机器视觉技术第三次作业

1913416 阎丝雨

一、 算法简介与代码实现

（一）迭代选择最优阈值并用阈值化方法实现图像的二值化

主要思想：图像分割后的两部分 A 和 B 的均值和基本保持稳定。也就是说，随着迭代的进行，取 [mean(A)+mean(B)]/2 最终的收敛值作为分割阈值。

为了提高收敛速度，本图将初始阈值 T0 设置为最大灰度值和最小灰度值的中间值。

- **输入图像并进行灰度化处理。**

```
I=imread('img.jpg');
figure(1);
imshow(I);
title('原图')
%图像灰度化处理
A = rgb2gray(I);
figure(2);
imshow(A);
A_max=max(A);
A_min=min(A);
title('灰度图')
```

- **设初始阈值，进行迭代得到最优阈值并用阈值化方法实现图像的二值化。**

```
T =(A_max+A_min)/2 ;    %取均值作为初始阈值
done = false;    %定义跳出循环的量
i = 0;

% while循环进行迭代
while ~done
    r1 = find(A<=T);  %小于阈值的部分
    r2 = find(A>T);   %大于阈值的部分
    Tnew = (mean(A(r1)) + mean(A(r2))) / 2;  %计算分割后的阈值均值的均值
    done = abs(Tnew - T) < 0.01;      %判断迭代是否收敛
    T = Tnew;       %如不收敛,则将分割后的均值的均值作为新的阈值进行循环计算
    i = i+1;
end
A(r1) = 0;    %将小于阈值的部分赋值为0
A(r2) = 1;    %将大于阈值的部分赋值为1    这两步是将图像转换成二值图像
```

（二）去除杂质-高斯滤波

高斯滤波是一种线性平滑滤波器，对于服从正态分布的噪声有很好的抑制作用。在此次处理中，选择高斯滤波进行图像预处理。

- **此段代码生成了五阶的标准差为 var 的高斯掩膜，对二值图像进行滤波操作，为后面的处理方面，又一次进行了简单的阈值化处理。**

```
%高斯滤波
var=0.5;
W = fspecial('gaussian',[5,5],var);
img= imfilter(A, W, 'replicate');
[m,n]=size(img);
for i=1:1:m*n
    if img(i)<0.9
       img(i)=0;
    else
       img(i)=1;
    end
end
```

（二）内边界跟踪算法

边界跟踪是基于边缘的分割常用方法之一，用于区域已分出（二值或已标注），但边界未知的情况。分为内边界与外边界。内边界为区域的一个子集。

内边界跟踪算法的具体内容如下（选取八邻域）：

1、从左上方开始搜索图像，直到找到一个新的区域的一个像素P_0，P_0是区域边界的起始像素

2、定义变量 dir，存储从前一个边界元素到当前边界元素的前一个移动方向，置 dir=7

3、按照**逆时针方向**搜索当前像素的3×3邻域，从以下的方向开始搜索邻域：

$$(\mathrm{dir}+7)\bmod 8\ (\mathrm{dir}\ 为偶数)$$

$$(\mathrm{dir}+6)\bmod 8\ (\mathrm{dir}\ 为奇数)$$

找到的第一个与当前像素值相同的像素是一个新的边界元素P_n，更新 dir 的值

4、如果当前的边界元素P_n等于第二个边界元素P_1，且前一个边界元素P_{n-1}等于P_0就停止；否则，重复第三步

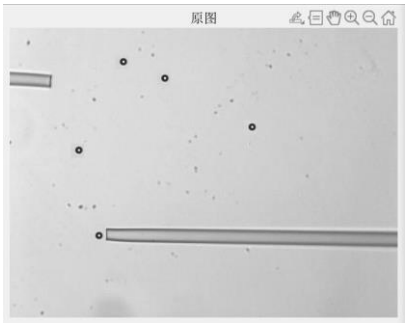
5、检测到的内边界由像素$P_0, \ldots P_{n-2}$构成

➢ 实现代码如下。

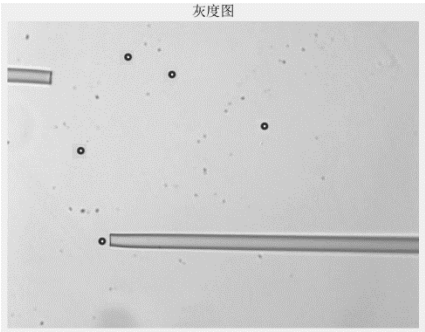
```
%内边界处理
imgn=zeros(m,n);          %边界标记图像
ed=[-1 -1;0 -1;1 -1;1 0;1 1;0 1;-1 1;-1 0]; %从左上角像素，逆时针搜索
for i=2:m-1
    for j=2:n-1
        if img(i,j)==1 && imgn(i,j)==0       %当前是没标记的白色像素
            if sum(sum(img(i-1:i+1,j-1:j+1)))~=9     %块内部的白像素不标记
                ii=i;          %像素块内部搜寻使用的坐标
                jj=j;
                imgn(i,j)=2;     %本像素块第一个标记的边界，第一个边界像素为2

                while imgn(ii,jj)~=2     %是否沿着像素块搜寻一圈了。
                    for k=1:8            %逆时针八邻域搜索
                        tmpi=ii+ed(k,1);          %八邻域临时坐标
                        tmpj=jj+ed(k,2);
                        if img(tmpi,tmpj)==1 && imgn(tmpi,tmpj)~=2  %搜索到新边界，并且没有搜索一
                            ii=tmpi;          %更新内部搜寻坐标，继续搜索
                            jj=tmpj;
                            imgn(ii,jj)=1;  %边界标记图像该像素标记，普通边界为1
                            break;
                        end
                    end
                end
```

二、 结果图像

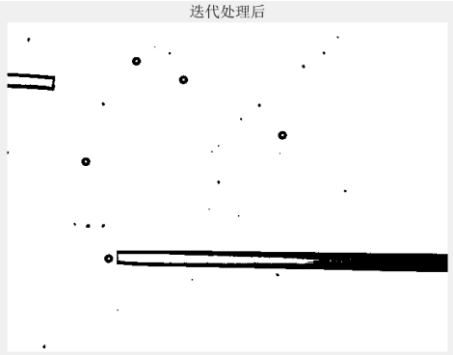
➢ **原图**



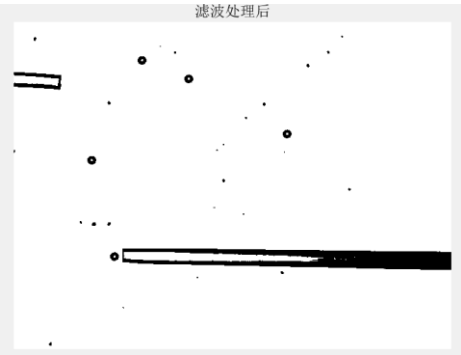
➢ **灰度图**



- **二值化图像**



- **高斯滤波**



- **内边界跟踪**