

# 十六元树、二值图象数据压缩\*

陈 根 明

(北京市自动化技术研究所, 北京)

袁 保 宗

(北方交通大学, 北京)

**摘要** 本文介绍了一种二值图象数据压缩的十六元树方法,它具有效率高、便于实现,无失真恢复等特点,是处理二值图象中极为实用的方法,

**关键词** 数字图象处理;数据压缩;二值图象;十六元树

## 1. 引言

虽然二值图象(如:边缘图象、增强分割后的图象、文字图象、图案图象等)的处理技术在理论上一般并不高深,但是,在各个普通行业的实际应用中,一个有趣的现象是二值图象技术占图象应用技术的很大比例。

二值图象的数据压缩技术可以减少图象占用的存储空间,增加传输、处理速度。这种技术的研究目标是寻找压缩效率高、运算速度快、便于实用的方法。在70年代中,出现了一种利用“四元树”<sup>[1]</sup>进行二值图象数据压缩的方法。但原始的四元树方法是从树根至树梢一层一层地建立压缩的树状数据结构,需要对图象数据反复地进行搜索,运算量较大。不久,一次搜索即可建立整个树状数据结构的四元树方法问世<sup>[1,2]</sup>,在图象处理技术中得到了广泛的延展和应用,四元树方法的特点是方法简单,无失真,便于应用,对于稀疏的二次图象压缩效率比较高。它的弱点是由于四元树的数据结构是以四位二进制数为基本单元,与通用计算机的数据结构不一致,给编程带来一定困难;对景物较稠密的图象,不但没有什么压缩效果,反而有可能造成压缩后的数据量大于原始的数据量。

针对二值图象的特点,我们提出一种能克服四元树弱点的数据压缩方法:十六元树法。它比四元树方法易于在机器上实现,速度快,无论对稀疏图象还是稠密图象,压缩效果一般都比四元树的方法要好,而且不失四元树法简便无失真的特点。

## 2. 十六元树法

十六元树法的原理可用图1、图2加以说明。它与四元树法的基本原理类似。压缩后的数据结构是十六元的树状结构(见图2),进行数据压缩时,按 $4 \times 4$ 的方式一级一级分割图象(见图1),进行搜索。每级分割后的十六个块分别按图1的顺序对应于十六元

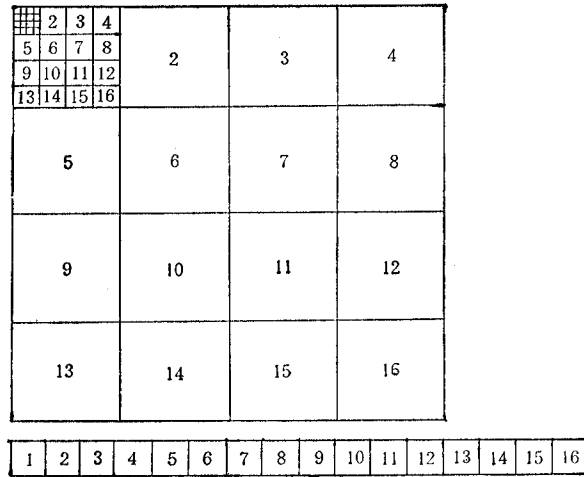


图 1

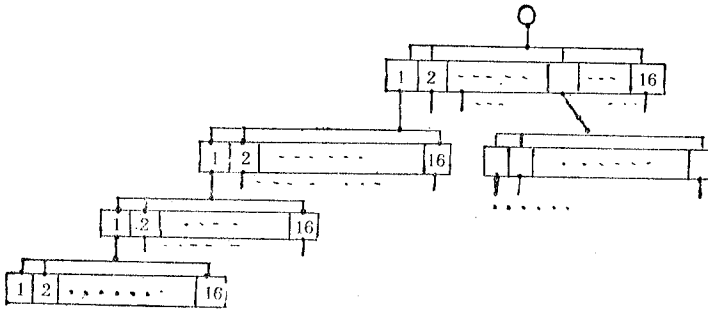


图 2

树当前级的十六个子树。十六元树数据的建立规则非常简单，如果数据压缩时的收缩当前级的十六块中有非零像素，它在数据树中所对应的当前级的分枝记为“1”，称为延伸分枝；如果此块中无非零像素，其对应的分枝即为“0”，称为终止分枝。对于值为“1”的延伸分枝，其下面又伸出十六个子分枝，分别对应于它所代表的块的再次 $4 \times 4$ 划分的十六个子块；对于值为“0”的终止分枝，其下面不再有子树，表示它所代表的当前级整个块内的像素值都为零。以这种方式一级一级地分割搜索下去，一直分割到 $4 \times 4$ 所划分后的块仅是 $1 \times 1$ 的子块，即仅一个像素大小，换言之，树的最末一级的各个分枝的数值分别仅表示了所有非零块中的各个像素的具体数值“1”和“0”。

对于 $256 \times 256$ 的图象，这种树共有四级，如图2所示。

二值图象的十六元树的数据压缩也与四元树一样可以一次搜索完成，即建立数据树时仅需对图象的所有各个像素访问一次，便可建立整个树形数据结构。由于这种一次搜索完成的原理同四元树非常相似，这里就不再赘述。

十六元树的二值图象恢复非常易于实现，沿着十六元数据树自树根至树梢进行搜索，遇到终止分枝便将相应级的整个相应块的像素都赋予零值，遇到延伸分枝则进一步向下搜索，搜索至树梢后，根据数据树建立时的顺序、规则、对图象中的各个像素进行赋值，当整个数据树搜索完毕后，它所对应的二值图象也就完成了。显而易见，十六元树的二值图

象数据压缩方法与四元树法一样,也是无失真的压缩和恢复。

十六元树法的主要优点有两个:(1)由于采用了十六分枝的树形数据结构,所以这种方法编程时也可全部采用十六位的数据形式,这时一般的通用计算机或专用计算机都是易于实现的,是四元树法所不如的。(2)压缩效率高。以  $256 \times 256$  的二值图象为例,十六元树法从树根搜索到树梢仅需四级,而四元树法则需要 8 级。换言之,即十六元树的覆盖速度比四元树法要快,这使得十六元树法的压缩效率大大提高,尤其是对稠密景物的二值图象,这种优点更为突出。十六元树法的优点可以用下面的例子给予部分的说明。

仍以  $256 \times 256$  的二值图象为例,四元树法的覆盖需要 8 级。这样,当数据树在全枝状态下,(即树的每个分枝都充满),整个树所需的数据量可以由下式进行计算:

$$4^1 + 4^2 + 4^3 + 4^4 + 4^5 + 4^6 + 4^7 + 4^8 = 87380\text{bit}$$

但是十六元树法的覆盖仅需要 4 级,其全枝状态下的数据量可用下式进行计算:

$$16^1 + 16^2 + 16^3 + 16^4 = 69904\text{bit}$$

可见,十六元树法的压缩效率是比较高的。当然,上述的例子是在图象极端稠密的假设下进行比较,在一般性的图象压缩处理中,十六元树法的压缩效果也是较高的。关于这一点,下节将用实验统计结果说明。

对于分辨率较高的图象如:  $512 \times 512$ 、 $1024 \times 1024$  的分辨率,十六元树法并不失其一般性,所有需要增加的工作,仅仅是在树的最高级稍作调整。如  $512 \times 512$  的图象,可先将图象等分为 4 个  $256 \times 256$  的方块,树的最高级可用一个 4bit 的字节,其每一个 bit 代表一个  $256 \times 256$  的子块,后续数据的压缩工作就仍然是完全按上述的十六元树的建树方法来进行,再如:  $1024 \times 1024$  的图象,其十六元树的数据结构的建立根本不用进行任何调整,只是整个树形结构又增加了一级。

### 3. 实验和结果

下面分别将四元树法和十六元树法对二十个景物稠密不同的二值图象作了数据压缩,结果列于表 1,从表 1 可以比较这两种方法的压缩效率。

表 1 压缩后数据量 (bit)

十六元	20904	32104	32912	29720	32136
四元	23872	38128	37088	31736	40888
十六元	11704	7152	8120	40240	42688
四元	14112	6976	7712	46512	51320
十六元	29712	33856	6368	23208	11952
四元	33864	38176	6192	24384	13728
十六元	29024	38984	21024	21992	23408
四元	31632	39496	24736	22968	28112

一般来讲,景物越稠密的图象,压缩后的数据量越大,这时,十六元树法的优点更突出。如表中的第五组数据,十六元树法为 32136bit,而四元树法为 40888bit。有时,四元树法的压缩量也优于十六元树法,如表中的第七组数据,十六元树法为 7152bit,四元树法

为 6976bit。不过在这种情况下,两种方法压缩后的数据量都比较小。

在表 1 中,如果以未经压缩的数据量作为 100% (65536bit),那么,从上述一组实验的结果来看,四元树法的统计平均压缩率为 42.7%,而十六元树法的统计平均压缩率为 38.1%,显然比四元树法的效率高,而且便于实现。十六元树法采用 FORTRAN 语言,压缩程序仅 37 条语句,恢复程序仅 31 条语句。

### 参 考 文 献

- [ 1 ] H. Samet, A. Rosenfeld, Qudtree Representation of Binary Image, Proc-Int Conf on Pattern Recognition, 5th, v2, Miami Beach, Fla, Qec. 1—4, 1980, Published by IEEE, Piscataway, NJ., 1980, 815—818.
- [ 2 ] M. A. Oliver, N .E. Wiseman, *Computer Journal*, 26(1983)1, 83—91.

## SIXTEEN TREE METHOD FOR BILEVEL IMAGE DATA COMPRESSING

Chen Genming

(*Beijing Automation Technology Research Institute, Beijing*)

Yuan Baozong

(*Northern Jiaotong University, Beijing*)

**Abstract** A sixteen tree method of data compressing of bilevel image is described. This method has high efficiency, and no information loss during compression; and it is easy to realize.

**Key words** Digital image processing, Data compression; Bilevel image; Sixteen tree