

3.38. 先应用平滑核,再应用拉普拉斯核,先平滑会导致一些高频细节被模糊掉,拉普拉斯核在应用时可能无法恢复已经模糊的细节,得到的结果会是细节不太清晰的图像,如在原始图像上进行增强的效果。  
 先应用拉普拉斯核会增强细节,然后再平滑时可能会降低增强的效果,但原本的高频信息仍会部分保留。  
 导致结果不如预期

3.45 (a) 最小的盒式核大小为  $3 \times 3$ , 该大小的盒式核能够填补最多3像素的裂隙, 因为它可以覆盖裂隙的两侧及中间部分, 从而将非零像素扩展到中间位置, 实现“连接”的效果。

(b) 在对图像进行平滑后, 原有的二值图像会在裂隙的区域产生灰度级变化, 需要重新进行阈值化处理。为了确保平滑后的连接不会断裂, 阈值的设定应保证将这些新连接区域保留为非零像素。  
 所以最小阈值应为1, 这样, 只要像素值大于0, 就会被视为1, 从而保留所有连接区域, 避免再次断裂。

2. 中值滤波 第一个  $3 \times 3$  区域  $\begin{bmatrix} 98 & 102 & 100 \\ 89 & 98 & 90 \\ 91 & 92 & 103 \end{bmatrix}$ , 排序后:

89, 90, 91, 92, 98, 98, 100, 102, 103. 中间值为98.

按这个步骤对整个矩阵进行中值滤波, 结果为

98	102	100	97	68
89	98	97	88	71
94	92	90	80	67
75	92	92	80	68
72	85	109	95	79

空间均值滤波

对于矩阵中第一个  $3 \times 3$  区域  $\begin{bmatrix} 98 & 102 & 100 \\ 89 & 98 & 90 \\ 91 & 92 & 103 \end{bmatrix}$

平均值计算得  $(98+102+100+89+98+90+91+92+103) \div 9 = 96.2$   
四舍五入后得 96.

应用到整个矩阵, 不包括边界, 并四舍五入取整后得结果为:

96 94 85

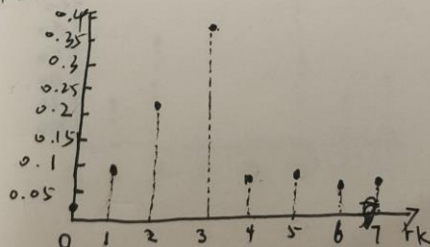
92 91 83

91 92 86

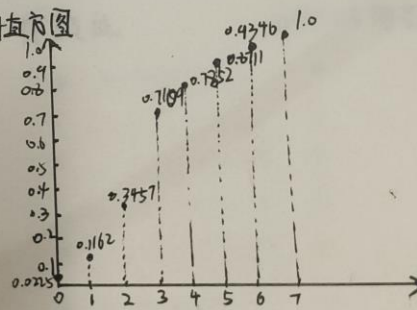
3. 1)

$r_k$	$n_k$	$P_r(r_k) = n_k / m \cdot n$
$r_0 = 0$	23	0.0225
$r_0 = 1$	96	0.0938
$r_0 = 2$	235	0.2295
$r_0 = 3$	374	0.3652
$r_0 = 4$	76	0.0742
$r_0 = 5$	88	0.0854
$r_0 = 6$	65	0.0635
$r_0 = 7$	67	0.0654

直方图



累计直方图



12) 均值化后的灰度级

$$\text{灰度级 } 0: [7 \times 0.0225 + 0.5] = 0$$

$$\text{灰度级 } 1: [7 \times 0.1162 + 0.5] = 1$$

$$\text{灰度级 } 2: [7 \times 0.3457 + 0.5] = 2$$

$$3: [7 \times 0.7109 + 0.5] = 5$$

$$4: [7 \times 0.7852 + 0.5] = 5$$

$$5: [7 \times 0.8711 + 0.5] = 6$$

$$6: [7 \times 0.9346 + 0.5] = 7$$

$$7: [\cancel{7 \times 1.0} + 0.5] = \cancel{7} = 7$$

重新计算均值化后的像素个数

$$\text{灰度级 } 0: 23 \text{ 个像素} \quad 0.0225$$

$$\text{灰度级 } 1: 46 \text{ 个} \quad 0.0938$$

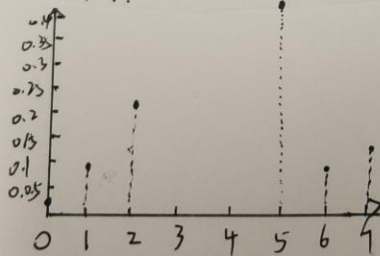
$$\text{灰度级 } 2: 233 \text{ 个} \quad 0.2295$$

$$\text{灰度级 } 5: 374 + 76 = 450 \text{ 个} \quad 0.4396$$

$$\text{灰度级 } 6: 38 \text{ 个} \quad 0.0859$$

$$\text{灰度级 } 7: 65 + 67 = 132 \text{ 个} \quad 0.1289$$

均值化的直方图



灰度级映射关系

原灰度级

均值化后的灰度级

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

计算中值滤波和空间均值滤波的代码:

```
import numpy as np
matrix = np.array([
    [98, 102, 100, 97, 68],
    [89, 98, 90, 88, 71],
    [94, 92, 103, 78, 67],
    [75, 87, 99, 80, 68],
    [72, 85, 109, 95, 79]
])

median_filtered = np.zeros(shape=(3, 3), dtype=int)
mean_filtered = np.zeros(shape=(3, 3), dtype=int)

for i in range(1, 4):
    for j in range(1, 4):
        window = matrix[i - 1:i + 2, j - 1:j + 2]
        median_filtered[i - 1, j - 1] = np.median(window)
        mean_filtered[i - 1, j - 1] = round(np.mean(window))

# 打印结果
print("中值滤波结果: ")
print(median_filtered)
print("\n空间均值滤波结果: ")
print(mean_filtered)
```