改进的快速 SPIHT 算法

柯 丽[1,2]，黄廉卿[1]

（1. 中国科学院 长春光学精密机械与物理研究所，吉林 长春 130033；
2. 中国科学院 研究生院，北京 100039）

摘 要：SPIHT 算法是一种简单、有效的嵌入式零树编码算法，但是，它需要大量存储空间，而且存在多次重复运算，因而复杂程度高，时间消耗大，不利于实时压缩。改进的 SPIHT 算法针对原算法的不足引入了“最小阈值”和“最小输出位”，同时改变了原算法的扫描顺序，降低了算法的复杂程度，并使其更有利于并行优化处理。实验证明，改进后的算法减少了编解码过程中的存储容量和时间消耗，而重建图像的峰值信噪比和人眼视觉效果与原算法相当。

关键词：图像压缩； 小波变换； 零树编码； 多级树集合分裂算法

中图分类号：TP301.6 **文献标识码**：A **文章编号**：1007-2276(2004)05-0509-04

Improved fast SPIHT algorithm

KE Li[1,2], HUANG Lian-qing[1]

(1. Changchun Institute of Optics, Fine Mechanics and Physics, Chinese Academy of Sciences, Changchun 130033, China;
2. Graduate School of Chinese Academy of Sciences, Beijing 100039, China)

Abstract: SPIHT algorithm is an embedded zerotree coding algorithm known for its high performance. However, it needs a large number of memories and many repeated calculations, so it is more complicated, and can't implement real-time compression. The improved algorithm modifies the SPIHT algorithm by use of two concepts—minimum threshold and minimum exported bit, changing the original scanning order. Experimental results show that it reduces both the memory requirement and the time consumption, at the same time the *PSNR* value and human's visual effect of the reconstructed image are close to SPIHT algorithm.

Key words: Image compression; Wavelet transform; Zerotree coding; SPIHT

0 引 言

小波变换在时域和频域都有良好的局部化特性，并且变换后的各子图像具有很强的相似性，为后续的编码带来了方便，使小波变换成功地应用于图像压缩领域[1~3]。

1993 年美国学者 Jerome M. Shapiro 根据一幅图像的小波变换在不同级之间的相似性，提出了嵌入式零树小波编码方法[1,1](EZW)，取得了良好的压缩效果。1996 年 A. Said 和 W. A. Pearlman 根据 Shapiro 零树编码算法的基本思想，提出了一种新的实现

收稿日期：2003-10-05； **修订日期**：2003-11-23

作者简介：柯丽(1977-)，女，辽宁庄河人，博士生，主要研究方向为数字图像处理与图像实时压缩。

方法，即多级树集合分裂算法[5](SPIHT)。它采用了空间方向树更有效地表示小波系数的零树结构，从而提高了编码效率。但是SPIHT算法需要大量存储空间，而且存在多次重复运算，不利于实时压缩，本文针对SPIHT算法存在的问题加以改进，降低了算法的中间存储量和时间消耗，使其可以应用于实时压缩。

1 SPIHT算法

1.1 算法概述

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[5]”结构，树根是最低频子带的节点。SPIHT算法根据空间方向树结构，将集合的分割策略定义为：

$$\begin{cases} Z(i,j) = c(i,j) + D(i,j) \\ D(i,j) = O(i,j) + L(i,j) \\ L(i,j) = \sum D(k,l) \qquad (k,l) \in O(i,j) \end{cases} \tag{1}$$

式中 $Z(i,j)$为空间方向树；$c(i,j)$为树上任一节点；$D(i,j)$表示节点$c(i,j)$的所有后代(子孙)的坐标集合；$O(i,j)$表示节点$c(i,j)$的直接后代(儿子)的坐标集合；$L(i,j)$表示节点$c(i,j)$除直接后代(儿子)外所有后代坐标的集合。

SPIHT算法编码过程分为排序和细化两部分[5]。排序过程根据上述集合分割策略，将空间方向树上的节点分类，过程中使用了3个链表记录相关信息：不重要集合链表(LIS)、不重要像素链表(LIP)及重要像素链表(LSP)。具体过程如下：

首先将一个空间方向树在初始化时分裂成树头节点$c(i,j)$和剩余集合$D(i,j)$，判断节点$c(i,j)$和集合$D(i,j)$的重要性，若$c(i,j)$重要则转到重要系数链表LSP中，若$D(i,j)$是重要的，则$D(i,j)$继续分裂为两个集合$O(i,j)$和$L(i,j)$。对集合$O(i,j)$中的每个元素分别进行重要性测试，把重要元素转到LSP中。对集合$L(i,j)$进行重要性测试，若$L(i,j)$重要，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，把它们转到LSP中。

细化过程是对LSP中的每一个表项$c(i,j)$进行的，对于阈值T_i来说，输出重要像素的绝对值$|c(i,j)|$二进制表示的是第i位有效位。

将n减1，阈值减半，进行重复排序和细化，直到编码结束。

1.2 算法分析

从上述编码过程来看，SPIHT算法存在以下几方面缺陷：

(1) 编码过程中需要占用大量内存。随着阈值的降低，扫描次数的增加，算法中用来存储重要系数坐标、重要集合坐标和不重要系数坐标的3个链表的存储空间越来越大。

(2) 在排序过程中存在大量的重复操作。每次变换阈值时，对上次遗留的非重要元素需要逐个与新阈值比较，增加了编码时间。

(3) 对所有频域进行同等重要的编码，没有充分利用小波变换的特点，以及人的视觉系统对不同频域图像敏感度不同的特性。

(4) 对图像各部分统一编码，环环相扣，不利于并行算法优化。

本文针对SPIHT算法的几点不足，提出了一种改进的快速压缩算法，在兼顾压缩比和保真度的情况下，减少了运算时间，并提高了实时性。

2 改进的SPIHT算法

2.1 算法描述

改进算法具体描述如下：

(1) 针对储存量大的问题，引入了“最小阈值”($T_{\min}$)和“最小输出位”($bit_{\min}$)的方法来解决。原算法的编码过程是一个随着阈值的降低逐次扫描的过程，每扫描一次，则多输出一个重要系数的二进制有效位。在一定的压缩比下，阈值减小到一定的大小则停止编码。

每一幅图像在不同的压缩比下，都存在一个最小阈值，大量的实验表明，一般的图像在相同的压缩比下，最小域值的取值大致相同。改进算法由大量实验统计，得出不同压缩比下的最小阈值，由该阈值推算出重要系数的最小输出位：

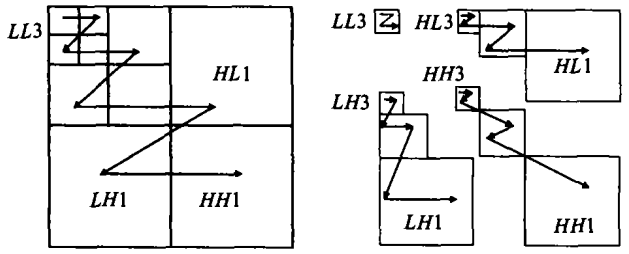
$$bit_{\min} = \log_2(T_{\min}) \tag{2}$$

在编码过程中，确定了某系数为重要系数后，确定其最小输出位之前的所有位一次性输出，而不将坐标移入LSP中保存，由此省去LSP链表，使原来的3个链表变为2个，节省了储存空间。

(2) 针对其重复扫描的问题,使用"最大值表"来解决。SPIHT 算法编码过程中将阈值不断减半,重复排序和细化,直到编码结束。改进的算法在第一遍排序过程中,与 SPIHT 算法一样,逐个元素比较,不同的是改进算法在此过程中同时建立最大值表,并存入 LIS 链表中对应元素的子孙节点中的系数最大值。所建立的最大值表与链表 LIS 大小相同,并且随着 LIS 大小的变化而变化。在以后的过程中,SPIHT 算法每将阈值减半一次,排序时都需要把所有子孙节点系数一一与阈值比较,而改进后的算法由于引进了最大值表,只将最大值表中的值与阈值比较即可,不需要逐个元素比较,计算量大大减少。

(3) 针对图像各部分统一编码不利于并行优化的问题,根据各个子图像的不同特点,改变原来的扫描路线,采用四路并行分块处理的方法,提高编码速度。原扫描路线是从最低频子带开始,结束于最高频子带。在移到下一子带之前,要把当前子带的系数全部扫描完。

改进的算法把扫描路线作一些调整,首先把小波变换后的图像分成 4 块(以 3 级小波分解为例):低频平滑部分($LL3$)、水平边缘细节($HL3$、$HL2$、$HL1$)、垂直边缘细节($LH3$、$LH2$、$LH1$)和对角线边缘细节($HH3$、$HH2$、$HH1$),再分别对每一块进行扫描编码,如图 1 所示。



(a) 原顺序　　(b) 新顺序

(a) Original scanning order　　(b) New scanning order

图 1　扫描顺序对比

Fig. 1　Comparison of scanning order of SPIHT algorithm and improved algorithm

这样改进的好处是可以针对不同块分别处理,对信息量高的块采用低压缩比、高保真编码方法;而对信息量低的块进行大压缩比压缩,在压缩比和峰值信噪比相当的情况下,提高恢复图像的人眼视觉质量。另外,改进后的编码将图像分为 4 块处理,同时将原 LIP、LIS 链表的储存量分布到各个部分,化整为零,提高了处理速度,且有利于硬件实现。

2.2　算法实现

对于每块的编码,仍延用零树思想[4],利用不同子带之间的父子关系进行编码。图像经小波变换之后,改变了图像的能量分布,低频子带集中了大部分能量,而高频子带中的大部分小波系数近似为 0。根据统计分析,一幅图像经 3 级小波变换之后,其低频子带 $LL3$ 占总能量的 95%以上,而最高频子带 $HH1$ 只有不到 1%。

由以上分析得知,对低频平滑部分($LL3$)需要使用高保真编码算法。$LL3$ 子带只有元素信息,不存在集合,可以省去 LIS。编码时首先将每一个小波系数与阈值比较,如果重要则将其最小输出位之前的所有位一次性输出,否则移入 LIP;然后将阈值减半,重复上述过程,直到达到预定的压缩比。在这一过程中,只需 LIP 一个链表,减少了储存空间,而且相对于其他各种编码方法,不需要进行任何量化和统计过程,大大降低了运算时间。

对于水平边缘细节和垂直边缘细节两部分($HL3\sim HL1$、$LH3\sim LH1$),由于它们的结构和能量分布都非常相似,可采用相同的压缩算法。对于这两部分细节,编码时引入"最小阈值"、"最小输出位"和"最大值表"的思想,使用改进的 SPIHT 算法进行编码。

对角线边缘细节部分($HH3\sim HH1$)包含的信息量最小,尤其是 $HH1$,它的信息量接近于 0,其数据量却是整幅图像的 1/4。根据该特点,对角线边缘细节部分可以将 $HH1$ 舍去,只对 $HH2$ 和 $HH3$ 进行编码,恢复时 $HH1$ 部分全用 0 填充,计算量大大减少。实验证明,采用上述方法对于大部分图像是可行的,但对于纹理细节丰富的图像,$HH1$ 部分仍占有一定的信息量,不能简单舍去,需要参与编码。对纹理细节丰富的图像,对角线边缘细节部分采用与水平、垂直边缘细节两部分相同的算法即可。

3　实验结果与分析

根据本文提出的改进算法,分别对 512×512 的标准灰度图像 Barbara、Lena 进行了编解码实验,在相同的压缩比情况下,从恢复图像的峰值信噪比和编码时间两方面与 SPIHT 算法进行了对比,实验结果

如表 1、表 2 所示。实验中采用了 D5/3 小波对图像进行 3 级小波分解，D5/3 小波是 Daubechies 系紧支集规范双正交小波基，适于图像快速压缩。图 2 为改进后的算法和原算法重建图像的视觉质量比较。

表 1 恢复图像的峰值信噪比比较

Tab. 1 Comparison of *PSNR*

Compression ratio	Barbara		Lena	
	SPIHT algorithm	Improved algorithm	SPIHT algorithm	Improved algorithm
2	37.34	37.37	41.69	39.87
4	34.60	34.42	38.63	37.91
6	31.52	31.48	36.44	36.62
8	27.83	29.60	34.58	35.10
10	26.75	27.36	32.56	33.57

表 2 编码时间比较

Tab. 2 Comparison of coding time

Compression ratio	Barbara		Lena	
	SPIHT algorithm	Improved algorithm	SPIHT algorithm	Improved algorithm
2	3.33	1.25	3.49	0.95
4	1.98	0.73	2.56	0.71
6	1.40	0.55	1.54	0.41
8	1.19	0.47	1.36	0.35
10	0.91	0.39	0.97	0.27

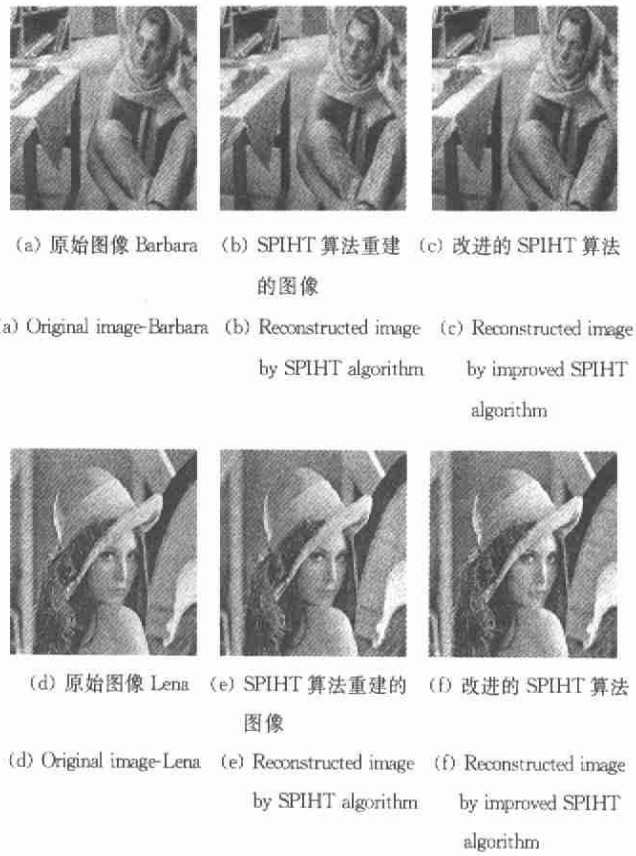
Barbara 属于纹理细节丰富的图像，对于这类图像，对角线边缘细节部分所用的算法与水平和垂直边缘细节两部分的算法相同，故所消耗的时间比 Lena 图像要多，相反 Lena 属于纹理较为简单的一般图像，可以把它的 $HH1$ 部分在编码过程中进行舍去处理，恢复时完全用 0 填充。

从表中的数据可以看出，改进后的算法在恢复图像质量方面与原算法相当，但编解码时间却大大减少。Barbara 图像的编码时间约为原算法的 1/3。对于 Lena 图像，由于它直接舍去了原图的 1/4 的数据量，编码时间约为原算法的 1/4。

从图 2 可以看出，在压缩比相同的情况下，新算法与原算法的恢复质量相当。

4 结 论

在研究了 SPIHT 算法的基础上，针对原算法的不足，提出了一种改进的 SPIHT 算法。在使用少量储存空间的条件下，减少了运算量，在压缩性能相同的情况下，编解码时间大大缩短。同时，新算法改变了原来的扫描顺序，便于并行处理，有利于实时编码处理的实现，在一定程度上提高了编码性能。



(a) 原始图像 Barbara (b) SPIHT 算法重建的图像 (c) 改进的 SPIHT 算法

(a) Original image-Barbara (b) Reconstructed image by SPIHT algorithm (c) Reconstructed image by improved SPIHT algorithm

(d) 原始图像 Lena (e) SPIHT 算法重建的图像 (f) 改进的 SPIHT 算法

(d) Original image-Lena (e) Reconstructed image by SPIHT algorithm (f) Reconstructed image by improved SPIHT algorithm

图 2 改进的算法与 SPIHT 算法重建结果的比较

Fig. 2 Comparison of image quality after being compressed by improved algorithm and SPIHT algorithm

参考文献：

[1] 陈武凡，杨丰，江贵平，等. 小波分析及其在图像处理中的应用[M]. 北京：科学出版社，2002.

[2] Rafael C Gonzalez, Richard E Woods. Digital image processing. 2nd ed[M]. 北京：电子工业出版社，2002.

[3] Antonini M, Barlaud M, Mathieu P, et al. Image coding using wavelet transform[J]. IEEE Trans Image Processing, 1992, 1(4): 719-746.

[4] Shapiro J M. Embedded image coding using zerotree of wavelets coefficients [J]. IEEE Trans Signal Processing, 1993, 41(12): 3445-3462.

[5] Amir Said, William A Pearlman. A new, fast, and efficient image code based on set partitioning in hierarchical trees [J]. IEEE Transactions on Circuits and Systems for Video Technology, 1996, 6(6): 243-250.