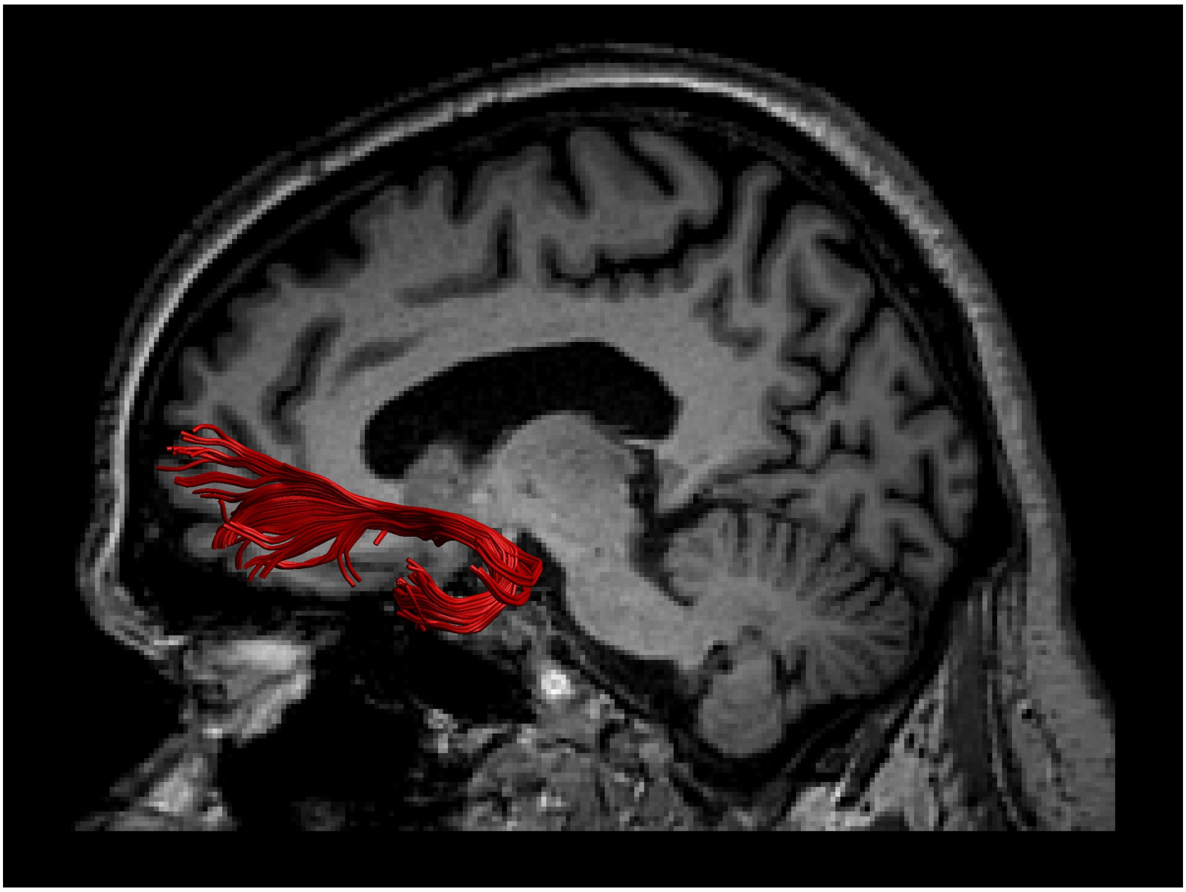
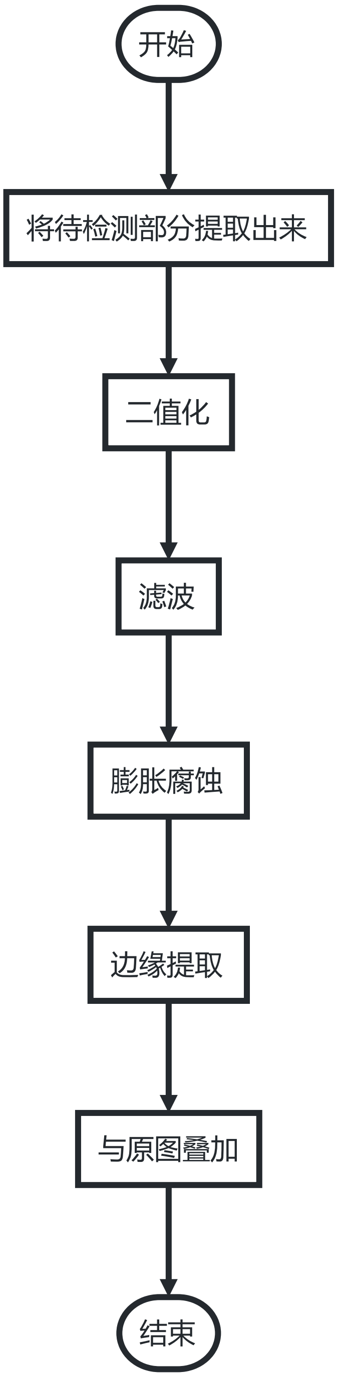
图像处理课程设计

题目一

将一张大脑核磁图像当中的空腔边缘提取出来



- 解题思路

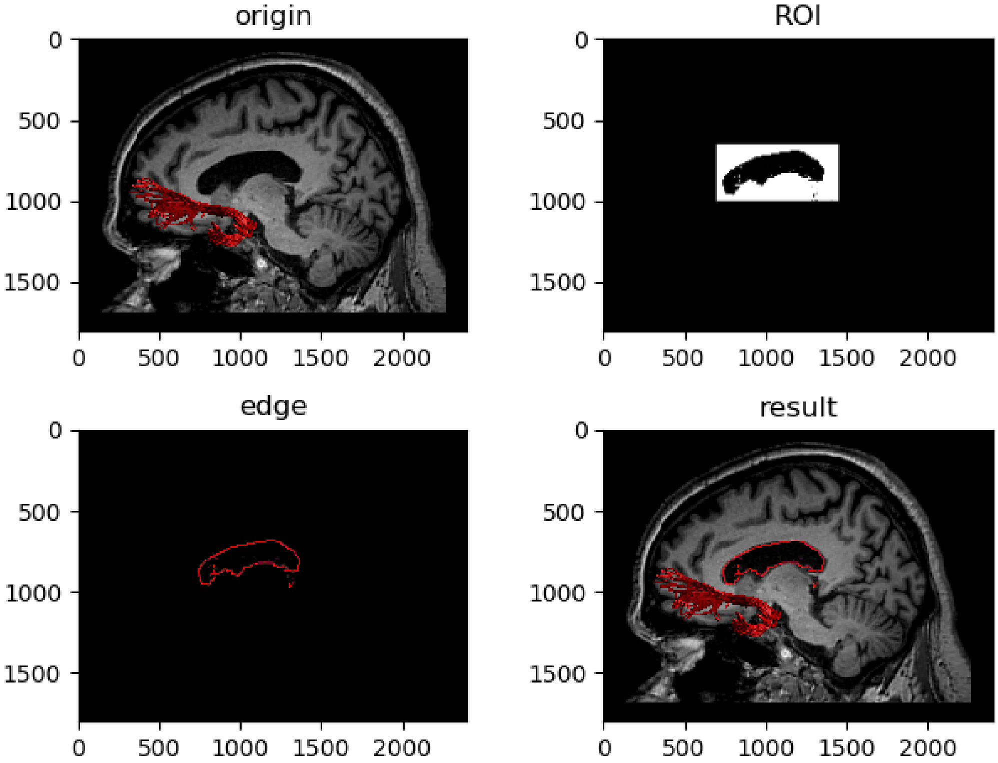


- 代码

```python
import cv2
import numpy as np
from pylab import *

# 读入图像
img_path='brain.jpg'
img = cv2.imread(img_path)

# 转化成灰度图像
img_gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)

# 获取横纵分辨率
row, col = img.shape[:2]

# 定义截取窗口顶点
bottom_left = [700, 1000]
top_left = [700, 650]
top_right = [1450, 650]
bottom_right = [1450, 1000]
# 获取ROI兴趣区域
vertices = np.array([bottom_left, top_left, top_right, bottom_right], dtype=np.int32)
```

```
roi_mask = np.zeros((row, col), dtype=np.uint8)
cv2.fillPoly(roi_mask, [vertices], 255)

# 与原图相与,获取ROI区域的图像
img_roi = cv2.bitwise_and(img_gray, img_gray, mask=roi_mask)
img_copy = img_roi

# 二值化图像
cv2.threshold(img_copy,14,255,0,img_copy)
# 均值平滑滤波
cv2.blur(img_copy,(10,10),img_copy)
# 各做4次膨胀和腐蚀
img_copy = cv2.erode(img_copy,None,iterations=4)
img_copy = cv2.dilate(img_copy,None,iterations=4)

# 使用Sobel算子获得图像边缘
sobelX = cv2.Sobel(img_copy,cv2.CV_64F,1,0)#x方向的梯度
sobelY = cv2.Sobel(img_copy,cv2.CV_64F,0,1)#y方向的梯度

sobelX = np.uint8(np.absolute(sobelX))#x方向梯度的绝对值
sobelY = np.uint8(np.absolute(sobelY))#y方向梯度的绝对值

img_copy = cv2.bitwise_or(sobelX,sobelY)#

# 去除ROI区域多余的矩形边缘
delta = 10

bottom_left = [700+delta, 1000-delta]
top_left = [700+delta, 650+delta]
top_right = [1450-delta, 650+delta]
bottom_right = [1450-delta, 1000-delta]

vertices = np.array([bottom_left, top_left, top_right, bottom_right], dtype=np.int32)
roi_mask = np.zeros((row, col), dtype=np.uint8)
cv2.fillPoly(roi_mask, [vertices], 255)

img_edge = cv2.bitwise_and(img_copy, img_copy, mask=roi_mask)
img_edge = cv2.cvtColor(img_edge,cv2.COLOR_GRAY2RGB)
red = cv2.imread('red.png')
img_edge = cv2.bitwise_and(img_edge,red)
# 将检测出的边缘与原图叠加
img_copy = cv2.bitwise_or(img,img_edge)

# 显示结果
fig = plt.figure()
subplot(221)
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
imshow(img)
title('origin')

subplot(222)
img_roi = cv2.cvtColor(img_roi, cv2.COLOR_GRAY2RGB)
imshow(img_roi)
title('ROI')

subplot(223)
img_edge = cv2.cvtColor(img_edge, cv2.COLOR_BGR2RGB)
imshow(img_edge)
title('edge')

subplot(224)
img_copy = cv2.cvtColor(img_copy, cv2.COLOR_BGR2RGB)
imshow(img_copy)
title('result')

show()
```

- 处理结果


origin
ROI
edge
result
0
500
1000
1500
0
500
1000
1500
2000