

第二章：图像变换+形态学

授课老师：李厚强 (lihq@ustc.edu.cn)

周文罡 (zhwg@ustc.edu.cn)

李 礼 (lil1@ustc.edu.cn)

胡 洋 (eeeyhu@ustc.edu.cn)

□ 图像变换

- 有时候图像的一些处理和分析不适合在像素域中完成，更适合在变换域中处理，由此引出来了图像变换

□ 图像变换

- ✓ 可分离和正交图像变换
- ✓ 离散傅立叶变换 (DFT)
- ✓ 离散余弦变换 (DCT)
- ✓ 沃尔什/哈达玛变换
- ✓ Karhunen-Loeve变换 (KLT)
- ✓ 小波变换 (DWT)

可分离和正交图象变换

□ 1-D变换

■ 正变换

正向变换核

$$T(u) = \sum_{x=0}^{N-1} f(x)h(x,u) \quad u = 0, 1, \dots, N-1$$

■ 反变换

反向变换核

$$f(x) = \sum_{u=0}^{N-1} T(u)k(x,u) \quad x = 0, 1, \dots, N-1$$

可分离和正交图象变换

□ 2-D变换

$$T(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) h(x, y, u, v) \quad (1)$$

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} T(u, v) k(x, y, u, v) \quad (2)$$

正向变换核

变换核与
原始函数及
变换后函数无关

反向变换核

可分离和正交图象变换

□ 可分离

$$h(x, y, u, v) = h_1(x, u)h_2(y, v)$$

1个2-D变换分成2个1-D变换

$$T(x, v) = \sum_{y=0}^{N-1} f(x, y)h_2(y, v) \quad T(u, v) = \sum_{x=0}^{N-1} T(x, v)h_1(x, u)$$

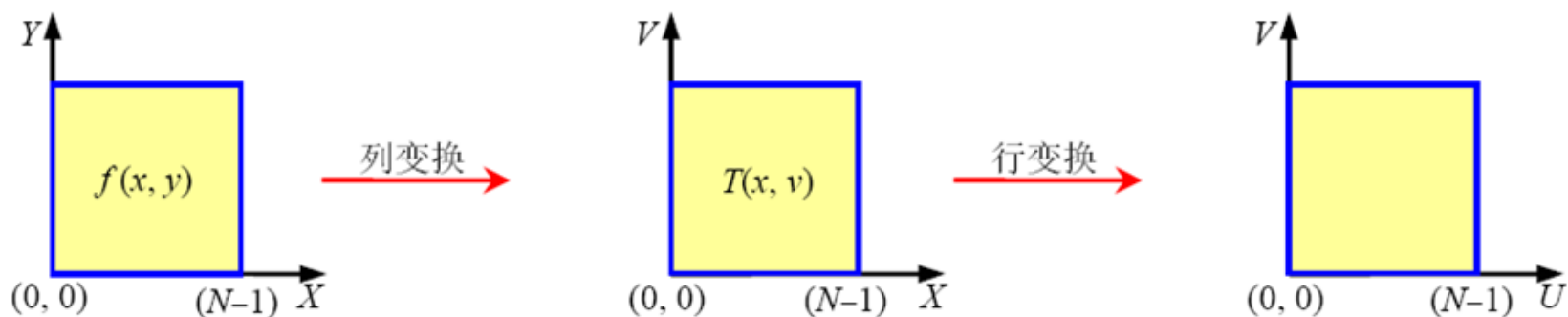
□ 对称

$$h(x, y, u, v) = h_1(x, u)h_1(y, v)$$

(h_1 与 h_2 的函数形式一样)

可分离和正交图象变换

- 具有可分离变换核的2-D变换可以分成两个步骤计算，每个步骤用一个1-D变换



由 2 步 1-D 变换计算 2-D 变换

可分离和正交图象变换

□ 可分离且对称

变换结果

$$T = AFA$$

对称变换矩阵

图象矩阵

反变换矩阵

$$BTB = BAFAB$$

$$F = BTB \quad B = A^{-1}$$

$$\hat{F} = BAFAB \quad B \neq A^{-1}$$



可分离和正交图象变换

□ 正交

考虑变换矩阵： $B = A^{-1}$ $F = BTB$

酉矩阵（*代表共轭）： $A^{-1} = A^{*T}$

如果A为实矩阵，且： $A^{-1} = A^T$

则A为正交矩阵，构成正交变换对

离散傅立叶变换 (DFT)

□ 二维离散傅立叶变换式

■ 对于 $N \times N$ 的二维矩阵（方阵），二维离散傅立叶变换对为：

$$F(u, v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \exp \left\{ \frac{-2\pi j (ux + vy)}{N} \right\}$$

$$f(x, y) = \frac{1}{N} \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) \exp \left\{ \frac{+2\pi j (ux + vy)}{N} \right\}$$

二维DFT的性质

□ 线性 $f_1(x, y) + f_2(x, y) \quad F_1(u, v) + F_2(u, v)$

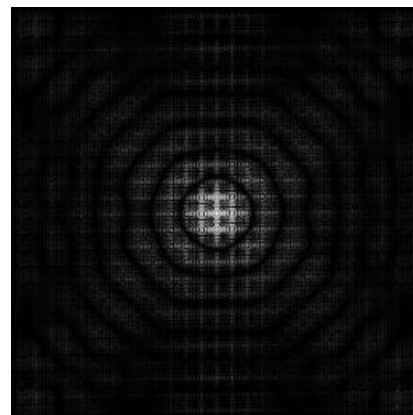
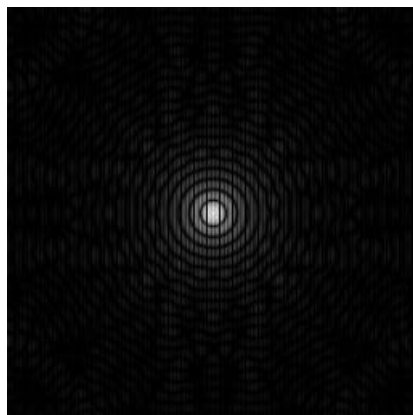
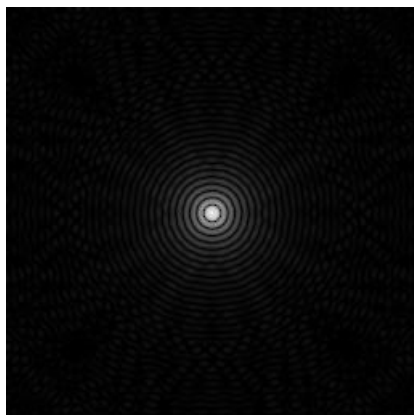
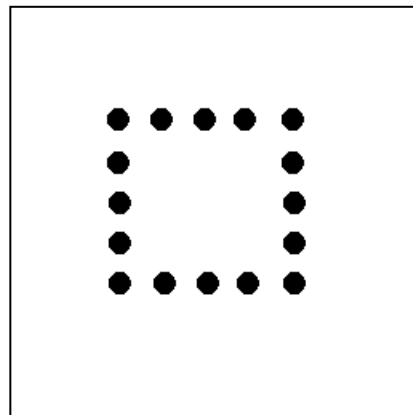
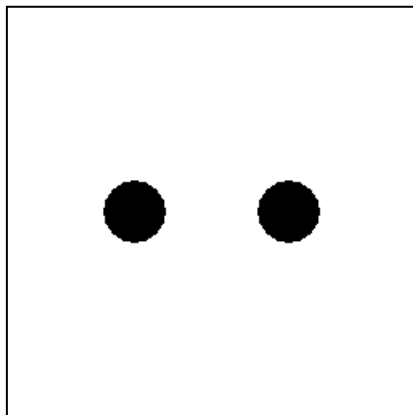
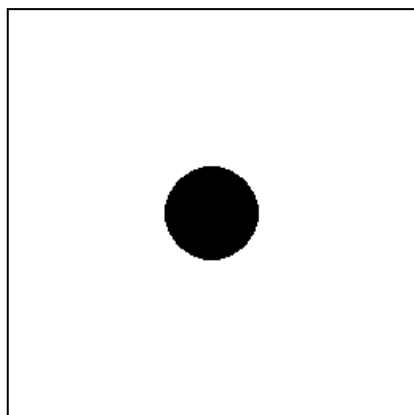
□ 比例 $f(ax, by) \quad \frac{1}{ab} F\left(\frac{u}{a}, \frac{v}{b}\right)$

□ 平移 $f(x-a, y-b) \quad e^{-j2\pi(au+bv)} F(u, v)$
 $e^{j2\pi(cx+dy)} f(x, y) \quad F(u-c, v-d)$

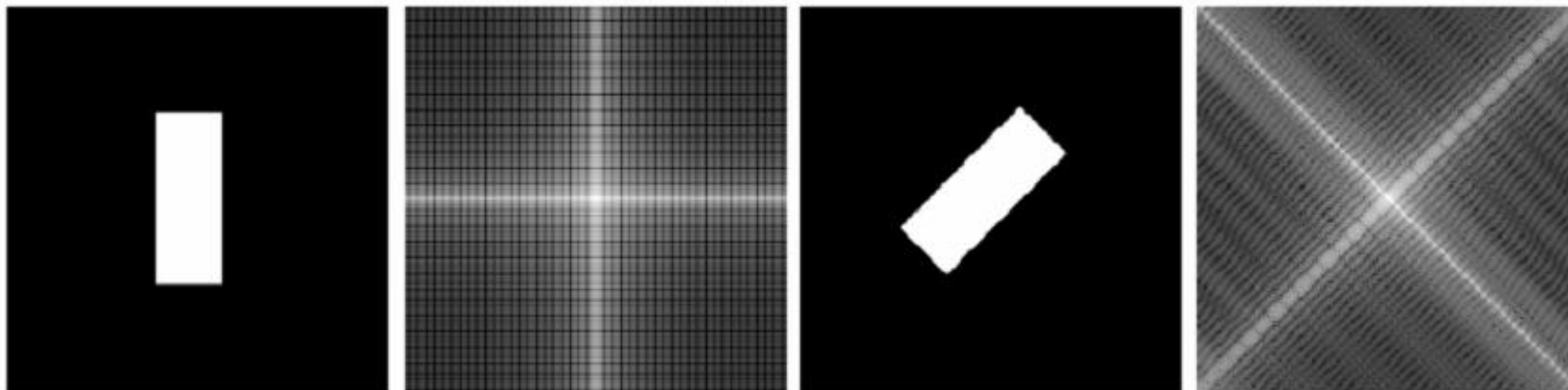
□ 卷积 $f_1(x, y) * f_2(x, y) \quad F_1(u, v) F_2(u, v)$
 $f_1(x, y) f_2(x, y) \quad F_1(u, v) * F_2(u, v)$

□ 旋转 $f(x \cos \theta + y \sin \theta, -x \sin \theta + y \cos \theta) \quad F(u \cos \theta + v \sin \theta, -u \sin \theta + v \cos \theta)$

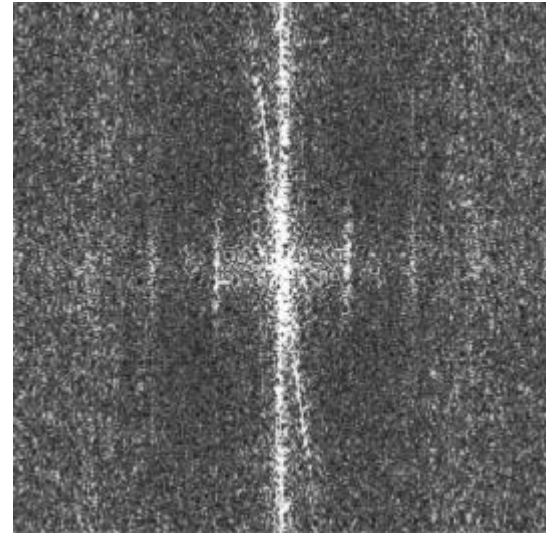
线性叠加及尺度变化



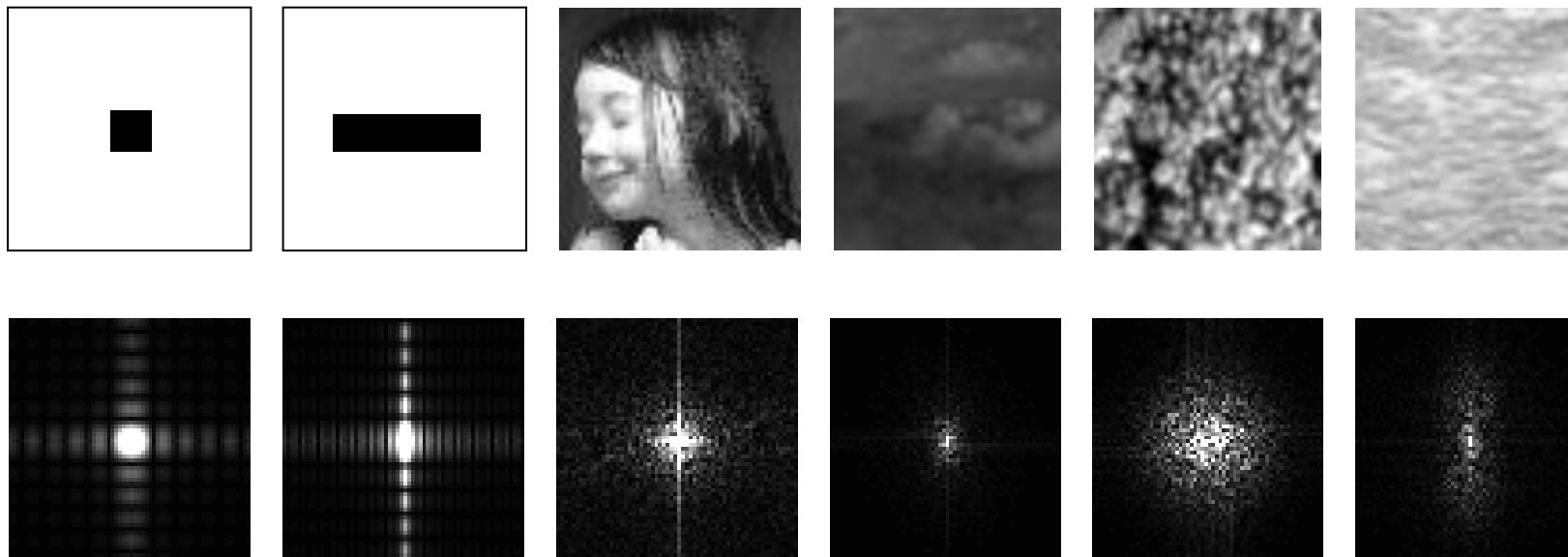
旋转性



实例



典型图象的频谱



离散余弦变换 (DCT)

□ 1-D离散余弦变换

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

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

$$a(u) = \begin{cases} \sqrt{1/N} & \text{当 } u = 0 \\ \sqrt{2/N} & \text{当 } u = 1, 2, \dots, N-1 \end{cases}$$

离散余弦变换 (DCT)

- 2-D离散余弦变换
- 一种可分离、正交、对称的变换

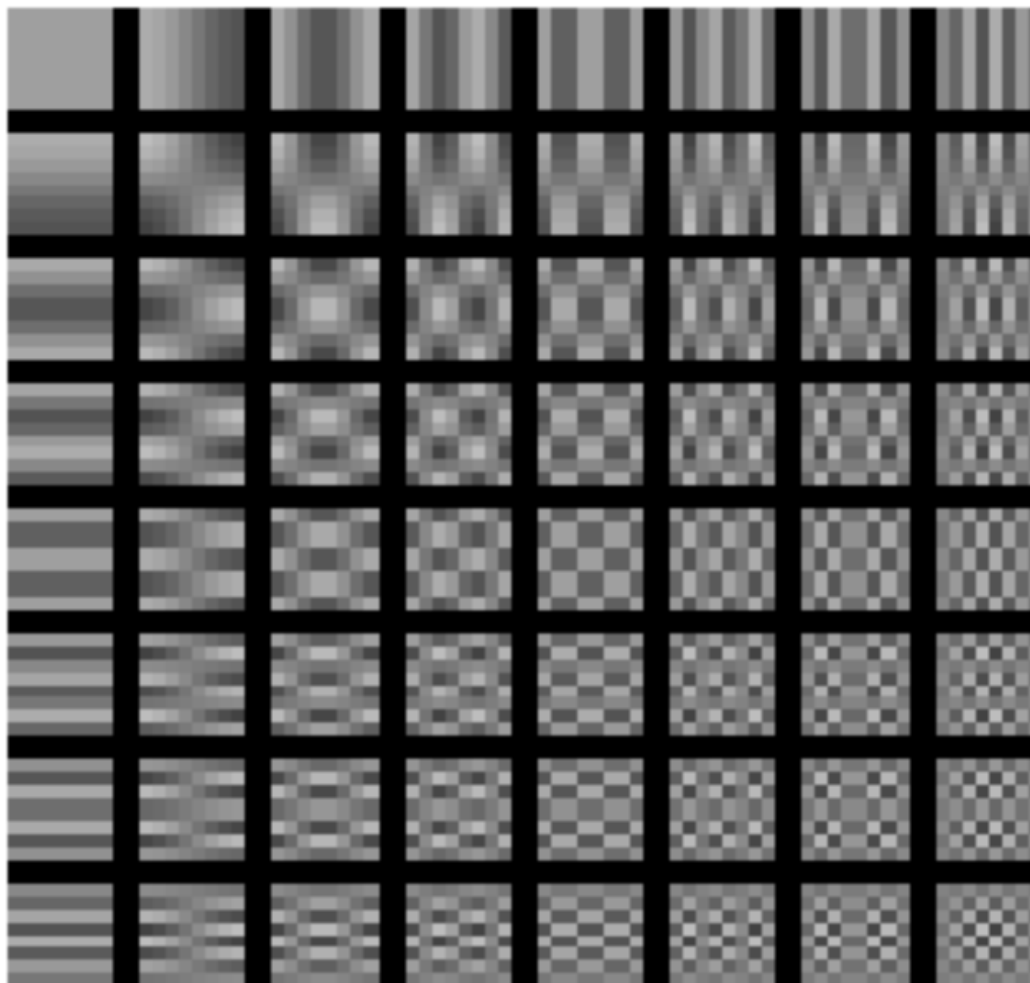
$$C(u, v) = a(u)a(v) \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) \cos \left[\frac{(2x+1)u\pi}{2N} \right] \cos \left[\frac{(2y+1)v\pi}{2N} \right]$$

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

讨论可分离性和对称性

$$h(x, y, u, v) = h_1(x, u)h_2(y, v) \quad h(x, y, u, v) = h_1(x, u)h_1(y, v)$$

DCT基函数

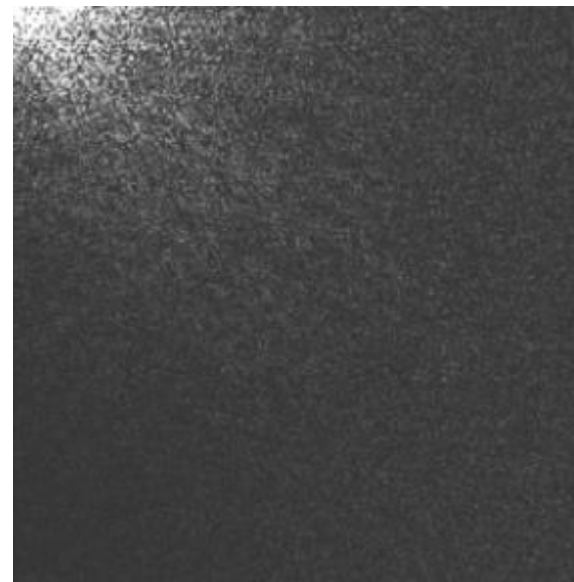




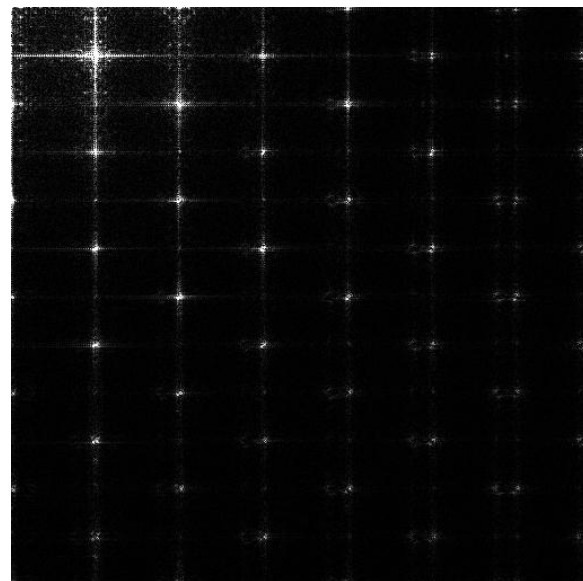
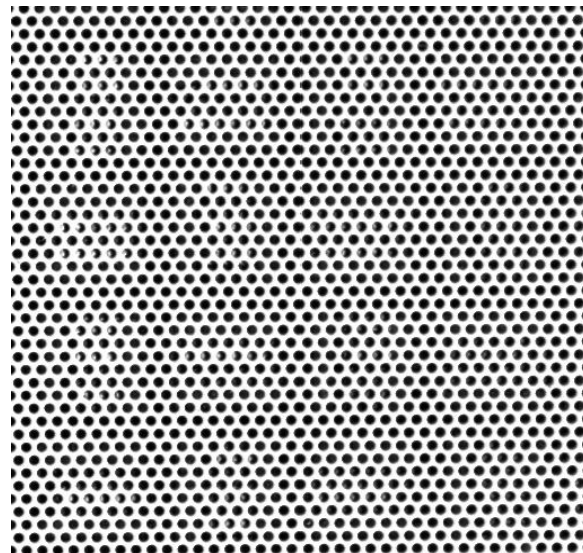
DCT的性质

- 实规范正交基
 - 基向量模长为1
- 与DFT的关系
 - DCT对应实偶函数的DFT
- 有快速算法
 - 比如类似FFT的算法
 - 如果计算某一个特定频率，还有其他更快的方式
- 能量压缩
 - 应用于JPEG压缩编码

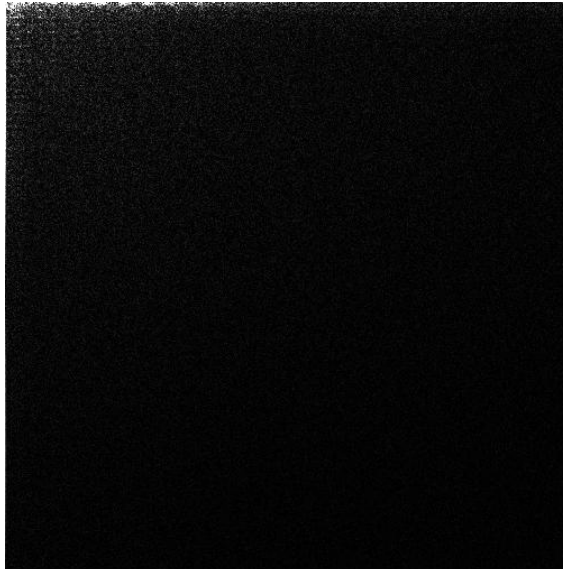
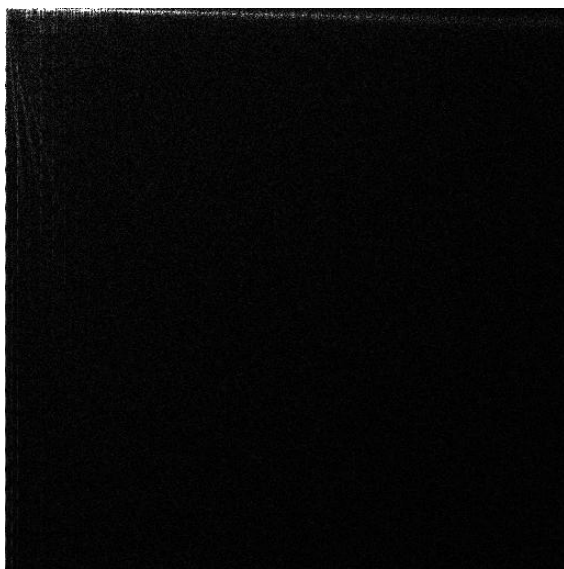
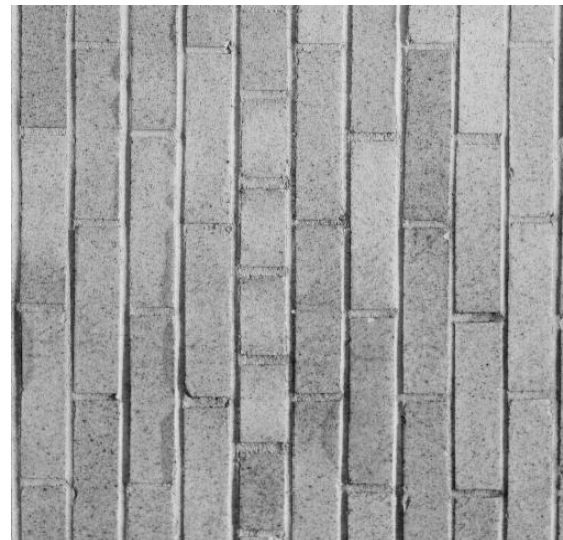
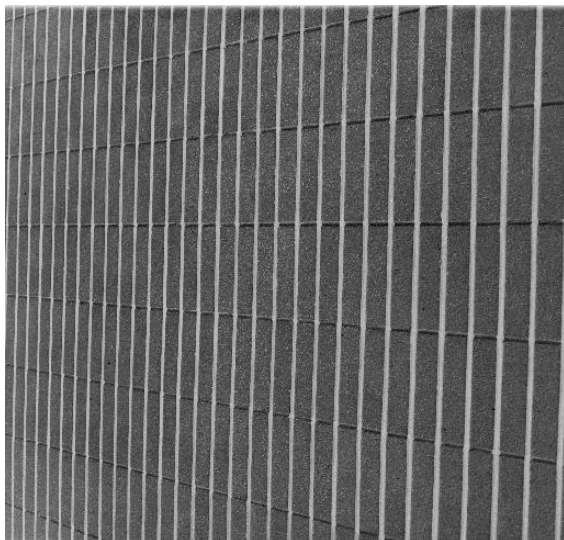
DCT变换结果示例



DCT变换结果示例



DCT变换结果示例



沃尔什/哈达玛变换

- 沃尔什 (Walsh) /哈达玛变换, 其基函数与DFT和DCT不同, 不是正弦形的, 而是方波的各种变形
- 在这类变换中, 哈达玛 (Hadamard) 变换在图象处理中应用比较广泛
- 运算简单, 只需加减运算
- 缺乏明确物理意义和较直观的解释

哈达玛变换的递推式

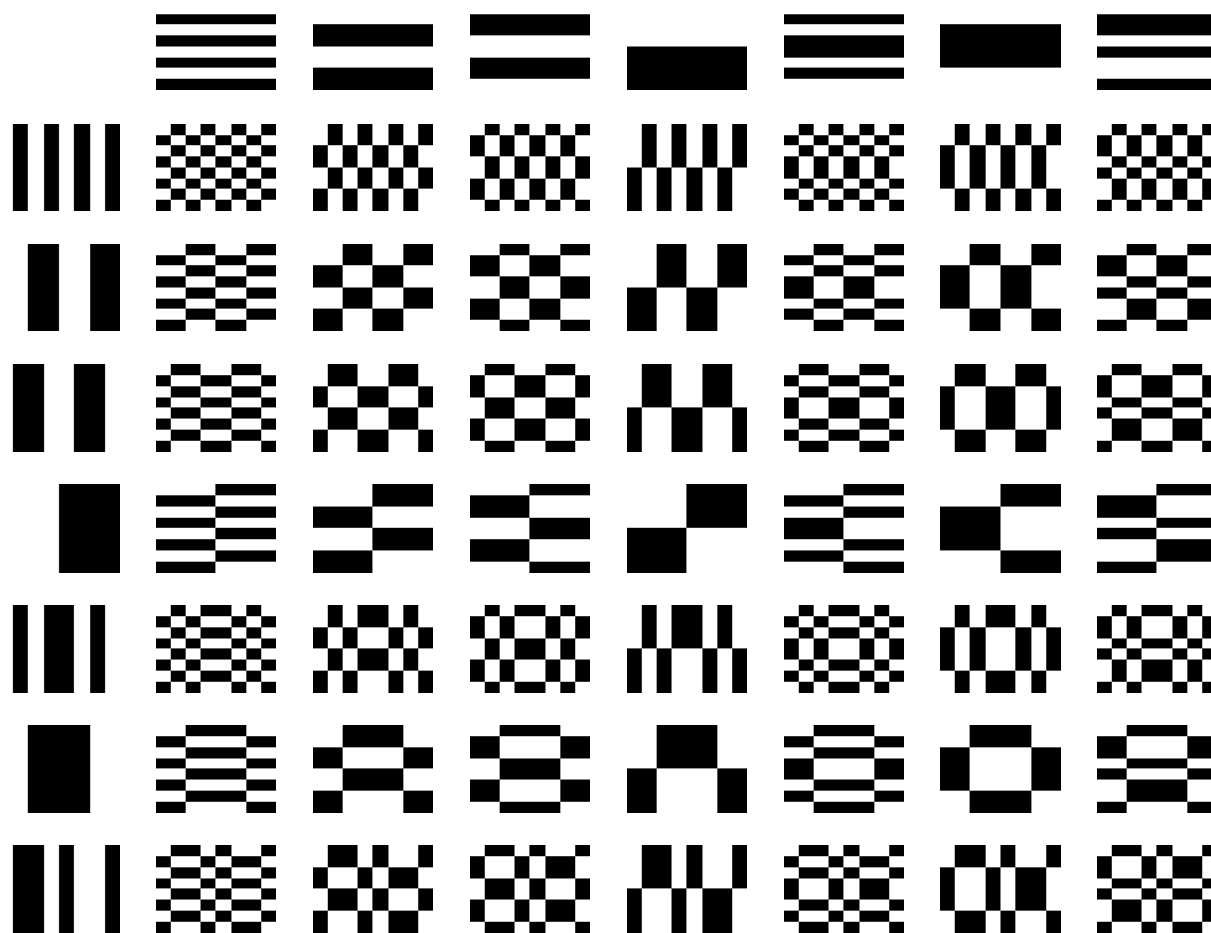
□ $2^K \times 2^K$ 哈达玛递推式：

$$H_2 = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H_{2N} = \frac{1}{\sqrt{2}} \begin{bmatrix} H_N & H_N \\ H_N & -H_N \end{bmatrix}$$

沃尔什/哈达玛变换

基
图
象





Karhunen-Loeve变换 (PCA)

- 优化目标：希望原始数据变换后的表达在每个维度上不存在（线性）相关性，因为相关性意味着数据的不同维度间不完全独立，就必然存在重复表示的信息。即：**数据不同维度的协方差为0**
- 希望由新的基所得到的数据表达的协方差矩阵中，除对角线上的方差元素外，其余所有的协方差元素全部为0（矩阵对角化）

协方差矩阵及优化目标

- 设原始数据为M个N维向量，首先将数据每个维度减去各自维度的均值，使每个维度的均值都变为0，记为矩阵X（每一列对应一个样本向量）
- 基变换矩阵记为矩阵P，则基变换后的数据可以记为：

$$Y = PX$$

- 显然，Y每个维度的均值也为0。因此，Y的协方差矩阵为：

$$\begin{aligned} D_Y &= \frac{1}{M} Y Y^T \\ &= \frac{1}{M} (PX)(PX)^T \\ &= \frac{1}{M} P X X^T P^T \\ &= P \left(\frac{1}{M} X X^T \right) P^T \\ &= P D_X P^T \end{aligned}$$

目标变换矩阵P：
能让原始数据协
方差矩阵对角化

协方差矩阵及优化目标

□ 我们知道：

- 协方差矩阵 D_X 是一个实对称矩阵
- 实对称矩阵不同特征值对应的特征向量必然正交
- 特征向量构成的变换矩阵可以使协方差矩阵对角化

□ P是协方差矩阵 D_X 的特征向量单位化后按行排列出的矩阵，其中每一行都是 D_X 的一个特征向量

□ 如果P按照特征值从大到小，将特征向量从上到下排列，则用P的前k行组成的矩阵乘以原始数据矩阵X，就得到了我们需要的降维后的数据矩阵Y

算法步骤

原始数据为M个N维向量：

1. 将原始数据按列组成N行M列的矩阵X；
2. 将X的每一行（每个维度）进行零均值化；
3. 求出协方差矩阵 $D_X = \frac{1}{M}XX^T$ ；
4. 求出 D_X 的特征值及对应的特征向量；
5. 将特征向量按对应特征值大小从上到下按行排列成矩阵，取前k行组成矩阵P；
6. $Y = PX$ 即为降维到k维后的数据。

K-L变换一例(1)

图象点序列:

(1, 1), (1, 2),

(2, 1), (2, 2), (2, 3),

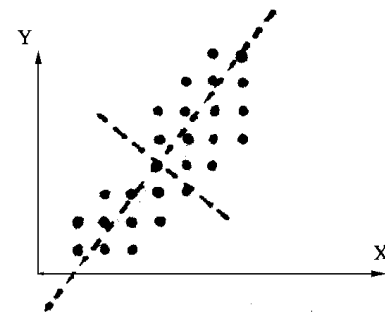
(3, 1), (3, 2), (3, 3),

(4, 2), (4, 3), (4, 4), (4, 5), (4, 6),

(5, 3), (5, 4), (5, 5), (5, 6), (5, 7),

(6, 4), (6, 5), (6, 6), (6, 7), (6, 8),

(7, 5), (7, 6), (7, 7), (7, 8)



均值
$$\begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix} = E \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 4.4 \\ 4.3 \end{bmatrix}$$

协方差
$$\Sigma_X = E \left\{ \begin{bmatrix} x - \bar{x} \\ y - \bar{y} \end{bmatrix} \begin{bmatrix} x - \bar{x} & y - \bar{y} \end{bmatrix} \right\} = \begin{bmatrix} 3.286 & 3.099 \\ 3.099 & 4.579 \end{bmatrix}$$

K-L变换一例(2)

- 在维数小时，由本征多项式为零求协方差矩阵的本征值：

$$\begin{bmatrix} 3.268 - \lambda & 3.099 \\ 3.099 & 4.579 - \lambda \end{bmatrix} = \lambda^2 - 7.865\lambda + 5.443 = 0$$

$$\lambda_1 = 7.098 \quad \lambda_2 = 0.768$$

- 再把本征值代入 $\Sigma_X \vec{\phi}_i = \lambda_i \vec{\phi}_i$ ，求出特征矢量：

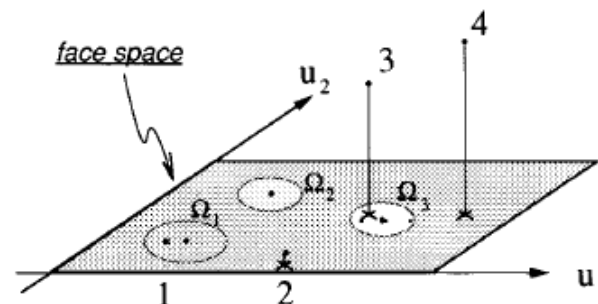
$$\vec{\phi}_1 = \begin{bmatrix} 1 \\ 1.23 \end{bmatrix} \quad \vec{\phi}_2 = \begin{bmatrix} -1.23 \\ 1 \end{bmatrix}$$

- 把相互垂直的二特征矢量作为新的坐标，新坐标的主轴方向为所变换数据方差最大的方向

K-L变换应用实例 —— 人脸识别



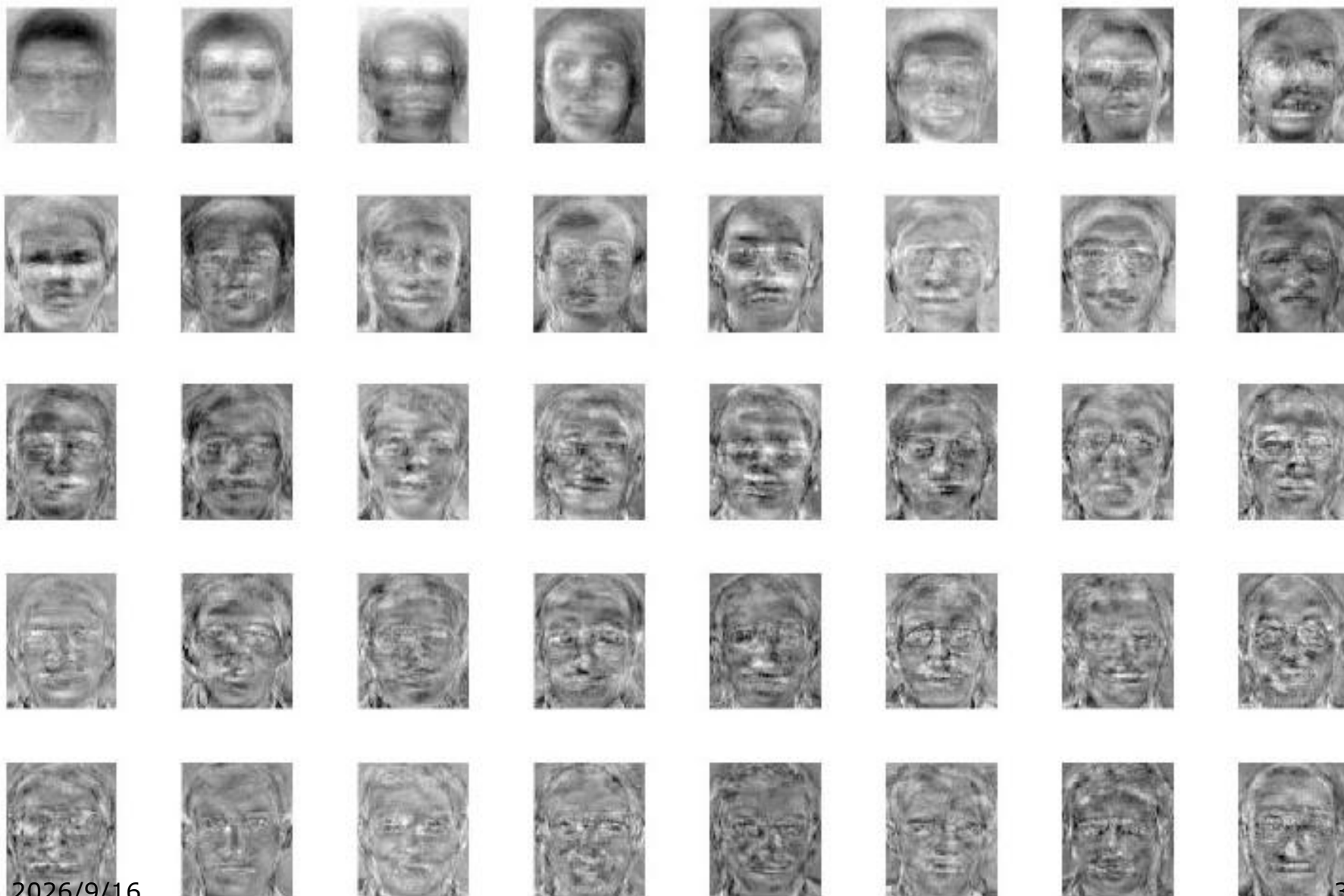
1. 把每一幅人脸列化，视为随机向量 F 的不同实现。
2. 估计 F 协方差矩阵 C ，并计算其特征值特征向量。
(C 是半正定矩阵，维数不大于图像数，对应不同特征值的特征向量正交，特征脸)
3. 选择对应少量最大特征值的特征向量组成特征脸空间
4. 每张脸映射为特征脸空间的点(计算内积)，以其坐标作为特征向量。
5. 采用模式识别方法，进行分类识别。
(如欧式距离)



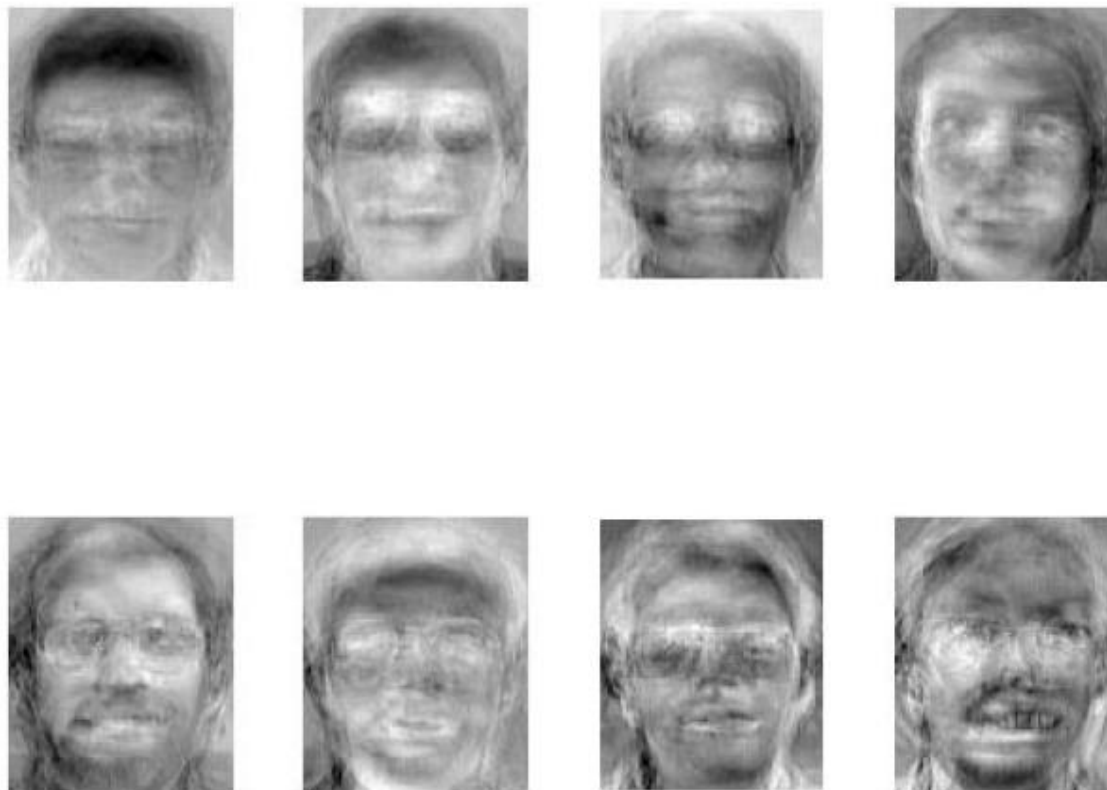
人脸库



特征脸



特征脸空间 (top 8)



小波变换 (DWT)

□ 傅立叶变换

- 变换之后丢掉了时域信息
- 无法定位相应的频率峰值的位置

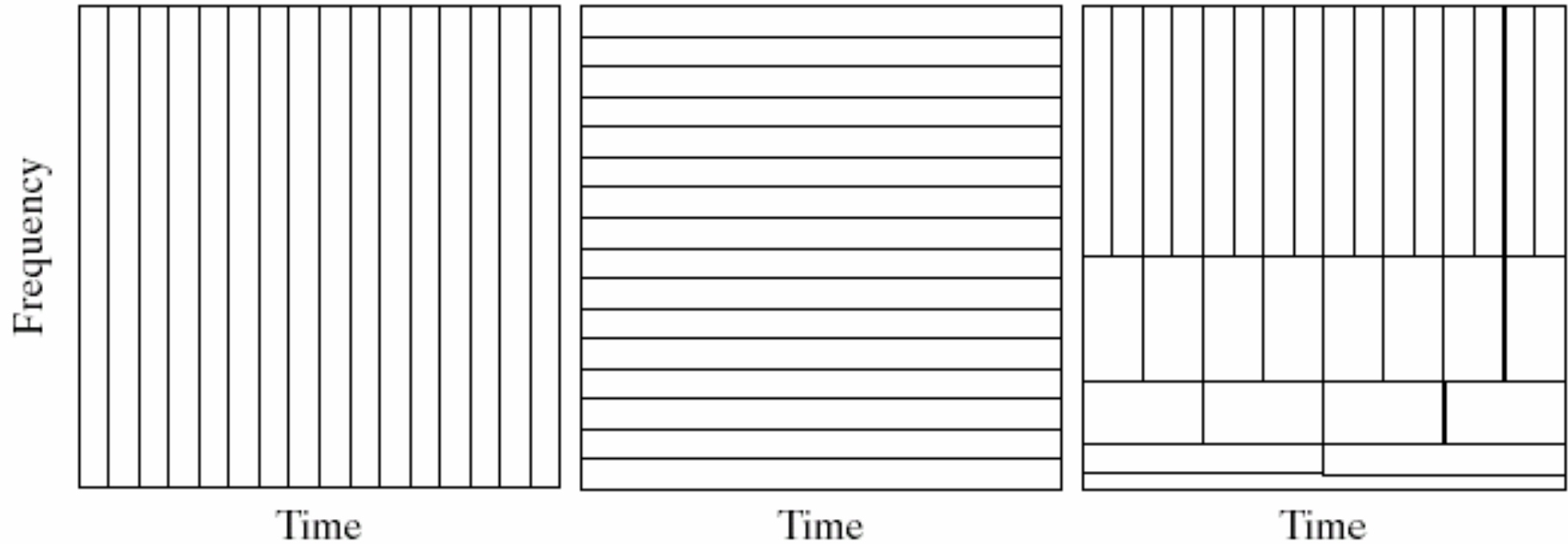
□ 时频域分析

- 小波变换在二维时频空间分析信号

□ 变换

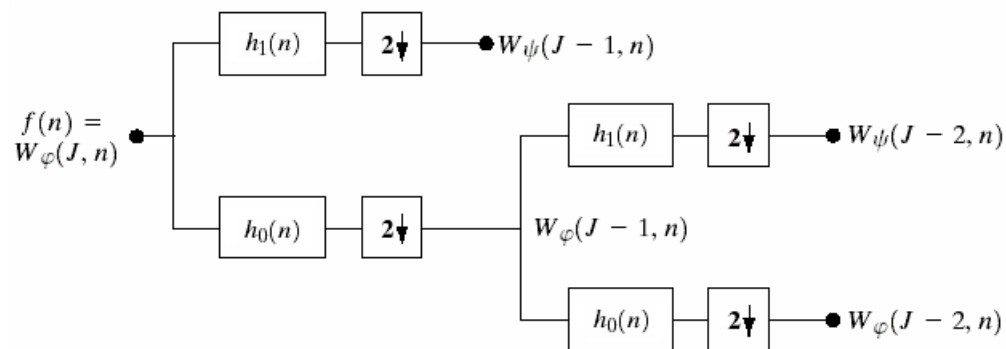
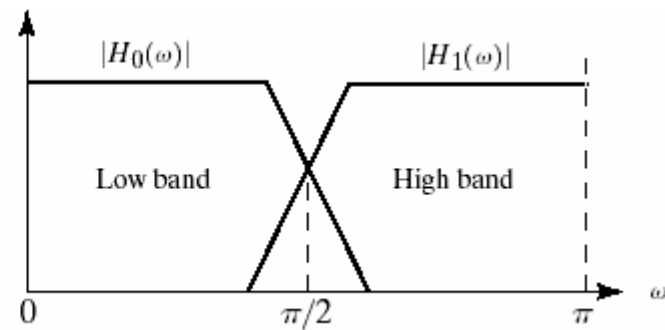
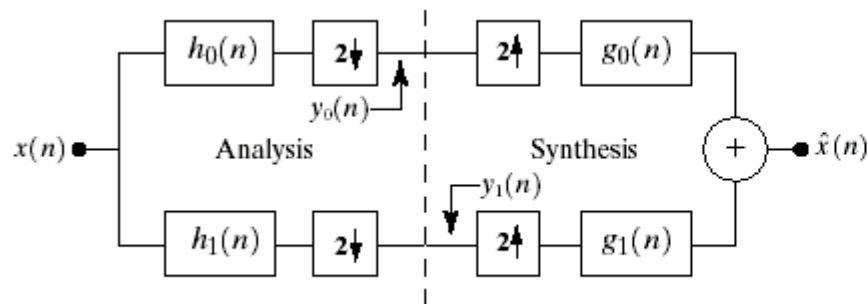
- 理想的基本小波是过程很短的振荡函数
- 过程很短：检测时域信号；振荡：检测频域信号
- 如同傅立叶变换有连续、离散的变换，小波也有连续、离散的变换

1维离散小波变换

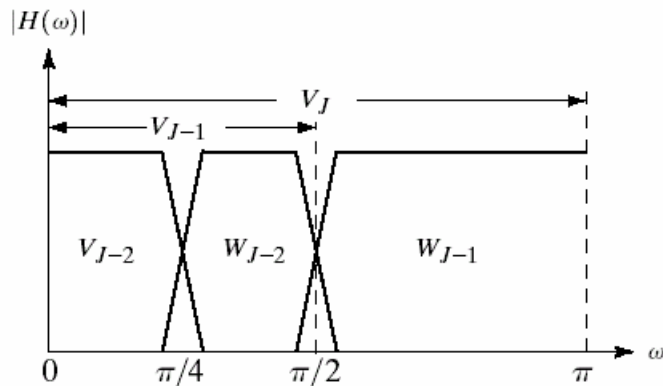


时频铺叠（从左到右：Dirac、Fourier、wavelet）

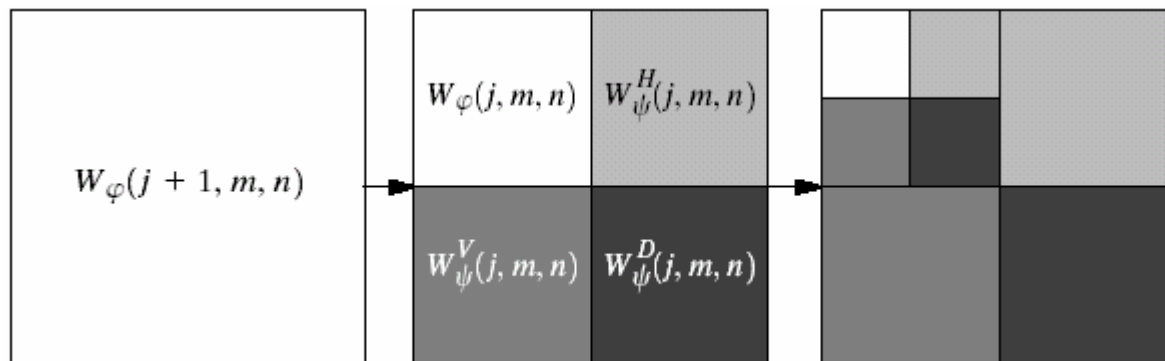
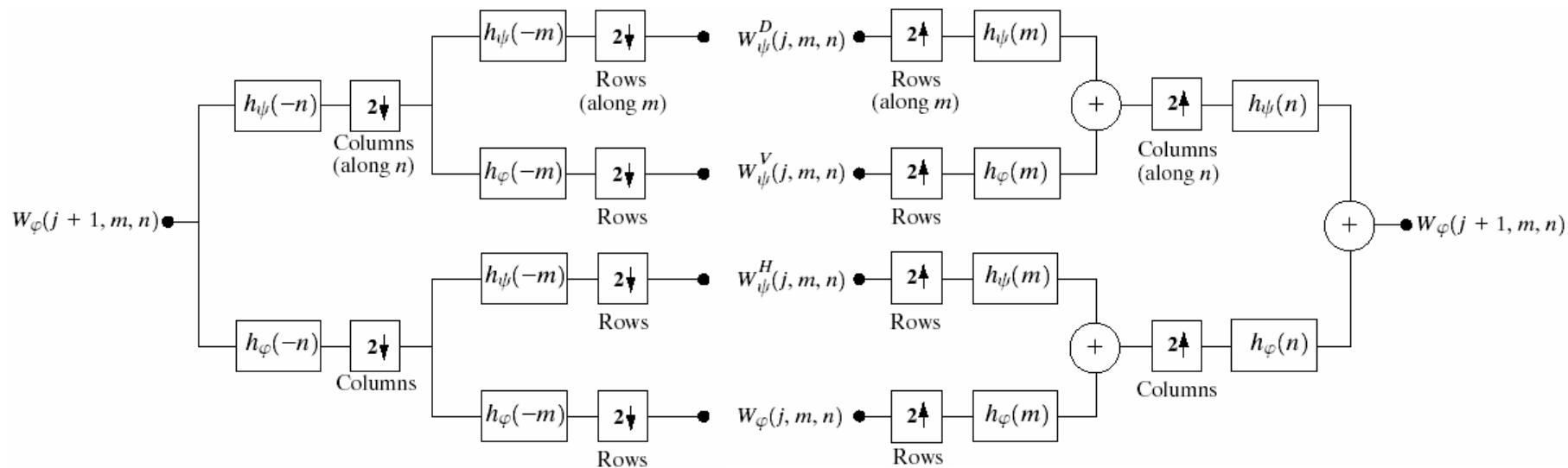
1维离散小波变换



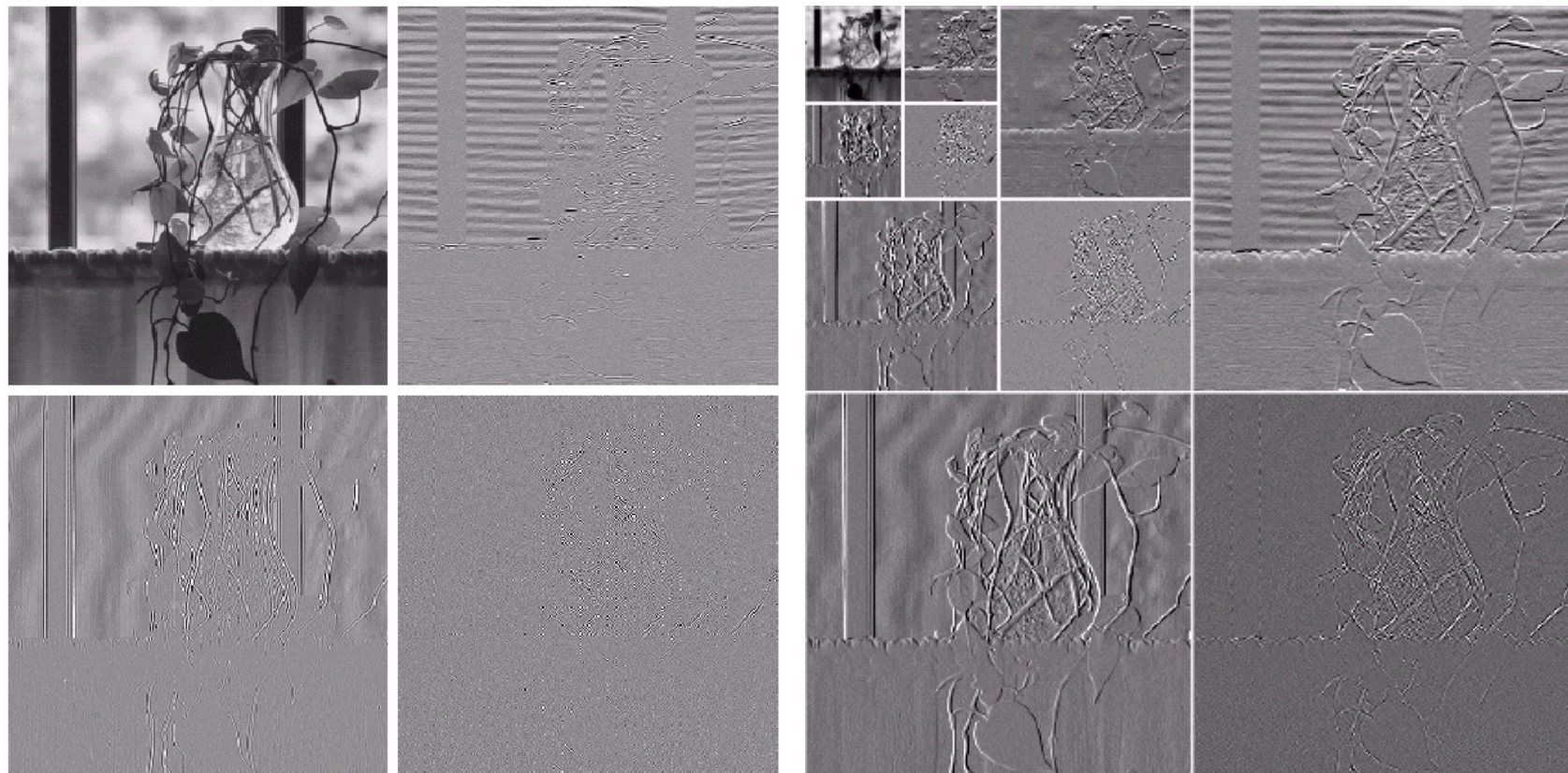
对低频进行
进一步分解



2维离散小波变换

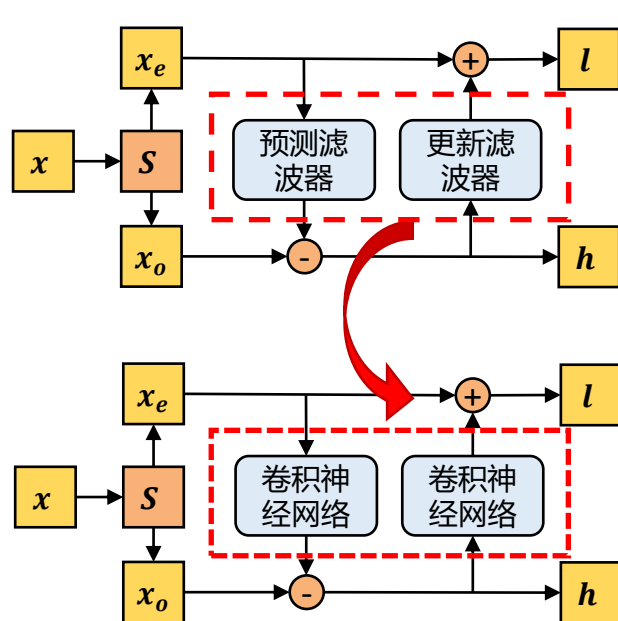


实例



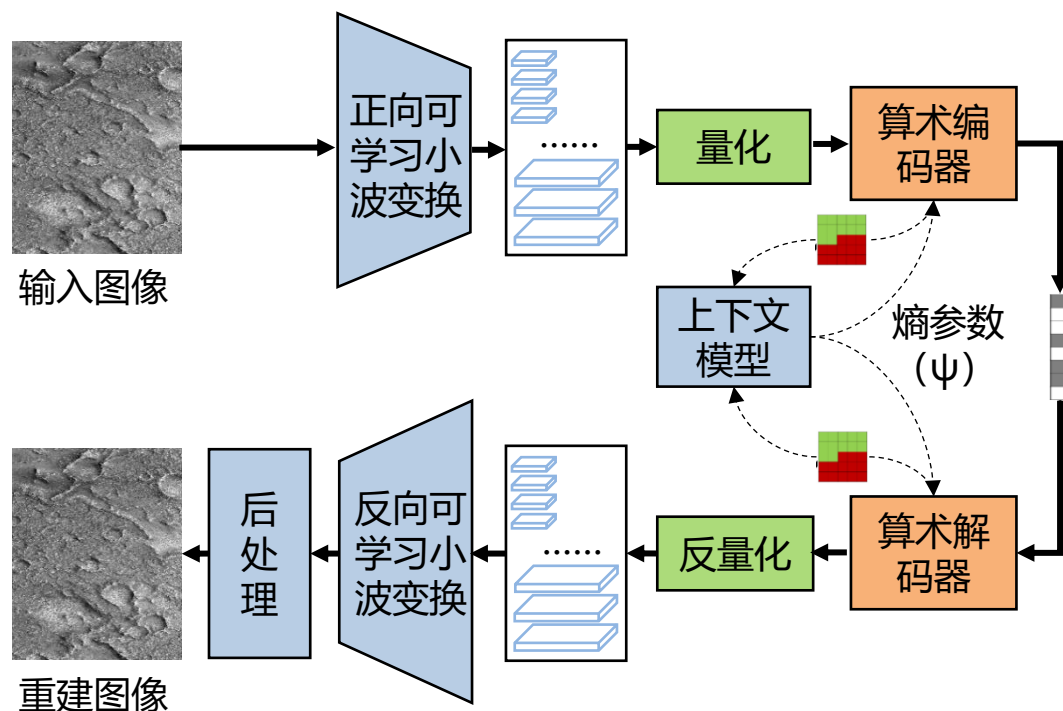
可学习类小波变换

- 将传统小波提升结构中的算子替换为非线性神经网络，建立可学习类小波变换的端到端图像编码框架



传统小波（上）和可学习小波（下）

类小波变换端到端编码框架



Ma, Haichuan, et al. "End-to-end optimized versatile image compression with wavelet-like transform." IEEE Transactions on Pattern Analysis and Machine Intelligence 44.3 (2020): 1247-1263.

□ 形态学

■ 二值形态学

- ✓ 基本定义
- ✓ 基本运算
- ✓ 实用算法

二值形态学

□ 基本集合定义

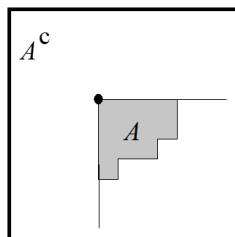
■ 集合：用大写字母表示，空集记为 $\emptyset$

■ 元素：用小写字母表示

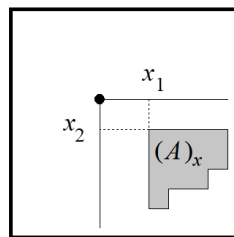
■ 子集：

■ 并集：

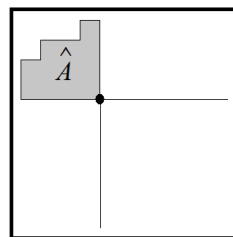
■ 交集：



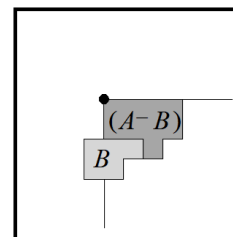
(a)



(b)



(c)



(d)

■ 补集： $A^c = \{x|x \notin A\}$

■ 位移： $(A)_x = \{y|y = a + x, a \in A\}$

■ 映像： $\hat{A} = \{x|x = -a, a \in A\}$

■ 差集： $A - B = \{x|x \in A, x \notin B\} = A \cap B^c$



二值形态学基本运算

□ 集合运算

- A为图象集合，B 为结构元素（集合）
- 数学形态学运算是用 B 对 A 进行操作
- 结构元素要指定1个原点（参考点）

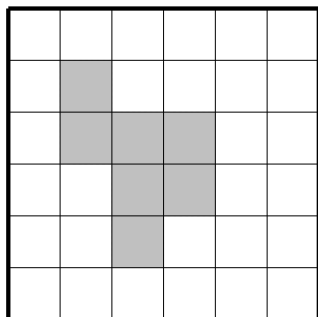
膨胀和腐蚀

□ 膨胀

- 膨胀的算符为 $\oplus$

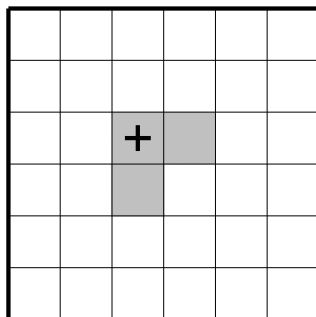
$$A \oplus B = \{x \mid [(\hat{B})_x \cap A] \neq \emptyset\}$$

集合 A



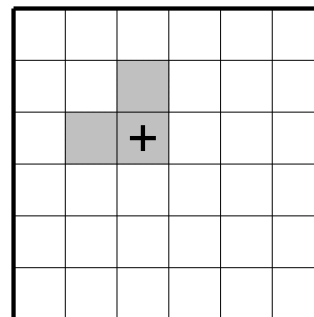
(a)

结构元素 B



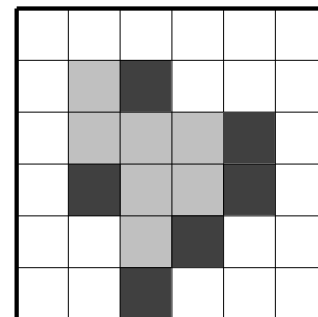
(b)

B 的映象



(c)

集合 $A \oplus B$



(d)

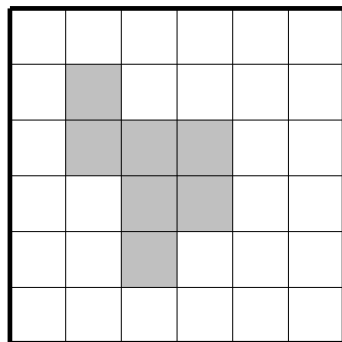
膨胀和腐蚀

□ 腐蚀

- 腐蚀的算符为 $\ominus$

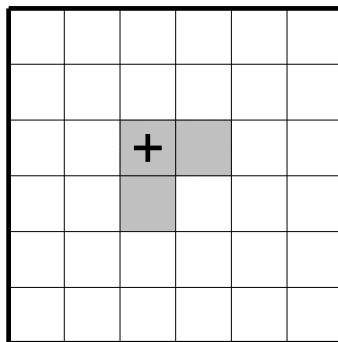
$$A \ominus B = \{x | (B)_x \subseteq A\}$$

集合 A



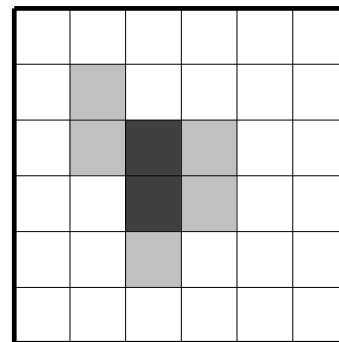
(a)

结构元素 B



(b)

集合 $A \ominus B$



(c)

膨胀和腐蚀

□ 原点不包含在结构元素中的膨胀和腐蚀

■ 原点包含在结构元素中

✓ 膨胀运算: $A \subseteq A \oplus B$

✓ 腐蚀运算: $A \ominus B \subseteq A$

■ 原点不包含在结构元素中

✓ 膨胀运算: $A \not\subseteq A \oplus B$

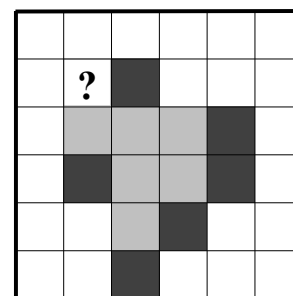
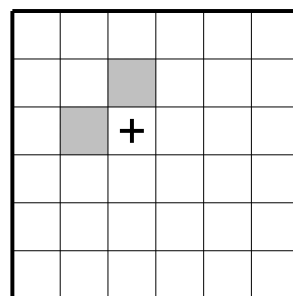
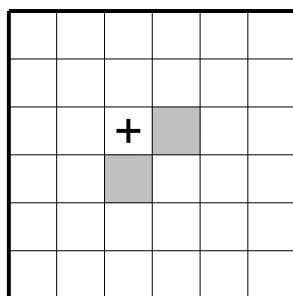
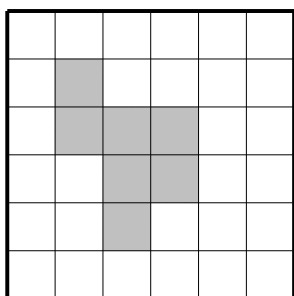
✓ 腐蚀运算: $A \ominus B \subseteq A$, 或 $A \ominus B \not\subseteq A$

膨胀和腐蚀

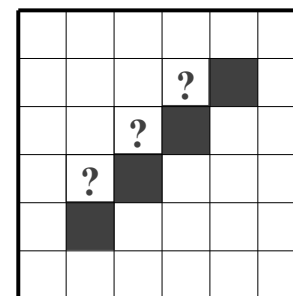
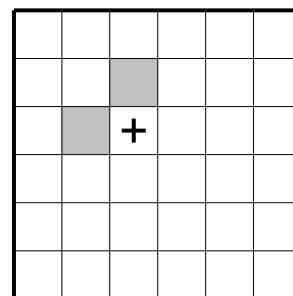
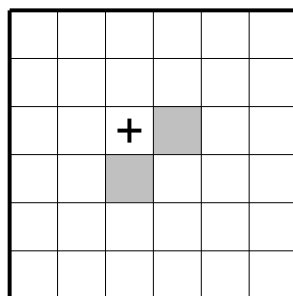
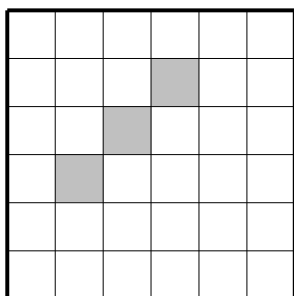
□ 原点不包含在结构元素中的时的膨胀运算

$$A \not\subset A \oplus B$$

$$A \oplus B = \{x \mid [(\hat{B})_x \cap A] \neq \emptyset\}$$



A 在膨胀中自身完全消失了

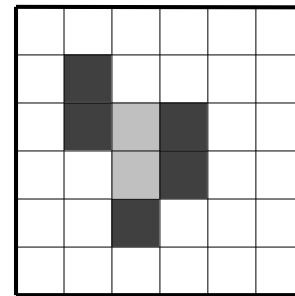
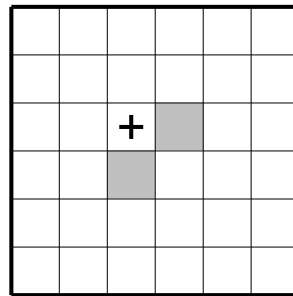
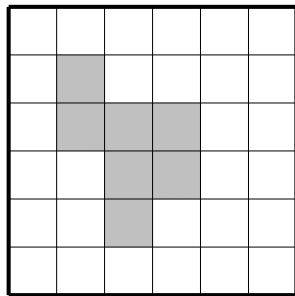


膨胀和腐蚀

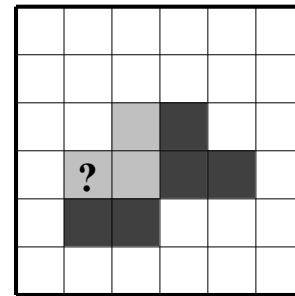
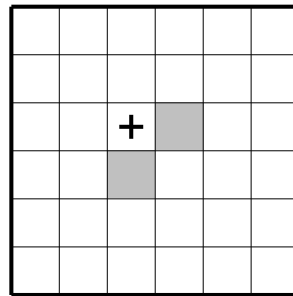
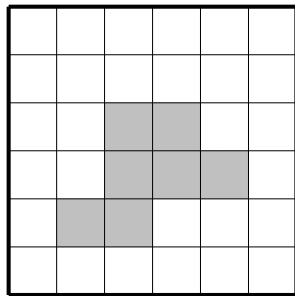
□ 原点不包含在结构元素中的时的腐蚀运算

$$A \ominus B \subseteq A$$

$$A \ominus B = \{x | (B)_x \subseteq A\}$$



$$A \ominus B \not\subseteq A$$



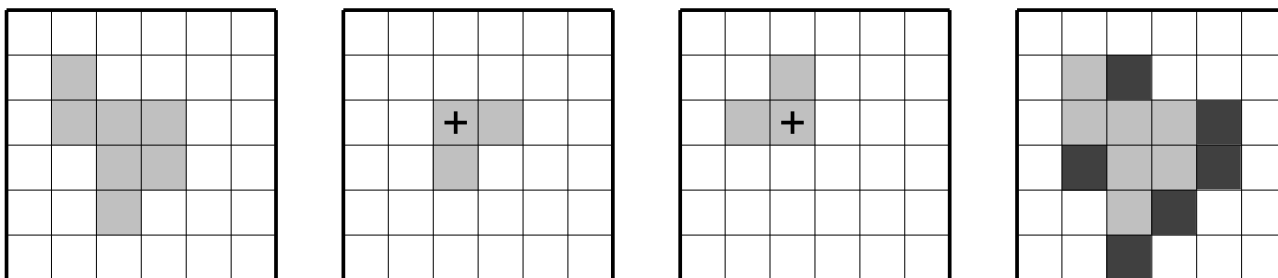
膨胀和腐蚀

□ 用向量运算实现膨胀和腐蚀

$$A \oplus B = \{x | x = a + b, \text{ 对于任意 } a \in A \text{ 和 } b \in B\}$$

$$A = \{(1, 1), (1, 2), (2, 2), (3, 2), (2, 3), (3, 3), (2, 4)\}$$

$$B = \{(0, 0), (1, 0), (0, 1)\}$$



$$A \oplus B = \{(1, 1), (2, 1), (1, 2), (2, 2), (3, 2), (4, 2), \\ (1, 3), (2, 3), (3, 3), (4, 3), (2, 4), (3, 4), (2, 5)\}$$

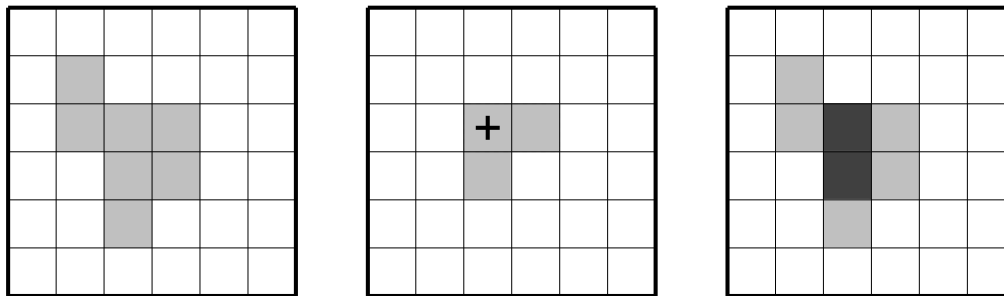
膨胀和腐蚀

□ 用向量运算实现膨胀和腐蚀

$$A \ominus B = \{x | (x + b) \in A \text{ 对每一个 } b \in B\}$$

$$A = \{(1, 1), (1, 2), (2, 2), (3, 2), (2, 3), (3, 3), (2, 4)\}$$

$$B = \{(0, 0), (1, 0), (0, 1)\}$$



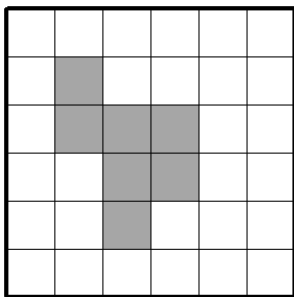
$$A \ominus B = \{(2, 2), (2, 3)\}$$

膨胀和腐蚀

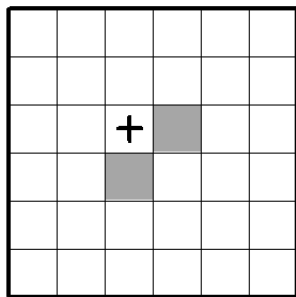
□ 用位移运算实现膨胀和腐蚀

按每个 b 来位移 A 并把结果或（OR）起来

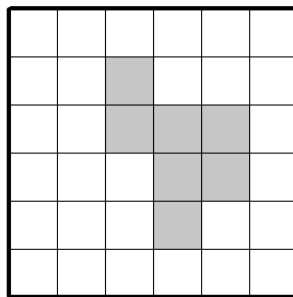
$$A \oplus B = \bigcup_{b \in B} (A)_b$$



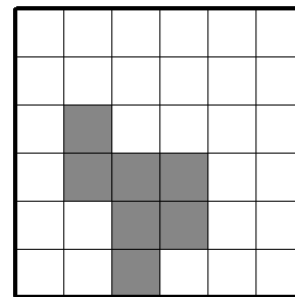
(a)



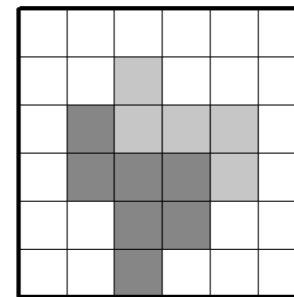
(b)



(c)



(d)



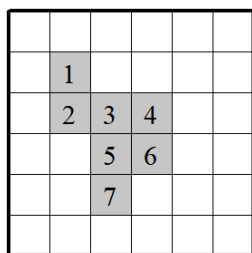
(e)

膨胀和腐蚀

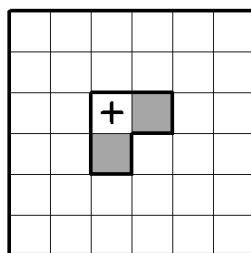
□ 用位移运算实现膨胀和腐蚀

按每个a来位移B并把结果或（OR）起来

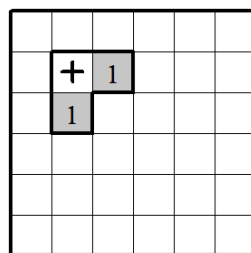
$$A \oplus B = \bigcup_{a \in A} (B)_a$$



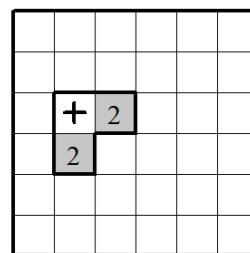
(a)



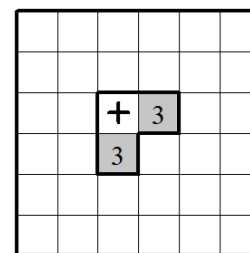
(b)



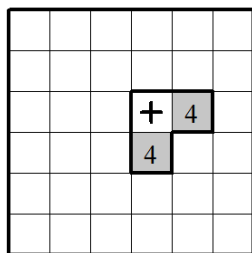
(c)



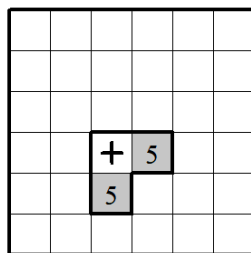
(d)



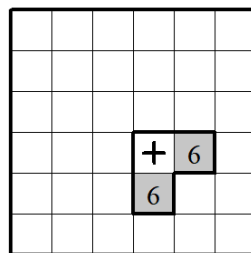
(e)



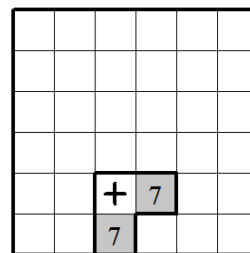
(f)



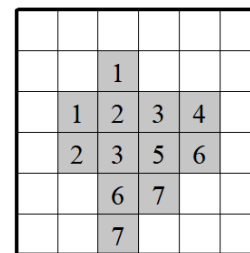
(g)



(h)



(i)



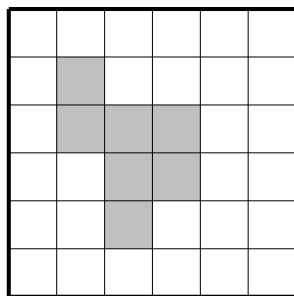
(j)

膨胀和腐蚀

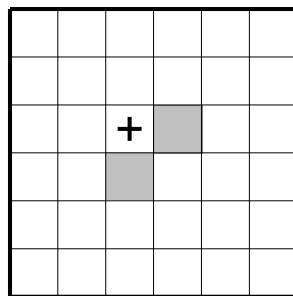
□ 用位移运算实现膨胀和腐蚀

按每个b来负位移A并把结果交（AND）起来

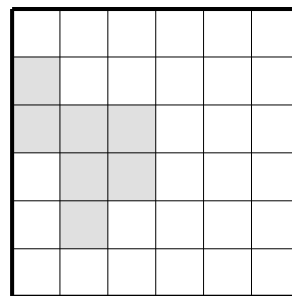
$$A \ominus B = \bigcap_{b \in B} (A)_{-b}$$



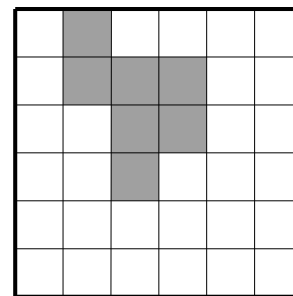
(a)



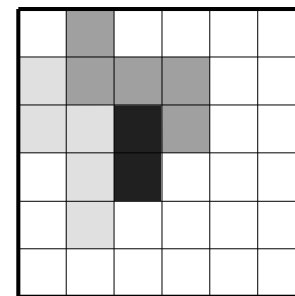
(b)



(c)



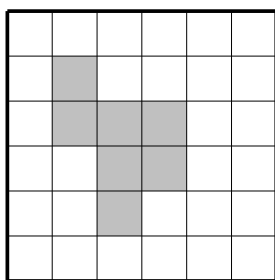
(d)



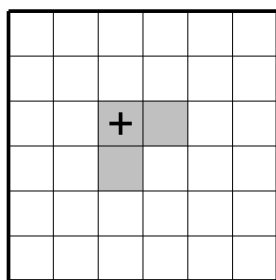
(e)

□ 膨胀和腐蚀的对偶性

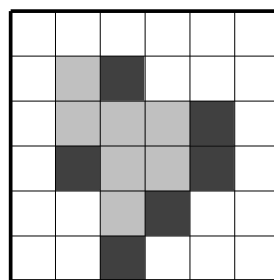
$$(A \oplus B)^c = A^c \ominus \hat{B} \quad (A \ominus B)^c = A^c \oplus \hat{B}$$



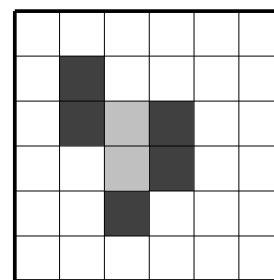
(a)



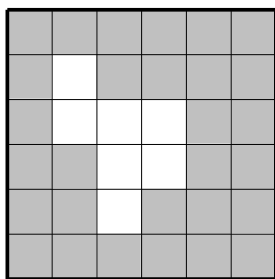
(b)



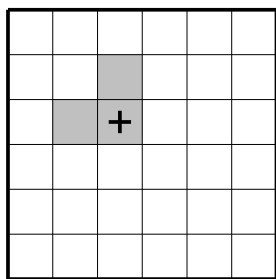
(c)



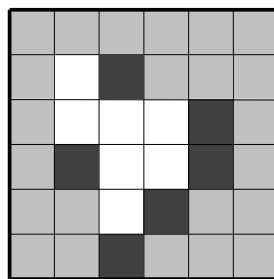
(d)



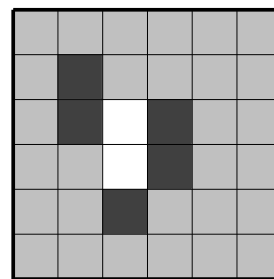
(e)



(f)



(g)



(h)

膨胀和腐蚀

□ 膨胀和腐蚀的对偶性

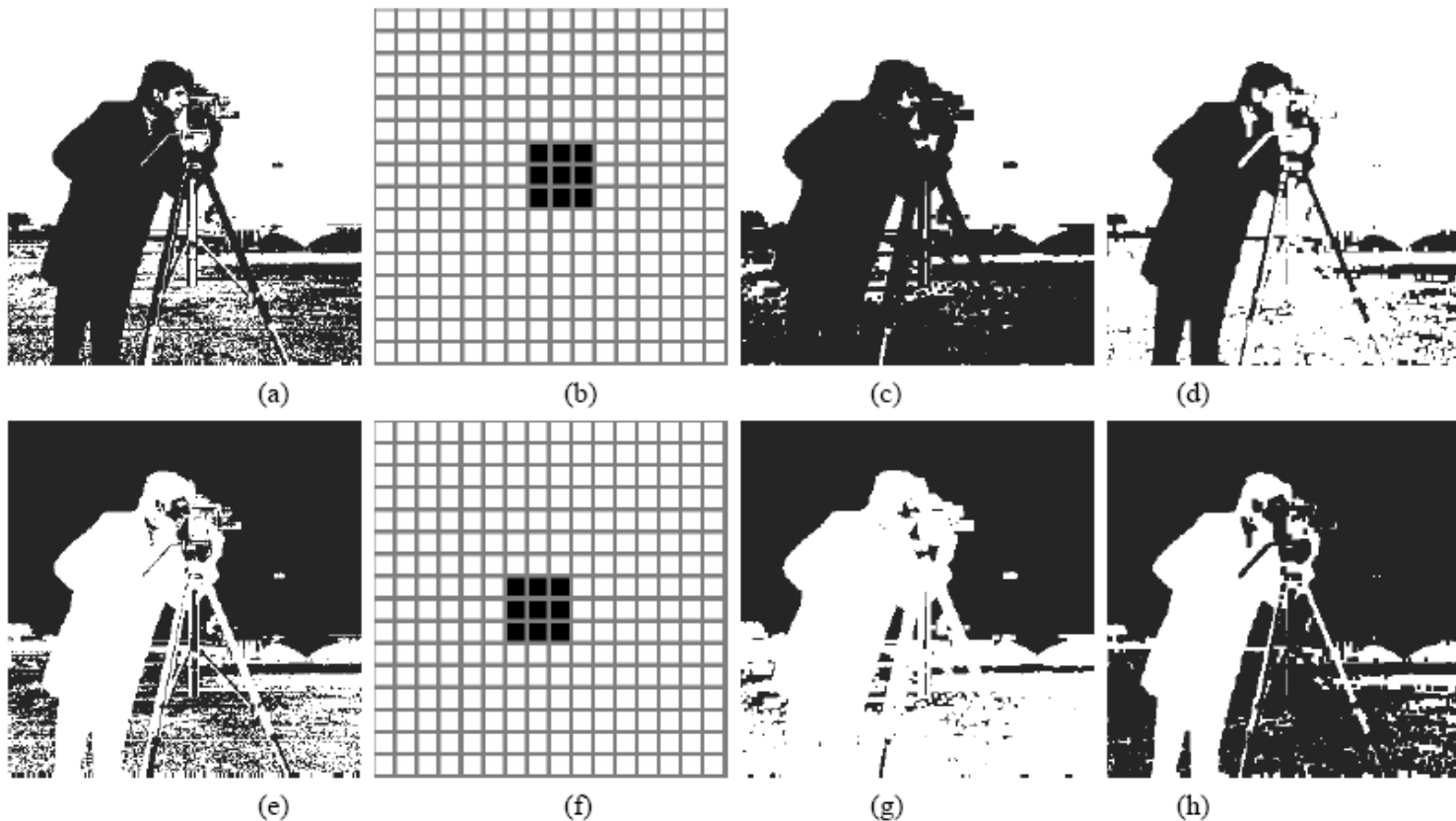


图 14.2.12 膨胀和腐蚀的对偶性验证实例

开启和闭合

□ 开启和闭合定义

- 膨胀和腐蚀并不互为逆运算
- 它们可以级连结合使用
- 开启：先对图象进行腐蚀然后膨胀其结果

$$A \circ B = (A \ominus B) \oplus B$$

- 闭合：先对图象进行膨胀然后腐蚀其结果

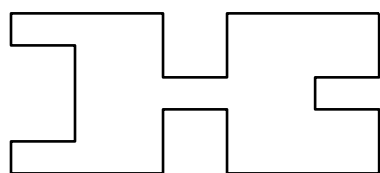
$$A \cdot B = (A \oplus B) \ominus B$$

- 开启和闭合不受原点是否在结构元素之中的影响

开启和闭合

□ 开启和闭合定义

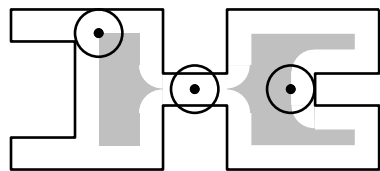
- 开启运算可以把比结构元素小的突刺滤掉
- 闭合运算可以把比结构元素小的缺口或孔填充上



(a)



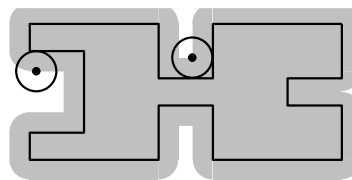
(b)



(c)



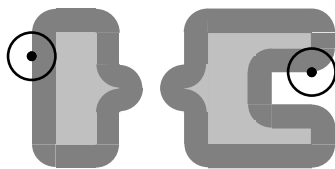
(d)



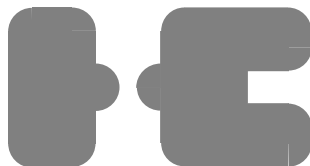
(e)



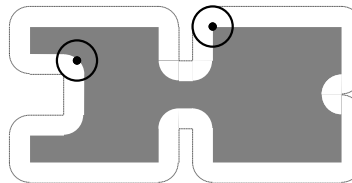
(f)



(g)



(h)



(i)



(j)

开启

闭合

开启和闭合

□ 开启和闭合定义



原图



(a)



(b)

图 14.2.14 开启和闭合实例

开启和闭合

□ 开启和闭合的对偶性

- 开启和闭合也具有对偶性

$$(A \circ B)^c = A^c \cdot \hat{B}$$

$$(A \cdot B)^c = A^c \circ \hat{B}$$

$$(A \circ B)^c = [(A \ominus B) \oplus B]^c = (A \ominus B)^c \ominus \hat{B} = A^c \oplus \hat{B} \ominus \hat{B} = A^c \cdot \hat{B}$$

$$(A \cdot B)^c = [(A \oplus B) \ominus B]^c = (A \oplus B)^c \oplus \hat{B} = A^c \ominus \hat{B} \oplus \hat{B} = A^c \circ \hat{B}$$

击中-击不中变换

□ 击中-击不中变换

- 形状检测的一种基本工具
- 对应两个操作，所以用到两个结构元素
- 设A为原始图象，E和F为一对不重合的集合

$$A \uparrow (E, F) = (A \ominus E) \cap (A^c \ominus F) = (A \ominus E) \cap (A \oplus F)^c$$

E : 击中结构元素

F : 击不中结构元素

击中-击不中变换

□ 击中-击不中变换

$$A \uparrow (E, F) = (A \ominus E) \cap (A^c \ominus F) = (A \ominus E) \cap (A \oplus F)^c$$

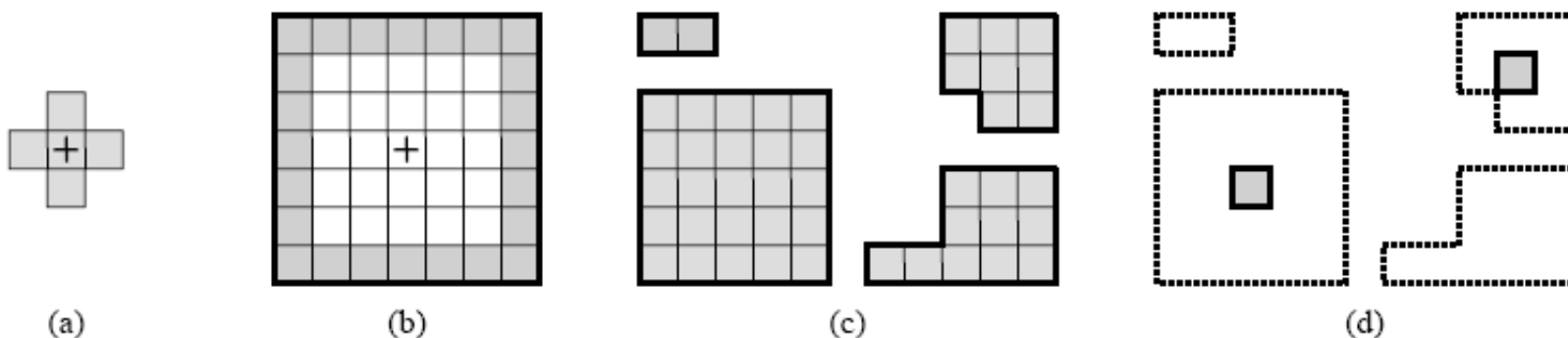


图 14.3.1 击中-击不中变换示例

- (a): 击中结构元素 (b): 击不中结构元素
(c): 原始图像 (d): 变换结果

击中-击不中变换

□ 击中-击不中变换 ((e)和(f)来自于别的变换)

击中变换: $[1 \ 1 \ 1]$

击不中变换: $\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$

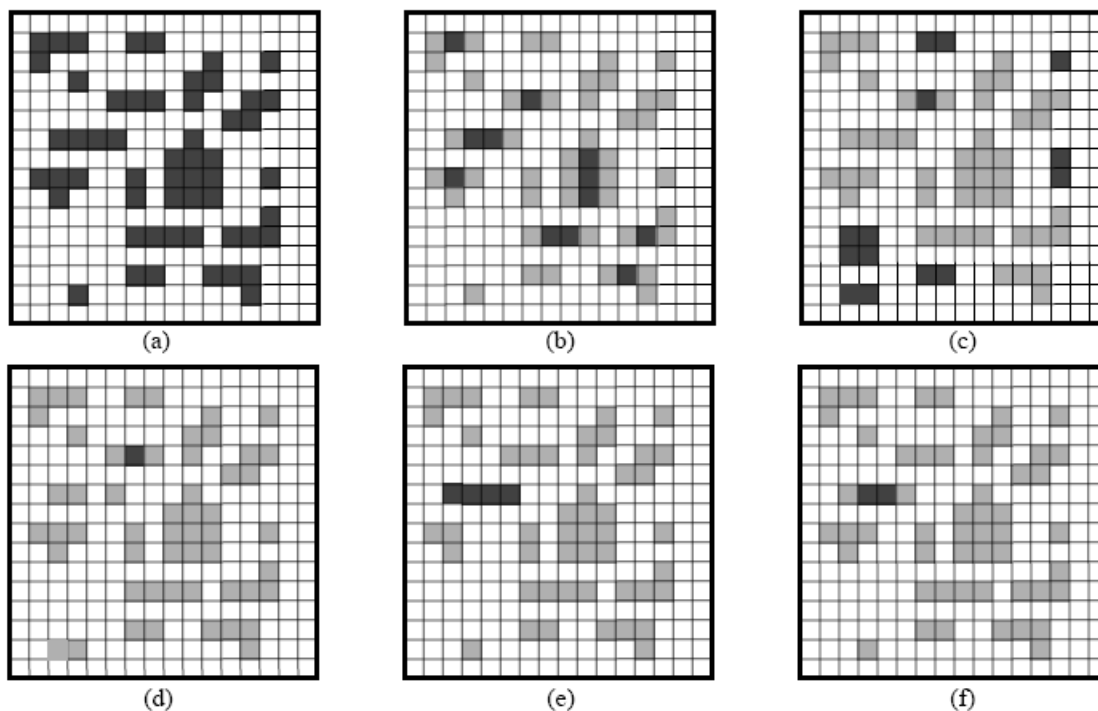
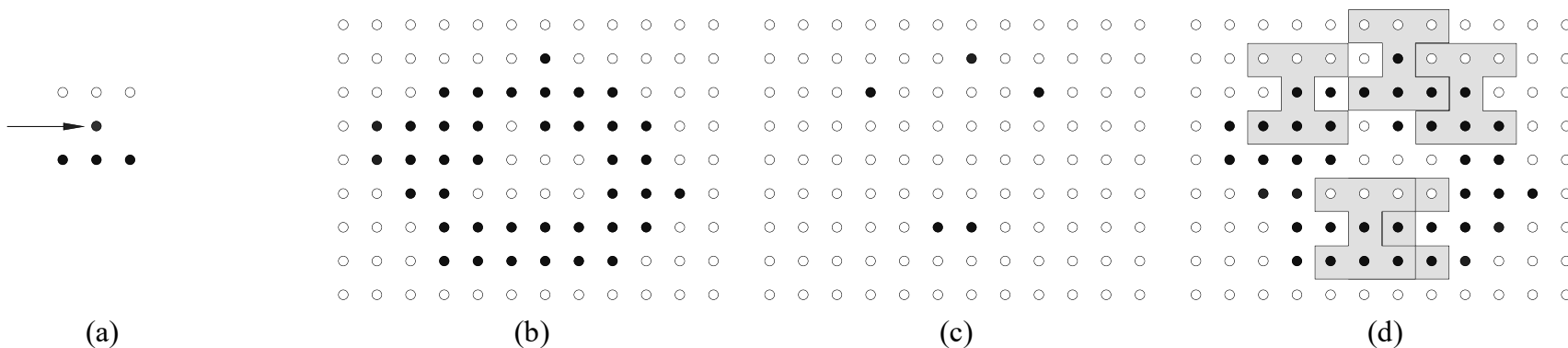


图 14.3.2 利用击中-击不中算子以提取包含水平方向上有连续 3 个像素的线段

击中-击不中变换

- 击中-击不中算子中的击中模板与击不中模板不重合，可以被结合成一个结构元素，1对应击中模板，0对应击不中模板，X表示不关心的像素
- 击中-击不中变换中的结构元素
 - $A \uparrow B$ 的结果中仍保留的目标象素对应在 A 中其邻域与结构元素 B 对应的象素

$$A \uparrow B = (A \ominus B_o) \cap (A^c \ominus B_b)$$





二值形态学实用算法

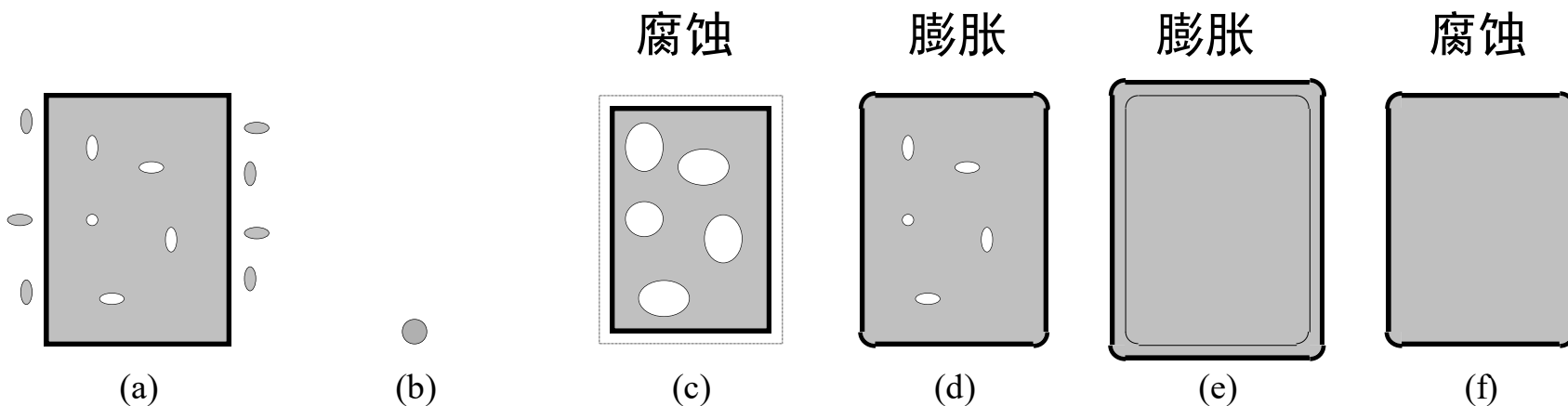
- ☐ 噪声滤除
- ☐ 目标检测
- ☐ 边界提取
- ☐ 区域填充

二值形态学实用算法-I

□ 噪声滤除

■ 先开启后闭合

$$\{[(A \ominus B) \oplus B] \oplus B\} \ominus B = (A \circ B) \cdot B$$



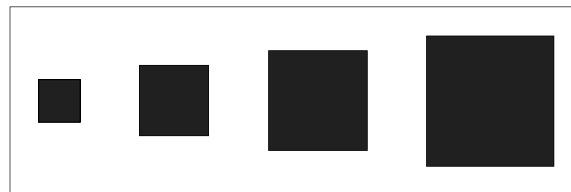
二值形态学实用算法-II

□ 目标检测（击中击不中变换）

■ 3×3 , 5×5 , 7×7 和 9×9 的实心正方形

3×3 实心正方形

9×9 方框



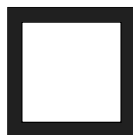
(a)

(b) : E

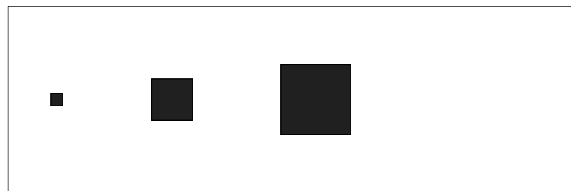
(c) : F



(b)



(c)



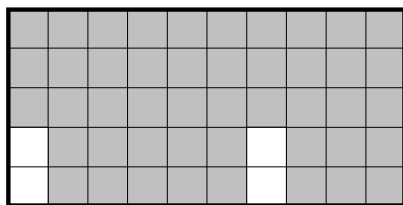
(d)

二值形态学实用算法-III

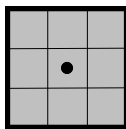
□ 边界提取

- 先用1个结构元素B腐蚀 A，再求取腐蚀结果和A的差集就可得到边界 $\beta(A)$

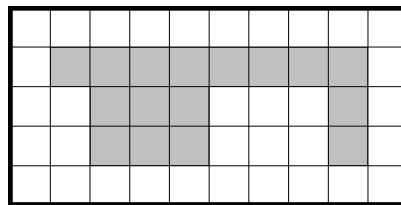
$$\beta(A) = A - (A \ominus B)$$



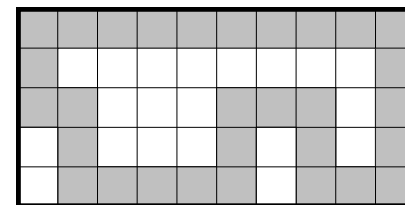
(a)



(b)



(c)



(d)

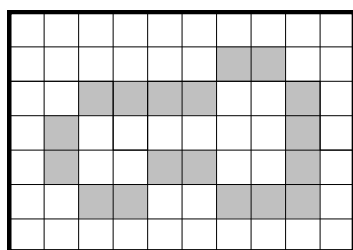
结构元素是8-连通的，而所得到的边界是4-连通的

二值形态学实用算法-IV

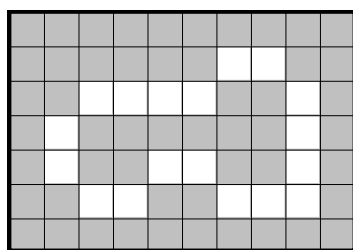
□ 区域填充

$$X_k = (X_{k-1} \oplus B) \cap A^c \quad k = 1, 2, 3, \dots$$

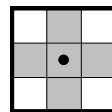
取内部一个点，按照模板膨胀，取交集，迭代多次；最后与（a）取并集



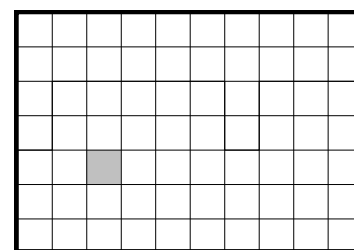
(a)



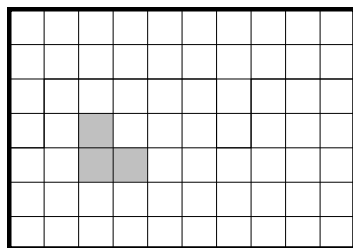
(b)



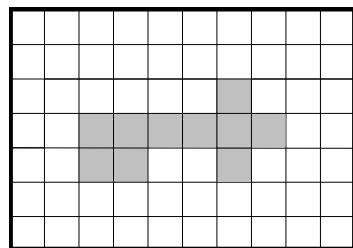
(c)



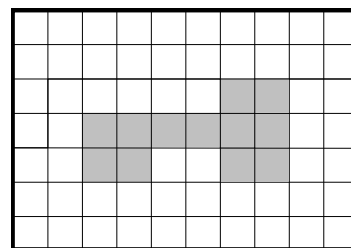
(d)



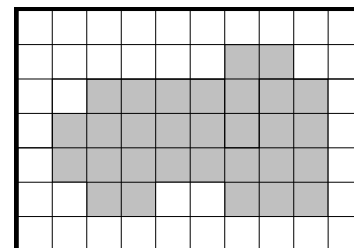
(e)



(f)



(g)



(h)

结构元素是4-连通的，而原填充的边界是8-连通的