MATLAB 综合实验之图像处理*

聂浩　无 31　2013011280

2015 年 8 月 22 日

1　基础知识

(1)　MATLAB 提供了图像处理工具箱，在命令窗口输入 help images 可查看该工具箱内的所有函数。请阅读并大致了解这些函数的基本功能。

感觉较为常用的函数有:

- image 建立图片对象, 在坐标轴中绘制, 颜色取决于现在的颜色设置
- imshow 显示图片 (按照原来图片的大小)
- imread 读取图片文件
- imwrite 写图片文件
- imabsdiff 比较两张图片的差异
- checkerboard 生成棋盘

(2)　利用 MATLAB 提供的 Image file I/O 函数分别完成以下处理：

(a)　以测试图像的中心为圆心，图像的长和宽中较小值的一半为半径画一个红颜色的圆：

因为这个图像非常小, 所以直接用循环就进行了处理, 没有进行太多优化. 图像如1, 代码见下一问。

(b)　将测试图像涂成国际象棋状的“黑白格”的样子，其中“黑”即黑色，“白”则意味着保留原图。用一种看图软件浏览上述两个图，看是否达到了目标。

因为该图像大小为,120×168, 长宽并不能被 8 整除，所以两边出现了黑边。图像如2
代码如下 (a3_1.m):

```
clear;close all;clc;
load('hall.mat');
[m,n,q]=size(hall_color);
```

*所有的.m 文件均采用 utf8 编码，windows 版 matlab 中打开可能会出现中文乱码的情况，请用其它编辑器打开

图 1: 绘制红色圆

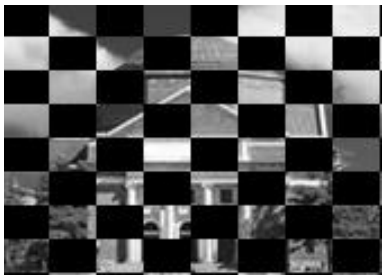

图 2: 绘制黑白格

```
r=0.5*min(m,n);
o=0.5*[m n];
circle=hall_color;
board=hall_gray;
step_m=round(m/8);
step_n=round(n/8);
for i=1:m
    for j=1:n
        if(sum(([i j]-o).^2)<=sum(r.^2))
            circle(i,j,1)=255;
            circle(i,j,2)=0;
            circle(i,j,3)=0;
        end
        if(mod(fix(i/step_m),2)==mod(fix(j/step_n),2))
            board(i,j)=0;
        end
    end
end
imshow(circle);
figure
imshow(board);
imwrite(circle,'circle.bmp');
```

```
imwrite(board,'board.bmp');
```

2 图像压缩编码

(1) 像的预处理是将每个像素灰度值减去 128，这个步骤是否可以在变换域进行？请在测试图像中截取一块验证你的结论。

根据二维 DCT 变换的定义式 $C = DPD^T$, 这是一个线性变换，所以变换前后处理是一致的。

在这里截取了 hall_gray(61:68,81:88), 两种处理次序后的绝对值差在 10^{-12} 数量级，可以认为这只是计算误差，故两者等价。

代码如下 (a3_2_1.m):

```
clc;clear;close all;
load('hall.mat');
in=hall_gray(61:68,81:88);
s1=dct2(in-128);
s2=dct2(in)-dct2(128*ones(size(in)));
e=imabsdiff(s1,s2)
```

(2) 请编程实现二维 DCT，并和 MATLAB 自带的库函数 dct2 比较是否一致。

我直接使用计算 D 矩阵然后相乘的方法进行计算，其计算复杂度为 $O(n^2)$。

系统的 DCT2 函数的调用了两次 DCT 函数，而 DCT 函数则使用了 FFT，因此其计算复杂度为 $O(n\log(n))$。

两者的误差在 10^{-12} 数量级，可以认为这只是计算误差。

在数据较大时，如图3, 系统 DCT2 函数快于我的 my_DCT2 函数。[1]

my_dct2 的代码如下：

```
function [C]=my_dct2(P)
[m n]=size(P);
D1=sqrt(2/m)*[sqrt(0.5)*ones(1,m);cos(kron((1:2:(2*m-1)),(1:m-1)')*pi
    /2/m)];
D2=sqrt(2/n)*[sqrt(0.5)*ones(1,n);cos(kron((1:2:(2*n-1)),(1:n-1)')*pi
    /2/n)];
C=D1*double(P)*D2';
return;
```

测试代码如下（a3_2_2.m）：

[1]因为计算机差异，具体值可能不同

Profile Summary

Generated 22-Aug-2015 15:30:37 using real time.

Function Name	Calls	Total Time	Self Time*	Total Time Plot (dark band = self time)
my_dct2	1	0.307 s	0.294 s	
dct2	1	0.151 s	0.016 s	
images/private/dct	2	0.135 s	0.135 s	
kron	2	0.013 s	0.013 s	

图 3: 将 hall_gray 重复 100 次后两种 DCT 变换所消耗的时间

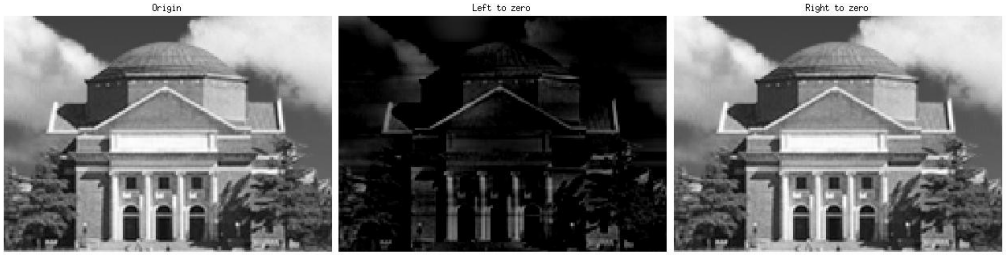

图 4: 左四列与右四列分别清零

```
clc;clear;close all;
load('hall.mat');
in=repmat(hall_gray,10,10)-128;
profile on;
s1=dct2(in);
s2=my_dct2(in);
profile viewer;
e=imabsdiff(s1,s2)
```

(3) 如果将 DCT 系数矩阵中右侧四列的系数全部置零，逆变换后的图像会发生什么变化？选取一块图验证你的结论。如果左侧的四列置零呢？

如图4，右侧四列都置零，逆变换后的图像变化不大，因为人眼对高频分量不敏感。当左侧四列都置零，逆变换图片变暗。因为很多低频分量，包括基频被滤掉，导致各点值偏小而发暗。

代码如下（a3_2_3.m）

```
clear;clc;close all;
load('hall.mat');
```

```
in=hall_gray;
subplot(1,3,1);
imshow(in);
title('Origin');
C=dct2(in);
[m,n]=size(in);
%左侧
C_l=C;C_l(:,(1:4))=0;
%右侧
C_r=C;C_r(:,(n-3:n))=0;
subplot(1,3,2)
imshow(uint8(idct2(C_l)));
title('Left to zero');
subplot(1,3,3);
imshow(uint8(idct2(C_r)));
title('Right to zero');
```

(4) 若对 DCT 系数分别做转置、旋转 90 度和旋转 180 度操作 (rot90)，逆变换后恢复的图像有何变化？选取一块图验证你的结论。

如图5，转置使得图像沿左上至右下的对角线翻转镜像；旋转 90° 使图像在之前的基础上还出现了黑白条纹；旋转 180° 后图像没有旋转，但是出现了黑白小斑点。

这是因为转置并未改变高低频信息，但两轴被交换，故出现翻转；旋转使得高频和低频分量的信息混淆，故高频相对之前被放大了——旋转 90° 只有一个方向的高频较明显，故为条纹；旋转 180° 则增强了两个方向的高频分量，故为斑点。

代码如下（a3_2_4.m）：

```
clear;clc;close all;
load('hall.mat');
in=hall_gray;
subplot(2,2,1);
imshow(in);
title('Origin');
C=dct2(in);
[m,n]=size(in);
C_tran=C';
C_90=rot90(C);
C_180=rot90(C_90);
subplot(2,2,2)
imshow(uint8(idct2(C_tran)));
```

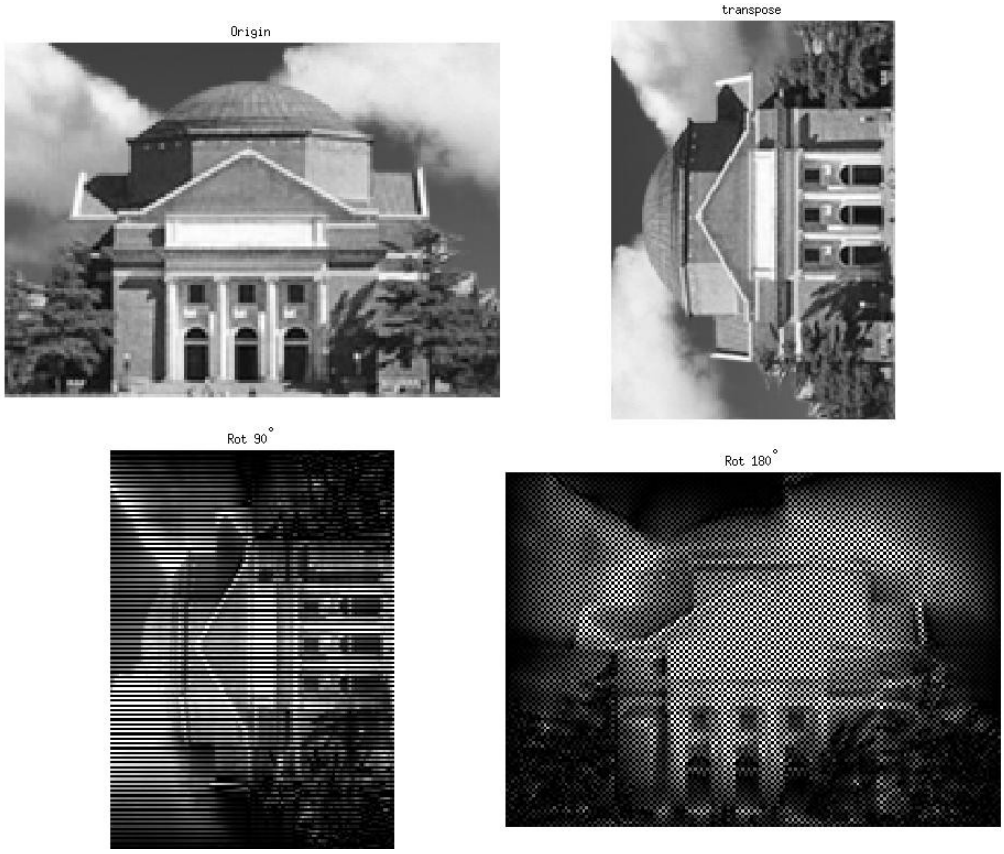

图 5: 左四列与右四列分别清零

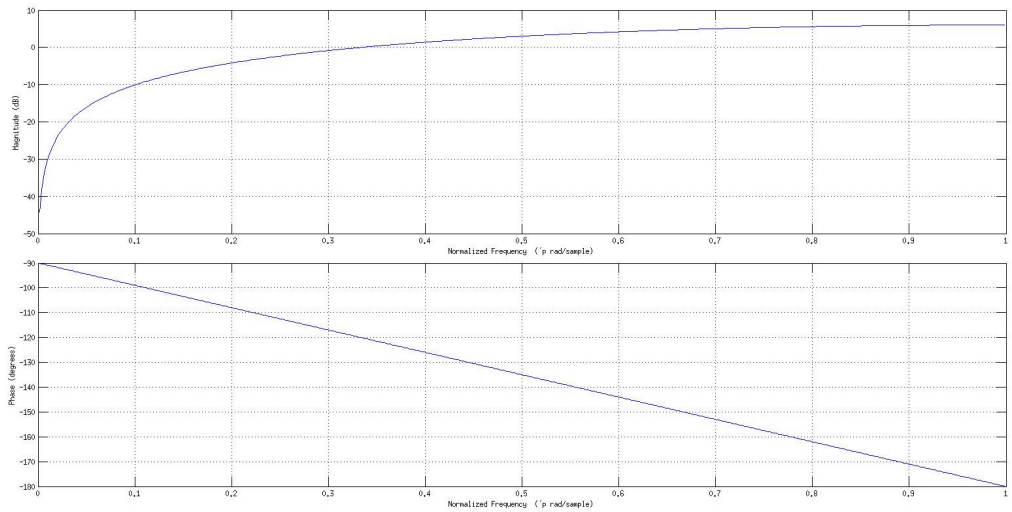

图 6: 差分的频率响应

```
title('transpose');
subplot(2,2,3)
imshow(uint8(idct2(C_90)));
title('Rot 90^{\circ}');
subplot(2,2,4);
imshow(uint8(idct2(C_180)));
title('Rot 180^{\circ}');
```

(5) 如果认为差分编码是一个系统，请绘出这个系统的频率响应，说明它是一个 _____(低通、高通、带通、带阻) 滤波器。DC 系数先进行差分编码再进行熵编码，说明 DC 系数的 _____ 频率分量更多。

差分编码的差分方程为 $y(n) = x(n-1) - x(n)$, 其系统函数为

$$H(z) = \frac{1}{z} - 1$$

仿真得到图6, 这是一个高通滤波器。说明 DC 系数的低频分量更多，这样处理可以压缩低频分量。代码如下:a3_2_5.m

```
clc;clear;close all;
b=[-1 1];
a=1;
freqz(b,a);
```

(6) DC 预测误差的取值和 Category 值有何关系？如何利用预测误差计算出其 Category?

DC 的预测误差 $\hat{c}_D = 0$ 时，Category=0，否则 $Category = fix(log2(abs(\hat{c}_D))) + 1$

(7) 你知道哪些实现 Zig-Zag 扫描的方法？请利用 MATLAB 的强大功能设计一种最佳方法。

按照最原始的思路，采用循环的方式，将元素依次放入一数组中，然后利用逻辑判断决定接下来去哪个元素。但是这样速度显然很低。更为直接的思路是利用查表法，因为该图像大小为 8×8, 直接构造一个查表矩阵是最好的，同时，通过查询[2], 将矩阵转换到一维处理是更为简便的方式，不过其 zigzag 的顺序和试验要求有一定出入，简单修改即可。

代码如下 (zigzag.m):

```
function [a]=zigzag(A)
zigtag=[1 ,2 ,9 ,17,10,3 ,4 ,11,18,...
        25,33,26,19,12,5 ,6 ,13,20,...
        27,34,41,49,42,35,28,21,14,...
        7 ,8 ,15,22,29,36,43,50,57,...
        58,51,44,37,30,23,16,24,31,...
        38,45,52,59,60,53,46,39,32,...
        40,47,54,61,62,55,48,56,63,...
        64];
%A变成1x64的矢量
aa = reshape(A',1,64);
a=aa(zigtag);
return;
```

测试该函数的代码 (a_3_7.m)

```
function [a]=zigzag(A)
zigtag=[1 ,2 ,9 ,17,10,3 ,4 ,11,18,...
        25,33,26,19,12,5 ,6 ,13,20,...
        27,34,41,49,42,35,28,21,14,...
        7 ,8 ,15,22,29,36,43,50,57,...
        58,51,44,37,30,23,16,24,31,...
        38,45,52,59,60,53,46,39,32,...
        40,47,54,61,62,55,48,56,63,...
        64];
%A变成1x64的矢量
aa = reshape(A',1,64);
```

[2]参照http://blog.sina.com.cn/s/blog_54e2ed7b0100mmb7.html

```
12 a=aa(zigtag);
13 return;
```