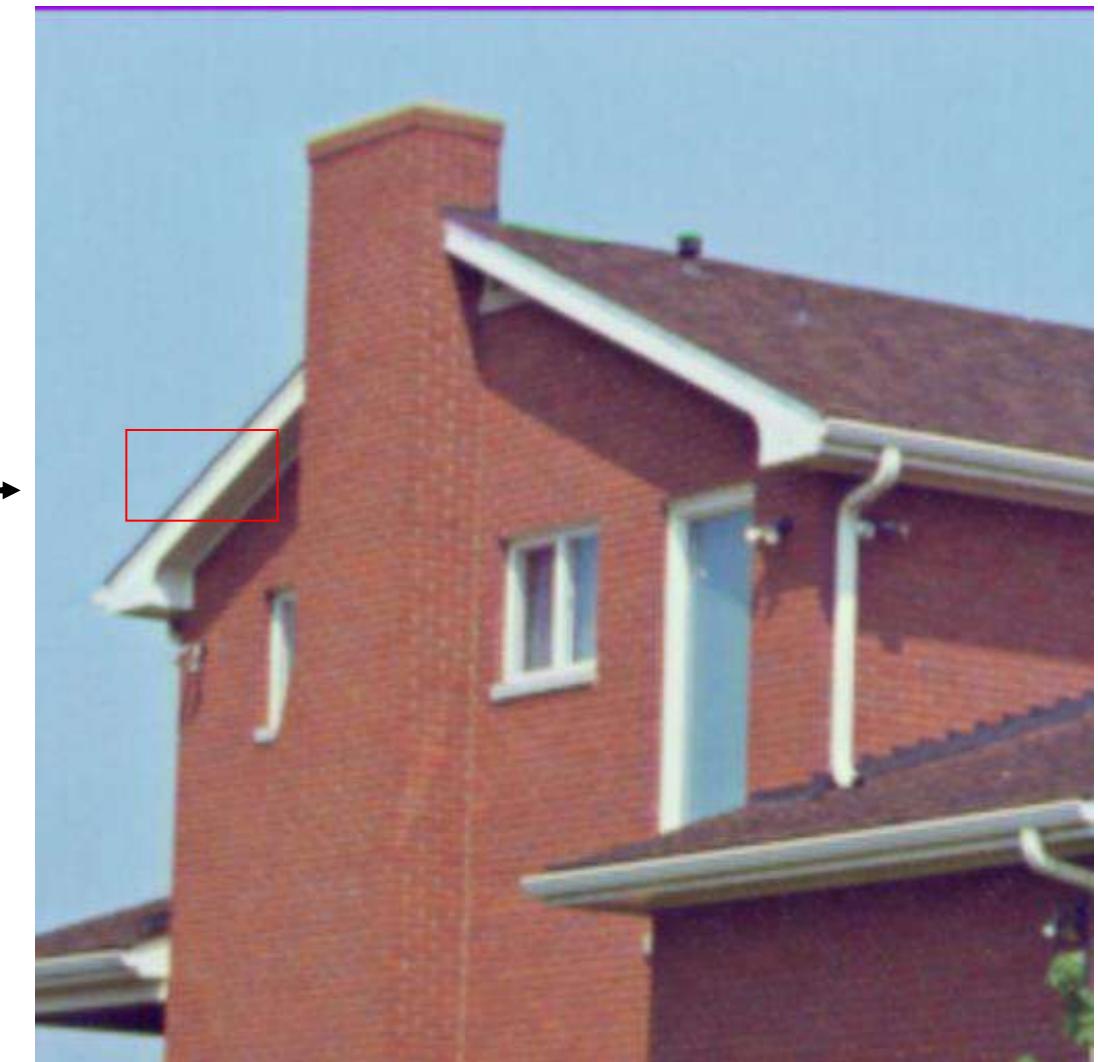


Output high-resolution image 512×512

Interpolated Results (II)



Input image 256×256



Output high-resolution image 512×512

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Interpolation & Superresolution (II)

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Bicubic Convolution Interpolation [1]

- An interpolation function is a special type of approximating function. A fundamental property of interpolation functions is that they must *coincide* with the sampled data at their interpolation nodes.

$$g(x_k) = f(x_k)$$

- For *equally spaced* data, many interpolation functions can be written in the form

$$g(x) = \sum_k c_k \cdot u\left(\frac{x - x_k}{h}\right)$$

One of the options in “imresize” — “cubic” was implemented according to this paper.

where h represents the sampling increment

x_k are the interpolation nodes

c_k are parameters which depend on the sampled data

u is the interpolation kernel(basis function)

g is the interpolation function

Cubic Convolution Interpolation Kernel

- The cubic convolution interpolation kernel is composed of **piecewise cubic polynomials** defined on the subinterval $(-2, -1)$, $(-1, 0)$, $(0, 1)$ and $(1, 2)$. Outside of the interval $(-2, 2)$, the interpolation kernel is *zero*.
- The interpolation kernel must be **symmetric**:

$$u(s) = \begin{cases} A_1|s|^3 + B_1|s|^2 + C_1|s| + D_1 & 0 < |s| < 1 \\ A_2|s|^3 + B_2|s|^2 + C_2|s| + D_2 & 1 < |s| < 2 \\ 0 & 2 < |s| \end{cases}$$

- $u(0) = 1$ and $u(n) = 0$ (when n is a nonzero integer)
 - This is because that the difference between the interpolation nodes x_j and x_k is $(j-k)h$, $g(x_j) = \sum c_k u(j-k) \rightarrow$ when $j = k$, $g(x_k) = f(x_k) \rightarrow c_k = f(x_k)$; $j \neq k$, $u(j-k) = 0$.



Cubic Convolution Interpolation Kernel

- The conditions $u(0) = 1$ and $u(1) = u(2) = 0$ provide four equations for these coefficients:

$$1 = u(0) = D_1$$

$$0 = u(1^-) = A_1 + B_1 + C_1 + D_1$$

$$0 = u(1^+) = A_2 + B_2 + C_2 + D_2$$

$$0 = u(2^-) = 8A_2 + 4B_2 + 2C_2 + D_2$$

- Three more equations are obtained from the fact that u' is *continuous* at the nodes 0, 1, and 2:

$$-C_1 = u'(0^-) = u'(0^+) = C_1$$

$$3A_1 + 2B_1 + C_1 = u'(1^-) = u'(1^+) = 3A_2 + 2B_2 + C_2$$

$$0 = u'(2^-) = u'(2^+) = 12A_2 + 4B_2 + C_2$$

Cubic Convolution Interpolation Kernel

- In addition to being 0 or 1 at the interpolation nodes, the interpolation kernel must be ***continuous*** and have ***a continuous first derivative***.
 - But we have only 7 equations for 8 unknown parameters of the interpolation kernel.
- The idea in this paper is to choose A_2 so that the cubic convolution interpolation function and the Taylor series expansion for f agree for as many terms as possible. $\rightarrow A_2 = -0.5$

$$u(s) = \begin{cases} \frac{3}{2}|s|^3 - \frac{5}{2}|s|^2 + 1 & 0 < |s| < 1 \\ -\frac{1}{2}|s|^3 + \frac{5}{2}|s|^2 - 4|s| + 2 & 1 < |s| < 2 \\ 0 & 2 < |s| \end{cases}$$

Interpolated Results (I)

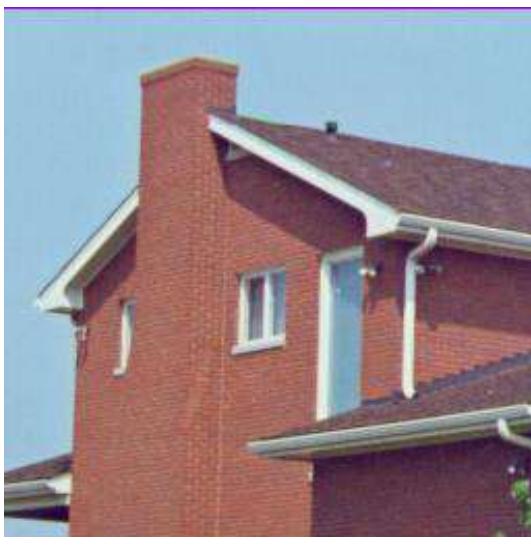


Input image 256×256

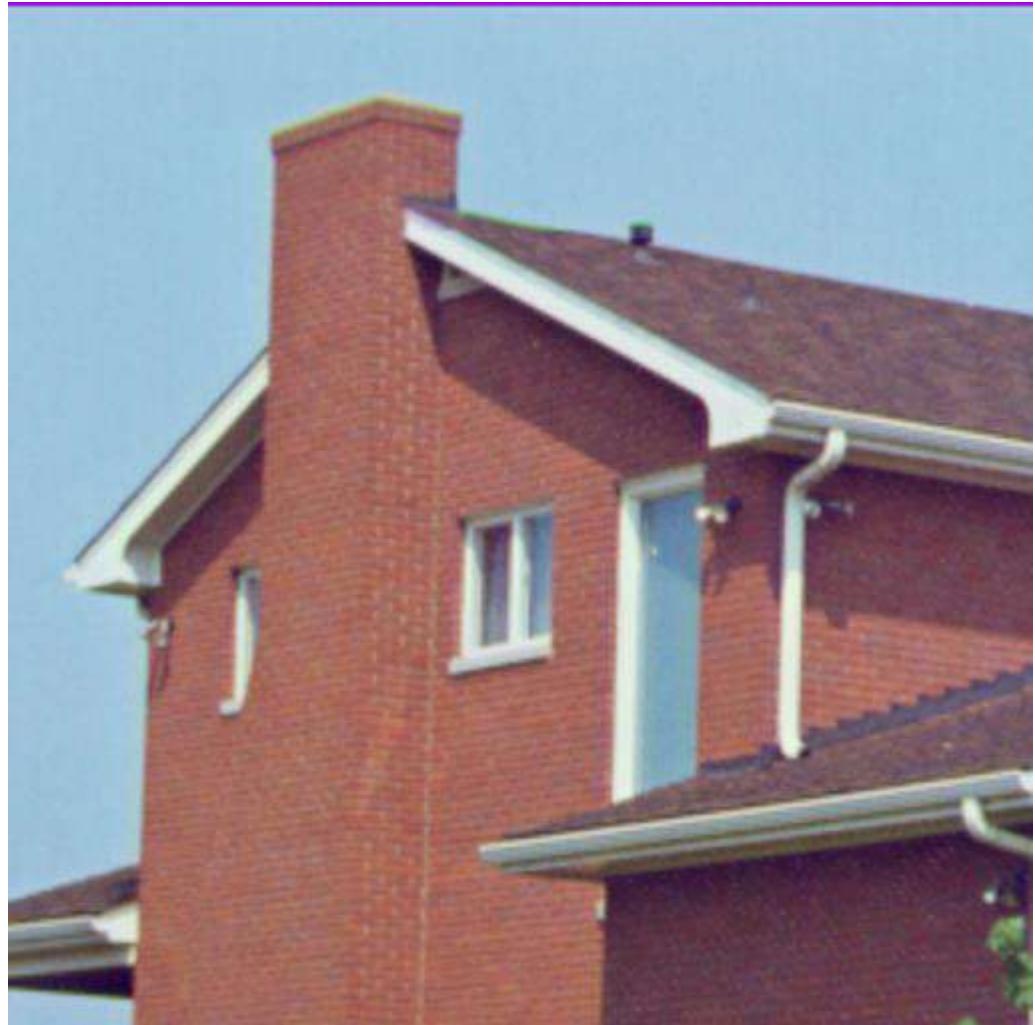


Output high-resolution image 512×512

Interpolated Results (II)



Input image 256×256



Output high-resolution image 512×512

References

- [1] R. G. Keys, “Cubic convolution interpolation for digital image processing,” IEEE Trans. On Acoustics, Speech, and Signal Processing, ASSP-29(6): 1153-1160, 1981.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Interpolation & Superresolution (III)

Dr. Xigun Lu

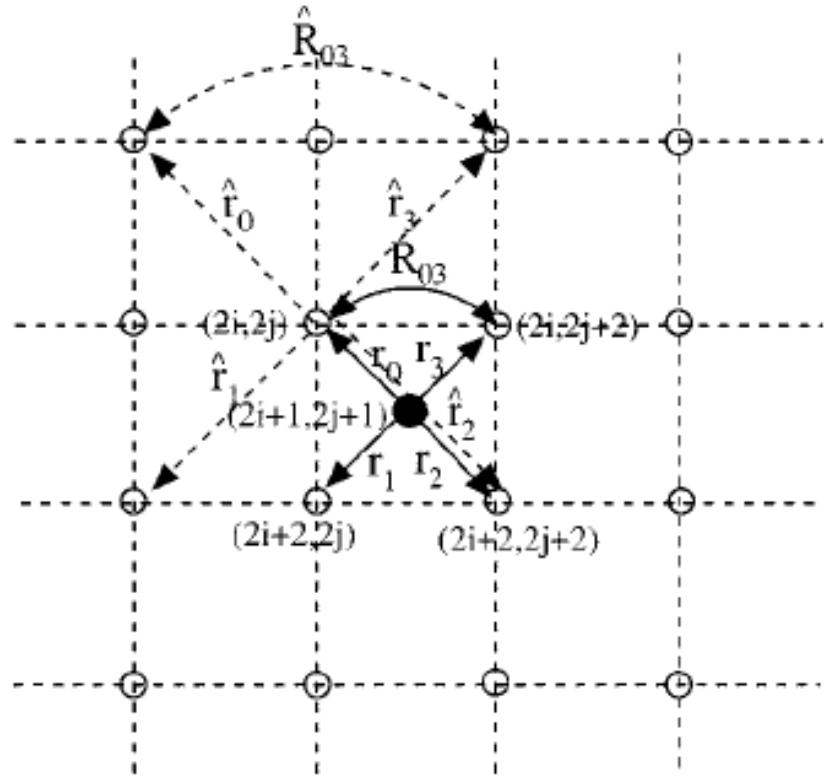
College of Computer Science

Zhejiang University

Edge-Directed Interpolation [2]

- The basic idea is to first estimate local covariance coefficients from a low-resolution image and then use these covariance estimates to adapt the interpolation at a higher resolution based on the *geometric duality* between the low-resolution covariance and the high-resolution covariance.
 - **Assumption:** natural image can be modeled as a *locally Gaussian process*.

Geometry Duality



$$\vec{\alpha} = R^{-1} \vec{r}$$

Fig. 1. Geometric duality when interpolating $\hat{Y}_{2i+1,2j+1}$ from $Y_{2i,2j}$.

$$\hat{Y}_{2i+1,2j+1} = \sum_{k=0}^1 \sum_{l=0}^1 \alpha_{2k+l} Y_{2(i+k), 2(j+l)}$$

Minimum Mean Square Error (MMSE)

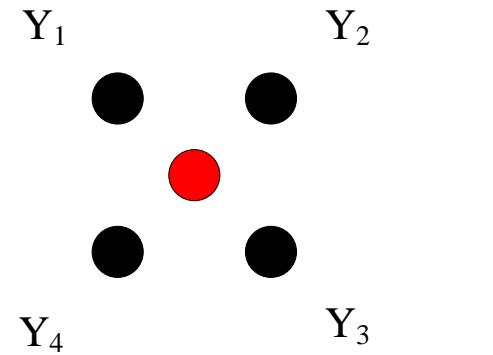
$$E = (\hat{Y} - Y)^2$$

$$= \left(\sum_{i=1}^4 \alpha_i Y_i - Y \right)^2$$

$$\frac{\partial E}{\partial \alpha_i} = (\hat{Y} - Y) \frac{\partial \hat{Y}}{\partial \alpha_i} = (\sum_{k=1}^4 \alpha_k Y_k - Y) Y_i = 0$$

$$\alpha_1 Y_1 Y_i + \alpha_2 Y_2 Y_i + \alpha_3 Y_3 Y_i + \alpha_4 Y_4 Y_i = Y Y_i$$

$$1 \leq i \leq 4$$



But the prediction
just based on only
the 4 neighbors may
be unreliable.

Implementation

- Natural image can be modeled as a **locally Gaussian process**.
- The low-resolution covariance \hat{R}_{kl} , \hat{r}_k can be estimated from a *local window* of the low-resolution image using the classical covariance method:

$$\hat{R} = \frac{1}{M^2} C^T C, \hat{\vec{r}} = \frac{1}{M^2} C^T \vec{y}$$

Where $\vec{y} = [y_1 \quad y_2 \quad \cdots \quad y_{M^2}]^T$ is the data vector containing the $M \times M$ pixels inside the *local* window and C is a $M^2 \times 4$ data matrix whose k th **row** vector is the four nearest neighbors of y_k .

$$\vec{\alpha} = (C^T C)^{-1} (C^T \vec{y})$$

Interpolated Results



Bicubic

Edge-directed interpolation

References

- [2] X. Li and M. T. Orchard, “New edge-directed interpolation,” IEEE Trans. On Image Processing, 10(10): 1521-1527, 2001.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Image Interpolation & Superresolution (IV)

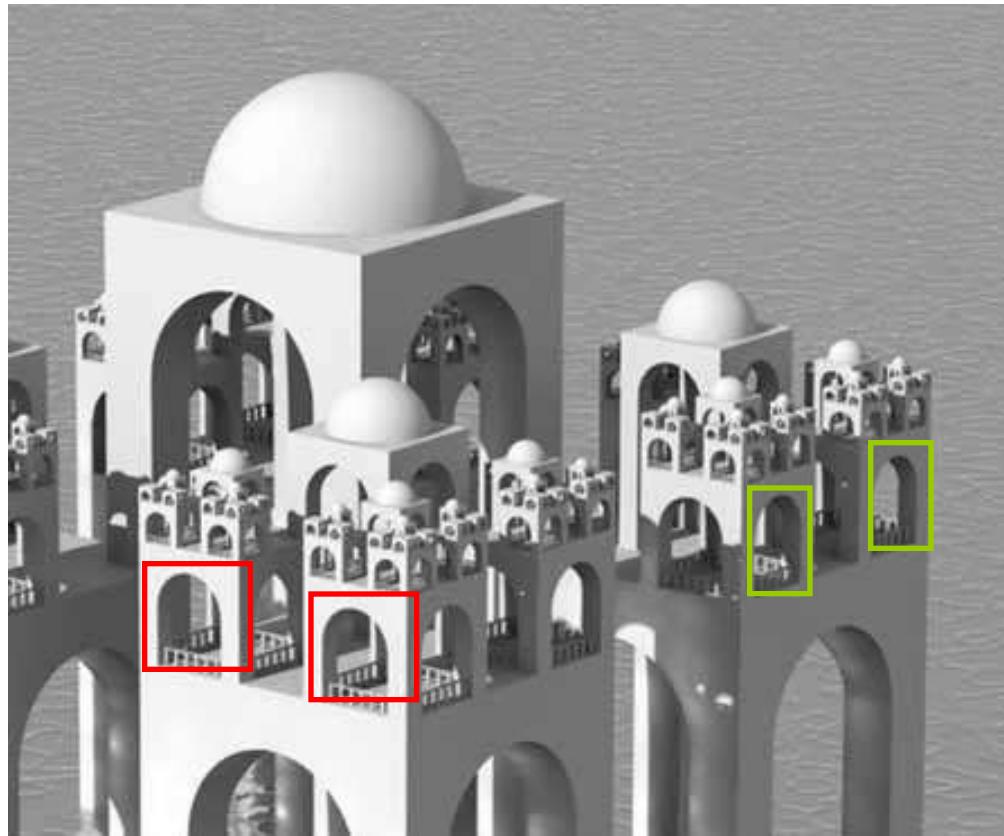
Dr. Xigun Lu

College of Computer Science

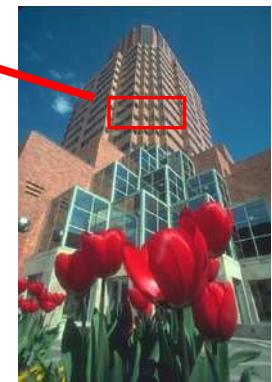
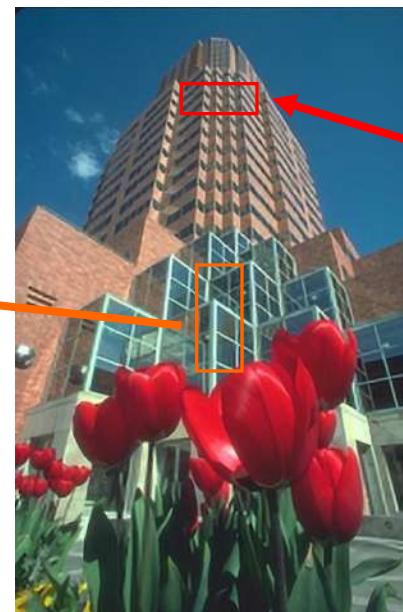
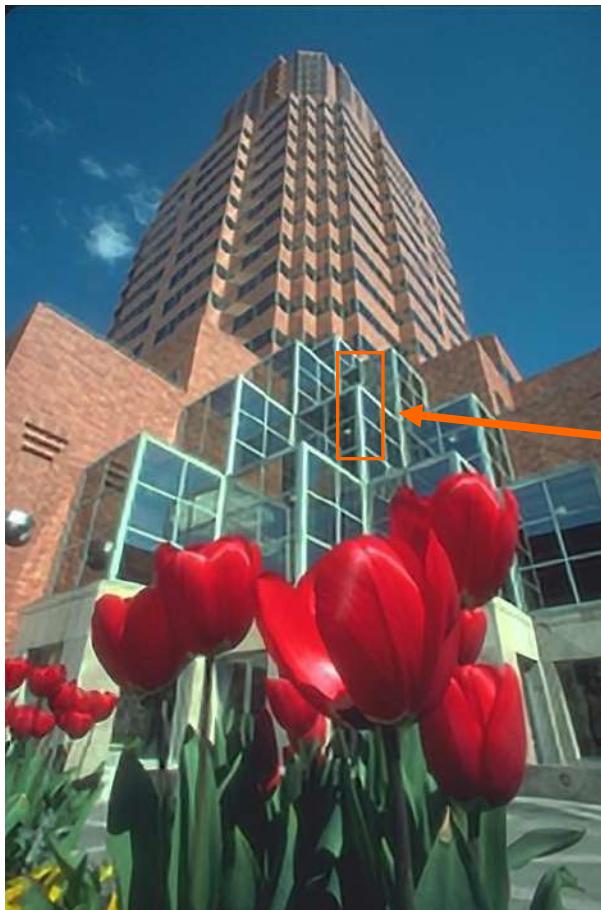
Zhejiang University

Superresolution from a single image [3]

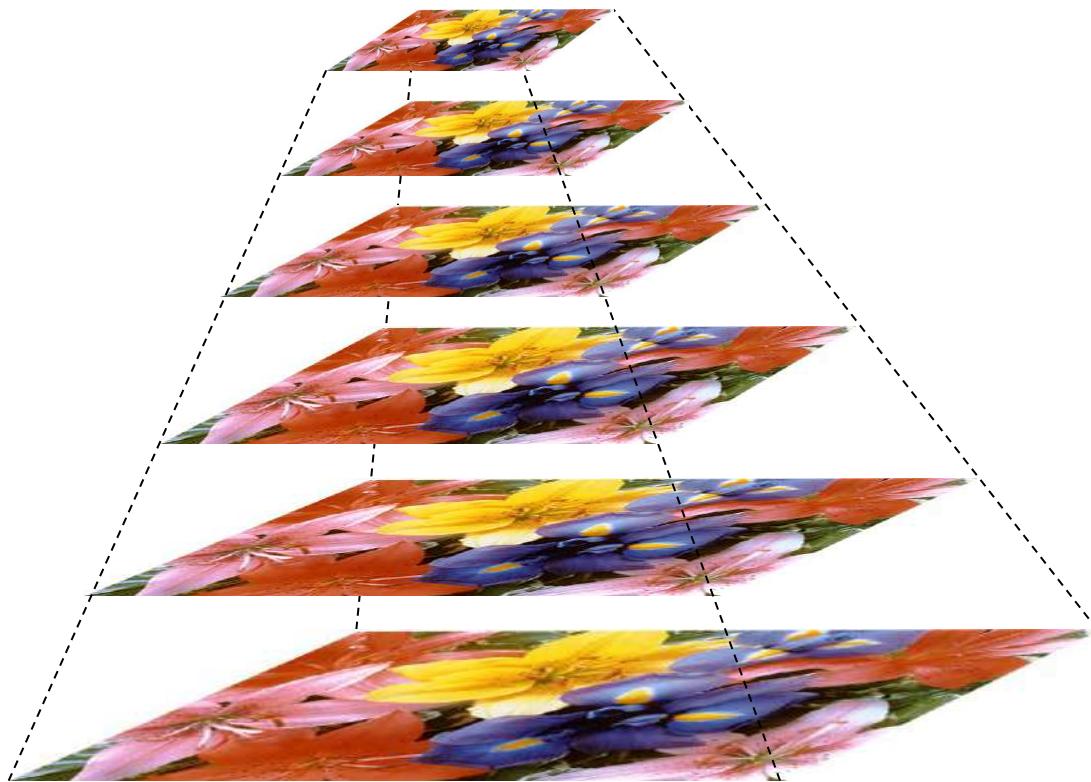
- Self-similarity in natural images



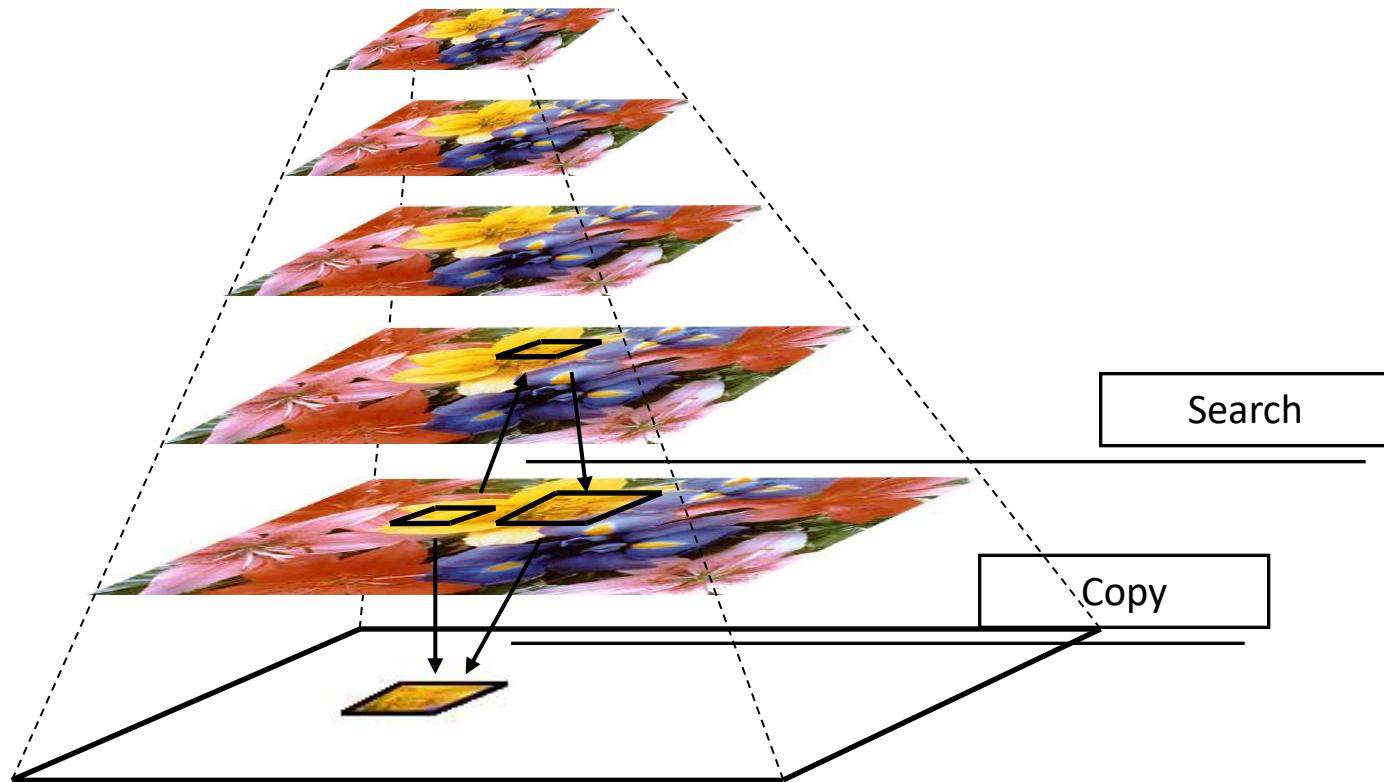
Self-similarity among different spatial scales



Gaussian Pyramid

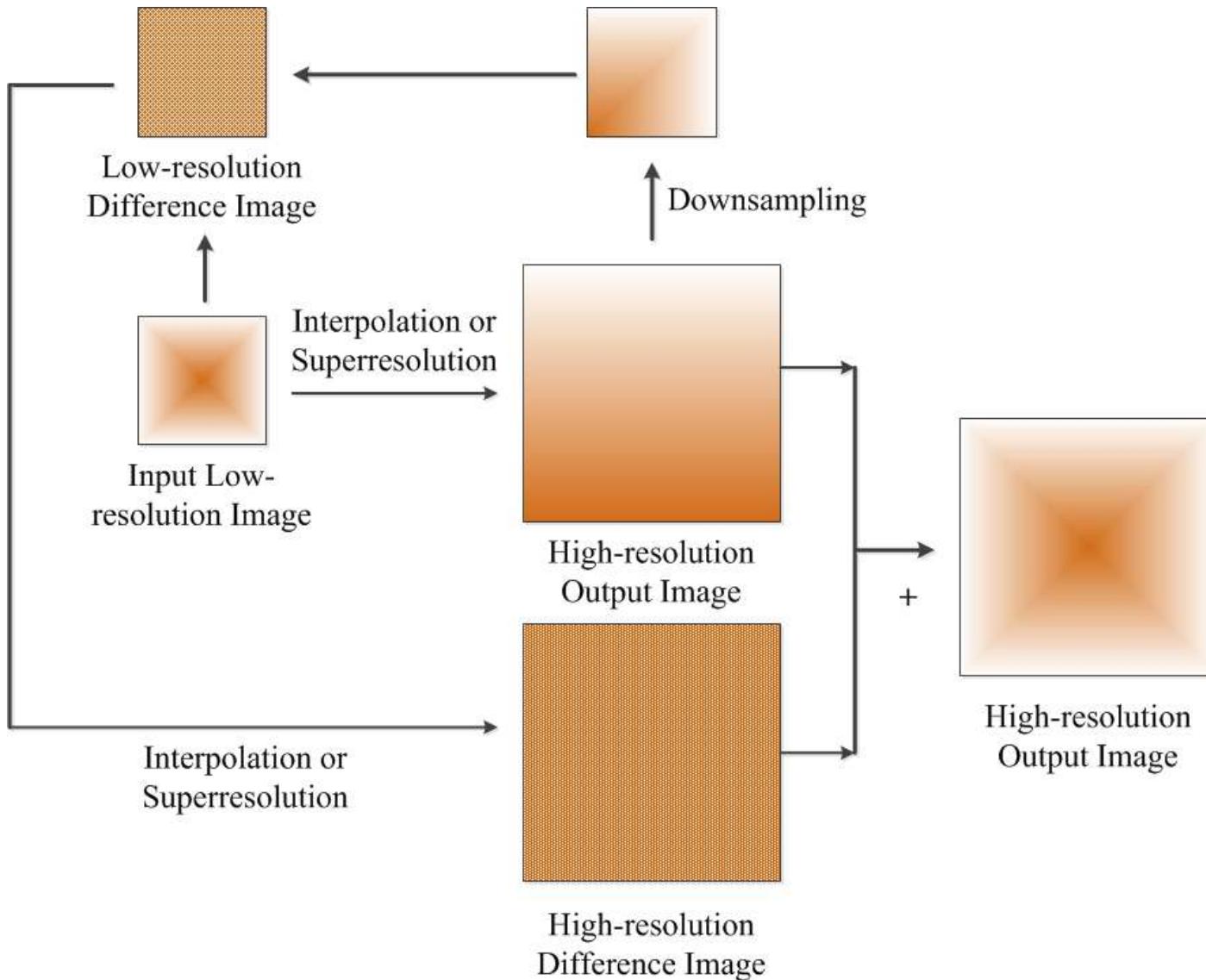


Looking for similar blocks among different scales

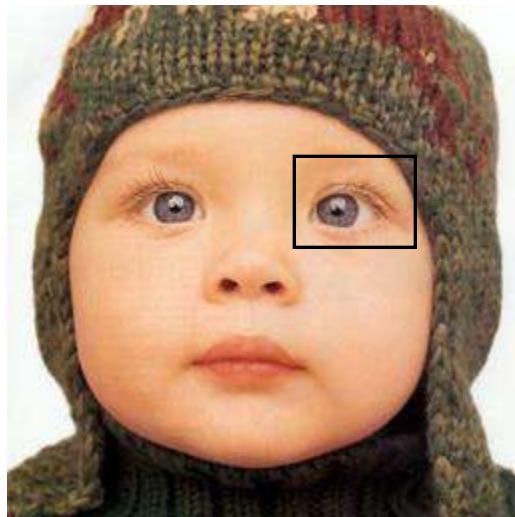


- Back-projection

Back-Projection



Experimental Results (I)



nearest



bilinear



bicubic



In scale BP



ICCV2009



Experimental Results (II)



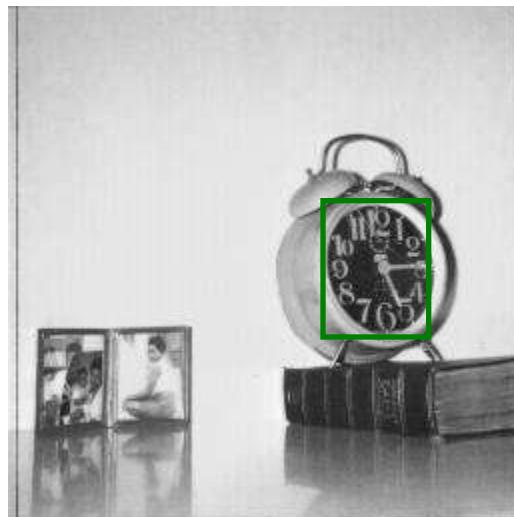
nearest

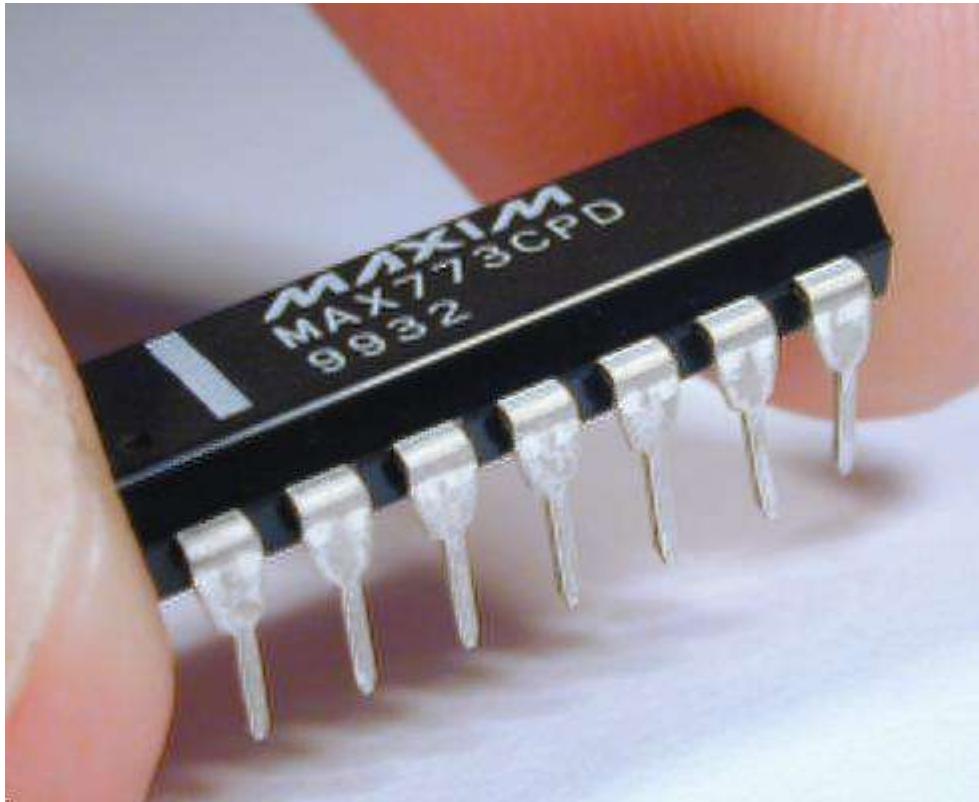
bilinear

bicubic

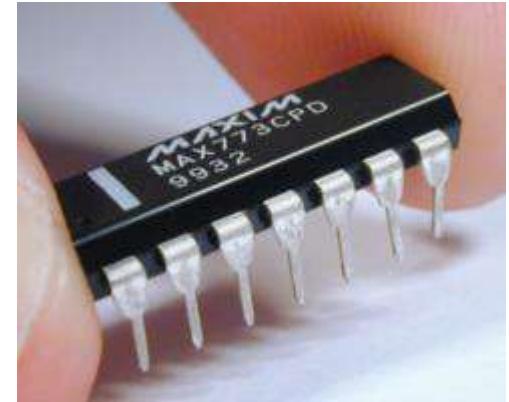
In scale BP

ICCV2009





clear



nearest

bilinear

bicubic

in scale BP

ICCV2009

References

- [3] D. Glasner, S. Bagon, and M. Irani, “Super-resolution from a single image,” In Proc. of ICCV, 2009.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Edge Detection — Canny

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Edge Detection

- The edge detection process serves to *simplify* the analysis of images by drastically reducing the amount of data to be processed, while at the same time *preserving useful structural information* about object boundaries [1].
- **Common criteria** relevant to edge detector performance
 - Low error rate
 - The edge points be well localized
 - Only one response to a single edge
- There is an **uncertainty** principle relating *detection* and *localization* of noisy step edges
 - The *tradeoff* between detection and localization can be varied by changing *the spatial width of the detector*.

Assumption of Canny Algorithm

- The image consists of the edge and additive white Gaussian noise.
 - Not true of corners

Canny Edge Detector [1]

- To smooth the image to eliminate any noise.
- To find the image gradient to highlight regions with high spatial derivatives.
- To track along these regions and suppresses any pixel that is not at the maximum (non-maximum suppression).
- Hypothesis is used to track along the remaining pixels that have not been suppressed. Hypothesis uses **two thresholds**
 - If the magnitude is below the first threshold, it is set to zero (made a **nonedge**).
 - If the magnitude is above the high threshold, it is made an **edge**.
 - And if the magnitude is between the two thresholds, then it is set to zero unless there is a path from this pixel to a pixel with a gradient above T2.

Step1: Gaussian Filter

- To filter out any noise in the original image before trying to locate and detect any edges.
- **The larger the width of the Gaussian mask, the lower is the detector's sensitivity to noise.**

$$\frac{1}{115}$$

2	4	5	4	2
4	9	12	9	4
5	12	15	12	5
4	9	12	9	4
2	4	5	4	2

Figure 3 Discrete approximation to Gaussian function with $\sigma=1.4$

Step2: Sobel Operator

- The Sobel operator performs a 2-D spatial gradient measurement on an image.
- The approximate absolute gradient magnitude (edge strength) at each point can be found.

$$|G| = |G_x| + |G_y|$$

- The edge direction
 $\theta = \arctan(G_y/G_x)$

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

G_x

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

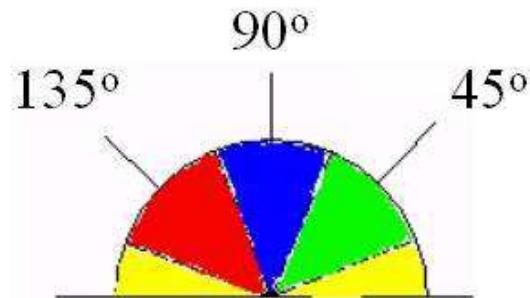
G_y

When the derivative in the x direction G_x is equal to zero, but the derivative in the y direction G_y is not zero, then the edge direction is equal to 90° .

Step3: Edge Direction

- **0 degrees** (in the horizontal direction)
- **45 degrees** (along the positive diagonal)
- **90 degrees** (in the vertical direction)
- **135 degrees** (along the negative diagonal).

x	x	x	x	x
x	x	x	x	x
x	x	a	x	x
x	x	x	x	x
x	x	x	x	x



Step4: Non-maximum Suppression

- Non-maximum suppression is used to trace along the edge in the edge direction and suppress any pixel value (sets it equal to 0) that is not considered to be an edge.

Step5: Hypothesis

- Hypothesis uses 2 thresholds, a high and a low.
- Any pixel in the image that has a value greater than T_1 is presumed to be an edge pixel, and is marked as such immediately. Then, any pixels that are connected to this edge pixel and that have a value greater than T_2 are also selected as edge pixels.
- If you think of following an edge, you need a gradient of T_2 to start but you don't stop till you hit a gradient below T_1 .

Canny Edge Detector Result



Input Gray Image



Detected Edge Image

Multiresolution SEL [2]

- 1) Multiresolution image pyramid
- 2) Enhancement — a Gaussian weighted gradient operator
- 3) Edge information contained in lower resolution will guide the sequential search at higher resolution according to log-likelihood statistic.
 - Both the strength of an edge and the past path of the edge are used to determine the search direction.
 - The strength of an edge is measured by the likelihood ratio of the conditional probability that an edge pixel exists. (dominating when SNR is high)
 - The past path of an edge is modeled by a Markov random chain. (dominating when SNR is low)
 - **Inaccuracies** may result for highly curved segments.

MSEL Result



Input Gray Image



Detected Edge Image

MSEL for Noisy Images



Input Noisy Image



Detected Edge Image

Canny for Noisy Image



Input Noisy Image



Detected Edge Image

MSEL vs. Canny



Canny Edge Image



MSEL Edge Image

References

- [1] J. Canny, “A Computational Approach to Edge Detection,” IEEE Trans. Pattern Analysis and Machine Intelligence, 8(6), pp. 679-698, 1986.
- [2] J.W. Cook and E.J. Delp, “Multiresolution sequential edge linking,” in Proc. IEEE Intl. Conf. Image Processing, pp.41-44, 1995.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

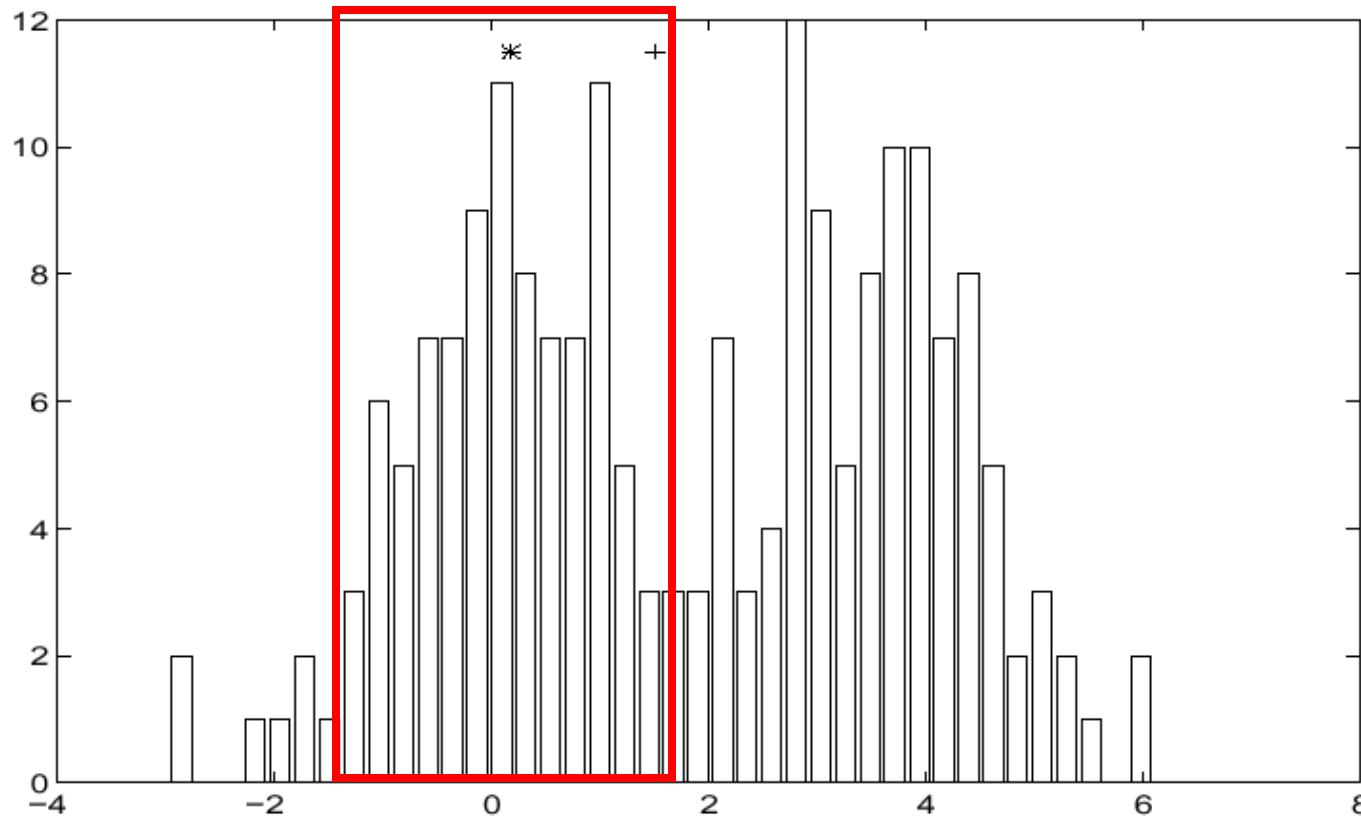
Color Image Segmentation — Mean Shift

Dr. Xigun Lu

College of Computer Science

Zhejiang University

Mean-shift Algorithm [1]



- Iterative Mode Search
 1. Initialize random seed, and fixed window
 2. Calculate center of gravity of the window (the “mean”)
 3. Translate the search window to the mean
 4. Repeat Step 2 until convergence

Background

- Feature space analysis is a widely used tool for solving low-level image understanding tasks.
 - Significant features correspond to high density regions in the feature space.
 - Feature space analysis is the procedure of recovering the centers of the high density regions.
 - Mean-shift – a simple nonparametric procedure for estimating density gradients.

Traditional Clustering Techniques

- Only reliable if the number of clusters is **small** and **known a prior**.
 - K-means
- Too often assume that the individual clusters obey **multivariate normal distributions**, i.e. the feature space can be modeled as a **mixture** of Gaussians.
- A strong artifact cluster may appear when several features are mapped into partially **overlapping** regions.

Spatial Constraints

- In image understanding tasks the data to be analyzed originates in the image domain. That is, the feature vectors satisfy additional **spatial constraints**.
- The feature space should be **isotropic**. A space is isotropic if the distance between two points is independent on the location of the point pair.

Mean-shift Algorithm

- Mean-shift was proposed in **1975** by Fukunaga and Hostetler.
- For the moment, assume the probability density function $p(\mathbf{x})$ of the p -dimensional feature vectors \mathbf{x} is unimodal.
- A sphere S_x of radius r , centered on \mathbf{x} contains the feature vectors \mathbf{y} such that $\|\mathbf{y} - \mathbf{x}\| \leq r$. The expected value of the vector $\mathbf{z} = \mathbf{y} - \mathbf{x}$, given \mathbf{x} and S_x is

$$\mu = E[\mathbf{z} | S_x] = \int_{S_x} (\mathbf{y} - \mathbf{x}) p(\mathbf{y} | S_x) d\mathbf{y} = \int_{S_x} (\mathbf{y} - \mathbf{x}) \frac{p(\mathbf{y})}{p(\mathbf{y} \in S_x)} d\mathbf{y}$$

Mean-shift Algorithm

$$\mu = E[\mathbf{z} | S_x] = \int_{S_x} (\mathbf{y} - \mathbf{x}) p(\mathbf{y} | S_x) d\mathbf{y} = \int_{S_x} (\mathbf{y} - \mathbf{x}) \frac{p(\mathbf{y})}{p(\mathbf{y} \in S_x)} d\mathbf{y}$$

- If S_x is sufficiently small, we can approximate

$$p(\mathbf{y} \in S_x) \approx p(\mathbf{x}) V_{S_x} \quad V_{S_x} = c \cdot r^p$$

- The first order approximation of $p(\mathbf{y})$ is

$$p(\mathbf{y}) = p(\mathbf{x}) + (\mathbf{y} - \mathbf{x})^T \nabla p(\mathbf{x})$$

- Where $\nabla p(\mathbf{x})$ is the gradient of the probability density function in \mathbf{x}

$$\mu = \int_{S_x} \frac{(\mathbf{y} - \mathbf{x})(\mathbf{y} - \mathbf{x})^T}{V_{S_x}} \frac{\nabla p(\mathbf{x})}{p(\mathbf{x})} d\mathbf{y} = \frac{r^2}{p+2} \frac{\nabla p(\mathbf{x})}{p(\mathbf{x})}$$

$$\mu = \int_{S_x} (\mathbf{y} - \mathbf{x}) p(\mathbf{y} | S_x) d\mathbf{y} = \int_{S_x} (\mathbf{y} - \mathbf{x}) \frac{p(\mathbf{y})}{p(\mathbf{y} \in S_x)} d\mathbf{y}$$

$$\therefore p(\mathbf{y} \in S_x) \approx p(\mathbf{x}) V_{S_x}$$

$$\therefore p(\mathbf{y}) = p(\mathbf{x}) + (\mathbf{y} - \mathbf{x})^T \nabla p(\mathbf{x})$$

$$\mu = \int_{S_x} (\mathbf{y} - \mathbf{x}) \frac{p(\mathbf{x}) + (\mathbf{y} - \mathbf{x})^T \nabla p(\mathbf{x})}{p(\mathbf{x}) V_{S_x}} d\mathbf{y}$$

$$= \int_{S_x} (\mathbf{y} - \mathbf{x}) \frac{1}{V_{S_x}} d\mathbf{y} + \int_{S_x} (\mathbf{y} - \mathbf{x})(\mathbf{y} - \mathbf{x})^T \frac{\nabla p(\mathbf{x})}{p(\mathbf{x}) V_{S_x}} d\mathbf{y}$$

$$= \int_{S_x} \frac{(\mathbf{y} - \mathbf{x})(\mathbf{y} - \mathbf{x})^T}{V_{S_x}} \frac{\nabla p(\mathbf{x})}{p(\mathbf{x})} d\mathbf{y} = \frac{r^2}{p+2} \frac{\nabla p(\mathbf{x})}{p(\mathbf{x})}$$

Mean-shift Vector

- The vector of difference between **the local mean** and **the center of the window** is proportional to the gradient of the probability density at \mathbf{x} .

$$\mu = \int_{S_x} (\mathbf{y} - \mathbf{x}) p(\mathbf{y} | S_x) d\mathbf{y} = \frac{r^2}{p+2} \frac{\nabla p(\mathbf{x})}{p(\mathbf{x})}$$

- This is beneficial when **the highest density region** of the probability density function is sought, such region corresponds to **large** $p(\mathbf{x})$ and **small** $\nabla p(\mathbf{x})$.
- Low density regions correspond to large mean shifts. The shifts are always in the direction of the probability density maximum.

Mean-shift Algorithm

- At the mode the mean shift is close to zero.
 - Choose the radius r of the search window.
 - Choose the initial location of the window.
 - Compute the mean shift vector and translate the search window by that amount.
 - Repeat till convergence.

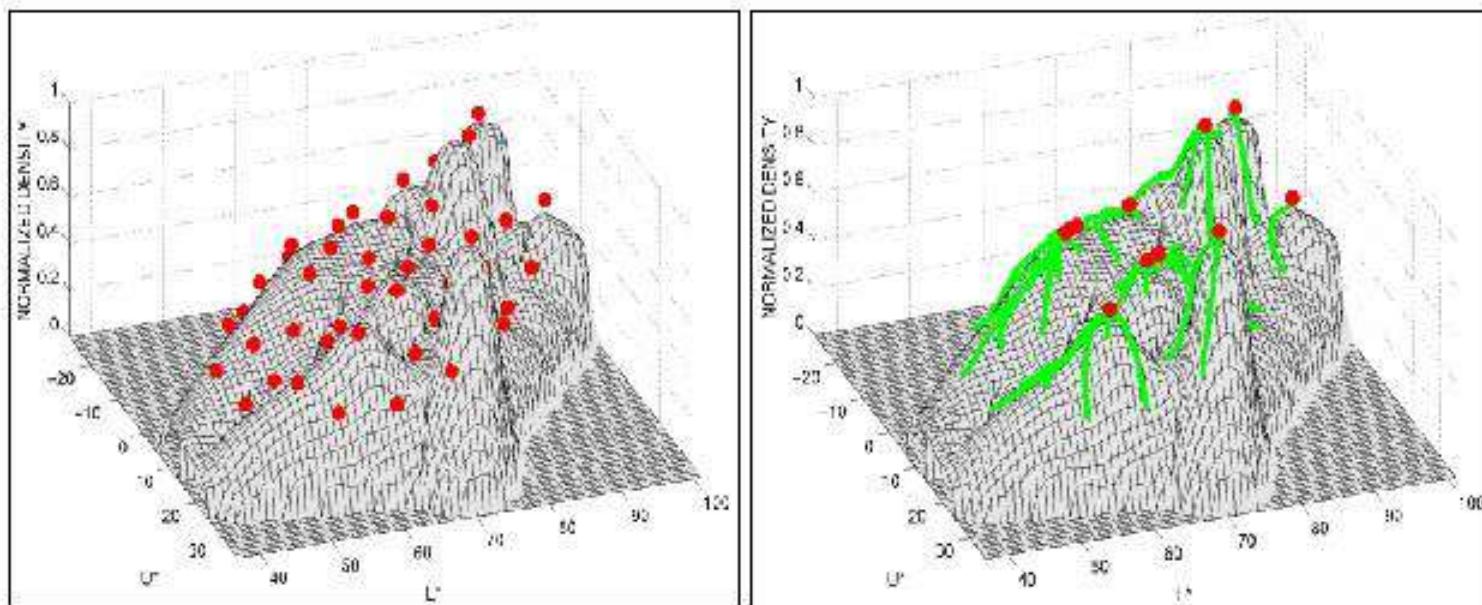
Feature Space Analysis

- Map the image domain into the feature space.
- Define an adequate number of search windows at random locations in the space.
- Find the high density region centers by applying the mean-shift algorithm to each window.
- Validate the extracted centers with image domain constraints to provide the feature palette.
- Allocate, using image domain information, all the feature vectors to the feature palette.

Mean-Shift of Multimodal

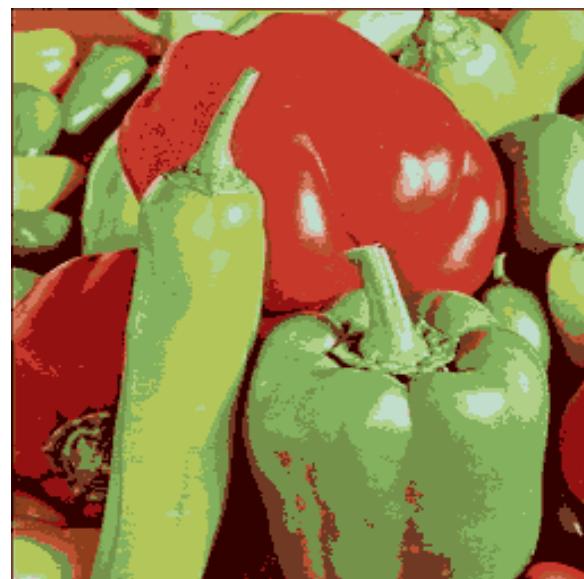
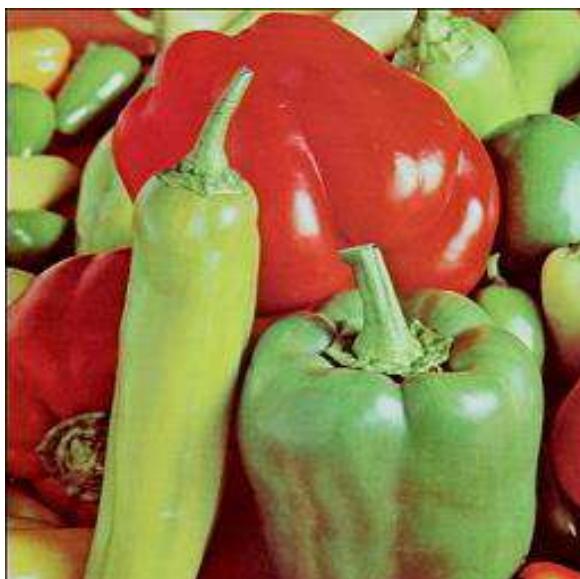
Multimodal Distributions

- Parallel processing of an initial tessellation.
- Pruning of mode candidates.
- Classification based on the basin of attraction.



Mean shift trajectories

Mean-shift for Natural Color Images



References

- [1] D. Comaniciu and P. Meer, “Robust analysis of feature spaces: color image segmentation,” in Proc. Of IEEE CVPR’1997.

Thank You!

Dr. Xigun Lu

xqlu@zju.edu.cn

Tracking

Dr. Xiqun Lu
College of Computer Science
Zhejiang University

Minimum Output Sum of Squared Error Filter (MOSSE) [1]

- The MOSSE filter is training **online**.
- MOSSE finds a filter \mathbf{h} that minimizes the sum of squared error between the actual output of the convolution and the desired output of the convolution. The minimization problem takes the form:

$$\min_{\mathbf{H}^*} \sum_i |\mathbf{F}_i \odot \mathbf{H}^* - \mathbf{G}_i|^2$$

where \odot denotes the Hadamard product.

MOSSE [1]

- Set the partial derivative of the above error function w.r.t. \mathbf{H} equals to zero, we have

$$E = \sum_i |\mathbf{F}_i \odot \mathbf{H} - \mathbf{G}_i|^2 = \sum_i (\mathbf{F}_i \odot \mathbf{H} - \mathbf{G}_i)^H (\mathbf{F}_i \odot \mathbf{H} - \mathbf{G}_i)$$

$$\frac{\partial E}{\partial \mathbf{H}} = \sum_i \mathbf{F}_i^H (\mathbf{F}_i \odot \mathbf{H} - \mathbf{G}_i) = 0$$

$$\mathbf{H}^* = \frac{\sum_i \mathbf{F}_i^H \odot \mathbf{G}_i}{\sum_i \mathbf{F}_i^H \odot \mathbf{F}_i}$$

- Regularization

$$\mathbf{H}^* = \frac{\sum_i \mathbf{F}_i^H \odot \mathbf{G}_i}{\sum_i \mathbf{F}_i^H \odot \mathbf{F}_i + \varepsilon}$$

where ε is the regularization parameter. This result suggests that adding the energy spectrum of **the background noise** to that of the training imagery will produce a filter with better in noise tolerance.

Updating — Running Average

$$\mathbf{H}_i^* = \frac{\mathbf{A}_i}{\mathbf{B}_i}$$

$$\mathbf{A}_i = \eta \mathbf{G}_i \odot \mathbf{F}_i^H + (1 - \eta) \mathbf{A}_{i-1}$$

$$\mathbf{B}_i = \eta \mathbf{F}_i \odot \mathbf{F}_i^H + (1 - \eta) \mathbf{B}_{i-1}$$

where η is the learning rate. This puts more weight on recent frames and lets the effect of previous frames decay exponentially over time.

Kernelized Correlation Filter (KCF) [2]

- Ridge regression
 - It admits a simple closed-form solution
 - Can achieve performance that is close to SVM
 - The goal of training is to find a function $f(\mathbf{z}) = \mathbf{w}^T \mathbf{z}$ that minimizes the squared error over sample \mathbf{x}_i and their regression targets y_i ,

$$\begin{aligned} & \min_{\mathbf{w}} \sum_{i=1}^N (f(\mathbf{x}_i) - y_i)^2 + \lambda \|\mathbf{w}\|^2 \\ &= \sum_{i=1}^N (\mathbf{w}^T \mathbf{x}_i - y_i)^2 + \lambda \|\mathbf{w}\|^2 \\ &= \sum_{i=1}^N (\mathbf{x}_i^T \mathbf{w} - y_i)^2 + \lambda \|\mathbf{w}\|^2 \\ &= \left\| \begin{pmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{pmatrix} \mathbf{w} - \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} \right\|^2 + \lambda \|\mathbf{w}\|^2 \\ & \text{Let } \mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_N^T \end{pmatrix}_{N \times M} \quad \mathbf{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix}_{N \times 1} \\ &= \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \lambda \|\mathbf{w}\|^2 \\ &= (\mathbf{X}\mathbf{w} - \mathbf{y})^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w}^T \mathbf{w} \\ &\frac{\partial E}{\partial \mathbf{w}} = \mathbf{X}^T (\mathbf{X}\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w} = 0 \rightarrow \mathbf{w} = (\mathbf{X}^T \mathbf{X} + \lambda I)^{-1} \mathbf{X}^T \mathbf{y} \end{aligned} \tag{2}$$

N — the number of training samples

If each training sample \mathbf{x} has the dimension of M , the computational complexity of this ridge regression is $O(M^3)$, since the main computational load is to compute $(\mathbf{X}^T \mathbf{X} + \lambda I)^{-1}$.

Kernelized Correlation Filter (KCF) [2]

- Cyclic shifts
- Permutation matrix

$$P = \begin{pmatrix} 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix} \quad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{N-1} \\ x_N \end{pmatrix} \quad P\mathbf{x} \rightarrow \begin{pmatrix} x_N \\ x_1 \\ \vdots \\ x_{N-2} \\ x_{N-1} \end{pmatrix} \quad (4)$$

- Each column has one and only one “1”, each row has one and only one “1”
- We can chain u shifts to achieve a larger translation by using the matrix power $P^u \mathbf{x}$.

Kernelized Correlation Filter (KCF) [2]

- Circularly shifted samples



(a)

(b)

(c)

(d)

(e)

(f)

(a) The base sample \mathbf{x} , (b) – (f) circularly shifted versions

- Due to the cyclic property, we get the same signal \mathbf{x} periodically every M shift.

$$\left\{ P^u \mathbf{x} \mid u = 0, 1, \dots, M - 1 \right\} \quad (5)$$

- Cyclic shifts will induce **distortion** to samples, except the base sample \mathbf{x} , the other circularly shifted samples are not the true negative samples but the virtual samples.
 - However, this undesirable property can be mitigated by appropriate **padding** and **windowing**.

Circulant Matrix

- To compute a regression with shifted samples, we can use the set of Eq.(5) as the rows of a data matrix \mathbf{X} :

$$\mathbf{X} = \begin{pmatrix} x_1 & x_2 & x_3 & \cdots & x_m \\ x_m & x_1 & x_2 & \cdots & x_{m-1} \\ x_{m-1} & x_m & x_1 & \cdots & x_{m-2} \\ \vdots & & & \ddots & \vdots \\ x_2 & x_3 & x_4 & \cdots & x_1 \end{pmatrix}$$

- Since the circulant matrix can be diagonalized by the DFT

$$\mathbf{X} = F diag(\hat{\mathbf{x}}) F^H \quad (7)$$

where $\hat{\mathbf{x}}$ denotes the DFT of the base signal \mathbf{x} , $\hat{\mathbf{x}} = F(\mathbf{x})$

Ridge Regression

- The DFT matrix F is a unitary matrix, and unitary matrix preserves the 2-norm.

$$E = \|\mathbf{X}\mathbf{w} - \mathbf{y}\|^2 + \lambda \|\mathbf{w}\|^2$$

$$= \|F\mathbf{X}\mathbf{w} - F\mathbf{y}\|^2 + \lambda \|F\mathbf{w}\|^2$$

$$= \|F\mathbf{X}F^H F\mathbf{w} - F\mathbf{y}\|^2 + \lambda \|F\mathbf{w}\|^2 \quad (F^H F = I)$$

$$= \|\hat{\mathbf{X}}\hat{\mathbf{w}} - \hat{\mathbf{y}}\|^2 + \lambda \|\hat{\mathbf{w}}\|^2 = (\hat{\mathbf{X}}\hat{\mathbf{w}} - \hat{\mathbf{y}})^H (\hat{\mathbf{X}}\hat{\mathbf{w}} - \hat{\mathbf{y}}) + \lambda \hat{\mathbf{w}}^H \hat{\mathbf{w}}$$

$$\frac{\partial E}{\partial \hat{\mathbf{w}}} = -\hat{\mathbf{X}}^H (\hat{\mathbf{X}}\hat{\mathbf{w}} - \hat{\mathbf{y}}) + \lambda \hat{\mathbf{w}} = 0$$

$$\hat{\mathbf{X}} = diag(\hat{\mathbf{x}}) \quad \hat{\mathbf{X}}^H = diag(\hat{\mathbf{x}}^*)$$

$$\hat{\mathbf{w}} = (\hat{\mathbf{X}}^H \hat{\mathbf{X}} + \lambda I)^{-1} \hat{\mathbf{X}}^H \hat{\mathbf{y}} = (diag(\hat{\mathbf{x}}^*) diag(\hat{\mathbf{x}}) + \lambda I)^{-1} (diag(\hat{\mathbf{x}}^*) \hat{\mathbf{y}})$$

$$= diag\left(\frac{\hat{\mathbf{x}}^*}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda}\right) \hat{\mathbf{y}} = \frac{\hat{\mathbf{x}}^* \odot \hat{\mathbf{y}}}{\hat{\mathbf{x}}^* \odot \hat{\mathbf{x}} + \lambda}$$

The computational complexity is $O(M \log M)$.

Nonlinear Regression

- Kernel trick — Mapping the inputs of a linear problem to a non-linear feature space $\varphi(\mathbf{x})$ with the kernel trick consists of:
 - 1) Expressing the solution \mathbf{w} as a linear combination of the samples:

$$\mathbf{w} = \sum_i \alpha_i \varphi(\mathbf{x}_i)$$

The variables under optimization are thus $\boldsymbol{\alpha}$, instead of \mathbf{w} .

- 2) The dot-products are computed using kernel function κ (e.g. Gaussian and Polynomial)

$$\varphi^T(\mathbf{x})\varphi(\mathbf{x}') = \kappa(\mathbf{x}, \mathbf{x}')$$

The dot-products between all pairs of samples are usually stored in a $N \times N$ **kernel matrix K** , with elements $K_{ij} = \kappa(\mathbf{x}_i, \mathbf{x}_j)$.

The regression function's complexity grows with the number of samples,

$$f(\mathbf{z}) = \mathbf{w}^T \mathbf{z} = \sum_{i=1}^N \alpha_i \kappa(\mathbf{z}, \mathbf{x}_i) \quad (15)$$

Fast Kernel Regression

- The kernelized version of ridge regression is given by

$$\mathbf{a} = (K + \lambda I)^{-1} \mathbf{y} \quad (16)$$

- In general, the kernel matrix K is not circulant.
- **Theorem 1.** Given circulant data matrix $C(\mathbf{x})$, the corresponding kernel matrix K is circulant if the kernel function satisfies $\kappa(\mathbf{x}, \mathbf{x}') = \kappa(P\mathbf{x}, P\mathbf{x}')$, for any permutation matrix P .
 - Radial Basis Function kernels — e.g., Gaussian
 - Dot-product kernels — e.g., linear, polynomial

Fast Kernel Regression

- Knowing which kernels we can use to make K circulant, it is possible to diagonalize Eq.(16) as in the linear case:

$$\hat{\mathbf{a}} = \frac{\hat{\mathbf{y}}}{\hat{\mathbf{k}}^{\mathbf{x}\mathbf{x}} + \lambda} \quad (17)$$

$\mathbf{k}^{\mathbf{x}\mathbf{x}}$ is the first row of the kernel matrix $K = C(\mathbf{k}^{\mathbf{x}\mathbf{x}})$.

$$\mathbf{k}_i^{\mathbf{x}\mathbf{x}} = \varphi^T(\mathbf{x})\varphi(P^{i-1}\mathbf{x})$$

$\hat{\mathbf{k}}^{\mathbf{x}\mathbf{x}}$ is the kernel correlation of \mathbf{x} with itself, in the Fourier domain.

Fast Detection

- To detect the target, we typically wish to evaluate $f(\mathbf{z})$ on several locations around the estimated location in the previous frame, i.e., for several candidate patches. These patches can be modeled by cyclic shifts.
- Denote by $K^{\mathbf{z}}$ the (asymmetric) kernel matrix between all training samples and all candidate patches. Since the samples and patches are cyclic shifts of base sample \mathbf{x} and base patch \mathbf{z} , respectively, each element of $K^{\mathbf{z}}$ is given by $\kappa(P^{i-1}\mathbf{z}, P^{i-1}\mathbf{x})$.
- It is easy to verify that this kernel matrix satisfies Theorem 1, and is circulant for appropriate kernels. $K^{\mathbf{z}} = C(\mathbf{k}^{\mathbf{xz}})$ where $\mathbf{k}^{\mathbf{xz}}$ is the kernel correlation of \mathbf{x} and \mathbf{z} .

$$\hat{f}(\mathbf{z}) = \hat{\mathbf{k}}^{\mathbf{xz}} \odot \hat{\mathbf{a}} \quad (22)$$

References

- [1] D.S. Bolme, J.R. Beveridge, B.A. Draper, and Y.M. Lui, “Visual object tracking using adaptive correlation filters,” in Proc. of CVPR, 2010.
- [2] J.F. Henriques, R. Caseiro, P. Martins, and J. Batista, “High-speed tracking with kernelized correlation filter” IEEE Trans. On Pattern Analysis and Machine Intelligence, vol.37, no.3, pp.583-596, Mar. 2015.

Thank You

Dr. Xiqun Lu
xqlu@zju.edu.cn