

3.3 直方图修正

■ 直方图定义 (Histogram):

为了有利于数字图像处理，必须引入离散形式。在离散形式下，用 r_k 代表离散灰度级，用 $P_r(r_k)$ 代表 $P_r(r)$ ，则有：

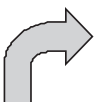
$$P_r(r_k) = \frac{n_k}{n} \quad 0 \leq r_k \leq 1$$
$$k = 0, 1, 2, \dots, l-1$$

式中 n_k 为图像中出现 r_k 这种灰度的像素数， n 是图像中像素总数，而 $\frac{n_k}{n}$ 就是概率论中所说的**频数**。

■ 在直角坐标系中作出 r_k 与 $P_r(r_k)$ 的关系图形，这个图形称为直方图。

3.3 直方图修正

1	2	3	4	5	6
6	4	3	2	2	1
1	6	6	4	6	6
3	4	5	6	6	6
1	4	6	6	2	3
1	3	6	4	6	6



1	2	3	4	5	6
5	4	5	6	2	14

灰度值
像素数

画出直方图，确当最佳阈值，阈值分割后二值图像

3.3.1 直方图均衡化

■ 一般实现方法采用如下几步：

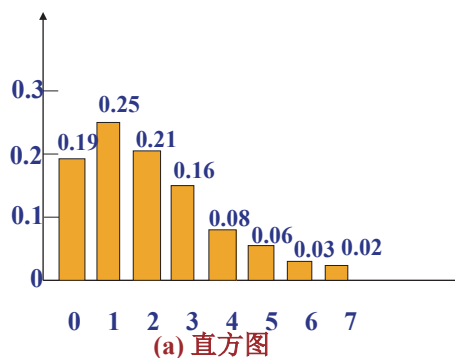
- 1、统计原始图像的直方图，求出 $P_r(r_k)$ ；
- 2、用累积分布函数作变换 $s_k = \sum_{j=0}^k P_r(r_j)$ ，求变换后的新灰度；
- 3、用新灰度代替旧灰度，求出 $P_s(s_k)$ ，这一步是近似的，力求合理，同时把灰度相等的或相近的合在一起。

3.3.1 直方图均衡化

直方图均衡化计算示例：

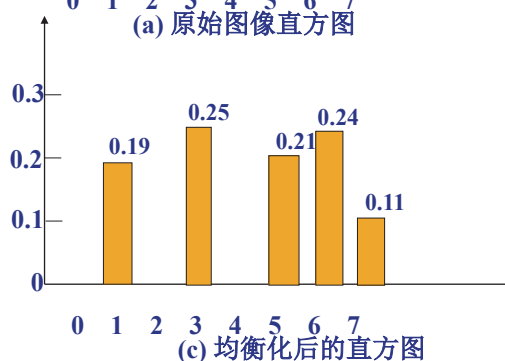
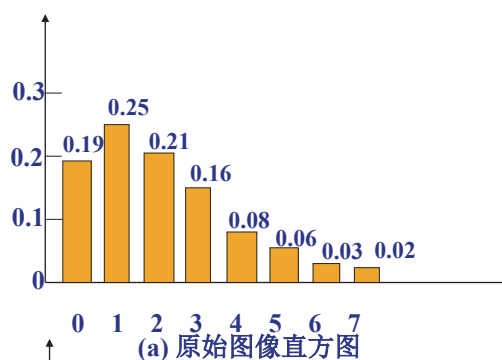
64×64大小的图像灰度分布表

r_k	n_k	$P_r(r_k) = \frac{n_k}{n}$
r0=0	790	0.19
r1=1/7	1023	0.25
r2=2/7	850	0.21
r3=3/7	656	0.16
r4=4/7	329	0.08
r5=5/7	245	0.06
r6=6/7	122	0.03
r7=1	81	0.02



序号	运算	步骤和结果							
1	列出原始图像灰度级 r_k	0	1	2	3	4	5	6	7
2	统计各灰度级像素个数 n_k	790	1023	850	656	329	245	122	81
3	计算原始直方图 $P_r(r_k)$	0.19	0.25	0.21	0.16	0.08	0.06	0.03	0.02
4	计算累积直方图 S_k	0.19	0.44	0.65	0.81	0.89	0.95	0.98	1.00
5	归一化灰度级 $r_k/(max-min)$	0	0.14	0.29	0.43	0.57	0.7	0.86	1
6	确定映射关系 ($r_k \rightarrow t_k$)	0→1	1→>3	2→>5	3,4→6	5,6,7→7			
7	统计新直方图各灰度级像素个数 n_k		790		1023		850	985	448
8	计算新直方图		0.19		0.25		0.21	0.24	0.11

直方图均衡化计算列表

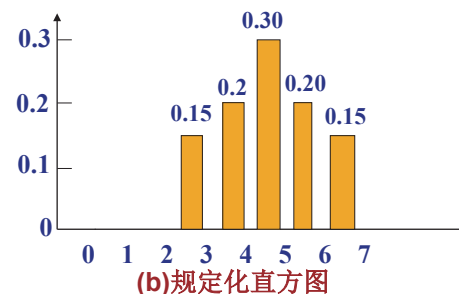
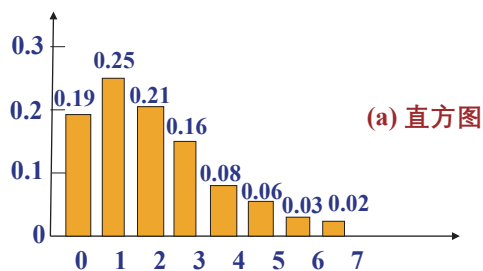


3.3.2 直方图规定化（直方图匹配）

直方图规定化计算示例1：

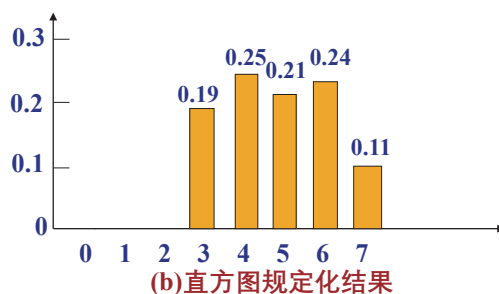
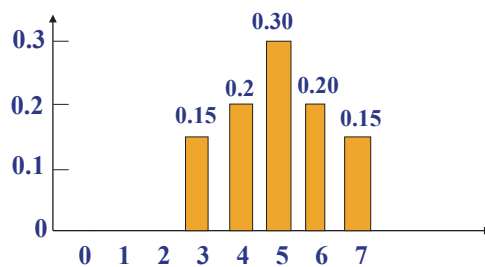
64×64大小的图像灰度分布表

r_k	n_k	$P_r(r_k) = \frac{n_k}{n}$
r0=0	790	0.19
r1=1/7	1023	0.25
r2=2/7	850	0.21
r3=3/7	656	0.16
r4=4/7	329	0.08
r5=5/7	245	0.06
r6=6/7	122	0.03
r7=1	81	0.02

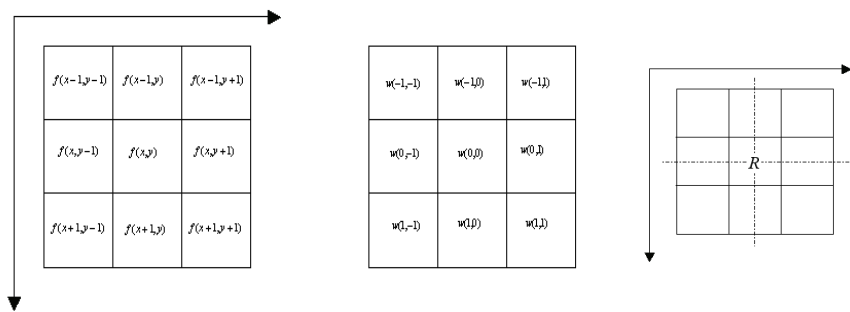


序号	运算	步骤和结果							
1	列出原始图像灰度级rk	0	1	2	3	4	5	6	7
2	统计各灰度级像素个数nk	790	1023	850	656	329	245	122	81
3	计算原始直方图Pr(rk)	0.19	0.25	0.21	0.16	0.08	0.06	0.03	0.02
4	计算累积直方图Sk	0.19	0.44	0.65	0.81	0.89	0.95	0.98	1.00
5	列出规定直方图zk	0	0	0	0.15	0.20	0.30	0.20	0.15
6	计算规定累积直方图G(zk)	0	0	0	0.15	0.35	0.65	0.85	1.00
7	确定映射关系(Sk->G(zk))	0->3	1->4	2->5	3,4->6	5,6,7->7			
8	计算新直方图				0.19	0.25	0.21	0.24	0.11

直方图规定化计算列表



3.4 空域滤波

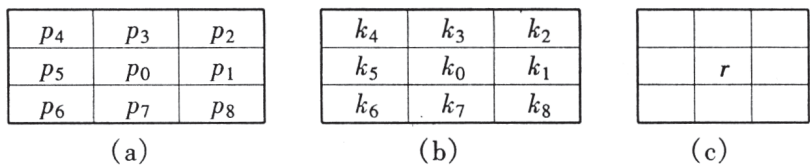


线性滤波, 滤波器模板 $m \times n$, 令 $m=2a+1$, $n=2b+1$, 则

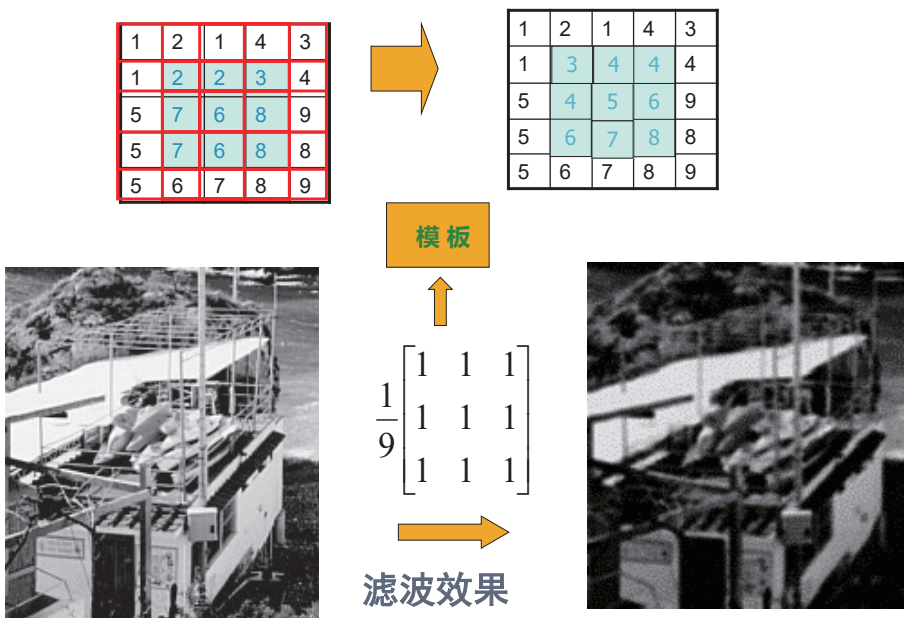
$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t)$$

空域滤波功能都是利用模板卷积, 具体过程如下:

- (1) 将模板在图像中按从左到右, 从上到下的顺序移动, 将模板中心与每个像素依次重合(边缘像素除外);
- (2) 将模板中的各个系数与其对应的像素一一相乘, 并将所有结果相加(或进行其他四则运算);
- (3) 将(2)中的结果赋给图像中对应模板中心位置的像素。



$$r = \sum_{i=0}^8 k_i p_i = k_0 p_0 + k_1 p_1 + \dots + k_8 p_8$$



$$\text{设原图像为} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 \\ 3 & 3 & 3 & 3 & 3 \\ 4 & 4 & 4 & 4 & 4 \end{bmatrix} \quad \text{模板} \quad \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1^* & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

$$\text{经过模板操作后的图像为} \begin{bmatrix} - & - & - & - & - \\ - & 2 & 2 & 2 & - \\ - & 3 & 3 & 3 & - \\ - & - & - & - & - \end{bmatrix}$$

3.4.4 非线性平滑滤波技术

- 作用：既消除噪声又保持细节（不模糊）
- 最知名的滤波器：中值（median）滤波器
- 方法：是用一个有奇数点的滑动窗口，将窗口中心点的值用窗口各点的中值代替。
- 分类：1D（1维）和2D

$$g_j = \text{median}[f_{j-r}, f_{j-r+1}, \dots, f_j, \dots, f_{j+r}]$$

- 二维中值滤波可由下式表示：

$$y_{ij} = Med_A \{f_{ij}\}$$

式中：A为窗口；{f_{ij}}为二维数据序列。

- (1) 将模板在图中漫游，并将模板中心与图中某个像素位置重合
- (2) 读取模板下各对应像素的灰度值
- (3) 将这些灰度值从小到大排成一列
- (4) 找出这些值里排在中间的一个
- (5) 将这个中间值赋给对应模板中心位置的像素

例：取3X3窗口

212	200	198
206	202	201
208	205	207

→

212	200	198
206	205	201
208	205	207

从小到大排列，取中间值

198 200 201 202 205 206 207 208 212

中值滤波的应用

1	2	1	4	3
1	2	2	3	4
5	7	6	8	9
5	7	6	8	8
5	6	7	8	9

3×3窗中值滤波
边界像素不处理→

1	2	1	4	3
1	2	3	4	4
5	5	6	6	9
5	6	7	8	8
5	6	7	8	9

2. 分离模型

❖ 例：一摄像机如图所示位置观察场景。摄像机中心位置 $(0, 0, 1)$ ，摄像机焦距 0.05m ，扫视角 135° ，倾斜角 135° ，现需要确定空间点 $W(1, 1, 0)$ 的图像平面坐标。

这个不重合的摄像机模型可通过以下一系列步骤转换为前面的重合模型：

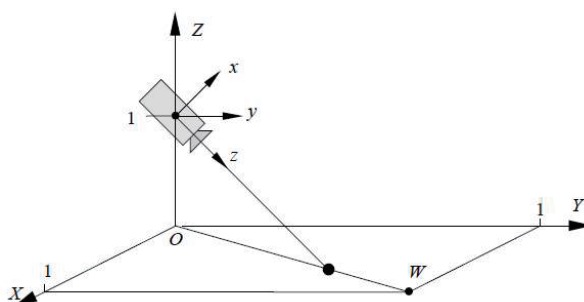
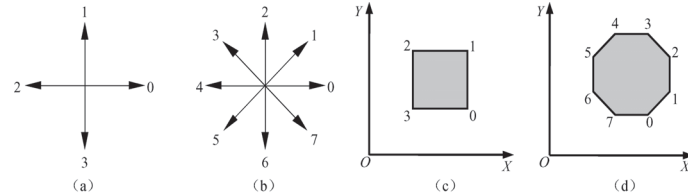


图 3.1.3 摄像机观察三维场景示意图

6.1 基于边界的表达

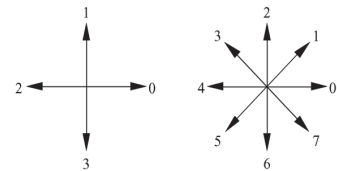
■ 链码

- 链码是一种用若干条具有特定长度和方向的线段连接起来表示目标边界的方法。
- 每个线段的长度固定而方向数目取为有限，所以只有边界的起点需用（绝对）坐标表示，其余点都可只用接续方向来代表偏移量。

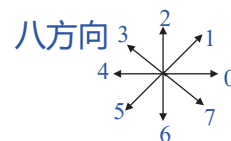
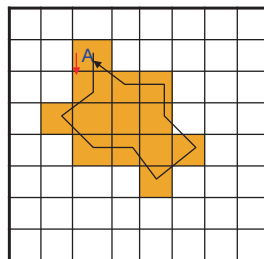


■ 链码算法

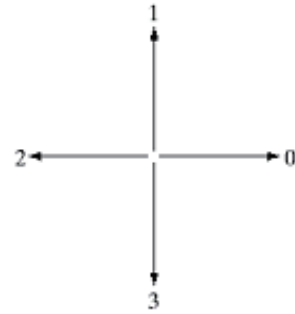
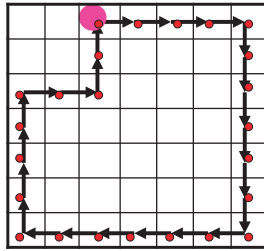
- 给每一个线段边界一个方向编码。
- 常用有4-链码和8-链码两种编码方法。
- 从起点开始，沿边界编码，至起点被重新碰到，结束一个对象的编码。



- 选边界上一点(用坐标表示)作为起点，其它点用方向数来表示：

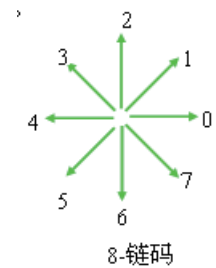
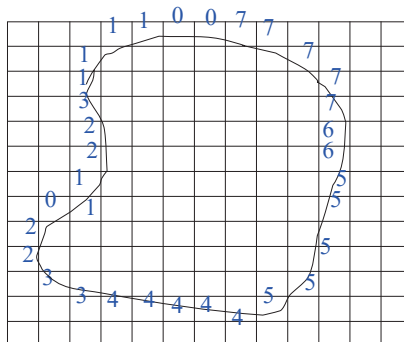


➤ 链码举例：



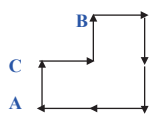
4-链码：00003333322222211110011

➤ 链码举例：



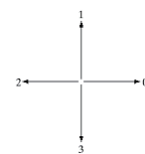
■ 链码问题1

对同一边界，如果用不同的边界点作为链码起点，得到的链码是不同的。



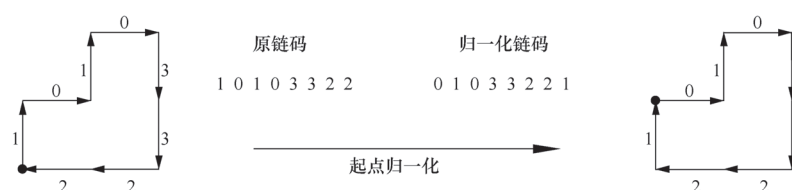
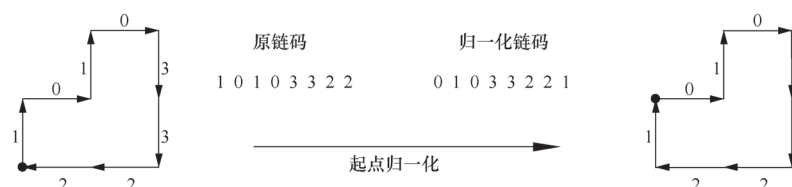
以A为起点，链码为：A: 10103322

以B为起点，链码为：B: 03322101



■ 改进：链码起点归一化

把链码看作一个由各方向数构成的自然数。将这些方向数依一个方向循环以使它们所构成的自然数的值最小。



■ 链码问题2

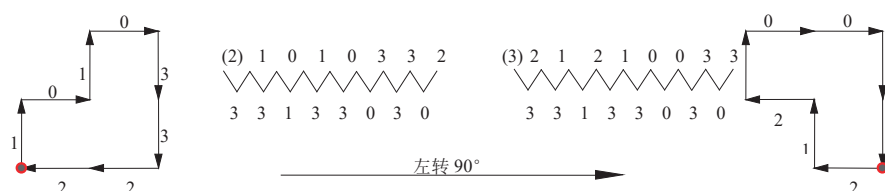
由于角度的不同，造成编码的不同。

■ 改进

通过使用链码的一阶差代替码子本身的方式。

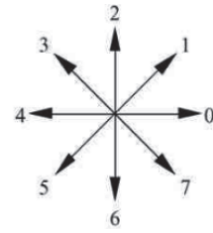
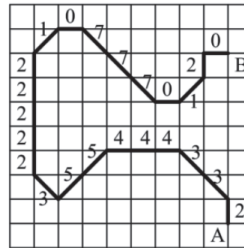
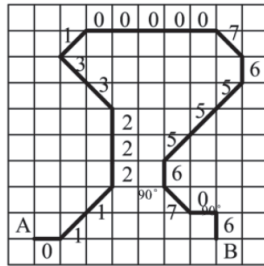
■ 链码旋转归一化

利用链码的一阶差分来重新构造一个序列（一个表示原链码各段之间方向变化的新序列）这个差分可用相邻两个方向数相减得到。（逆时针）



■ 链码旋转归一化

利用链码的一阶差分来重新构造一个序列（一个表示原链码各段之间方向变化的新序列）这个差分可用相邻两个方向数相减得到。



01122233100000765556706
1010010670000777001116

23344455322222107770120
1010010670000777001116