拓展资源 8.3 实 验 指 导

8.3.1 用 MATLAB 生成 LOG 算子的图像

1. 实验内容

用 MATLAB 生成一幅 Laplacian of Gaussian（LOG）算子的图像，并对结果进行分析。

2. 实验原理

Laplacian 算子是二阶导数算子，它是一个标量，具有各向同性的性质。因为 Laplacian 算子是二阶导数算子，所以对噪声很敏感，一般要先进行平滑滤波，再进行二阶微分。常用的平滑函数为高斯函数，高斯平滑滤波器对去除服从正态分布的噪声是很有效的。二维高斯函数及其一、二阶偏导数如下所示。

$$h(x,y)=\frac{1}{2\pi\sigma^2}\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\frac{\partial h(x,y)}{\partial x}=-\frac{x}{2\pi\sigma^4}\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}},\quad \frac{\partial h(x,y)}{\partial y}=-\frac{y}{2\pi\sigma^4}\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}$$

$$\frac{\partial^2 h(x,y)}{\partial x^2}=\frac{1}{2\pi\sigma^4}\left(\frac{x^2}{\sigma^2}-1\right)\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}},\quad \frac{\partial^2 h(x,y)}{\partial y^2}=\frac{1}{2\pi\sigma^4}\left(\frac{y^2}{\sigma^2}-1\right)\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}$$

其中，σ为高斯分布的标准方差，它决定了高斯滤波器的宽度，用该函数对图像进行平滑滤波，结果为

$$g(x,y)=h(x,y)\otimes f(x,y)$$

其中，⊗为卷积符号，图像平滑后再应用 Laplacian 算子，结果为

$$\nabla^2 g(x,y)=\nabla^2(h(x,y)\otimes f(x,y))$$

利用线性系统中卷积和微分可以交换次序的性质，得到 Laplacian of Gaussian（LOG）算子为

$$\nabla^2 h(x,y)=\nabla^2\left(\frac{1}{2\pi\sigma^2}\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}\right)=\frac{1}{\pi\sigma^4}\left[\frac{x^2+y^2}{2\sigma^2}-1\right]\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}$$

这种边缘检测方法也称为 Marr 边缘检测方法。

3. 实验方法及程序

给定变量 x，y 的位于包括原点的一个范围，按照 LOG 算子的表达式用 MATLAB 程序语言进行实现。其参考 MATLAB 程序设计如下。

```
clear;
x=-2:0.06:2
y=-2:0.06:2
sigma=0.6
y=y';
```

```
    for i=1:(4/0.06+1)
          xx(i,:)=x;
          yy(:,i)=y;
    end
    r=1/(2*pi*sigma^4)*((xx.^2+yy.^2)/(sigma^2)-2).*exp(-(xx.^2+yy.
^2)/(sigma^2));
    colormap(jet(16));
    mesh(xx,yy,r);
```

实验结果如图 8.2 所示。

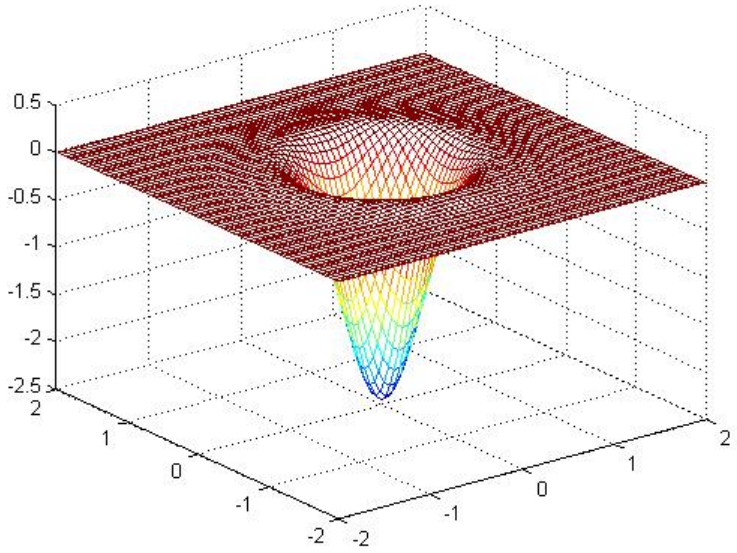


图 8.2　LOG 算子的图像

4. 思考题

（1）对参考程序给出功能注释。

（2）对实验得到的图像进行分析。

8.3.2　用分水岭算法分割图像

1. 实验内容

用分水岭算法对一幅彩色图像进行分割，并对结果进行分析。

2. 实验原理

分水岭算法（watershed）是一种借鉴了形态学理论的分割方法，在该方法中，将一幅图像看成一个拓扑地形图，其中灰度值 $f(x, y)$对应地形高度值。高灰度值对应着山峰，低灰度值对应着山谷。水总是朝地势底的地方流动，直到某一局部低洼处才停下来，这个低洼处被称为吸水盆地，最终所有的水会分聚在不同的吸水盆地，吸水盆地之间的山脊被称为分水岭。

水从分水岭流下时，它朝不同的吸水盆地流去的可能性是相等的。将这种想法应用于图像分割，就是要在灰度图像中找出不同的吸水盆地和分水岭，由这些不同的吸水盆地和分水岭组成的区域即为要分割的目标。

分水岭阈值选择算法可以看成一种自适应的多阈值分割算法，在图像梯度图上进行阈值选择时，经常遇到的问题是如何恰当地选择阈值。阈值若选得太高，则许多边缘会丢失或边缘出现破碎现象；阈值若选得太低，则容易产生虚假边缘，而且边缘变厚导致定位不精确。分水岭阈值选择算法可以避免这个缺点。

MATLAB 图像处理工具箱中的 watershed 函数可用于实现分水岭算法，该函数的调用语法为

$$\boldsymbol{L}=\text{watertshed}(f)$$

其中，f 为输入图像，$\boldsymbol{L}$ 为输出的标记矩阵，其元素为整数值，第一个吸水盆地被标记为 1，第二个吸水盆地被标记为 2，以此类推。分水岭被标记为 0。

3. 实验方法及程序

将一幅 RGB 图像转换成灰度图像，然后用分水岭算法对图像进行分割。利用 MATLAB 工具进行实验编程。其参考程序设计如下。

```
f=imread('ie_s_rice.bmp');
f=rgb2gray(f);
subplot(2,2,1);
imshow(f);
title('(a)原始图像');
subplot(2,2,2);
f=double(f);
hv=fspecial('prewitt');
hh=hv.';
gv=abs(imfilter(f,hv,'replicate'));
gh=abs(imfilter(f,hh,'replicate'));
g=sqrt(gv.^2+gh.^2);
subplot(2,2,2);
L=watershed(g);
wr=L==0;
imshow(wr);
title('(b)分水岭');
f(wr)=255;
subplot(2,2,3);
imshow(uint8(f));
title('(c)分割结果');
rm=imregionalmin(g);
subplot(2,2,4);
imshow(rm);
```

```
title('(d)局部极小值');
```

实验结果如图 8.3 所示。

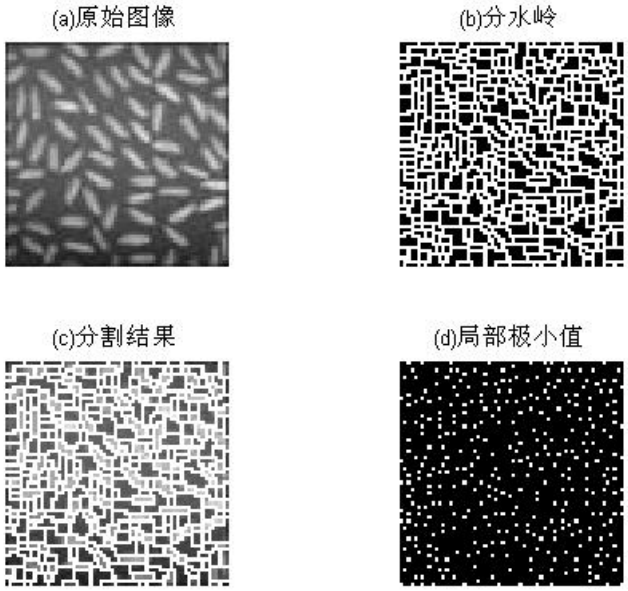


图 8.3　图像及其分水岭算法分割结果

4. 思考题

（1）对参考程序给出功能注释。

（2）对实验结果进行分析。

8.3.3　用区域分裂合并法分割图像

1. 实验内容

选择一幅灰度图像，用区域分裂合并法进行分割，并对分割结果进行分析。

2. 实验原理

区域分裂合并法是按照某种一致性准则分裂或合并区域。当一个区域不满足一致性准则时被分裂成几个小区域，当相邻区域性质相似时合并成一个大区域。它的研究重点是分裂和合并规则的设计。分裂合并法可以先进行分裂运算，再进行合并运算；也可以分裂和合并运算同时进行，经过连续的分裂和合并，最后得到图像的精确分割。具体实现时，分裂合并算法通常是基于四叉树数据表示方式进行的。具体算法如下。

（1）设整幅图像为初始区域。

（2）对每一区域 R，如果 $P(R)$=False，则把该区域分裂成 4 个子区域。

（3）重复步骤（2），直到没有区域可以分裂。

（4）对图像中任意两个相邻的区域 R_1 和 R_2，如果 $P(R_1 \cup R_2)$=True，则把这两个区域合并成一个区域。

（5）重复步骤（4），直到没有相邻区域可以合并，算法结束。

其中，P 表示具有相同性质的逻辑谓词。

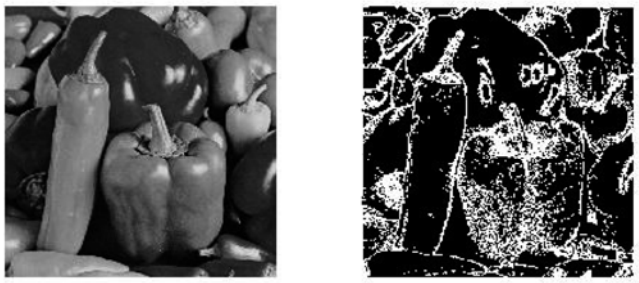
3. 实验方法及程序

对一幅灰度图像用区域分裂合并法进行分割。利用 MATLAB 工具进行实验编程。其参考程序设计如下。

```
function quadtree(x)
f=imread(x);
q=2^nextpow2(max(size(f)));
[m n]=size(f);
f=padarray(f,[q-m,q-n],'post');
mindim=2;
s=qtdecomp(f,@split,mindim,@predicate);
lmax=full(max(s(:)));
g=zeros(size(f));
marker=zeros(size(f));
for k=1:lmax
       [vals,r,c]=qtgetblk(f,s,k);
       if ~isempty(vals)
          for i=1:length(r)
             xlow=r(i);ylow=c(i);
             xhigh=xlow+k-1;
             yhigh=ylow+k-1;
             region=f(xlow:xhigh,ylow:yhigh);
             flag=feval(@predicate,region);
             if flag
                 g(xlow:xhigh,ylow:yhigh)=1;
                 marker(xlow,ylow)=1;
             end
          end
       end
end
g=bwlabel(imreconstruct(marker,g));
g=g(1:m,1:n);
f=f(1:m,1:n);
subplot(1,2,1), imshow(f),title('原始图像');
subplot(1,2,2), imshow(g),title('分割后图像');
end
function v=split(b,mindim,fun)
k=size(b,3);
v(1:k)=false;
```

```
for i=1:k
       quadrgn=b(:,:,i);
       if size(quadrgn,1)<=mindim
          v(i)=false;
          continue;
       end
       flag=feval(fun,quadrgn);
       if flag
          v(i)=true;
       end
end
end
function flag=predicate(region)
sd=std2(region);
m=mean2(region);
flag=(sd>5)&(m>0)&(m<200);
end
```

实验结果如图 8.4 所示。



（a）原始图像　　（b）分割后图像

图 8.4　图像及其分裂合并法分割结果

4. 思考题

（1）对参考程序给出功能注释。

（2）了解函数 qtdecomp 的用法和处理过程。

（3）对分割结果进行分析。