第七届“认证杯”数学中国

数学建模网络挑战赛

承　诺　书

我们仔细阅读了第七届“认证杯”数学中国数学建模网络挑战赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的, 如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们接受相应处理结果。

我们允许数学中国网站(www.madio.net)公布论文，以供网友之间学习交流，数学中国网站以非商业目的的论文交流不需要提前取得我们的同意。

我们的参赛队号为：#4585

参赛队员　(签名)　：

队员 1：冯帆

队员 2：刘浩乐

队员 3：郭佳皓

参赛队教练员 (签名)：

参赛队伍组别：本科组

第七届“认证杯”数学中国

数学建模网络挑战赛

编 号 专 用 页

参赛队伍的参赛队号：（请各个参赛队提前填写好）：

#4585

竞赛统一编号（由竞赛组委会送至评委团前编号）：

竞赛评阅编号（由竞赛评委团评阅前进行编号）：

2014 年第七届“认证杯”数学中国
数学建模网络挑战赛第二阶段论文

题　　目　基于双三次插值和拉普拉斯算子的图像放大算法

关 键 词　图像放大 双三次插值 拉普拉斯算子 锐化 图像熵

摘　　要：

图像放大即将低分辨率的图像转换为高分辨率图像的一种图像处理技术。较之低分辨率图像，高分辨率图像需要用更多的像素点来表现图像，所以图像放大的过程就是向低分辨的原始图像中添加像素点的过程，即插值过程。为了充分利用原始图像的信息，尽量减少图像失真现象的出现，我们采用双三次插值方法对原始图像进行初步放大。我们逆向地将放大后图像上的点映射到原始图像上的相应位置，通过水平、垂直双向加权计算距离该位置最近的 16 个原始图像的点的像素值来得到放大后图像上点的像素值。这样能够在考虑相邻像素点像素值的影响的同时考虑各邻点间像素值变化率的影响，较大程度地保留原始图像的信息。

由于插值算法本质是对图像点做平均化处理，处理后得到的图像损失了部分边缘轮廓信息，这使得放大后的图像出现模糊不清的现象。为了对图像的边缘轮廓信息进行补充，在已经由双三次插值得到平滑图像的基础上，我们采取了对孤立像素的响应要比对边缘或线的响应更强烈的拉普拉斯算子对放大后图像进行基于边缘检测的锐化处理。在检测边缘点的过程中，我们通过计算出图像上各个点的拉普拉斯值来突出图像中的局部边缘，根据图像中每个点的相邻点的正负号变化情况判定该点是否为像素值出现突变的零穿越点，最后通过比较零穿越点像素值的一阶差分值与既定阈值的大小来确定图像的边缘点。在实际操作中，通过反复比较调整拉普拉斯算子的模板，我们最终得到了能较好使图像清晰化的模型。

最后为了检验图像放大质量，我们采取能反映图像信息量的图像熵作为评价指标。通过纵向比较放大后图像和原始图像的图像熵、横向比较不同算法得到的图像的图像熵，我们认为基于双三次插值和拉普拉斯算子的图像放大模型能够将图像进行放大，并且放大后得到的图像是能较好反映原始图像信息，同时满足人视觉上的清晰度需求的。

参赛队号：　#4585

参赛密码 _____

所选题目：　　B　　题

Abstract

Image magnification is the technique aimed at transfer the low-resolution image to high-resolution image. Compared to the low-resolution image, the high-resolution image needs more points to present the image. So the progress of image magnification is the progress of adding points to the low-resolution original image which is interpolation actually. To fully use the information of the original image and avoid distortion, we adapt bi-cubic interpolation algorithm to magnify the image preliminarily. We find out the magnified image’s points’ corresponding points on the primary image inversely. Then after weighed summing the 16 points which have the shortest distance to the point on the primary image by both horizontal and vertical directions. By this way, we can take the influence of both neighboring points’ pixel and the changing speed between the neighboring points into consideration. The information of primary image can be kept in a large scale.

As the cure of interpolation is performing an average on the image, the output will lose some information on outlet. As a result, the magnified image can be fuzzy. To replenish the information on outlet, we use Laplacian operator which is more sensitive to isolated points than edge and line to perform sharpening on the magnified image based on edge measurement. During the process of measuring the edge points, we accentuate the partial edge by figuring every point’s Laplacian value. And the zero-crossing points with sudden change on pixel is determine by wether its neighboring points have differed plus-minus signs. Finally, the edge points is the zero-crossing points whose first difference value is larger than the threshold value. In the operating, we compare and adjust the templet of Laplacian operator, and derive a well-performing model which can magnify image well and clearly.

To value the quality of the magnification, we adapt image entropy which can present the amount of information as the evaluating indicator. By comparing the image entropy of the magnified image with the original image and among the image derived by different magnification algorithm, we confirm that the image magnification algorithm based on bi-cubic interpolation and Laplacian operator can accomplish the magnification well and clearly.

1. 问题重述

图像放大即将低分辨率的图像转换为高分辨率图像的一种图像处理技术。较之低分辨率图像，高分辨率图像需要用更多的像素点来表现图像，所以图像放大的过程就是向低分辨的原始图像中添加像素点的过程。要使放大后的图像不出现失真的现象，就要充分利用原始图像中的信息，对新添加的点进行限制。但原始图像中的信息是有限的，即使对其进行充分利用也会使放大后的图像出现噪声、伪色增强、轮廓不明显等问题。所以要提高图像放大质量，就必须在完成图像的初步放大后在图像边缘轮廓等方面进行信息补充。

2. 问题分析

考虑到向图像添加像素点，并且充分应用原图像，我们采用插值方法来完成初步的图像放大工作。常用的图像插值方法有最邻近插值、双线性插值等。但最邻近插值得到的放大图像锯齿现象严重，而双线性插值方法中，待插值点的像素值只由其在原图像中与其相邻的4个点的像素值决定，对于原图像的信息利用率较低。在双三次插值方法中，每个新插入的图像点的像素值由其在原图像上的相邻16个点共同决定，能高效利用原图像的信息，较大程度上避免失真现象的出现。所以我们考虑采用能在充分利用原图像信息的同时不过分加重计算负担的双三次插值方法对原始图像进行初步放大。

插值方法本质上是一种平均化处理，这样虽然减少了能使图像失真的噪声信息，但同时也去除了与噪声信息同样位于高频段的图像边缘信息，使得放大后的图像变得不清晰。为了提高图像放大质量，我们采用基于边缘检测的锐化方法来对图像边缘轮廓部分进行信息补充。由于已经采用双三次插值法对原图像进行了平滑处理，所以此处我们采用对孤立像素的响应要比对边缘或线的响应更强烈的拉普拉斯算子进行边缘检测。然后用检测出的边缘信息对初始放大后的图像进行锐化。

最后，为了评价算法的好坏，我们需要对放大后图像的质量进行检测，即检测放大后图像的信息量。对此，我们采用图像熵这个指标来进行检测，从而比较不同算法放大图像的能力和对于原始图像信息利用的程度。

3. 符号说明

符号	含义
$S(w)$	双三次插值的基函数
$f(i,j)$	原始图像(i,j)处图像点的像素值
u	放大后图像上的点与它映射到的原始图像位置的最近相邻十六个采样点的垂直距离

v	放大后图像上的点与它映射到的原始图像位置的最近相邻十六个采样点的水平距离
H	拉普拉斯算子模板矩阵
$L(i,j)$	初步放大后图像(i,j)处图像点的拉普拉斯值
Q	根据 Shannon 熵定义得出的图像的图像熵值
$p(x_i)$	第 i 级像素出现的概率
i	图像的像素级，$\in[0,\quad 255]$
m	被测点数，$\in Z$
n	采样点数，$\in Z$

表 1

4. 双三次插值图像放大

4.1 双三次插值算法

我们对原始图像的放大处理从原始图像出发，采用逆向映射方法，即在放大后图像中找到与之对应的原始图像中的某个或某几个像素。这样能够保证输出图像中的每个像素都有一个确定值。

双三次插值考虑到周围四个直接相邻像素点灰度值的影响，同时还考虑到各邻点间灰度值变化率的影响。在这种方法中，放大后图像上的点的像素值可以通过它映射到的原始图像位置的最近相邻十六个采样点的加权平均得到。在加权计算过程中，我们用到两个多项式插值三次函数，水平方向和垂直方向各使用一个。这样我们就得到一个连续的插值函数，它的一阶偏导数连续，并且它的交叉导数处处连续。

我们采用如下插值基函数来拟合数据：

$$S(w)=\begin{cases}1-2|w|^2+|w|^3 & |w|<1\\ 4-8|w|+5|w|^2-|w|^3 & 1\le|w|<2\\ 0 & |w|\ge 2\end{cases} \tag{1}$$

双三次插值公式如下：

$$f(i+u,j+v)=ABC \tag{2}$$

其中，A、B、C均为矩阵，其形式如下：

$$A=[S(1+u) \quad S(u) \quad S(1-u) \quad S(2-u)] \tag{3}$$

$$B=\begin{bmatrix} f(i-1,j-2) & f(i,j-2) & f(i+1,j-2) & f(i+2,j-2) \\ f(i-1,j-1) & f(i,j-1) & f(i+1,j-1) & f(i+2,j-1) \\ f(i-1,j) & f(i,j) & f(i+1,j) & f(i+2,j) \\ f(i-1,j+1) & f(i,j+1) & f(i+1,j+1) & f(i+1,j+1) \end{bmatrix} \tag{4}$$

$$C=[S(1+v) \quad S(v) \quad S(1-v) \quad S(2-v)]^T \tag{5}$$

其中$f(i,j)$表示原始图像(i,j)处像素点的灰度值，u，v表示放大后图像上的点与它映射到的原始图像位置的最近相邻十六个采样点的水平、垂直距离。

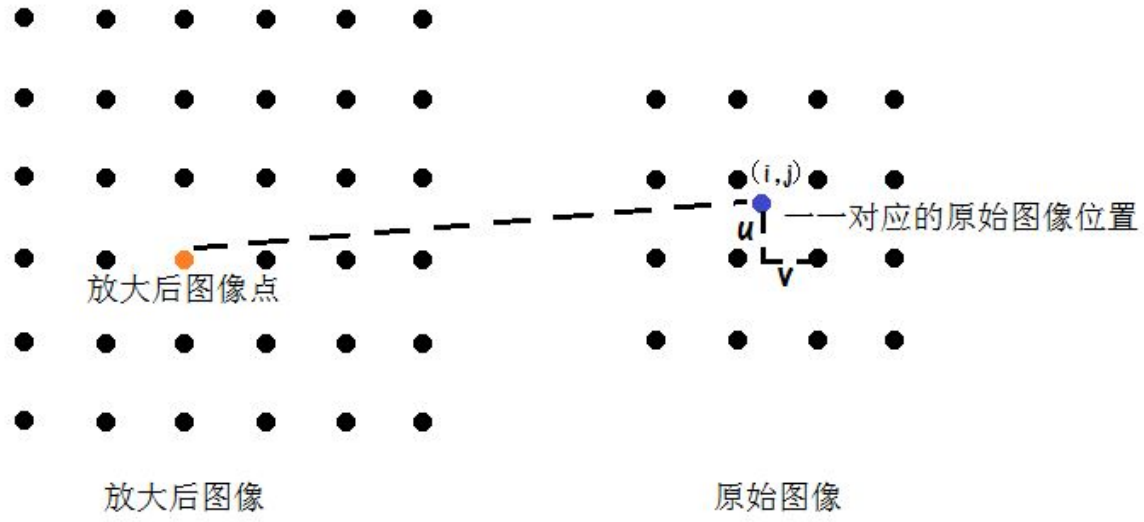


图 1 双三次插值算法说明

4.2 图像处理效果

图 2 原始图像

图 3 双三次插值算法的 4 倍放大图

双三次插值算法较清晰、平滑地将原始图像放大了，但图像线条仍有模糊现象存在。

5. 基于拉普拉斯算子的图像锐化处理

5.1 拉普拉斯算子

拉普拉斯算子是一个与边缘方向无关的边缘点检测算子。由于它对孤立像素的响应要比对边缘或线的响应更强烈，所以我们在用双三次插值对原始图像进行平滑放大后再进行基于拉普拉斯算子的图像锐化处理。

拉普拉斯算子是一种二阶微分算子。一个连续的二元函数 $f(x,y)$，其拉普拉斯运算定义为

$$\nabla^2 f = \frac{\partial f}{\partial x^2} + \frac{\partial f}{\partial y^2} \tag{6}$$

对于数字图像，拉普拉斯算子可以简化为

$$L(i,j) = 4f(i,j) - f(i+1,j) - f(i-1,j) - f(i,j+1) - f(i,j-1) \tag{7}$$

表示成卷积的形式即为

$$L(i,j)=\sum_{r=-k}^{k}\sum_{s=-l}^{l}f(i-r,j-s)H(r,s) \tag{8}$$

式中，$i,j=0,1,2,\ldots,N-1;k=1,l=1,H(r,s)$取样如下式

$$H_1=\begin{bmatrix}0&-1&0\\-1&4&-1\\0&-1&0\end{bmatrix} \tag{9}$$

在图像锐化处理过程中，函数的拉普拉斯算子也是借助模板来实现的。模板取样将直接影响锐化的效果。常用的模板有

$$Gx=\begin{bmatrix}0&-1&0\\-1&4&-1\\0&-1&0\end{bmatrix}\quad Gy=\begin{bmatrix}0&-1&0\\-1&5&-1\\0&-1&0\end{bmatrix}$$

5.2 边缘检测

边缘检测的基本思想是首先利用边缘增强算子，突出图像中的局部边缘，然后定义像素的“边缘强度”，通过设置阈值的方法提取边缘点集。检测边缘的过程就是寻找过零点的过程。

具体的边缘检测算法如下：

（1）用双三次插值算法将原始图像进行平滑放大，得到初步放大后的图像；

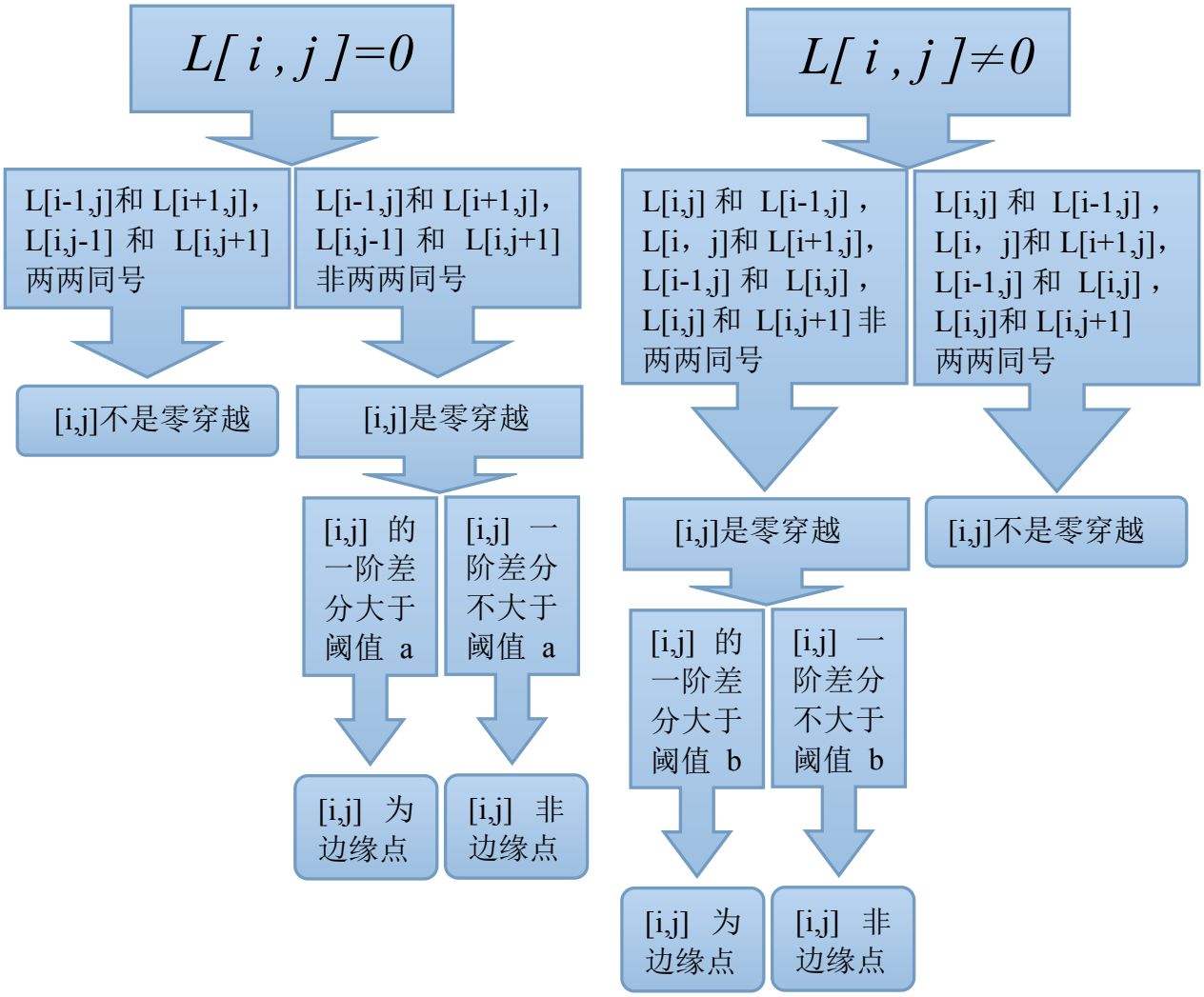
（2）对得到的图像进行过零检测。具体方法为：设得到的图像的一阶微分图像的每个点的像素值为 f[i,j]，设该点的拉普拉斯值为 L[i,j]。

（3）接下来按照下面的规则进行检测：

若 L[i,j]=0，则看数对(L[i-1,j],L[i+1,j])或(L[i,j-1],L[i,j+1])中是否包含正负号相反的两个数。只要这两个数对中有一个包含正负号相反的两个数，则点[i,j]是零穿越。然后看点[i,j]对应的一阶差分值是否大于一定的阈值，若是，则点[i,j]是边缘点，否则不是；

若 L=[i,j]不为 0，则看 4 个数对(L[i,j],L[i-1,j])、(L[i,j],L[i+1,j])、(L[i-1,j],L[i,j])、(L[i,j],L[i,j+1])中是否包含正负号相反的值。若有，那么在点[i,j]附近有零穿越，看点[i,j]对应的一阶差分值是否大于一定的阈值，若是，则将点[i,j]作为边缘点。

将上述过程制作流程图如下：


L[i , j]=0
L[i-1,j]和 L[i+1,j]，L[i,j-1] 和 L[i,j+1] 两两同号
L[i-1,j]和 L[i+1,j]，L[i,j-1] 和 L[i,j+1] 非两两同号
[i,j]不是零穿越
[i,j]是零穿越
[i,j] 的一阶差分大于阈值 a
[i,j] 一阶差分不大于阈值 a
[i,j] 为边缘点
[i,j] 非边缘点
L[i , j]≠0
L[i,j] 和 L[i-1,j]，L[i，j]和 L[i+1,j]，L[i-1,j] 和 L[i,j]，L[i,j] 和 L[i,j+1] 非两两同号
L[i,j] 和 L[i-1,j]，L[i，j]和 L[i+1,j]，L[i-1,j] 和 L[i,j]，L[i,j]和 L[i,j+1] 两两同号
[i,j]是零穿越
[i,j]不是零穿越
[i,j] 的一阶差分大于阈值 b
[i,j] 一阶差分不大于阈值 b
[i,j] 为边缘点
[i,j] 非边缘点

5.3 图像锐化处理效果

图 4 双三次插值锐化处理效果图

6. 模型检验

6.1 图像客观评价准则

常用的放大性能评价准则有信息熵、 交叉熵、PS NR 等.

（1）图片熵：熵是信息论中用来度量信息不确定性的概念，熵越大，信息不确定性越大，对应的信息量越小；熵越小，信息不确定性越小，对应信息量越多。对应到图像上则是图像熵。通过比较原始图像和放大后图像的图像熵，我们可以评价图像放大算法对原始图像的利用程度，从而评价和比较算法的好坏。

在此我们采用 Shannon 熵的定义式：

$$Q=-\sum_{i=1}^{n}p(x_i)\log(x_i) \tag{10}$$

其中，p(xi)为第 i 级像素出现的概率，i 表示图像的像素级，i 属于[0,255]。

在实际应用中，我们发现采用定义式计算图像熵计算量大，且对于像素级概率为 0 的点无法计算。因此，我们采用如下近似计算公式

$$Q=\frac{\left[\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{2}\right]^{2}}{\sum_{i=1}^{m}\sum_{j=1}^{n}x_{ij}^{4}} \tag{11}$$

其中，m 为被测点数，n 为采样点数，m,n 属于 Z。

交叉熵：交叉熵可度量两幅图像间的差异, 交叉熵越小, 图像间的差异就越小。若理想参考图像为 R,放大后图像为 F, 则交叉熵定义为:

$$\mathrm{CERF}=\sum_{i=0}^{\mathrm{L}-1}P_{R_i}\log\frac{P_{R_i}}{P_{F_i}} \tag{12}$$

（2）峰值信噪比：本文中应用的峰值信噪比（PSNR）定义为:

$$\mathrm{PSNR}=10\lg\frac{r_{\max}^{2}}{MSE} \tag{13}$$

其中 $r_{\max}=2^{h}-1$，h 表示像素点占用的 2 进制位数, 一般取 h = 8。

通过对……四种算法得到的放大后图像进行图像熵检验，我们得到如下结果

评价准则	最邻近	三次卷积	分形放大	双三次锐化
熵	7.6927	7.5991	7.5994	7.6034
交叉熵	0.0931	0.0765	0.0743	0.0572
PSNR	21.985	22.314	22.048	23.019

表2

从表 2 看出双三次插值锐化处理算法的熵较大、交叉熵较小、PSNR 较大, 说明本算法相对其他几种算法是最有效的。

6.2 图像处理的比较

通过将对最邻近插值算法、三次卷积放大算法、分形放大算法，以及双三次插值锐化处理算法处理图像的结果进行比较，通过对几何结构的保留、原图像内容的保留、线条的保护 3 个方面，我们对 4 种算法的优缺点进行评价。

（1）对于几何结构的保留：为了考察最邻近插值算法、三次卷积放大算法、分形放大算法，以及双三次插值锐化处理算法这四种方法对于图像的几何结构的保留，我们做了多幅图像进行比较，并选取了一幅具有代表性的图像进行说明。分形图像，既可以说明这四种方法对于图像的重要区域的几何性质的变化，也可以比较它们在保留图像全局几何结构中的作用。

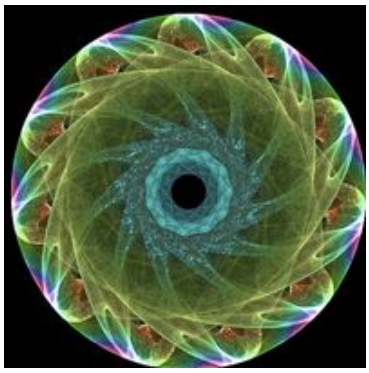


图5 原始图像

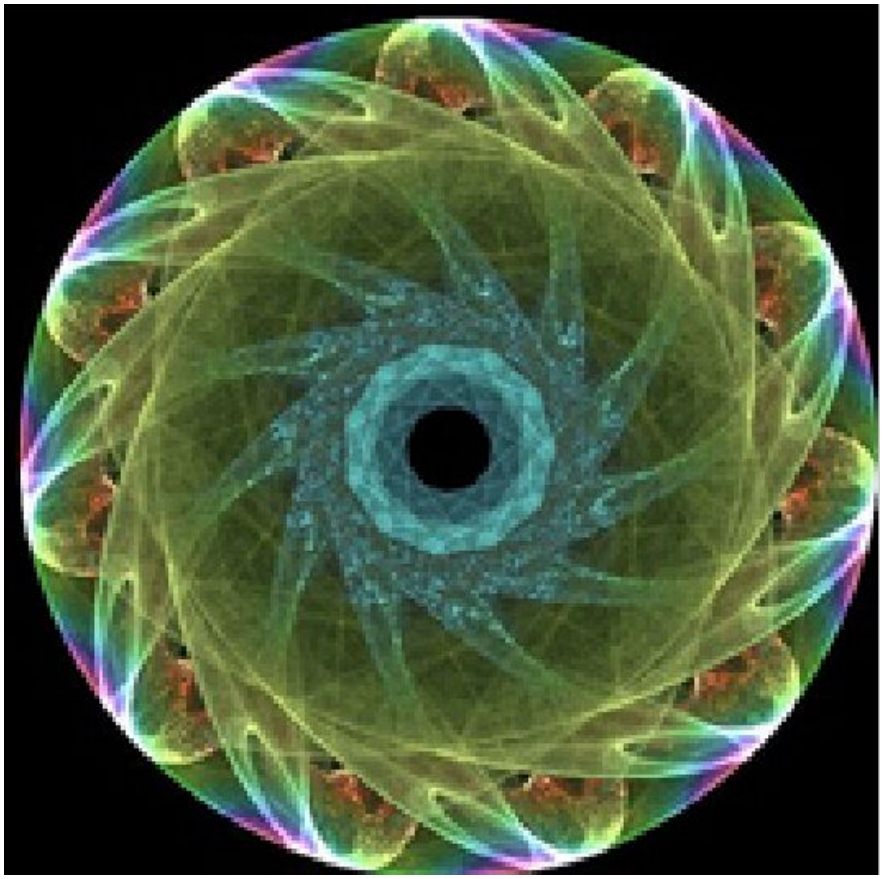


图6 最邻近插值法4倍放大

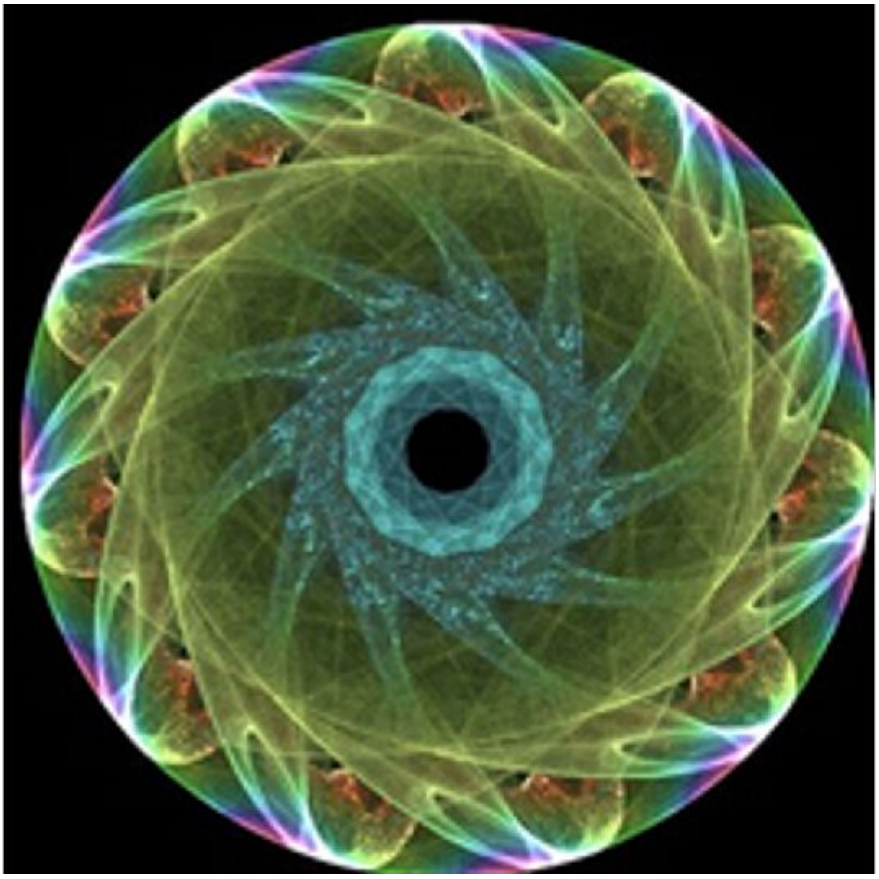


图7 三次卷积法放大4倍

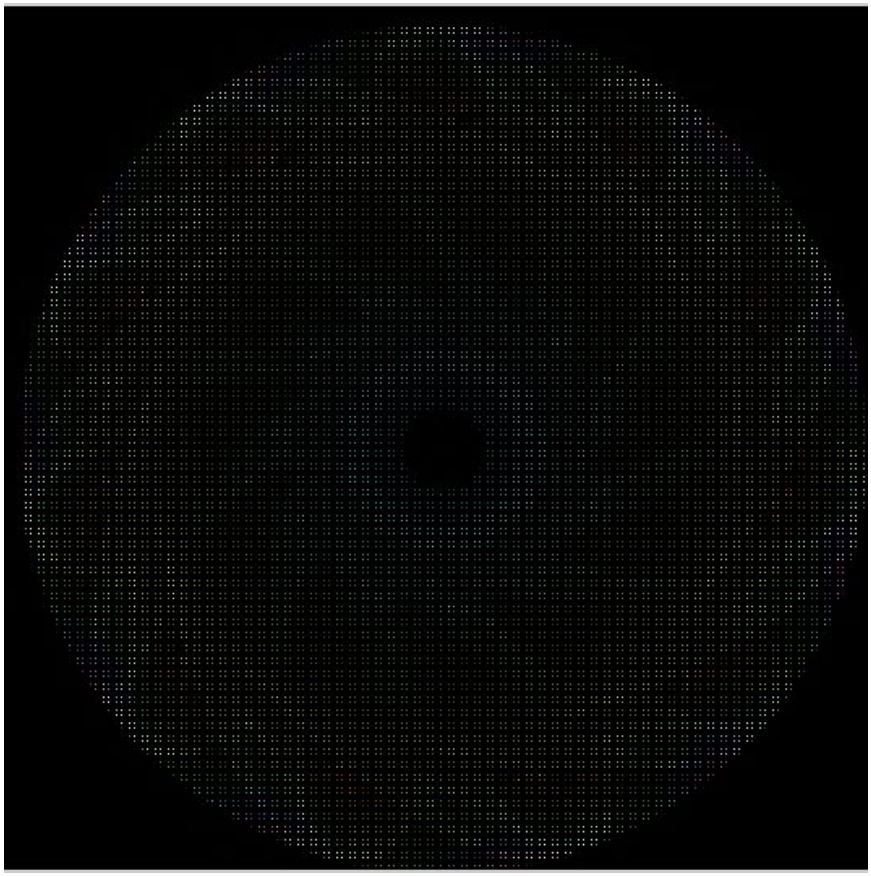


图8 分形算法放大4倍

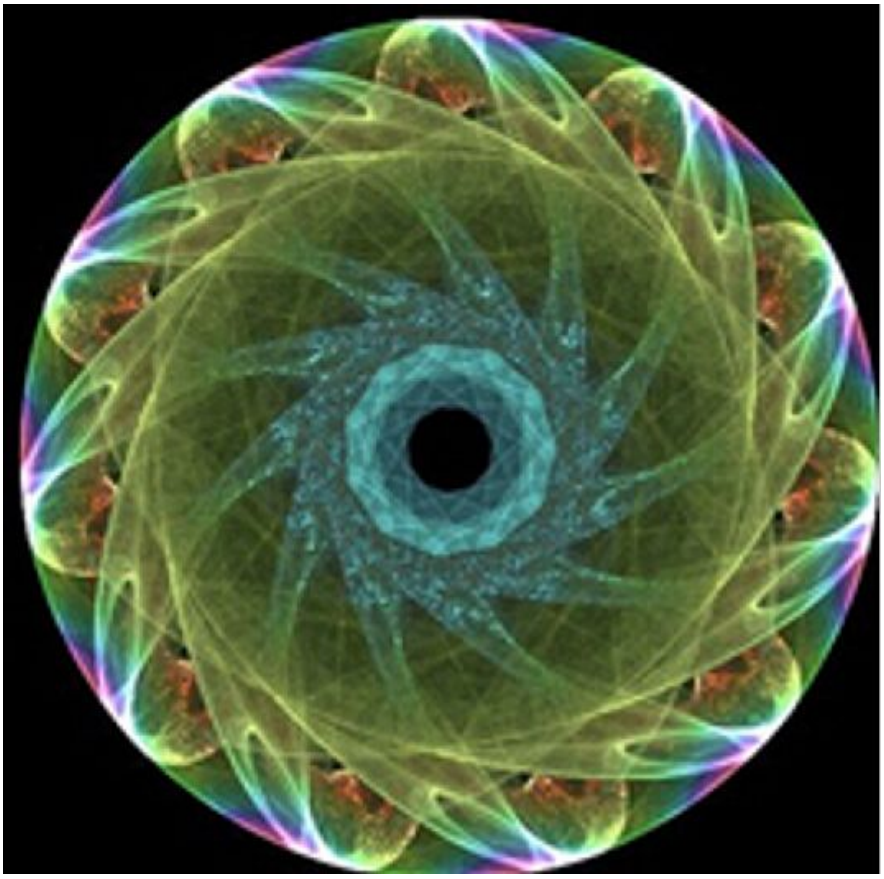


图9 双三次插值算法放大4倍

算法	重要区域几何性质	全局相对几何结构
最邻近插值	有微小变化	保证不变
三次卷积放大	保证不变	保证不变
分形放大	有微小变化	保证不变
双三次插值锐化	保证不变	保证不变

表3

（2）对于原图像内容的保留：

图 10 原始图像

图 11 最邻近插值算法放大 4 倍

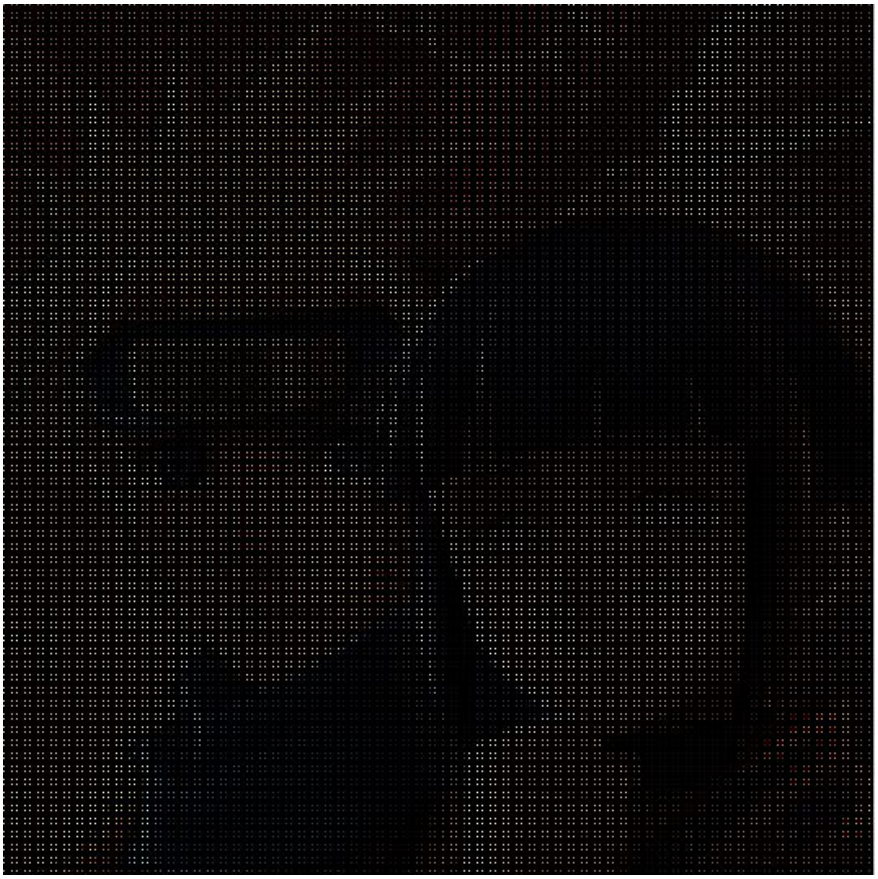


图12 分形算法放大4倍

图13 三次卷积算法放大4倍

图14 双三次插值锐化放大4倍

算法	原始图像内容保留
最邻近插值	否
三次卷积放大	是
分形放大	否
双三次插值锐化	是

表4

从以上评价中我们可以看出，不管是在图像客观评价还是在放大处理比较方面，双三次插值锐化处理算法都有着优于其他几种算法性能。

参考文献

[1] 赵海峰,周永飞,黄子强.图像放大算法比较研究[A].西安:现代电子技术,2010(24).
[2] 干宗良,齐丽娜. 基于模糊推理的图像放大算法[A].北京:中国电子科学研究院学报,2010-10(5).
[3] 任浩,谢磊,陈惠芳.基于动态边缘检测的图像锐化算法[A].杭州:杭州电子科技大学学报,2012-8(4).
[4] 张阿珍,刘政林,邹雪城,向祖权.基于双三次插值算法的图像缩放引擎的设计[A].西安:微电子学与计算机2007,24(1).
[5]吴海波,刘钊.基于拉普拉斯算子的彩色图像锐化处理[A].太原:电脑开发与应用,2008,21(9).
[6]王会鹏,周利莉,张杰.一种基于区域的双三次图像插值算法[A].上海:计算机工程,2010,36(19).
[7]申家全,闫怀志,胡昌振.基于图像熵的探地雷达杂波抑制效果评价[A].新乡:电波科学学报,2011-4,26(2)
[8]朱铮涛,黎绍发,陈华平.基于图像熵的自动聚焦函数研究.长春:光学 精密工程,2004-10,12(5)

附录

程序：

1，最邻近法代码：

```
A = imread('03.jpg');%读取图像信息
imshow(A); %显示原图
title('原图 128*128');
Row = size(A,1); Col = size(A,2);%图像行数和列数
nn=8;%放大倍数
m = round(nn*Row);%求出变换后的坐标的最大值
n = round(nn*Col);
B = zeros(m,n,3);%定义变换后的图像
for i = 1 : m
for j = 1 : n
x = round(i/nn); y = round(j/nn);%最小临近法对图像进行插值
if x==0
    x = 1;
end
if y==0
    y = 1;
end
if x>Row
    x = Row;
end
if y>Col
    y = Col;
end
B(i,j,:) = A(x,y,:);
end
end
B = uint8(B);%将矩阵转换成 8 位无符号整数
figure;
imshow(B);
title('最邻近插值法放大 8 倍 1024*1024');
```

2，图像质量评价

```
a=imread('eyechart1.bmp','bmp');
b=a([203:396],[249:440]);
a=imread('eyechart2.bmp','bmp');
c=a([203:396],[249:440]);
a=imread('eyechart3.bmp','bmp');
d=a([62:406],[60:395]);
e=imresize(d,[length(b(:,1)),length(b(1,:))],  'bicubic');%  由  于
```

eyechart3.bmp 和 eyechart1.bmp,eyechart2.bmp 比例不一样，这里要进行比例

```
调整
imwrite(b,'area_eyechart1.bmp','bmp');
imwrite(c,'area_eyechart2.bmp','bmp');
imwrite(e,'area_eyechart3.bmp','bmp');
subplot(1,3,1);
imshow(e);
title('eyechart3.bmp 截取部分，参考图象');
hold on;
subplot(1,3,2);
imshow(b);
title('eyechart1.bmp 截取部分');
hold on;
subplot(1,3,3);
imshow(c);
title('eyechart2.bmp 截取部分');

% 计算污染图象相对于源图象的质量
clc;
clear;
PSNRenable=1;%PSNR 计算使能，为 0 不计算，为 1，计算
KBlurenable=1;%模糊系数 KBlur 计算使能，为 0 不计算，为 1，计算
Qenable=1;%质量指数 Q 计算使能，为 0 不计算，为 1，计算
for m=1:2
imsrcnamehead='area_eyechart3';%源图象文件名头
imsrcnameext='bmp';%源图象文件名扩展
if m==1
imdstname=strcat('area_eyechart1','.',imsrcnameext);%污染图象文件名
elseif m==2%以 area_eyechart2.bmp 为测试图象
imdstname=strcat('area_eyechart2','.',imsrcnameext);%污染图象文件名
end
iminfo=imfinfo(strcat(imsrcnamehead,'.',imsrcnameext));%源图象信息读取
imsrc=imread(strcat(imsrcnamehead,'.',imsrcnameext));%源图象读取
imdst=imread(imdstname,imsrcnameext);%污染图象读取
doubleimsrc=double(imsrc);%转换为浮点类型
doubleimdst=double(imdst);%转换为浮点类型
%源图象和污染图象读取
W=iminfo.Width;%图象度
H=iminfo.Height;%图象高

%PSNR 计算
if PSNRenable==1
PSNR=0.0;%PSNR 赋初值
for j=1:H
```

```
  for i=1:W
PSNR=PSNR+double((doubleimsrc(j,i)-doubleimdst(j,i))*(doubleimsrc(j,i)
-doubleimdst(j,i)));
  end
end
PSNR=PSNR/W/H;
PSNR=10*log10(255*255/PSNR)
%PSNR 计算完毕
end

%模糊系数 KBlur 计算
if KBlurenable==1
Sin=0.0;%Sin 赋初值
Sout=0.0;
for j=2:H-1
  for i=2:W-1
t=doubleimsrc(j-1,i+1)+doubleimsrc(j+1,i-1)-doubleimsrc(j-1,i-1)-doub
leimsrc(j+1,i+1);
      if t<0 t=-t;
      end
      Sin=Sin+t;%源图象邻域边缘能量计算
t=doubleimdst(j-1,i+1)+doubleimdst(j+1,i-1)-doubleimdst(j-1,i-1)-doub
leimdst(j+1,i+1);
      if t<0 t=-t;
      end
      Sout=Sout+t;%污染图象邻域边缘能量计算
  end
end
KBlur=Sout/Sin
end
%KBlur 计算完毕

%质量指数 Q 计算
if Qenable==1
Q=0.0;%Q 赋初值
Qnum=0;%图象以 7X7 块大小计算每块的 Q，逐象素的移动块窗口，这里 Qnum 为
块数量的计数
for j=4:H-3
  for i=4:W-3
      midsrc=0.0;
      middst=0.0;
      varsrc=0.0;
      vardst=0.0;%源图象和污染图象块内的平均值和方差赋初值
      varsrcdst=0.0;%源图象和污染图象块内的协方差赋初值
```

```
      for n=-3:3
          for m=-3:3
              midsrc=midsrc+doubleimsrc(j+n,i+m);
              middst=middst+doubleimdst(j+n,i+m);
          end
      end
      midsrc=midsrc/49;
      middst=middst/49;
      %源图象和污染图象块内的平均值计算
      for n=-3:3
          for m=-3:3
varsrc=varsrc+(doubleimsrc(j+n,i+m)-midsrc)*(doubleimsrc(j+n,i+m)-mid
src);
vardst=vardst+(doubleimdst(j+n,i+m)-middst)*(doubleimdst(j+n,i+m)-mid
dst);
varsrcdst=varsrcdst+(doubleimsrc(j+n,i+m)-midsrc)*(doubleimdst(j+n,i+
m)-middst);
          end
      end
      varsrc=varsrc/48;
      vardst=vardst/48;
      varsrcdst=varsrcdst/48;
      if ((varsrc+vardst)*(midsrc*midsrc+middst*middst))~=0 %分母不为
零的块才计算质量指数 Q
Q=Q+4*varsrcdst*midsrc*middst/((varsrc+vardst)*(midsrc*midsrc+middst*
middst));
      %源图象和污染图象块内 Q 计算完毕
      Qnum=Qnum+1;%块计数加 1
      end
  end
end
Q=Q/Qnum
end
%质量指数 Q 计算完毕
end
```

3，分形放大算法

```
I1=imread('10.jpg');
%for a=1:1:3
%I=I1(1:200,1:205,a);
%I=im2double(I1);
i=1;
j=1;
l=1;
```

```
k=4;
[Irows,Icols,Iwth]=size(I1);
J=zeros(k*Irows,k*Icols,Iwth);
[m,n,l]=size(J);
for l=1:Iwth
    I=I1(1:Irows,1:Iwth,l);
    I=im2double(I1);
for i=1:Irows
     for j=1:Icols
         J(k*(i-1)+1,k*(j-1)+1,l)=I(i,j,l);
     end
end
for i=1:(m-1)
     for j=2:(n-1)
         if rem(i,2)==0&&rem(j,2)==0

J(i,j,l)=(J(i-1,j-1,l)+J(i+1,j-1,l)+J(i+1,j+1,l)+J(i-1,j+1,l))/4;
           if J(i,j,l)~=0
               A=J(i,j,l);
               B=2*log10(A);
               J(i,j,l)=A+B;
               if J(i,j,l)>255
                    J(i,j,l)=255;
               end
           else J(i,j,l)=0;
           end
         end
     end
end
for i=1:(m-1)
    for j=2:(n-2)
          if rem(i,2)==0&&rem(j,2)~=0

J(i,j,l)=(J(i,j-1,l)+J(i+1,j,l)+J(i-1,j,l)+J(i,j+1,l))/4;
             if J(i,j,l)~=0
                A=J(i,j,l);
                B=2*log10(A);
                J(i,j,l)=A+B;
                if J(i,j,l)>255
                    J(i,j,l)=255;
                end
             else J(i,j,l)=0;
             end
          end
```

```
     end
end
for j=1:(n-1)
     for i=2:(m-2)
          if rem(i,2)~=0&&rem(j,2)==0

J(i,j,l)=(J(i,j-1,l)+J(i+1,j,l)+J(i-1,j,l)+J(i,j+1,l))/4;
               if J(i,j,l)~=0
                  A=J(i,j,l);
                  B=2*log10(A);
                  J(i,j,l)=A+B;
                  if J(i,j,l)>255
                     J(i,j,l)=255;
                 end
               else J(i,j,l)=0;
               end
          end
     end
     end
for i=1
     for j=2:(n-1)
         if rem(j,2)==0
             J(i,j,l)=(J(i,j-1,l)+J(i,j+1,l)+J(i+1,j,l))/3;
         end
     end
end
for i=m-1
     for j=2:(n-1)
         if rem(j,2)==0
             J(i,j,l)=(J(i,j-1,l)+J(i,j+1,l)+J(i-1,j,l))/3;
         end
     end
end
for j=1
     for i=2:(m-1)
         if rem(i,2)==0
             J(i,j,l)=(J(i-1,j,l)+J(i,j+1,l)+J(i+1,j,l))/3;
         end
     end
end
for j=n-1
     for i=2:(m-1)
         if rem(i,2)==0
              J(i,j,l)=(J(i-1,j,l)+J(i,j-1,l)+J(i+1,j,l))/3;
```

```
          end
      end
end
for i=1:m
      J(i,n,1)=J(i,n-1,1);
end
for j=1:n
     J(m,j,1)=J(m-1,j,1);
end
%J=uint8(J);
%imshow(J)
%imwrite(J,'fx1.png')
end
imshow(J)
```

4，拉普帕斯算子过滤+双三次插值

```
%A=imread('03.jpg'); %读取原图像
f = imread('08.jpg');
f1=imresize(f,4,'bicubic'); %双三次插值放大 4 倍
%w = fspecial('laplacian',0)
%g1 = imfilter(f,w,'replicate');
%imshow(g1)

%f2=im2double(f);  %将 f 转换为归一化的 double 类图像
%g2=imfilter(f2,w,'replicate');
%imshow(g2,[])
%g=f2-g2;
%figure,imshow(g)

w8=[1 1 1;1 -14 1;1 1 1];
g8=f1-imfilter(f1,w8,'replicate');
%g0=imresize(g8,4,'bicubic'); %双三次插值放大 8 倍
figure,imshow(g8)
title('双三次插值后锐化 4 倍图');
```

5，图像熵

```
clc;
clear;
[filename pathname filter] = uigetfile('03.jpg','图像文件');
if filter == 0
return;
end
X = fullfile(pathname,filename);
I=imread(X);
```

```
imshow(I);
[ix,iy,iz]=size(I);
for z=1:iz
    I=I(1:ix,1:iy,z);
P1=imhist(I)/(ix*iy);
temp=double(I);
temp=[temp,temp(:,1)];
CoefficientMat=zeros(256,256);
for x=1:ix
  for y=1:iy
i=temp(x,y); j=temp(x,y+1);
CoefficientMat(i+1,j+1);
CoefficientMat(i+1,j+1)+1;
   end
end
%P2=CoefficientMat./(ix*iy);
P2=CoefficientMat./(ix*iy);
H1=0;H2=0;
for i=1:256
if P1(i)~=0
H1=H1-P1(i)*log2(P1(i));
end
end
end
```