

多媒体技术基础 Project3

1. Jpeg 编码过程

1.1 读取 bmp 数据

即读取 bmp 每个像素的 rgb 数据：按照 bmp 的大小分割为 8×8 的块，创建对应的块用于存储 bmp 的 rgb 数据。每次读取一个像素数据，计算好这个像素存储的块以及块内偏移量，按顺序读取 bgr 数据存入刚计算好的位置

1.2 rgb 转化为 ycbcr

即将 rgb 数据转为 ycbcr 数据：创建对应的 ycbcr 的块数组用于存储 ycbcr 数据，循环遍历 rgb 块数组，每次向 rgb_to_ycbcr 函数传入一个块的 rgb 数据与存储 ycbcr 数据的块，然后按照公式将 rgb 数据转化为 ycbcr 数据并存储。转化时已经将数据变换到 -128 到 127 的范围，方便下一步 DCT 变换

1.3 DCT 离散余弦变换

即按照公式将 ycbcr 数据进行离散余弦变换：离散余弦变换是将 8×8 的块按照公式将块中的每个数据进行变换。分块已经在步骤一中分好。

循环遍历 ycbcr 块数组，每次向 fdct 函数传入一个 ycbcr 的块，进行离散余弦变换。

1.4 量化

即将经过 DCT 变换的数据按照亮度和色度量化表进行量化：量化是将每个块 DCT 后的数据除以亮度或者色度表对应的数据，所得商进行取整并存储，即可完成量化。

创建量化数据块数组用于存储量化后的数据，循环遍历 ycbcr 块数组（已经经过 DCT 变换），每次向 jpeg_quant 函数传入 ycbcr 的块和存储量化数据的 quant 块，函数内部安好量化表进行计算，并存储量化后的数据

1.5 ZigZag 排序

由于数据是按行列来连续存储，在行列交换区数据联系不大，且量化后，数据基本在块的左上角，所以按照 ZigZag，对每个块的数据重新进行排序，排序后的数据基本后面部分全为 0，有用数据集中在前面，这样也有利于后面的 AC 系数的行程长度编码

将量化后的数据，调用 jpeg_compress 函数，每次传入一个块的 Y 或者 Cb 或者 Cr 数据，按照 ZigZag 将传入数据重新排序

1.6 DC 差分脉冲编码调制与 AC 系数行程长度编码

由于相邻块的 DC 系数差别不大，所以采用 DC 差分脉冲调制编码，记录下两个 DC 系数之间的差，那么压缩时只记录差比记录 DC 原值可以节省很多空间。

由于经过量化与 ZigZag 排序，AC 系数基本大部分为 0，且集中在数据的后面，对 AC 系数进行行程长度编码，记录两个非 0 数据间 0 的个数与该位置的值得值，可以节省很多的空间。且对 AC 系数编码前进行判断，如果剩下

的数据全为 0，直接使用 EOB 代替，可以省下更多的空间

在 `jpeg_compress` 函数里面，进行 ZigZag 排序后，即进行 DC 差分脉冲编码，并压缩写入（压缩数据在第七步骤），写入 DC 系数后，对 AC 系数也进行行程长度编码，进行压缩写入。

1.7 熵编码（Huffman 压缩编码）与数据写入

由于 DC 差分脉冲编码与 AC 行程长度编码的长度仍然较大，所以对这两个要进行 Huffman 压缩编码。针对 Y（亮度）的 DC 与 AC 系数，按照标准亮度的 Huffman 表进行压缩编码，针对 CbCr（色度）的 DC 与 AC 系数，按照标准色度的 Huffman 表进行压缩编码。

在 `jpeg_compress` 函数中，传入时数据是 Y 或者 Cb 或者 Cr 已经对应的 Huffman 压缩表，按照 Huffman 表对数据进行压缩，并且每个块按照 Y、Cb、Cr 的顺序进行压缩写入。

2. Huffman 原理

2.1 原理

依据信源字符出现的概率大小来构造代码，对出现概率较大的信源字符，给予较短码长，对于出现概率较小的信源字符，给予较长的码长，最后使得编码的平均码字最短，以此来提高数据压缩率，提高传输效率。

2.2 压缩步骤

- 1 将信源符号按概率递减顺序排列；
- 2 把两个最小的概率加起来，作为新符号的概率；
- 3 重复前两步，直到概率和达到 1 为止；
- 4 在每次合并消息时，将被合并的消息赋予 1 和 0 或 0 和 1；
- 5 寻找从每一信源符号到概率为 1 的路径，记录下路径上的 1 和 0；
- 6 对每一符号写出从码树的根到中节点 1、0 序列，该序列就是对应的 huffman 编码
- 7 将信源每个字符按照对应的编码进行写入

2.3 jpeg 中的应用

Jpeg 进行 DCT、量化后，使用 DC 差分脉冲编码调制和 AC 行程长度编码对量化数据进行编码，虽然编码的有多种，但是绝大多数的编码取值大都相同，而且以上两种编码占用的位数较多，需要进一步进行压缩以减少大小。所以就采用了 Huffman 编码，经过事先的统计生成 Huffman 表，为经常出现的 DC 和 AC 编码分配较短的位，为较少出现的编码分配较长的位，这样用短的 Huffman 码替换长度 DC 和 AC 编码，就可以很大程度上节约空间。

3. DCT 原理

3.1 原理

离散余弦变换是与傅里叶变换相关的一种变换，可分离的变换，其变换核为余弦函数。DCT 除了具有一般的正交变换性质外，它的变换阵的基向

量能很好地描述人类语音信号和图像信号的相关特征。用于对信号和图像（包括静止图像和运动图像）进行有损压缩。由于大多数图像的高频分量比较小，相应的图像高频分量的 DCT 系数经常接近于 0，再加上高频分量中只包含了图像的细微的细节变化信息，而人眼对这种高频成分的失真不太敏感，因此考虑将这一些高频成分予以抛弃，从而降低需要传输的数据量。操作以后，传送 DCT 变换系数的所需要的编码长度要远远小于传送图像像素的编码长度。到达接收端之后通过反离散余弦变换就可以得到原来的数据，虽然这么做存在一定的失真，但人眼是可接受的，而且对这种微小的变换是不敏感的。

3.2 Jpeg 中的应用

在 jpeg 压缩算法中，将输入图像划分为 8×8 的图像块，对每个图像块数据作 DCT 变换，编程 DCT 数据；然后量化以进一步减少数据量；最后使用 DC 差分脉冲编码调制和 AC 行程长度编码以及 Huffman 编码来完成压缩任务。解压缩时首先对每个图像块作逆量化和 DCT 逆变换，然后将图像拼接成一副完整的图像。