**第九讲作业**

**廖晓村-201918014628043-人工智能学院**

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

运行方式：解压文件，直接在解压目录设置为matlab的工作路径，分别运行main\_1.m、main\_2.m、main\_3.m三个文件

运行程序的结果图，如果图形窗口较小，会出现部分像素不显示。麻烦老师在运行结果后，将图形窗口放大至屏幕最大查看结果。图形窗口放大后效果更好，我也没懂为什么小窗口会出现像素不显示的这种情况。

main\_1.m：运行时间大约2-3分钟

main\_2.m：运行时间大约2-3分钟

main\_3.m：运行时间大约7-8分钟

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

1. **主要函数实现**
2. **二值化函数**

function [ output ] = rgb1gray(f, method)

% 该函数 g = rgb1gray(f, method). 函数功能是将一幅

% 24 位的RGB 图像, f, 转换成灰度图像, g. 参数 method

% 是一个字符串，当其值为’average’ 时，采用第一种转换

% 方法，当其值为’NTSC’时，采用第二种转换方法。

% 将’NTSC’做为缺省方式。

if nargin<2 %输入参数小于2，即为缺省模式

%采用‘NTSC’方式，灰度化图像

output=0.2989\*f(:,:,1)+0.5870\*f(:,:,2)+0.1140\*f(:,:,3);

else

if strcmp(method,'average') %判断是否为‘average’方式

%采用‘average’方式，灰度化图像

output=1/3\*f(:,:,1)+1/3\*f(:,:,2)+1/3\*f(:,:,3);

elseif strcmp(method,'NTSC') %判断是否为‘NTSC’方式

%采用‘NTSC’方式，灰度化图像

output=0.2989\*f(:,:,1)+0.5870\*f(:,:,2)+0.1140\*f(:,:,3);

end

end

End

1. **利用迭代法进行阈值分割**

**2.1 原理**

1. 遍历整幅图像，求取图像的最大灰度和最小灰度
2. 令k=0，初始阈值
3. 用阈值T（k）将图像分为两部分：G1（k）由灰度值大于T（k）的像素组成；G2（k）由灰度值小于等于T（k）的像素组成
4. 对G1（k）和G2（k）的所有像素求取均值

 

1. 计算新阈值：
2. 如果，则最终阈值取T（k+1）,否则转至第三步。

**2.2 代码实现**

function [ output , thread1 ] = Diedai( f )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Diedai():利用迭代法寻求图像的阈值

% f 输入图像

% output 输出图像

% thread1 阈值

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fmax = max(max(f)); %计算最大像素点

fmin = min(min(f)); %计算最小像素值

thread0 = double((fmax+fmin)/2); %计算阈值初始值

%%%%%%% 分别计算像素值小于等于k和大于k的像素平均值 %%%%%%%%%

aver\_0\_k = sum(f(find(f<=thread0)))/(length(find(f<=thread0)));

aver\_k\_255 = sum(f(find(f>thread0)))/(length(find(f>thread0)));

thread1 = 0.5\*(aver\_k\_255+aver\_0\_k); %更新阈值

while abs(double(thread1-thread0))>=0.0000001%迭代更新

thread0 = thread1; %更新前一个阈值

%%%%%%% 分别计算像素值小于等于k和大于k的像素平均值 %%%%%%%%%

aver\_0\_k = sum(f(find(f<=thread0)))/(length(find(f<=thread0)));

aver\_k\_255 = sum(f(find(f>thread0)))/(length(find(f>thread0)));

thread1 = 0.5\*(aver\_k\_255+aver\_0\_k); %更新当前阈值

end

output = zeros(size(f));

output(find(f>thread1)) = 1; %大于阈值设置为白色

output(find(f<=thread1)) = 0; %小于阈值设置为黑色

%output = f; %将f值赋值给输出变量

end

1. **腐蚀运算**

代码实现说明：首先将图像用0填充，将模板在原图上滑动，每个位置求交集并判断是否都属于原图的目标区域，为了解决模板中有不考虑的点，这里利用matlab的NaN处理，每次计算只考虑不是NaN模板的像素。

function [ output ] = fushi( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% fushi()实现对图像的腐蚀

% f:输入的二值图像

% w1:腐蚀模板

% output：腐蚀后的图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[wx,wy]=size(w1); %求w1矩阵的大小

[fx,fy]=size(f); %求f矩阵的大小

Paddingf = zeros(fx+wx-1,fy+wy-1); %初始化填充图像矩阵

total = sum(sum(w1(find(isnan(w1)==0)),1),2);

%将原图矩阵覆盖至填充图像矩阵相应位置

Paddingf((wx+1)/2:(wx+1)/2+fx-1,(wy+1)/2:(wy+1)/2+fy-1)=f(:,:);

output=zeros(fx,fy); %初始化腐蚀图像矩阵

for i = (wx+1)/2:(wx+1)/2+fx-1 %循环计算

for j = (wy+1)/2:(wy+1)/2+fy-1

%逐个点计算腐蚀结果

%利用NaN来计算不需要考虑的模板上的点

imt = min(Paddingf(i-(wx-1)/2:i+(wx-1)/2,j-(wy-1)/2:j+(wy-1)/2),w1);

in\_result = sum(sum(imt(find(isnan(imt)==0)),2),1);

output(i-(wx-1)/2,j-(wy-1)/2) = (in\_result == total);

end

end

end

1. **膨胀运算**

代码实现说明：首先将图像用0填充，将模板在原图上滑动，每个位置求交集并判断是否有属于原图的目标区域的像素，为了解决模板中有不考虑的点，这里利用matlab的NaN处理，每次计算只考虑不是NaN模板的像素。

function [ output ] = pengzhang( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% pengzhang()实现对图像的膨胀 0是目标 1是背景

% f:输入的二值图像

% w1:膨胀模板

% output：膨胀后的图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[wx,wy]=size(w1); %求w1矩阵的大小

[fx,fy]=size(f); %求f矩阵的大小

Paddingf = zeros(fx+wx-1,fy+wy-1); %初始化填充图像矩阵

%将原图矩阵覆盖至填充图像矩阵相应位置

Paddingf((wx+1)/2:(wx+1)/2+fx-1,(wy+1)/2:(wy+1)/2+fy-1)=f(:,:);

output=zeros(fx,fy); %初始化腐蚀图像矩阵

for i = (wx+1)/2:(wx+1)/2+fx-1 %循环计算

for j = (wy+1)/2:(wy+1)/2+fy-1

%逐个点计算膨胀结果

%利用NaN来计算不需要考虑的模板上的点

imt = min(Paddingf(i-(wx-1)/2:i+(wx-1)/2,j-(wy-1)/2:j+(wy-1)/2),w1);

in\_result = sum(sum(imt(find(isnan(imt)==0)),2),1);

output(i-(wx-1)/2,j-(wy-1)/2) = (in\_result ~=0);

end

end

end

1. **开运算**

function [ output ] = kaical( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% kaical()实现开运算

% f:输入的二值图像

% w1:模板

% output：开运算后的图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

output = fushi(f,w1); %先腐蚀

output = pengzhang(output,w1); %后膨胀

end

1. **闭运算**

function [ output ] = bical( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% kaical()实现闭运算

% f:输入的二值图像

% w1:模板

% output：闭运算的图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

output = pengzhang(f,w1); %先膨胀

output = fushi(output,w1); %后腐蚀

end

1. **击中或击不中变换**

function [ output] = hit\_nothit( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% hit\_nothit()实现对图像的击中击不中变换

% f:输入的二值图像

% w1:模板

% output：结果图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

AD = fushi(f,w1); %求w1对f的腐蚀

Ac = transnot(f); %求f的补集

Bc = transnot(w1); %求w1的补集

ADc = fushi(Ac,Bc); %求w1的补集对f的补集的腐蚀

output = min(AD,ADc); %求击中击不中结果

end

1. **矩阵求非**

由于用NaN表示不考虑的点，我们对模板取非，仅仅只需要对考虑的点取非，但是matlab不支持对NaN取非，因此，自定义一个取非的函数。

function [ output] = transnot( mat )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% transnot()实现对矩阵进行取非 nan的值不变

% mat：输入矩阵

% output：取非结果

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

output = mat; %初始化结果矩阵

index1 = find(mat==1); %找到原矩阵为1的索引

output(index1) = not(mat(index1));%取反

index2 = find(mat==0); %找到原矩阵为0的索引

output(index2) = not(mat(index2));%取反

end

1. **边界获取**

利用原图减去腐蚀图像即可得到边界。

function [ output] = getmargin( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% getmargin()实现对目标获取边界

% f:输入的二值图像

% w1:模板

% output：获取边界结果结果图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

fushif = fushi(f,w1); %求f对w1的腐蚀

output = f-fushif; %f-腐蚀图像为边界

end

1. **细化**

function [ output] = xihua( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% xihua()实现对图像的细化

% f:输入的二值图像

% w1:模板

% output：细化结果图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

AD = hit\_nothit(f,w1); %击中击不中变换

Ac = transnot(AD); %求击中击不中的补集

output = min(Ac,f); %求细化结果

end

1. **形态学骨架提取**

function [ output,k] = getbone( f , w1 )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% getbone()实现对图像的骨架提取

% f:输入的二值图像

% w1:模板

% output：骨架图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

output = zeros(size(f)); %初始化骨架图像

k = 0; %统计迭代次数

while sum(sum(f,1),2)~=0 %当f未被腐蚀空集、继续运算

sk = f - min(f,(kaical(f,w1)));%计算中间结果sk

output = max(output,sk); %计算前k次并集

k = k + 1; %更新迭代次数

f = fushi(f,w1); %对f进行腐蚀

end

output = double(output);

end

1. **裁剪**

function [ output] = cut( f , k)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% cut()实现对图像的裁剪

% f:输入的二值图像

% k:细化次数

% output：裁剪结果图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

kerl1 = [nan 0 0;1 1 0;nan 0 0]; %定义八个模板

kerl2 = rot90(kerl1);

kerl3 = rot90(kerl2);

kerl4 = rot90(kerl3);

kerl5 = [1 0 0;0 1 0;0 0 0];

kerl6 = rot90(kerl5);

kerl7 = rot90(kerl6);

kerl8 = rot90(kerl7);

frea = f;

for i = 1:k %对8个模板进行k轮细化

f = xihua(f,kerl1);f = xihua(f,kerl2);

f = xihua(f,kerl3);f = xihua(f,kerl4);

f = xihua(f,kerl5);f = xihua(f,kerl6);

f = xihua(f,kerl7);f = xihua(f,kerl8);

end

x2 = zeros(size(f)); %计算端点

x2 = max(x2,hit\_nothit(f,kerl1));

x2 = max(x2,hit\_nothit(f,kerl2));

x2 = max(x2,hit\_nothit(f,kerl3));

x2 = max(x2,hit\_nothit(f,kerl4));

x2 = max(x2,hit\_nothit(f,kerl5));

x2 = max(x2,hit\_nothit(f,kerl6));

x2 = max(x2,hit\_nothit(f,kerl7));

x2 = max(x2,hit\_nothit(f,kerl8));

H = ones(3);

x3 = min(frea,pengzhang(x2,H));

output = max(f,x3); %求裁剪后的结果

end

1. **距离变换**

function [ output] = distran( f , a , b )

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% distran()实现对图像的 距离变换

% f:输入的二值图像

% output：骨架提取结果图像

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

index = find(f==0); %获取f等于0的像素位置

f(index)= 5000\*(ones(length(index),1)); %背景元素放大5000

for ii=2:size(f,1) %模板1从上到下，从左到右移动

for jj=2:size(f,2)-1

temp0=f(ii,jj); %计算模板与图像对应位置和的最小值

temp1=min(f(ii,jj-1)+a,temp0);

temp2=min(f(ii-1,jj-1)+b,temp1);

temp3=min(f(ii-1,jj)+a,temp2);

temp4=min(f(ii-1,jj+1)+b,temp3);

f(ii,jj)=temp4;

end

end

for ii=size(f,1)-1:-1:1 %模板1从上到下，从左到右移动

for jj=size(f,2)-1:-1:2

temp0=f(ii,jj); %计算模板与图像对应位置和的最小值

temp1=min(f(ii,jj+1)+a,temp0);

temp2=min(f(ii+1,jj+1)+b,temp1);

temp3=min(f(ii+1,jj)+a,temp2);

temp4=min(f(ii+1,jj+1)+b,temp3);

f(ii,jj)=temp4;

end

end

output = round(f(:,2:end-1)/3); %取整 求距离

end

1. **距离变换法获取骨架**

function [output,margin\_f,dis\_f,inter\_dis\_f] = getbonedis( f,w1,a,b)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% getbone\_dis() 获取局部最大值

% f:输入的二值图像

% a,b:分别为模板的里元素的值

% w1:边界获取是腐蚀图像所用模板

% output：获取边界结果结果图像

% margin\_f:目标的边界

% disf：全局距离变换

% inter\_dis\_f: 局部距离变换

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

margin\_f = getmargin(f,w1); %获取边界函数

dis\_f = distran(margin\_f,a,b); %就是距离变换值

inter\_dis\_f = dis\_f.\*f(:,2:end-1); %计算内部的距离值

[sizex,sizey] = size(inter\_dis\_f);

centerf = zeros(sizex+2,sizey+2);

centerf(2:end-1,2:end-1) = inter\_dis\_f;

output = zeros(sizex,sizey);

for i = 1:sizex

for j = 1:sizey

if(centerf(i+1,j+1)>0)

temp1 = centerf(i+1,j+1)>=centerf(i,j);

temp2 = centerf(i+1,j+1)>=centerf(i,j+1);

temp3 = centerf(i+1,j+1)>=centerf(i,j+2);

temp4 = centerf(i+1,j+1)>=centerf(i+1,j);

temp5 = centerf(i+1,j+1)>=centerf(i+1,j+2);

temp6 = centerf(i+1,j+1)>=centerf(i+2,j+0);

temp7 = centerf(i+1,j+1)>=centerf(i+2,j+1);

temp8 = centerf(i+1,j+1)>=centerf(i+2,j+2);

output(i,j) = temp1\*temp2\*temp3\*temp4\*temp5\*temp6\*temp7\*temp8;

end

end

end

end

1. **主文件main\_1.m**

timg\_rgb = imread('timg.jpg'); %读入原图像

timg\_gray = rgb1gray(timg\_rgb); %灰度图像

timg\_2b = Diedai(timg\_gray); %图像二值化

timg\_2b = not(timg\_2b); %二值化取反，目标点为1

kerl1 = [1 1 1;1 1 1;1 1 1]; %模板

timg\_fushi = fushi(timg\_2b,kerl1); %腐蚀图像

timg\_pengzhang = pengzhang(timg\_2b,kerl1);%膨胀图像

timg\_kaical = kaical(timg\_2b,kerl1); %开运算图像

timg\_bical = bical(timg\_2b,kerl1); %闭运算图像

timg\_rely = bical(timg\_kaical,kerl1); %先开运算 后闭运算

figure(1) %绘图句柄

subplot(2,2,1) %分割绘图窗口

imshow(timg\_rgb) %显示原图

title('原图像') %添加标题

subplot(2,2,2) %分割绘图窗口

imshow(timg\_gray) %显示灰度图

title('灰度图像') %添加标题

subplot(2,2,3) %分割绘图窗口

imshow(not(timg\_2b)) %显示迭代法 --二值化图像

title('迭代法--二值化图像') %添加标题

subplot(2,2,4) %分割绘图窗口

imshow(not(timg\_rely)) %显示开运算、闭运算图像

title('开运算、后闭运算图像') %添加标题

figure(2) %绘图句柄

subplot(2,2,1) %分割绘图窗口

imshow(not(timg\_fushi)) %显示腐蚀图像

title('腐蚀后的图像') %添加标题

subplot(2,2,2) %分割绘图窗口

imshow(not(timg\_pengzhang)) %显示膨胀图像

title('膨胀后的图像') %添加标题

subplot(2,2,3) %分割绘图窗口

imshow(not(timg\_kaical)) %显示开运算图像

title('开运算的图像') %添加标题

subplot(2,2,4) %分割绘图窗口

imshow(not(timg\_bical)) %显示闭运算图像

title('闭运算的图像') %添加标题

1. **主文件main\_2.m**

gujia\_gray = imread('gujia.png'); %读入原图像

gujia\_2b = Diedai(gujia\_gray); %图像二值化

[gujia\_bone1,k] = getbone(gujia\_2b,ones(3));%形态学骨架提取

%%%%%% 调用自定义函数，距离变换获取骨架 %%%%%%%

[gujia\_bone2,margin\_fgujia,dis\_fgujia,...

inter\_dis\_fgujia]=getbonedis(gujia\_2b,ones(3),3,4);

figure(3) %绘图句柄

subplot(1,2,1) %分割窗口

imshow(gujia\_2b,[]) %显示二值图像

title('二值图') %添加标题

subplot(1,2,2) %分割窗口

imshow(margin\_fgujia,[]) %显示边界

title('边界图像') %添加标题

figure(4) %绘图句柄

subplot(1,2,1) %分割窗口

imshow(gujia\_bone1,[]) %显示形态学骨架提取

title('形态学骨架提取') %添加标题

subplot(1,2,2) %分割窗口

imshow(gujia\_bone2,[]) %距离变换法骨架提取

title('距离变换法骨架提取') %添加标题

figure(5) %绘图句柄

subplot(1,2,1) %分割窗口

imshow(dis\_fgujia,[]) %全局距离变换值

title('全局距离变换值') %添加标题

subplot(1,2,2) %分割窗口

imshow(inter\_dis\_fgujia,[]) %局部距离变换值

title('局部距离变换值') %添加标题

1. **主文件main\_2.m**

timg\_rgb = imread('timg.jpg'); %读入原图像

timg\_gray = rgb1gray(timg\_rgb); %灰度图像

timg\_2b = Diedai(timg\_gray); %图像二值化

timg\_2b = not(timg\_2b); %二值化取反，目标点为1

[timg\_bone1,k] = getbone(timg\_2b,ones(3));%形态学骨架提取

[timg\_cut] = cut(timg\_2b,2); %对二值图像进行裁剪

timg\_de = timg\_2b-min(timg\_2b,timg\_cut); %求取裁剪算法所裁剪的“枝条”

%%%%%% 调用自定义函数，距离变换获取骨架 %%%%%%%

[timg\_bone2,margin\_f,dis\_f,inter\_dis\_f]=getbonedis(timg\_2b,ones(3),3,4);

figure(10) %绘图句柄

subplot(2,2,1) %分割窗口

imshow(timg\_bone1,[]) %显示形态学骨架提取图像

title('形态学骨架提取图像') %添加标题

subplot(2,2,2) %分割窗口

imshow(timg\_bone2,[]) %显示距离变换法骨架提取

title('距离变换法骨架提取') %添加标题

subplot(2,2,3) %分割窗口

imshow(timg\_cut,[]) %显示裁剪结果

title('裁剪结果') %添加标题

subplot(2,2,4) %分割窗口

imshow(not(timg\_de),[]) %显示所裁剪的“枝条”

title('裁剪算法所裁剪的“枝条”') %添加标题

1. **实验结果**

**2.1 main\_1.m**

该文件主要用于对图像二值化，阈值分割，腐蚀，膨胀等运算，并显示结果。





结果分析：腐蚀图像，会把目标变小，膨胀图像会使目标变大。利用开运算可以消除噪声。利用闭运算可以填充小的空洞。常用的操作是先开运算，再闭运算。

**2.2 main\_2.m**

该文件主要用于对图像的目标进行骨架提取，包括形态学骨架提取，和距离变换法骨架提取，并显示结果。同时也绘制了距离变换法下，对于目标边界的局部距离值和全局距离值的图像。







结果分析：利用自定义的边界提取函数，可以获取目标的边界，效果很好。利用形态学骨架提取和距离变换法骨架提取，提取的骨架效果也很好，但是会出现有些地方，尤其是目标本身很细的地方，出现断裂。根据全局距离变换值，我们可以看出越远离目标，越白，因为距离越大。最后一幅图显示了目标内部的距离值，可以看出，在骨架的地方距离值最大，显示效果则越白。

**2.3 main\_3.m**

该文件用于指纹图像的形态学骨架提取和距离变换法骨架提取，与裁剪。并显示出效果图。



结果分析：利用形态学骨架提取和距离变换法骨架提取，提取的骨架效果也很好，对比原图，很明显指纹变细了。根据裁剪的效果图可以看出，对某些小“纸条”进行裁剪，同过最后一幅图，裁剪算法所裁剪的“枝条”，都是一些零散的点，这些都是原指纹图的“枝条”。