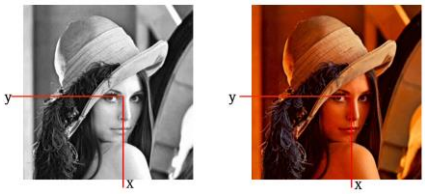
图像空间域滤波

什么是图像

- 定义为二维函数f(x,y),其中，x,y是空间坐标，f(x,y) 是点（x,y）的幅值
- 灰度图像是一个二维灰度（或亮度）函数f(x,y)
- 彩色图像由三个（如RGB,HSV）二维灰度（或亮度）函数 f(x,y)组成



图像增强

图像增强分为两类：

- 空间域增强：对图像的像素直接处理
- 频域增强： 修改图像的傅里叶变换

空间域增强：

$$g(x,y)=T[f(x,y)]$$

- f(x,y)是原图像
- g(x,y)是处理后的图像
- T是作用于f的操作，定义在(x,y)的邻域

图像增强

空间域增强： $g(x,y)=T[f(x,y)]$

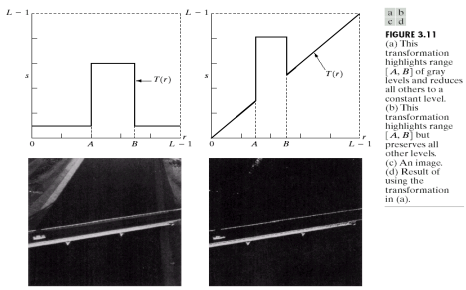
简化形式： $s=T(r)$

- r 是f(x,y)在任意点(x,y)的灰度级
- s是g(x,y)在任意点(x,y)的灰度级

图像增强-点运算

幂次变换：$s=cr^{\gamma}$

图像增强-点运算



a b
c d
FIGURE 3.11
(a) This transformation highlights range $[A, B]$ of gray levels and reduces all others to a constant level. (b) This transformation highlights range $[A, B]$ but preserves all other levels. (c) An image. (d) Result of using the transformation in (a).

图像增强-代数运算

- 算术运算：加， 减，乘， 除:一幅图像取反和另一幅图像相乘
- 逻辑运算：非， 与，或， 异或

图像增强-代数运算

加 $C(x,y) = A(x,y) + B(x,y)$

减 $C(x,y) = A(x,y) - B(x,y)$

乘 $C(x,y) = A(x,y) * B(x,y)$

除:一幅图像取反和另一幅图像相乘

图像增强-代数运算

去除叠加性噪声

对于原图像f(x,y),有一个噪声图像集 $\{g_i(x,y)\}$

其中：$g_i(x,y) = f(x,y) + h(x,y)_i$

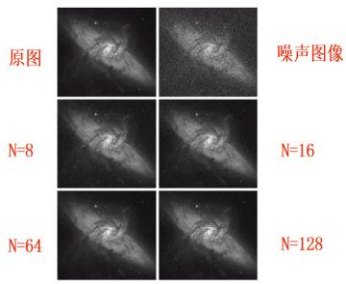
假设噪声h(x,y)均值为0，且互不相关 N个图像的均值定义为

$$g(x,y) = 1/N(g_0(x,y)+g_1(x,y)+\ldots+ g_N(x,y))$$

期望值E(g(x,y)) = f(x,y)

图像增强-代数运算

去除叠加性噪声—星系图举例

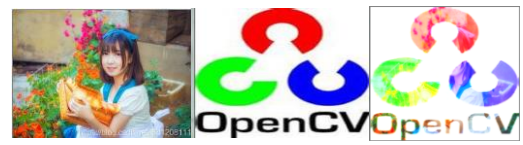


图像增强-代数运算

生成图像叠加效果

$$g(x,y) = \alpha f(x,y) + \beta h(x,y)$$

其中 $\alpha + \beta = 1$



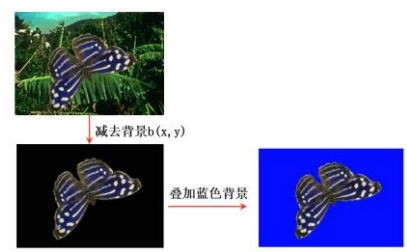
图像增强-代数运算

图像减法的主要应用举例

- 显示两幅图像的差异，检测同一场景两幅图像之间的变化如：视频中镜头边界的检测
- 去除不需要的叠加性图案
- 图像分割：如分割运动的车辆，减法去掉静止部分，剩余的是运动元素和噪声

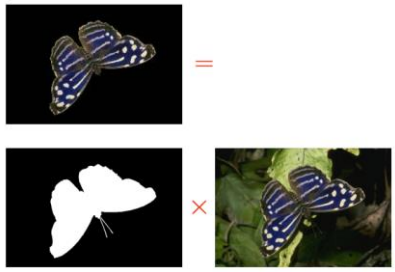
图像增强-代数运算

图像减法



图像增强-代数运算

图像乘法



13

图像增强-代数运算

非的定义 g(x,y) = 255 - f(x,y)

主要应用举例

- 获得一个阴图像
- 获得一个子图像的补图像

14

图像增强-代数运算

图像的非运算

15

图像增强-代数运算

图像的与运算

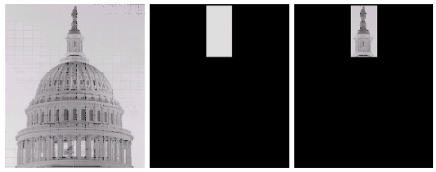
$$g(x,y) = f(x,y) \wedge h(x,y)$$

求两个子图像的相交子图

16

图像增强-代数运算

图像的与运算



17

图像增强-代数运算

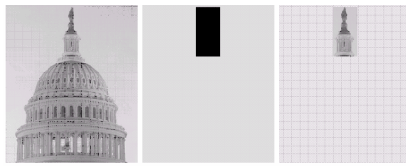
图像的或运算

$$g(x,y) = f(x,y) \vee h(x,y)$$

18

图像增强-代数运算

图像的或运算



19

图像增强-代数运算

图像的异或运算

$$g(x,y) = f(x,y) \oplus h(x,y)$$

20

图像增强-代数运算

Demo & Practice

1. 用sigmod 函数处理shenzhen灰度图

$$\frac{1}{1+e^{-ax}} - 0.5 \quad a = 0.04$$

2. 用shenzhen mask图同原图相与，得到子弹头建筑

OpenCV 常见的函数

cv2.add(src1, src2,)
cv2.subtract(src1, src2)
cv2.bitwise_and(src1, src2)
cv2.bitwise_or(src1, src2)
cv2.bitwise_xor(src1, src2)
cv2.bitwise_not(src1, src2)
cv2.multiply(src1, src2)
cv2.divide(src1,src2)

21

图像直方图

图像直方图的定义

一个灰度级在范围[0，L-1]的数字图像的直方图是一个离散函数

$$h(r_k) = n_k$$

Nk 是图像中灰度级为 rk 的像素个数
rk 是第k个灰度级，k = 0,1,2,…,L-1

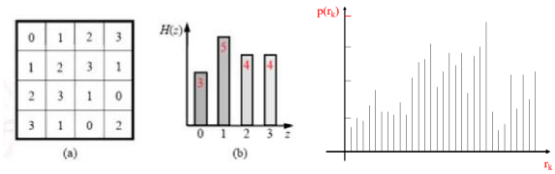
由于rk的增量是1，直方图可表示为：$h(k) = n_k$

$$h(k) = n_k / N$$

22

图像直方图

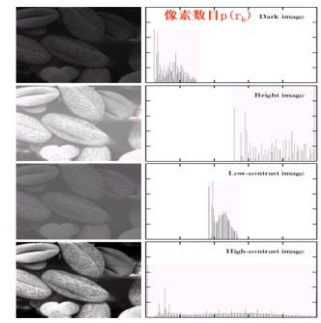
图像直方图, 概率密度



23

图像直方图

图像效果与直方图效果



24

图像直方图的应用1

直方图比对，计算图片相似度

图像直方图的应用1

直方图比对，计算图片相似度

相关系数

$$d(H_1,H_2)=\frac{\sum_I(H_1(I)-\overline{H_1})(H_2(I)-\overline{H_2})}{\sqrt{\sum_I(H_1(I)-\overline{H_1})^2\sum_I H_2(I)-\overline{H_2})^2}}$$

卡方系数

$$d(H_1,H_2)=\sum_i\frac{(H_1(i)-H_2(i))^2}{H_1(i)}$$

图像直方图的应用1

直方图比对，计算图片相似度

OpenCV中使用compareHist函数计算连个直方图的相似程度：

1.相关系数的标准(method=CV_COMP_CORREL) 值越大，相关度越高，最大值为1，最小值为0

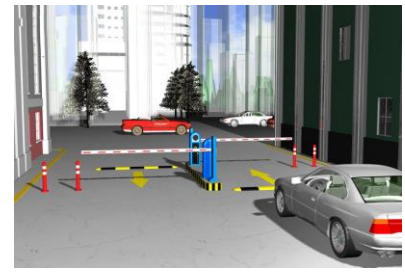
2 卡方系数的标准(method=CV_COMP_CHISQR) 值越小，相关度越高，最大值无上界，最小值0

3 相交系数的标准(method=CV_COMP_INTERSECT)值越大，相关度越高，最大值为9.455319，最小值为0

4 巴氏系数的标准(method=CV_COMP_BHATTACHARYYA) 值越小，相关度越高，最大值为1，最小值为0

图像直方图的应用2

直方图用于车牌识别



图像直方图的应用2

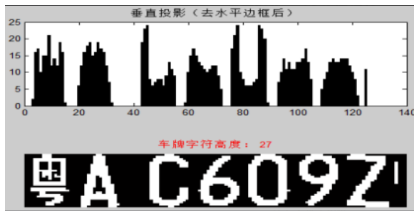
直方图用于车牌识别

图像直方图的应用2

直方图用于车牌识别

图像直方图的应用2

直方图用于车牌识别



31

图像直方图的应用2

直方图能用吗？

32

图像直方图的应用2

直方图能用吗？

33

图像增强-直方图

直方图均衡化

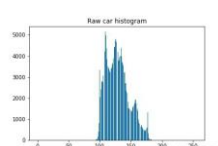
希望一幅图像的像素占有全部可能的灰度级且分布均匀，能够具有高对比度

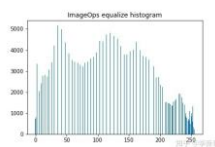
使用的方法是灰度级变换：s = T(r)

基本思想是把原始图的直方图变换为均匀分布的形式，这样就增加了像素灰度值的动态范围，从而达到增强图像整体对比度的效果

34

图像增强-直方图





35

图像增强-直方图

直方图均衡化

$$s=T(r) \quad 0 \leqslant r \leqslant 1$$

T(r) 满足下列两个条件

1. T(r) 在区间 0 ≤ r ≤ 1中为单值且单调递增
2. 当 0 ≤ r ≤ 1时, 0 ≤ T(r) ≤ 1

- 条件（1）保证原图各灰度级在变换后仍保持从黑 到白（或从白到黑）的排列次序
- 条件（2）保证变换前后灰度值动态范围的一致性

36

图像增强-直方图

直方图均衡化

T(r) 逆函数 $r = T^{-1}(s) \quad 0 \le s \le 1$

令$P_r(r)$和$P_s(s)$分别表示随机变量r和s的概率密度函数

如果$P_r(r)$和T(r)已知，且$T^{-1}(s)$满足条件1，则 $P_s(s) = P_r(r)\left|\frac{dr}{ds}\right|$

假设 $s = T(r) = \int_0^r P_r(w)dw$

$$\frac{ds}{dr} = \frac{dT(r)}{ds} = \frac{d}{ds}\left[\int_0^r p_r(w)dw\right] = p_r(r)$$

$P_S(s)=1$

均匀概率密度函数

图像增强-直方图

直方图均衡化:离散情况

离散值，处理的是它函数概率的和，而不是概率密度函数的积分

计算原图像的灰度直方图 $P(S_k) = \frac{n_k}{n}$，

其中n为像素总数，n_k为灰度级S_k的像素个数

计算原始图像的累积直方图 $CDF(S_k) = \sum_{i=0}^{k}\frac{n_i}{n} = \sum_{i=0}^{k}P_s(S_i)$

$D_j = 255 \cdot CDF(S_i)$，其中 D_j是目的图像的像素

图像增强-直方图

直方图均衡化

3	6	6	9
0	1	3	5
2	2	3	5
3	5	7	7

pix（像素值）	Ni	Pi=Ni/imgsize	sumPi	sumPi*256-1	结果四舍五入
0	1	0.0625	0.0625	15.625	16
1	1	0.0625	0.125	31	31
2	2	0.125	o.25	63	63
3	4	0.25	0.5	127	127
5	3	0.1875	0.6875	175	175
6	2	0.125	0.8125	207	207
7	2	0.125	0.9375	239	239
9	1	0.0625	1	255	255

16	207	207	255
0	31	127	175
31	31	127	175
127	175	239	239

图像增强-直方图

Demo & Practice

1. **统计像素的分布**
2. gImg_equ = cv2.equalizeHist(gImg)

Review-20210928

- **图像间的运算：**
  - 加、减、乘、除
  - 与、或、非等：

OpenCV 常见的函数

cv2.add(src1, src2,)

cv2.subtract(src1, src2)

cv2.bitwise_and(src1, src2)

cv2.bitwise_or(src1, src2)

cv2.bitwise_xor(src1, src2)

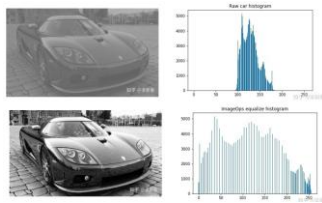
cv2.bitwise_not(src1, src2)

cv2.multiply(src1, src2)

cv2.divide(src1,src2)

Review-20210928

- 图像直方图及均衡：



动机：希望一幅图像的像素占有全部可能的灰度级且分布均匀，能够具有高对比度

Review-20210928

- 图像直方图及均衡：

计算原图像的灰度直方图 $P(S_k)=\frac{n_k}{n}$，
其中n为像素总数，n_k为灰度级S_k的像素个数

计算原始图像的累积直方图 $CDF(S_k)=\sum_{i=0}^{k}\frac{n_i}{n}=\sum_{i=0}^{k}P_s(S_i)$

$D_j=255\cdot CDF(S_i)$，其中 D_j是目的图像的像素

Review-20210928

- 图像直方图及均衡：

计算原图像的灰度直方图 $P(S_k)=\frac{n_k}{n}$，
其中n为像素总数，n_k为灰度级S_k的像素个数

计算原始图像的累积直方图 $CDF(S_k)=\sum_{i=0}^{k}\frac{n_i}{n}=\sum_{i=0}^{k}P_s(S_i)$

$D_j=255\cdot CDF(S_i)$，其中 D_j是目的图像的像素

Review-20210928

3	6	6	9
0	1	3	5
2	2	3	5
3	5	7	7

pix（像素值）	Ni	Pi=Ni/imgsize	sumPi	sumPi*256-1	结果四舍五入
0	1	0.0625	0.0625	15.625	16
1	1	0.0625	0.125	31	31
2	2	0.125	0.25	63	63
3	4	0.25	0.5	127	127
5	3	0.1875	0.6875	175	175
6	2	0.125	0.8125	207	207
7	2	0.125	0.9375	239	239
9	1	0.0625	1	255	255

16	207	207	255
0	31	127	175
31	31	127	175
127	175	239	239

图像增强-直方图

直方图规定化 ： 直方图匹配，用于将图像变换为某一特定的灰度分布

在一些模式识别问题中，为了增强数据的一致性，有时需要对图象的直方图进行匹配处理，即使得输入的图象经过变换后具有与参考图象相同的直方图，然后再进行后续处理。直方图匹配是通过以直方图均衡化变换为中介来实现的。

借助直方图变换实现规定/特定的灰度映射

图像增强-直方图

直方图规定化

$P_r(r)$ $P_z(z)$ 假设分别为原始图像和希望得到的图像的概率密度函数
r, z: 分别代表原始图像和希望得到图像的灰度级

首先对原始图像进行直方图均衡化，即求变换函数 $S=T(r)=\int_0^r P_r(r)dr$

对目标图像进行直方图均衡化处理 $V=G(Z)=\int_0^z p_z(z)dz$

它的逆变换为 $Z=G^{-1}(V)$

因为都是均衡化，所以 S== V

$Z=G^{-1}(S)$ $Z=G^{-1}(T(r))$

图像增强-直方图

直方图规定化：离散情况

(1) 对原始直方图进行灰度均衡化

$$t_k = E_{Hs}(s_i) = \sum_{i=0}^{k} p_s(s_i) \qquad k = 0,\ 1,\ \cdots,\ M-1$$

(2) 对目标图进行灰度均衡化

$$v_l = E_{Hu}(u_j) = \sum_{j=0}^{l} p_u(u_j) \qquad l = 0,\ 1,\ \cdots,\ N-1$$

(3) 将原始直方图对应映射到规定直方图：t = v

图像增强-直方图

直方图规定化：离散情况

1. 对源图进行直方图均衡化
2. 对目标图进行进行直方图均衡化
3. 确定源图像直方图与规定直方图的对应映射关系，原则是针对源图像均衡化后的直方图的每一个灰度级概率密度，查找最接近的规定直方图灰度概率密度，建立灰度映射表。
4. 根据映射结果对像素点进行处理

图像增强-直方图

直方图规定化：离散情况

映射规则：

单映射规则：从小到大依次找能使下式最小的k和l

$$\left| \sum_{i=0}^{k} p_s(s_i) - \sum_{j=0}^{l} p_u(u_j) \right| \qquad \begin{matrix} k = 0,\ 1,\ \cdots,\ M-1 \\ l = 0,\ 1,\ \cdots,\ N-1 \end{matrix}$$

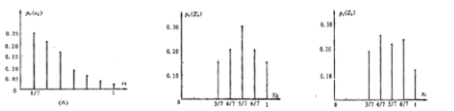
图像增强-直方图

序号	运算	步骤和结果							
1	原始图像灰度级	0	1	2	3	4	5	6	7
2	原始直方图各灰度级像素	790	1023	850	656	329	245	122	81
3	原始直方图$P(r)$	0.19	0.25	0.21	0.16	0.08	0.06	0.03	0.02
4	原始累积直方图V_1	0.19	0.44	0.65	0.81	0.89	0.95	0.98	1.00
5	规定直方图$P(z)$	0	0	0	0.15	0.20	0.30	0.20	0.15
6	规定累积直方图V_2	0	0	0	0.15	0.35	0.65	0.85	1.00
7	映射$\lvert V_2 - V_1 \rvert$最小	3	4	5	6	6	7	7	7
8	确定映射关系	0->3	1->4	2->5	3,4->6		5,6,7->7		
9	变换后直方图	0	0	0	0.19	0.25	0.21	0.24	0.11

图像增强-直方图

直方图规定化 vs. 直方图均衡化

直方图均衡化：自动增强 效果不易控制 总得到全图增强的结果	直方图规定化：有选择地增强 须给定需要的直方图 可特定增强的结果



原图像的直方图　　规定的直方图　　规定化后图像的直方图

图像增强-直方图

练习：直方图规定化

k	$p(s_k)$	$p(u_k)$
0	0.1	0
1	0.05	0.3
2	0.15	0
3	0.2	0.45
4	0.2	0
5	0.15	0
6	0.05	0.25
7	0.1	0

图像增强-空间滤波

- 空间滤波和空间滤波器的定义：使用空间模板进行的图像处理，被称为空间滤波。
- 模板本身被称为空间滤波器

在 $M\times N$ 的图像f上，使用 $m\times n$ 的滤波器：

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

w(s,t)是滤波器系数，f(x,y)是图像值

图像增强-空间滤波

平滑空间滤波器的作用

- ✓ 模糊处理：去除图像中一些不重要的细节
- ✓ 减小噪声

平滑空间滤波器的分类

- ✓ 线性滤波器：均值滤波器
- ✓ 非线性滤波器：最大值滤波器，中值滤波器，最小值滤波器

图像增强-空间滤波

线性滤波器的定义

在 $M\times N$ 的图像f上，使用 $m\times n$ 的线性滤波器：

其中，m=2a+1,n=2b+1,

w(s,t)是滤波器系数，f(x,y)是图像值

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

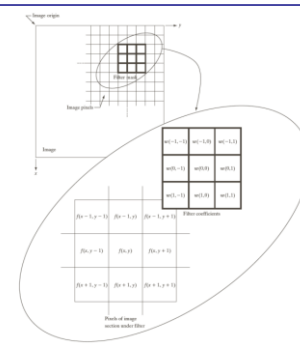
空间滤波的简化形式：

$$g(x,y)=w_1z_1+w_2z_2+\cdots w_{nn}z_{mn}=\sum_{i=1}^{mn}w_iz_i$$

其中，w是滤波器系数，z是与该系数对应的图像灰度值，mn为滤波器中包含的像素点总数

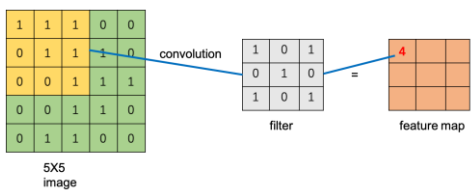
图像增强-空间滤波

线性滤波器的定义



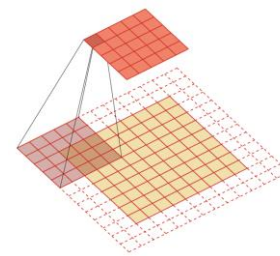
图像增强-空间滤波

图像卷积



图像增强-空间滤波

图像卷积



图像增强-空间滤波

均值滤波器: 包含在滤波器邻域内像素的平均值

作用

- 减小图像灰度的"尖锐"变化，减小噪声
- 由于图像边缘是由图像灰度尖锐变化引起的，所以也存在边缘模糊的问题

a

$\frac{1}{9}\times$

1	1	1
1	1	1
1	1	1

b

$\frac{1}{16}\times$

1	2	1
2	4	2
1	2	1

61

图像增强-空间滤波

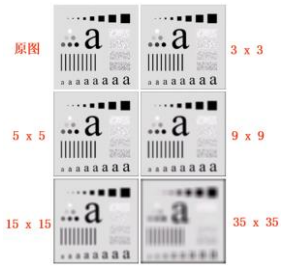
像素的加权平均, 表明一些像素更为重要

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

62

图像增强-空间滤波

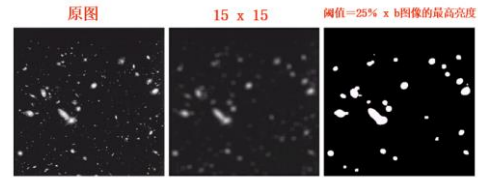
Different filter size:



63

图像增强-空间滤波

Mean Filter



a b c

FIGURE 3.36 (a) Image from the Hubble Space Telescope. (b) Image processed by a 15 × 15 averaging mask. (c) Result of thresholding (b). (Original image courtesy of NASA.)

64

图像增强-空间滤波

非线性滤波器:

统计排序滤波器:

- 一种非线性滤波器
- 基于滤波器所在图像区域中像素的排序，由 排序结果决定的值代替中心像素的值

分类

- 中值滤波器： 用像素领域内的中间值代替该像素
- 最大值滤波器：用像素领域内的最大值代替该像素
- 最小值滤波器：用像素领域内的最小值代替该像素

65

图像增强-空间滤波

统计排序滤波器:

- 中值滤波器
  - ✓主要用途：去除噪声
  - ✓计算公式：$R = mid\ \{z_k \mid k = 1,2,\ldots,n\}$
- 最大值滤波器
  - ✓主要用途：寻找最亮点
  - ✓计算公式：$R = max\ \{z_k \mid k = 1,2,\ldots,n\}$
- 最小值滤波器
  - ✓主要用途：寻找最暗点
  - ✓计算公式：$R = min\ \{z_k \mid k = 1,2,\ldots,n\}$

66

图像增强-空间滤波

中值滤波器

用模板区域内像素的中间值，作为结果值

$$R = mid\ \{z_k \mid k = 1, 2, \ldots, n\}$$

强迫突出的亮点（暗点）更象它周围的值，以消除孤立的亮点（暗点）

将模板区域内的像素排序，求出中间值

中值滤波算法的特点

- 在去除噪音的同时，可以比较好地保留边的锐度和图像的细节
- 能够有效去除脉冲噪声：以黑白点叠加在图像上

图像增强-空间滤波

中值滤波算法的实现

将模板区域内的像素排序，求出中间值

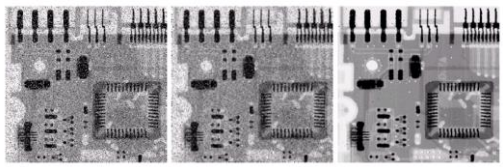
例如：3x3的模板，第5大的是中值，
5x5的模板，第13大的是中值，
7x7的模板，第25大的是中值，
9x9的模板，第41大的是中值。

如（10,15,20,20,20,20,20,25,100）

图像增强-空间滤波

中值滤波器

原图　3x3均值滤波　3x3中值滤波



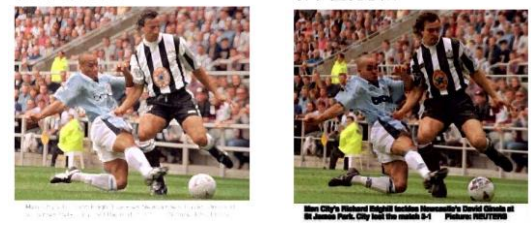
a b c

FIGURE 3.37 (a) X-ray image of circuit board corrupted by salt-and-pepper noise. (b) Noise reduction with a 3×3 averaging mask. (c) Noise reduction with a 3×3 median filter. (Original image courtesy of Mr. Joseph E. Pascente, Lixi, Inc.)

图像增强-空间滤波

最大最小值滤波器:

最大值滤波器　最小值滤波器



图像增强-空间滤波

Opencv functions:

cv2.blur(img,ksize) 均值滤波　$K = \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$

cv2.medianBlur(img, k) 中值滤波

```
#用中值法
for y in xrange(1,myh-1):
    for x in xrange(1,myw-1):
        lbimg[y,x]=np.median(tmpimg[y-1:y+2,x-1:x+2]
```


图像增强-空间滤波

Practice:

实现：基于阈值的（超限）邻域平均法

如果某个像素的灰度值大于其邻域像素的平均值，且达到了一定水平，则判断该像素为噪声，继而用邻域像素的均值取代这一像素值：

$$g(i,j) = \begin{cases} \frac{1}{N \times N} \sum_{(x,y) \in A} f(x,y), & \left| f(i,j) - \frac{1}{N \times N} \sum_{(x,y) \in A} f(x,y) \right| > T \\ f(i,j), & \text{其它} \end{cases}$$

T为某一阈值

图像增强-空间滤波

锐化滤波器：

锐化滤波器的主要用途

- 突出图像中的细节，增强被模糊了的细节
- 印刷中的细微层次强调。弥补扫描对图像的钝化
- 超声探测成像，分辨率低，边缘模糊，通过锐化来改善
- 图像识别中，分割前的边缘提取
- 锐化处理恢复过度钝化、暴光不足的图像

图像增强-空间滤波

锐化滤波器：

微分滤波器的原理：均值产生钝化的效果，而均值与积分相 似，由此而联想到，微分能不能产生相反 的效果，即锐化的效果？结论是肯定的。

锐化可用微分来完成，而微分算术的响应强度与图像在该点的突变程度有关

锐化滤波器的分类

- 一阶微分滤波器 - 梯度算子
- 二阶微分滤波器 - 拉普拉斯算子

图像增强-空间滤波

锐化滤波器：

图像函数f (x, y)在点(x, y)处的梯度是一个矢量，定义为

$$\frac{\partial f(x,y)}{\partial x} = \lim_{\epsilon \to 0} \frac{f(x+\epsilon,y) - f(x,y)}{\epsilon}$$

$$\frac{\partial f(x,y)}{\partial y} = \lim_{\epsilon \to 0} \frac{f(x,y+\epsilon) - f(x,y)}{\epsilon}$$

因为图像是一个离散的二维函数

$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y) = gx \quad \text{垂直方向}$$

$$\frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y) = gy \quad \text{水平方向}$$

图像增强-空间滤波

锐化滤波器：

图像函数f (x, y)在点(x, y)处的梯度是一个矢量，矢量模为

$$M(x,y) = mag(\nabla f) = \sqrt{g_x^2 + g_y^2} \qquad M(x,y) \approx |g_x| + |g_y|$$

梯度的方向在函数f(x, y)最大变化率的方向上

$$\alpha(x,y) = \arctan\left[\frac{g_y}{g_x}\right]$$

图像增强-空间滤波

锐化滤波器：

刚才是图像的垂直和水平梯度，但我们有时候也需要对角线方向的梯度，定义如下

$$g_x = \frac{\partial f}{\partial x} = f(x+1,y+1) - f(x,y)$$

$$g_y = \frac{\partial f}{\partial y} = f(x+1,y) - f(x,y+1)$$

-1	0
0	1

0	-1
1	0

Roberts 算子

2*2大小的模板在概念上过于简单，不是特别有效。

图像增强-空间滤波

锐化滤波器：

由梯度的计算可知，

1) 图像中灰度变化较大的边缘区域其梯度值大，

2) 灰度变化平缓的区域其梯度值较小，

3) 而在灰度均匀区域其梯度值为零。

图 (b)是采用水平垂直差分法对图(a)锐化的结果，锐化后仅留下灰度值急剧变化的边沿处的点。

图像梯度锐化结果

(a) 二值图像； (b) 梯度运算结果

图像增强-空间滤波

锐化滤波器： Prewitt 算子

Prewitt算子是一阶微分算子的边缘检测，利用像素点上下、左右邻点的灰度差，在边缘处达到极值检测边缘，去掉部分伪边缘，对噪声具有平滑作用。两个方向模板一个检测水平边缘，一个检测垂直方向的边缘。

z_1	z_2	z_3
z_4	z_5	z_6
z_7	z_8	z_9

3*3 模板

水平方向：$g_x = \frac{\partial f}{\partial x} = (z_7 + z_8 + z_9) - (z_1 + z_2 + z_3)$

垂直方向：$g_y = \frac{\partial f}{\partial y} = (z_3 + z_6 + z_9) - (z_1 + z_4 + z_7)$

-1	-1	-1
0	0	0
1	1	1

水平

-1	0	1
-1	0	1
-1	0	1

垂直

79

图像增强-空间滤波

锐化滤波器：Sobel算子

Sobel算子是在Prewitt算子的基础上改进的，在中心系数上使用一个权值2，相比较Prewitt算子，Sobel模板能够较好的抑制（平滑）噪声。

$g_x = \frac{\partial f}{\partial x} = (z_7 + 2z_8 + z_9) - (z_1 + 2z_2 + z_3)$

$g_y = \frac{\partial f}{\partial y} = (z_3 + 2z_6 + z_9) - (z_1 + 2z_4 + z_7)$

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

水平

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

垂直

80

图像增强-空间滤波

锐化滤波器：二阶微分算子

基于二阶微分的图像增强—拉普拉斯算子 $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

$$\frac{\partial f}{\partial x} = f'(x) = f(x+1) - f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = \frac{\partial f'(x)}{\partial x} = f'(x+1) - f'(x)$$
$$= f(x+2) - f(x+1) - f(x+1) + f(x)$$
$$= f(x+2) - 2f(x+1) + f(x)$$

$$\frac{\partial^2 f}{\partial x^2} = f''(x) = f(x+1) + f(x-1) - 2f(x)$$

81

图像增强-空间滤波

锐化滤波器：二阶微分算子

基于二阶微分的图像增强—拉普拉斯算子 $\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$

$$\nabla^2 f(x,y) = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2}$$
$$= \{f(x+1,y) + f(x-1,y) - 2f(x,y)\} + \{f(x,y+1) + f(x,y-1) - 2f(x,y)\}$$
$$= f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y)$$

0	1	0
1	-4	1
0	1	0

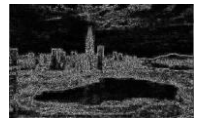
82

图像增强-空间滤波

锐化滤波器：二阶微分算子

为了让该mask在45度的方向上也具有该性质，对该filter mask进行扩展

1	1	1
1	-8	1
1	1	1



83

图像增强-空间滤波

opencv

dst=cv2.Sobel(src, ddepth, dx, dy, ksize)

1. 第一个参数是需要处理的图像；
2. 第二个参数是图像的深度，-1表示采用的是与原图像相同的深度。目标图像的深度必须大于等于原图像的深度；
3. dx和dy表示的是求导的阶数，0表示这个方向上没有求导，一般为0、1、2。

其中ddepth表示目标图像的所需深度，它包含有关图像中存储的数据类型的信息，可以是unsigned char（CV_8U），signed char（CV_8S），unsigned short（CV_16U）等等。

当ddepth=-1时，表示输出图像与原图像有相同的深度。

84

图像增强-空间滤波

opencv

dst=cv2.Laplacian(src, ddepth, ksize)

1. 第一个参数是需要处理的图像；
2. 第二个参数是图像的深度，-1表示采用的是与原图像相同的深度。目标图像的深度必须大于等于原图像的深度；
3. ksize: 是算子的大小，必须为1、3、5、7。默认为1

其中ddepth表示目标图像的所需深度，它包含有关图像中存储的数据类型的信息，可以是unsigned char（CV_8U），signed char（CV_8S），unsigned short（CV_16U）等等。

当ddepth=-1时，表示输出图像与原图像有相同的深度。

图像增强-空间滤波

opencv

dst=cv.filter2D(src, ddepth, kernel)

src	原图像
dst	目标图像，与原图像尺寸和通过数相同
ddepth	目标图像的所需深度

其中ddepth表示目标图像的所需深度，它包含有关图像中存储的数据类型的信息，可以是unsigned char（CV_8U），signed char（CV_8S），unsigned short（CV_16U）等等。

当ddepth=-1时，表示输出图像与原图像有相同的深度。

Review：20211019

平滑空间滤波器的分类

- 线性滤波器：均值滤波器
- 非线性滤波器: 最大值滤波器, 中值滤波器, 最小值滤波器

锐化滤波器的分类

- 一阶微分滤波器 - 梯度算子
- 二阶微分滤波器 - 拉普拉斯算子

图像锐化目的：加强图像轮廓和边缘，使图像看起来比较清晰、以便于对目标的识别和处理。图像锐化和平滑恰恰相反，它是通过增强高频分量来减少图像中的模糊，因此也称为高通滤波

Review：20211019

$$\frac{\partial f(x,y)}{\partial x} = f(x+1,y) - f(x,y) = gx$$

$$\frac{\partial f(x,y)}{\partial y} = f(x,y+1) - f(x,y) = gy$$

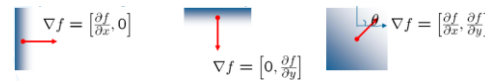
$$M(x,y) = mag(\nabla f) = \sqrt{g_x^2 + g_y^2}$$

$$\alpha(x,y) = \arctan\left[\frac{g_y}{g_x}\right]$$

梯度方向和边缘的方向总是正交(垂直)

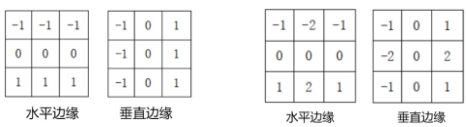
Gy梯度：反映像素垂直方向的变化，检测水平边缘

Gx梯度：反映像素水平方向的变化，检测垂直边缘



Review：20211019

锐化滤波器： Prewitt 算子， Sobel算子

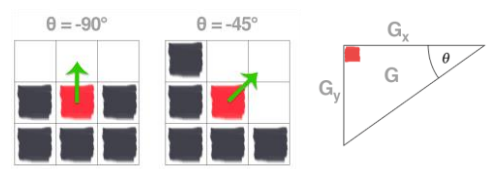


dst=cv2.Sobel(src, ddepth, dx, dy, ksize)

$$G_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * A \quad \text{and} \quad G_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * A$$

Review：20211019

理解梯度的角度



Review：20211019

锐化滤波器：二阶微分算子

$$\nabla^2 f(x,y)=\frac{\partial^2 f}{\partial x^2}+\frac{\partial^2 f}{\partial y^2}$$
$$=\{f(x+1,y)+f(x-1,y)-2f(x,y)\}+\{f(x,y+1)+f(x,y-1)-2f(x,y)\}$$
$$=f(x+1,y)+f(x-1,y)+f(x,y+1)+f(x,y-1)-4f(x,y)$$

0	1	0
1	-4	1
0	1	0

1	1	1
1	-8	1
1	1	1

dst=cv2.Laplacian(src, ddepth, ksize)

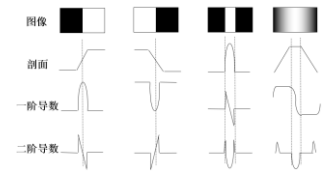
91

图像边缘检测

边缘的定义：指图像中灰度有显著变化的像素点的集合，从信号研究的频域角度而言，这些像素点信息属于高频信号区域；图像边缘往往都是闭合的连线

边缘的分类

- 阶跃状
- 阶梯状
- 脉冲状
- 屋顶状



边缘检测的基本思想：**计算局部微分算子**

92

图像边缘检测

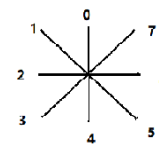
滤波 → 增强 → 检测 → 定位

1. 平滑图像、去除噪声，但是同时会减弱一定的边缘信息；
2. 求梯度值，
3. 梯度幅度值判定，初步确定图像边缘点，有时某些梯度幅度值较大点并不一定是边缘点，例如纹理图像；
4. 精确定位边缘位置
5. 边缘提取要求输出的是一个二值化图像，只有黑白两个灰度，一个表示边缘，另一个表示背景，最后还需要把边缘细化，使效果更清晰。

93

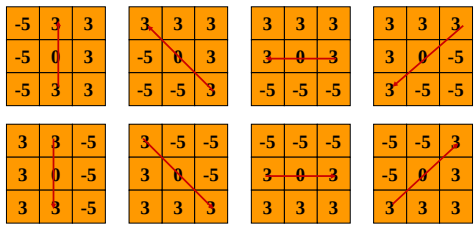
图像边缘检测

Kirsch算子是R.Kirsch提出来一种边缘检测算法，它采用8个模板对图像上的每一个像素点进行卷积求导数，这8个模板代表8个方向，对图像上的8个特定边缘方向作出最大响应，运算中取最大值作为图像的边缘输出。



94

图像边缘检测

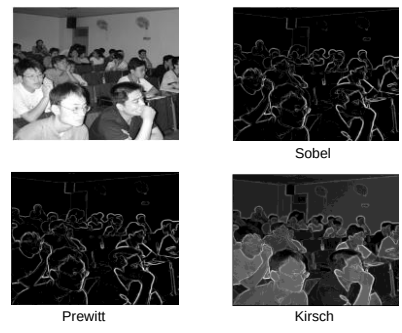


Kirsch算子特点

- 在计算边缘强度的同时可以得到边缘的方向
- 各方向间的夹角为45º

95

图像边缘检测



96

图像边缘检测

Marr算子: Laplacian of a Gaussian（LOG）

- Marr算子是在Laplacian算子的基础上实现的，它得益于对人的视觉机理的研究，有一定的生物学和生理学意义。
- 由于Laplacian算子对噪声比较敏感，为了减少噪声影响，提出了将高斯滤波和拉普拉斯检测算子结合在一起进行边缘检测的方法：先对图像进行平滑，然后再用Laplacian算子检测边缘。
- 平滑函数应能反映不同远近的周围点对给定像素具有不同的平滑作用，因此，平滑函数采用正态分布的高斯函数，即：

$$G_\sigma(x,y)=\frac{1}{2\pi\sigma^2}\exp(-\frac{x^2+y^2}{2\sigma^2})$$

图像边缘检测

卷积操作具有结合律，因此我们先将高斯平滑滤波器与拉普拉斯滤波器进行卷积，然后利用得到的混合滤波器去对图片进行卷积以得到所需的结果。

两个优点：

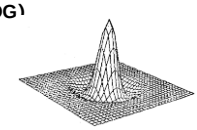
1. 由于高斯和拉普拉斯核通常都比图像小得多，所以这种方法通常只需要很少的算术运算。
2. LoG (' Laplacian of Gaussian')内核的参数可以预先计算，因此在运行时只需要对图像执行一遍的卷积即可。

$$LoG(x,y)=-\frac{1}{\pi\sigma^4}[1-\frac{x^2+y^2}{2\sigma^2}]e^{-\frac{x^2+y^2}{2\sigma^2}}$$

图像边缘检测

Marr算子: Laplacian of a Gaussian（LOG）

$$LoG(x,y)=-\frac{1}{\pi\sigma^4}[1-\frac{x^2+y^2}{2\sigma^2}]e^{-\frac{x^2+y^2}{2\sigma^2}}$$



墨西哥草帽

算法步骤如下：

1. 滤波：首先对图像f(x,y)进行平滑滤波，其滤波函数根据人类视觉特性选为高斯函数，
2. 增强：对平滑图像进行拉普拉斯运算，
3. 检测：利用二阶导数算子过零点的性质，可确定图像中阶跃边缘的位置

图像边缘检测

Marr算子: Laplacian of a Gaussian（LOG）

由于的平滑性质能减少噪声的影响，所以当边缘模糊或噪声较大时，利用检测过零点能提供较可靠的边缘位置。在该算子中，σ的选择很重要，σ小时边缘位置精度高，但边缘细节变化多；σ大时平滑作用大，但细节损失大，边缘点定位精度低。应根据噪声水平和边缘点定位精度要求适当选取σ。

图像边缘检测

Marr算子: Laplacian of a Gaussian（LOG）

下面是σ = 10时，Marr算子的模板：

−34	−25	−18	−12	−8	−7	−8	−12	−18	−25	−34
−25	−15	−7	−1	3	4	3	−1	−7	−15	−25
−18	−7	2	8	12	14	12	8	2	−7	−18
−12	−1	8	15	20	21	20	15	8	−1	−12
−8	3	12	20	24	26	24	20	12	3	−8
−7	4	14	21	26	27	26	21	14	4	−7
−8	3	12	20	24	26	24	20	12	3	−8
−12	−1	8	15	20	21	20	15	8	−1	−12
−18	−7	2	8	12	14	12	8	2	−7	−18
−25	−15	−7	−1	3	4	3	−1	−7	−15	−25
−34	−25	−18	−12	−8	−7	−8	−12	−18	−25	−34

图像边缘检测

Canny算子

Marr（LoG）边缘检测方法类似，也属于是先平滑后求导数的方法。John Canny研究了最优边缘检测方法所需的特性，给出了评价边缘检测性能优劣的三个指标：

1. 好的信噪比，即将非边缘点判定为边缘点的概率要低，将边缘点判为非边缘点的概率要低；
2. 高的定位性能，即检测出的边缘点要尽可能在实际边缘的中心；
3. 对单一边缘仅有唯一响应，即单个边缘产生多个响应的概率要低，并且虚假响应边缘应该得到最大抑制。

图像边缘检测

Canny算子

1. 减少噪音：由于边缘检测易受图像中的噪声影响，因此第一步是使用5x5高斯滤波器去除图像中的噪声.

$$\sigma = 1.4 \qquad B = \frac{1}{159}\cdot\begin{bmatrix} 2 & 4 & 5 & 4 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix}$$

$$g_\sigma(m,n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{m^2+n^2}{2\sigma^2}} \cdot f(m,n)$$

图像边缘检测

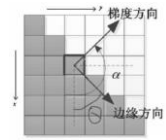
Canny算子

2. 计算图像梯度：对平滑后的图像使用sobel算子在水平与竖直方向上计算一阶导数，得到图像梯度（Gx和Gy）。根据梯度图找到边界梯度和方向

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \qquad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

$$\mathrm{G} = \sqrt{G_x^2 + G_y^2}$$

$$\theta = \operatorname{atan2}(G_y, G_x)$$

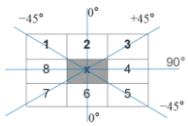


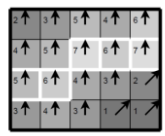
图像边缘检测

Canny算子

根据角度对幅值进行非极大值抑制：将模糊的边界变得清晰（sharp）

1. 将其梯度方向近似为以下值中的一个（0,45,90,135,180,225,270,315）（即上下左右和45度方向）
2. 比较该像素点，和其梯度方向正负方向的像素点的梯度强度
3. 如果该像素点梯度强度最大则保留，否则抑制（删除，即置为0）





以第二排第三个像素点为例，由于梯度方向向上，则将这一点的强度（7）与其上下两个像素点的强度（5和4）比较，由于这一点强度最大，则保留。

图像边缘检测

Canny算子

用双阈值算法检测和连接边缘

非极大抑制后图像中仍然有很多噪声点

对非极大值抑制图像作用两个阈值th1和th2，两者关系th1=0.4th2。我们把梯度值小于th1的像素的灰度值设为0，得到图像1。然后把梯度值小于th2的像素的灰度值设为0，得到图像2。由于图像2的阈值较高，去除大部分噪音，但同时也损失了有用的边缘信息。而图像1的阈值较低，保留了较多的信息，可以以图像2为基础，以图像1为补充来连结图像的边缘。

图像边缘检测

Canny算子

用双阈值算法检测和连接边缘

链接边缘的具体步骤如下：
对图像2进行扫描，当遇到一个非零灰度的像素p(x,y)时，跟踪以p(x,y)为开始点的轮廓线，直到轮廓线的终点q(x,y)。

考察图像1中与图像2中q(x,y)点位置对应的点s(x,y)的8邻近区域。如果在s(x,y)点的8邻近区域中有非零像素s(x,y)存在，则将其包括到图像2中，作为r(x,y)点。从r(x,y)开始，重复第一步，直到我们在图像1和图像2中都无法继续为止。

图像边缘检测

Canny算子

edge = cv2.Canny(image, threshold1, threshold2)

必要参数：

1. 第一个参数是需要处理的原图像，该图像必须为单通道的灰度图；
2. 第二个参数是阈值1；
3. 第三个参数是阈值2。

图像边缘检测

高通滤波

边缘是由灰度级跳变点构成的。因此具有较高的空间频率。所以采用高通滤波的方法让高频分量顺利通过，使低频分量得到抑制，就可增强高频分量，使图像的边缘或线条变的清晰，实现图像的锐化。

在空间域中，让图像和高通滤波器的冲击响应函数进行卷积。

$$H_1=\begin{bmatrix}0 & -1 & 0\\ -1 & 5 & -1\\ 0 & -1 & 0\end{bmatrix}$$

$$H_2=\begin{bmatrix}-1 & -1 & -1\\ -1 & 9 & -1\\ -1 & -1 & -1\end{bmatrix}$$

109