多媒体视频图像 实验一

计 22 黄杰 2012011272

EXP1

一、 **实验任务**

a) 将指定图片转化为灰度图
b) 使用 1D-DCT 对全图进行处理
c) 使用 2D-DCT 对全图进行处理
d) 使用 2D-DCT 对分块后的图片进行处理
e) 计算上述三种处理结果 PSNR 值与时间复杂度
f) 调整 DCT 系数进行处理

二、 **实验设计**

a) 首先封装实现了两个函数，第一个函数为 psnr，用以计算两个矩阵的 psnr 值，即计算图片处理传输前后的 PSNR 值。另一个函数为 coefficients 主要作用是将经过 DCT 后的矩阵进行系数处理，将除了矩阵左上部分的其余部分置为零值。
b) 在主程序 exp1 中，主要进行函数调用的工作，调用 rgb2gray 函数将原图转化为灰度图。然后调用 dct 与 dct2 函数进行 dct 操作。下一步调用 coefficients 函数对 DCT 后的矩阵进行处理，接着调用 idct 函数将图片解码，最后比较原图片和传输后的图片的 psnr 值，并输出处理的时间。在这里为了避免随机因素的影响，计算了 20 遍来求平均处理时间。
c) 将中间不同情况处理的图片输出到文件系统中，文件名命名规则为，1D_,2D_区分是一维 dct 还是二维 dct，是否含有 8_8 表示是否进行分块。c_coefficients，coefficients 表示处理的系数，为 1 即不进行处理。
d) 将上述得到的结果，在 result.xlsx 中进行整理。

三、 **实验结果**

不同情况下 PSNR 的值：

	1D	2D	2D-block
1	312.8807	312.8206	313.8932
1/4	36.2345	36.2345	34.8839
1/16	29.9222	29.9222	28.2162
1/64	25.8642	25.8642	23.6717

不同情况下运行 20 次的时间

	1D-DCT	2D-DCT	2D-DCT-block	1D-IDCT	2D-IDCT	2D-IDCT-block
1	0.947982	1.11122	17.683823	1.759842	1.709424	18.527721
1/4				1.709325	1.698194	19.429547
1/16				1.726767	1.735154	18.543279
1/64				1.695963	1.751273	17.782099

不同情况下图像对比：

原始图	灰度图
	

1D-DCT	2D-DCT	2D-DCT-block
		
1D-DCT 1/4	**1D-DCT 1/16**	**1D-DCT 1/64**
		
2D-DCT 1/4	**2D-DCT 1/16**	**2D-DCT 1/64**
		
2D-DCT 1/4 -block	**2D-DCT 1/16 -block**	**2D-DCT 1/64 -block**
		

四、 实验分析

a) PSNR 值比较

在不使用 coefficients 系数的时候，1D-DCT 与 2D-DCT 的 PSNR 值接近，2D-DCT-blocks 的 PSNR 值略比 1D-DCT 与 2D-DCT 大，即得到的图片的效果最好，但因为差异微小直接从图片中并不能发现很大的差异。

在使用 coefficients 系数的时候，coefficients 系数的值越小，得到的 PSNR 值越小，即图片的还原度越差。但此时的 2D-DCT-block 的效果反而不如 1D-DCT 与 2D-DCT 了。此时已经可以从图片中观察到明显的模糊现象，没有分 block 的图片整体模糊，而分了 block 的图片可以观察到小正方形之间明显的间隔。

b) 时间复杂度比较

设正方形图片边长为 N

1D 算法：计算每行的 DCT 的时候，需要计算 N 个点，每次计算需要用到 N*1 次计算，一共需要计算 N 行。同时需要再按列计算一次，两次合计复杂度为 O(N^3)

2D 算法：一共需要计算 N*N 和点，每次计算需要用到 N*N 次计算，所以共计算时间复杂度为 O(N^4)

2D-block 复杂度：共计算 N*N 个点，每个点计算需要用到 8*8 次计算，所以时间复杂度为 O(64*N^2)

从理论的时间复杂度上分析，应当是 2D-block<1D<2D，但从实际的实验结果发现 1D 与 2D 的时间类似，而 2D-block 的速度却最慢。我认为这是由于两个原因造成的第一，实验中的 N=512，并不保证 N 的值足够大，即常数对时间的影响较大。第二，实验中直接使用 matlab 的 dct 函数，在考虑到 dct 计算中每个点之间的计算并不相关可以并行，所以在封装好的函数中可能存在并行的情况。而对 2D-block 的实现是采用循环的并没有并行计算，所以造成了所用时间反而比其他两个更长的情况。

c) 综合考虑

综合以上两者，当我们在需要较高的计算精度的时候可以采用 2D-block 的方式，但当我们需要考虑运算时间的时候可以采用 1D-DCT 与 2D-DCT。同时如果图片较大的时候，可以采用 1D-DCT 那样速度会更加快。

EXP2

一、 实验任务

a) 将图片划分为 8*8 的方格，使用 2D-DCT 进行变换
b) 使用矩阵 Q 量化 DCT 系数
c) 使用 2D-IDCT，并计算 PSNR 值，并对所有 blocks 计算平均 psnr。
d) 利用系数 a 乘以矩阵 Q，并重复计算绘制 psnr-a 曲线
e) 使用 cannon 以及 nikon 矩阵重复实现。
f) 分析设计更优的量化矩阵。

二、 实验设计

a) 矩阵划分调用 mat2cell 函数，使用方式与 exp1 中类似设计 exp_quantization.m 函数，作用是给定原矩阵和量化矩阵 Q 计算对每个方块的 PSNR 值和整体图片的 PSNR 值。并将得到的图片结果保存在文件系统中。
b) 在 exp2.m 中记录了三个待计算的矩阵，调用 exp_quantization 函数进行计算，并绘制不同 a 情况下的曲线。Psnr 实验结果保存在 result.xlsx 中。
c) 根据人眼敏感情况，对不同频率分量的不同，具体实现说明在第四部分实验分析中。

三、 实验结果

原图像不同情况下的 psnr 值：

	AV_PSNR	PSNR
1Q	38.357522	36.099615
Cannon	51.636021	49.619022
Nikon	52.677666	51.135644
Test	41.7824	39.0409

测试图像不同情况下 psnr 值：

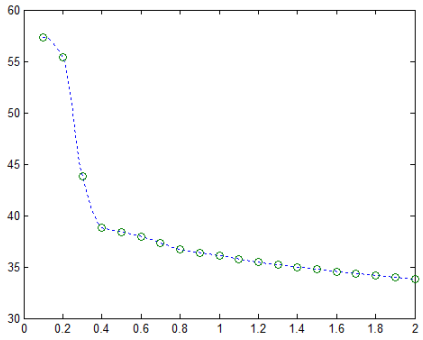
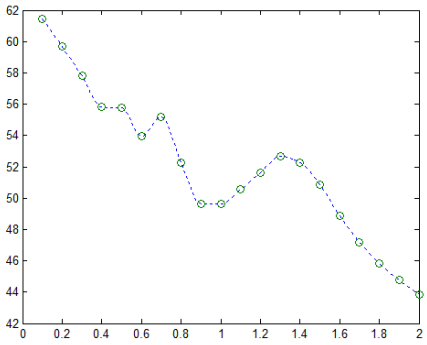
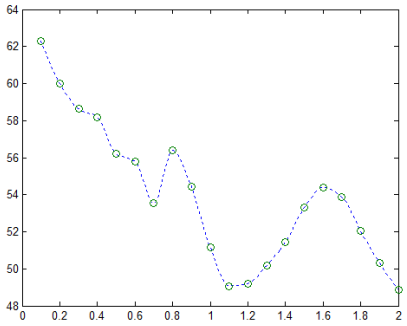
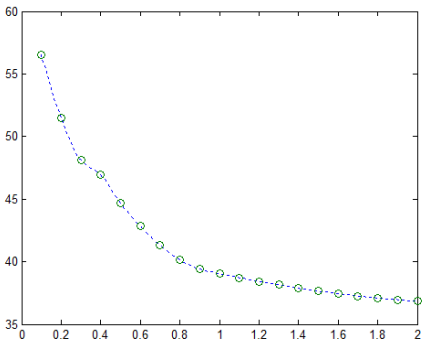
	AV_PSNR	PSNR
1Q	31.6888	31.1763
Cannon	39.7762	39.2372
Nikon	41.4101	40.8795
Test	33.7645	32.9068

Test 矩阵为（设计方案于实验分析中）：

```
test =

    1.5000    2.2500    3.3750    5.0625    7.5938   11.3906   17.0859   25.6289
    2.2500    3.3750    5.0625    7.5938   11.3906   17.0859   25.6289   38.4434
    3.3750    5.0625    7.5938   11.3906   17.0859   25.6289   38.4434   57.6650
    5.0625    7.5938   11.3906   17.0859   25.6289   38.4434   57.6650   86.4976
    7.5938   11.3906   17.0859   25.6289   38.4434   57.6650   86.4976  129.7463
   11.3906   17.0859   25.6289   38.4434   57.6650   86.4976  129.7463  194.6195
   17.0859   25.6289   38.4434   57.6650   86.4976  129.7463  194.6195  291.9293
   25.6289   38.4434   57.6650   86.4976  129.7463  194.6195  291.9293  437.8939
```

原图曲线绘制结果：

Q：a-psnr 值曲线	Cannon : a-psnr 曲线
	
Nikon：a-psnr 曲线	**Test：a-psnr 曲线**
	

原始图片结果：

原始灰度图	Q 处理结果
	
Cannon 处理结果	Nikon 处理结果
	

尝试 Cannon 与 Nikon 在不同图片上的结果

原始灰度图	Q 处理结果
	
Cannon 处理结果	Nikon 处理结果
	

四、 实验分析

a) a 与 psnr 的关系

无论是 Q 还是 Cannon 或是 Nikon，随着 a 的增大，psnr 的值呈现减小趋势。这是由于在 DCT 计算过程中，round(dct2(c{i,j})./Q).*Q;随着 Q 的值的增大，完成变换后的矩阵中的值变小越接近与零，即对于一些信息丢失更多，所以得到的 psnr 值会减小。

但对于不同的矩阵，在实际效果中存在着差异，只有 Q 是单调减小的，而 Cannon 与 nikon 都不是单调减小的，在 1 左右有一个极小值的点。这可能是不同矩阵在实际计算过程中，对于不同的图片会有不同的效果。

b) 不同矩阵比较

无论是 Cannon 矩阵或是 Nikon 矩阵的计算效果都比 Q 矩阵要好一点，而这两个矩阵之间的差距不大，Nikon 略好一点。

c) 相关因素分析与矩阵设计

由于人的眼睛对于低频的区分能力较强，但对高频分量的区分能力较弱。所以在有损压缩时，应当更多的去除高频的分量。在 DCT 问题中，矩阵左上角的部分为低频分量，右下角的部分为高频分量，所以我们在矩阵设计的时候使得左上角元素较小右下角元素较大。观察 Nikon 矩阵与 Cannon 矩阵发现其左上角到右下角为线性增加，所以设计 test 矩阵左上角到右下为指数增加，实验发现所得的 PSNR 比 Q 矩阵稍优一些，但比 Nikon 矩阵与 Cannon 矩阵还是略差一些。

d) Cannon 与 Nikon 在不同图像上的尝试

尝试使用了不同的图像，使用四种矩阵进行处理并就是那了 PSNR 值，得到的结论与原始图像基本一致。