该系列文章是讲解Python OpenCV图像处理知识，前期主要讲解图像入门、OpenCV基础用法，中期讲解图像处理的各种算法，包括图像锐化算子、图像增强技术、图像分割等，后期结合深度学习研究图像识别、图像分类应用。希望文章对您有所帮助，如果有不足之处，还请海涵~

该系列在github所有源代码：https://github.com/eastmountyxz/ImageProcessing-Python
PS：请求帮忙点个Star，哈哈，第一次使用Github，以后会分享更多代码，一起加油。

同时推荐作者的C++图像系列知识：
[数字图像处理] 一.MFC详解显示BMP格式图片
[数字图像处理] 二.MFC单文档分割窗口显示图片
[数字图像处理] 三.MFC实现图像灰度、采样和量化功能详解
[数字图像处理] 四.MFC对话框绘制灰度直方图
[数字图像处理] 五.MFC图像点运算之灰度线性变化、灰度非线性变化、阈值化和均衡化处理详解
[数字图像处理] 六.MFC空间几何变换之图像平移、镜像、旋转、缩放详解
[数字图像处理] 七.MFC图像增强之图像普通平滑、高斯平滑、Laplacian、Sobel、Prewitt锐化详解

前文参考：
[Python图像处理] 一.图像处理基础知识及OpenCV入门函数
[Python图像处理] 二.OpenCV+Numpy库读取与修改像素
[Python图像处理] 三.获取图像属性、兴趣ROI区域及通道处理
[Python图像处理] 四.图像平滑之均值滤波、方框滤波、高斯滤波及中值滤波
[Python图像处理] 五.图像融合、加法运算及图像类型转换
[Python图像处理] 六.图像缩放、图像旋转、图像翻转与图像平移
[Python图像处理] 七.图像阈值化处理及算法对比

本篇文章主要讲解Python调用OpenCV实现图像腐蚀和图像膨胀的算法，基础性知识希望对您有所帮助。
**1.基础理论**
**2.图像腐蚀代码实现**
**3.图像膨胀代码实现**

PS：文章参考自己以前系列图像处理文章及OpenCV库函数，同时部分参考网易云视频，推荐大家去学习。同时，本篇文章涉及到《计算机图形学》基础知识，请大家下来补充。

PSS：文章参考自己以前系列图像处理文章及OpenCV库函数，同时部分参考网易云lilizong老师的视频，推荐大家去学习。同时，本篇文章涉及到《计算机图形学》基础知识，请大家下来补充。

PSS：2019年1~2月作者参加了CSDN2018年博客评选，希望您能投出宝贵的一票。我是59号，Eastmount，杨秀璋。投票地址：
https://bss.csdn.net/m/topic/blog_star2018/index


五年来写了314篇博客，12个专栏，是真的热爱分享，热爱CSDN这个平台，也想帮助更多的人，专栏包括Python、数据挖掘、网络爬虫、图像处理、C#、Android等。现在也当了两年老师，更是觉得有义务教好每一个学生，让贵州学子好好写点代码，学点技术，"师者，传到授业解惑也"，提前祝大家新年快乐。2019我们携手共进，为爱而生。

---

# 一. 基础知识

（注：该部分参考作者论文《一种改进的Sobel算子及区域择优的身份证智能识别方法》）

图像的膨胀（Dilation）和腐蚀（Erosion）是两种基本的形态学运算，主要用来寻找图像中的极大区域和极小区域。其中膨胀类似于“领域扩张”，将图像中的高亮区域或白色部分进行扩张，其运行结果图比原图的高亮区域更大；腐蚀类似于“领域被蚕食”，将图像中的高亮区域或白色部分进行缩减细化，其运行结果图比原图的高亮区域更小。

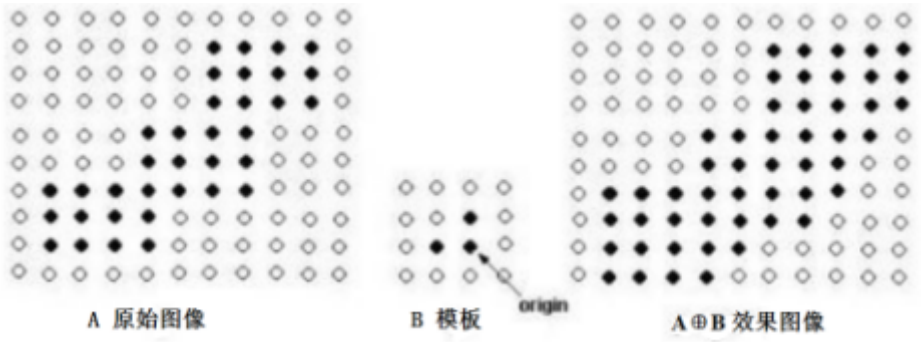
**1.图像膨胀**

膨胀的运算符是“⊕”，其定义如下：

$$A \oplus B = \{x \mid (B)_x \cap A \neq \Theta\}$$

该公式表示用B来对图像A进行膨胀处理，其中B是一个卷积模板或卷积核，其形状可以为正方形或圆形，通过模板B与图像A进行卷积计算，扫描图像中的每一个像素点，用模

板元素与二值图像元素做“与”运算，如果都为0，那么目标像素点为0，否则为1。从而计算B覆盖区域的像素点最大值，并用该值替换参考点的像素值实现膨胀。下图是将左边的原始图像A膨胀处理为右边的效果图A⊕B。



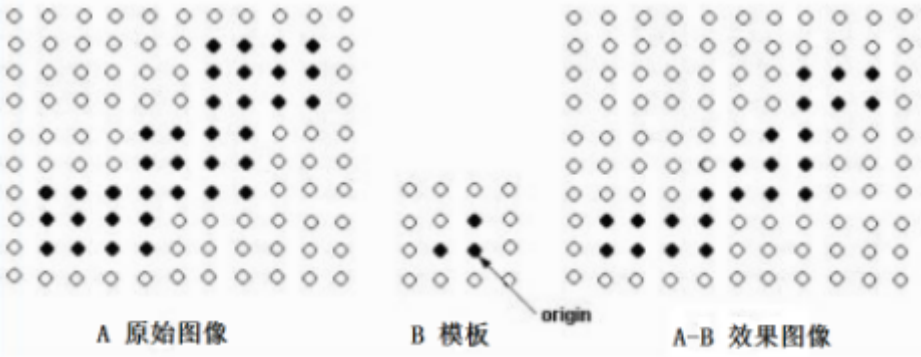

处理结果如下图所示：

**2.图像腐蚀**

腐蚀的运算符是“ - ”，其定义如下：

$$A - B = \{x \mid B_x \subseteq A\}$$

该公式表示图像A用卷积模板B来进行腐蚀处理，通过模板B与图像A进行卷积计算，得出B覆盖区域的像素点最小值，并用这个最小值来替代参考点的像素值。如图所示，将左边的原始图像A腐蚀处理为右边的效果图A-B。




处理结果如下图所示：

原始图像

腐蚀图像



# 二. 图像腐蚀代码实现

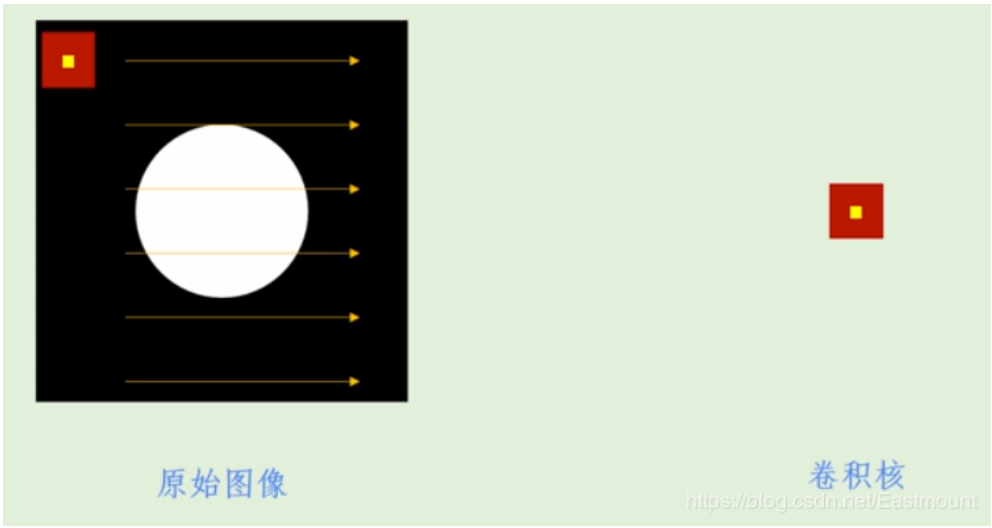
**1.基础理论**

形态学转换主要针对的是二值图像（0或1）。图像腐蚀类似于“领域被蚕食”，将图像中的高亮区域或白色部分进行缩减细化，其运行结果图比原图的高亮区域更小。其主要包括两个输入对象：

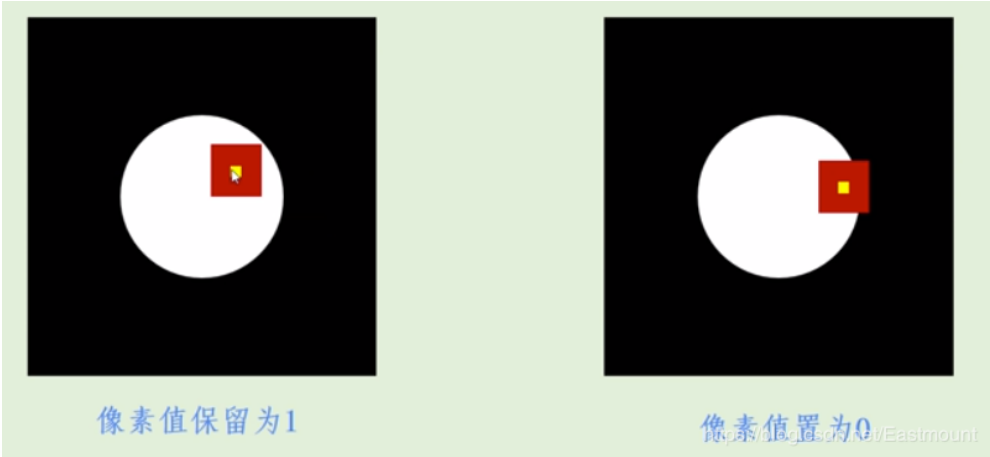
(1)二值图像

(2)卷积核

卷积核是腐蚀中的关键数组，采用numpy库可以生成。卷积核的中心点逐个像素扫描原始图像，如下图所示：



被扫描到的原始图像中的像素点，只有当卷积核对应的元素值均为1时，其值才为1，否则其值修改为0。换句话说，遍历到的黄色点位置，其周围全部是白色，保留白色，否则变为黑色，图像腐蚀变小。

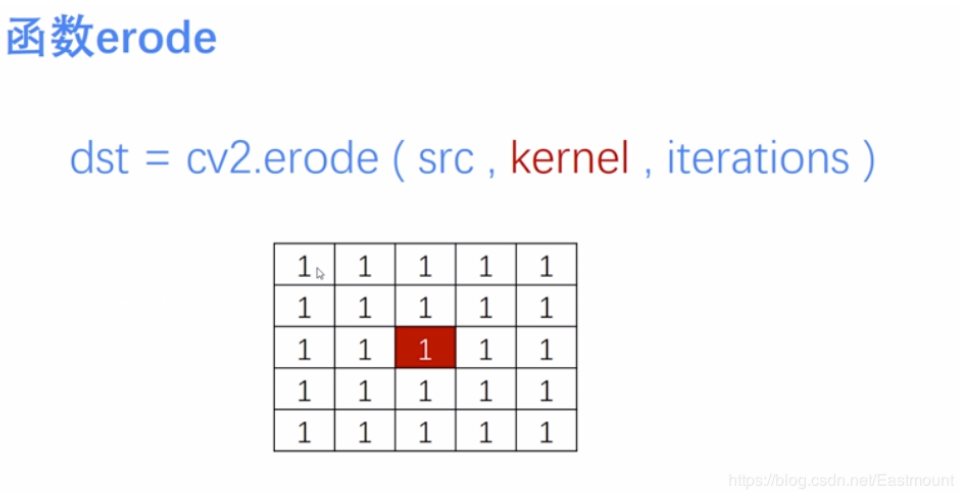



**2.函数原型**

图像腐蚀主要使用的函数为erode，其原型如下：

**dst = cv2.erode(src, kernel, iterations)**

参数dst表示处理的结果，src表示原图像，kernel表示卷积核，iterations表示迭代次数。下图表示5*5的卷积核，可以采用函数 np.ones((5,5), np.uint8) 构建。




注意：迭代次数默认是1，表示进行一次腐蚀，也可以根据需要进行多次迭代，进行多次腐蚀。

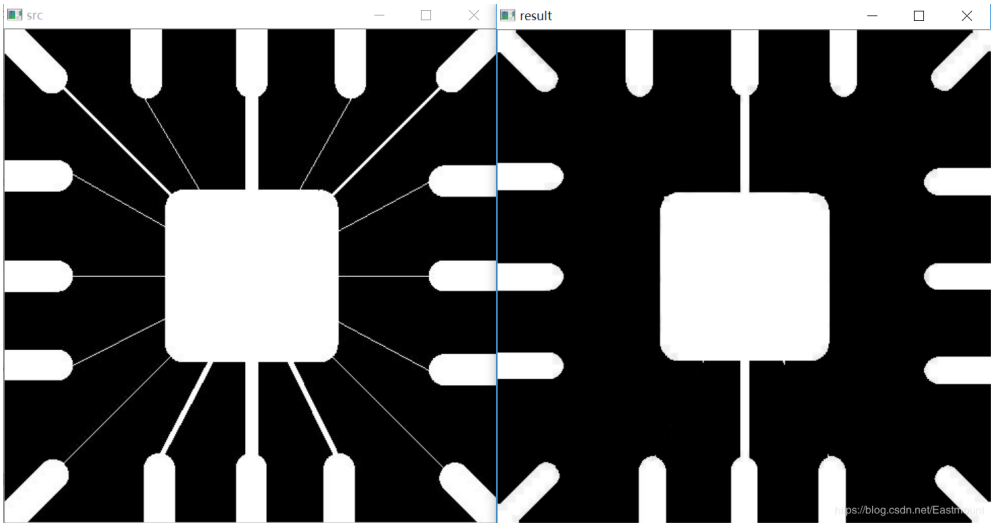
**3.代码实现**

完整代码如下所示：

```
#encoding:utf-8
import cv2
import numpy as np
```

```
#读取图片
src = cv2.imread('test01.jpg', cv2.IMREAD_UNCHANGED)

#设置卷积核
kernel = np.ones((5,5), np.uint8)

#图像腐蚀处理
erosion = cv2.erode(src, kernel)

#显示图像
cv2.imshow("src", src)
cv2.imshow("result", erosion)

#等待显示
cv2.waitKey(0)
cv2.destroyAllWindows()
```

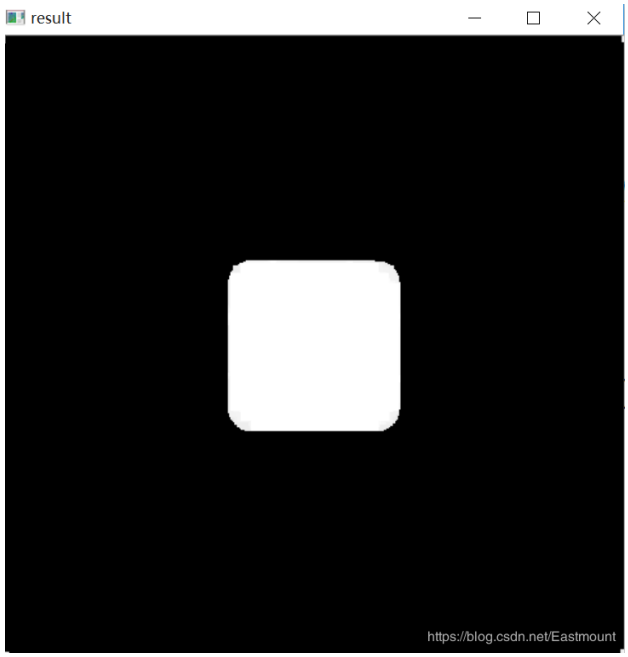
输出结果如下图所示：




由图可见，干扰的细线被进行了清洗，但仍然有些轮廓，此时可设置迭代次数进行腐蚀。

**erosion = cv2.erode(src, kernel,iterations=9)**

输出结果如下图所示：




# 三. 图像膨胀代码实现

**1.基础理论**

图像膨胀是腐蚀操作的逆操作，类似于“领域扩张”，将图像中的高亮区域或白色部分进行扩张，其运行结果图比原图的高亮区域更大，线条变粗了，主要用于去噪。

(1) 图像被腐蚀后，去除了噪声，但是会压缩图像。

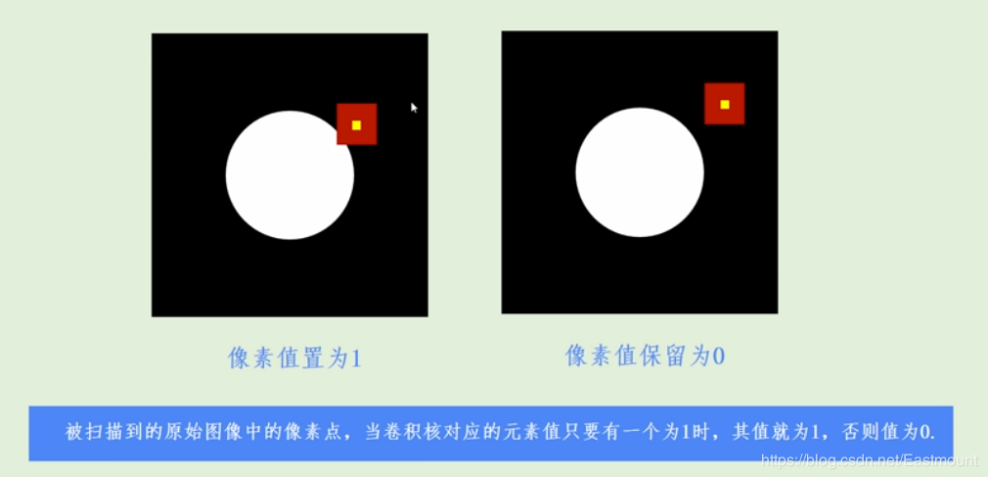
(2) 对腐蚀过的图像，进行膨胀处理，可以去除噪声，并且保持原有形状。


它也包括两个输入对象：

(1)二值图像或原始图像

(2)卷积核

卷积核是腐蚀中的关键数组，采用numpy库可以生成。卷积核的中心点逐个像素扫描原始图像，如下图所示：




被扫描到的原始图像中的像素点，当卷积核对应的元素值**只要有一个为1时，其值就为1，否则为0。**

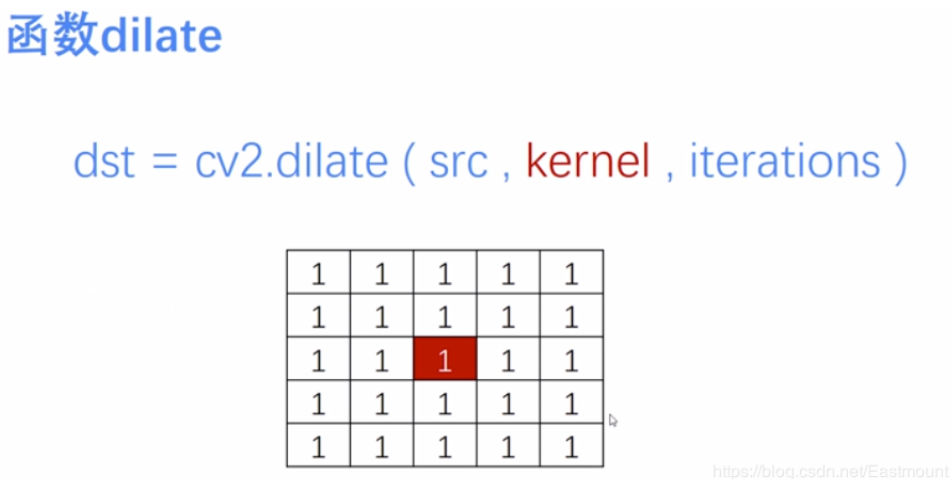
**2.函数原型**

图像膨胀主要使用的函数为dilate，其原型如下：

**dst = cv2.dilate(src, kernel, iterations)**

参数dst表示处理的结果，src表示原图像，kernel表示卷积核，iterations表示迭代次数。

下图表示5*5的卷积核，可以采用函数 np.ones((5,5), np.uint8) 构建。




注意：迭代次数默认是1，表示进行一次膨胀，也可以根据需要进行多次迭代，进行多次膨胀。通常进行1次膨胀即可。

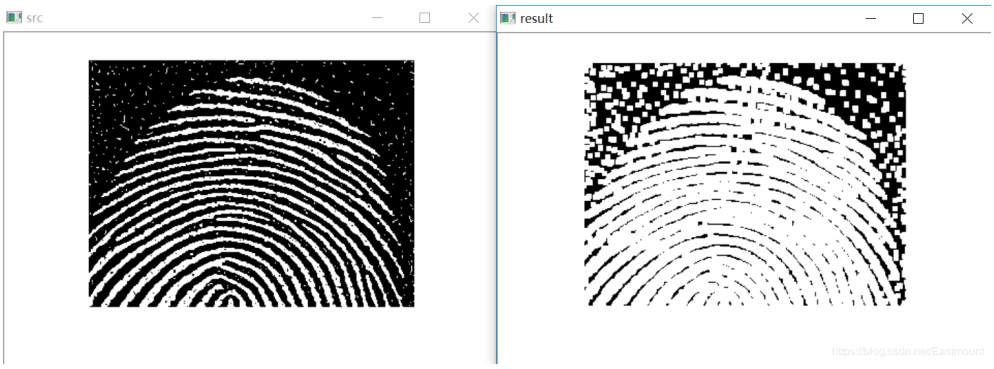
**3.代码实现**

完整代码如下所示：

```
#encoding:utf-8
import cv2
import numpy as np

#读取图片
```

```
src = cv2.imread('test02.png', cv2.IMREAD_UNCHANGED)

#设置卷积核
kernel = np.ones((5,5), np.uint8)

#图像膨胀处理
erosion = cv2.dilate(src, kernel)

#显示图像
cv2.imshow("src", src)
cv2.imshow("result", erosion)

#等待显示
cv2.waitKey(0)
cv2.destroyAllWindows()
```

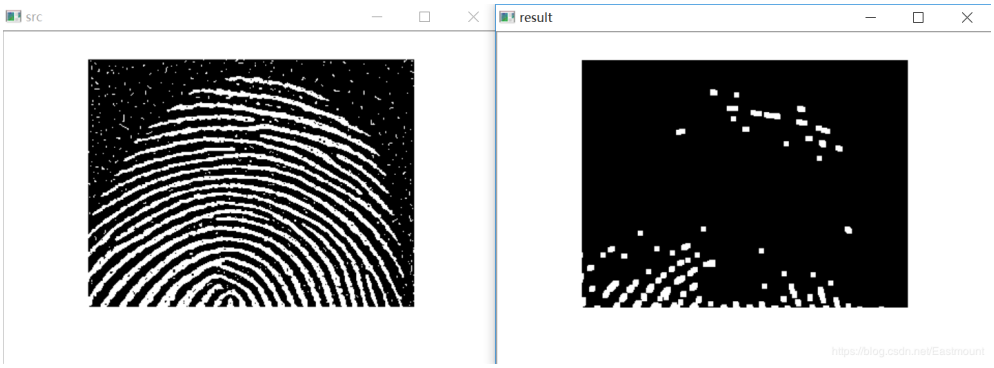
输出结果如下所示：




图像去噪通常需要先腐蚀后膨胀，这又称为开运算，下篇文章将详细介绍。如下图所示：

erosion = cv2.erode(src, kernel)

result = cv2.dilate(erosion, kernel)




希望文章对大家有所帮助，如果有错误或不足之处，还请海涵。最近经历的事情太多，有喜有悲，关闭了朋友圈，希望通过不断学习和写文章来忘记烦劳，将忧郁转换为动

力。哎，总感觉自己在感动这个世界，帮助所有人，而自己却…保重。

（By：Eastmount 2018-10-31 下午4点 https://blog.csdn.net/Eastmount/）