第 34 卷 第 4 期 计 算 机 工 程 2008 年 2 月
Vol.34 No.4 Computer Engineering February 2008
·多媒体技术及应用· 文章编号：1000—3428(2008)04—0241—03 文献标识码：A 中图分类号：TP391.41

改进的多分辨率SPIHT算法

宋春林[1]，冯 瑞[2]，金 炜[1]，郭爱煌[1]
(1. 同济大学信息与通信工程系，上海 200092；2. 复旦大学计算机科学与技术系，上海 200433)

摘 要：由于引入了小波树，隐藏了扫描路径，因此 SPIHT 算法能获得较高压缩比，同时保持较好的图像解码质量。而多分辨率 SPIHT 算法能根据接收方的分辨率需求，使解码器根据不同信道条件选择图像还原分辨率。但是，该算法按照分辨率级成组扫描处理每级 LIP, LIS, LSP 表，更新下一级表时，会造成重复比较和冗余编码，既浪费执行时间，又增加了计算复杂度。该文改进了该算法，简化了原有算法流程，减少了编码冗余。理论分析和实验表明，在保持较高 PSNR 的同时，该算法明显提高了编码速度。

关键词：图像压缩；小波分析；SPIHT 算法；多分辨率

Improved Multi-resolution SPIHT Algorithm

SONG Chun-lin[1], FENG Rui[2], JIN Wei[1], GUO Ai-huang[1]
(1. Department of Information and Communications Engineering, Tongji University, Shanghai 200092;
2. Department of Computer Science and Engineering, Fudan University, Shanghai 200433)

【Abstract】 Since the introduction of wavelet tree and certain pass, SPIHT can achieve high compression ratio as well as high image quality. Later multi-resolution SPIHT is proposed to enable the decoder to select certain resolution as channel condition permits. However, the original algorithm produces repeated comparison and redundant coding in resolution-based sorting, wasting time and increasing complexity when sorting LIP, LIS LSP, tables while updating them at next level. In the paper, a reduced algorithm is introduced to decrease comparison time so as to promote compression rate with high PSNR. Theoretical and simulation analysis indicates that the proposed method can promote the encode speed and the PSNR can be guaranteed.

【Key words】 image compression; wavelet analysis; SPIHT; multi-resolution

1 概述

自 1993 年Shapiro提出嵌入式零树小波(EZW)以来[1]，许多学者致力于零树编码的研究，并对它进行了改进，其中最有影响的是A. Said和W. Pearlman提出的分级树集合划分(SPIHT)算法[2]，其产生的嵌入式码流，可进行速率控制，并且压缩比大，算法简单，易于实现。然而，由于信道条件或者传输速率受限，解码器往往并不能够或者并不需要还原全分辨率图像，而只是希望获得小分辨率图像，如果解码器仍然对比特流进行全分辨率解码，这样既浪费执行时间，又占用存储空间。于是，有学者提出了多分辨率SPIHT算法[3]，一种在SPIHT上实现多分辨率选择的压缩编码算法，它支持渐进传输，能根据分辨率要求还原图像，同时获得较高的峰值信噪比。但是，该算法存在以下 2 点缺陷：(1)在处理LIP, LIS, LSP表，进行像素或者子集重要性判断的时候，SPIHT算法往往重复比较像素值与阈值，增加了执行时间。(2)在按照分辨率级成组扫描LIP, LIS, LSP表的时候，LIS表扫描产生的不重要像素会被添加到下一级的LIP表中，当扫描下一级LIP表的时候，就会重复进行重要性判断并且输出冗余比特，造成编码时间增加，降低了压缩比。尽管不少学者针对此问题进行了一些改进[4-8]，但仍然没有很好地解决这 2 个问题：重复比较和编码冗余。

本文针对以上 2 个问题分别提出改进方案：(1)引入重要性矩阵。在每次阈值扫描之前，分别生成像素和子集重要性矩阵，排序扫描 LIP, LIS 表任何需要进行重要性判断的地方，只要参考重要性矩阵，就能得到判断结果，这样在处理像素或者子集的时候，就能够减少比较判断次数；(2)改变扫描方式。在多分辨率算法中，将原算法按照分辨率级扫描改为按照有序表扫描，即分开扫描各级 LIP, LIS, LSP 表，就能避免下一级 LIP 表不重要像素的编码冗余。

2 SPIHT 算法

SPIHT叫做分级树集合划分算法[2]，利用小波变换的能量集中性和小波系数的各级子带相似性，在特殊的扫描方式下，能有效地从高能量层到低能量层渐进编码，优化系数输出。该算法使用隐藏路径的扫描方式，不产生额外的路径信息而达到提高压缩比的目的，解码时采取统一分集规则[2,4]。为了说明分集规则，现引入以下符号，对某一节点(i, j)：

$O(i, j)$ —— 此节点的所有孩子坐标集。

$D(i, j)$ —— 此节点的所有子孙坐标集(包括孩子)。

$L(i, j)$ —— 此节点的所有非孩子子孙坐标集。

很明显，$L(i, j)=D(i, j)-O(i, j)$。

分集规则如下：如果 $D(i, j)$ 关于当前阈值是重要的，则将其分解为 $L(i, j)$ 和 4 个 $O(i, j)$；如果 $L(i, j)$ 关于当前阈值是重要的，则将其分解为 4 个 $D(i, j)$。接着引入 3 个有序表

基金项目：国家自然科学基金资助项目(60571049)；上海市科委专项基金资助项目(05dz12006)；同济大学工科发展基金资助项目(0800219040)

作者简介：宋春林(1973－)，男，副教授、博士，主研方向：小波理论，图像处理；冯 瑞，副教授、博士；金 炜，硕士；郭爱煌，教授、博士

收稿日期：2007-03-08 **E-mail：**songchunlin@mail.tongji.edu.cn

来存放扫描信息，它们分别如下：

LSP 表 —— 存放重要系数坐标。

LIP 表 —— 存放不重要系数坐标。

LIS 表 —— 存放不重要子集坐标。

SPIHT 算法的具体步骤如下：

(1)初始化LIP, LIS, LSP表，并确定最大扫描阈值：LIP={(1,1), (1,2), (2,1), (2,2)}; LIS={D(1,2), D(2,1), D(2,2)}; LSP=Φ。阈值$T=2^n$，其中，$n=\left\lfloor \mathrm{lb}\max\left|(i,j)\right|\right\rfloor$。

(2)对于给定的阈值，依次进行 LIP, LIS 表的排序扫描：依次判断 LIP 表中每个小波系数的重要性，如果(i, j)重要，则输出“ 1 ”及其符号位，正负分别用“ 1 ”、“ 0 ”表示，并将此节点自 LIP 表中删除，添加到 LSP 表的末端；如果(i, j)不重要，则输出“ 0 ”，不删除此节点，仍然将其保留在 LIP 表中。然后排序扫描 LIS 表中的每个子集，对重要的子集进行分集操作，分集时采用不同的方式处理 $D(i,j)$和 $L(i,j)$，算法流程如图 1 所示。

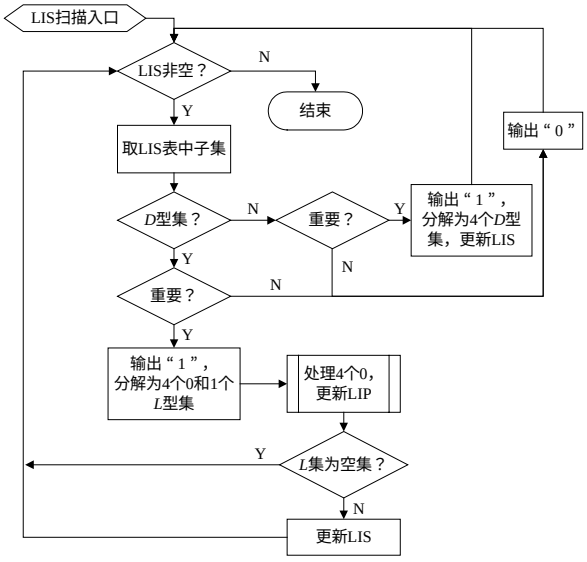

图 1　LIS 表排序扫描流程

(3)对同一阈值，依次进行LSP表中每个节点的精细扫描：输出LSP 表中非新添加节点所对应小波系数的二进制表示的第 n+1 位，此次扫描结束。

(4)进行下一次扫描：阈值 $T\leftarrow T/2$, $n\leftarrow n$-1，重复(2)和(3)，直到阈值或者比特率符合编码器要求。

(5)输出编码结果。

3　SPIHT 算法分析

分析图 1 SPIHT 算法，LIS 表中 2 种类型的子集坐标(D 型和 L 型子集)，分别对应了 2 条支路，这就增加了算法的复杂度，并且占用了大量空间存储子集类型信息。而且在扫描 LIS 表，进行分集操作的时候，先要对某个集进行重要性判断，如果这一节点是孤立零点，则该集会被分为 4 个 O 和 1 个 L 型集，L 型集被添加到 LIS 表后端，当扫描到这个 L 型集的时候，又要进行重要性判断，而这次的判断实际上先前已经进行过，是重复工作。

有学者提出一种改进[4]，不引入L型子集，而是直接将原来的L型分解为 4 个O型和 4 个D型，即分集规则定义为重要的该子集分解为 4 个孩子进行处理，并且将 4 个孩子子集直接添加到LIS表后端。这样一来，虽然减少了L型子集带来的复杂度，但是会增加输出码元数量，同时仍然没有解决D型子集的重复比较问题。

而后提出的多分辨率SPIHT[3]，引入多级LIP, LIS, LSP表，按照小波系数的能量或者频率，分别对应不同分辨率k或者称之为小波树级：最低分辨率k=K对应图像左上角的2×2 像素，最高分辨率k=1 对应小波树的叶子。每次阈值扫描时，按照分辨率级依次处理此级LIP, LIS, LSP表，即先处理最低分辨率的LIP, LSP, LIS表，提高分辨率，处理下一级的LIP, LSP, LIS表，直到最高级分辨率k=1 处理完毕，本次阈值扫描完毕，流程如图 2 所示。

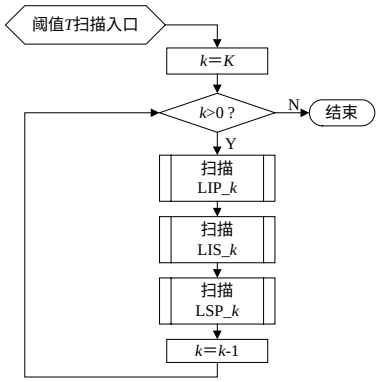

图 2　多分辨率 SPIHT 算法流程

然而，在处理某一级 LIS_k 表时，编码器或者解码器将不重要像素添加到下一级的 LIP_k-1 表末端并输出“ 0 ”，在下一次扫描 LIP_k-1 表的时候，将重复进行重要性判断，并且输出这些不重要系数的“ 0 ”，造成编码冗余。

4　本文改进算法

本文提出的“ 简化的多分辨率 SPIHT 算法 ”，通过引入像素和子集重要性矩阵去除重复比较，通过修改扫描流程去除编码冗余，算法的主要思想如下：

(1)引入 2 个重要性矩阵。在每次阈值扫描之前，进行像素和子集重要性判断，形成像素重要性矩阵 $\boldsymbol{P}$ 和子集重要性矩阵 $\boldsymbol{I}$，分别记录像素和子集的重要性判断结果。排序扫描流程中需要进行重要性判断的时候，只需读取相应矩阵中对应位置的数值即可，$\boldsymbol{P}$, $\boldsymbol{I}$ 生成规则如图 3 所示，对某一给定阈值 T，将 $\boldsymbol{P}$, $\boldsymbol{I}$ 矩阵所有位置置零。

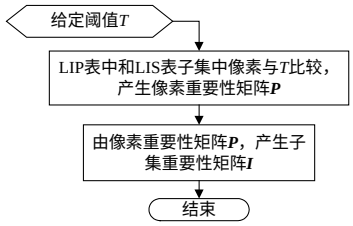

图 3　$\boldsymbol{P}$, $\boldsymbol{I}$ 矩阵生成流程

(2)改变扫描方式。如果某一像素重要，则在 $\boldsymbol{P}$ 矩阵中相应位置标示“ 1 ”，$\boldsymbol{P}$ 矩阵大小为原图像大小；从空间树的叶子开始扫描 $\boldsymbol{P}$ 矩阵，如果该像素重要($\boldsymbol{P}$ 矩阵相应位置为“ 1 ”)，则在 $\boldsymbol{I}$ 矩阵中，其父母相应位置标示“ 1 ”。然后扫描上一级节点，处理同上，直到扫描到左上顶端的那个根节点为止，$\boldsymbol{I}$ 矩阵大小为原图像大小的 1/4。

(3)给定扫描阈值，在排序扫描中，按照有序表进行扫描，如图 4 所示，依次从最低分辨率到最高分辨率，先对所有分辨率级的 LIP 表进行扫描，然后进行各级 LIS 扫描，同时更新本级的 LIS 表和下一级的 LIP, LSP, LIS 表。

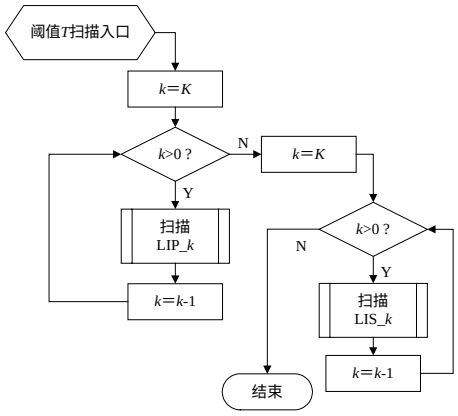

图 4　多分辨率 SPIHT 算法排序扫描流程

简化的多分辨率 SPIHT 算法实现步骤如下：

(1)初始化各级 LIP, LIS, LSP 表，并确定最大阈值；

(2)对于给定的阈值，生成像素重要性矩阵 ***P*** 和子集重要性矩阵 ***I***；

(3)依次进行各级 LIP 表排序扫描，从最低分辨率 $k=K$ 到最高分辨率 $k=1$；

(4)依次进行各级 LIS 表排序扫描，每级扫描时，更新下一级 LIP, LSP 和 LIS 表；

(5)依次进行各级 LSP 表精细扫描；

(6)$T \leftarrow T/2$, $n \leftarrow n-1$，重复(2)~(5)，直到阈值或者比特率符合编码器要求；

(7)输出编码结果。

例如，对图 5 的 8×8 图像进行改进后多分辨率 SPIHT 编码，不引入 *L* 型子集。

63	-34	49	10	7	13	-12	7
-31	23	14	-13	3	4	6	-1
15	14	3	-12	5	-7	3	9
-9	-7	-14	8	4	-2	3	2
-5	9	-1	47	4	6	-2	2
3	0	-3	2	3	-2	0	4
2	-3	6	-4	3	6	3	6
5	11	5	6	0	3	-4	4

图 5　小波变换后 8×8 图像

假设当前阈值 $T=32$，进行排序扫描，LIP_3={(1,1), (1,2), (2,1), (2,2)}, LIS_3={(1,2), (2,1), (2,2)}, LIP_2, LIP_1, LIS_2 都为空集，依次扫描 LIP_3, LIP_2, LIP_1 后，输出排序码流为 11 10 0 0，LIP 表更新为 LIP_3={(2,1), (2,2)}；LIP_1, LIS_2 仍然为空集。

进行 LIS_3 表的扫描，输出排序码流为 1 11000, 1 0000, 0，更新 LIS_2 表，LIP_2 表为 LIS_3={(2,2)}；LIS_2={(1,3), (1,4) , (2,3) , (2,4) , (3,1) , (3,2) , (4,1) , (4,2)}；LIP_2={(1,4) , (2,3) , (2,4) , (3,1) , (3,2) , (4,1) , (4,2)}。

进行 LIS_2 表的扫描，输出排序码流为 0, 0, 0, 0, 0, 1 01100, 0, 0；更新 LIP_1 表为 LIS_2 ={(1,3), (1,4) , (2,3) , (2,4) , (3,1), (4,1) , (4,2)}；LIP_1 ={(5,3), (6,3), (6,4)}。

至此，关于阈值 $T=32$ 的排序扫描结束，共输出 31 个排序路径信息码元，其中含路径码元共 27 个。比之原算法按照分辨率级成组扫描 LIP_*k*, LIS_*k* 表来说，仅仅一次阈值排序扫描就节省路径码元 10 个，*x* 消除冗余占总路径信息码元数的 27.03%。

5　算法仿真结果及分析

对 Lena 图(256×256)进行 Haar 小波变换后系数全分辨率编码，不引入 *L* 型子集，解码器的不同分辨率解码结果比较如图 6 和表 1 所示。实验表明：在保持较高 PSNR 的同时，本文改进算法明显减少了比较次数，提高了编码速度。

图 6　Lena 图像多分辨率解码图

表 1　Lena 多分辨率解码数据

解码分辨率	处理码元数	所占比例/(%)	PSNR/dB
256×256	87 031	100.000	33.961
128×128	28 058	32.240	33.677
64×64	7 506	8.625	33.535
32×32	2 392	2.748	33.423

6　结束语

本文改进了多分辨率 SPIHT 算法，通过引入 2 个重要性矩阵以及改变扫描方式，减少了重要性判断时的比较次数，降低了编码冗余，缩短了解码时间。对于无需最高分辨率图像的解码器来讲，此改进算法节省了大量的带宽和解码时间。

参考文献

[1] Shapiro J. Embedded Image Coding Using Zero Trees of Wavelet Coffients[J]. IEEE Trans. on Signal Processing, 1993, 41(12): 3445-3462.

[2] Said A, Pearlman W A. A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees[J]. IEEE Transactions on Circuits and Systems for Video Technology, 1996, 6(3): 245-247.

[3] Cho S, Pearlman W A. A Full Featured, Error Resilient, Scalable Wavelet Video Codec Based on the Set Partitioning in Hierarchical Trees(SPIHT) Algorithm[J]. IEEE Transactions on Circuits and Systems for Video Technology, 2002, 12(3): 157-160.

[4] 柯　丽, 黄廉卿. 提升方案结合改进 SPIHT 的快速图像压缩方法[J]. 光电工程, 2005, 32(1): 60-61.

[5] 贾志科, 崔慧娟, 唐　琨. 改进的 SPIHT 静止图像压缩编码算法[J]. 清华大学学报: 自然科学版, 2001, 41(7): 25-28.

[6] KiLyug K. Persormance Improvement of the SPIHT Coder[J]. Signal Processing: Image Communication, 2004, 19(1): 29-36.

[7] 苏卫东, 慈林林, 陈晓峰. 基于子带极值阶梯性的 SPITH 算法改进方案[J]. 计算机工程, 2005, 31(16): 152-154.

[8] 陈仁喜, 赵忠明. 一种采用分层多阈值的多分辨率渐进图像编码算法[J]. 计算机工程, 2006, 32(12): 218-220.