目录

[第一部分：第二章图像旋转 2](#_Toc50150810)

[最邻近插值方法 2](#_Toc50150811)

[双线性插值法 3](#_Toc50150812)

[第二部分：血管分割 5](#_Toc50150813)

[方法一：RCOtsu算法 5](#_Toc50150814)

[方法二：基于K-Means实现图像分割 8](#_Toc50150815)

[方法三：基于边缘检测的分割方法 10](#_Toc50150816)

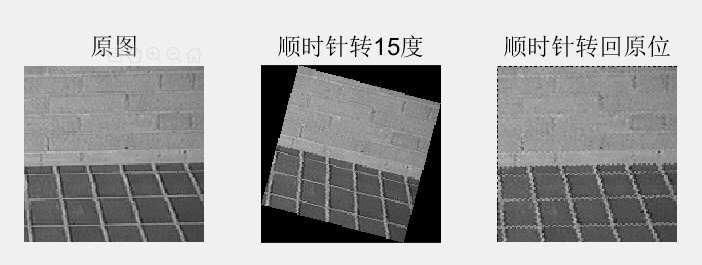
[第三部分：总结 13](#_Toc50150817)

[总结本课程学会的东西 13](#_Toc50150818)

[对课程的改进意见 14](#_Toc50150819)

# 第一部分：第二章图像旋转

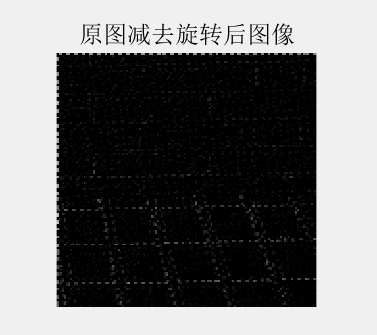
## 最邻近插值方法



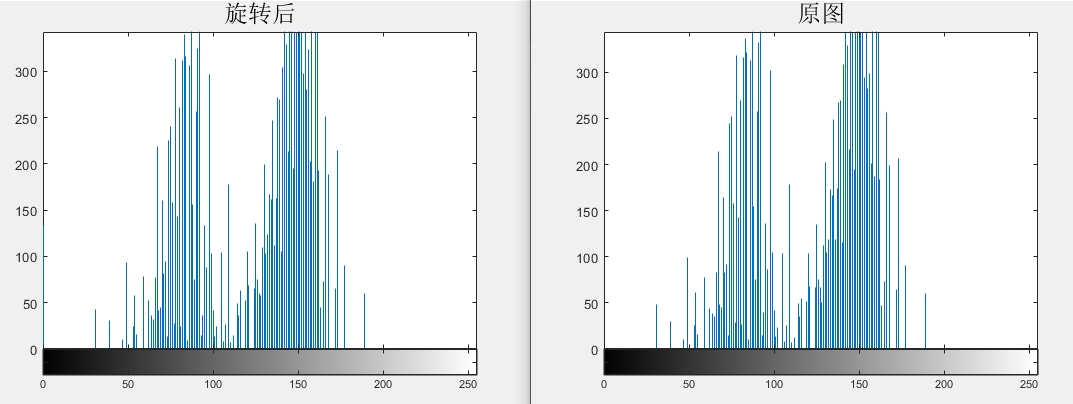
能够看到使用最邻近方法插值来旋转图像，图像的锯齿边较为明显，‘顺时针转回原位’的图像边缘处上方与左方有明显的黑块，可能的原因是在进行裁剪与原图对齐时，由于图像的锯齿明显，因此在旋转时混入了黑色的背景色块，因此在裁剪之后留下了一部分。

从总体上看，对‘原图’以及‘顺时针转回原位’求灰度值的平均值，得到‘原图’的平均值为123.6，‘顺时针转回原位’为122.8，图像矩阵大小为132\*135=17820，即经过最邻近插值法旋转后丢失了灰度。

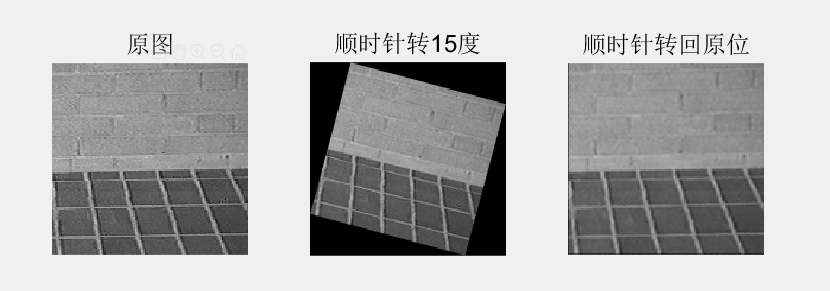
下图为‘原图’减去‘顺时针转回原位’的图像，能够看到有灰色的矩形残留，可能的原因是由于最邻近方法的插值精度不高而导致的灰度锯齿覆盖了原有的灰度像素。



从灰度分布直方图上来看，二者并无显著差别。



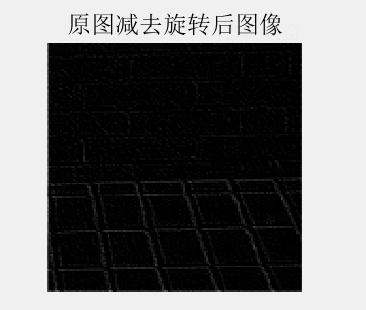
## 双线性插值法



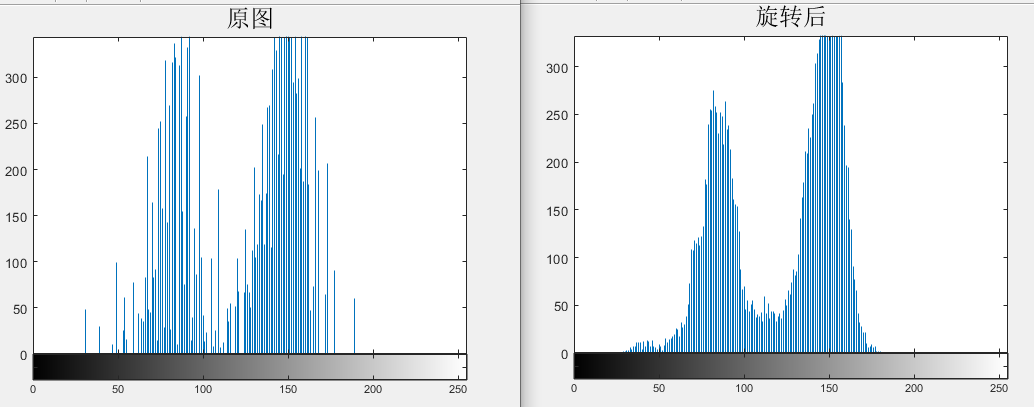
能够看到使用双线性插值方法来旋转图像，旋转后的效果较最邻近算法好。

从总体上看，对‘原图’以及‘顺时针转回原位’求灰度值的平均值，得到‘原图’的平均值为123.6，‘顺时针转回原位’为122.6，图像矩阵大小为132\*135=17820，即经过双线性插值法旋转后丢失了灰度。

下图为‘原图’减去‘顺时针转回原位’的图像，能够看到依然存在灰色的矩形残留，但与最邻近方法不同的是，下面的灰色残留较为平滑，而最邻近方法中的灰色矩形残留依然具有明显的锯齿状。



从灰度分布直方图上来看，旋转厚度灰度分布较原图更加连续。



# 第二部分：血管分割

## 方法一：RCOtsu算法

RCOtsu算法通过将灰度阈值rk限制在之内，在值域内遍历，使得类内差异

最小，所得rk即为所求的阈值。

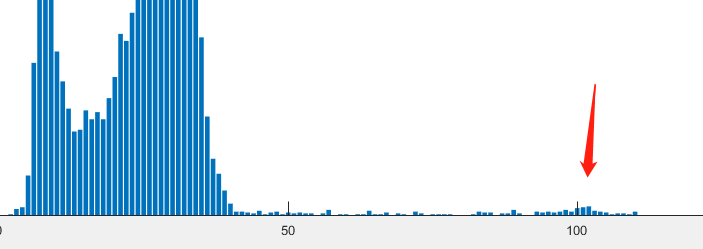
在这里使用视觉判断来近似地估计参数。

阈值范围计算方法：

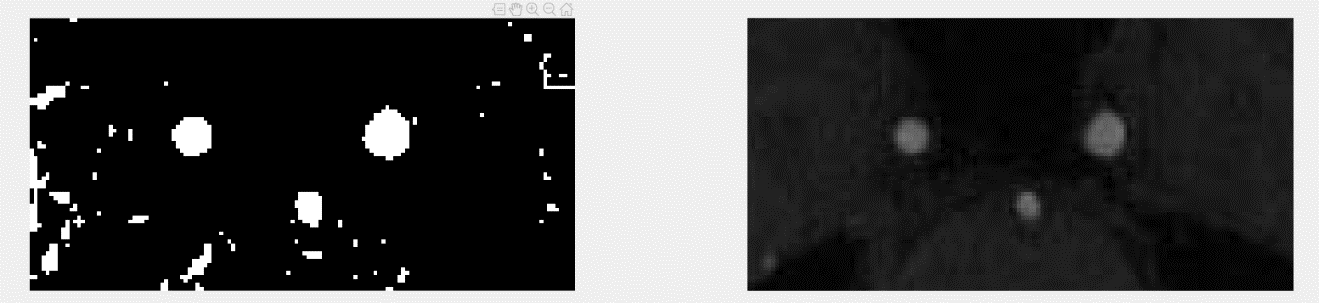




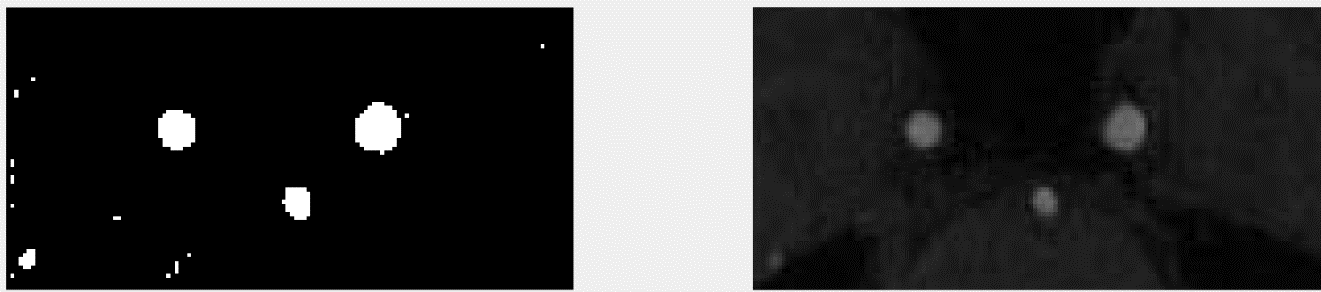
其中，H(i)为灰度为i时的累加频率，Hlb和Hhb通过面积估计来得到。其中，H(i)为灰度为i时的累加频率，Hlb和Hhb通过面积估计来得到。

通过图像工具可以得到高亮点的灰度值大概在100左右，而根据灰度分布直方图能够看到100左右的灰度频数仅仅占很小一部分，因此初始时把背景所占范围设置成0.90~0.95。

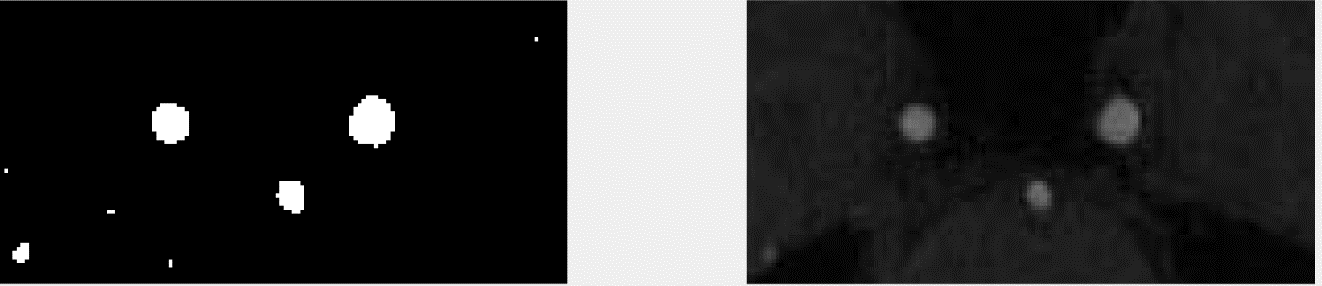
能够看到分割效果比Otsu算法好了很多，但仍有部分噪点，猜想可能是因为阈值范围设置略宽，导致一些相似灰度的像素也被划分为目标物体，因此继续调整阈值范围



当范围设置为0.965~0.985时，分割结果变好了，白色的噪点消失了很多，继续缩小阈值范围



范围设置为0.972~0.985，可以看到白色噪点减少了，已经接近了我们想要的分割效果。



如果需要得到更加完美的分割结果，需要对血管面积进行更精确的估计来得到背景所占的范围，从而缩小rk的范围。

##### 使用dice系数来平均分割结果

下面为dice函数

function [d] = dice(label,pred)

label = logical(label);

X\_Y = 2\*label.\*pred;

XaddY= label+pred;

d = sumall(X\_Y)/sumall(XaddY) ;

end

d = dice(ref,pred)

得到当前的dice系数为 0.9278

## 方法二：基于K-Means实现图像分割

基本思想：

以空间中的k个点为中心进行聚类，对最靠近它们的对象归类，类别数为k。不断迭代，逐次更新各聚类中心的值，直至得到最好的聚类结果。

主要函数代码：

function [mu,mask]=kmeans(ima,k)

%功能·：运用K-means算法对图像进行分割

% 输入 ima-输入的灰度图像 K-分类数

%输出 mu -均值类向量 mask-分类后的图像

ima=double(ima);

copy=ima;

ima=ima(:); % 将矩阵拉平

mi=min(ima);%找到图像灰度最小值

ima=ima-mi+1;

s=length(ima);%有多少灰度级

%计算图像灰度直方图

m=max(ima)+1;%图像最大灰度值

h=zeros(1,m);

hc=zeros(1,m);

for i=1:s

if (ima(i)>0)

h(ima(i))=h(ima(i))+1;%灰度值i累加

end

end

ind =find(h);

h1=length(ind);

%初始化质心

mu=(1:k)\*m/(k+1);

%start process

while(true)

oldmu=mu;

%现有的分类·

for i=1:h1

c=abs(ind(i)-mu);

cc=find(c==min(c));

hc(ind(i))=cc(1);

end

%重新计算均值

for i=1:k

a=find(hc==i);

mu(i)=sum(a.\*h(a))/sum(h(a));

end

if(mu==oldmu)

break;

end

end

%计算生成分类后的图像

s=size(copy);

mask=zeros(s);

for i=1:s(1)

for j=1:s(2)

c=abs(copy(i,j)-mu);

a=find(c==min(c));

mask(i,j)=a(1);

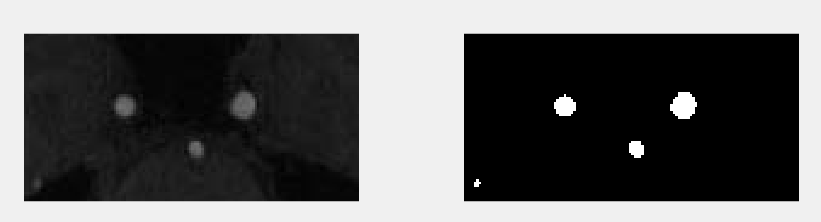
end

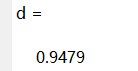
end

mu=mu+mi-1;mask = mask - 1; % 调整为 二值图

end

分割结果



使用方法一中的dice函数评估，得到当前的dice系数为 

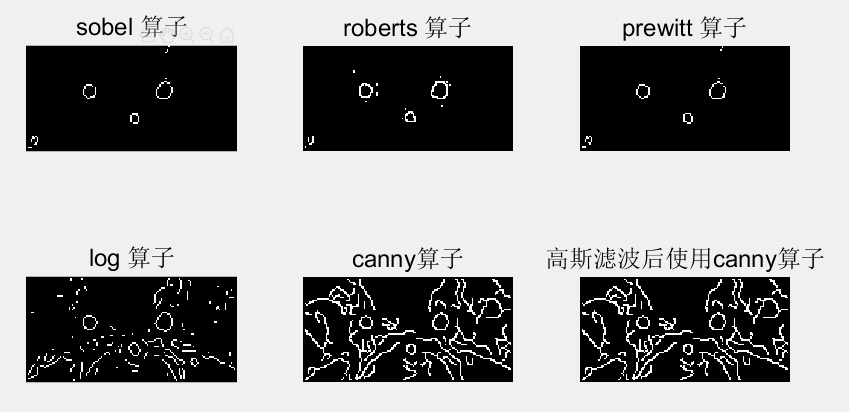
## 方法三：基于边缘检测的分割方法

基本思想：

先确定图像中的边缘像素，然后再把这些像素连接在一起就构成所需的区域边界。边缘检测可以利用微分算子来确定。

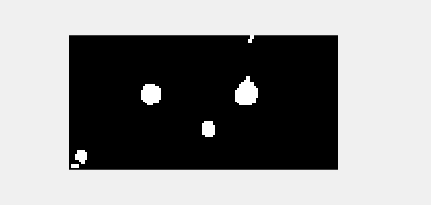
在这里，使用了Sobel算子、Roberts算子、Prewitt算子、Log算子、Canny算子、高斯滤波后再使用Canny算子进行边缘检测。

通过下图来对比各种算子的分割结果

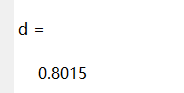


由图可见，sobel、roberts、prewitt算子的效果较好，这里使用prewitt算子进行边缘检测。

经过边缘检测后，需要对边缘内部进行填充，由于边缘不一定是连续的，因此需要对二值图像进行一次膨胀操作使，再使用matlab自带的imfill函数进行填充。下图为运行代码后的分割结果:



使用dice函数评估分割结果，得到



此时d值比前两种方法低一点，猜测是膨胀操作致使血管面积更大了，而且部分微小面积的点也被扩大了，这个时候降低了分割的效果。

代码如下：

I = imread('MRA.pgm');

ref = imread('MRA\_Reference.pgm');

img= I;

[m,n]=size(img);

BW1=edge(img,'sobel'); %用Sobel算子进行边缘检测

BW2=edge(img,'roberts');%用Roberts算子进行边缘检测

BW3=edge(img,'prewitt'); %用Prewitt算子进行边缘检测

BW4=edge(img,'log'); %用Log算子进行边缘检测

BW5=edge(img,'canny'); %用Canny算子进行边缘检测

h=fspecial('gaussian',5);%?高斯滤波

BW6=edge(img,'canny');%高斯滤波后使用Canny算子进行边缘检测

subplot(2,3,1), imshow(BW1);

title('sobel 算子','fontSize',18);

subplot(2,3,2), imshow(BW2);

title('roberts 算子','fontSize',18);

subplot(2,3,3), imshow(BW3);

title('prewitt 算子','fontSize',18);

subplot(2,3,4), imshow(BW4);

title('log 算子','fontSize',18);

subplot(2,3,5), imshow(BW5);

title('canny算子','fontSize',18);

subplot(2,3,6), imshow(BW6);

title('高斯滤波后使用canny算子','fontSize',18);

filter = ones(2,2);

BW3=imdilate(BW3,filter);

BW3=imfill(BW3,'holes');

figure();

imshow(BW3);

d= dice(ref,BW3)

% dice函数

function [d] = dice(label,pred)

label = logical(label);

X\_Y = 2\*label.\*pred;

XaddY= label+pred;

d = sumall(X\_Y)/sumall(XaddY) ;

end

% 矩阵所有元素相加

function [res] =sumall(x)

res= sum(sum(x));

end

# 第三部分：总结

## 总结本课程学会的东西

首先非常感谢胡老师为我们讲述这门课程，我在本课程中学习到的东西包括但不限于：

1. 学习了关于数字图像基础的知识，例如空间、灰度、像素之间的一些基本关系、图像的插值算法、数字图像处理的数学工具等。
2. 学习了关于灰度的详细知识，对灰度的直方图处理，空间滤波器、滤波算子等。
3. 学习了形态学图像处理相关的知识，腐蚀、膨胀、开操作、闭操作，以及这些方法的基本应用。
4. 学习了图像分割中的传统处理方法，如边缘检测法，分水岭算法、otsu的阈值算法、基于区域的分割算法等。
5. 了解了医学图像处理领域的目前现状。
6. 通过作业的形式，实际动手实践了所学的内容，加深了对知识的理解。

## 对课程的改进意见

以下意见仅从个人角度出发，仅针对于课程、教学相关，本人愚见如下：

1. 对于相应的知识点，希望能有多方面的例子来帮助理解。
2. 对于部分重要的知识点，希望可以更详细地讲述。而对于类似综述性质的知识点可以让同学们课下自行了解。
3. 在布置作业时，尽量使用ppt文字的形式来布置，尽量避免课上口头布置，因为口头布置容易产生歧义，这样降低了沟通效率，增加了沟通成本。
4. 对于各章作业，希望能够有一个解析或者详细的思路来帮助同学们自行校对，起到一个改错的过程，这样可以更充分地吸收知识点。