二值形态学腐蚀膨胀实验报告

516030910257 胡雨奇

# 实验内容

实现二值图像针对特定结构元素进行形态学腐蚀膨胀的操作

# 实验环境

Python 3.5 for windows

Windows 10，64位

IDE：python IDLE

# 核心代码和注解

## bi\_dilation(ori\_img,se,center=None)

1. # ori\_img & se : array ( binary img ),0==0,!0==1
2. # center (x,y) or [x,y]
3. # ret\_val: ret\_img ( binary img )
4. **def** bi\_dilation(ori\_img,se,center=None):
5. img\_height = len(ori\_img)
6. img\_width = len(ori\_img[0])
7. se\_height = len(se)
8. se\_width = len(se[0])
9. # init center
10. **if** center==None:
11. center = [int(se\_height/2),int(se\_width/2)]
12. # init ret\_img
13. ret\_height = img\_height+se\_height-1
14. ret\_width = img\_width+se\_width-1
15. ret\_img = [[0]\*ret\_width **for** k **in** range(ret\_height)]
16. # create ret\_img
17. **for** h **in** range(img\_height):
18. **for** w **in** range(img\_width):
19. **if** ori\_img[h][w]!=0:
20. **for** se\_h **in** range(se\_height):
21. **for** se\_w **in** range(se\_width):
22. **if** se[se\_h][se\_w]!=0:
23. ret\_img[h+se\_h][w+se\_w]=255
24. **return** array(ret\_img)

输入：

ori\_img，原始图像矩阵，类型为numpy.array

se，结构元素矩阵，类型为numpy.array

center，二元元组或二元列表，代表结构元素的中心

输出：

矩阵，膨胀后的图像

详解：

Line5-11，提取原始图像和结构元素的长宽值和初始化中心的坐标

Line13-15，初始化输出矩阵

Line17-23，按以下膨胀规则生成输出矩阵

Rule：遍历原始图像的矩阵，将结构元素矩阵的中心叠放在该像素上，如果重叠部分有同为1的像素，则在输出图像对应结构元素的像素置1.

## bi\_erosion(ori\_img,se,center=None)

1. # ori\_img & se : array ( binary img ),0==0,!0==1
2. # center (x,y) or [x,y]
3. # ret\_val: ret\_img ( binary img )
4. **def** bi\_erosion(ori\_img,se,center=None):
5. img\_height = len(ori\_img)
6. img\_width = len(ori\_img[0])
7. se\_height = len(se)
8. se\_width = len(se[0])
9. # init center
10. **if** center==None:
11. center = [int(se\_height/2),int(se\_width/2)]
12. # init ret\_img
13. ret\_height = img\_height-se\_height+1
14. ret\_width = img\_width-se\_width+1
15. ret\_img = [[0]\*ret\_width **for** k **in** range(ret\_height)]
16. # create ret\_img
17. **for** h **in** range(ret\_height):
18. **for** w **in** range(ret\_width):
19. flag = True
20. **for** se\_h **in** range(se\_height):
21. **for** se\_w **in** range(se\_width):
22. th\_h = int(h+se\_h)
23. th\_w = int(w+se\_w)
24. **if** se[se\_h][se\_w]!=0 **and** ori\_img[th\_h][th\_w]==0:
25. flag=False
26. **if** flag:
27. ret\_img[h][w]=255
28. **return** array(ret\_img)

输入：

ori\_img，原始图像矩阵，类型为numpy.array

se，结构元素矩阵，类型为numpy.array

center，二元元组或二元列表，代表结构元素的中心

输出：

矩阵，腐蚀后的图像

详解：

Line5-11，提取原始图像和结构元素的长宽值和初始化中心的坐标

Line13-15，初始化输出矩阵

Line17-27，按以下膨胀规则生成输出矩阵

Rule：遍历原始图像的矩阵，将结构元素矩阵的中心叠放在该像素上，如果重叠部分全部均为1的像素，则在输出图像对应结构元素的中心像素位置置1.

# 实验效果展示

## 原始图像

注释：二值图像

## 结构元素

（2\*2尺寸的正方形se）

## 腐蚀后的图像



注释：效果较为成功

## 膨胀后的图像



注释：由于原始图像在靠近线元素的附近有椒盐噪声（肉眼不可见），导致膨胀后无法识别原始图像

## 开操作之后膨胀



注释：用相同结构元素做开操作，除去了小的椒盐噪声，膨胀后效果较好

## 用于实验的代码

见附件1。

# 附件1：

1. **from** PIL **import** Image
2. **from** numpy **import** \*
4. **import** morphologyoperation as mo
6. **def** convert():
7. img = Image.open("test.jpg")
8. img1 = img.convert("L")
9. img1.save("L.jpg")
10. img2 = array(img1)
12. **for** line **in** img2:
13. **for** i **in** range(len(line)):
14. line[i] = (line[i]<188)\*255
15. **print**(img2)
16. img2 = Image.fromarray(img2)
17. img2.save("1.jpg")

20. **def** dilation\_test():
21. img = Image.open("1.jpg")
22. img = array(img)
23. se = [[1,1],[1,1]]
24. img = mo.bi\_dilation(img,se)
25. img = Image.fromarray(img)
26. img = img.convert("RGB")
27. img.save("dilation.jpeg")

30. **def** erosion\_test():
31. img = Image.open("1.jpg")
32. img = array(img)
33. se = [[1,1],[1,1]]
34. img = mo.bi\_erosion(img,se)
35. img = Image.fromarray(img)
36. img = img.convert("RGB")
37. img.save("erosion.jpeg")
38. **print**(img.size)
40. **def** combine\_test():
41. img = Image.open("1.jpg")
42. img = array(img)
43. se = [[1]\*2]\*2
44. img = mo.bi\_erosion(img,se)
45. img = mo.bi\_dilation(img,se)
46. img = mo.bi\_dilation(img,se)
47. img = Image.fromarray(img)
48. img = img.convert("RGB")
49. img.save("combind.jpg")

52. combine\_test()
53. ##dilation\_test()
54. ##erosion\_test()