第13卷 第2期
1987 年 3 月

自 动 化 学 报
ACTA AUTOMATICA SINICA

Vol. 13, No. 2
Mar., 1987

P-四 元 树 表 示

严 成 文
(北 京 大 学)

摘 要

本文提出了一种新的用四元树表示区域的方法——P-四元树表示. 这种表示与四元树表示相比较,约节省存贮空间94%. P-四元树表示不但保持了四元树表示所具有的快速运算特性,而且显示了它的"基元-基元"的整体运算优越性. P-四元树可作为以微机为主的图象信息系统中理想的数据结构.

区域表示在图象处理、计算机图形学等方面起着重要的作用. 传统的、直观的表示区域的方法是二元数组表示. 随着图象处理技术在各方面的广泛应用和图象信息系统的飞速发展, 二元数组表示的空间效率与其"样点-样点"的运算方式都已不能适应发展的需要. 取而代之的应是既紧凑又便于作各种图象处理运算的表示方法. 为此, 人们相继建立了边界链码、游程编码、中轴变换等表示方法. 这些表示方法虽具有一定的紧凑性, 但一些基本的图象处理运算变得难以实现了. 例如: 用边界链码表示很难做区域的交、并运算,游程编码和中轴变换难以确定边界特征等等. 七十年代初, A. Klinger[1] 首先提出了用四元树表示二值图象的方法. 此后, A. Klinger 和他的学生[2-4]以及 Steiglitz 和 Hunter[5]对这一方法作了进一步的研究. 但是,真正广泛的研究, 是在 H. Samet 给出了为人们普遍接受的四元树的机器内部表示以后才开始的. H. Samet 等人的工作表明,许多图象处理运算都可在四元树表示的基础上快速实现[6-12]. 用四元树表示便于运算的特性归结于它的阶层结构和"方块-方块"的运算方式. 关于四元树表示的空间复杂性,人们一直直观地认为四元树表示是紧凑的. 事实上,对相当大的一类图象,四元树表示的空间效率不能令人满意. 因而,其实用性,特别是在以微机为主的图象信息系统上, 受到了限制. 本文提出了一种优于四元树表示的方法——P-四元树表示.

一、记号与定义

设 T 是一个树, S_T 是一个由树组成的集合. 记: $NL(T)$ 为树 T 上的叶子结点数; $NG(T)$ 为树 T 上的非叶子结点数; $NN(T)$ 为树 T 上的结点数; $NL[S_T]$ 为 $\sum_{T\in S_T} NL(T)$; $NG[S_T]$ 为 $\sum_{T\in S_T} NG(T)$; $NN[S_T]$ 为 $\sum_{T\in S_T} NN(T)$.

本文于 1984 年 12 月 24 日收到.

定义 1. 表示 $2^n \times 2^n$ 二值图象的四元树全体，称为 n 年四元树林. 记为 $Q(n)$.

定义 2. 表示 $2^n \times 2^n$ 二值图象的 P-四元树全体称为 n 年 P-四元树林. 记为 $PQ(n)$.

定义 3. 任给一个尺寸为 $2^i \times 2^i$ $(i > 0)$ 的二值图象方块，如果它的四个四分方块均是单值的，则称它为(图象)基元. 只含"0"的基元叫空基元. 只含"1"的基元叫满基元. 空基元和满基元统称为单值基元. 不是单值的基元叫半基元.

如不考虑基元的尺寸，四位码就足以表示所有的基元了. 图 1 给出了尺寸为 2×2 的所有基元. 基元的下面是它的表示码.

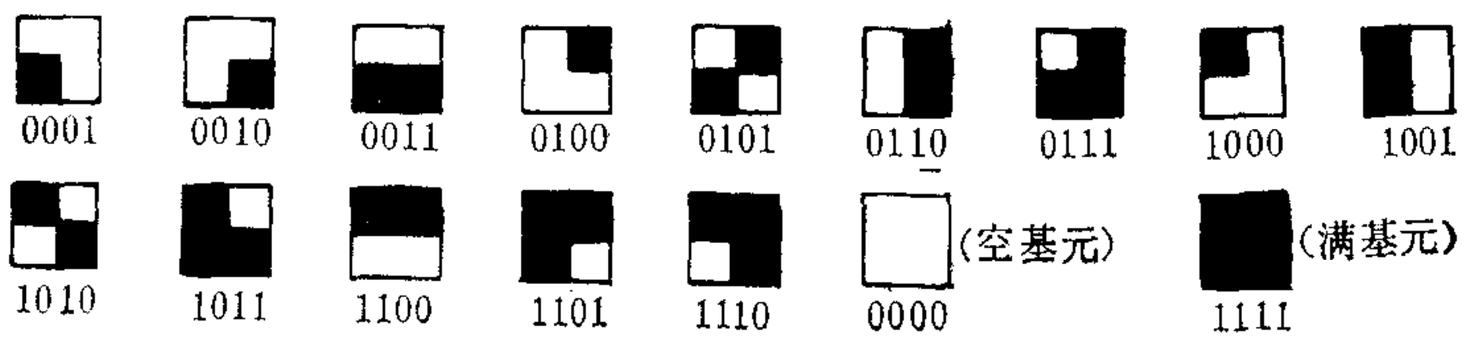


图 1

二、四元树表示的空间复杂性

本节在"每一个 $2^n \times 2^n$ 二值图象等概出现"的假定下，给出了四元树表示的平均空间复杂性，给出了典型的"背景"上有若干个"物体"的二值岩芯图象的实验数据. 理论分析和实验结果表明，对于相当大的一部分二值图象，四元树表示要比二元数组表示耗费更多的空间. 以下假定：每一个 $2^n \times 2^n$ 二值图象等概出现.

引理 1. n 年四元树林 $Q(n)$ 中的叶子结点数约为 $\frac{29}{32} \cdot 2^{2n} \cdot 2^{2^{2n}}$.

证明略.

定理 1. 四元树表示的平均存贮需要量约为 $\frac{29}{24} \cdot (10n + 7) \cdot 2^{2n}$.

证明. 设 T 是一个四元树. 由归纳易证

$$NL(T) = 3 \cdot NG(T) + 1. \tag{1}$$

对(1)式两边对 $Q(n)$ 求和得

$$NL[Q(n)] = 3 \cdot NG[Q(n)] + 2^{2^{2n}}. \tag{2}$$

易得

$$NN[Q(n)] = (4 \cdot NL[Q(n)] - 2^{2^{2n}})/3.$$

利用引理 1 得

$$NN[Q(n)]/2^{2^{2n}} \approx \frac{29}{24} \cdot 2^{2n},$$

即 $Q(n)$ 中四元树的平均结点数约为 $\frac{29}{24} \cdot 2^{2n}$.

在机器内部，图象四元树中每个结点被表为六个场. 由于四元树可能是完全的，前五个场各需 $2n + 1$ 个比特，最后一个场要 2 个比特. 故每个结点要 $10n + 7$ 个比特. 因

此，四元树表示的平均存贮需要量约为 $\frac{29}{24}\cdot(10n+7)\cdot 2^{2n}$.

对于 $n=8$ 四元树表示的平均存贮需要量是二元数组的 105 倍. 由此可以看出，对于相当大的一部分二值图象，用四元树表示比二元数组表示耗费更多的空间. 对许多区域图象的实验也充分说明了这一点. 表 1 列出了典型的背景上有若干个"物体"的二值岩芯图象的实验数据.

表 1

G	$NN(Q(G))$	$NN(PQ(G))$	$\frac{SR(Q(G))}{2^{16}}$	$\frac{SR(PQ(G))}{2^{16}}$	$\frac{SR(PQ(G))}{SR(Q(G))}$
A001Z	3413	1505	4.53	0.344	0.076
A005Z	3857	1729	5.12	0.395	0.077
A009Z	4949	2273	6.57	0.520	0.079
A013Z	6401	3001	8.50	0.686	0.080
A017Z	2929	1265	3.89	0.289	0.074
A021Z	5485	2541	7.28	0.581	0.079
A025Z	6325	2961	8.40	0.677	0.080
A029Z	5581	2589	7.40	0.592	0.079
A033Z	6293	2945	8.35	0.674	0.080
A037Z	6909	3253	9.17	0.744	0.081
A041Z	3625	1613	4.81	0.369	0.076
A045Z	5549	2573	7.37	0.580	0.079

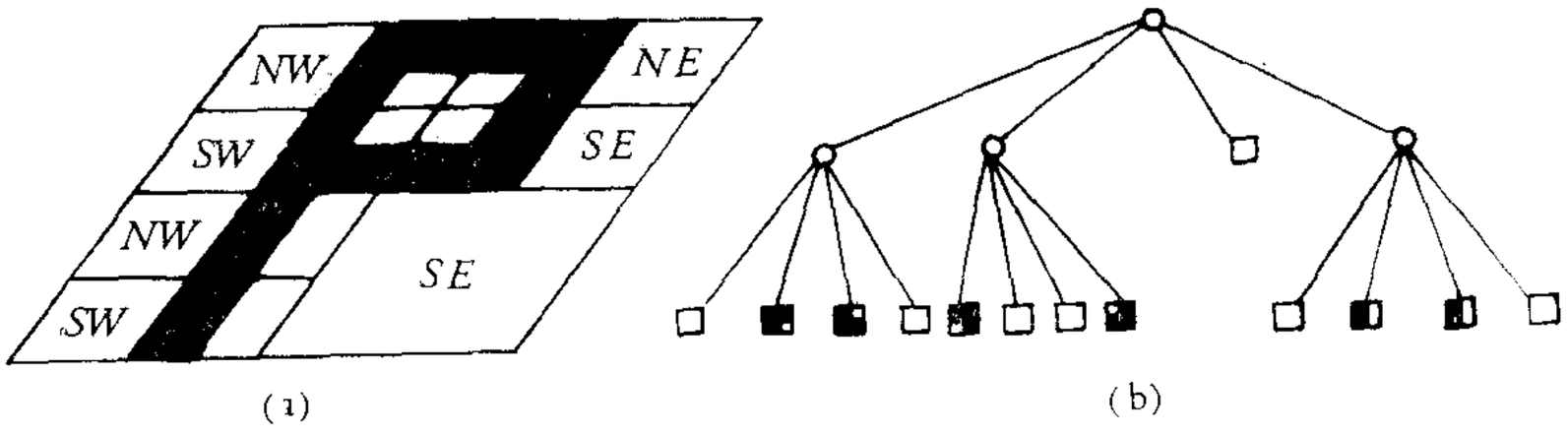
注：G 为 256×256 的二值岩芯图象，$Q(G)$ 为二值图象 G 的四元树，$PQ(G)$ 为二值图象 G 的 P-四元树，$SR(Q(G))$ 为四元树 $Q(G)$ 的存贮需要量(单位：比特)，$SR(PQ(G))$ 为 P-四元树 $PQ(G)$的存贮需要量(单位：比特).

三、P-四元树表示

1. 图象的 P-四元树

设所给的图象是一个 $2^n\times 2^n$ 二元数组. "0"代表背景点，"1"代表物体点. 所给图象的P-四元树构造如下：首先给一个树根结点，用以表示整个图象. 如果整个图象是一个基元，则这个图象的 P-四元树是一个单叶树. 否则，将图象分割成四个四分方块，给树根结点四个儿子结点，依次表示图象的西北、东北、东南和西南四分方块. 重复以上的过程于四分方块，直到不能继续分割为止. 图 2 展示了 P-四元树表示的分割过程和图象的 P-四元树.

在 P-四元树中，每个结点都表示一个图象方块. 非叶子结点称为 NPRIM 结点. 每个叶子结点表示一个基元. 代表空基元、满基元和半基元的叶子结点分别称为 EPRIM，FPRIM，PPRIM 结点. EPRIM 和 FPRIM 结点统称为 UPRIM 结点.



(a) (b)

图 2

P-四元树也可以解释为图象的一种描述树. 其中基元集合是由 16 个属性基元组成. 每个基元是一个符号对 $(L;S)$, L 是基元的句法标记, S 是一个二维词意向量(基元的尺寸,基元中"物体"的颜色).

2. P-四元树的机器内部表示

树在机器内部是由包含若干个记录的表来表示的. 设 P_T 是一个 P-四元树,表示为由 $NG(P_T)$ 个记录组成的表. 形式上,每一个非叶子结点对应一个记录. 每个记录分为四个部分,依次称为 NW, NE, SE, SW 部分, 分别存贮这个记录所对应的非叶子结点的四个儿子结点. 所存贮的儿子结点属非叶子结点的称为非叶子部分,它由两个场组成.第一个场存贮儿子结点的类型——NPRIM, PPRIM, FPRIM, EPRIM; 第二个场含有一个指示器, 指示这个儿子结点作为一个非叶子结点所对应的记录的起始地址. 所存贮的儿子结点是叶子结点的称为叶子部分, 它也是由两个场组成. 第一个场与非叶子部分的第一个场相同, 第二个场存贮的是基元的编码. 图 3 是图 2(*b*) 中的 P-四元树的机器内部表示. 可以看到,在上述的 P-四元树的内部表示中, 每个结点"寄居"在其父结点所对应的记录里,这样就避免了为叶子结点单独开辟记录而引起的空间浪费.

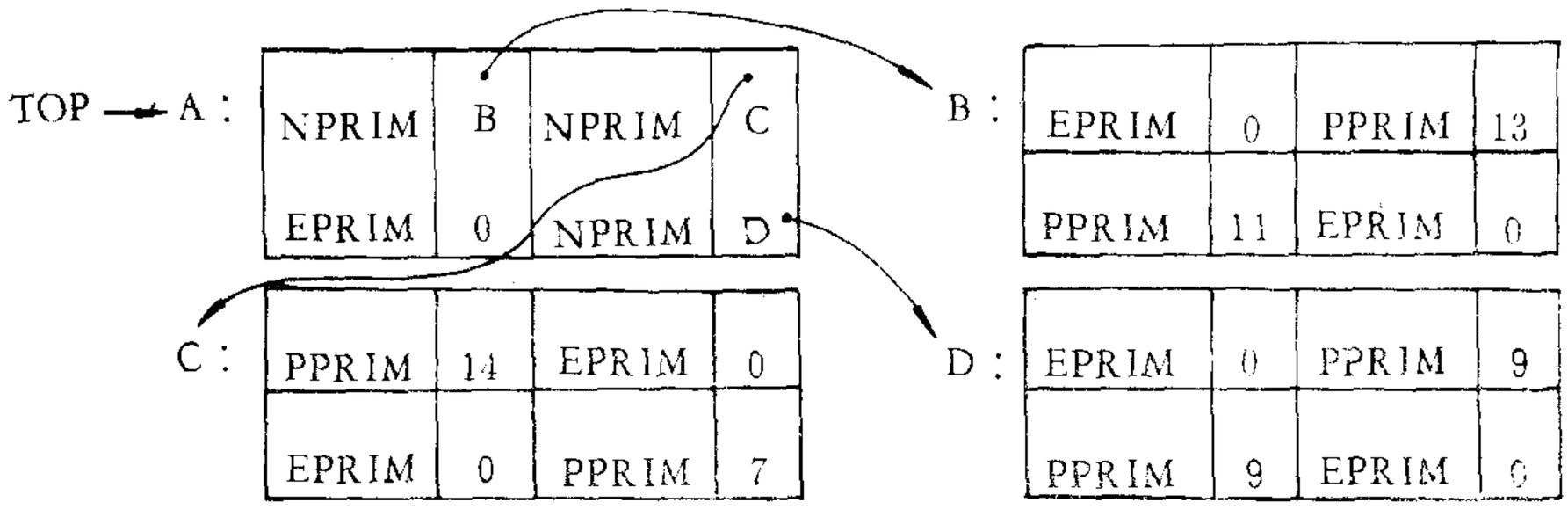


图 3

给定一个非叶子结点所对应的记录 Q 和 $I \in \{NW, NE, SW, SE\}$. 用 SON (Q, I) 表示记录 Q 的 I 部分. 给定一个部分 P, TYPE (P) 表示 P 的第一个场的内容. 如果 P 是非叶子部分, POINTER (P) 表示 P 的第二个场——指示场的内容. 如果 P 是叶子部分,用 CODE (P) 表示它的第二个场的内容——基元的编码. 另外,用 CODE(P, I) 表示 CODE (P) 的 I 位码. 完全的 P-四元树有 $(2^{2n}-1)/3$ 个结点,其中的非叶子结点数约为 $2^{2n}/12$. 故每个指示场需 $2n-3$ 比特,类型场需 2 比特,基元的表示码需 4 比特. 因此,每个非叶子部分要 $2n-1$ 比特,每个叶子部分要 6 比特. 实际上,非叶子部分要比

叶子部分长得多．为了方便，给每一个部分同样的长度——$2n-1$ 个比特．

3. *P*-四元树表示的空间复杂性及比较

引理 2. n 年 P-四元树林 $PQ(n)$ 中的叶子结点数约为 $\frac{1}{4}\cdot 2^{2n}\cdot 2^{2^2n}$.

证明略．

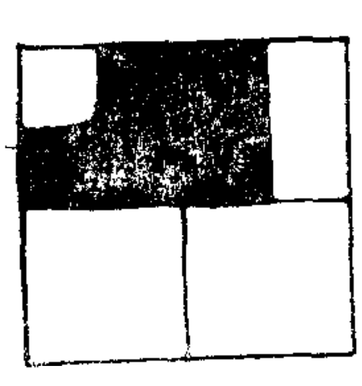
定理 2. P-四元树表示的平均存贮需要量约为 $\frac{1}{3}\cdot(2n-1)\cdot 2^{2n}$.

证明类似定理 1，故省略．

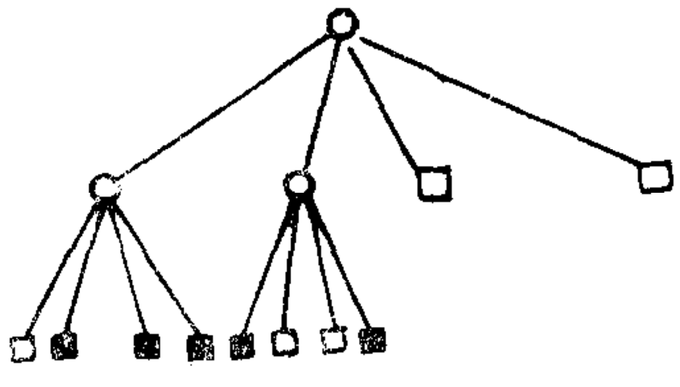
结合定理 1 与定理 2，P-四元树表示的平均存贮量与四元树表示的平均存贮量之比约为 5.52%．因此，P-四元树表示相对于四元树表示的平均数据压缩量约为 94%．表 1 对四元树、P-四元树以及二元数组表示的存贮需要量作了比较．

四、*P*-四元树表示的运算

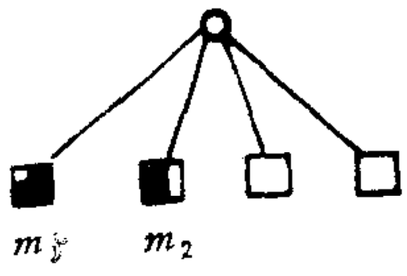
纵观现有的四元树算法[8—12]，它们的执行过程具有相同的模式——遍历四元树，在遍历的同时，对某些结点作指定的“操作”．这些指定的操作因算法而异，有的需访问先辈结点，有的则不需要．不需访问先辈结点的四元树算法[8]，可以很容易地在 P-四元树表示上建立起来，它们的执行时间与相应的四元树算法的执行时间是同阶的．为了在 P-四元树表示上实现访问先辈结点的操作，我们引进一个一维数组 $AP[1:n]$，其中 n 与 P-四元树树根的阶相同．在遍历 P-四元树时，AP 始终记着当前结点的先辈结点．如果遍历的当前结点是 k 阶的，则 $AP(i)(k<i\leqslant n)$ 是有意义的，它存放着当前结点的 i 阶先辈结点，称 AP 为先辈链．遍历一次 P-四元树，保持当前结点的先辈链所需的记忆次数与 P-四元树中的非叶子结点数相同．显然，以如此小的时间代价换取可观的空间，对于空间优先的系统来说是值得的．最多用两个先辈链，文献[9—12]的算法就可以移植到 P-四元树表示上来，且算法复杂性的阶不变．进一步，P-四元树表示还有其整体运算优越性．这里以计算图 4(a) 中图象区域周长为例，说明 P-四元树表示的整体运算优越性．计算图 4(a) 中区域周长的 P-四元树算法执行如下：遍历图 4(a) 中的 P-四元树，遍历到结点 m_1 时，访问其四个方向上的邻结点，从而确定基元 m_1 的“外周长”．遍历到结点 m_2 时，只需访问其三个方向上的邻结点．基元的“内周长”由基元本身决定．访问结点的个数为 16．如利用文献[9]中的算法，对每一个黑叶子结点都要访问其四个方向上的邻结点．不难计算，利用文献[9]中算法，计算图 4(a) 中区域周长所需访问的结点数为 64．由此可见，P-四元树表示的整体运算特性，使得 P-四元树算法访问结点的个数大大减少了，从而



(a)



(b)



(c)

图　4

提高了运算速度.

五、结 束 语

P-四元树表示作为一种新的表示方法，具有紧凑性和运算优越性. 在硬件上和软件上实现 *P*-四元树中基元之间的直接运算，以及在 *P*-四元树表示上实现新的图象处理运算，仍是继续研究的方向. 在三.1 中讨论了 *P*-四元树表示与图象描述树的关系. 以 *P*-四元树作为图象描述树，结合已较为成熟的句法方法，作一些模式识别方面的工作更是有趣的探讨.

本文是在潘君卓副教授和孙靖老师的指点下完成的，并承蒙程民德教授和石青云教授的细心审阅，作者在此表示衷心感谢.

参 考 文 献

[1] Klinger, A., Data Structure and Pattern Recognition, Proc. 1 IJCPR, 1973, 119—122.

[2] Klinger, A. and Dyer, C. R., Experiments in Picture Representation Using Regular Decomposition, *Computer Graphics and Image Processing* **5**(1976), 68—105.

[3] Alexandridis, N. and Klinger, A., Picture Decomposition, Tree Data-Structure, and Identifying Directional Symmetries as Node Combinations, *Computer Graphics and Image Processing* **8**(1978), 43—77.

[4] Klinger, A. and Rhodes, M. L., Organization and Access of Image Data by Areas, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **1**(1979), 50—60.

[5] Hunter, G. M. and Steiglitz, K., Operations On Images Using Quad Trees, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **1**(1979), 145—153.

[6] Samet, H., Region Representation: Quadtrees From Binary Arrays, *Computer Graphics and Image Processing* **13**(1980), No. 1, 88—93.

[7] Samet, H., Region Representation: Quadtrees From Boundary Codes, *Comm. CAM* **23**(1980), 163—170.

[8] Shneier, M., Linear-Time Calculations of Geometric Properties Using Quadtrees, *Computer Graphics and Image Processing* **16**(1981), 296—302.

[9] Samet, H., Computing Perimeters of Images Represented by Quadtrees, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **3**(1981), 683—687.

[10] Dyer, C. R., Computing the Euler Number of an Image From its Quadtree, *Computer Graphics and Image Processing* **13**(1980), 270—276.

[11] Samet, H., Connected Component Labelling Using Quadtrees, *J. ACM* **28**(1981), 487—501.

[12] Samet, H., A Distance Transform for Images Represented by Quacltrees, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **4**(1982), 298—303.

P-QUADTREE REPRESENTATION

YAN CHENGWEN

(*Department of Mathematics, Peking University*)

ABSTRACT

A new method for region representation———*P*-quadtree representation is proposed in this paper. Compared with quadtree representation, the average space saving of *P*-quadtree representation is about 94%. Various operations can be performed on *P*-quadtree representation as fast as on quadtree representation. Furthermore, the primitive by primitive operation mode of the *P*-quadtree representation makes it more efficient. Actually, *P*-quadtree is a promising data structure for image processing systems based on microcomputers.