第7章 目标表达

前言

- 图像的分割产生了一系列代表图像中感兴趣目标的连通区域，即目标
- 对这些目标采用合适的数据结构表达(representation)

  1) 节省空间；2) 利于特征选取；3) 便于特征计算
- 以恰当的形式描述(description)其特征，以便进行后续的图像分析

  1) 通用性：平移/旋转/尺度/仿射

  2) 有效性：针对后续分析

表达和描述

- 表达 VS 描述
  - 表达和描述的侧重不同（数据结构 VS 特征）
  - 表达限制了描述的精度和选择; 描述又决定了应用中采用表达的方式
  - 表达和描述抽象的程度不同，但其分别的界限是相对的

- 表达和描述的分类：
  - 内部(internal): 侧重于目标区域信息，如纹理、颜色—反射性质
  - 外部(external): 侧重于边界形状信息，如周长、曲率
  - 其它视角：局部/整体、空域/变换域

7.1 基于边界的表达

基于边界象素点进行

7.1.1 技术分类

7.1.2 链码

7.1.3 边界段和凸包

7.1.4 边界标记

7.1.5 多边形

7.1.6 地标点

7.1.1 技术分类

（1）　参数边界：将目标的轮廓线表示为参数曲线

（2）　边界点集合：各点间没有顺序

（3）　曲线逼近：用几何基元近似地逼近

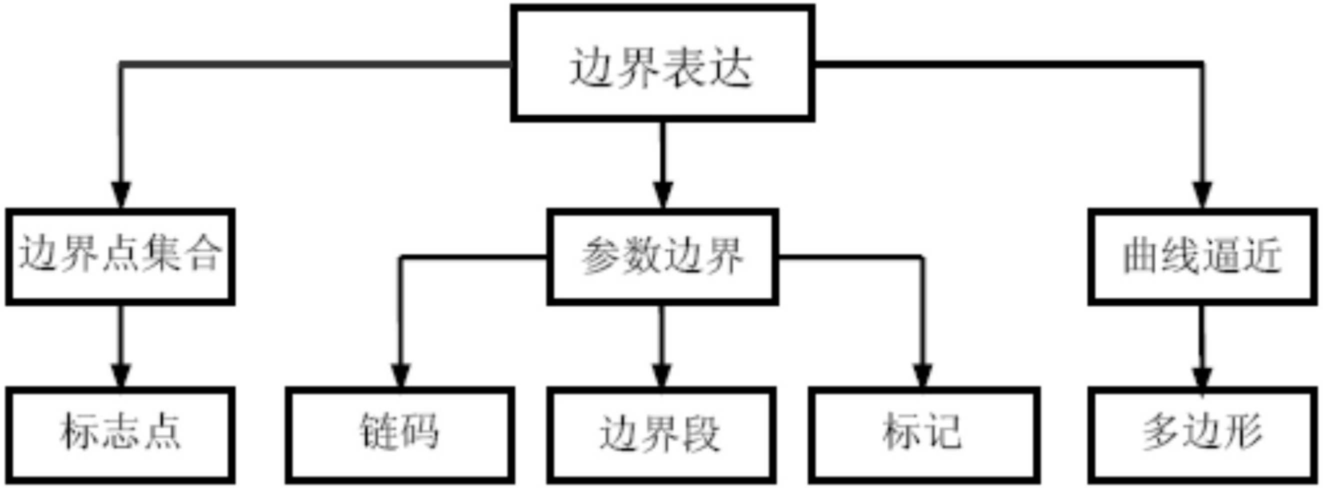
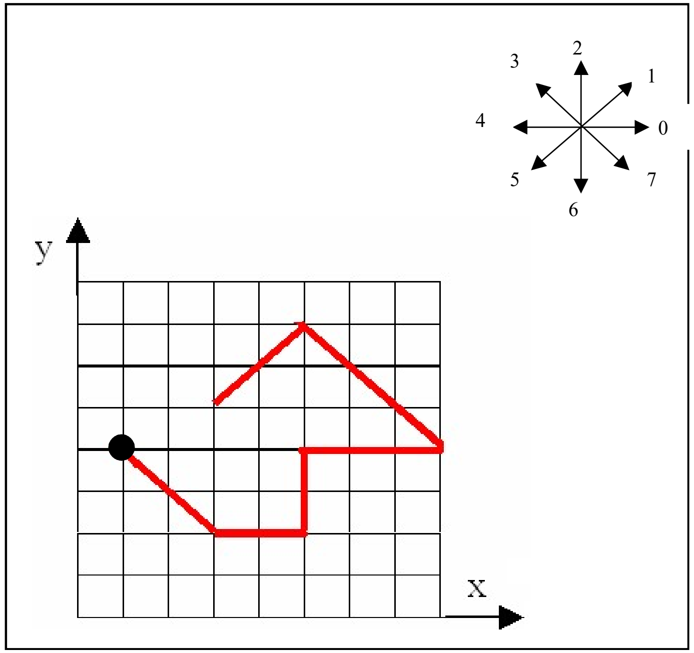


图 8.1.1　基于边界表达技术的分类

7.1.2 链码（Chain Code)

方向链：**8向链码、4向链码**



标识码

起始符: 0405X

X是起始点的坐标，为3位八进制数

结束符： 0400

0405001040500477002200033550400

与起点有关，受平移、旋转、尺度影响。受噪声影响。

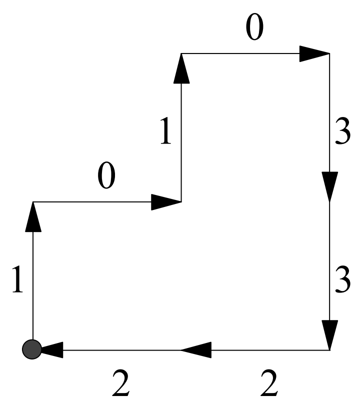
7.1.2 链码

链码归一化

① 起点归一化

将链码看作由方向数构成的自然数

选取值最小的自然数顺序



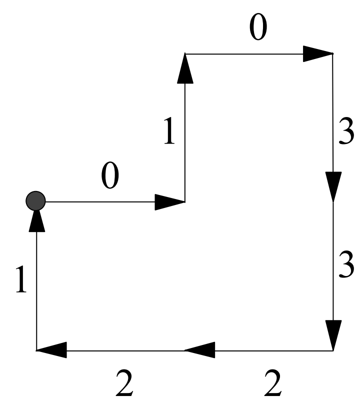
原链码

1 0 1 0 3 3 2 2

归一化链码

0 1 0 3 3 2 2 1

起点归一化



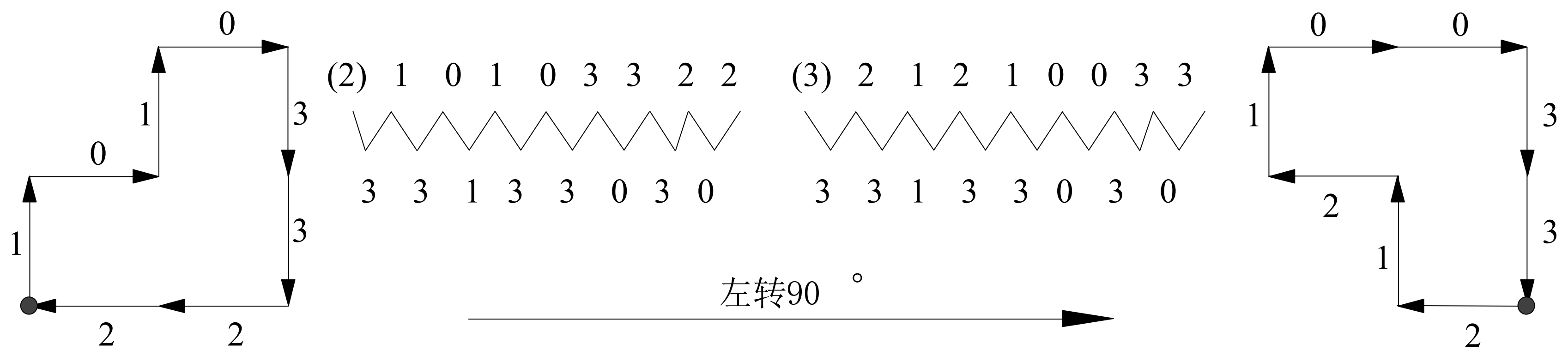
7.1.2 链码

链码归一化

② 旋转归一化

利用链码的一阶差分

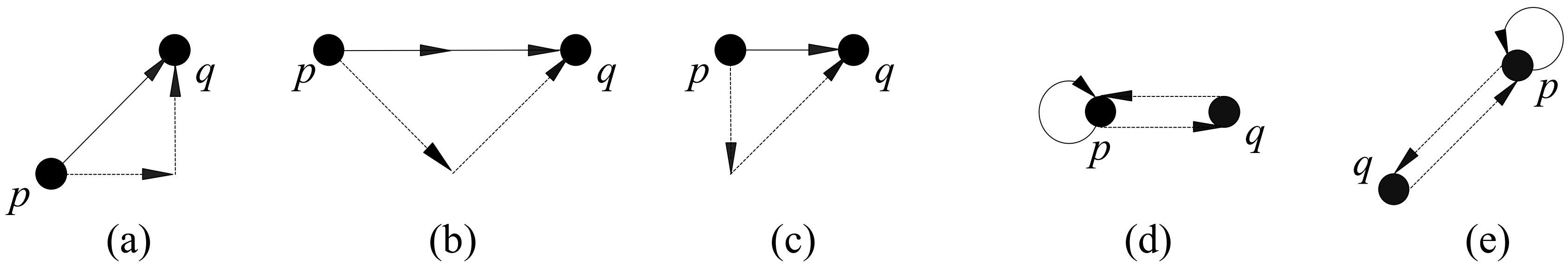
差分码不随轮廓旋转而变化



7.1.2 链码

链码平滑

将原始的链码序列用较简单的序列代替



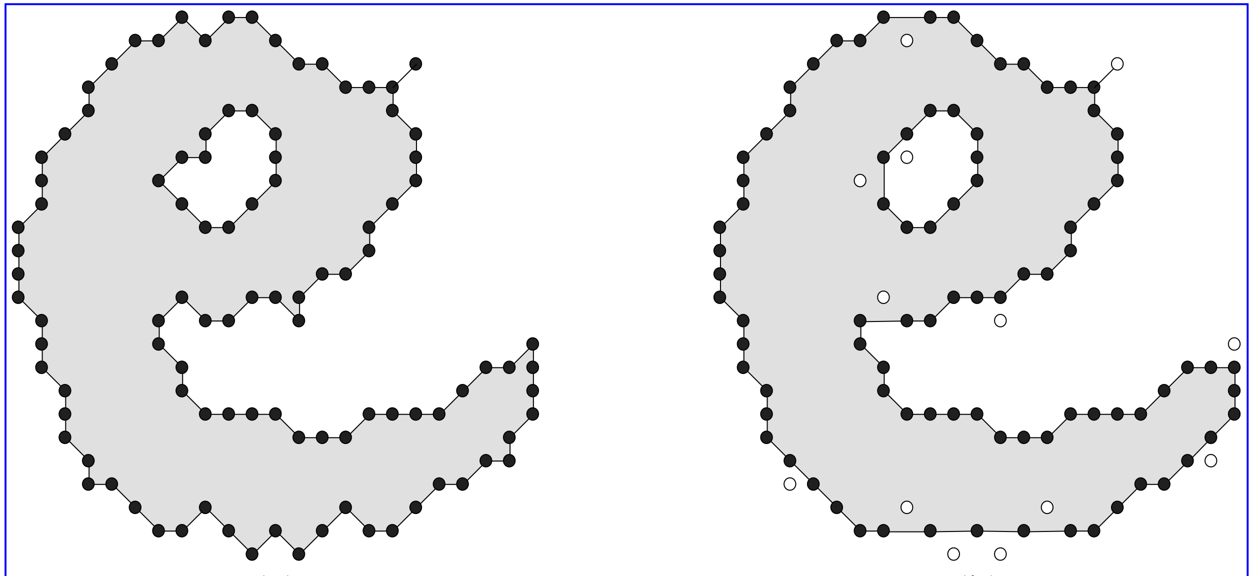
虚线箭头：原始的在象素p和q之间的8-连通链码

实线箭头：用来替换原始序列的新序列

7.1.2 链码

链码平滑示例

空心圆：平滑后被除去的原轮廓点



(a) (b)

7.1.3 边界段和凸包

- 把边界分解成若干段分别表示
- 借助凸包（包含目标的最小凸形）概念
- 节省表达数据量
- 便于符号表达
- 当感兴趣的形状信息存在于边缘凹陷处时，尤其适用

7.1.3 边界段和凸包

- 根据凸包把边界分解
- 目标：象素集合S
- 分解 凸包(Convex Hull)：包含S的最小凸形H

  凸残差：D = H – S
- 在进行凸包分解时，可以先对边界进行平滑

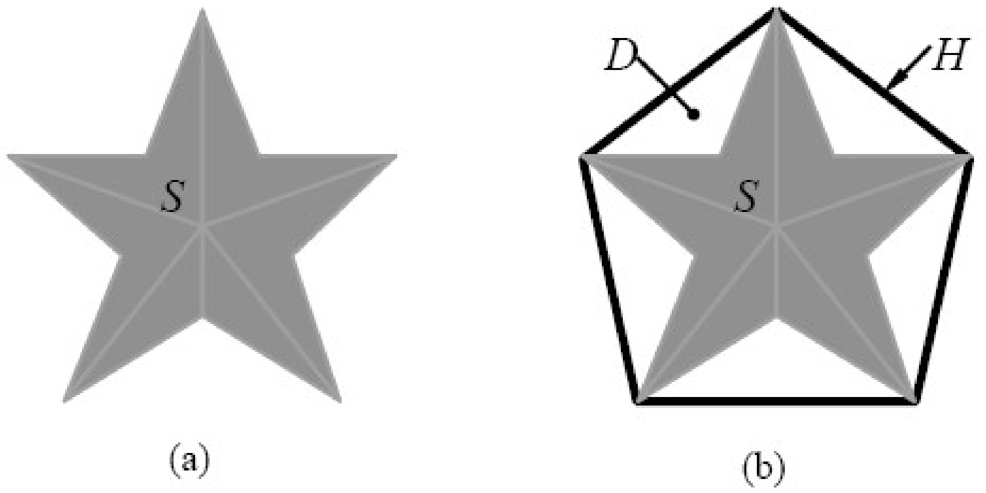


图 8.1.7 区域的凸包

7.1.3 边界段和凸包

- 利用区域凸包分解边界段

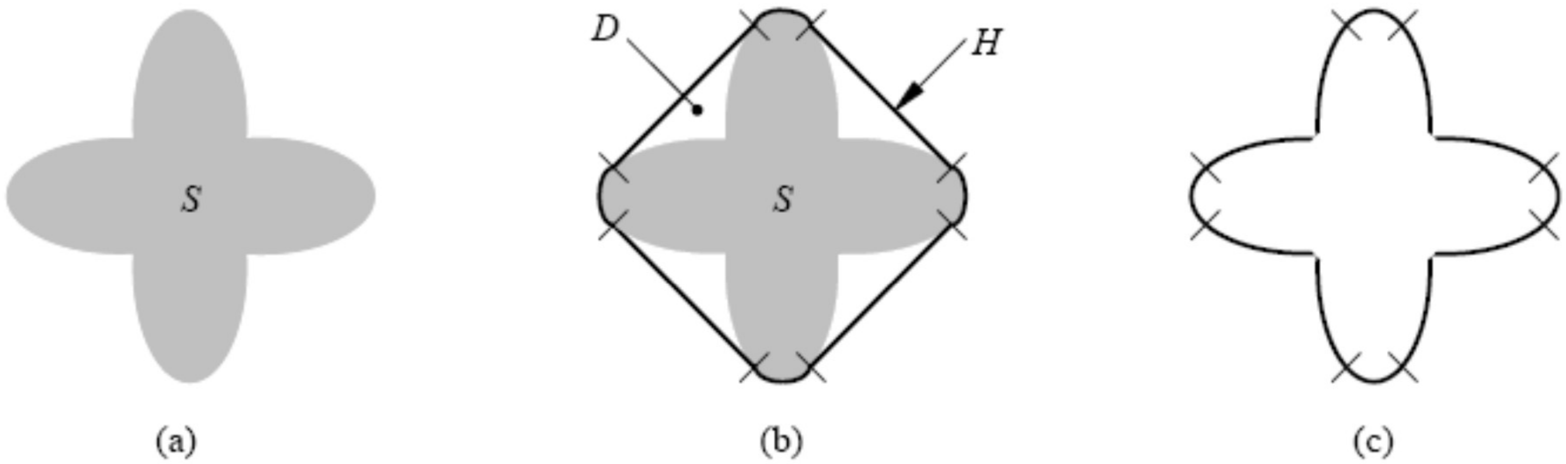
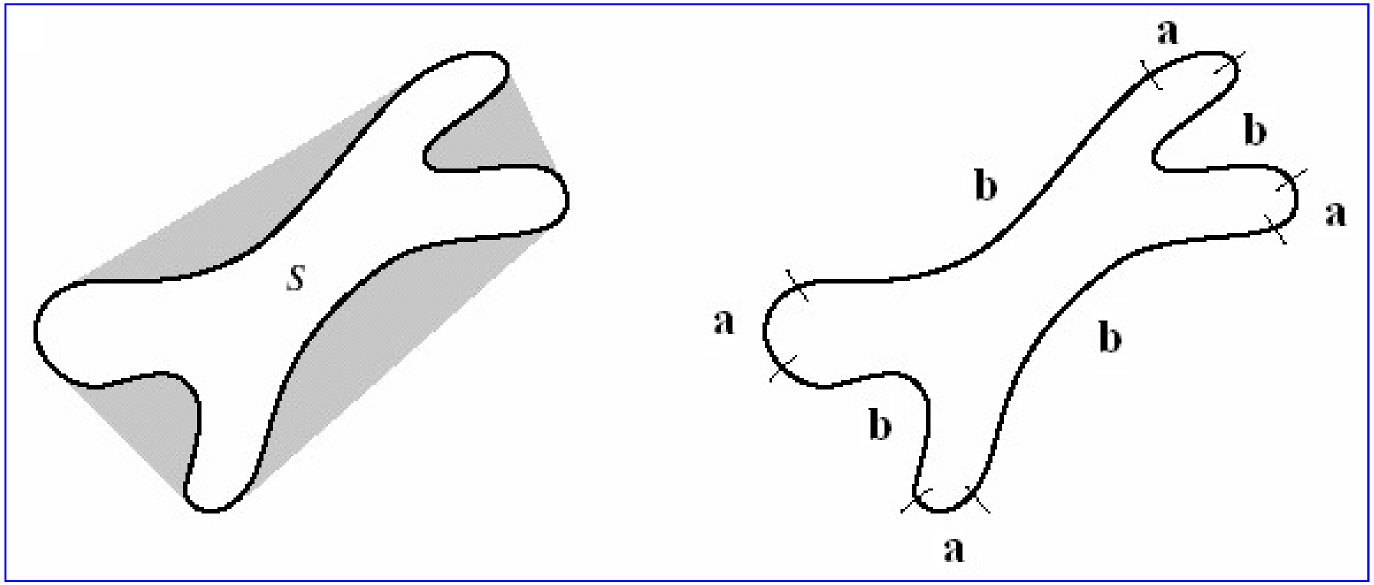


图 8.1.8　利用区域凸包将区域边界分段

7.1.3 边界段和凸包

- 凸包同样适用于区域的表达



若以a,b分别表示凸和凹部，则该染色体可以表示为abababab

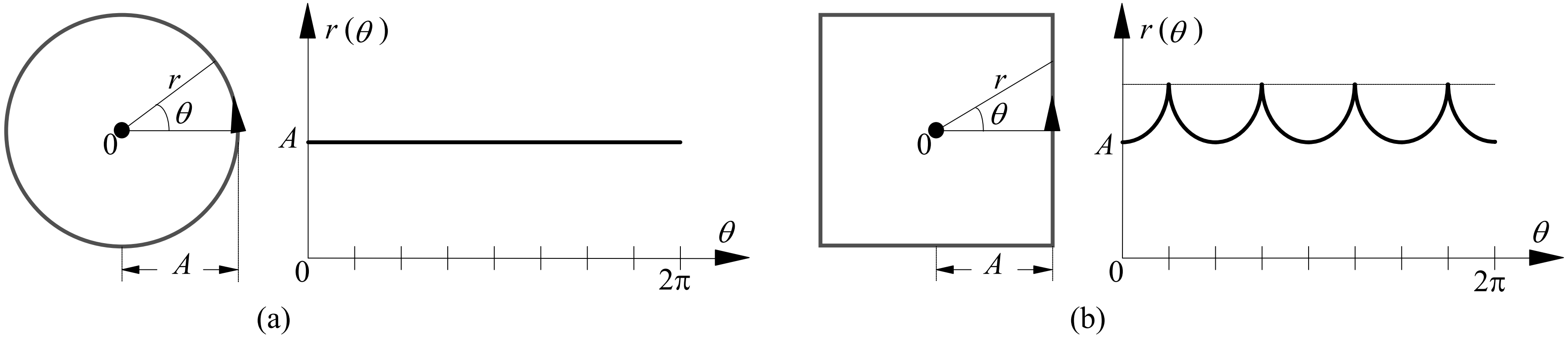
7.1.4 边界标记

- 边界标记(signature) 把2-D边界用1-D的较容易描述的函数形式来表达
- 标记可由广义的投影产生
- 投影可以是水平的、垂直的、对角线的、或放射的、旋转的
- 投影并不是一种能保持信息的变换，将2-D平面上的区域边界变换为1-D的曲线有可能丢失信息

7.1.4 边界标记

1. 距离为角度的函数

先对给定的物体求出重心，然后把边界点与重心的距离作为角度的函数

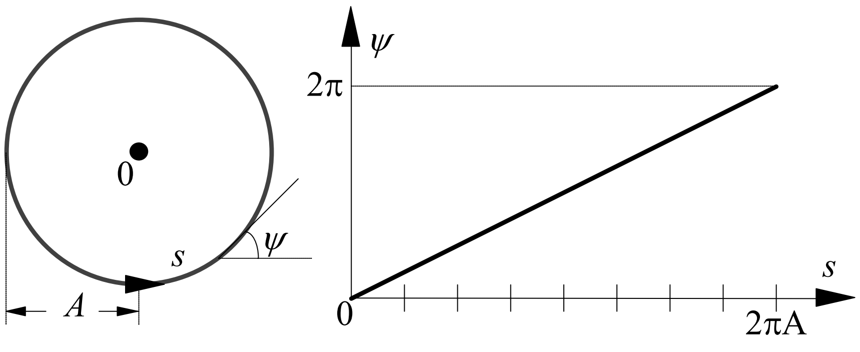


不受目标平移影响，但会随目标旋转或放缩而变化

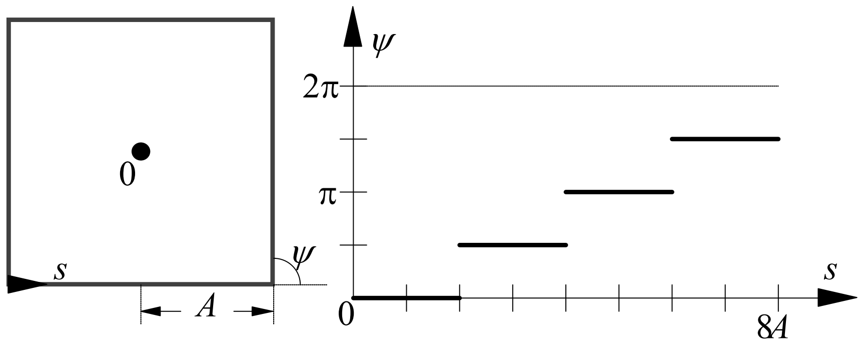
7.1.4 边界标记

2. ψ - s曲线（切线角为弧长的函数）

沿边界围绕目标一周，在每个位置作出该点切线与一个参考方向（如横轴）的角度值



(a)



(b)

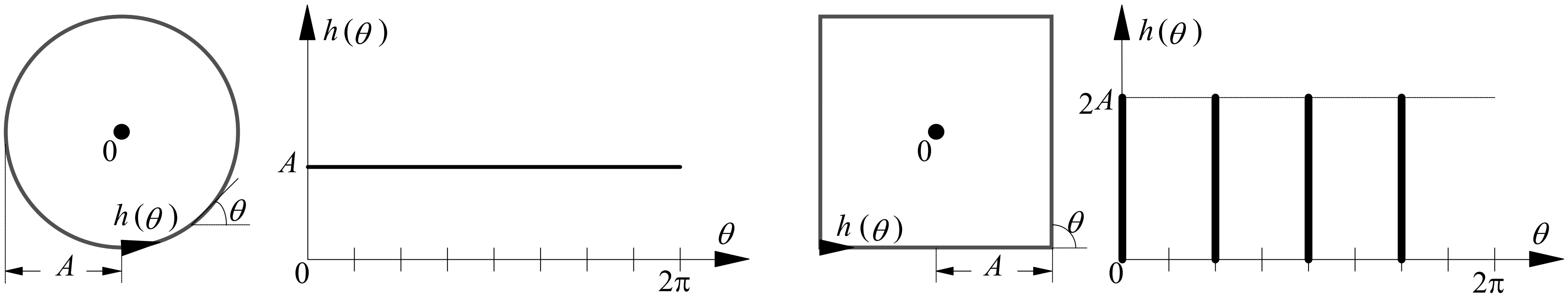
水平直线段对应边界上的直线段（ψ不变）

7.1.4 边界标记

3. 斜率密度函数

将ψ-s曲线沿 ψ轴投影

切线角的直方图$h(\theta)$



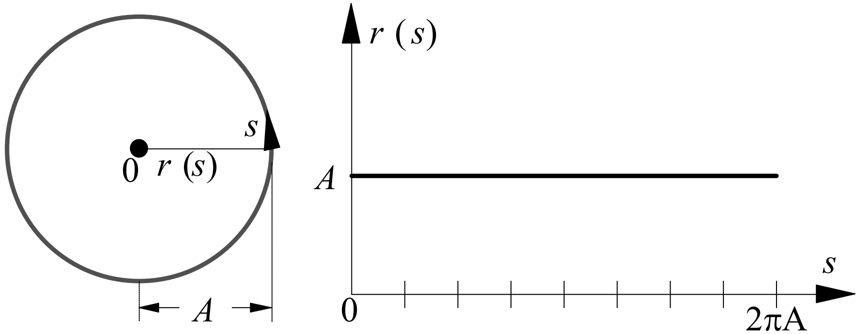
(a) (b)

切线角有较快变化的边界段对应较深的谷

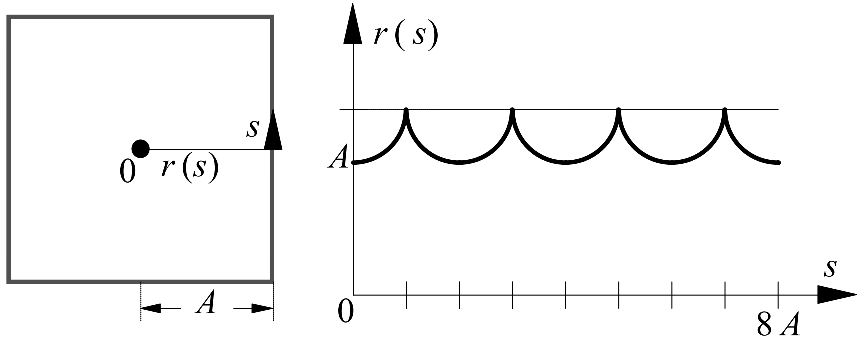
7.1.4 边界标记

4. 距离为弧长的函数

将各个边界点与目标重心的距离作为边界点序列（围绕目标得到）的函数



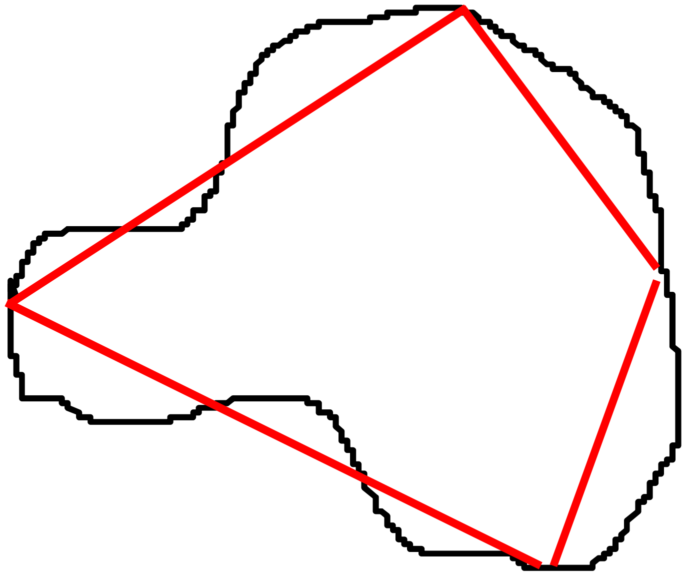
(a)



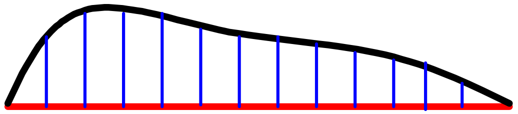
(b)

与距离为角度的函数相比？

7.1.5 多边形近似

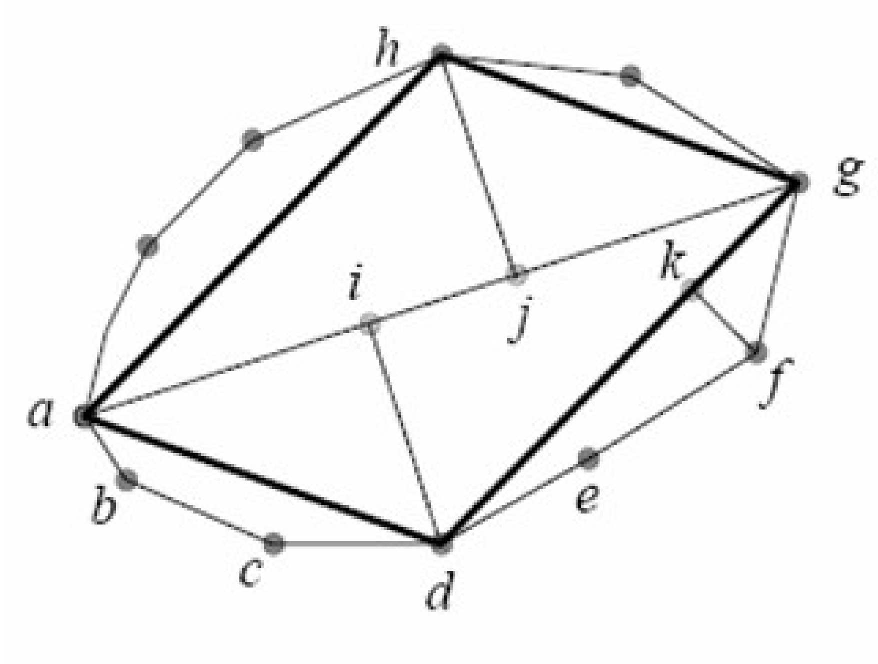


误 差 计 算



7.1.5 多边形近似

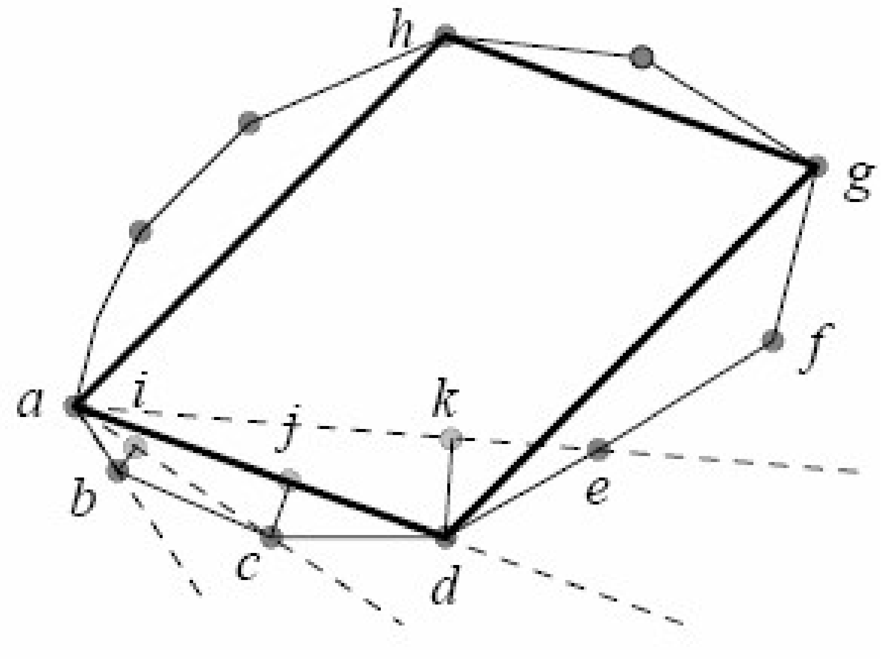
- 分裂算法
  - 先连接边界上相距最远的两个点（即把边界分成两部分），然后根据一定的准则进一步分解边界，构成多边形逼近边界，直到拟合误差满足一定的条件。



7.1.5 多边形近似

□ 聚合算法

- 先选一个边界点为起点，用直线依次连接该点与相邻的边界点，直至拟合误差超过某个限度。
- 然后以线段的另一段为起点继续连接边界点，直至绕边界一周。



与起点有关
贪婪算法

7.1.5 多边形近似

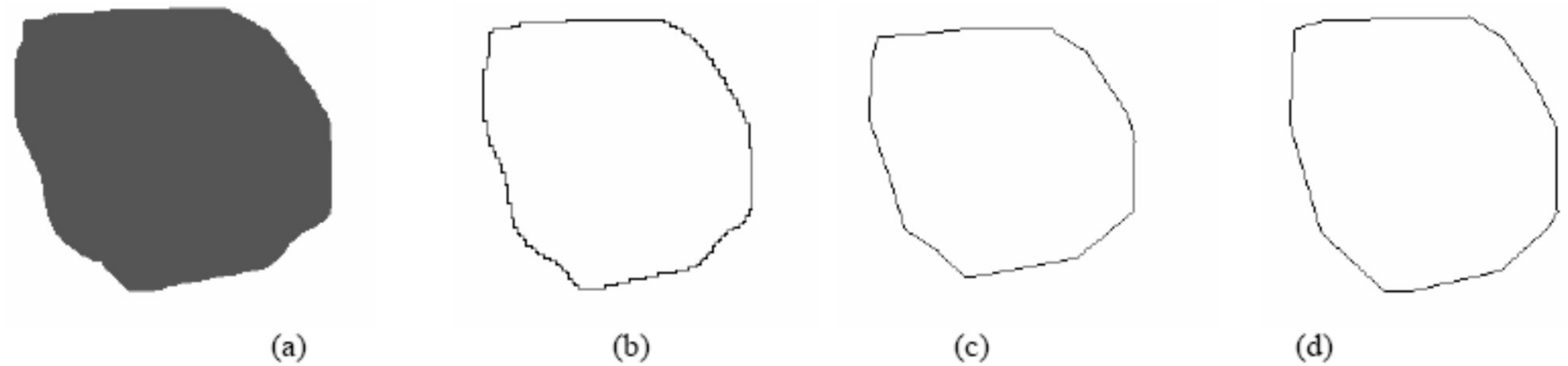


图 8.1.16 多边形边界表达示例

(a)分割后图象；

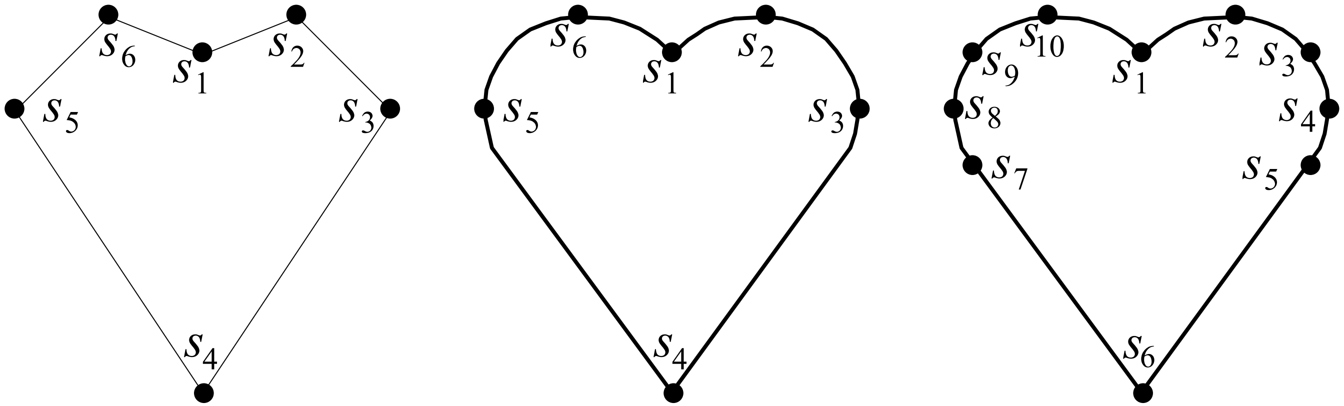
(b)链码-112bit；

(c)聚合逼近多边形-272bit；

(d)分裂逼近多边形-224bit

7.1.6 地标点

- 标志点或地标点（Landmark Points）
  - 具有某种几何特性的点，如极值点、大曲率点。一种近似表达方法。



(a) (b) (c)

准确表达　　　　　　　　近似表达

7.1.6 地标点

具有顶点$S_1 = (1, 1)$，$S_2 = (1, 2)$，$S_3 = (2, 1)$的三角形

方式	表达	解释
$2n$-矢量	$\boldsymbol{S}_\mathrm{o} = [1, 1, 1, 2, 2, 1]$	$\boldsymbol{S}_\mathrm{o}$是一个$2n\times 1$的实坐标矢量
$2n$-集合	$\boldsymbol{S}_\mathrm{f} = \{1, 1, 1, 2, 2, 1\}$	$\boldsymbol{S}_\mathrm{f}$是一个包含$2n$个实坐标的集合
矢量-平面	$\boldsymbol{S}_\mathrm{v} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ 2 & 1 \end{bmatrix}$	$\boldsymbol{S}_\mathrm{v}$是一个$n\times 2$的矩阵，每行包含一个标志点的$x$-和$y$-实坐标
复数-平面	$\boldsymbol{S}_\mathrm{c} = \begin{bmatrix} 1+j \\ 1+2j \\ 2+j \end{bmatrix}$	$\boldsymbol{S}_\mathrm{c}$是一个$n\times 1$的复数矢量，每个复数表示一个标志点的$x$-和$y$-坐标

7.2 基于区域的表达

基于区域的象素点进行

7.2.1 技术分类

7.2.2 空间占有数组

7.2.3 四叉树

7.2.4 金字塔

7.2.5 围绕区域

7.2.6 骨架

7.2.1 技术分类

(1)　区域分解：简单的单元形式

(2)　围绕区域：外接圆，外包围矩形

(3)　内部特征：内部象素集合

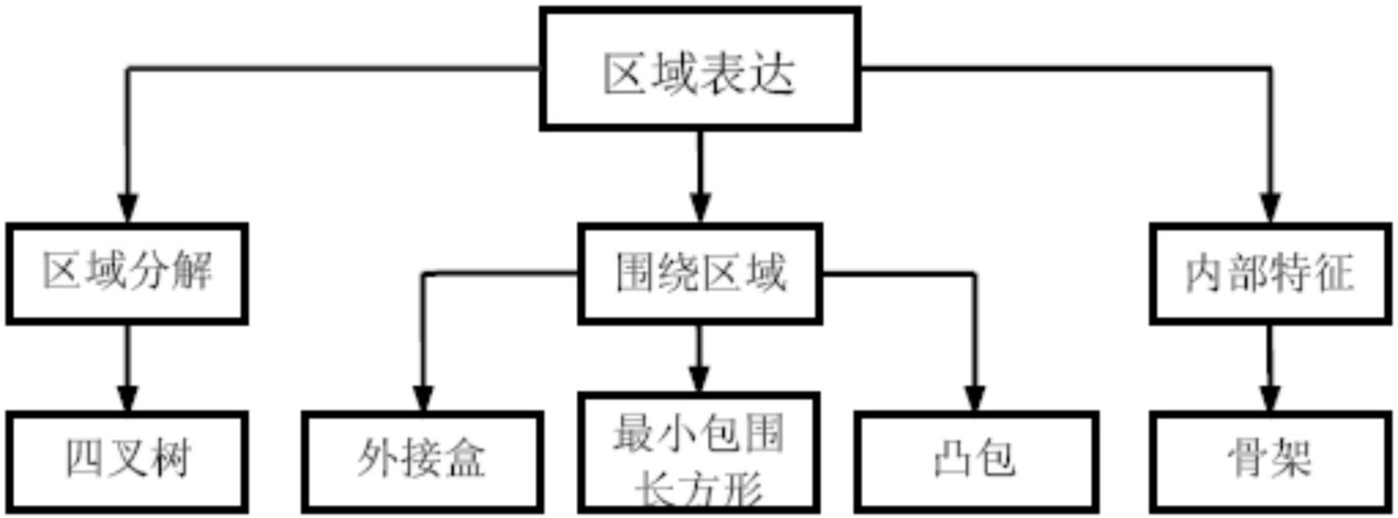


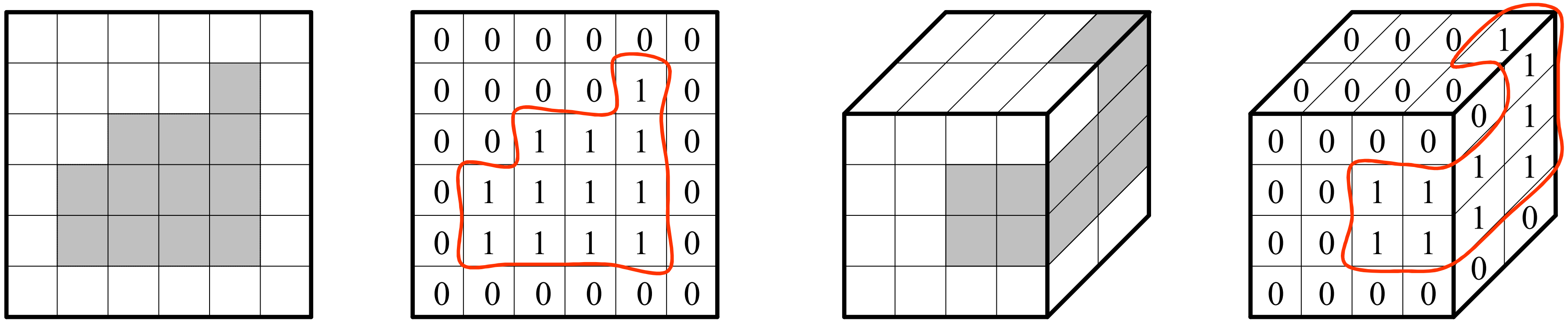
图 8.2.1　基于区域表达技术的分类

7.2.2 空间占有数组

对图象 $f(x, y)$ 中任一点 (x, y)：

如果它在给定的区域内，就取 $f(x, y)$ 为1

否则就取 $f(x, y)$ 为0



(a) (b)

所有 $f(x, y)$ 为1的点组成的集合就代表了所要表示的区域

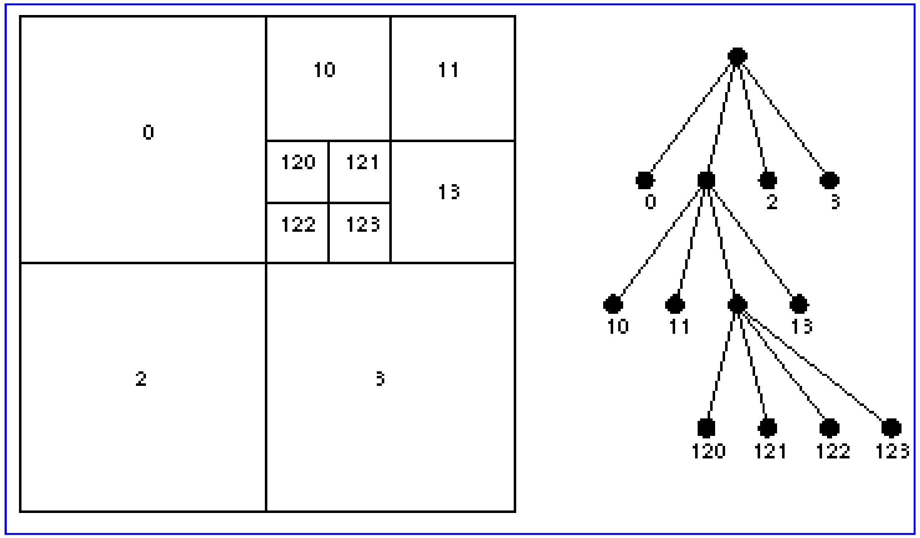
7.2.3 四叉树

基本思路：分层分解图象

利用金字塔式的数据结构

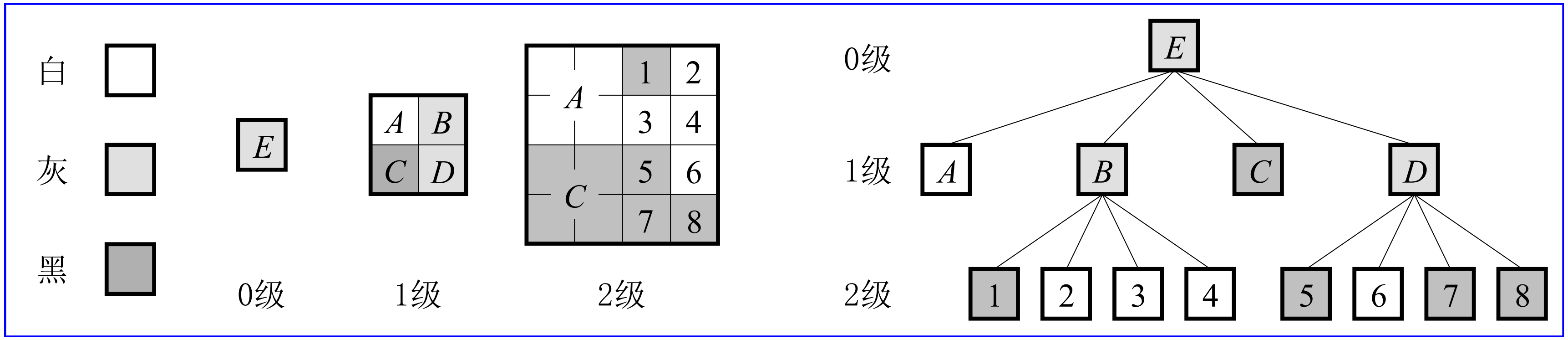
四叉树表达法：每次将图象一分为四，编码方式与金字塔同。

树结构 $T = \{节点集, 弧集\}$。



7.2.3 四叉树

四叉树表达图示



节点分为3类：目标节点、背景节点、混合节点

表达优点：常用于“粗略信息优先”显示

结点数目上限

$$N=\sum_{k=0}^{n}4^{k}=\frac{4^{n+1}-1}{3}\approx\frac{4}{3}4^{n}$$

编码方式

(1) 位置码

对于$2^N \times 2^N$的图用N位码编码

同一父节点的四块顺时针编号为1，2，3，4

1	2
4	3

(2) 灰度值

灰度值只需记平均值g_o和差值g_i　$i=1, 2, 3$

$$g_0 = \frac{1}{4}(f_0 + f_1 + f_2 + f_3)$$

$$g_i = f_i - g_0$$

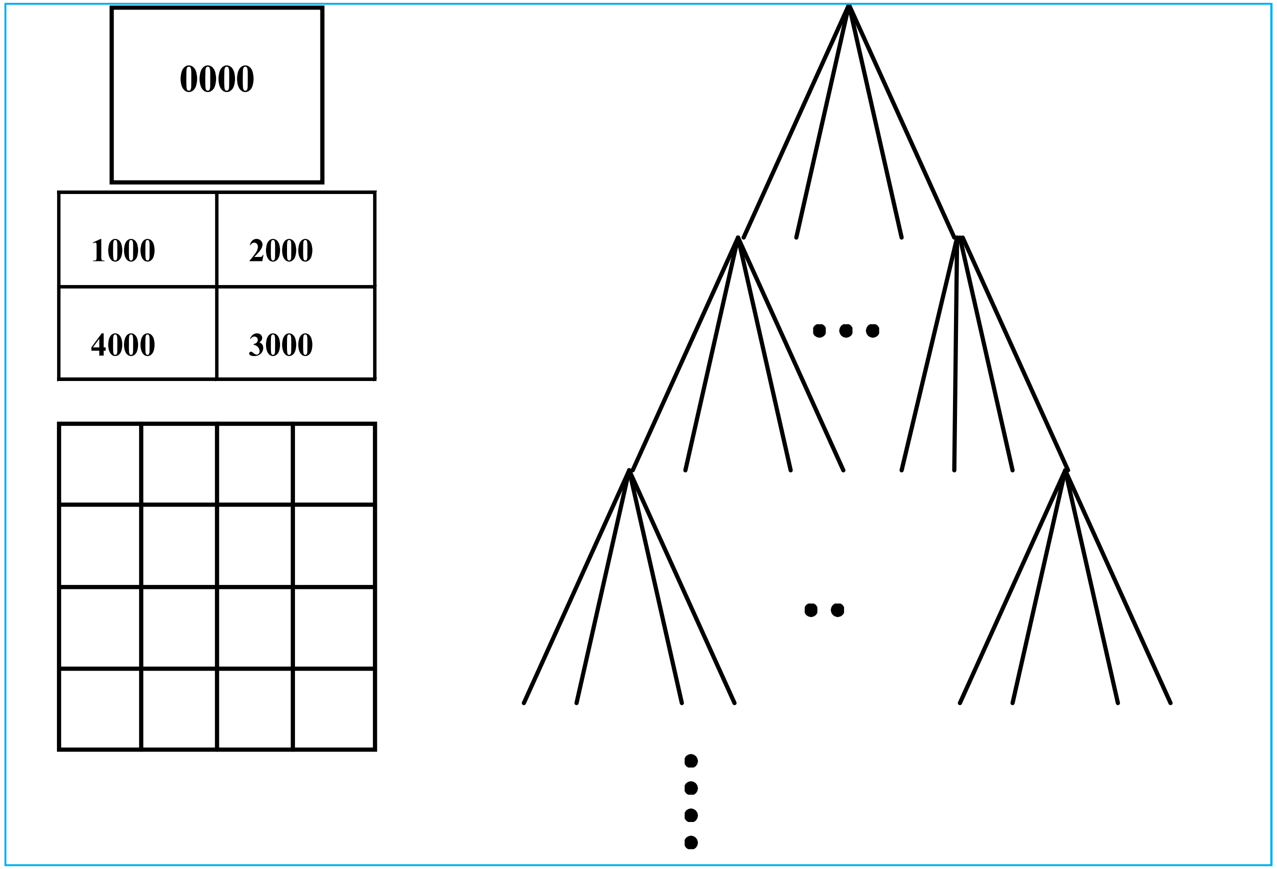
$$f_0 = g_0 - \sum_{i=1}^{3} g_i$$

$$f_i = g_0 + g_i \qquad i = 1,2,3$$

f_0	f_1
f_3	f_2

例：N=4　　　16×16

1	2
4	3



1100，1200，2100，2200

1400，1300，2400，2300

4100，4200，3100，3200

4400，4300，3400，3300

问：2140？　3421？　3331？

数据块左上角的坐标

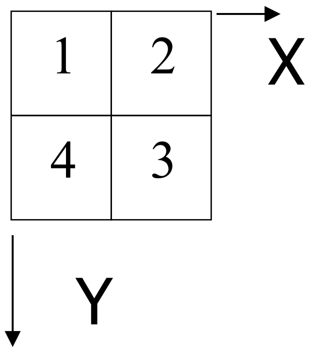
坐标原点在图的左上角，且第一个象素坐标取（1，1）

对非零码， 码值为1、4时，X坐标值取0

码值为2、3时，X坐标值取2^d

码值为1、2时，Y坐标值取0

码值为3、4时，Y坐标值取2^d



d 为从右到左数时码的位数，

例：码 2 3 1 0 $x = 2^3 + 2^2 + 0 + 1 = 13$

位数d 3 2 1 0 $y = 0 + 2^2 + 0 + 1 = 5$

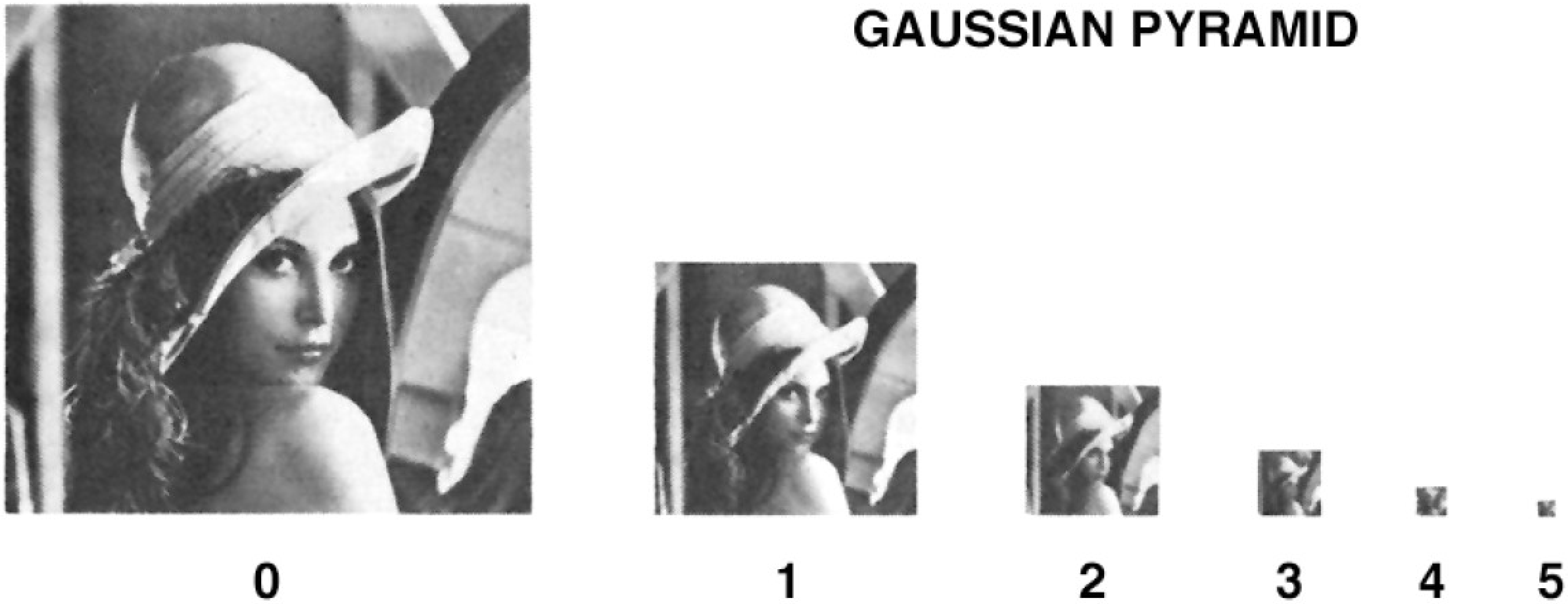
由编码的值可知：

数据块的大小、数据块位置、相邻的情况

- 数据块的大小
  - 大小是 $2^k \times 2^k$，　k为0码的个数
- 数据块位置
  - 数据块左上角坐标可由码求出
- 相邻的情况
  - 同一父节点的四块相邻，即右起第一个同一位上的非零码依次为1，2，3，4，其余的码相同的四块相邻
  - 其他相邻情况由块的位置和块的大小导出

7.2.4 金字塔

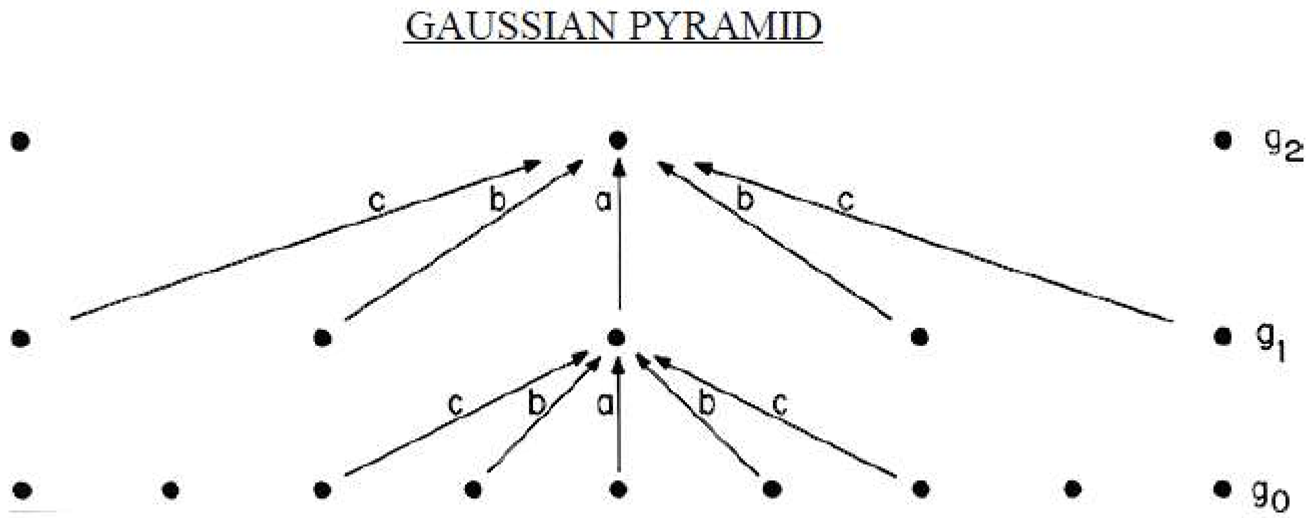
- 金字塔表示（多分辨）



P. Burt, E. H. Adelson, “The Laplacian Pyramid as a Compact Image Code” , IEEE Trans. Comm. 1983.

7.2.4 金字塔

- Gaussian金字塔表示



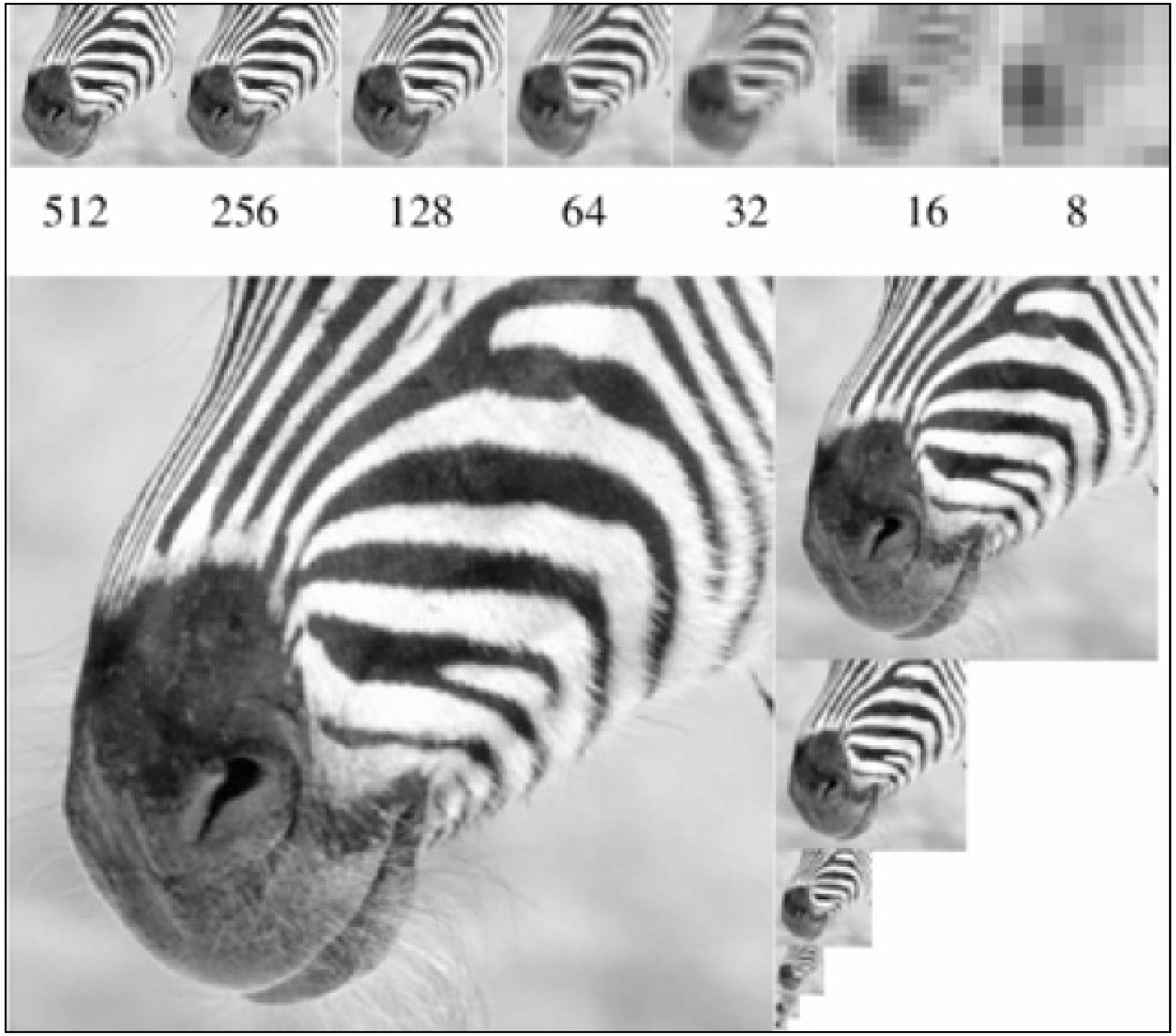
g_0 = IMAGE

g_L = REDUCE $[g_{L-1}]$

P. Burt, E. H. Adelson, "The Laplacian Pyramid as a Compact Image Code" , IEEE Trans. Comm. 1983.

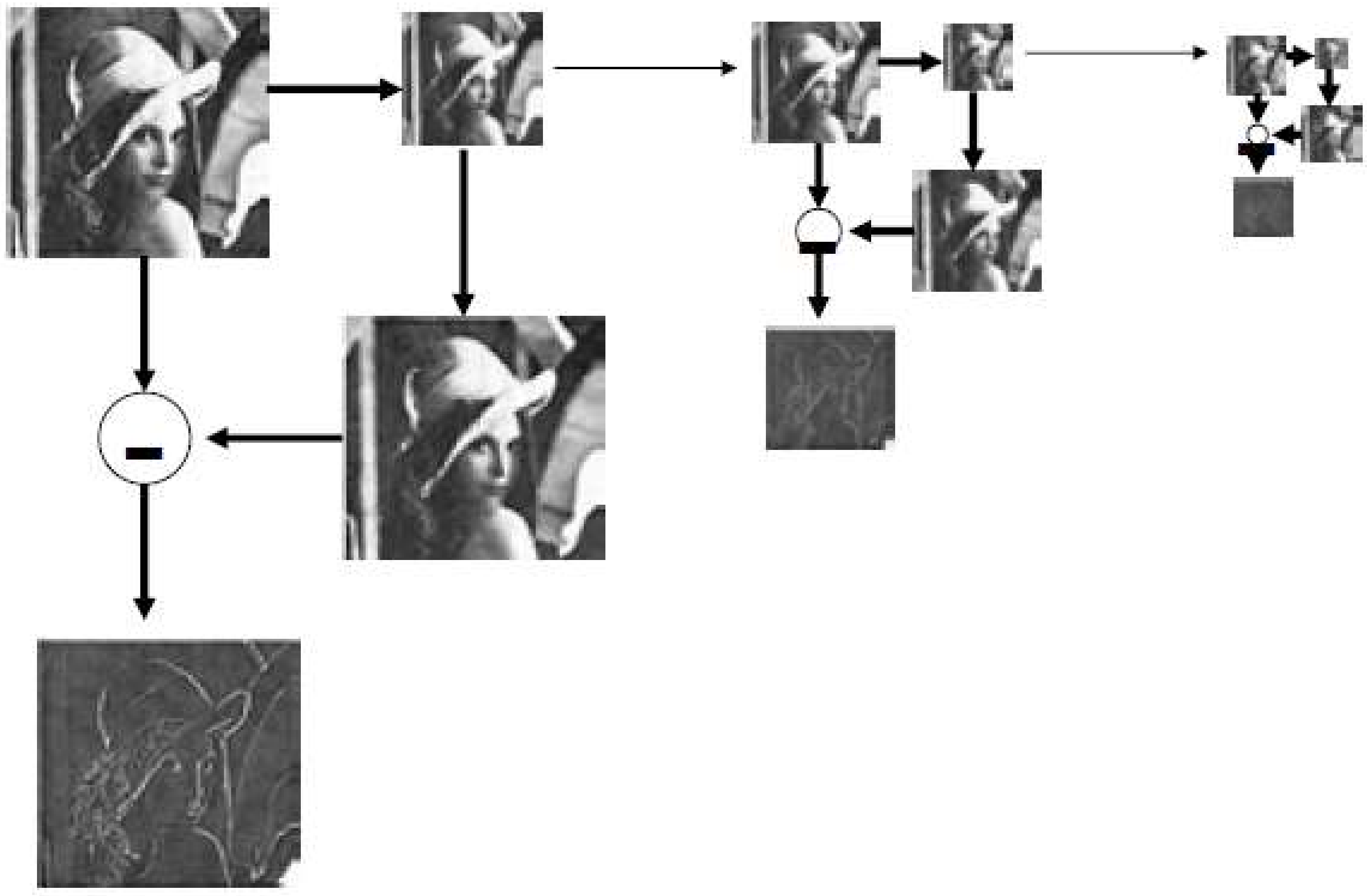
7.2.4 金字塔

- Gaussian金字塔表示



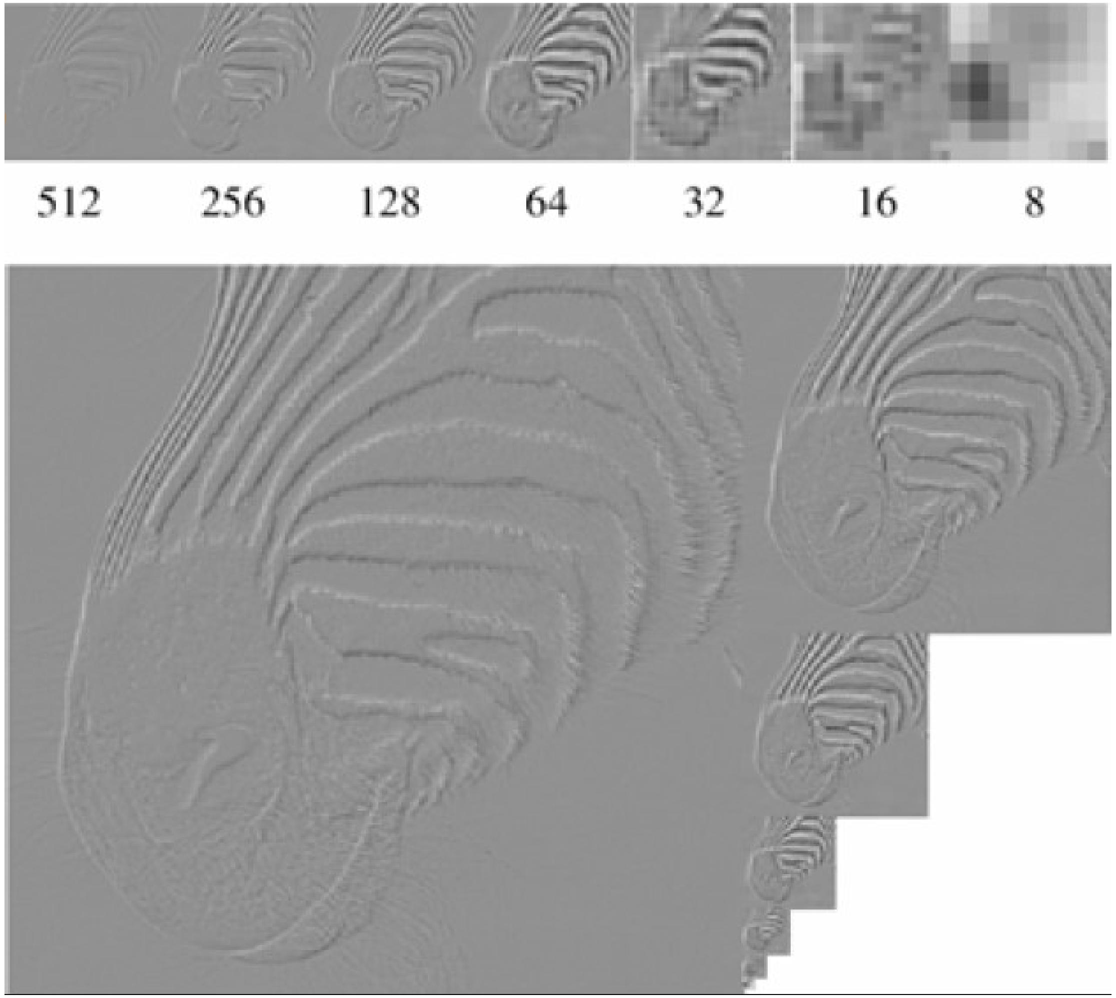
7.2.4 金字塔

- Laplacian金字塔表示



7.2.4 金字塔

- Laplacian金字塔表示



7.2.5 围绕区域

(1) 外接盒（Feret box）：包含目标区域的最小的长方形（朝向特定的参考方向）

(2) 围盒（minimum enclosing rectangle，MER）：包含目标区域的（可朝向任何方向）最小长方形

(3) 凸包：见7.1.3小节

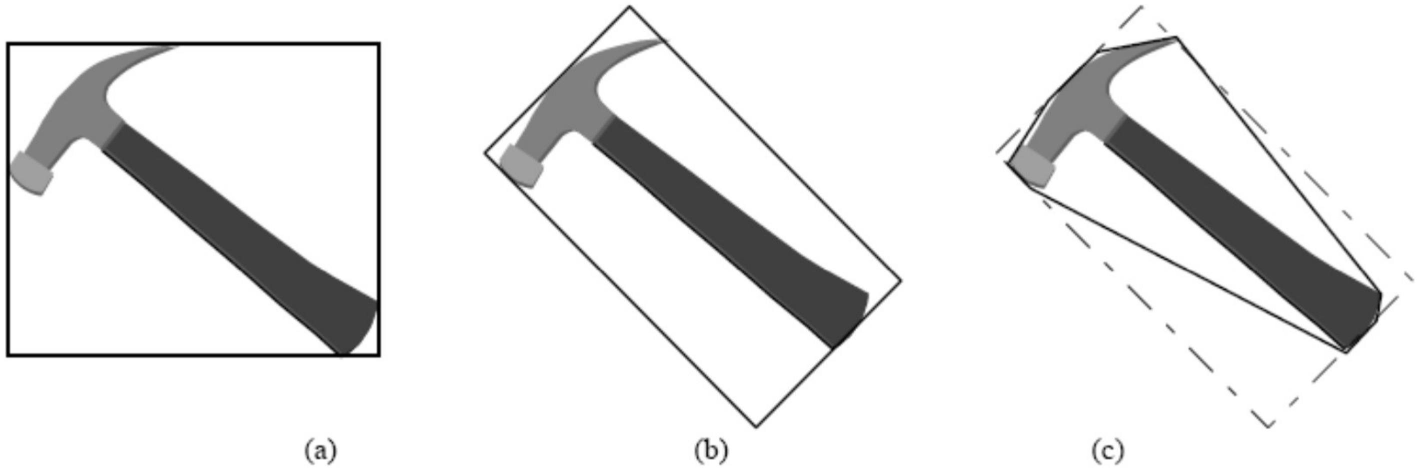


图 8.2.5　对同一个区域的三种围绕区域表达技术

7.2.6 骨架

1. 骨架的定义和特点

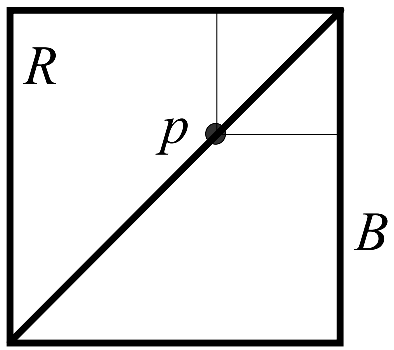
- 骨架点

  与（两个）轮廓点距离最小的点

$$d_s(p,B) = \inf\{d(p,z) \mid z \subset B\}$$

骨架点的确定

- 区域 R
- 轮廓 B
- 骨架点 p



7.2.6 骨架

1. 骨架的定义和特点

- 较细长的物体其骨架提供较多信息；较粗短的物体其骨架提供的信息较少
- 骨架受噪声的影响较大

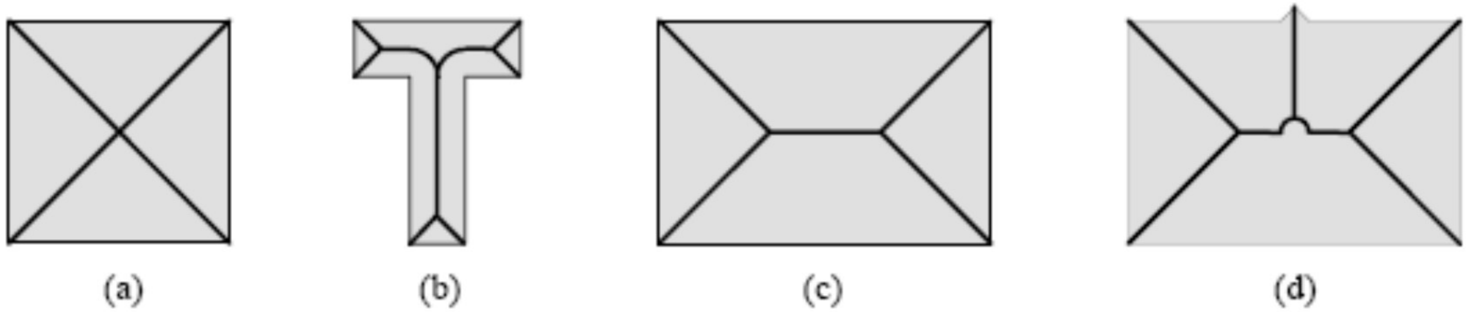
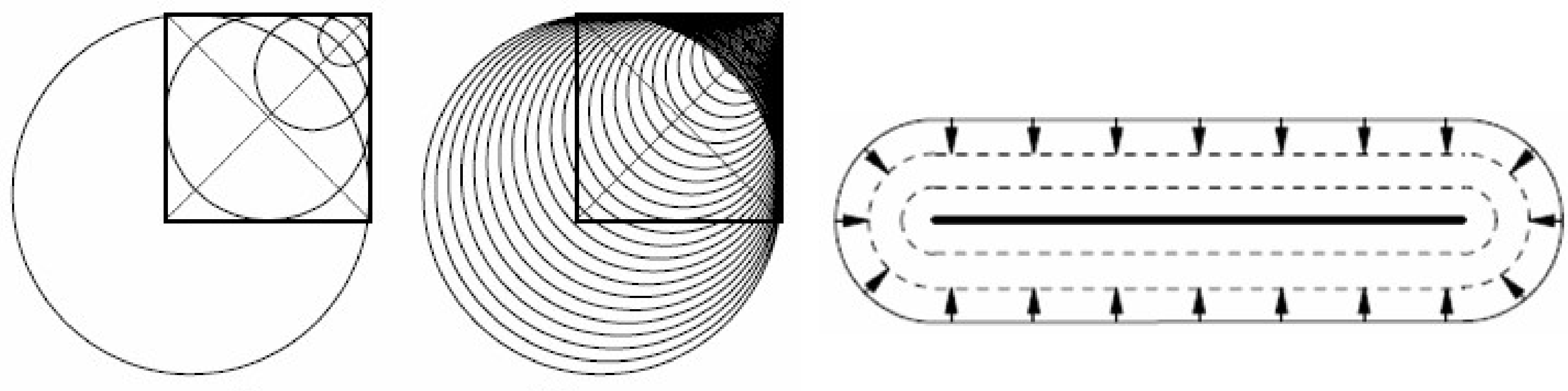


图 8.2.8 用欧氏距离算出的一些骨架的示例

7.2.6 骨架

1. 骨架的定义和特点

- 恢复原始区域(假设骨架点到边界距离已知)：沿骨架作相切圆，取包络。
- 波传播的解释：grass fire



(a) (b)

7.2.6 骨架

- 骨架的性质（实际中有时并不能完全满足）
  - S完全包含在R中
  - S为单像素宽
  - S与R有相同的连通元数
  - S的补与R的补有相同的连通元数
  - 可以根据S重建R（假设骨架点到边界距离已知）

7.2.6 骨架

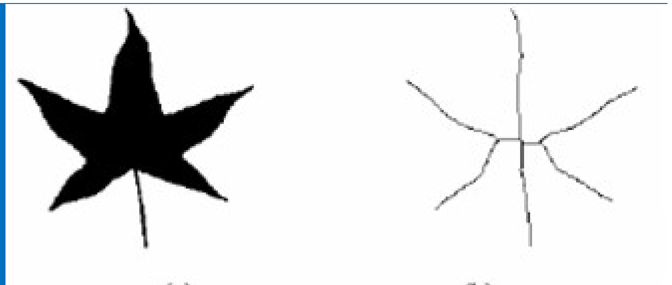
- 直接利用定义计算骨架点，代价太大。
- 实际中可采用逐次消除边界点的迭代细化算法。
- 这个过程中，有3个限制条件需要满足:

  (1)不消去线段端点

  (2)不中断原来连通的点

  (3)不过多侵蚀区域

7.2.6 骨架


(a) (b)

Step1：

(1) 标记同时满足下列条件的边界点

$\text{i)}\ 2 \le N(p_1) \le 6;\quad \text{ii)}\ S(p_1) = 1;\quad \text{iii)}\ p_2 \cdot p_4 \cdot p_6 = 0;\quad \text{iv)}\ p_4 \cdot p_6 \cdot p_8 = 0;$

其中$N(p_1)$是p_1邻域的非零点数；

$S(p_1)$是从p_2开始顺时针转一圈后,$0 \to 1$变化的的次数。

(2) 当所有边界点都检验完毕后，将标记点去掉。

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

Step2：

(1) 标记同时满足下列条件的边界点

$\text{i)}\ 2 \le N(p_1) \le 6;\quad \text{ii)}\ S(p_1) = 1;\quad \text{iii)}\ p_2 \cdot p_4 \cdot p_8 = 0;\quad \text{iv)}\ p_2 \cdot p_6 \cdot p_8 = 0;$

(2) 当所有边界点都检验完毕后，将标记点去掉。

Step3: 重复Setp1,2直到没有点满足标记条件

7.2.6 骨架

算法解释

$$\text{i)}\ 2 \le N(p_1) \le 6;\quad \text{ii)}\ S(p_1)=1;\quad \text{iii)}\ p_2 \cdot p_4 \cdot p_6 = 0;\quad \text{iv)}\ p_4 \cdot p_6 \cdot p_8 = 0;$$

$$\text{i)}\ 2 \le N(p_1) \le 6;\quad \text{ii)}\ S(p_1)=1;\quad \text{iii)}\ p_2 \cdot p_4 \cdot p_8 = 0;\quad \text{iv)}\ p_2 \cdot p_6 \cdot p_8 = 0;$$

- 条件 i) 保证不消去线段端点和过多侵蚀区域。
- 条件 ii) 保证不中断原本连通的点。
- 条件 iii)和 iv)保证消去的点不是骨架点

p_9	p_2	p_3
p_8	p_1	p_4
p_7	p_6	p_5

0	1	0
0	p	0
0	0	0

1	1	1
1	p	1
1	0	1

0	0	0
1	p	1
0	0	0

0	1	1
0	p	0
1	1	0

1	1	0
1	p	0
0	0	0

0	0	0
0	p	1
0	1	1

7.3 基于变换的表达

7.3.1 技术分类

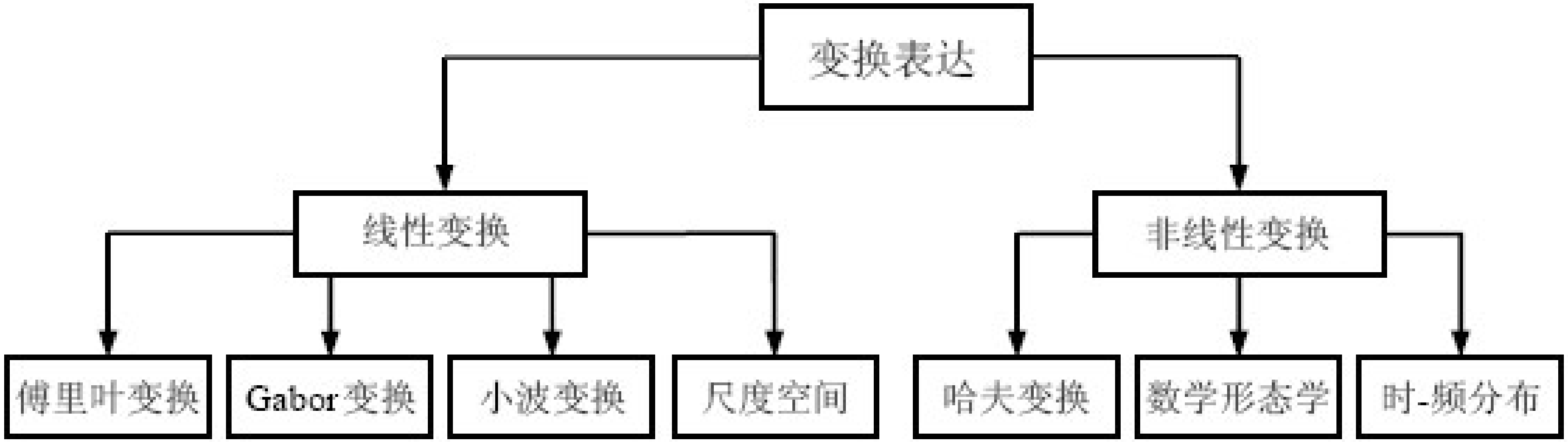
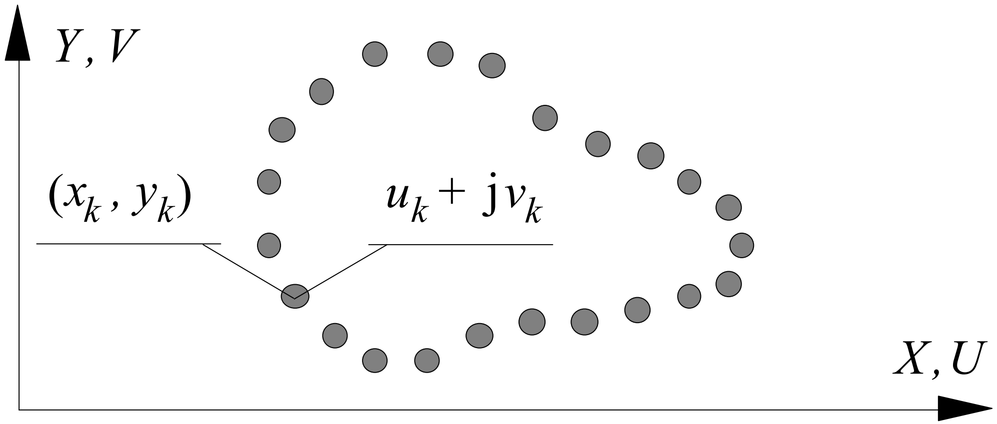


图 8.3.1　基于变换表达技术的分类

7.3.2 傅里叶变换表达

离散傅里叶变换表达

将XY平面中的曲线段转化为复平面UV上的点序列



将2-D的问题简化为1-D的问题

7.3.2 傅里叶变换表达

从1个封闭边界可得到1个复数序列

$$s(k) = u(k) + \mathrm{j}v(k) \qquad k = 0, 1, \cdots, N-1$$

将序列进行傅里叶变换

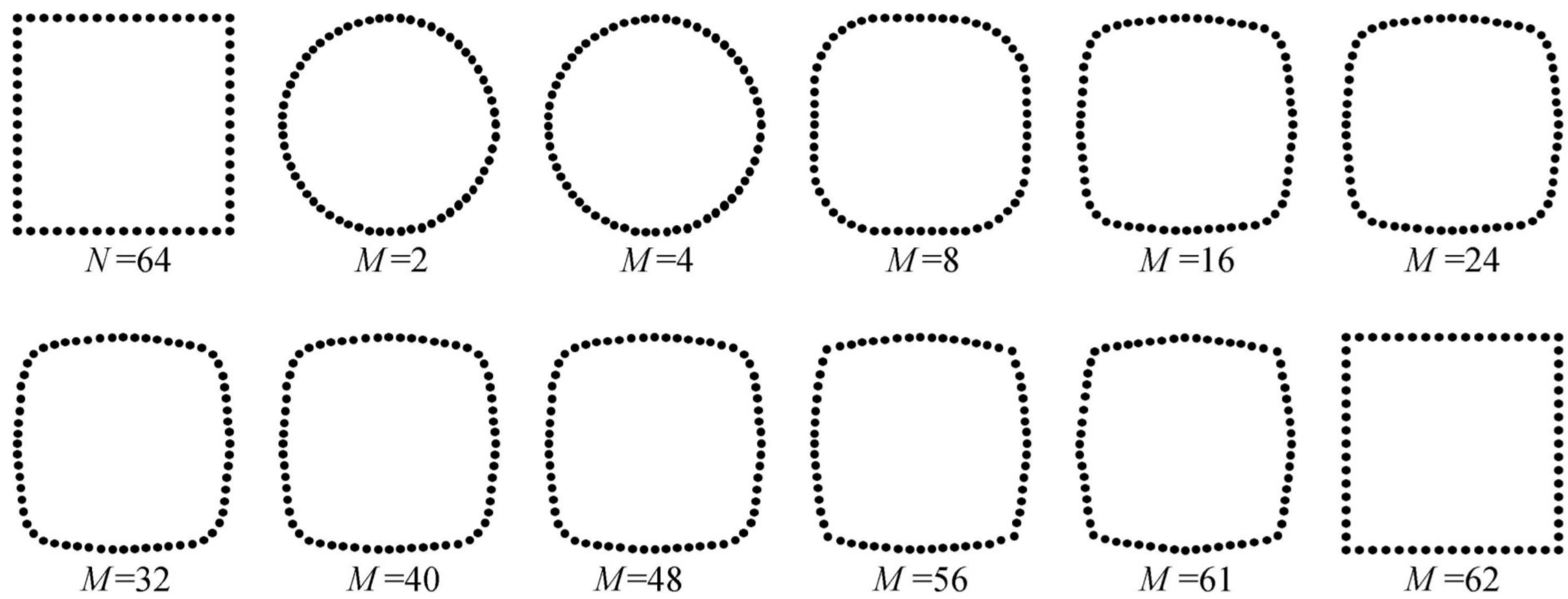
$$S(w) = \frac{1}{N}\sum_{k=0}^{N-1} s(k)\exp[-\mathrm{j}2\pi wk/N] \qquad w = 0, 1, \cdots, N-1$$

取傅里叶变换系数表达轮廓

$$\hat{s}(k) = \sum_{w=0}^{M-1} S(w)\exp[\mathrm{j}2\pi wk/N] \qquad k = 0, 1, \cdots, N-1$$

7.3.2 傅里叶变换表达

利用边界傅里叶变换的前M个系数可用较少的数据量表达边界的基本形状



7.3.2 傅里叶变换表达

傅里叶变换表达受边界平移、旋转、尺度变换以及计算起点（傅里叶描述与从边界点建立复数序列对的起始点有关）的影响

变换/变化	边界点序列	傅里叶变换系数序列
平移$(\Delta x, \Delta y)$	$s_t(k) = s(k)+\Delta xy$	$S_t(w) = S(w)+\Delta xy \bullet \delta(w)$
旋转(θ)	$s_r(k) = s(k)\exp(j\theta)$	$S_r(w) = S(w)\exp(j\theta)$
尺度(C)	$s_c(k) = C \bullet s(k)$	$S_c(w) = C \bullet S(w)$
起点(k_0)	$s_p(k) = s(k-k_0)$	$S_p(w) = S(w)\exp(-j2\pi k_0 w/N)$