图像形态学操作

1

图像二值化

图像二值化是图像处理的基本技术，也是图像处理中一个非常活跃的分支，其应用领域非常广泛，特别是在图像信息压缩、边缘提取和形状分析等方面起着重要作用，成为其处理过程中的一个基本手段。

二值化的目的是将上步的图像增强结果转换成黑白二值图像，从而能得到清晰的边缘轮廓线，更好地为边缘提取、图像分割、目标识别等后续处理服务。

二值化的基本过程如下：

1. 对原始图像作中低通滤波，进行图像的预处理，降低或去除噪声；
2. 用算法确定最佳阈值T：凡是像素的灰度值大于这个阈值的设成255，小于这个阈值的设成0。

2

图像二值化

$$g(x,y)=\begin{cases}255\ (白) & f(x,y)\geqslant T\\ 0\ (黑) & f(x,y)<T\end{cases}$$

选取合适的分割阈值可以说是图像二值化的重要步骤，过高的阈值会导致一些真实边缘的丢失，过低的阈值又会产生一些无谓的虚假边缘。

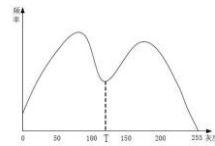
1. **双峰法**
2. **P参数法**
3. **大津法（Otsu法或最大类间方差法）**
4. **最大熵阈值法**
5. **迭代法（最佳阈值法）**

3

图像二值化

双峰法

在一些简单的图像中，物体的灰度分布比较有规律，背景与各个目标在图像的直方图各自形成一个波峰，即区域与波峰一一对应，每两个波峰之间形成一个波谷。那么，选择双峰之间的波谷所代表的灰度值T作为阈值，即可实现两个区域的分割。



4

图像二值化

P参数法

当不同区域（即目标）之间的灰度分布有一定的重叠时，双峰法的效果就很差。如果预先知道每个目标占整个图像的比例P，则可以采用P参数法进行分割。

假设已知整个直方图中目标区域所占的比例为P1：

1) 计算图像的直方图分布P(t)，其中t=0,1,2,…,255表示图像的灰度值；

2) 从最低的灰度值开始，计算图像的累积分布直方图。

$$p_1(t)=\sum_{i=0}^{t}p(i)\qquad t=0,1,2,\ldots,255,$$

3) 计算阈值T，有 $T=\operatorname{argmin}|p_1(t)-p_1|$

需要预先知道目标区域的P值，因此成为P参数法。

也就是说，阈值就是与P1最为接近的累积分布函数所对应的灰度值t。

5

图像二值化

大津法（Otsu法或最大类间方差法）

由Otsu 于1979 年提出的，是基于整幅图像的统计特性实现阈值的自动选取

基本思想是用某一假定的灰度值t将图像的灰度分成两组，当两组的类间方差最大时，此灰度值t就是图像二值化的最佳阈值。

设图像有M 个灰度值，取值范围在0～M-1，在此范围内选取灰度值t，将图像分成两组G0和G1，G0包含的像素的灰度值在0～t，G1的灰度值在t+1～M-1，用N 表示图像像素总数，ni表示灰度值为i 的像素的个数。

6

图像二值化

大津法（Otsu法或最大类间方差法）

已知：每一个灰度值i 出现的概率为 $p_i = \frac{n_i}{N}$

$$\omega_0 = \sum_{i=0}^{t} p_i \qquad \omega_1 = \sum_{i=t+1}^{M-1} p_i = 1 - \omega_0$$

平均灰度值：$u_0 = \sum_{i=0}^{t} i p_i \qquad u_1 = \sum_{i=t+1}^{M-1} i p_i$

图像的总平均灰度为：$u = \omega_0 \times u_0 + \omega_1 \times u_1$

间类方差为：$g(t) = \omega_0 (u_0 - u)^2 + \omega_1 (u_1 - u)^2 = \omega_0 \omega_1 (u_0 - u_1)^2$

最佳阈值为：T = argmax[g(t)]　使得类间方差最大时所对应的t值

图像二值化

大津法（Otsu法或最大类间方差法）

算法可这样理解：阈值T 将整幅图像分成前景和背景两部分，当两类的类间方差最大时，此时前景和背景的差别最大，二值化效果最好。因为方差是灰度分布均匀性的一种度量，方差值越大，说明构成图像的两部分差别越大，当部分目标错分为背景或部分背景错分为目标都会导致两部分差别变小，因此使类间方差最大的分割阈值意味着错分概率最小。

大律法得到了广泛的应用，但是当物体目标与背景灰度差不明显时，会出现无法忍受的大块黑色区域，甚至会丢失整幅图像的信息。

图像二值化

最大熵阈值法

将信息论中的shannon熵概念用于图像分割，其依据是使得图像中目标与背景分布的信息量最大，即通过测量图像灰度直方图的熵，找出最佳阈值。

根据shannon熵的概念，对于灰度范围为0,1,2,…,M-1的图像，其直方图的熵定义为（仅仅是定义）：

$$H = -\sum_{i=0}^{M-1} p_i \ln p_i$$　pi为灰度值为i的像素在整体图像中的概率。

图像二值化

最大熵阈值法

设阈值t将图像划分为目标O和背景B两类，他们的概率分布分别为

O区：$\frac{p_i}{P_t}$　i=0,1，…，t;

B区：$\frac{p_i}{1-P_t}$　i=t+1,t+2，…，M-1;

$$P_t = \sum_{i=0}^{t} p_i$$

则目标O和背景B的熵函数分别为：

$$H_O(t) = -\sum_{i=0}^{t} \frac{p_i}{P_t} \ln \frac{p_i}{P_t} = \ln P_t + \frac{H_t}{P_t}$$

$$H_B(t) = -\sum_{i=t+1}^{L-1} \frac{p_i}{1-P_t} \ln \frac{p_i}{1-P_t} = \ln(1-P_t) + \frac{H-H_t}{1-P_t}$$

图像二值化

最大熵阈值法

图像的总熵为　$H(t) = H_O(t) + H_B(t) = \ln P_t (1-P_t) + \frac{H_t}{P_t} + \frac{H-H_t}{1-P_t}$

最佳阈值T为使得图像的总熵取得最大值：T=argmax[H(t)]

此方法不需要先验知识，而且对于非理想双峰直方图的图像也可以进行较好的分割。缺点是运算速度较慢不适合实时处理。仅仅考虑了像素点的灰度信息，没有考虑到像素点的空间信息，所以当图像的信噪比降低时分割效果不理想。

图像二值化

迭代法（最佳阈值法）　迭代法是基于逼近的思想

1）选择一个初始阈值T(j)，通常可以选择整体图像的平均灰度值作为初始阈值。j为迭代次数，初始时j=0。

2）用T(j)分割图像，将图像分为2个区域 $C_1^{(j)}$ 和 $C_2^{(j)}$

3）计算两区域的平均灰度值，设 $N_1^{(j)}$, $N_2^{(j)}$ 分别为第j次迭代时，区域C1 和C2的像素个数，f(x,y)表示图像中（x,y）点的灰度值。

$$u_1^{(j)} = \frac{1}{N_1^{(j)}} \sum_{f(x,y) \in C_1^{(j)}} f(x,y) \qquad u_2^{(j)} = \frac{1}{N_2^{(j)}} \sum_{f(x,y) \in C_2^{(j)}} f(x,y)$$

图像二值化

迭代法（最佳阈值法） 迭代法是基于逼近的思想

4）再计算新的门限值，即 $T(j+1)=\frac{u_1^{(j)}+u_2^{(j)}}{2}$

5）令j=j+1，重复2）~4），直到T(j+1)与T(j)的差小于规定值。

13

图像二值化

OpenCV

cv2.threshold(src, thresh, maxval, type)：固定阈值

- src - 输入数组/图像（多通道，8位或32位浮点）
- thresh - 阈值
- maxval - 最大值
- type - 阈值类型
- dst - 输出数组/图像（与src相同大小和类型以及相同通道数的数组/图像）

14

图像二值化

OpenCV cv2.threshold(src, thresh, maxval, type)：固定阈值

THRESH_BINARY	$\texttt{dst}(x,y)=\begin{cases}\texttt{maxval} & \text{if } \texttt{src}(x,y)>\texttt{thresh}\\ 0 & \text{otherwise}\end{cases}$
THRESH_BINARY_INV	$\texttt{dst}(x,y)=\begin{cases}0 & \text{if } \texttt{src}(x,y)>\texttt{thresh}\\ \texttt{maxval} & \text{otherwise}\end{cases}$
THRESH_TRUNC	$\texttt{dst}(x,y)=\begin{cases}\texttt{threshold} & \text{if } \texttt{src}(x,y)>\texttt{thresh}\\ \texttt{src}(x,y) & \text{otherwise}\end{cases}$
THRESH_TOZERO	$\texttt{dst}(x,y)=\begin{cases}\texttt{src}(x,y) & \text{if } \texttt{src}(x,y)>\texttt{thresh}\\ 0 & \text{otherwise}\end{cases}$
THRESH_TOZERO_INV	$\texttt{dst}(x,y)=\begin{cases}0 & \text{if } \texttt{src}(x,y)>\texttt{thresh}\\ \texttt{src}(x,y) & \text{otherwise}\end{cases}$

15

图像二值化

OpenCV

cv2.adaptiveThreshold(src, maxValue, adaptiveMethod, thresholdType, blockSize, C])

adaptiveMethod:决定如何计算阈值

- cv2.ADAPTIVE_THRESH_MEAN_C: 阈值是邻域的平均值
- cv2.ADAPTIVE_THRESH_GAUSSIAN_C: 阈值是邻域值的高斯加权和

blockSize: 决定了邻域的大小

C: 从计算的平均值或加权平均值中减去的常数

16

图像二值化

OpenCV

Otsu's Binarization是一种基于直方图的二值化方法，它需要和threshold函数配合使用

```
ret, img2 = cv2.threshold(img,0,255,cv2.THRESH_BINARY + cv2.THRESH_OTSU)
```

17

图像形态学操作

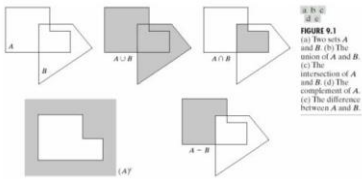
- 集合论基础知识
- 膨胀和腐蚀(Dilation & Erosion)：产生滤波器作用
- 开操作和闭操作(Opening & Closing)：产生滤波器作用
- 形态学的主要应用：边界提取、区域填充、连通分量的提取、凸壳、细化、粗化等

18

图像形态学操作

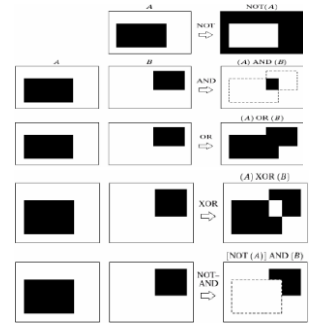
形态学图像处理的基本运算有4个：膨胀、腐蚀、开操作和闭操作

集合的并、交、补、差



图像形态学操作

二值形态学



图像形态学操作

膨胀：使图像扩大，跟卷积操作类似。

1. 用结构元素，扫描图像的每一个像素；
2. 用结构元素与其覆盖的二值图像做"与"运算
3. 如果都为0，结果图像的该像素为0，否则为1

膨胀的作用：

1. 用3x3的结构元时，物体的边界沿周边增加一个像素
2. 把目标周围的背景点合并到目标中，目标之间存在细小的缝隙，膨胀可能将不同目标连通在一起
3. 填补分割后物体中的空洞

图像形态学操作

腐蚀：使图像变小。

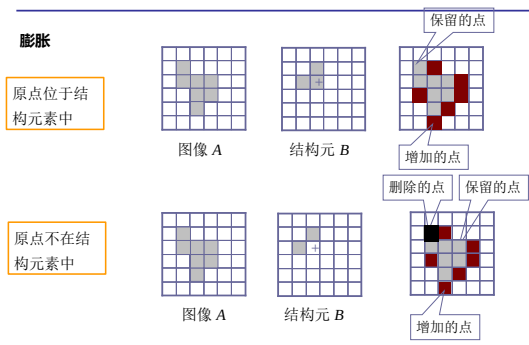
1. 用结构元素，扫描图像的每一个像素；
2. 用结构元素与其覆盖的二值图像做与运算
3. 如果结果都为1，结果图像的该像素为1，否则为0

腐蚀的作用：

1. 用3x3的结构元时，物体的边界沿周边减少一个像素
2. 消除掉图像中小于结构元大小的目标物体
3. 若物体之间有细小的连通，选择适当的结构元，可以将物体分开。
4. 不同的结构元及其不同的原点，产生不同的结果

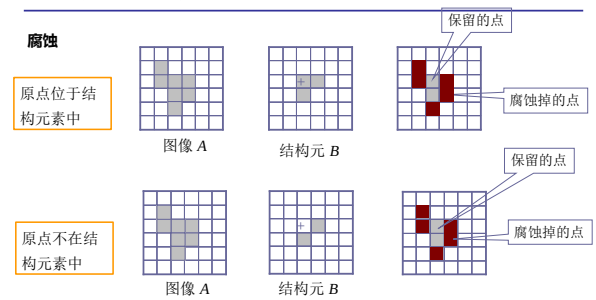
图像形态学操作

膨胀



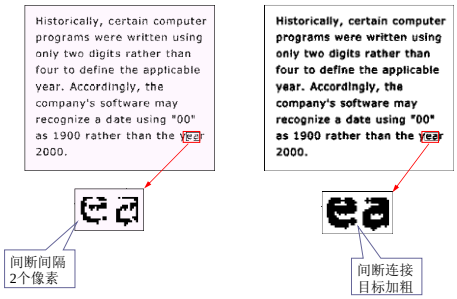
图像形态学操作

腐蚀



图像形态学操作

膨胀的应用



图像形态学操作

腐蚀的应用

图像内部边长为1、3、5、7、9和15像素的正方形图像　　一次腐蚀　　一次膨胀

结构元素为13x13，主要目的“滤除掉小于13个像素的小目标。

图像形态学操作

OpenCV

```
# 定义kernel
kernel = np.zeros((3,3), np.uint8)  or
kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3) )

result = cv2.dilate(img, kernel,iterations=3)        # 膨胀运算
result = cv2.erode(img, kernel,iterations=2)         # 腐蚀运算
```


图像形态学操作

膨胀和腐蚀组合：开闭组合运算

图像开运算是图像**依次经过腐蚀、膨胀**处理后的过程。图像被腐蚀后，去除了噪声，但是也压缩了图像；接着对腐蚀过的图像进行膨胀处理，可以去除噪声，并保留原有图像

图像闭运算是图像**依次经过膨胀、腐蚀**处理后的过程。图像先膨胀，后腐蚀，它有助于关闭前景物体内部的小孔，或物体上的小黑点。

图像形态学操作

开闭运算

开运算：腐蚀+膨胀　　**闭运算：膨胀+腐蚀**

图像形态学操作

OpenCV

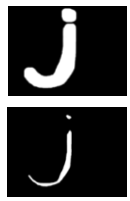
```
kernel = np.ones((15,15), np.uint8)        # 定义kernel

result = cv2.morphologyEx(img1, cv2.MORPH_OPEN, kernel)     # 开运算
result = cv2.morphologyEx(img1, cv2.MORPH_CLOSE, kernel)    # 闭运算
```

图像形态学操作

梯度运算：形态学方式　　膨胀图像减去腐蚀图像的结果，得到图像的轮廓

原图



dst = cv2.morphologyEx(src, cv2.MORPH_GRADIENT, kernel)

图像形态学操作

练习

1. 长度为3的十字架核的原点是哪里?

2. Removed the unwanted lines in morph_test.png