Atitt jpeg的压缩原理

本文介绍JPEG压缩技术的原理，对于DCT变换、Zig-Zag扫描和Huffman编码，给出一个较为清晰的框架。

　8×8图像块经过 DCT 变换之后得到的 DC 直流系数有两个特点，一是系数的数值比较大，二是相邻8×8图像块的 DC 系数值变化不大。根据这个特点，JPEG 算法使用了差分脉冲调制编码（DPCM）技术，对相邻图像块之间量化 DC 系数的差值（Delta）进行编码。即充分利用相邻两图像块的特性，来再次简化数据。  
　　即上面的 DC 分量-26，需要单独处理。  
　　而对于其他63个元素采用zig-zag（“Z”字形）行程编码，以增加行程中连续0的个数。

1. 将原始图像分为8\*8的小块, 每个block里有64pixels：
2. B. 将图像中每个8\*8的block进行DCT变换：

将低频部分集中在每个8\*8块的左上角，高频部分在右下角，所谓JPEG的有损压缩，损的是量化过程中的高频部分。为什么呢？因为有这样一个前提：低频部分比高频部分要重要得多，romove 50%的高频信息可能对于编码信息只损失了5%。

 C. 量化：

所谓量化就是用像素值÷量化表对应值所得的结果。由于量化表左上角的值较小，右上角的值较大，这样就起到了保持低频分量，抑制高频分量的目的。JPEG使用的颜色是YUV格式。我们提到过，Y分量代表了亮度信息，UV分量代表了色差信息。相比而言，Y分量更重要一些。我们可以对Y采用细量化，对UV采用粗量化，可进一步提高压缩比。所以上面所说的量化表通常有两张，一张是针对Y的；一张是针对UV的。

通过量化可以reducing the number of bits and eliminating some of the components，达到通低频减高频的效果，如下图所示就是两张量化表的例子.

   D. 编码分类：

编码信息分两类，一类是每个8\*8格子F中的[0,0]位置上元素，这是**DC**（直流分量），代表8\*8个子块的平均值，JPEG中对F[0,0]单独编码，由于两个相邻的8×8子块的DC系数相差很小，所以对它们采用[差分编码DPCM](http://en.wikipedia.org/wiki/Delta_encoding)，可以提高压缩比，也就是说对相邻的子块DC系数的差值进行编码。

E. 编码格式：

上面，我们得到了DC码字和 AC行程码字。为了进一步提高压缩比，需要对RLE编码结果再进行熵编码，这里选用Huffman编码。Huffman编码具体不讲啦，详细地看我以前的这[篇Huffman编码——原理与实现吧](http://blog.csdn.net/abcjennifer/article/details/8020695)。

上面就是整个JPEG压缩算法的基本步骤。JPEG算法之所以采用DCT变换，是因为经过DCT变换得到的系数C是稀疏的，也就是只有少数位置的值是比较大的， 其他的都接近于0，因此我们就用0来代替，这样就可以省下了不少的存储空间。

JPEG压缩原理 - Rachel Zhang的专栏 - 博客频道 - CSDN.NET.html

JPEG图片存储格式及原理 - huangxy10的专栏 - 博客频道 - CSDN.NET.html