第5章 静态图像压缩技术标准

随着传真机、数码相机等图像采集设备的快速发展，以及通信和 Internet 技术的飞速发展，如何有效地传输和存储大量的图像数据成为亟待解决的问题。为了实现静态图像数据的有效压缩，各国学者针对二值图像和灰度图像提出了多种压缩技术，并随着压缩技术的发展，国际组织制定了一系列图像压缩的编码标准。

本章首先介绍静态图像压缩技术的工作原理，然后介绍基于小波的图像压缩技术，最后对静态图像压缩编码标准做详细的介绍。

5.1 静态图像压缩编码

5.1.1 静态图像压缩编码原理

图像数据是用来表示图像信息的，如果不同的方法为表示相同的信息使用了不同的数据量，那么使用较多数据量的方法中，有些数据必然代表了无用的信息，或者是重复地表示了其他数据表示的信息，前者称为数据冗余，后者称为不相干信息。图像压缩编码的主要目的，就是通过删除冗余的或者是不相干的信息，以尽可能低的数码率来存储和传输数字图像数据。图像压缩编码技术可以追溯到 1948 年提出的电视信号数字化，迄今已经有近 70 年的历史了。

图像编码压缩是指在满足一定图像质量的条件下，用尽可能少的数据量来表示图像。编码技术比较系统的研究始于 Shannon 信息论，从此理论出发可以得到数据压缩的两种基本途径。一种是联合信源的冗余度融于信源间的相关性之中，去除它们之间的相关性，使之成为或基本成为不相干信源，如预测编码、变换域编码、混合编码等，但也都受信息熵的约束。总体上可以概括为熵编码、预测编码、变换编码，也称为三大经典编码方法。另一种是设法改变信源的概率分布，使其尽可能的非均匀，再用最佳编码方法使码长逼近信源熵。使用此途径的压缩方法其效率一般以其熵为上界，压缩比饱和于 10∶1，如 Huffman 编码、算术编码、行程编码等。随着人们对传统压缩编码方法的深入研究和应用，逐渐发现了这些传统方法的许多缺点，如高压缩比时恢复图像会出现方块效应、人眼视觉系统(HVS)的特性不易被引入算法中等。为了克服这些缺点，1985 年 M. Kunl 等人提出了第二代图像压缩编码的概念。经过 30 多年的发展，在这一框架下，人们提出了几种新的编码方法：分形编码、小波变换编码和基于模型的编码方法等。于是，对数据压缩技术的研究就突破了传统

Shannon 理论的框架，使得压缩效率得以极大提高。

数字图像的冗余主要表现为以下几种形式：空间冗余、时间冗余、信息熵冗余、结构冗余和知识冗余。图像数据的这些冗余信息为图像压缩编码提供了依据。图像编码的目的就是充分利用图像中存在的各种冗余信息，特别是空间冗余、时间冗余以及视觉冗余，以尽量少的比特数来表示图像。利用各种冗余信息，压缩编码技术能够很好地解决在将模拟信号转换为数字信号后所产生的带宽需求增加的问题，它是使数字信号走上实用化的关键技术之一，虽然表示图像需要大量的数据，但是图像数据是高度相关的，或者说存在冗余信息，去掉这些信息后可以有效压缩图像，同时不会损害图像的有效信息。

5.1.2　静态图像压缩编码分类

图像压缩分为无损压缩和有损压缩。有损压缩分为预测编码、变换编码、混合编码，有损编码分为 JPEG、MPEG。无损编码分为 Huffman 编码、游程编码、算术编码。目前常用的数字图像无损压缩编码方法可分为两大类：一是冗余压缩法，也称为无损压缩法；另一无损压缩的算法删除的仅仅是冗余的信息，因此可以在解压缩时精确地恢复原图像。有损压缩算法把不相干的信息也删除了，解压缩时只能对图像进行类似的重构，而不能精确地复原，所以有损压缩算法可以达到更高的压缩比。对于多数图像来说，为了达到更高的压缩比，保真度的轻微损失是可以接受的；有些图像不允许进行任何修改，只能对它们进行无损压缩。无损压缩利用数据的统计特性进行数据压缩，其压缩率一般为 2∶1～5∶1。有损压缩不能完全恢复数据，而是利用人的视觉特性（人的眼睛好比是一个“积分器”）使解压缩后的图像看起来与原始图像一样。图 5.1 给了依据压缩原理对现有主要的静态图像压缩算法行分类的结果。

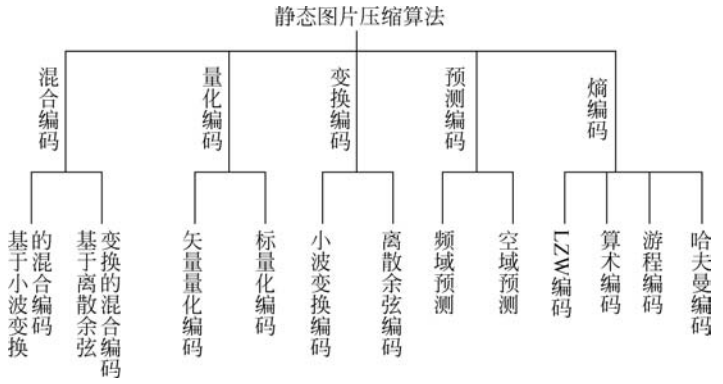

图 5.1　静态图片压缩算法分类

5.2　基于小波变换的图像压缩技术

随着科学技术特别是计算机技术的发展以及互联网的普及，许多应用领域（如卫星监测、地震勘探、天气预报）都存在海量数据传输或存储问题，如果不对数据进行压缩，数量巨大的数据就很难存储、处理和传输。因此，伴随小波分析的诞生，数据压缩一直是小波分析

的重要应用领域之一，并由此带来巨大的经济效益和社会效益。

5.2.1 数字图像的小波变换

数字图像的小波变换实际上是二维离散小波变换，在实际应用中用两次一维小波变换来实现一次二维小波变换。二维图像信号可用一维矩阵表示，可先对该矩阵的行(列)进行一维行(列)小波变换，再对变换后的系数进行一维列(行)小波变换。图像信号经过两次一维小波变换后，将图像分割成四个频带，即水平方向、垂直方向和对角线方向的高频部分和低频部分，低频部分再继续分解，这样图像信号被分解成许多具有不同空间分辨率、不同频率特性和方向特性的子图像信号，使得图像信号的分解更适合于人的视觉特性和特征数据压缩的要求。

低频子带的小波系数代表着图像信号的整体特征，远远大于其他子带的小波系数；高频子带的信息反映了图像的边缘、纹理等细节信息，它反映了图像信号的细节变化。因此小波变换同时不但具有良好的时频局部性，而且具有良好的空间方向性特点，这正反映了原始图像的像素及其间的相关性。小波变换前后，其能量不变，且主要集中在低频部分，并随着小波变换级数的增高，其能量集中特性越好，因而数据压缩的效果也将会越好。

目前，典型的小波图像编码都是嵌入式编码特性。所谓嵌入式编码是指编码器输出的码流具有这样的特点：一个低比特编码嵌入在码流的开始部分，即从嵌入式的起始至某一位置这段码流取出后，它相当于一个更低码率的完整的码流，由它可以解码重构该图像。与原码流相比，这部分码流解码出的图像具有更低的质量或分辨率，但解码后的图像是完整的。因此，嵌入式编码器可以在编码过程的任一点停止编码，解码器也可以在获得的码流的任一点停止解码，其解码效果只是相当于一个更低码率的完整的码流的解码效果。嵌入式码流中比特的重要性是按次序排列的，排在前面的比特更重要，显然，嵌入式码流非常适合用于图像的渐近传输、图像浏览和因特网上的图像传播。

1992 年，A. S. Lewis 和 G. Knowles 首先介绍了一种树形数据结构来表示小波变换的系数。1993 年，美国学者 J. M. Shapiro 把这种树形数据结构叫作“零树(Zerotree)”，并且开发了一个效率很高的算法用于熵编码，他的这种算法叫作嵌入式零树小波(Embedded Zerotree Wavelet，EZW)算法。EZW 算法首先完整地提出了基于比特连续逼近的图像编码方法：按位平面(Bit-plane)分层进行孤立系数和零树的判决和熵编码，而判决阈值则逐层折半递减，故可称之为多层(或位平面)零树编码方法。EZW 方法充分应用小波变换的时频局部化特性，具有编码效率高、嵌入式码流结构和运算复杂度较低等显著特点，对小波的图像压缩的研究起到了显著的推动作用。此后，围绕 EZW 方法，涌现出了许多基于零树的改进算法，如分层树的集划分(Set Partitioning in Hierarchical Tree，SPIHT)、集合分裂嵌入块编码(Set Partitioned Embedded block Coder，SPECK)、可逆的嵌入小波压缩法(Compression with Reversible Embedded Wavelets，CREW)等。

5.2.2 EZW 算法

从 5.2.1 节知道小波系数的分布特点是，越往低频方向子带系数值越大，包含的图像信息越多，低频部分对应于图像信号的整体特征，包含图像的大部分信息，而高频部分对应于

原图像的边沿、纹理等细节信息，对视觉来说不太重要。这样对相同数值的系数选择先传较低频的系数的重要比特，后传输较高频系数的重要比特。正是由于小波系数具有的这些特点，它非常适合于嵌入式图像的编码算法。

嵌入式零树小波编码即 EZW 编码基于以下三个主要思想，即：①利用小波变换在不同尺度间固有的相似性来预测重要信息的位置；②逐次逼近量化小波系数；③使用自适应算术编码来实现无损数据压缩。其算法框图如图 5.2 所示。

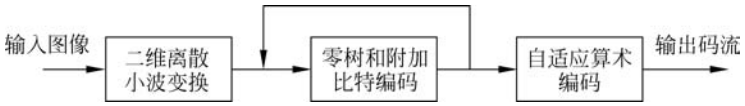

图 5.2　EZW 编码算法框图

EZW 算法中，嵌入式码流的实现是由零树结构结合逐次逼近量化实现的，零树结构的目的是为了高效地表示小波变换系数矩阵中非零值的位置。

1. 零树表示

一幅经过小波变换的图像按其频带从低到高形成一个树状结构，树根是最低频子带的节点（见图 5.3 左上角），它有三个孩子，分别位于三个次低频子带的相应位置，其余子带（最高频子带除外）的节点都有四个孩子位于高一级子带的相应位置，这样如图 5.3 所示的三级小波分解就形成了深度为 4 的树。

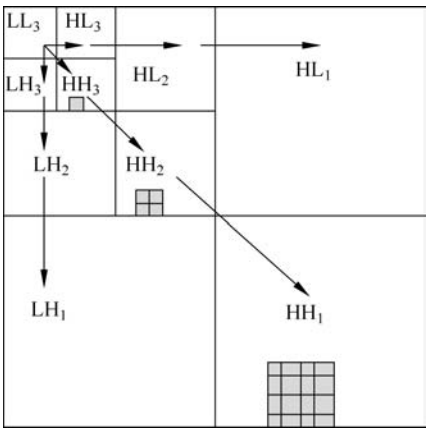

图 5.3　三级小波分解及零树结构示意图

经小波变换后的变换系数矩阵经过量化后产生大量零符号，编码的后续过程就是有效地表示那些非零符号，包括非零符号的位置和大小。量化过程产生零符号和非零符号的过程也等价为一个门限过程。对于一个给定的门限 T，如果一个小波系数 X_k 满足 $|X_k|<T$，则称小波系数 X_k 是无效的，产生零符号，否则产生非零符号。表示量化后非零值位置的过程，称为有效值映射。如果一个小波系数在一个粗的尺度上关于给定的门限 T 是无效的，之后在较细的尺度上在同样的空间位置中的所有小波系数也关于门限 T 是无效的，则称这些小波系数形成了一个零树。这时，在粗的尺度上的那个小波系数称为母体，它是树根，在较细尺度上相应位置上的小波系数称为孩子。正是通过这种零树结构，使描述有效系数

($|X_k|\geqslant T$)的位置信息大为减少。

为了构成一个完整的有效值映射,需要定义 4 种要素:零树根、孤立零点、正有效系数、负有效系数。其中,于一个给定的阈值 T,如果系数 X_k 本身和它的所有的子孙都小于 T,则该点就称为零树根;如果系数本身小于 T,但其子孙至少有一个大于或等于 T,则该点就称为孤立零点。在编码时分别用 4 种符号与之对应,即用 PSO、NEG 分别表示正、负有效系数,IZ、ZTR 分别表示孤立零点和零树根。

使用这 4 种符号,可以按一定顺序扫描小波变换系数矩阵,从而形成一个符号表,也就是要得到的有效值映射。扫描开始阈值为小波系数最大值的一半,并且化为整数。对于绝对值大于阈值的小波系数被认为是有效的,该阈值扫描编码结束后,将阈值减半,直到达到一定的压缩比或字节预算后停止扫描。扫描过程从低频子带 LL_N 开始,按如图 5.4 所示的次序,在每个子带中,按从上到下、从左到右的次序,当遇到一个系数是正有效值时,将 PSO 放入表中;若是负有效值时,将 NEG 放入表中;若是孤立零点时,将 IZ 放入表中;若遇到一个系数是零树根时,将 ZTR 放入表中,同时对 ZTR 的所有子孙系数进行标注。解码过程按照编码的子带顺序进行。

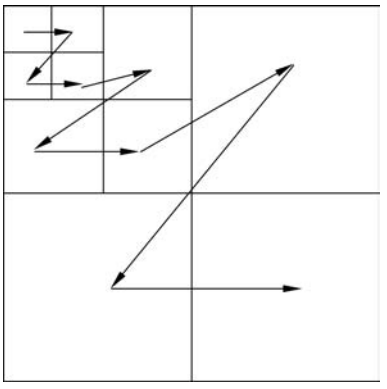

图 5.4 子带扫描次序

2. 逐次逼近的嵌入式编码

为了使得零树表示结构成为一个有效的嵌入式码流,编码的核心采用了逐次逼近的量化方法(Successive-Approximation Quantization,SAQ)。逐次逼近量化与比特平面的有效性编码相联系,连续地应用一系列阈值 $T_0,T_1,\cdots,T_{N-1}$ 来确定比特平面的有效性,其中,阈值序列的选取是 $T_i=T_{i-1}/2$,初始阈值 T_0 的选取使所有的变换系数满足 $|x_{\max}|<2T_0$,这里 $x_{\max}$ 是小波变换系数矩阵中的最大绝对值。SAQ 实际上就是一种按比特平面的编码方式。

在利用 SAQ 量化的编码过程中,依次形成两个表,一个是主表,一个是副表。对于一个给定的门限式 T_i(第一次为 T_0),首先进行一遍主扫描,生成主表。主表包含的是以式为门限的有效值映射,在形成主表的同时,把新出现的有效值幅度也加入到一个幅度表中(幅度表只是编码过程中出现的中间表,并不是编码器输出项)。为了不影响后续更小的门限的有效值映射有效性,将已经发现的有效值位置的系数置为 0(这些情况暂存在幅度表中)。

对于门限 T_i,在进行完一遍主扫描后,紧接着进行副扫描,副扫描是对已发现的有效值进行更细化的表示。若当前门限是 T_0,进行完主扫描后,已发现的有效值的幅度处于 $2T_0$ 和 T_0 之间,如果没有进一步细化表示,解码器仅知道这些值处于 $2T_0$ 和 T_0 之间,一般可能会用 $T_0+T_0/2$ 作为它的重构值。副扫描的目的是进一步细化这些值。用 0 或 1 进一步描述一个值是处于该区间的上半部还是下半部,提高了一倍表示精度。后续过程中,门限 T_i 的情况也类似,不同的是,后续过程中不仅是对本次新发现的有效值进行细化,也对以前的有效值进行细化,而幅度表中有效值的排列是以可分辨出的有效值按从大到小顺序排列。

这些 0,1 串构成副表。

零树表示和 SAQ 结合构成了编码器的工作方式概括如下：首先确定 T_0，进行第一遍主扫描，构成有效值映射，形成主表；然后进行第一遍副扫描，细化有效值的表示。更新门限 $T_1=T_0/2$，进行新一遍主扫描，新的主扫描时，对已经发现有效值的位置不需要扫描，将它们设为 0 以便发现零树，因为门限已减半，会发现一些新的有效值；主扫描结束，进入新一遍副扫描，对原已发现的有效值和本次新发现的有效值进行细化处理；继续取 $T_2=T_1/2$ 作为新门限，这个过程继续进行下去，直到预设条件达到结束。

5.2.3　SPIHT 算法

EZW 算法由于编码时形成多棵零树，因而要多次扫描图像，造成效率很低。而且每一棵树必须在前一棵树形成之后才能形成，所以也很难用并行算法优化；对所有的频域进行等同有效性的编码，不能充分利用小波变换的特点。

人们通过研究分析 EZW 编码算法的不足，提出了多种改进的嵌入零树图像编码算法，SPIHT 算法就是其中之一。SPIHT 算法继承了 EZW 算法的特点，但在两个方面有本质的不同：分割系数的方式和如何传输有效系数的位置信息。SPIHT 算法把对有效系数位置的传输隐含在算法的执行过程之中，并因此可以在允许峰值信噪比下降 0.3～0.6dB 时不采用算术编码，使执行速度更快。SPIHT 算法不同于 EZW 算法利用零树根来表示大量的无效小波系数，它采用树的分割方法，力图使无效系数保持在更大子集中。分割判决用二进制的方式传送给接收端，从而提供了比 EZW 更有效的图编码方法。

1. SPIHT 的概念

图像经过 k 级小波变换后形成了 $3k+1$ 个子带，按其频带从低到高形成一个“空间方向树”结构，树根是最低频子带的节点，用 H 表示所有根节点组成的集合。为了描述 SPIHT 算法，首先定义几个有用的用于划分空间方向树结构的集合，如表 5.1 所示。

表 5.1　用于划分空间方向树结构的集合

符号	含　义
$O(i,j)$	表示位于(i,j)位置的小波变换系数的 $C(i,j)$的子女的坐标集合。在每一个节点，一个系数可能有 4 个子女或没有子女，故 $O(i,j)$的大小为 4 或 0
$D(i,j)$	表示位于(i,j)位置的小波变换系数的所有子孙的坐标集合
$L(i,j)$	表示 $C(i,j)$的所有子孙的坐标集合，但是去掉它的直接子女集合
$S_n(T)$	集合是否有效的标志位

SPIHT 算法的基本的集划分准则简单归纳如下。

(1) 对于所有的$(i,j)\in H$，形成初始划分集合为$\{(i,j)\}$和 $D(i,j)$。

(2) 如果 $D(i,j)$是有效的，它被划分为 $L(i,j)$加上 4 个单元素集合$(k,l)\in O(i,j)$。

(3) 如果 $L(i,j)$是有效的，它被划分为 4 个集合 $D(k,l)$，这里$(k,l)\in O(i,j)$。

SPIHT 算法编码过程分为分类扫描和细化两部分。分类扫描过程根据上述集合划分准则，将空间方向树上的节点分类，过程中使用到 3 个表，并对 3 个表进行动态更新。

(1) 无效集表：LIS(List of Insignificant Sets)，每个记录都是坐标形式(i,j)，它代表

一个集合 $D(i,j)$或 $L(i,j)$。在 LIS 表中,元素 $D(i,j)$称为 A 型,$L(i,j)$称为 B 型。

(2) 无效像素表: LIP(List of Insignificant Pixels),每个记录也都是坐标(i,j)形式,它代表(i,j)位置有一个无效值。

(3) 有效像素表: LSP(List of Significant Pixels),每个记录也都是坐标(i,j)形式,它代表(i,j)位置有一个有效值。

2. SPIHT 编码

首先将一个空间方向树在初始化时输出 $n=\lfloor\log_2(\max\{|C(i,j)|\})\rfloor$,其中,$C(i,j)$为小波系数,将 LSP 置为空表,将坐标$(i,j)\in H$ 加入到 LIP 中,$(i,j)\in H$ 带有子孙的坐标加入到 LIS,并作为 $D(i,j)$类集合。

其次,在分类扫描过程中,先对 LIP 的每个记录进行扫描,若该记录有效,将其移到 LSP,并输出 $C(i,j)$的符号位; 然后对 LIS 的每个记录扫描,若这个记录为 $D(i,j)$类集合且是有效的,则 $D(i,j)$继续分裂为两个集合 $O(i,j)$和 $L(i,j)$,对集合 $O(i,j)$的每个记录分别进行有效性测试,把有效记录移到 LSP 中,将无效记录移到 LIP; 对集合 $L(i,j)$的记录进行有效性测试,若有效,则 $L(i,j)$分裂为 4 个集合,并对这 4 个集合进行判断,如此重复,对每棵树进行分裂和判断直到找出所有重要元素,将它们移到 LSP。

对有效值表 LSP 细化过程,在本次门限 $T=2^n$ 的扫描过程中,新移入 LSP 的值不做细化处理,输出 $C(i,j)$的第 n 个有效位。

令 $n=n-1$,进行下一比特平面扫描分类和细化过程,直到完成编码。

SPIHT 算法描述如下。

第一步: 算法初始化。

得到 $n=\lfloor\log_2(\max\{|C(i,j)|\})\rfloor$。

置 LSP 为空表,将坐标$(i,j)\in H$ 加入到 LIP,$(i,j)\in H$ 中带有子孙的加入到 LIS,并作为 $D(i,j)$类集合。

第二步: 分类扫描过程。

(1) 对 LIP 的每个记录(i,j) 输出 $S_n(i,j)$,如果 $S_n(i,j)=1$,将(i,j)移动 LSP,并输出 $C(i,j)$的符号位。

(2) 对 LIS 的每个记录(i,j)。

① 如果这个记录代表一个 $D(i,j)$类集合,则输出 $S_n(D(i,j))$; 如果 $S_n(D(i,j))=1$,则对每一个$(k,l)\in O(i,j)$输出 $S_n(k,l)$; 如果 $S_n(k,l)=1$,将(k,l)加入到 LSP,并输出 $C(i,j)$的符号位; 如果 $S_n(k,l)=0$,将(k,l)加入到 LIP。如果 $L(i,j)$不为空集合,将(i,j)加入到 LIS 的尾部,并标明它是 $L(i,j)$类型集合,转到②,如果 $L(i,j)$是空集合,将(i,j)从 LIS 中移出。

② 如果这个记录代表一个 $L(i,j)$类集合,则输出 $S_n(L(i,j))$; 如果 $S_n(L(i,j))=1$,则将每个$(k,l)\in O(i,j)$加入到 LIS 的尾部,并标记为 $D(k,l)$类型,从 LIS 中移去(i,j)项。

第三步: 对 LSP 中每一个(i,j)(除了当前这遍扫描产生的之外),输出 $C(i,j)$的第 n 个最高有效值。

第四步: $n=n-1$,返回第二步。

解码过程：编码器将上述扫描过程中输出的码存入文件，更有效的方式是采用算术编码后存入文件。文件中存放的有关信息，如存放的图像尺寸大小、小波变换分层等，在解码器中由这些信息可生成初始的三个表 LIP，LIS，LSP。接下来做与编码操作相同的扫描工作，只不过编码器是判断并输出码，解码器是读入并恢复相应表格的内容。

5.3　静态图像压缩标准

静态图像编码技术的发展和广泛应用也促进了许多有关国际标准的制定，这方面的工作主要是由国际标准化组织（International Standardization Organization，ISO）和国际电信联盟（ITU）负责研究和制定的。目前，由这两个组织制定的有关图像编码的国际标准涵盖了从二值到灰度的图像编码标准，根据各标准所使用的技术不同，可分为以下几类标准。

（1）二值图像压缩编码标准。

（2）基于 DCT 的静态灰度（彩色）图像压缩编码标准。

（3）基于 DWT 的静态灰度（彩色）图像压缩编码标准。

5.3.1　二值图像压缩编码标准

图像分为彩色图像和灰度图像两大类，二值图像是只有黑白两种灰度级的特殊灰度图像，例如文件、工程图、指纹卡片、手写文字、地图、报纸等，它广泛应用于传真业务、文字资料的数字化存储等。二值图像信源编码的目的和灰度图像的编码一样，也是为了减少表示图像所需的比特数。

ITU 和 ISO 于 1993 年联合成立的 JBIG（Joint Bi-Level Image Experts Group）针对三类传真机一维编码标准——G3 标准的缺陷，制定了二值图像压缩的 T.82 国际标准 ISO/IEC 11544，也称为 JBIG 标准。用于传真机的 G3 标准在编码时对不同的黑白游程长度采用统计的修正哈夫曼编码，对像素变化较快的图像来说，压缩效果很差。JBIG 标准采用基于对条件概率连续预测的算术编码来代替修正哈夫曼编码，使得文档和图像的压缩比能提高数倍。同时，JBIG 也有一些不足，如没有被大多数文档系统支持，解压时间较长，处理照片的功能不足等，最重要的是 IBM、AT&T 等公司掌握着 JBIG 的技术专利，限制了该技术的应用。

1. G3 和 G4 压缩编码标准

G3 和 G4 压缩编码标准是由 CCITT 的两个组织（Group3 和 Group4）负责制定的，最初它是为传真应用而设计的，现也被应用于其他方面。G3 和 G4 压缩编码技术的基本思想是：游程编码与静态的哈夫曼编码相结合。编码过程按行扫描像素，记录 0 值和 1 值的游程长度，然后给游程长度编码，并且黑和白的长度分别使用不同的编码。其中，G3 采用一维编码与二维编码结合的技术，每一个 K 行组的最后 $K-1$ 行（$K=2$ 或 4），有选择地用二维编码方式。G4 标准是 G3 标准的简化或改进版本，采用二维压缩编码和固定的哈夫曼编码表，每一个新图像的第一行的参考行是一个虚拟的白行。

下面简单介绍一维编码和二维编码技术的基本编码过程。

1）一维编码

一维编码的基本思想是：按行编码，逐行扫描，编码方式为游程编码加哈夫曼编码。具体的编码过程如下。

(1) 图像首、尾编码方式如下。

① 图像首行：用一个 EOL(000000000001)开始。

② 图像结尾：用连续 6 个 EOL 结束。

(2) 每一行行首、行尾编码方式如下。

① 行首：用一个白游程码开始，如果行首是黑像素，则用零长度的白游程开始。

② 行尾：用一个 EOL 结束。

(3) 图像内部编码方式。

游程长度小于或等于 63 的用哈夫曼编码。

游程长度大于 63 的用组合编码，即大于 63 的长度哈夫曼编码加上小于 63 的余长度哈夫曼编码。

2）二维编码

二维编码的基本思想是：假设相邻两行改变元素位置的情况很多，且上一行改变元素距当前行改变元素的距离小于游程长度，则编码过程可以利用上一行相同改变元素的位置，来为当前行编码，从而降低编码长度。

与二维编码相关的几个定义如图 5.5 所示。

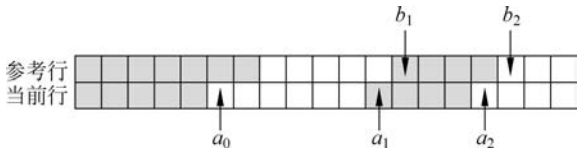

图 5.5　二维编码中的符号定义

(1) 当前行：要编码的扫描行。

(2) 参考行：当前行的前一行。

(3) 改变元素：与前一个像素值不同的像素。

(4) 参考元素：一共有 5 个(当前行 3 个，参考行 2 个)。

① a_0：当前处理行上，与前一个像素值不同的像素。行首元素是本行的第一个 a_0。

② a_1：a_0 右边下一个改变元素。

③ a_2：a_1 右边下一个改变元素。

④ b_1：参考行上在 a_0 右边，且与 a_0 值相反的改变元素。

⑤ b_2：b_1 右边下一个改变元素。

根据以上定义的 5 个参考元素位置，可以有三种情况，即：b_2 在 a_1 的左边，a_1 到 b_1 之间的距离大于 3，a_1 到 b_1 之间的距离小于或等于 3。二维编码则分别对上面三种情况进行编码处理。

(1) 通过编码模式。

① 条件：b_2 在 a_1 的左边，即参考行的两个改变元素都在 a_1 的左边(如图 5.6 所示)。

② 编码：把这种情况定义为通过模式，并编码为 0001(如表 5.2 所示)。

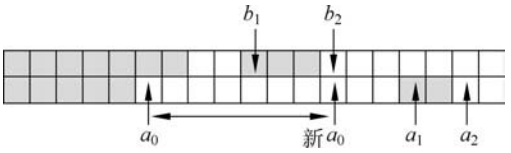

图 5.6　通过编码模式工作原理

③ 操作：把 a_0 移到 b_2 的下面，重新定义 b_1 和 b_2，再进行模式判断。

(2) 水平编码模式。

① 条件：a_1 到 b_1 之间的距离大于 3(如图 5.7 所示)。

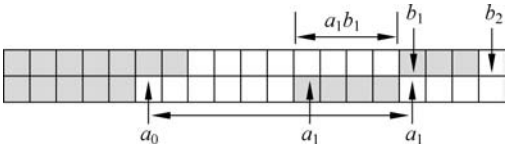

图 5.7　水平编码模式工作原理

② 编码：把这种情况定义为水平模式，并编码为 001(如表 5.2 所示)，此时 a_0 到 a_2 间的编码直接使用一维编码，即 a_0 到 a_2 间的数据编码为 $001+M(a_0a_1)+M(a_1a_2)$，其中，M 代表一维游程编码。

③ 操作：把 a_0 移到 a_2，重新定义其他的 4 个元素，再进行模式判断。

(3) 垂直编码模式。

① 条件：a_1 到 b_1 之间的距离小于或等于 3(如图 5.8 所示)。

② 编码：把这种情况定义为垂直模式，a_1 和 b_1 之间的位置关系有 7 种情况，分别对这 7 种情况进行哈夫曼编码(如表 5.2 所示)。

③ 操作：把 a_0 移到 a_1，重新定义其他 4 个元素，再进行模式判断。

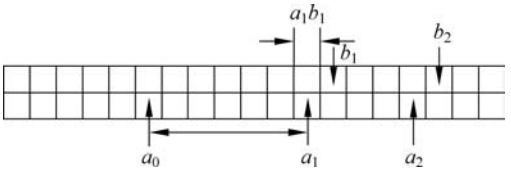

图 5.8　垂直编码模式工作原理

表 5.2　二维编码哈夫曼码表

模　　式	码　　字
Pass	0001
Horizontal	$001+M(a_0a_1)+M(a_1a_2)$
Vertical	
a_1 below b_2	1
a_1 one to the right of b_1	011
a_1 two to the right of b_1	000011
a_1 three to the right of b_1	0000011
a_1 one to the left of b_1	010

续表

模　式	码　字
a_1 two to the left of b_1	000010
a_1 three to the left of b_1	0000010
Extension	0000001××××

2. JBIG 压缩编码标准

JBIG 专家组于 1993 年制定了针对二值图像压缩的 ITU-T 建议 T.82 国际标准 ISO/IEC 11544，称为 JBIG 或 JBIG1。其产生背景是由于传真图像在伪灰度处理条件下，二值序列中短游程的比例上升，用 MH 码、MR 码、MMR 码压缩效率没有达到预计效果，同时对图像存储和传输的要求也不断提高。JBIG 码采用的正是基于对条件概率连续性预测算术编码方法。为了确定条件概率，建议 T.82 描述了基于有限个像素点的上下文参考模板，分别由当前扫描行与上一行以及前两行的像素构成，其中有一个像素可在其默认位置上移动，称为自适应像素。同样是无失真压缩，但由于 JBIG 算法能自适应图像的特征，故与 MH、MR、MMR 编码相比较，对打印字符的扫描图像，压缩比是 MMR 的 1.1～1.5 倍；对计算机生成的打印字符图像，压缩比是 MMR 的 5 倍；而对由二值来表示的“半色调”图像，压缩比是 MMR 的 2～30 倍。MH、MR、MMR 和 JBIG 编码方法均可用于 G3 传真机，MMR 和 JBIG 还可用于 G4 机。

尽管在二值图像的产生过程中可能会有信息损失（譬如通过某个阈值对灰度图像进行二值化），但在随后的编码阶段，现有的二值图像压缩标准都采用完全可逆的熵编码，因而是严格信息保持的。由于大多数二值图像的信宿是人，即解码恢复后的二值图像最终是用人类视觉来感知的，因此，若允许压缩过程引入人眼难以察觉的失真，就不仅有望大幅度地提高数据压缩比，而且还能打破保持型编码在压缩方法选择上对于人们的束缚。事实上，由于很难把压缩损伤和图像噪声区别开来，故采用实际有损、但视觉感知无损或近似无损的二值图像压缩方法也是合理的。这与灰度图像压缩领域人们已经形成的共识完全类似，因此，JBIG 专家组又制定了一个有损二值图像压缩编码标准 JBIG2。

5.3.2 JPEG 压缩编码标准

JPEG 是 Joint Photographic Experts Group（联合图像专家组）的缩写，文件扩展名为 .jpg 或 .jpeg，是最常用的图像文件格式，由一个软件开发联合会组织制定，是一种有损压缩格式，能够将图像压缩在很小的存储空间，图像中重复或不重要的资料会被丢失，因此容易造成图像数据的损伤。尤其是使用过高的压缩比例，将使最终解压缩后恢复的图像质量明显降低，如果追求高品质图像，不宜采用过高压缩比例。

JPEG 包括三种算法：基本系统（Baseline System）、扩展系统（Extended System）和无失真系统。其中，基本系统基于 DCT 变换和可变长编码压缩技术，在保证图像还原质量的前提下，能提供高达 100∶1 的压缩比，但由于编码过程中有失真，故重建图像不能精确再现原始图像，其失真度同压缩比直接相关，所有的 JPEG 编码器和解码器都支持基本系统。

JPEG 的压缩模式有以下几种：①顺序式编码（Sequential Encoding），依次将图像由左

到右、由上到下顺序处理；②递增式编码(Progressive Encoding)，当图像传输的时间较长时，可将图像分为数次处理，以从模糊到清晰的方式来传送图像(效果类似 GIF 在网络上的传输)；③无有损编码(Lossless Encoding)；④阶梯式编码(Hierarchical Encoding)。

最常用的 JPEG 编码是基于 DCT 的顺序型模式，又称为基本系统。下面将针对这种系统讲述其编码过程。JPEG 编/解码流程如图 5.9 和图 5.10 所示。

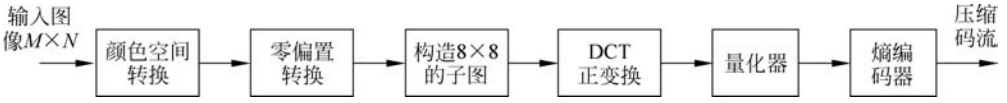

图 5.9　JPEG 编码器流程

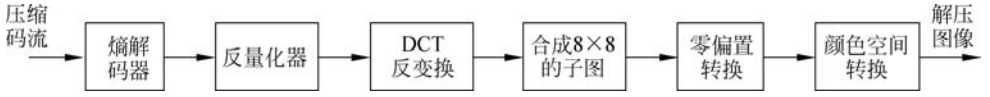

图 5.10　JPEG 解码器流程

从图 5.9 中可以看出，JPEG 编码过程包含以下几个主要步骤。

(1) 进行颜色空间转换。JPEG 采用的是 YC_bC_r 颜色模型，首先要从 RGB 空间转换到 YC_bC_r。

(2) 进行零偏置转换。对于灰度级是 2^n 的像素，减去 2^{n-1}，替换像素本身，对于 $n=8$ 的灰度图像，即将 0～255 的值域减去 128，转换位值为 128～127。这样像素的绝对值出现 3 位十进制的概率大大减少，有利于后面的熵编码。

(3) 块准备。块准备将一帧图像分成 8×8 的数据块。假设一个彩色图像由 3 种分量——光亮度 Y 和两个色差 U，V 表示，图像的大小为 480 行，每一行有 640 个像素。如果假设色度分解为 4∶1∶1，则亮度分量就是一个 640×480 的数值矩阵，色差分量是一个 320×240 的数值矩阵，色差是一个 320×240 的数值矩阵。为了满足 DCT 过程的要求，块准备必须画出 4800 个亮块和两份 1200 个色差块，共计 7200 个数据块。同时将原始图像的采样数据从无符号整数变成有符号整数。即：若采样精度为 P 位，采样数据范围为 $[0,2^{P-1}]$，则变成范围$[-2^{P-1},2^{P-1}-1]$，以此作为 DCT 正变换的输入。在解码的输出端经 DCT 反变换后，得到一系列 8×8 的图像数据块，需将其数值范围由$[-2^{P-1},2^{P-1}-1]$再变回到$[0,2^{P-1}]$的无符号整数，才能重构图像。

(4) DCT 变换。每个初始块由 64 个表示样本信号特定分量的振幅值组成，该振幅是一个二维的空间坐标的函数，可用 $a=f(\boldsymbol{x},\boldsymbol{y})$表示，其中，$\boldsymbol{x}$，$\boldsymbol{y}$ 是两个二维空间向量。在经过离散余弦变换之后，该函数变为 $c=g(F_x,F_y)$，其中，F_x 和 F_y 分别是各个方向空间频率，结果是一个 64 个数值的方阵，每一个值表示一个 DCT 系数，也就是一个特定的频率值，而不再是信号在采样点(x,y)的振幅。这样，经过 8×8 DCT 正变换，8×8 的采样值块变换成 8×8 的 DCT 系数块。DCT 逆变换能把 64 个 DCT 系数重建为 64 点的图像。但由于计算过程中的误差及系数的量化，这 64 点图像是不能完全恢复的。

(5) 量化。为了达到压缩的目的，DCT 系数需做量化处理。例如，利用人眼的视觉特性，对在图像中占有较大能量的低频成分，赋予较小的量化间隔和较少的比特表示，以获得较高的压缩比。JPEG 的量化采用线性均匀量化器，量化公式为：

$$Cq(u,v)=\text{Integer}(\text{Round}(C(u,c)/Q(u,v)) \tag{5.1}$$

其中，$Q(u,v)$是量化器步长，它是量化表的元素，量化元素随 DCT 系数的不同和彩色分量的不同有不同的值。量化表的大小也为 8×8，和 64 个变换系数一一对应。在 JPEG 算法中，对于 8×8 的亮度信息和色度信息，分别给出了默认的量化表。这个量化表是在实验的基础上，结合人眼的视觉特性而获得的。

(6) DCT 直流值和 AC 交流系数的编码。DC 系数差分编码与 AC 系数游程编码，64 个变换系数中，DC 系数处于左上角，它实际上等于 64 个图像采样值的平均值。相邻的 8×8 子块之间的 DC 系数有较强的相关性。JPEG 对于量化后的 DC 系数采用差分编码，二相邻块的 DC 系数的差值为 $DC_i-(DC_i-1)$。其余 63 个 AC 系数量化后通常出现较多零值，JPEG 算法采用游程编码，并建议在 8×8 矩阵中按照 Z 形的次序进行，可增加零的连续次数。

系数编码后都采用统一的格式表示，包含两个符号的内容：第一个符号占 1B，对于 DC 系数而言，它的高 4 位总为零；对于 AC 系数而言，它表示到下一个非零系数前所包含的连续为零的系数个数。第一个字节的低 4 位表示 DC 差值的幅值编码所需的比特数，或表示 AC 系数中下一个非零幅值编码所需的比特数。第二个符号字节表示 DC 差值的幅值，或下一个非零 AC 系数的幅值。还需要指出，由于第一个字节中只有 4 位表示游程长度，最大值为 15。当游程长度大于 15 时，可以插入一个或多个(10)字节，直至剩下的 AC 系数零的个数小于 15 为止。因此，63 个 AC 系数表示为由两个符号对组成的序列，其中也可能插入 10B，块结束字节以全零表示。

(7) 熵编码。对于上面给出的码序列，再进行统计特性的熵编码。这仅对于序列中每个符号对中的第一个字节进行，第二个幅值字节不作编码，仍然直接传送。JPEG 建议使用两种熵编码方法：哈夫曼编码和自适应二进制算术编码。对于哈夫曼编码，JPEG 提供了针对 DC 系数、AC 系数使用的哈夫曼表(包括对于图像的亮度值与色度值两种情况)，在编码和解码时使用。JPEG 解码器能够同时存储最多 4 套不同的熵编码表。

JPEG 的解码过程是上述过程的逆过程。

5.3.3 JPEG2000 压缩编码标准

随着多媒体应用领域的快速增长，传统的 JPEG 压缩技术已经无法满足人们对数字化多媒体图像资料的要求，网上 JPEG 图像只能一行一行地下载，直到全部下载完毕，才可以看到整个图像，如果只对图像的局部感兴趣，也只能将整个图片下载来再处理。JPEG 格式图像文件体积仍然较大；JPEG 格式属于有损压缩，当被压缩图像上有大片近似颜色时，会出现马赛克现象；同样由于有损压缩的原因，许多对图像质量要求较高的应用，JPEG 无法胜任。

JPEG2000 由 ISO 和 IEC(国际电工协会)JTC1 SC29 标准化小组命名为“ISO 15444”，制定始于 1997 年 3 月，但因为无法很快确定算法，直到 2000 年 3 月，规定基本编码系统的最终协议草案才出台。JPEG2000 采用改进的压缩技术来提供更高的解像度，其伸缩能力可以为一个文件提供从无损到有损多种画质和解像选择。JPEG2000 被认为是互联网和无线接入应用的理想影像编码解决方案。在压缩率相同的情况下，JPEG2000 的信噪比将比 JPEG 提高 30%左右。JPEG2000 拥有 5 种层次的编码形式：彩色静态画面采用的 JPEG

编码、二值图像采用 JBIG、低压缩率图像采用 JPEGLS 等，成为应对各种图像的通用编码方式。在编码算法上，JPEG2000 采用离散小波变换核 bit plain 算术编码（MQ coder）。此外，JPEG2000 还能根据用户的线路速度以及利用方式，以不同的分辨率以及压缩率发送图像。

1. JPEG2000 标准的组织结构

JPEG2000 标准可分为以下几个部分。

（1）PART1：JPEG2000 图像编码系统，是 JPEG2000 标准的核心系统。

（2）PART2：扩展系统，在核心系统上添加了一些功能。

（3）PART3：运动 JPEG2000，主要针对运动图像提出的解决方案。

（4）PART4：兼容性。

（5）PART5：定义参考软件。目前有两个：基于 Java 的 JJ2000，基于 C 的 Jasper。

（6）PART6：定义复合文件格式，主要针对印刷和传真应用。

（7）PART7：学术报告，介绍实现 PART1 所需要的最小支持环境（已取消）。

（8）PART8：JPSEC 与 JPEG2000 安全应用有关。

（9）PART9：JPP 为分配 JPEG2000 应用定义了一套高级网络协议。

（10）PART10：JP 3D 与三维数据和浮点数据压缩有关。

（11）PART11：JPWL 使用 JPEG2000 处理无线应用。

（12）PART12：是对第三部分的补充。

（13）PART13：2004 年 3 月建立，主要是对 JPEG2000 编码器进行标准化。

2. JPEG2000 的特性

推进 JPEG2000 发展的关键因素并不是它比 JPEG 有高的压缩比，而是它提供了一种具有丰富特征的新编码系统和一种全新的图像再现形式。为了具体理解这一点，首先看一下 JPEG2000 PART 1 的关键特征。

（1）优良的压缩特性：高比特率时，人工痕迹微乎其微。JPEG2000 的压缩率比 JPEG 平均提高约 20%。在低比特率时，JPEG2000 比 JPEG 有明显的提高。压缩增益高于 JPEG，主要采用 DWT 和最新编码算法。

（2）多分辨率：JPEG2000 提供无缝图像压缩，每一个样本成分可以从 1b 到 16b。

（3）按像素精度和分辨率渐进传输（渐进解码和信噪比可伸缩）：JPEG2000 可以根据 SNR、分辨率、空间位置和图像分量的不同要求提供渐进的码率组织。

（4）无损和有损压缩：JPEG2000 仅通过一种整数小波变换的压缩提供有损和无损两种压缩方式。

（5）随机码流访问和处理（感兴趣区域）：JPEG2000 码流提供一些机制支持随机访问或感兴趣区域的访问。

（6）比特误差的鲁棒性：JPEG2000 在有噪声的通信信道如无线信道中引入误码的鲁棒性，它是通过包含再同步标记来实现的，编码与独立的小块有关，能够检查每一块中隐藏的错误。

（7）连续构建功能：JPEG2000 允许图像从顶到底连续编码，不需要缓存。

(8) 灵活的文件格式：JP2 和 JPX 文件格式允许处理彩色空间信息、元数据和一些网络互动应用。

3. JPEG2000 的基本原理

JPEG2000 的编码和解码器框图如图 5.11 所示。编码过程主要分为以下几个过程：预处理、核心处理和位流组织。预处理部分包括对图像分片、直流电平位移和分量变换。核心处理部分由离散小波变换、量化和编码组成。位流组织部分则包括区域划分、码块、层和包的组织。解码器是编码器的反过程，首先对码流进行解包和熵解码，然后反向量化和离散小波变换，对反变换的结果进行后期处理合成，得到重构图像数据。

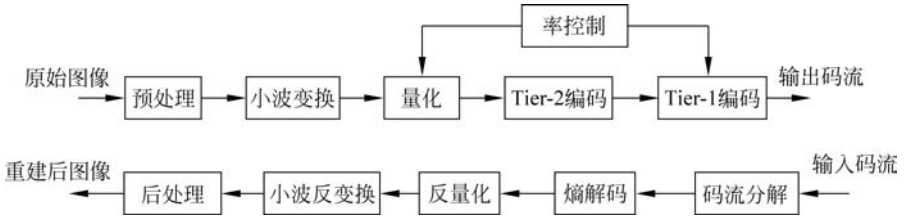

图 5.11 JPEG2000 的编码和解码器框图

1) 数据预处理

数据预处理一般包括三种操作：区域划分、DC 电平位移、分量变换。

(1) 图像分片。

分片指的是把源图像分割成相互不重叠的矩形块——图像片，每一个图像片作为一个独立的图像进行压缩编码。编码中的所有操作都是针对图像片进行的。图像片是进行变换和编解码的基本单元。图像的分片降低了对存储空间的要求，并且由于它们重构时也是独立进行的，所以可以用来对图像特定区域而不是整幅图像进行解码。当然，图像分片会影响图像质量。比较小的图像片会比大图像片产生更大的失真。图像分片在低比特率表示图像的时候所造成的图像失真会更加严重。

(2) 直流电平位移。

在对每一图像片进行正向离散小波变换之前，都要进行直流电平位移。目的是在解码时，能否从有符号的数值中正确恢复重构的无符号样本值。直流电平位移是对仅有无符号数组成的图像片的像素进行的，电平位移并不影响图像的质量。解码端，在离散小波反变换之后，对重构的图像进行反向直流电平位移。

(3) 分量变换。

JPEG2000 支持多分量图像。不同的分量不需要相同的比特深度，也不需要都是无符号或有符号数。对于可恢复系统，唯一的要求就是每一个输出分量图像的比特深度必须跟相应输入分离图像的比特深度保持一致。

2) 离散小波变换

不同于传统的 DCT 变换，小波变换具有对信号进行多分辨率分析和反映信号局部特征的特点。通过对图像进行离散小波变换，得到小波系数，而分解的级数视具体情况而定，小波系数图像由几种子带系数图像组成，这些子带系数图像描述的是图像水平和垂直方向

的空间频率特性。不同子带的小波系数反映图像不同空间分辨率特性。通过多级小波分解，小波系数既能表示图像中局部区域的高频信息，也能表示图像中的低频信息。这样，即使在低比特率的情况下，也能保持较多的图像细节，另外，下一级分解得到的系数所表示图像在水平和垂直方向的分辨率，就可以得到不同空间分辨率的图像。

小波变换因其具有这种优点被 JPEG2000 标准所采用。在编码系统中，对每个图像进行 Mallat 塔式小波分解。经过大量的测试，JPEG2000 选用两种小波滤波器：LeGall 5/3 滤波器和 Daubechies 9/7 滤波器。前者可用于有损或无损图像压缩，后者只能用于有损压缩。

在 JPEG2000 标准中，小波滤波器有两种实现模式：基于卷积和基于提升机制的。具体实现时，对图像边缘都要进行周期对称延伸，可以防止滤波器对图像边缘操作时产生失真。另外，为了减小变换时所需空间开销，标准中还应用了基于行的小波变换技术。

3）量化

由于人类视觉系统对图像的分辨率要求有一定的局限，通过适当的量化减小变换系数的精度，可在不影响图像主观质量的前提下，达到图像压缩的目的。量化的关键是根据变换后图像的特征、重构图像质量要求等因素设计合理的量化步长。量化操作是有损的，会产生量化误差。不过一种情况除外，那就是量化步长是 1，并且小波系数都是整数，利用可恢复整数 5/3 的小波滤波器进行小波变换得到的结果就符合这种情况。

在 JPEG2000 标准中，每一个子带可以有不同的量化步长。但在一个子带中只有一个量化步长。量化以后，每一个小波系数由两部分来表示：符号和幅值。对于无损压缩，量化步长必须是 1。

4）熵编码

图像经过小波变换、量化后，在一定程度上减少了空间和频域上的冗余度，但是这些数据在统计意义上还存在一定的相关性，为此采用熵编码可以消除数据间的统计相关性。JPEG2000 使用 EBCOT 编码器对每个编码块进行熵编码。EBCOT 的编码过程可分为以下两个步骤。

（1）嵌入式码块编码。

经过小波变换和量化，图像分量矩阵成为整数系数的子带矩阵，每个子带又被划分为大小相同的矩形码块，而每个码块将被独立编码，每个码块又可分成位平面，即比特层，从最高有效位平面开始逐平面编码直到最低位平面。对位平面编码时，为了获得细化的嵌入式码块位流，每个位平面又进一步分成子位平面，称为编码通道。位平面上的每个系数位必须而且只能在其中一个通道上进行编码。这三个通道依次为：重要性传播通道、幅度细化通道、清除通道。在这三个编码通道上分别进行 4 种编码操作：有效性编码、符号编码、幅度细化编码和清除编码。在清除编码中，根据适当的条件进行游程编码，以减少算术编码时二进制符号个数。

（2）分层组织码块的嵌入式压缩位流。

在第二步编码算法中，采用 PCRD 率失真优化算法思想，对所有码块的嵌入式压缩位流进行适当的截取，分层组织，形成整个图像具有质量可分级的压缩码流。

为了更好地应用 JPEG2000 压缩码流的功能，JPEG2000 标准规定了存放压缩位流和所需参数的格式，把压缩码流以包为单元，进行组织，形成最终的压缩码流。

4. JPEG2000 中的小波提升算法

二维离散小波变换最有效的实现方法之一是采用 Mallat 的塔式分解方法，在图像的水平和垂直方向交替采用低通和高通滤波。这种传统的基于卷积的离散小波变换计算量大，对存储空间的要求高，提升小波的出现有效地解决了这一问题。提升算法相对于 Mallat 算法，是一种更为快速有效的小波变换实现方法，它不依赖于傅里叶变换，继承了第一代小波的多分辨率特性，在空域完成了对双正交小波滤波器的构造。Daubechies 已经证明，任何离散小波变换或具有有限长滤波器的两阶滤波变换都可以被分解成为一系列简单的提升步骤，所有能够用 Mallat 算法实现的小波，都可以用提升算法来实现。

提升算法使用了基本的多项式插补来获取信号的高频分量，之后通过构建尺度函数来获取信号的低频分量。提升算法的基本思想是通过一个基本小波，逐步构建出一个具有更加良好性质的新小波。一个规范的提升算法有 3 个步骤：分解；预测；更新。提升算法的分解步骤如图 5.12 所示。

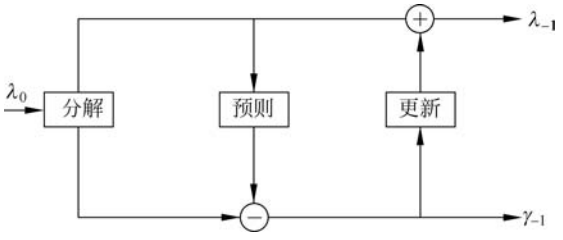

图 5.12　提升算法的分解步骤

设有数据列 λ_0，由于数据之间有某种相关性，可以将它用更为紧凑的格式来表示。也就是说，寻找原数据列的一个子集，使它能够表示原始信号所包含的信息。下面按照提升算法的 3 个步骤分为 3 个部分来进行讨论。

(1) 分解：将数据列 λ_0 分解成为两个小的子集 λ_{-1} 和 γ_{-1}。假定相邻的数据间有最大的相关性(在实际应用中也往往是这种情况)，按照数据的奇偶序号对数据列进行间隔采样，即：

$$\lambda_{-1,k} = \lambda_{0,2k}, \quad k \in Z$$
$$\gamma_{-1,k} = \lambda_{0,2k+1}, \quad k \in Z \tag{5.2}$$

这种情况就是前面提到的基本小波。用基本小波对信号进行分解与重构如图 5.13 所示。

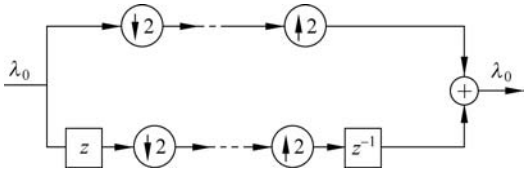

图 5.13　信号小波分解与重构示意图

（2）预测：设一个与数据无关的预测函数 P，使得

$$\gamma_{-1}=P(\lambda_{-1}) \tag{5.3}$$

那么，用 λ_{-1} 来表示原始的数据，称 γ_{-1} 为小波系数。用 γ_{-1} 和它的预测值之间的差值来代替 γ_{-1}。如果使用合理的预测，那么差值将包含比原来 γ_{-1} 少得多的信息。即：

$$\gamma_{-1}=\gamma_{-1}-P(\lambda_{-1}) \tag{5.4}$$

则此时的小波系数 γ_{-1} 表明了由预测函数 P 引入的误差。

此时，可以用较小的数据序列 λ_{-1} 和小波系数 γ_{-1} 来代替原始的数据。如果有一个好的预测，那么两个子集$\{\lambda_{-1},\gamma_{-1}\}$将产生比原来的序列 λ_0 更为紧凑的表示。对这个算法进行周期重复，将 λ_{-1} 抽取成两个序列 λ_{-2} 和 γ_{-2}，然后用 γ_{-2} 和 $P(\lambda_{-2})$之间的差值来代替 γ_{-2}，这样经过 n 步，就可以用小波表示$\{\lambda_{-n},\gamma_{-n},\cdots,\gamma_{-1}\}$取代原来的数据序列。假使小波系数是由于某种相关性的预测模型得到的，那么$\{\lambda_{-n},\gamma_{-n},\cdots,\gamma_{-1}\}$将给出一个比原序列更为紧凑的表示。

（3）更新思想是通过寻找 λ_{-1}，从而使得对于某一个度量标准 Q，例如平均值，λ_{-1} 和 λ_0 具有相同的值：

$$Q(\lambda_{-1})=Q(\lambda_0) \tag{5.5}$$

考虑使用已经计算出的小波 γ_{-1} 来更新 λ_{-1}，从而使 λ_{-1} 保持上述性质。再构造一个操作 U 并做如下计算更新 λ_{-1}：

$$\lambda_{-1}=\lambda_{-1}+U(\gamma_{-1}) \tag{5.6}$$

重复这个算法，从而得到了如式 5.7 所示的小波变换公式：

$$\begin{cases}\{\lambda_j,\gamma_j\}=\mathrm{Split}(\lambda_{j+1})\\ \gamma_{j^-}=P(\lambda_j)\\ \lambda_{j^+}=U(\gamma_j)\end{cases},\quad j=-1,\cdots,-n \tag{5.7}$$

对于提升算法，如果得到了正变换，就可以立即得到反变换，需要做的只是改变加减的符号。这是提升算法的一个非常优良的特性。小波的反变换如式 5.8 所示。

$$\begin{cases}\lambda_{j+1,}=\mathrm{Join}(\lambda,\gamma_j)\\ \lambda_{j^-}=U(\gamma_j)\\ \gamma_{j^+}=P(\lambda_j)\end{cases},\quad j=-n,\cdots,-1 \tag{5.8}$$

由此，得到了提升算法有效的实现步骤，如图 5.14 和图 5.15 所示，即任何有限长滤波器都可以用基本小波通过有限数目预测和更新步骤得到。

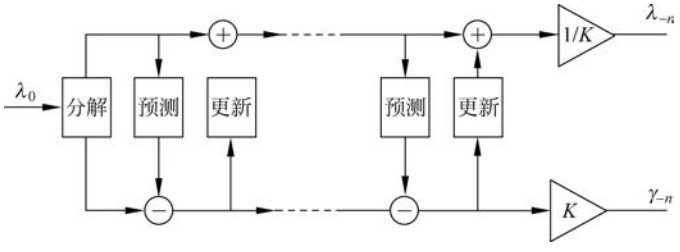

图 5.14　小波正变换的提升实现

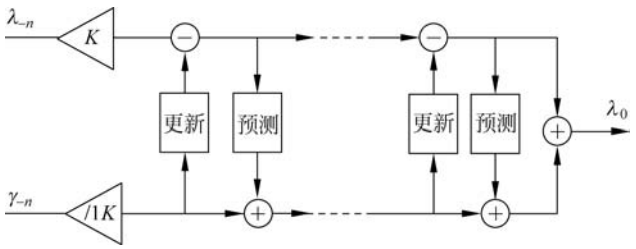

图 5.15 小波反变换的提升实现

5. EBCOT 编码算法

最佳截断嵌入码块编码(Embedded Block Coding with Optimized Truncation, EBCOT)是 David Taubman 在 2000 年发表的一种新的熵编码算法。JPEG2000 的小波系数量化编码采用 EBCOT 编码,EBCOT 量化编码是 JPEG2000 标准的核心。EZW 或 SPIHT 算法更关注子带间系数之间的相关性,而很少利用子带内系数的相关性。EBCOT 算法则更注重子带内相邻系数之间的相关性,同时它通过率失真优化达到很高的编码效率。EBCOT 编码中各小波子带划分为更小的码块,以码块为单位独立做编码。EBCOT 算法采用了内嵌块部分比特平面编码和率失真压缩技术,对内嵌比特平面编码产生的码流按贡献分层,以获得分辨率渐近性和 SNR 渐近特性。在比特平面编码时,不同的码块产生的比特流长度不同,它们对恢复图像质量的贡献不同。利用率失真最优原则对每一码块产生的码流按照对恢复图像质量的贡献分层截取,最后按逐层、逐块的顺序输出码流。比较而言,EBCOT 的压缩性能比 EZW/SPIHT 略有提升,解决了 EZW/SPIHT 误码影响整幅图像的问题,但其复杂度也有所提高。

EBCOT 编码分为两部分:第 1 部分 Tierl,将每个子带划分为独立的编码块,然后对每个编码块独立进行嵌入式编码扫描,每个编码块的比特层编码,最后对编码扫描结果进行 MQ 算术编码,得到嵌入式码流;第 2 部分 Tier2,根据输出码率的要求,组合每个编码块的嵌入式码流,对所有编码块的编码流进行优化截断排序、打包等处理,得到 JPEG2000 的码流。EBCOT 嵌入式比特层编码主要包括嵌入式比特层编码、优化截断位流排序和 MQ 算术熵编码。MQ(Multiple Quantization,多路量化)编码是一种用于二进制数据的自适应算术编码。先将小波变换后的各个子带系数分成小的码块,然后将这些码块中的系数按照一定的量化步长进行量化,量化后的结果送入 EBCOT 编码器进行编码。

在 Tierl 中,将每个码块中的小波系数分解成位平面,即一个个的比特层。编码从码块的非零最高有效位平面(MSB 平面)逐个平面编码直到最低位平面(LSB 平面)。对每一层位平面上的比特进行 3 次扫描,根据一定的规则将该位平面上的比特分在 3 个不同的编码通道上。接着对这 3 次扫描过程的结果依次进行位平面的条件编码,条件编码后产生的上下文信息和编码码流再一同送入 MQ 算术编码器,进行算术编码。

经过 Tierl 编码,输出的是一系列经过编码的通道。在有损编码情况下,编码器可以只在码流中保留一部分重要的编码通道,而舍弃其他的编码通道,这是除了量化以外的另一个信息损失的主要原因。

在 Tier2 中,实质就是让 Tierl 输出的一系列编码通道信息形成数据单元——包,这些

包随后输出给最终码流，这样就完成了编码部分。

1）Tier1 层

优化截断的嵌入式编码 EBCOT 在 Tierl 中通过位平面编码器和自适应算术编码器，实现了数据的压缩。如图 5.16 所示编码块数据经过位平面编码器后输出相应的上下文和判决在经过自适应算术编码后计算生成相应的压缩数据。

图 5.16　Tier1 编码器结构

（1）位平面编码算法。

位平面编码器是对小波变换后子带信息分解为多个位平面，并对各个位平面进行 3 通道的编码并输出上下文和数据位。它是根据编码样本点及其周围相邻样本点的重要性状态信息，使用 4 种编码算法对各个位平面的样本点进行编码的。在编码前需要对子带进行划分，把子带划分为大小相同的码块。

经过小波变换后，系数分布在不同的子带内，为了便于对子带内数据进行编码，需对各个子带进行一定标准的划分，划分后的单元称为“码块”，对子带划分的标准如下。

① 子带内部码块的大小需相同（边缘的码块除外）。

② 码块的边缘不能超出子带的边界。

③ 码块的大小为 2 的整数次幂，JPEG2000 推荐的大小一般为 32×32 或者 64×64。

对码块内的数据进行编码时，每个系数值都是由符号位和幅度值组成的。码块中的数据值被分成若干位平面，最高的位平面为符号位平面，然后剩下的都是幅度位平面。在对这些幅度位平面编码时，从非零的最高位平面开始编码，一直到编码完所有的位平面后结束对该码块的编码。码块内的各个位平面又被划分为不同的条带，即位平面中数据值从上到下每四个作为一列，从左侧第一列开始到最后一列结束组成一个条带。对位平面进行扫描编码时，按照从上到下的顺序依次编码各个条带直到所有条带都编码结束后，再进行下一个位平面的编码。而在条带内进行扫描编码时，应从第一列的第一个数据开始编码直到第四个数据结束后，再从第二列开始下一列的编码直到条带的最后一列。在编码的过程中除了对第一个幅度位平面进行一次扫描，剩余的幅度位平面都将进行 3 次扫描。

在进行位平面编码时，除了最高位平面只进行清除通道的编码外，其余的位平面都必须依次进行 3 个通道的扫描编码：重要性通道编码、幅度细化通道编码和清除通道编码。位平面的编码过程为：首先判断所扫描编码的位平面是否是最高位平面，如果是就直接进入清除通道的编码；如果不是对其进行重要通道的扫描编码，接着重复扫描系数并判断属于幅度细化通道的样本点，对其进行相应的编码，最后对于没有在前两个通道编码的系数进行清除通道的编码。

三个通道编码原理如下。

① 重要性传播通道：处理的是不重要的且有非零上下文的系数，包括重要性编码算子和符号编码算子。

② 幅度细化通道：处理的是比较重要的系数，包括幅度细化编码算子。

③ 清理通道：处理的是上面两个过程未处理的系数，即不重要的且有零上下文的系数，清理通道也处理一列 4 个系数的游程编码，包括清理编码算子。

在对通道进行编码时，采用 4 种编码算法对扫描样本点进行编码，分别是零编码(Zero Code，ZC)、符号编码(Sign Code，SC)、幅度细化编码(Magnitude Code，MRC)和游程编码(Run Length Code，RLC)。

EBCOT 算法中通过对同一个位平面中样本点及其 8 个相邻样本的重要性状态信息来判别编码。如图 5.17 所示，X 为编码样本点，H_0 和 H_1 为水平领域，V_0 和 V_1 为垂直领域，$D_0 \sim D_3$ 为对角邻域值。编码样本点的 8 个领域可以构造出 256 种上下文，由于存在相似的概率统计等特点，JPEG2000 选取合并为 19 种上下文编码模型，来描述其编码过程。

D_0	V_0	D_1
H_0	X	H_1
D_3	V_1	D_2

图 5.17 样本点 X 的领域示意图

① 零编码(ZC)。

在重要性传播编码通道和清除编码通道的编码过程中，将会用到零编码算法。零编码生成 9 种内容模型。这些内容模型是根据待编码数据比特周围的 8 个相邻数据重要性情况生成的。一般说来，四周 8 个比特数据的重要性可以产生 256 个内容模型，但许多内容模型具有相同的概率估计，所以被合并为 9 个内容模型。上下文数据依赖于(在给定小波分解级数上)编码子块位于哪个子带。生成的上下文内容模型(Context)仅便于识别不同的上下文内容模型，即使用的识别符根据实现而定。生成的上下文内容模型由当前重要性比特周围(水平方向、垂直方向、对角线方向)的状态值的总和确定。具体编码规则如表 5.3 所示，表中“x”表示任意值。编码的最终目标是判断系数的主要状态，如果系数重要输出“1”，如果系数不重要输出“0”。

表 5.3 零编码编码规则表

LL 和 LH 子带			HL 子带			HH 子带		上下文标号
$\sum H$	$\sum V$	$\sum D$	$\sum H$	$\sum V$	$\sum D$	$\sum(H+V)$	$\sum D$	
2	x	x	x	2	x	x	$\geqslant 3$	8
1	$\geqslant 1$	x	$\geqslant 1$	1	x	$\geqslant 1$	2	7
1	0	$\geqslant 1$	0	1	$\geqslant 1$	0	2	6
1	0	0	0	1	0	$\geqslant 2$	1	5
0	2	x	2	0	x	1	1	4
0	1	x	1	0	x	0	1	3
0	0	$\geqslant 2$	0	0	$\geqslant 2$	$\geqslant 2$	0	2
0	0	1	0	0	1	1	0	0
0	0	0	0	0	0	0	0	0

② 符号编码(SC)。

符号编码主要用在重要性传播编码通道和清除编码通道的编码过程中。该操作编码系数的正负符号，出现在 ZC 或 RLC 编码后。邻居系数的符号状态与当前要编码系数符号有着密切的关系，可以利用这个关系提高符号编码的效率。在符号编码中引入了 5 个上下文状态，研究表明已编码符号信息与直接相邻的 4 个系数符号状态最相关，每个邻居系数可

能的符号状态有正、负以及不重要三种状态，这样邻居系数符号状态总共有 3^4 种，除去一些对称状态，进一步把状态减少到 5 种。具体编码规则如表 5.4 所示，其中，“1”表示在垂直或者水平方向的相邻两个数据都为重要并且符号都为正，或者只有一个是重要的情况；“0”表示两个方向的相邻数据中两个数据都不重要或者都为重要但是具有不同的符号；－1 表示的情况和 1 的情况相反。

表 5.4　符号位编码算法规则表

H	V	X	上下文	H	V	X	上下文
1	1	0	13	0	－1	1	10
1	0	0	12	－1	1	1	11
1	1	0	11	－1	0	1	12
0	1	0	10	－1	－1	1	13
0	0	0	9				

符号编码的第一步计算对应的上下文内容模型；第二步计算要输出的 0、1 二值符号 symbol。注意符号编码输出的 0、1 符号并不是真正的系数符号，而是与异或比特进行异或运算后的值，SC 操作输出的 symbol 计算公式为：symbol＝sign⊕X，系数为正 sign＝0，为负 sign＝1。

③ 幅度细化编码(MR)。

幅度细化编码主要用于幅度细化编码通道的编码过程中，该操作完成对系数幅值的细化。实验分析表明，系数幅值比特位与已编码的该系数和邻居系数的幅值位没有很强的相关性，所以该编码仅使用 3 种上下文。具体编码规则如表 5.5 所示。可以看出，幅度细化编码除了和编码数据相邻的水平和垂直数据的重要性有关之外，它还和数据是否是第一次被“幅度细化”编码有关，不同的内容模型还与在当前重要性状态变量周围的水平、垂直、对角线状态值总和有关。

表 5.5　幅度精炼编码算法规则表

是否为第一次“幅度精练”编码	$H+V+D$	上下文
否	x	16
是	≥1	15
是	0	14

④ 游程编码(RLC)。

游程编码算法应用在清除编码通道的编码中，它用到了两种内容模型。仅当一个编码列(4 个比特数据)的所有相邻数据都不重要时，开始进行游程编码处理。这时，如果一列中的 4 个数据也为不重要数据，则统一编码为一个内容模型(CX＝17)和编码数据 symbol＝0；如果 4 个数据中至少有一个变为重要，则首先将其表示一个内容模型(CX＝17)，其对应编码数据 symbol＝1。编码 4 个数据中的第一个重要数据的位置信息(00～11)，这时采用一个“游程编码”的内容模型进行编码(CX＝18)，然后编码第一个重要数据的符号位，随后按照零编码算法进行数据的编码处理。

(2) MQ 编码器。

JPEG2000 所用的 MQ 算术编码器是一种特殊的二进制算术编码器,属于自适应二进制算术编码器。所谓自适应算术编码是指编码系统用来划分区间的当前符号概率估计,是可以根据已经传输和编码的信息串进行调整的。一个自适应二进制算术编码需要使用统计模型,以便用来选择编码区间划分时所用的条件概率估计。MQ 算术编码器概率估计依赖于编码的某些"特征"(也就是上面所说的"上下文"),故又称为基于上下文的二进制算术编码。

递归编码间隔细分是自适应算术编码的基础,在编码的过程中,输入的二进制数据 0、1 被分为大概率符号(MPS)和小概率符号(LPS),在每次区间划分之前都要判定 MPS 的含义,也就是要确定输入的"0"或"1"symbol 符号为 MPS 还是 LPS,然后再进行区间划分。概率区间被划分为 MPS 的编码间隔和 LPS 的编码间隔,其长度由每个信源符号的概率决定。自适应算术编码器采用一个可以对原始数据快速适应的概率自动估计模型表,如表 5.6 所示,共有 47 项。表中的一行就是一个状态,每个状态中除含有小概率符号(LPS)的概率 Q_e 外,还含有下一个状态的索引 NMPS 和 NLPS,以及是否需要交换 MPS 和 LPS 所代表符号的标志 SWITCH。

表 5.6 概率估值表

I	Q_e	NMPS	NLPS	SWITCH
0	0.503 937	1	1	1
1	0.304 715	2	6	0
2	0.140 650	3	9	0
3	0.063 012	4	12	0
…	…	…	…	…
45	0.000 023	45	43	1
46	0.503 937	16	46	0

位平面编码产生的 symbol,CX 输入 MQ 编码器进行编码,首先根据 symbol 的上下文(CX)在上下文表中查找出该上下文对应的小概率符号的概率索引 I 和大概率符号 MPS,然后利用该概率索引 I 在概率估计表中查找出对应的 LPS 的概率 Q_e,最后根据 symbol 是否为 MPS 以及 Q_e 的值进行编码,生成压缩比特流 CD。

编码时设置两个专用寄存器 A 和 C,A 寄存器中的数值为子区间的宽度,C 寄存器中的数值为子区间的起始位置,用$[C,C+A]$表示它的编码区间。区间 A 保持在 $0.75 \leqslant A < 1.5$ 的范围内,当整数值降到小于 0.75 时,通过重归一化把它加倍,即把 A 限制在十进制范围 0.75～1.5 内。重归一化发生时,调用概率估计模型,为当前正被编码的上下文确定一个新的概率估计,概率区间的划分可以使用简单的算术近似方法。如果 LPS 当前的概率估计值是 Q_e,则子区间的精确计算如下进行。

如果输入符号位 MPS:
$$\begin{aligned} A &= A \times (1 - Q_e) \\ C &= C + A \times Q_e \end{aligned}$$

如果输入符号位 LPS:
$$\begin{aligned} A &= A \times Q_e \\ C &\text{ 不变} \end{aligned}$$

为了减少乘法运算，在 A 满足 $0.75 \leqslant A \leqslant 1.5$ 时，可以将上述方程简化为：

如果输入符号位 MPS：$\begin{aligned} A &= A - Q_e \\ C &= C + Q_e \end{aligned}$

如果输入符号位 LPS：$\begin{aligned} A &= Q_e \\ C &\text{ 不变} \end{aligned}$

MQ 编码器的主流程图如图 5.18 所示。首先初始化编码器，然后读出 symbol 和 CX 并送到编码器中，编码器根据 CX 得到当前的 MPS 和 Q_e，进行 MPS 或者 LPS 编码，然后根据需要更新概率模型和进行重归一化操作，重复以上过程直到所有的编码位都编码完成为止。最后进行编码器清空，完成当前部分的 MQ 编码。

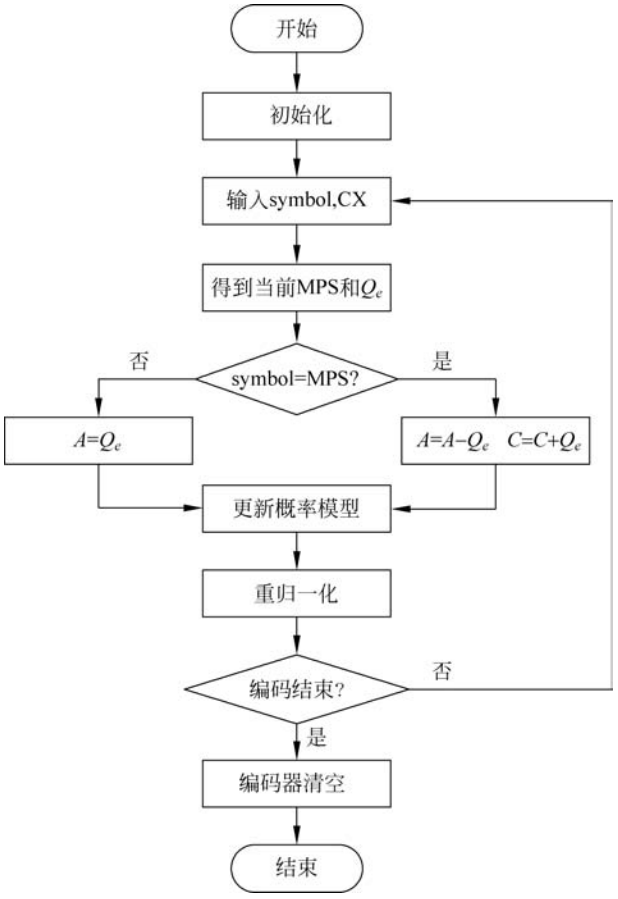

图 5.18　MQ 编码器的主流程图

2）Tier2 层

第一阶段块编码得到的仅仅是各个独立码块的码流，为了使解码得到的图像具有不同的特性，必须对这些码流进行有效的组织。从上面的介绍可以知道，以码块为单位的码流是按照块的不同失真度组织的，随着块码流的增加，失真度减少。为了使得全图像在一定码长下的失真度最小，就要从每块中裁剪部分码流组织在一起，这个过程称为打包过程，也就是第二阶段编码。这一过程实现了一定保真度和分辨率的码率可伸缩性和渐进性。

码流的基本单位是包，每一个包由两部分组成：包头(Packet Header)和包体(Packet Body)。包头的信息主要是这个包里面包含哪一个编码通道，包体则是实际的编码通道数据。在编码流中，包头和包体可以一起出现也可以分开出现，这主要是取决于编码中的具体参数选择。

在 JPEG2000 编解码系统中，EBCOT 算法是其重要的组成部分。而 EBCOT 算法当中的第一阶段块编码又是整个算法的核心，它占用了大量的编码时间，具体如表 5.7 所示。

由表 5.7 可以看出，无论是无损压缩还是有损压缩，EBCOT 算法都占整个编码器耗时的 70%以上，而其中的位平面编码时间，更是占到整个编码耗时的 50%以上。所以，自从 EBCOT 算法提出后，由于第一阶段块编码的运算量比较大、编码速度较慢，针对这种情况的优化研究很有必要。目前很多学者提出了相应的改进方法，比较有代表性的有样点省略法和群列省略法。

表 5.7 JPEG2000 中编码器各模块耗时分析

编码器各模块		无损压缩	有损压缩
DWT		12.2%	20.1%
量化		5.8%	5.5%
EBCOT	位平面编码	55.0%	51.8%
	MQ 算术编码	8.2%	6.9%
	Tier2	13.9%	11.7%
其他		4.9%	4.0%
总计		100%	100%

小结

基于 DCT 和 DWT 的混合图像压缩技术是静态图像编码的主流技术，以这两种变换为基础分别制定了 JPEG 标准和 JPEG2000 标准。目前，JPEG2000 标准已经非常成熟，相应的压缩编码芯片已经在多个领域得到了广泛应用，在视频压缩中也有应用。本章首先介绍了小波变换的基本概念，接着介绍了小波提升算法原理及其在 JPEG2000 中的应用，最后对 JPEG2000 编解码器中耗时最大的 EBCOT 算法进行了深入的介绍。

习题

1. 试绘出并简述 JPEG 图像压缩算法的总体流程。

2. 未经压缩的数字图像是如何表示和存储的？

3. 设某一幅图像共有 4 个灰度级，各灰度级出现的概率分别为 1/2、1/4、1/8 和 1/8，试计算图像的熵是多少？采用不等长编码的效率是多少？