图像分割

图像分割（Segmentation）指的是将数字图像细分为多个图像子区域（像素的集合）的过程。图像分割的目的是简化或改变图像的表示形式，使得图像更容易理解和分析。对于医学图像，分割区域通常对应特定的解剖结构，如大脑、心脏和病灶等。医学图像分割也可以看作是从图像中提取出感兴趣的解剖结构。

医学图像分割在临床上常用于对骨骼的分割、对血管的分割和对内脏的分割等等。常用的方法包括Region Growing, Watershed, Level Set和Fuzzy Connectedness等方法。

使用Region Growing进行医学图像分割的代码如下：

def itkRegionGrow(im\_arr, type\_str, seedlist, lower=0, upper=1):

func\_1 = 'CT'

func\_2 = 'CC'

# get an image from the array input

image = itkEdgePreservedSmoothing.itkEdgePreservedSmoothing(im\_arr, 'CF')

image = sitk.GetImageFromArray(im\_arr)

image = sitk.Cast(image, sitk.sitkFloat32)

# find out the way to process the image according to type\_str

if type\_str == func\_1:

im\_new = sitk.ConnectedThreshold(image, seedlist, lower, upper)

elif type\_str == func\_2:

im\_new = sitk.ConfidenceConnected(image, seedlist)

else:

print('Please check your spelling,'

'and try again.')

return im\_new

这其中，func\_1和func\_2分别为Region Growing的两种类型ConnectedThreshold和ConfidenceConnected，可以选用适当的类型进行图像分割。

使用Fuzzy Connectedness进行图像分割的代码如下：

def itkVectorFuzzyConnectednessImageFilter(path, seedList, iterations=4, multiplier=4.5, radius=1, replaceValue=1):

img = sitk.GetArrayFromImage(sitk.ReadImage(path))

result = sitk.VectorConfidenceConnected(img, seedList, iterations, multiplier, radius, replaceValue)

return result

使用Level Set可以对3维医学图像（比如CT图）进行图像分割，代码如下：

def itkImageLevelSetSegmentation(Im\_arr, type\_str, seedlist=0):

# define the six way

func\_1 = 'SD'

func\_2 = 'GAC'

func\_3 = 'TH'

func\_4 = 'FA'

func\_5 = 'FAB'

func\_6 = 'LP'

# get an image from the array input

image = sitk.GetImageFromArray(Im\_arr)

# find out the way to process the image according to type\_str

# smooth the image

if type\_str == func\_1:

image = sitk.Cast(image,sitk.sitkFloat32)

image\_pre=sitk.CannyEdgeDetection(image)

im\_new = sitk.ShapeDetectionLevelSet(image,image\_pre)

elif type\_str == func\_2:

seg = itkRegionGrow.itkRegionGrow(Im\_arr, 'CC', seedlist)

init\_ls = sitk.SignedMaurerDistanceMap(seg, insideIsPositive=True, useImageSpacing=True)

im\_new = sitk.GeodesicActiveContourLevelSet(init\_ls, sitk.Cast(image, sitk.sitkFloat32))

elif type\_str == func\_3:

seg = itkRegionGrow.itkRegionGrow(Im\_arr, 'CC', seedlist)

stats = sitk.LabelStatisticsImageFilter()

stats.Execute(image, seg)

lower\_threshold = stats.GetMean(1)-1.5\*stats.GetSigma(1)

upper\_threshold = stats.GetMean(1)+1.5\*stats.GetSigma(1)

init\_ls = sitk.SignedMaurerDistanceMap(seg, insideIsPositive=True, useImageSpacing=True)

im\_new = sitk.ThresholdSegmentationLevelSet(init\_ls,sitk.Cast(image,sitk.sitkFloat32), lower\_threshold, upper\_threshold)

elif type\_str == func\_4:

im\_new = sitk.FastMarching(image, seedlist)

elif type\_str == func\_5:

im\_new = sitk.FastMarchingBase(image, seedlist)

elif type\_str == func\_6:

seg = itkRegionGrow.itkRegionGrow(Im\_arr, 'CC', seedlist)

init\_ls = sitk.SignedMaurerDistanceMap(seg, insideIsPositive=True, useImageSpacing=True)

im\_new = sitk.LaplacianSegmentationLevelSet(init\_ls, sitk.Cast(image, sitk.sitkFloat32))

else:

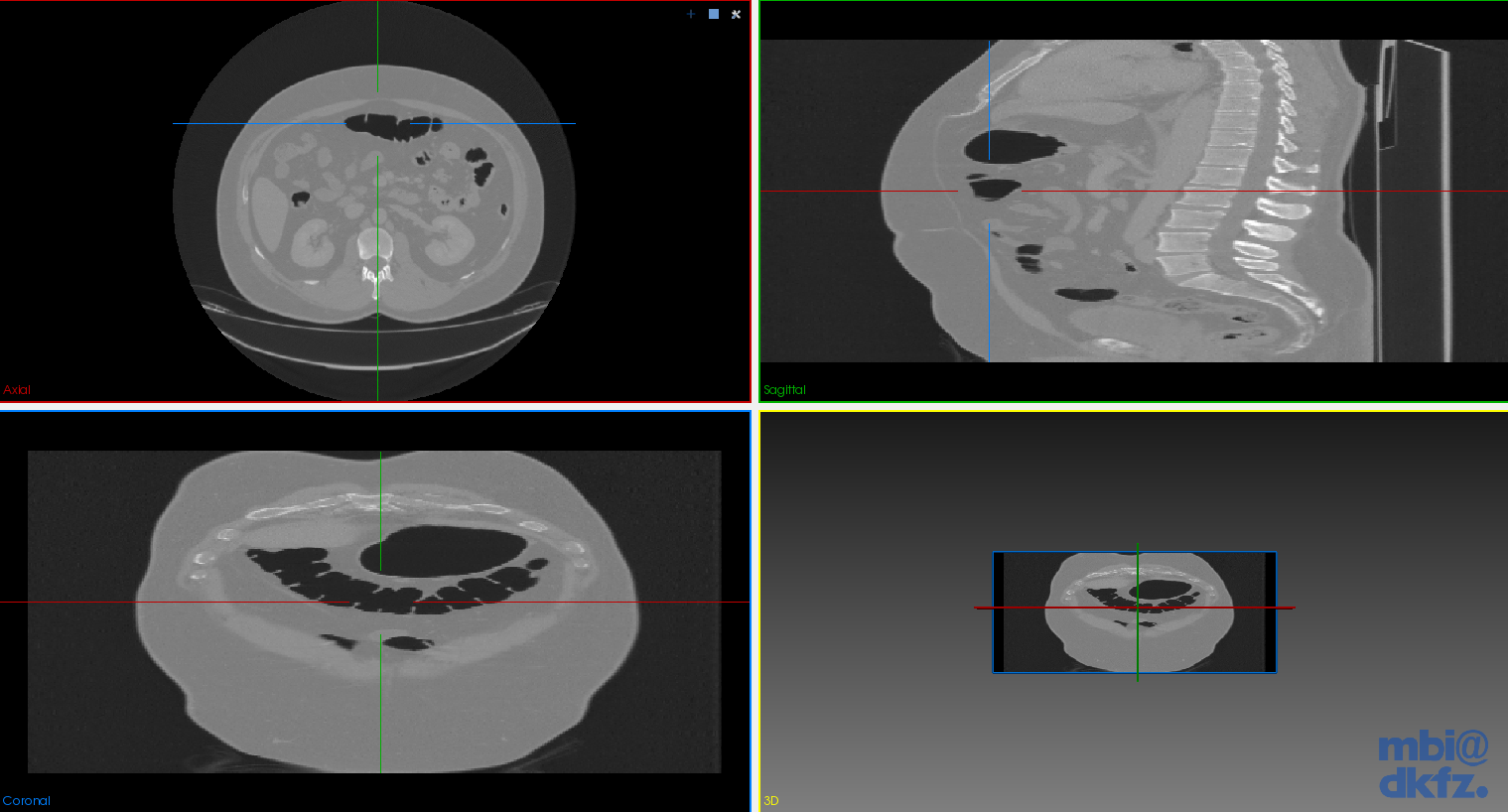
print('Please check your spelling,'

'and try again.')

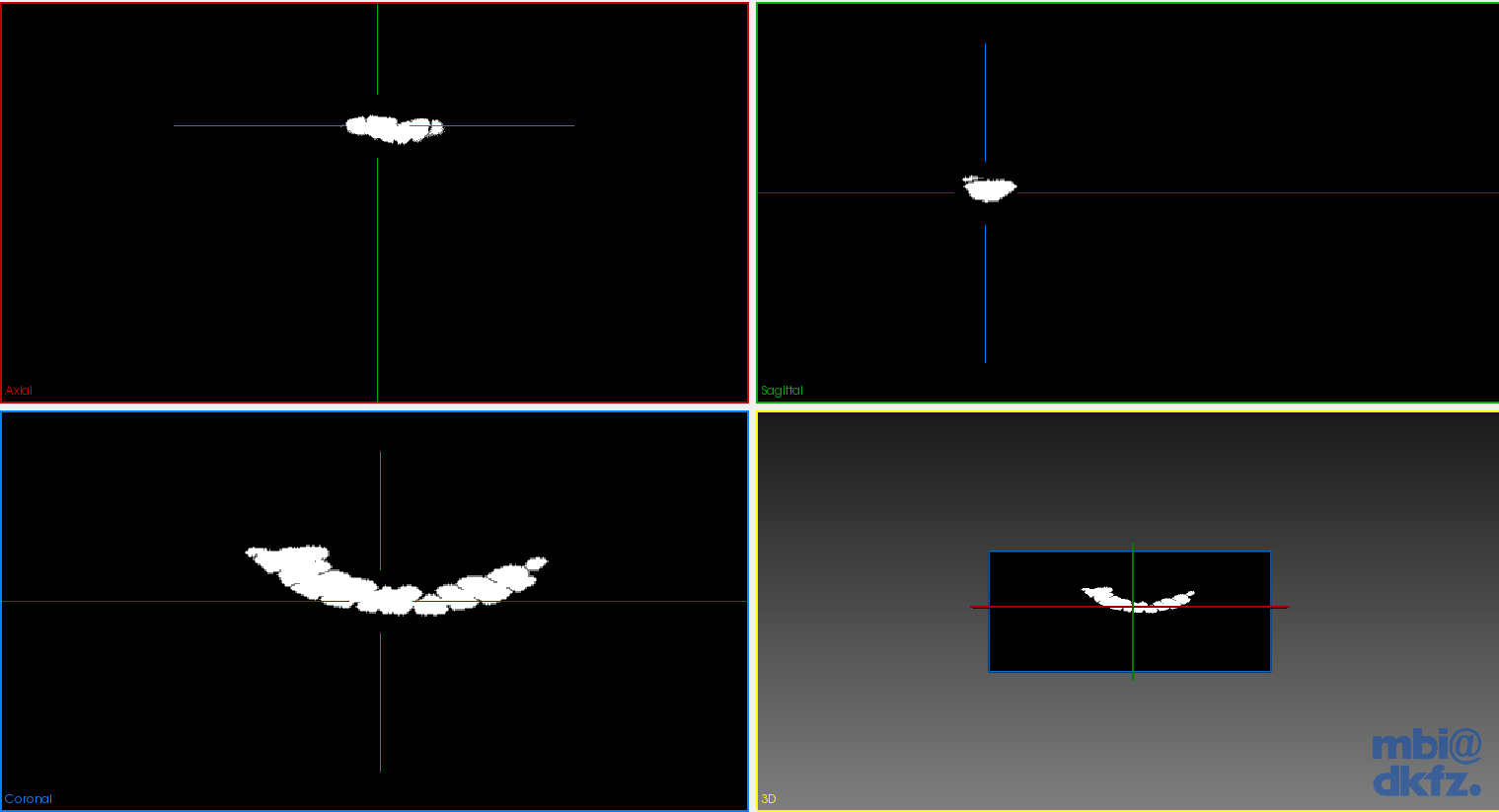
return im\_new

定义了六种类型的Level Set。和前几种方法一样，输入为含图像信息的矩阵，输出也是包含图像信息的矩阵。

以下为原图：



以下为图像经过分割后的图：



曲面处理

面绘制技术是利用计算机模拟光线照射到网格曲面上时，在屏幕上所投影得到的平面图像。现代计算机可以实现实时的面绘制计算，即便当曲面发生旋转、平移、变形等运动时也可以实时地反应在屏幕上。

简单的面绘制程序如下：

import vtk

filename = "Liver.stl"

reader = vtk.vtkSTLReader()

reader.SetFileName(filename)

mapper = vtk.vtkPolyDataMapper()

mapper.SetInputConnection(reader.GetOutputPort())

actor = vtk.vtkActor()

actor.SetMapper(mapper)

# Create a rendering window and renderer

ren = vtk.vtkRenderer()

renWin = vtk.vtkRenderWindow()

renWin.AddRenderer(ren)

# Create a renderwindowinteractor

iren = vtk.vtkRenderWindowInteractor()

iren.SetRenderWindow(renWin)

# Assign actor to the renderer

ren.AddActor(actor)

# Enable user interface interactor

iren.Initialize()

renWin.Render()

iren.Start()

曲面处理的种类非常多，包括曲面平滑、网格简化、网格重构、曲面逻辑运算、细分曲面等等。

细分曲面是最简单的操作，代码如下：

import pymesh

# get the path of the original mesh and set the path of the Subdivided mesh

filepath="./Spleen\_3D-interpolation.stl"

filenew="./Spleen\_3D-interpolation\_new.stl"

#get the original mesh

mesh=pymesh.load\_mesh(filepath)

#subdivide the mesh

mesh\_new=pymesh.subdivide(mesh,order=3,method='simple')

#write the subdivided mesh as a new file

pymesh.save\_mesh(filenew,mesh\_new)

效果如下：

