MINISTÉRIO DA CIÊNCIA E TECNOLOGIA
INSTITUTO DE PESQUISAS ESPACIAIS

AUTORIZAÇÃO PARA PUBLICAÇÃO
AUTHORIZATION FOR PUBLICATION

PALAVRAS CHAVES/KEY WORDS

AUTORES/AUTHORS

Quadtrees
Estruturas de árvores
Representaçao de imagens
Operaçoes sobre imagens
Computação Gráfica
Processamento de Imagens

AUTORIZADA POR/AUTHORIZED BY

Ralf Gielow
Pres. Cons. Pós-Graduação

AUTOR RESPONSAVEL
RESPONSIBLE AUTHOR

João Pedro C. Cordeiro

DISTRIBUIÇÃO/DISTRIBUTION

- [ ] INTERNA / INTERNAL
- [x] EXTERNA / EXTERNAL
- [ ] RESTRITA / RESTRICTED

REVISADA POR / REVISED BY

Valter Rodrigues

CDU/UDC

621.376.5

DATA / DATE

ABRIL, 1990

PUBLICAÇÃO Nº
PUBLICATION NO

INPE-5069-TDL/408

ORIGEM
ORIGIN

PG/CPD

PROJETO
PROJECT

FRH/CAP

Nº DE PAG.
NO OF PAGES

81

ULTIMA PAG.
LAST PAGE

70

VERSÃO
VERSION

Nº DE MAPAS
NO OF MAPS

TÍTULO/TITLE

Representação e Manipulação de Imagens Gráficas utilizando Árvores Quaternárias

AUTORES/AUTHORSHIP

João Pedro Cerveira Cordeiro

RESUMO-NOTAS/ABSTRACT-NOTES

Conversão entre representações, pesquisa de adjacência, operações de conjunto e transformações são tarefas básicas em sistemas que manipulam imagens gráficas em diversas áreas de pesquisa e aplicação. Neste trabalho são elaborados e implementados procedimentos para pesquisa de adjacência, operações de união, intercessão e rotação por ângulos retos, sobre imagens representadas por árvores quaternárias (quadrees). A geração de árvores quaternárias a partir de imagens dadas nas formas matricial, raster e vetorial, e da representação raster de uma quaternária de imagem, são abordadas neste trabalho. Alguns aspectos da estrutura de árvore quaternária clássica e linear são considerados.

OBSERVAÇÕES/REMARKS

Dissertação de Mestrado em Computação Aplicada, aprovada em 12 de setembro de 1989.

Aprovada pela Banca Examinadora
em cumprimento a requisito exigido
para a obtenção do Título de Mestre
em Computação Aplicada

Dr. Flávio Roberto Dias Velasco

Presidente

Dr. Valter Rodrigues

Orientador

Dr. Clésio Saraiva dos Santos

Membro da Banca
-Convidado-

Dr. Luiz Alberto Vieira Dias

Membro da Banca

Dr. José Antonio Gonçalves Pereira

Membro da Banca

Candidato: João Pedro Cerveira Cordeiro

São José dos Campos, 12 de setembro de 1989

Ao Chippe, à Nonna e Anna.

AGRADECIMENTOS

Agradeço ao Dr. Valter Rodrigues pela orientação e confiança que trouxe a certeza de que este trabalho teria um final feliz.

À Marisa da Motta pelo carinho e apoio, além de sua efetiva colaboração.

A Guaracy Erthal pela sua paixão por imagens e valiosa fonte de referências.

Aos colegas do CPD, particularmente a Antônio Mariano Gomes e Miguel Adrian, pelo incentivo.

À Coordenação dos Cursos de Pós-Graduação do INPE, em particular a Adélio Gurgel do Amaral Neto.

Agradecimentos especiais àqueles que têem me incentivado em todas as fases de minha vida, Potyguara e Teresinha.

ABSTRACT

Convertion between representations, neighbour finding and set operations are basic tasks for systems that deal with graphic images in several research and application areas. This work focuses on the implementation of some procedures for neighbour finding, basic set operations of union and intersection, rotation by right angles, involving images represented by quadtrees, as well as the generation of a quadtree structure from the matrix, raster and vector format. The task of obtaining a raster image from its quadtree is also implemented. Some aspects of the classical and linear quadtree structure are considered.

SUMÁRIO

LISTA DE FIGURAS

CAPÍTULO 1

INTRODUÇÃO

1.1 - REPRESENTAÇÃO DE IMAGENS

Uma **cena**, em princípio, poderia ser descrita através de uma distribuição contínua de intensidades e freqüências de energia associadas a cada ponto do espaço tridimensional. Nesse contexto, uma imagem pode ser definida como uma restrição bidimencional de uma tal distribuição. A captação de uma imagem através de sensores permite a obtenção de informação, que pode ser chamada informação visual. As retinas dos olhos, por exemplo, fornecem informação de extrema importância para a sobrevivência no mundo animal; a gelatina de um filme fotográfico capta a informação visual a partir de um processo químico; sensores eletrônicos em satélites registram o brilho refletido ou emitido por pontos na superfície da Terra, compondo imagens úteis em áreas de aplicação diversas, como o Monitoramento da Superfície e da Atmosfera Terrestre; a câmara de TV permite o desenvolvimento de interessantes aplicações em Visão Computacional, como o modelamento do universo visual de robôs industriais.

Matematicamente (Mascarenhas e Velasco, 1984), pode-se definir imagem como uma função que mapeia um ponto (x,y) de um subconjunto do plano, que será chamado plano de imagem, em um valor I(x,y), associado à intensidade e cor nesse ponto. Pode-se notar que uma tal definição permite desvincular o conceito de imagem ao de cena, o que permite que imagens arbitrárias possam ser geradas diretamente no plano de imagem.

Para que uma imagem possa ser efetivamente manipulada por computador, deve estar em algum formato digital de dados. Imagens analógicas, como as obtidas por sensores químicos, são **digitalizadas** por uma técnica que consiste na aplicação de um processo conhecido como **amostragem**, a fim de determinar as dimensões dx e dy de elementos de área que integram a área total de definição da imagem. Sobre esses elementos, regularmente espaçados, chamados pixels (de "picture element") é aplicado um processo conhecido por **quantização**, que consiste na tomada das médias de valores de intensidades por pixel, discretizada em múltiplos de um valor dI. A imagem resultante desse processo é conhecida por **imagem digital** e os valores associados à quantidade dI são identificados em níveis de cinza ou cores por números inteiros positivos. Essas imagens são armazenadas em memórias secundárias, em geral sob a forma de arquivos sequenciais, onde cada registro corresponde a uma linha horizontal da imagem. Esse formato de dados de imagem será denominado neste trabalho como formato **raster**.

Outro formato de dados de imagem útil em aplicações que envolvem a representação de feições lineares em imagens, como pontos e linhas, é o formato **vetorial**, onde dados como vértices, interseções e segmentos de reta devem ser registrados e já constituem naturalmente um formato digital.

Imagens digitais podem ser obtidas diretamente sobre um plano de imagem, sem nenhuma vinculação com cenas. A classe de imagens correspondente caracteriza as aplicações típicas da área de **Computação Gráfica**, embora o termo imagem gráfica possa ser usado ao se referir a qualquer formato digital de imagem. Muitas aplicações em áreas como Computer Aided Design, Processamento de Imagens, Inteligencia Artificial,

Reconhecimento de Padrões, e outras. Até mesmo o desenho de interfaces de aplicativos, em geral exige a representação, visualização e manipulação de imagens gráficas.

Uma representação comum em processamento de imagens e reconhecimento de padrões é conhecida como **imagem binária.** Aqui apenas dois valores são associados aos pixels, por exemplo, 0 ou 1, indicando, respectivamente os pixels que pertencem ou não a um objeto ou região numa imagem.

O formato raster permite modelar uma imagem gráfica como uma coleção de pixels organizados no plano de imagem. A representação mais natural desse formato é uma matriz bidimensional de valores que identificam cores, ou níveis de cinza ou alguma outra classe de informação de interesse. Sobre essa matriz é, em geral, imposto um sistema de coordenadas, de tal modo que a origem (0,0) é associada ao canto superior esquerdo da imagem. O pixel diagonalmente oposto, numa imagem de dimensões $2^n x 2^n$, terá as coordenadas $(2^n-1, 2^n-1)$.

No caso do formato vetorial, usado para representar pontos e linhas definidas por seqüências de segmentos conexos de reta, onde não existe a noção de cor, a representação mais natural é dada pela lista de coordenadas de vértices dos segmentos que descrevem a imagem. Outro formato em uso corrente é o **comprimento de carreira**, uma variação do formato raster, que permite representar cada linha de uma imagem como uma seqüência de pares de números, um dos quais indica a cor ou nível de cinza associado a cada pixel de uma carreira, isto é, um segmento de uma linha, composto de pixels de mesma cor; o outro número indica seu comprimento em número de pixels adjacentes. Outro formato, muito usado para representar bordas de regiões é o **código de cadeia**, que modela a borda

como uma seqüência de dígitos que representam direções tomadas a cada pixel no percorrimento de uma borda.

Klinger (1971) utiliza, a fim de minimizar o número de regiões pesquisadas, necessárias para localizar objetos em uma cena, um esquema para divisão de imagens em blocos regulares, combinando a divisão da imagem alternadamente em quatro ou nove blocos. A divisão em quatro blocos permite determinar características de simetria enquanto a divisão em nove blocos permite detectar regiões adjacentes, possivelmente informativas. A aplicação proposta envolvia a representação de objetos no espaço de visão de um robô, a fim de permitir seu deslocamento O princípio utilizado veio a ser conhecido por Princípio da **Decomposição Regular.**

Na maioria de suas aplicações, esse princípio implica na subdivisão da imagem em quatro quadrantes que podem ser ou não subdivididos em subquadrantes, conforme os pixels de sua região satisfaçam ou não algum critério pré estabelecido de homogenidade. O princípio de decomposição de imagem resultante permite o modelamento de imagens por estruturas de árvore de grau quatro, que será discutido nesse trabalho, e cujas aplicações estendem-se a qualquer área onde exista a necessidade de manipular dados gráficos.

A partir da premissa abordada por Harmon (1973) de que a informação visual particionada por áreas é suficiente para pessoas no reconhecimento de faces humanas, Rhodes e Klinger (1976) propõem um sistema gráfico interativo para a obtenção de retratos falados a partir de descrições de feições e sua subsequente modificação. O princípio da decomposição regular é utilizado para obter modelos condensados e aproximados de imagens raster que possam ser usado em memórias rápidas, e como ferramenta

para implementar programas que acessem, modifiquem ou emitam porções isoladas de imagens.

1.2 - PROPÓSITO DO TRABALHO

Este estudo sobre representações de imagens gráficas baseadas em estruturas de árvores quaternárias, pretende demonstrar a viabilidade de sua utilização na implementação de tarefas e operações básicas em Computação Gráfica e áreas correlatas. O trabalho não se detém a aplicações e áreas específicas, procurando abordar apenas aspectos da representação e manipulação, embora muito da pesquisa desenvolvida a respeito, e do material utilizado, esteja voltado para aplicações em Processamento de Imagens, Reconhecimento de Padrões, Visão Computacional, além da Computação Gráfica.

Uma aplicação onde o tipo de representação aqui abordado tem sido extensivamente explorado é em Sistemas Geográficos de Informação (Samet et al., 1984; Peuquet, 1984), onde são mantidos dados de mapas que representam regiões conexas, cujos pixels estão associados a um valor comum de classe, que determina um critério natural de homogenidade, facilitando sua representação por árvores. Esquemas de endereçamento podem ser obtidos com base em estruturas de árvores (Klinger e Rhodes, 1979). Esses **mapas temáticos**, como são conhecidos em cartografia, em geral são dados por linhas poligonias que definem bordas de regiões de imagem e podem ser modelados no formato vetorial. Representações baseadas na decomposição regular têm sido pesquisadas e implementadas para essa classe de imagens, cujas caracteristicas de homogenidade favorecem sua representação por árvores.

Um sistema que utilize imagens gráficas representadas por árvores exige que sejam previstos

procedimentos para conversão entre as diversas representações em uso corrente e a representação por árvores quaternárias. Neste trabalho é abordado a geração, a partir das representações matricial e raster, baseado em trabalhos de Samet (1980b, 1981a). Também é apresentado um procedimento para conversão no sentido oposto, de uma árvore quaternária para a representação raster da imagem correspondente, implementado neste trabalho com base na proposta de Samet (1984a). Este procedimento permite que se mantenha a compatibilidade com os meios mais comuns de armazenamento e dispositivos de vídeo, em geral baseados na representação raster.

Visando a representação de dados vetoriais, foram extraídos conceitos da estrutura proposta por Samet et al. (1985) para representar feições lineares de mapas poligonais em um sistema geográfico de informação, denominada **árvore quaternária de segmentos**.

Nas seções que se seguem são apresentados os conceitos básicos de Decomposição Regular, Árvore Quaternária convencional ou sob a forma linearizada. Também algumas aplicações são apontadas. No Capítulo 2 é discutida a manipulação de árvores quaternárias através da pesquisa de adjacência entre os nós, rotação por ângulos retos e operações de conjunto. No Capítulo 3, são abordados procedimentos para conversão de uma imagem matricial ou raster para a estrutura de árvore equivalente e da representação por árvore quaternária para a representação raster. O Capítulo 4 discute a representação por estruturas de árvore, de linhas poligonais dadas por sua representação vetorial.

As tarefas, operações e funções discutidas a partir do Capítulo 2 foram implementadas neste trabalho e aplicadas sobre imagens sintéticas, no sentido de que suas

representações são geradas diretamente, em uma matriz binária, na memória principal. As árvores que as representam também poderiam ser chamadas sintéticas, por existirem apenas como estruturas montadas temporariamente em memória. As técnicas de armazenamento e acesso a imagens representadas por árvores, embora não sejam o tema central deste trabalho, têm sido fonte de muitas pesquisas e aplicações em bancos de dados cartográficos (Samet, 1985), compressão e transmissão de imagens (White, 1987).

A eficiência espacial e temporal dos procedimentos desenvolvidos não é analizada neste trabalho. Alguns dados obtidos por outros autores e referências sobre esse aspecto são fornecidos a fim de orientar o leitor interessado.

1.3 - PRINCÍPIO DA DECOMPOSIÇÃO REGULAR

O princípio da **decomposição regular** consiste na subdivisão de uma matriz de imagem em quatro sub-matrizes, que correspondem a seus quadrantes, os quais são tratados como novas imagens, subimagens da inicial. Um **quadrante** é parte de uma matriz de imagem, que usualmente corresponde a um quarto, embora qualquer outra fração (Klinger, 1971; Ahuja, 1983) possa ser usada. Além disso, é desejável (Samet, 1984b) que a partição seja um modelo infinitamente repetitivo e decomponível, para que possa ser usado em imagens de qualquer tamanho ou resolução. Desse modo uma imagem fica organizada em **níveis de subdivisão** constituídos de regiões disjuntas, que serão chamadas genericamente de **blocos**, cuja união, a cada nível, é a imagem inteira. O termo regular indica o fato de que os quadrantes, partes de um bloco, são de mesma forma e tamanho.

Em uma imagem dada por uma matriz $2^n x 2^n$, o primeiro nível de subdivisão consiste em quatro blocos com $2^{n-1} x 2^{n-1}$ pixels, associados aos quadrantes da imagem inicial, e rotulados pelas **direções secundárias** NO, NE, SO e SE. O nível k de subdivisão de uma imagem consistirá de 2^{2k} blocos, matrizes binárias $2^{n-k} x\ 2^{n-k}$, que a compõem. A Figura 1.1 ilustra esse processo para uma matriz $2^3 x 2^3$ que representa uma imagem binária.

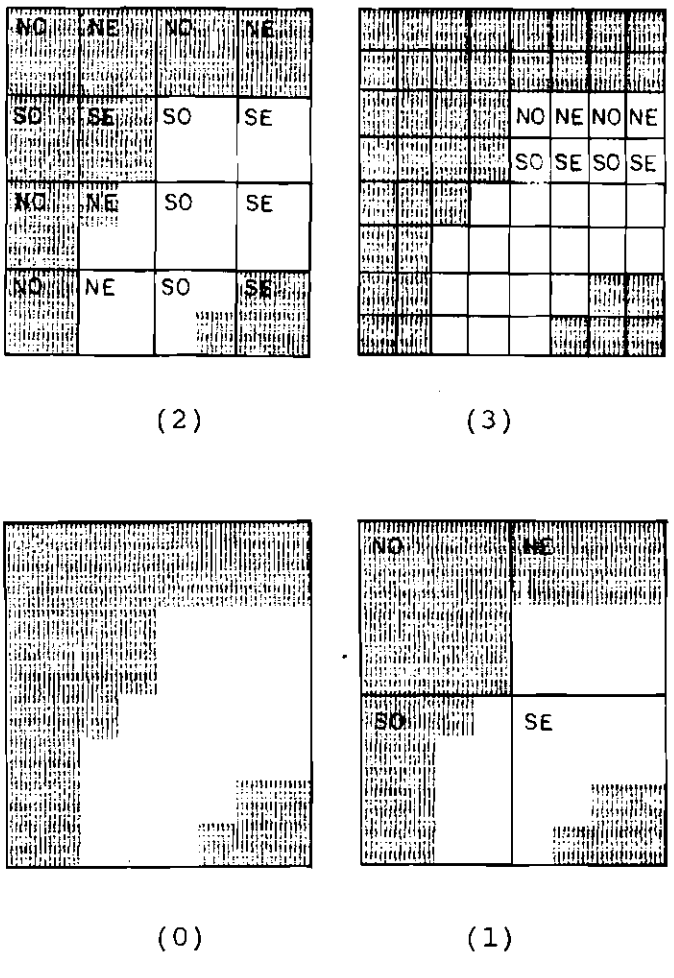

Fig. 1.1 - Níveis de subdivisão de uma imagem $2^3 x 2^3$.

A vantagem da decomposição regular é permitir mecanismos para localização de subconjuntos, onde tarefas

gráficas específicas possam ser resolvidas de maneira mais simples. Para isso, é preciso que se estabeleça critérios para essa decomposição em função da aplicação específica a que se destina. Em geral, são utilizados **critérios de homogenidade** quanto a cor ou níveis de cinza.

Klinger e Dyer (1976) utilizam o princípio descrito em Processamento de Imagens, a fim de representar imagens complexas e analisar seu efeito sobre procedimentos para segmentação de imagens em regiões a serem identificadas por um valor de classe que indica alguma propriedade dessas regiões. O processo envolve a identificação de subconjuntos ou objetos de interesse em uma imagem digitalizada. A técnica proposta visava o descarte de áreas não-informativas para o objetivo da classificação, através de um critério de homogenidade dado por uma função discriminante que, aplicada a pixels de um bloco, podia retornar um entre os valores: "informativo", "não-informativo" e "incerto", o que determinava a subdivisão ou não do bloco em quadrantes. Era proposta a representação de uma imagem por uma árvore de grau quatro, contendo apenas nós associados a blocos considerados "informativos" (terminais) ou "incertos" (intermediários).

Dois tipos de problemas estratégicos surgem no modelamento de cenas. O primeiro é a comunicação com o usuário de sistemas gráficos. O segundo é a determinação do subconjunto visível da cena. O primeiro tem sido abordado, em parte, pela observação de que o universo pode ser organizado hierárquicamente em objetos compostos de objetos que, por sua vez são compostos de outros objetos, e assim por diante. Essa observação tem sido usada para a organização de interfaces entre usuários e dados, desde os mais antigos até os mais recentes projetos de sistemas gráficos (ANSI, 1985). Como esta hierarquia de objetos deve ser usada na resolução de problemas de comunicação, é

natural que o seja também na abordagem do problema do subconjunto visível. O princípio da decomposição regular oferece uma hierarquia no espaço da imagem, que facilita um efetivo modelamento hierárquico do espaço de objetos de uma cena. Uma vantagem desse princípio é permitir que a resolução seja tomada como uma variável dependente da complexidade dos objetos contidos na imagem.

1.3.1 - ÁRVORE QUATERNÁRIA DE UMA IMAGEM

Procedimentos sobre árvores, em geral, se baseiam nas técnicas de percurso em profundidade (pós-ordem), onde um nó só é visitado após terem sido todos os seus nós filhos, e em largura (pré-ordem), onde o nó raiz de cada subárvore é visitado antes de seus nós filhos (Knuth, 1973). Procedimentos sobre árvores diferem pela ação a ser executada sobre cada nó da estrutura e pelo seu conteúdo. Na maioria das linguagens de programação, o percurso de nós de uma árvore ou o processo de subdivisão, podem ser implementados através do uso de recursividade.

O princípio da decomposição regular permite modelar uma imagem como uma estrutura de **árvore quaternária**, denominada em geral por **"quadtree"**. Se a imagem inteira for homogênea, a árvore consiste em um único nó terminal, com o registro da informação associada a seus pixels. Caso contrário, é subdividida e associada a um nó raiz com quatro nós filhos que serão associados aos subquadrantes: NO, NE, SO, SE, dessa imagem. O processo é então repetido para cada subquadrante desse primeiro nível de subdivisão da imagem, e cada bloco de imagem é tratado como uma imagem inteira e, assim, sucessivamente. Os nós-folha representam blocos homogêneos de imagem e os intermediários, blocos que necessitam de subdivisão em quadrantes, cujas características de posição e tamanho são determinadas pelas características da imagem inicial e pelo

nível de subdivisão ou de árvore. Cada **nível de árvore** é identificado a exatamente um nível de subdivisão da imagem.

Uma representação exata de uma imagem em escala de cinza pode ser obtida se for utilizado como critério de homogeniade o valor do nível de cinza de cada pixel, embora possa não ser econômico para esse tipo de imagem, onde são raras as ocorrências de grandes blocos homogêneos. Nesse caso, pode-se considerar **representações aproximadas**, como em Tanimoto e Pavlids (1975).

Uma seqüência de aproximações de uma imagem em escalas de cinza pode ser obtida se cada nó intermediário de sua representação exata informar o valor médio dos níveis de cinza da área representada, ou o nível predominante. Cada nível mais baixo pode assim fornecer uma descrição mais precisa da imagem. Rosenfeld (1980) utiliza a média ponderada dos valores de níveis de cinza em um certo bloco. White (1987) define uma cor para nós intermediários quando alguma cor é predominante no bloco.

Para representar regiões de imagem, cada nó pode ser implementado por um registro com seis campos, onde cinco são ponteiros para quatro nós filhos e para o nó pai. O sexto campo informa o valor associado aos pixels do bloco correspondente ao nó que está sendo representado, dado por alguma informação de interesse como a cor ou o nível de cinza. Numa imagem binária, esse campo informa apenas um entre os valores, **branco**, **preto**, ou **cinza**.

Se os pixels de um bloco em uma imagem binária são todos brancos ou pretos, exclusivamente, este será representado por um nó terminal (folha) com o valor **branco** ou **preto** indicado. Caso contrário, será representado por um nó não-terminal com ponteiros para quatro nós filhos, raízes das subárvores quaternárias associadas aos

quadrantes nas direções secundárias NO, NE, SO e SE da imagem original, com o valor **cinza** indicado. A Figura 1.2 mostra uma imagem 2^3x2^3 e sua árvore quaternária. Os rótulos em cada bloco representado indicam suas posições relativas nos níveis de subdivisão em que estão definidos.

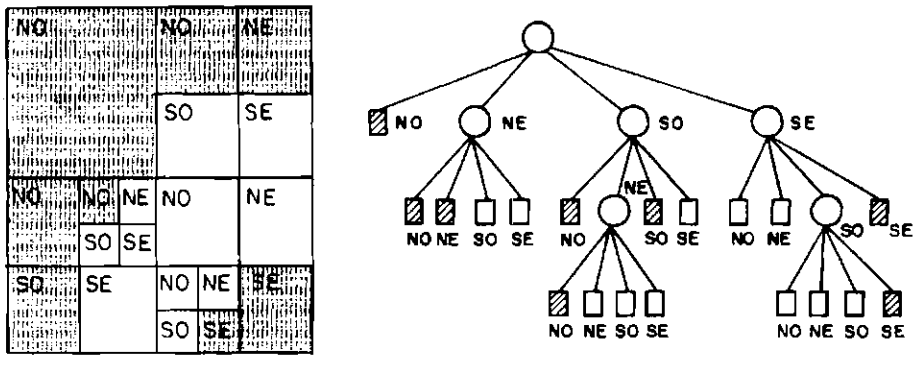

Fig. 1.2 - Decomposição regular e árvore associada.

Um bloco no nível k de subdivisão fica associado a um nó de nível n-k da estrutura de árvore que representa uma imagem 2^nx2^n, e corresponde a um entre os 2^{2k} possíveis blocos $2^{n-k}x2^{n-k}$ que a compõem. Os pixels são representados pelos nós de nível 0 da árvore. Desse modo, cada nível determina um grau de resolução da imagem, o que possibilita a representação de imagens aproximadas, desde que os nós **cinza** também informem um valor, que corresponda à média dos valores em seus nós filhos.

Para representar dados lineares como pontos, segmentos e bordas o nó deve informar sobre coordenadas e/ou coeficientes angulares, e os critérios de subdivisão tornam-se mais complexos do que no caso regional.

1.3.2 - ÁRVORE QUATERNÁRIA LINEAR DE IMAGENS

Um bloco no nível k de subdivisão de uma imagem $2^n x 2^n$ pode ser identificado à seqüência de quadrantes associados aos blocos que o contêm, nos níveis anteriores de subdivisão, a partir da imagem inteira. Deste modo, a seqüência $\{S_1, \ldots, S_k\}$, resultante, onde S_i = NO, NE, SO, ou SE, é suficiente para determinar as características de posição e tamanho do bloco. A posição é dada pelos tipos de quadrante na seqüência, enquanto o tamanho, é dado em função da cardinalidade da seqüência. Uma seqüência de k elementos determina um bloco $2^{n-k} x 2^{n-k}$.

Através da associação dos digitos 0, 1, 2, 3 aos tipos de quadrante NO, NE, SO, SE, respectivamente, a seqüência de quadrantes que localiza um bloco permite determinar um número inteiro que, expresso na base 4 de enumeração, implicitamente fornece as características de tamanho e posição de blocos de imagem. Esse número será referido nesse trabalho como o **código quaternário** associado a um bloco de imagem; os dígitos que o definem serão referidos como códigos locacionais. A coleção de tais códigos quaternários fornece um modelo alternativo para representar imagens, conhecido por Árvore Quaternária Linear (Gargantini, 1982), e possibilita economia de memória, pois, nesse caso os nós intermediários são dispensáveis na estrutura resultante. Em imagens binárias uma das regiões, **preta** ou **branca**, também é dispensável.

O problema de associar pixels a códigos quaternários pode ser generalizado para o de associar pares ordenados de números inteiros ao conjunto dos inteiros positivos. O conjunto de pares ordenados é contável, isto é, existe uma função biunívoca que associa a cada inteiro positivo um único par ordenado de inteiros. A demonstração desse fato pode ser obtida considerando-se a representação

binária dos elementos de um par ordenado inteiros. Assim, um par (x,y) terá suas coordenadas dadas por:

$$x = x_1x_2...x_n \text{ e } y = y_1y_2...y_n \; ; \quad x_i \text{ e } y_i \text{ em } \{0, 1\}.$$

Uma função biunívoca que enumera esse conjunto, é obtida pela associação de um inteiro positivo, z, que pode ser expresso na base 4 de enumeração por:

$$z = z_1z_2...z_n \; ; \quad z_i \text{ em } \{0, 1, 2, 3\},$$

cujos dígitos são obtidos de x e y através da associação:

$$z_i = y_ix_i.$$

Esta função, restrita a uma matriz 2^3x2^3, é ilustrada pela Figura 1.3, junto à relação dos valores de sua aplicação para os 10 primeiros inteiros positivos.

	z_{10}	z_4	z_2	$(x,y)_2$	$(x,y)_{10}$
	0	00	0000	(00,00)	(0,0)
00000101001110010111011	1	01	0001	(01,00)	(1,0)
00200301201310210311211	2	02	0010	(00,01)	(0,1)
02002103003112012113013	3	03	0011	(01,01)	(1,1)
02202303203312212313213	4	10	0100	(10,00)	(2,0)
20020121021130030131031	5	11	0101	(11,00)	(3,0)
20220321221330230331231	6	12	0110	(10,01)	(2,1)
22022123023132032133033	7	13	0111	(11,01)	(3,1)
22222322233322323332333	8	20	1000	(00,10)	(0,2)
	9	21	1001	(01,10)	(1,2)

Fig. 1.3 - Códigos quaternários associados a pixels.

Essa função, restrita a um plano de imagem $2^n x 2^n$, pode ser útil para determinar o código quaternário a partir das coordenadas de um pixel e vice versa. Neste trabalho, essa função é utilizada na Seção 3.3, na localização do nó que determina um segmento de uma linha em uma representação raster. Na Seção 4.4 ela é utilizada para a determinação do nó cujo bloco contém um certo segmento e do caminho desde a raiz até um certo nó, a partir das coordenadas do primeiro pixel do bloco associado.

Por exemplo, na decomposição da região mostrada na Figura 1.4, o número inteiro decimal que rotula cada bloco, quando expresso na base quatro, corresponde ao código quaternário do seu primeiro pixel e determina, junto à informação referente ao nível de subdivisão a que pertence, a seqüência de códigos locacionais que identifica o caminho a ser percorrido desde a raiz até o nó associado ao bloco em uma estrutura de árvore quaternária convencional.

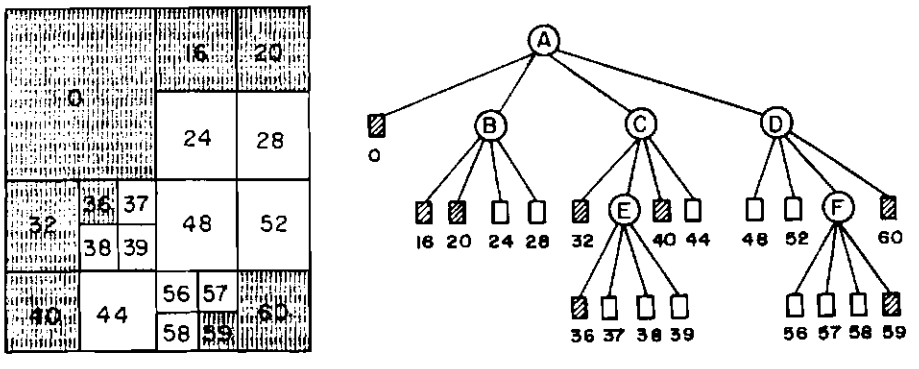

Fig. 1.4 - Códigos quaternários associados a blocos.

Uma das interessantes propriedades desses códigos é que permitem definir sistemas de endereçamento de nós onde a ordem crescente de endereços corresponde ao

percurso em pré-ordem da árvore convencional que os represente. Uma das primeiras utilizações desses códigos foi desenvolvida por Morton (1966) na indexação de dados geodéticos. O esquema foi efetivamente aplicado em representação de imagens por Gargantini (1982). O armazenamento em bancos de dados geográficos tem fornecido também muitas aplicações (Samet et al., 1984; Peuquet, 1984).

Como muitas operações sobre árvores quaternárias podem ser realizadas em tempo proporcional ao número de nós, pode ser vantajoso, em termos de velocidade, a manipulação de imagens dadas por estruturas de árvore.

1.4 - APLICAÇÕES

O armazenamento de imagens em arquivos organizados por códigos quaternários é uma das aplicações mais simples e imédiatas, representando uma significativa economia de memória, facilitando a manipulação de imagens por sistemas de configuração limitada e o endereçamento de pixels e blocos em memórias secundárias. Klinger et al. (1979) utilizam o princípio de subdivisão no desenvolvimento de métodos eficientes de recuperação de áreas em imagens armazenadas em arquivos seqüênciais e de acesso direto. Samet et al. (1984) utilizam um sistema de endereçamento em bancos de dados geográficos, onde o endereço de um bloco é dado pelo código quaternário derivado das coordenadas (x,y) de seu primeiro pixel (correspondente ao canto superior esquerdo) e do nível na árvore.

Em operações como o cálculo de propriedades geométricas de imagens, a representação por árvores tem se mostrado bastante útil. Dyer (1980) apresenta um procedimento que calcula o número de Euler de uma imagem

binária em tempo proporcional ao total de nós terminais da estrutura; Atkinsons e Gargantini (1985) implementam a contagem de regiões e "buracos", e o nível em que estejam aninhados, em imagens binárias, em tempo proporcional ao número de componentes conexas multiplicado pelo de pixels pertencentes às bordas; Samet (1981c) propõe um procedimento para o cálculo do perímetro de regiões em tempo proporcional ao número de nós-folha na estrutura.

A viabilidade de implementar um Sistema Geográfico de Informação baseado em representações por árvores quaternárias é demonstrada num trabalho de Samet et al. (1984), onde são mantidas representações para dados de três tipos: pontos, linhas e regiões. O sistema proposto consiste em quatro níveis de programas. O nível mais baixo, o núcleo, controla a interface entre arquivos de árvores quaternárias e programas que manipulam imagens. O nível seguinte implementa comandos de edição e funções primitivas de banco de dados. O terceiro nível de programas permite a composição e o acesso conveniente das funções de banco de dados implementadas pelo segundo nível. No nível mais alto é implementada uma linguagem de questões ("queries") que fornece uma interface "amigável" entre o usuário final e o sistema de banco de dados.

CAPÍTULO 2

MANIPULAÇÃO DE ÁRVORES QUATERNÁRIAS

Nesse capítulo será discutido o procedimento para a detecção de adjacência entre pixels e blocos, que constitui uma tarefa básica na implementação de muitas outras operações sobre imagens; a rotação por ângulos retos, afim de ilustrar um caso simples de transformação; e as operações de união e interseção de imagens, fundamentais para a obtenção de novas imagens a partir do confronto da informação de duas ou mais outras.

2.1 - ADJACÊNCIA ENTRE OS NÓS

Dois nós, P e Q, de uma árvore quaternária são considerados **adjacentes** quando existirem ao menos dois pixels, p e q, em seus blocos associados, vizinhos em alguma das **direções principais** N, S, L e O, na imagem vista como uma matriz. A relação de adjacência nessas 4 direções será denominada **4-adjacência.** Se forem tomadas também as direções NO, NE, SO e SE, tem-se a relação de **8-adjacência.**

Assim, um nó P é dito adjacente a um nó Q, na direção d, se Q for adjacente ao lado ou esquina d de P, d = N, S, L, O, NO, NE, SO e SE. Por exemplo, na imagem da Figura 1.2 o nó "37" é vizinho na direção SO do nó "24". Dois nós podem ser adjacentes por um lado e por uma esquina simultaneamente, como é o caso do nó "0", adjacente ao "32" nas direções N e NE . A relação de adjacência aplica-se também aos nós não terminais (**cinza**), como o nó "E", vizinho do nó "32" na direção E.

A relação descrita não é uma função no sentido matemático, podendo haver mais de um nó adjacente a um nó P, numa direção d, como os nós "37", "39", "E" e "C"

na Figura 1.4 que são todos vizinhos do nó "48" na direção O. Pode-se destacar a adjacência por um lado inteiro (p.ex. "0" e "32"), a adjacência por um segmento de lado (p.ex. "36" e "32") e outros conceitos, conforme a aplicação objetivada.

No contexto que se segue, um nó será considerado maior, igual ou menor que um outro qualquer, conforme o total de pixels de seus blocos correspondentes na imagem. É possível definir funções úteis para determinar um nó adjacente a um certo nó P, em situações mais restritas. Neste trabalho é utilizada a função que determina o menor nó, Q, maior ou igual a P, cujo bloco associado seja adjacente por lado inteiro ao bloco associado a P. Essa função não é uma correspondência biunívoca. Um nó pode ser adjacente numa direção dada a vários outros, como é o caso, na imagem da Figura 1.4, do nó "0", que é o menor adjacente ao lado N de "32", "36" e "37", maior ou igual a cada um deles. Também não é simétrica; o nó "0" é o menor adjacente ao lado O de "24", mas o menor adjacente ao lado E de "0" não é o "24", e sim o nó **cinza**, "B".

Métodos para a determinação de nós adjacentes em árvores quaternárias são comuns em muitas aplicações típicas sobre imagens gráficas, tais como o cálculo de propriedades geométricas como o número de Euler (Dyer, 1980) e o perímetro (Samet, 1981c); nas conversões entre representações (Samet, 1980a, 1980b, 1981a); em operações de percurso de bordas (Samet, 1980b); na determinação de componentes conexas (Samet, 1981b); e outras.

A implementação desenvolvida neste trabalho localiza o nó Q, **menor adjacente maior ou igual** a um certo nó P, isto é, de nível menor ou igual de subdivisão da imagem, por um processo recursivo, que percorre uma

seqüência ascendente de nós ancestrais de P, na busca do ancestral comum ao nó P e seu nó adjacente, Q, numa dada direção principal. A seguir, é percorrida a **seqüência descendente** dos descendentes do ancestral encontrado, que são também ancestrais de Q.

A função descrita tem como parâmetros uma direção d (= N, S, L, ou O); um ponteiro P; e o nível do nó, e deve retornar o nó adjacente na direção desejada. O nó pai de um nó dado pelo ponteiro P será designado por pai(P); o filho de um certo nó P, cujo tipo de quadrante é qd (= NO, NE, SO, ou SE) será denominado filho (qd, P).

Inicialmente, é determinado o **tipo de quadrante** associado ao nó adjacente procurado, Q, que será referido como o **reflexo** do tipo de quadrante associado a P, na direção d, passada como parâmetro. Os resultados possíveis são dados na Tabela 2.1.

TABELA 2.1

REFLEXO DE QUADRANTES NAS DIREÇÕES N, S, L e O.

Quadrante	Dir.	Reflexo
NO	N/S	SO
	E/O	NE
NE	N/S	SE
	E/O	NO
SO	N/S	NO
	E/O	SE
SE	N/S	NE
	E/O	SO

Caso o tipo de quadrante refletido, qd, indique que o adjacente é um irmão de P, então o nó filho (qd,pai(P)) será retornado pela função. O ancestral comum é portanto o nó pai(P). Caso contrário, a função é chamada recursivamente para localizar o nó adjacente ao pai(P), o nó pai(Q). Em alguma chamada recursiva será encontrado o **ancestral comum** de P e Q procurado, quando então tem inicio o retorno às chamadas recursivas que levaram até esse nó na seqüencia ascendente, o que corresponde ao percurso da seqüência descendente até o nó procurado.

No processo ascendente, cada nova chamada recebe como parâmetros o nó pai do atual e o nível a que pertence. Após encontrar o ancestral comum, será retornado o ponteiro inicial para os descendentes cujo tipo de quadrante associado foi obtido anteriormente como sendo o reflexo do tipo de quadrante do nó percorrido no processo ascendente, no mesmo nível em processamento.

Por exemplo, na localização do nó adjacente na direção leste, de um nó P, o ancestral comum fica determinado como o primeiro nó acessado a partir de seu filho do tipo NO ou SO, pois o nó adjacente é do tipo NE ou SE e, portanto, irmão no mesmo nível do nó NO ou SO encontrado na ascendencia. O adjacente NE ou SE é então o primeiro a ser percorrido na seqüência descendente que, efetivamente, localiza o nó vizinho procurado ao atingir o nível do nó P, como é ilustrado pela Figura 2.1. No caso da adjacência na direção S, o ancestral comum será o primeiro nó acessado a partir de seu filho do tipo NO ou NE. No caso N, o ancestral comum deve ser acessado a partir de seu filho do tipo SO ou SE. Na direção O o ancestral comum deve ser acessado a partir de seu filho do tipo NE ou SE.

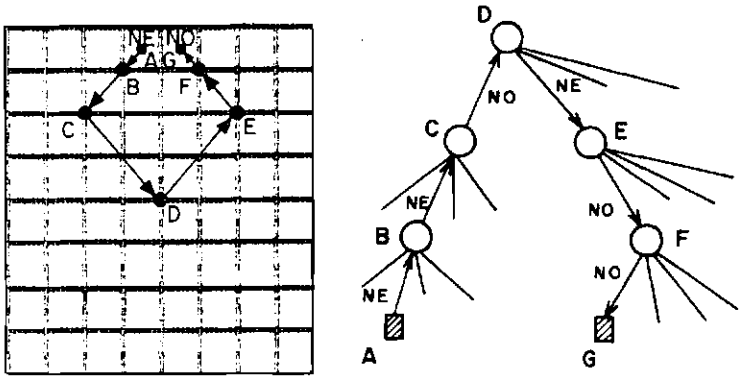

Fig. 2.1 - Adjacente leste de um pixel NE.

Assim, por exemplo, se na pesquisa do adjacente sul de um nó P do tipo SO a seqüência ascendente até o ancestral comum for (SO, SO, SE, SO, SO, NE), então a seqüência descendente determinada é (SE, NO, NO, NE, NO, NO). Nesse caso foi preciso ascender seis níveis na árvore a partir de P, sendo, portanto, necessário descender seis níveis a partir do ancestral comum, tomando as direções refletidas obtidas nas chamadas de mesmo nível da seqüencia ascendente, a fim de determinar o nó Q adjacente a P na direção S.

Os procedimentos para determinação da adjacência em direções diagonais são um pouco mais complexos, podendo, entretanto, ser descritos a partir de aplicações múltiplas dos procedimentos para direções verticais. Samet (1982) apresenta alguns procedimentos para a determinação de adjacências entre nós vizinhos nas direções horizontal e vertical, e desenvolve um modelo (Samet et al., 1985) para analisar a eficiência desses procedimentos em termos do número de nós visitados.

2.2 - ROTAÇÃO POR ÂNGULOS RETOS

A **rotação** de uma imagem, representada por uma estrutura de árvore quaternária, por ângulos múltiplos de 90^{o} é um caso particular, de fácil implementação, de transformação sobre imagens. O procedimento baseia-se no percurso em pré-ordem da árvore de uma certa imagem e simultânea geração da árvore quaternária da imagem resultante do processo de rotação. Cada nó visitado dá origem à geração de um nó na árvore resultante em função do tipo de quadrante dos nós examinados e do sentido de rotação estabelecido, que pode ser horário ou anti-horário. A Figura 2.2 ilustra este processo para o caso anti-horário.

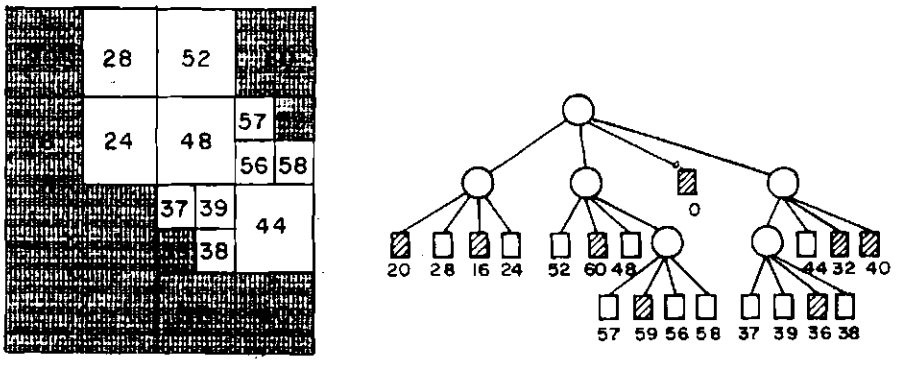

Fig. 2.2 - Rotação anti-horária de 90^{o} da Figura 1.2.

Para a rotação anti-horária por um ângulo reto, de uma imagem dada por sua árvore quaternária os tipos de quadrantes associados aos nós gerados na árvore resultante são mostrados, em função do tipo original, pelo mapeamento descrito na Tabela 2.2.

TABELA 2.2

ROTAÇÃO POR ANGULOS RETOS (ANTI-HORÁRIA)

Tipo de Quadrante	
Original	Resultado
NO	SO
NE	NO
SO	SE
SE	NE

Os nós **cinza** dão origem a nós **cinza** na árvore resultante, com seus nós filhos permutados segundo o mapeamento da Tabela 2.2, em função do sentido e do ângulo (multiplo de 90^{o}) de rotação da imagem. O caso da rotação no sentido horário parte de um mapeamento análogo.

2.3 - OPERAÇÕES DE CONJUNTO

Operações de conjunto são fundamentais na implementação de outras operações sobre imagens como superposição, janelamento, tranformações (Hunter e Steiglitz, 1979b; Bauer, 1985). A obtenção da árvore correspondente à imagem determinada pela união ou interseção de duas outras, dadas por suas árvores quaternárias, consiste no percurso em pré-ordem, simultâneo das duas árvores. Em cada passo, dois nós correspondentes são comparados, dando origem a um novo nó na árvore resultante. A implementação desenvolvida nesse trabalho é baseada no esquema proposto por Hunter e Steiglitz (1979a) para implementação da operação da superposição de duas ou mais imagens.

A operação de **união** é implementada por uma função que recebe como parâmetros os ponteiros para os nós raiz de duas árvores quaternárias, as quais representam duas imagens $2^n x 2^n$, e retorna o ponteiro para a raiz da árvore da imagem união entre as duas imagens dadas. Ao comparar dois nós, quatro tipos de atitudes podem ser tomadas conforme o valor de cada um.

Quando ambos os nós comparados forem **cinza**, a função é chamada recursivamente para determinar a união entre os nós filhos de cada subárvore, associada, respectivamente, aos quadrantes NO, NE, SO e SE. Após o retorno da chamada correspondente ao filho SE, pode ser gerado um nó **preto** ou **cinza** na árvore resultante. Caso contrário, quando pelo menos um nó terminal for comparado, é gerado um nó na árvore união, que pode ser **preto**, **branco** ou **cinza**, conforme os valores dos nós comparados considerando-se que a região de interêsse da imagem, a **figura**, é constituída pelos pixels pretos, enquanto os pixels brancos correspondem ao **fundo**. As seguintes atitudes são previstas:

- Se um dos nós for **preto**, é gerado um nó **preto**.
- Se ambos forem **brancos**, é gerado um nó **branco**.
- Se um for **cinza** e o outro **branco**, o nó **cinza** é copiado.

A **interseção** entre dois nós é implementada de maneira análoga, com os nós **pretos** e **brancos** desempenhando papéis invertidos.

Quando ambos forem **cinza**, chamadas recursivas determinam a interseção dos filhos correspondentes em cada subárvore, podendo dar origem à geração de um nó **branco** ou **cinza**. Caso contrário, um nó **branco**, **preto** ou **cinza**,

conforme os valores dos nós comparados, nas seguintes situações:

- Se um dos nós for **preto**, é gerado um nó **preto**.
- Se ambos forem **brancos**, é gerado um nó **branco**.
- Se um for **cinza** e o outro **branco**, o nó **cinza** é copiado.

A Figura 2.3 mostra a decomposição em blocos e as árvores resultantes da aplicação dos procedimentos para determinação da intercessão entre duas imagens 2^3x2^3.

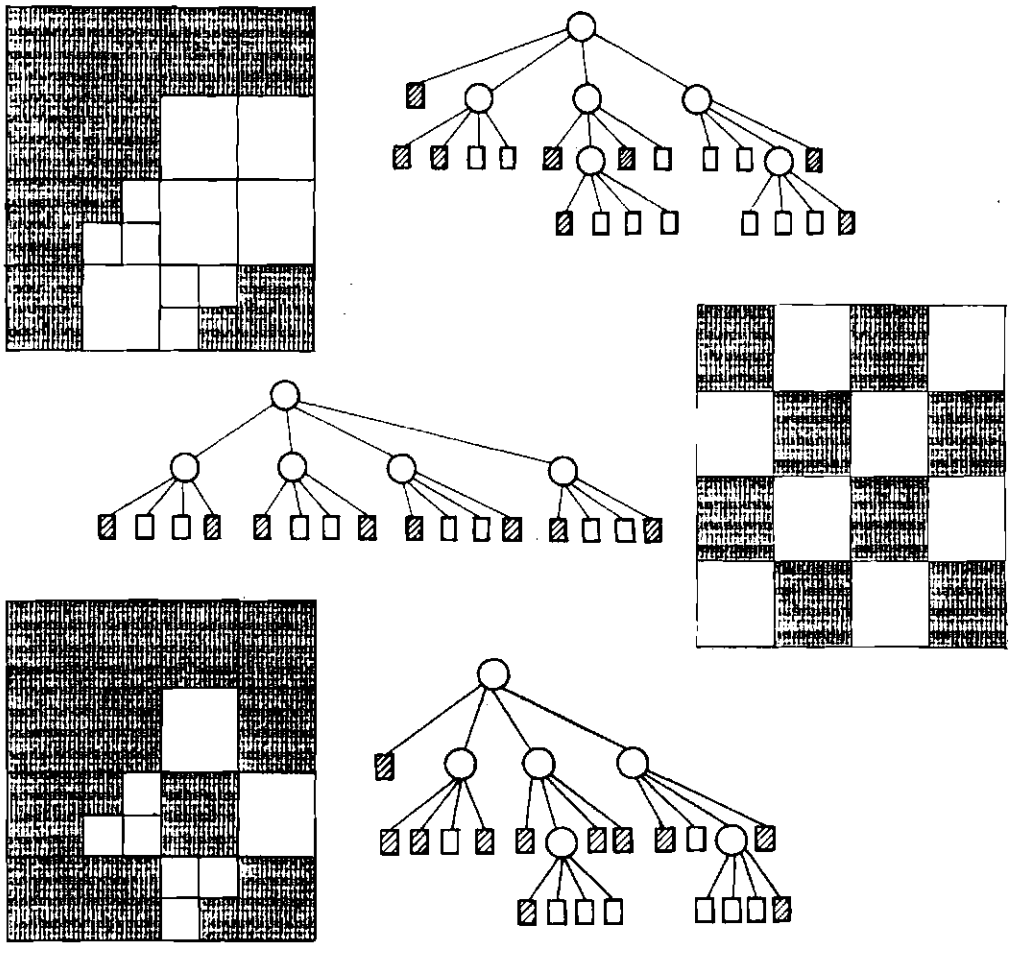

Fig. 2.3 - Interseção de duas imagens 2^3x2^3.

O resultado da união ou interseção de blocos associados a dois nós **cinza** pode ser um bloco homogêneo. Essa homogeneiade não é percebida na geração do nó associado a esse bloco resultante, de modo que, quando dois nós **cinza** são comparados, o nó resultante deve ser condensado, isto é, as situações de agrupamento entre nós homogêneos devem ser verificadas antes da efetiva geração do nó resultante.

Operações de conjunto são usadas como base para implementação de tarefas envolvidas na montagem de **"questões"** sobre imagens em bancos de dados cartográficos. Num Sistema Geográfico de Informação, pode-se desejar determinar, por exemplo, áreas adequadas para o plantio de uma certa cultura, que pode ser descrita através de parâmetros de classe, como "tipo de solo", "índice pluviométrico" e "faixa de altitude". Supondo-se que uma tal região seja caracterizada pelo tipo de solo TS_1 ou TS_2, índice pluviométrico na faixa Ip_1, e faixa A_1 de altitude, então a descrição dessa área pode ser expressa em termos de operações de conjunto como:

inter (inter (união (TS_1,TS_2), Ip_1), A_1)

Bauer (1985), utiliza árvores quaternárias lineares na implementação das operações de união, interseção e diferença entre imagens, concluindo que a eficiência temporal de seus procedimentos depende linearmente do número de nós.

CAPÍTULO 3

REPRESENTAÇÃO DE DADOS REGIONAIS

A efetiva geração em memória principal de árvores quaternárias a partir das representações matricial e raster de imagens que representam regiões é aqui discutida, bem como a conversão no sentido inverso, que permite obter a representação raster de uma imagem a partir de sua árvore quaternária.

3.1 - ÁRVORE QUATERNÁRIA A PARTIR DE MATRIZ

A construção da árvore quaternária a partir de uma imagem binária, dada sob a forma de matriz $2^n x 2^n$ é implementada neste trabalho com base no procedimento proposto por Samet (1980a), que consiste na geração em pós-ordem dos nós associados a pixels e blocos homogêneos. Cada pixel da imagem inicial é examinado uma única vêz, a partir do pixel **extremo-NO**, o primeiro pixel da imagem, que corresponde às coordenadas (0,0). O procedimento é implemetado em uma única fase que resulta em uma estrutura de árvore mínima, no sentido de que apenas blocos homogeneos maximais são representados por nós-folhas. A Figura 3.1 ilustra a seqüência de percurso dos pixels de uma matriz de imagem para a geração de sua árvore quaternária.

O processo consiste em chamadas recursivas a uma função que recebe como parâmetros um bloco de imagem dado pelo par de coordenadas (x,y) de seu primeiro pixel e pelo nível k da árvore que está sendo processado. A partir do primeiro pixel do bloco, cada um é examinado numa ordenação análoga à descrita na Seção 1.3.

A função é inicializada com as coordenadas (0,0) do primeiro pixel da imagem e o nível máximo, n, da árvore que será gerada, retornando um ponteiro para seu nó raiz.

O nível k, passado como parâmetro, indica que está sendo processado um bloco $2^k x 2^k$. Se k for positivo, deve ser decrementado e passado como parâmetro para o processamento recursivo dos quatro quadrantes $2^{k-1} x 2^{k-1}$, na seqüência NO, NE, SW, SE de suas posições no bloco. Quando o nível for 0, as coordenadas passadas como parâmetro indicam o primeiro e único pixel de um bloco do nível n de subdivisão. Neste caso, a cor associada a este pixel será retornada.

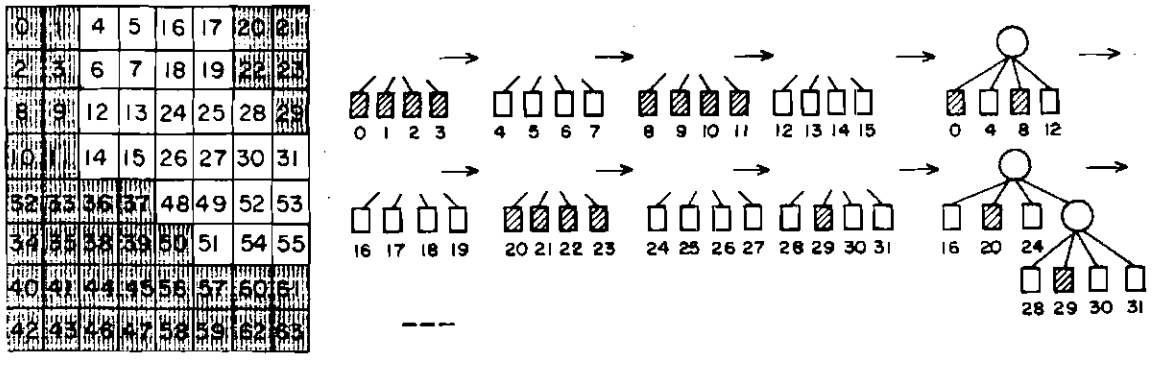

Fig. 3.1 - Árvore quaternária de uma matriz de imagem.

Quando quatro pixels, partes de um mesmo bloco de nível n-1 de subdivisão, são homogêneos, nenhum nó é gerado na árvore, apenas a informação da cor do bloco é retornada pela função. Caso contrário, um nó **cinza** de nível 1 será gerado na árvore, junto a seus quatro nós filhos, **brancos** e/ou **pretos**, e um ponteiro para o nó **cinza** é retornado pela função. Após o processamento dos quatro pixels de um bloco associado a um quadrante do tipo SE, no

nível n-1 de subdivisão, a função terá completado o processamento de um bloco 2^2x2^2 da imagem, e novamente deve decidir quanto à geração de um nó **cinza** de nível 2 ou o retorno da cor do bloco correspondente de nível n-2 de subdivisão. Assim, sucessivamente vão sendo determinados os níveis superiores da árvore.

Desse modo, após o processamento do quadrante SE de um bloco 2^kx2^k, pode ou não ser gerado um nó **cinza** de nível k, dependendo dos resultados obtidos em cada um de seus quadrantes $2^{k-1}x2^{k-1}$. Quando um nó **cinza** é gerado, os quadrantes homogêneos darão origem a nós terminais, **branco** ou **preto**, conforme a cor retornada pela função ao processá-lo, e quadrantes não homogêneos já estarão representados por nós **cinza**.

As coordenadas do primeiro pixel de um quadrante são obtidas das do primeiro pixel do bloco que o contém e do nível de árvore passado como parâmetro. Por exemplo, se as coordenadas do primeiro pixel forem (x,y) e o nível k, então as do seu quadrante NO serão também (x,y); as do NE serão $(x+2^k,y)$; as do quadrante SO, $(x,y+2^k)$; e as do SE $(x+2^k,y+2^k)$.

Assim, o procedimento prevê a construção imédiata de nós **cinza**, enquanto os nós **pretos** e **brancos** são gerados apenas quando os blocos que os contêm já não puderem participar de algum agrupamento de blocos homogêneos nos níveis superiores de árvore, ou seja, apenas quando um nó **cinza** for gerado, como é mostrado na Figura 3.1.

3.2 - ÁRVORE QUATERNÁRIA A PARTIR DE RASTERS

A implementação desenvolvida, é baseada na proposta de Samet (1981a). Cada linha horizontal de uma

imagem binária $2^n x 2^n$ é processada. Uma linha é uma cadeia de bits de comprimento 2^n. Cada pixel é examinado uma única vez na seqüencia ilustrada na Figura 3.2, dando origem a um novo nó na árvore resultante, de modo que ao final do procedimento cada pixel estará representado por um nó de nível 0. As situações de agrupamento não são consideradas durante o processo de obtenção da árvore, e sim, em uma segunda fase do procedimento, dando origem a uma árvore quaternária típica e liberando o exesso de área de memória alocada para a geração temporária de 4^n nós terminais e 4^n-1 intermediários.

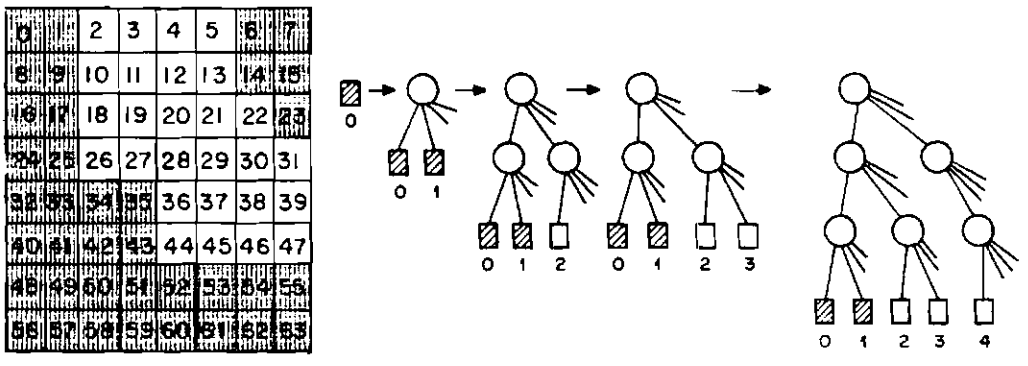

Fig. 3.2 - Árvore quaternária de uma imagem raster.

Inicialmente é gerado um nó terminal, P, que informa o valor de cor associado ao primeiro pixel da primeira linha da imagem. O exame do segundo pixel vai determinar a geração do nó adjacente ao nó P na direção Leste por um procedimento análogo ao descrito na Seção 2.1 para a determinação do nó adjacente a um certo nó em uma direção dada, com apenas uma diferença; neste caso, muitos nós da seqüência ascendente e descendente podem ainda não ter sido inseridos na árvore. O problema é contornado pela geração desses nós sempre que necessário, com o valor **cinza** indicado.

Cada pixel de uma linha, com exeção do primeiro, dá origem à geração de um nó adjacente na direção leste do nó recém-criado, de modo que o processo ascendente só é revertido quando um nó **cinza** correspondente ao pai de um nó associado a um quadrante do tipo NO ou SO for acessado ou gerado. No processo descendente, os ponteiros são tomados segundo o quadrante refletido, com relação ao lado leste, daquele que foi percorrido no processo ascendente e no mesmo nível. Quando o nível 0 for atingido, um nó Q é gerado e a cor associada ao pixel correspondente é registrada em seu campo de valor. O primeiro pixel de cada linha, a partir da segunda, é obtido pelo processo de geração do nó adjacente na direção sul ao nó que representa o primeiro pixel da linha anteriormente processada. A Figura 3.2 ilustra o processamento dos quatro primeiros pixels da primeira linha da imagem.

Um modo de prever situações de agrupamento de pixels, ou nós de níveis mais altos, quando forem homogêneos quanto à cor, é obtido considerando o fato de que agrupamentos podem ocorrer apenas em linhas ímpares, onde os tipos de quadrantes associados aos pixels examinados são sempre SO ou SE, após o exame de um pixel SE.

A ocorrência de agrupamento no nível 1 da árvore pode acarretar outros nos níveis superiores. Numa imagem $2^n x 2^n$, toma-se um pixel de coordenadas $(a.2^i-1, b.2^j-1)$ com a, b, i, j, inteiros. Um par desse tipo corresponde sempre ao último pixel de um ou mais quadrantes do tipo SE na imagem, de modo que os agrupamentos resultantes são determinados por blocos homogêneos que contenham esse par (agrupamentos em níveis superiores). O número máximo de agrupamentos é dado pelo mínimo entre i e j. Por exemplo, o pixel 27 na imagem da Figura 3.2 corresponde às coordenadas $(2^2-1, 2^2-1)$, portanto o processamento desse

pixel pode provocar até 2 agrupamentos nos níveis superiores da árvore.

3.3 - RASTERS A PARTIR DE ÁRVORE QUATERNÁRIA

Samet (1984) descreve e compara quatro procedimentos semelhantes para a obtenção da representação raster a partir de uma estrutura de árvore quaternária. Esses procedimentos são apresentados de um modo evolutivo, no sentido de que o segundo resulta de modificações feitas no primeiro e assim sucessivamente.

Aqui, é mostrado o percurso da árvore de uma imagem numa ordenação ditada pelas linhas que vão sendo produzidas. Para cada linha, cada nó terminal associado a um bloco que a intercepte é visitado da esquerda para a direita. Por exemplo, na Figura 3.3, para a obtenção da primeira linha da representação raster, é visitado o nó A, seguido dos nós B, C, D, E. Cada nó **preto** ou **branco** de nível k na árvore será visitado 2^k vezes, para a geração de 2^k segmentos com 2^k pixels de comprimento. A imagem resultante desse processo é dada sob a forma de uma seqüência ordenada de linhas que a compõem.

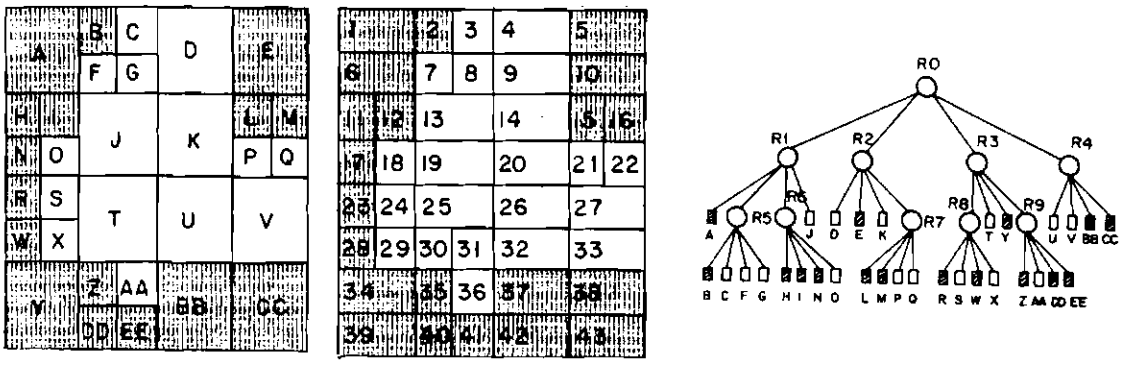

Fig. 3.3 - Imagem raster de uma árvore quaternária.

O que distingue esses procedimentos é, bàsicamente, o número de visitas aos nós-folhas que são iniciadas na raiz. A geração de cada linha consiste na geração dos segmentos que a compõem, contidos nos blocos que a interceptam da esquerda para a direita.

O primeiro procedimento gera cada segmento de cada linha a partir de um percurso independente, iniciado na raiz da árvore, para atingir o nó que o determina.

O segundo procedimento visita, a partir da raiz, apenas os nós que determinam o segmento mais à esquerda de cada linha gerada, os nós restantes são obtidos utilizando a estrutura de árvore para encontrar o nó adjacente, na direção leste, que irá determinar o próximo segmento a ser produzido. Sempre que o vizinho leste não for encontrado, o processo retorna ao nó raiz, para iniciar a geração de uma nova linha.

O terceiro procedimento inicia o percurso da árvore para a geração de todos, com exeção do primeiro segmento de um bloco mais à esquerda, pelo seu nó terminal associado. Apenas na geração desse primeiro segmento o percurso da árvore deverá ser iniciado na raiz. Os segmentos seguintes em uma linha horizontal são obtidos pela determinação do nó vizinho na direção principal leste, que determina o próximo segmento a ser gerado. A não-existência de um tal nó, remete o processamento para o nó inicial dessa mesma linha; se o bloco correspondente intercepta a próxima linha a ser gerada, o nó será novamente utilizado como ponto de partida do percurso da árvore, senão o procedimento será iniciado pela raiz.

No quarto procedimento, apenas a determinação da primeira linha da representação é iniciada na raiz da árvore. Ao término da geração de uma linha o processo

retorna ao nó inicial desta. Caso este não intercepte a próxima linha a ser produzida, é utilizado o procedimento para determinação do vizinho na direção principal sul deste nó, afim de determinar o primeiro segmento da nova linha.

As seqüências formadas pelas seqüências de nós percorridos no processo de geração das três primeiras linhas da imagem da Figura 3.5, é mostrada a seguir, conforme cada um dos procedimentos discutidos, aqui identificados como "Primeiro", "Segundo", "Terceiro" e "Quarto". As seqüências a), b) e c) referem-se à geração da primeira, segunda e terceira linhas, respectivamente.

Primeiro:

a) R1,A; R1,R5,B; R1,R5,C; R2,D; R2,E;

b) R1,A; R1,R5,F; R1,R5,G; R2,D; R2,E;

c) R1,R6,H; R1,R6,I; R1,J; R2,K; R2,R7,L; R2,R7,M.

Segundo:

a) R1,A; R1,R5,B; R5,C; R5,R1,R0,R2,D; R2,E; R2,R0;

b) R1,A; R1,R5,F; R5,G; R5,R1,R0,R2,D; R2,E; R2,R0;

c) R1,R6,H; R6,I; R6,R1,J; R1,R0,R2,K; R2,R7,L; R7,M; R7,R2,R0.

Terceiro:

a) R1,A; R1,R5,B; R5,C; R5,R1,R0,R2,D; R2,E; R2,R0;

b) A; R1,R5,F; R5,G; R5,R1,R0,R2,D; R2,E; R2,R0;

c) R1,R6,H; R6,I; R6,R1,J; R1,R0,R2,K; R2,R7,L; R7,M; R7,R2,R0.

Quarto:

a) R1,A; R1,R5,B; R5,C; R5,R1,R0,R2,D; R2,E; R2,R0;

b) A; R1,R5,F; R5,G; R5,R1,R0,R2,G; R2,E; R2,R0;

c) H; R6,I; R6,R1,J; R1,R0,R2,K; R2,R7,L; R7,M; R7,R2,R0.

No último procedimento, a determinação do nó H é feita a partir da determinação do nó vizinho na direção principal sul do nó A, o que exige o percurso da seqüência de nós A,R1,R6,H.

A determinação do nó vizinho leste (ou sul) faz uso, numa primeira fase, da função estudada no Capítulo 2, que determina o menor nó adjacente a um certo nó dado, numa direção especificada, que seja maior ou igual a esse nó. Se for encontrado um nó **branco** ou **preto** terminal, o nó e o tamanho do bloco associado são retornados, determinando assim o próximo segmento a ser gerado como saída. Caso contrário, numa segunda fase, é ativado um procedimento que encontra o nó terminal descendente do nó **cinza** encontrado, cujo bloco associado contém o próximo segmento a ser gerado. Samet (1984a) implementa essa fase por um procedimento recursivo, que tem como parâmetros um ponteiro para o nó **cinza** encontrado, o tamanho do bloco e as coordenadas dos pixels inicial e final, retornando um ponteiro para o nó terminal que determina o segmento procurando.

A implementação proposta neste trabalho, para a segunda fase do procedimento descrito no parágrafo anterior, que encontra o nó terminal que irá determinar o segmento da linha a ser gerada, faz uso do conceito de **código quaternário**, discutido na Seção 1.3.2. As coordenadas (x,y) do primeiro pixel do próximo segmento a

ser gerado e o nível do nó **cinza** adjacente encontrado na primeira fase são suficientes para localizar o nó terminal procurado. O procedimento para intercalação de bits descrito na Seção 1.3.2 permite a obtenção do código quaternário do pixel (x,y), cujos dígitos determinam a seqüência de ponteiros a ser percorrida desde o nó **cinza** ao nó que determina o próximo segmento a ser produzido. O nível, k, indica a ordem do dígito que fornece o **código locacional** que determina o primeiro ramo (NO, NE, SO, ou SE) a ser percorrido em direção ao pixel (x,y). Enquanto forem encontrados nós **cinza**, novos ramos são tomados de acordo com os dígitos de ordem mais baixa até que um nó terminal seja localizado. Este, certamente irá conter o pixel (x,y) e, portanto, determinar o próximo segmento da linha que está sendo produzida.

CAPÍTULO 4

REPRESENTAÇÃO DE FEIÇÕES LINEARES

Dados de regiões têm representações simples e naturais, baseadas na subdivisão recursiva da imagem até que todos os blocos sejam homogêneos quanto à cor, ou nível de cinza. Esses esquemas simples de subdivisão não funcionam bem com dados vetoriais em geral, onde vértices, segmentos de reta e interseções de segmentos devem ser registrados. Nesse capítulo serão discutidas algumas técnicas para a representação desse tipo de dados, bem como a implementação de uma estrutura para a representação exata de segmentos baseada no princípio da decomposição regular.

4.1 - REPRESENTAÇÃO DE PONTOS

Para representar **dados pontuais** pode ser usada uma estrutura de árvore análoga à usada para **dados regionais**, formada pela subdivisão recursiva em quadrantes até que nenhum bloco contenha mais de um ponto. Quando são armazenados dados regionais, o campo de **valor** associado a um nó-folha corresponde à cor da região. Para pontos e linhas, outras informações devem ser registradas nesse campo. Na representação de pontos proposta por Samet et al. (1984), são armazenadas suas coordenadas (x,y) com relação às do primeiro pixel do bloco correspondente, de modo que, se for necessário o conhecimento das coordenadas globais desse ponto, basta que sejam somadas às do ponto inicial do bloco. Os nós que não representam pontos têm o valor **branco** registrado. Em conseqüência, não é possível registrar mais de um ponto por nó.

Para inserir um ponto numa tal estrutura, primeiramente é localizado o nó-folha associado ao maior

bloco que o contenha exclusivamente. Se este for **branco**, as coordenadas relativas do ponto são determinadas e registradas no campo de valor do nó encontrado. Caso contrário, o nó encontrado já contém a informação relativa a algum outro ponto e, se não for coincidente com o que se deseja inserir, o nó deve ser subdividido. Nesse processo de subdivisão serão gerados três nós **brancos** e um que deverá conter as coordenadas do ponto encontrado. A seguir, o processo para inserir o novo ponto prossegue recursivamente sobre esses nós gerados. Para a remoção da informação relativa a um ponto, o nó que o contém é localizado e, a seguir, a informação é removida e o nó encontrado é transformado em um nó **branco**. Deve ser verificada a possibilidade de agrupamento entre o novo nó **branco** e seus nós irmãos. A Figura 4.1 ilustra o processo de subdivisão em blocos para o armazenamento de uma coleção de pontos.

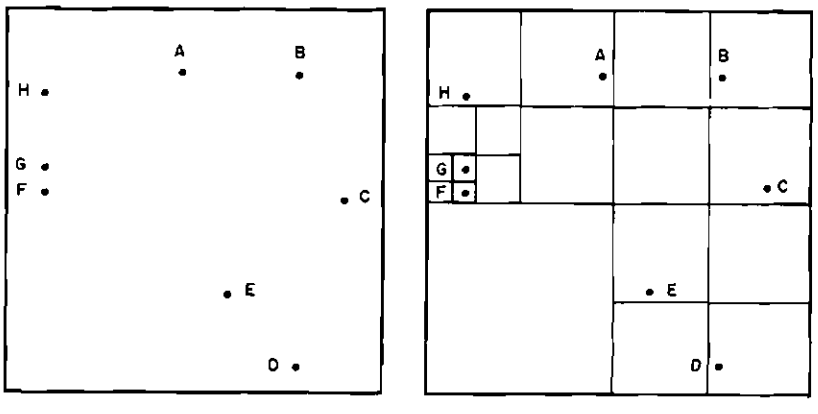

Fig. 4.1 - Árvore quaternária para representar pontos.

4.2 - REPRESENTAÇÃO DE LINHAS POLIGONAIS

Curvas, bordas, segmentos e linhas em geral, constituem os elementos lineares de uma imagem como o curso de um rio, o contorno de uma ilha e o traçado de uma estrada. A representação desses elementos por seqüência de segmentos conexos de reta, conhecida por representação

vetorial, é muito usada em computação gráfica e areas afins. Algumas técnicas têm sido desenvolvidas para representar linhas com base no princípio de decomposição regular.

Hunter e Steiglitz (1979a) apresentam uma proposta bastante simples, baseada na representação de regiões. No processo de subdivisão, uma imagem ou subimagem é particionada sempre que interceptar algum segmento de reta da representação vetorial de uma linha. A árvore resultante é essencialmente uma representação de duas classes distintas e complementares, a linha é representada por nós **pretos** no nível mais baixo de subdivisão e o resto da imagem define a classe representada pelos nós **brancos**, como ilustrado pela Figura 4.2.

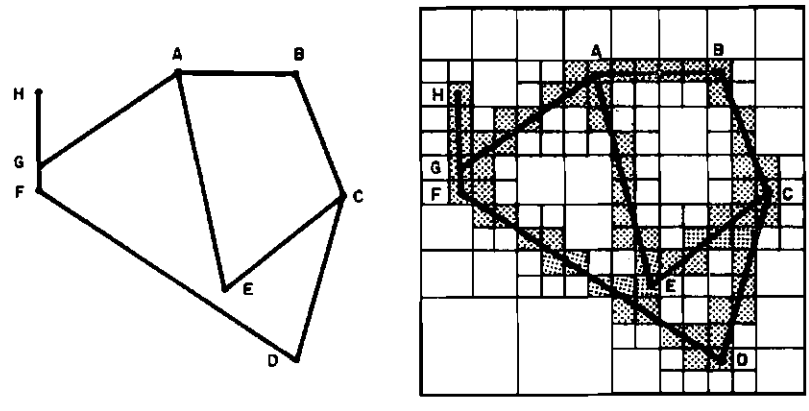

Fig. 4.2 - Segmentos como regiões de largura mínima.

Samet et al.(1984) propõem um modo de representar linhas que particionam um plano de imagem, ainda com base na árvore de dados regionais, conhecido por **árvore quaternária de linhas**. Essas linhas correspondem a bordas de regiões, de modo que poucas modificações são necessárias nas implementações baseadas em árvores usadas em representação de regiões. A estrutura é obtida por um processo análogo ao utilizado na Seção 3.1 para a geração da árvore de uma imagem matricial. Quando um pixel

pertencente a uma borda é percorrido e um nó de nível 0 é gerado, deve ser registrado, além da cor, o lado do bloco contido nessa borda. Em cada nível da árvore gerada, quatro nós terminais serão agrupados ou não, segundo critérios que envolvem a cor e a configuração de suas bordas. Lados parcialmente contidos na borda não são indicados.

A informação registrada é também hierárquica, no sentido de que os nós intermediários podem informar a presença de bordas adjacentes a seus lados, desde que essas não formem alguma junção em "T" com algum dos blocos descendentes ao longo dessa borda. Por exemplo, na Figura 4.3, o lado sul do bloco correspondente à raiz indica a presença de uma borda. Por outro lado, o lado oeste do filho SO da raiz não indica, embora exista, uma borda, devido à presença de uma junção em T entre seus filhos NO e SO.

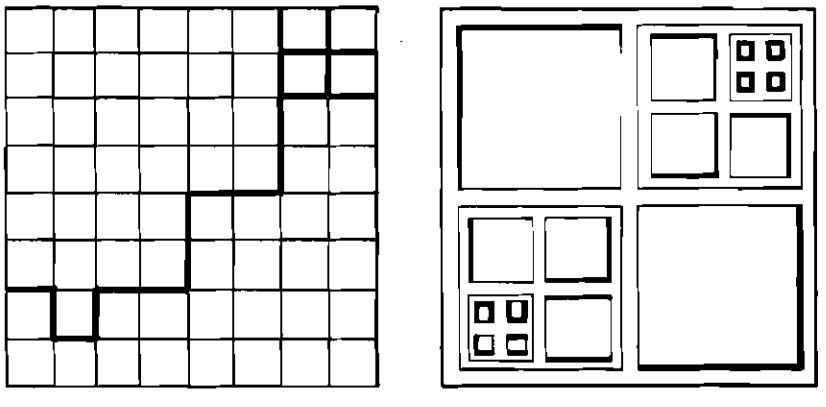

Fig. 4.3 - Árvore de linhas para um mapa 2^3x2^3.

O número de nós gerados na estrutura é igual ao que seria no caso da árvore regional, embora tarefas como percurso de bordas e superposição sejam bastante

facilitadas pela presença de informação em nós intermediários.

Alguns requisitos desejáveis para a representação de feições lineares são:

a) a possibilidade de representar mais de uma feição por nó;

b) a eficiência na representação de interseções;

c) o uso de uma quantidade fixa de memória por nó.

d) a representação exata de segmentos de reta.

e) inserções e remoções de segmentos consistentes.

A **árvore quaternária de bordas**, proposta por Shneier (1981a), baseia-se na decomposição regular de uma região com bordas e curvas em quadrantes até que cada quadrante intercepte um único segmento que possa ser aproximado, com uma certa precisão, por um segmento de reta.

A estrutura é similar à utilizada para representação de aproximações de imagens, onde, à média de níveis de cinza de um bloco, indicada em nós intermediários, é acrescentado um termo de erro associado à representação, por segmento de reta, de uma borda que intercepte o bloco. A informação em um nó inclui a **magnitude**, **direção**, **interseção** com um lado do bloco e o termo de erro direcional.

O campo de **magnitude** indica a presença ou não de alguma borda no bloco. Magnitude baixa indica regiões não-informativas, que podem em geral ser registradas em poucos nós correspondentes a grandes blocos de imagem. Apenas uma borda pode ser indicada por nó. O campo de

direção fornece o coeficiente angular do segmento representado, na precisão indicada pelo campo **termo de erro**. O campo de **interseção** indica o ponto em um dos lados do bloco que é interceptado por alguma borda. Um último campo é necessário para indicar se a borda representada tem um ponto inicial (ou final) no interior desse bloco. Em caso afirmativo, esse ponto será indicado no campo de interseção.

Desse modo são geradas árvores nas quais bordas longas e retilíneas são armazenadas em poucos nós; porém, vértices e interseções de bordas acarretam uma proliferação de nós de nível baixo gerados nas vizinhanças desses pontos, que são tratados como segmentos de borda de comprimento mínimo, como ilustra a Figura 4.4. Uma característica dessa estrutura é permitir a representação de mais de uma feição linear por árvore. Na verdade, o termo de erro, permite representar uma família de aproximações da imagem exata, conforme a precisão exigida, que é comparada com o termo de erro de cada nó afim de determinar se é ou não necessária a pesquisa de seus nós filhos, que deverão conter representações mais precisas da mesma borba.

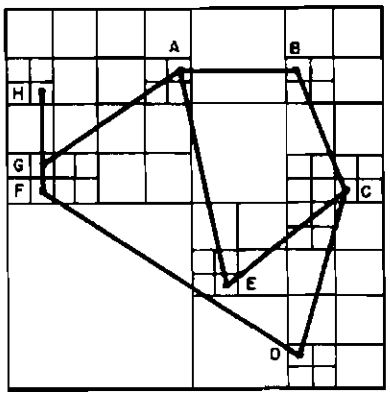

Fig. 4.4 - Árvore de bordas de linhas poligonais.

Operações de percurso de bordas podem ser implementadas com base em procedimentos muito parecidos com os utilizados, no caso regional, para determinação de vizinhanças nas direções principais e diagonais. Operações de conjunto são quase idênticas ao caso regional e muitas operações geométricas são também facilitadas.

Samet et al. (1984) utilizam uma variante da árvore de bordas, a **árvore linear de bordas** que faz uso de códigos quaternários (Gargantini, 1982) para a representação de blocos que contêm segmentos de bordas, aproximados por segmentos de retas. Aqui, o problema da codificação de vértices e interseções é contornado pela definição de um nível máximo de subdivisão que, quando atingido, pode dar origem à geração de um nó do tipo **ponto**, no qual é informado o número de segmentos que o atinge.

Na estrutura proposta nesse caso, os nós são mantidos por uma lista de códigos quaternários, ordenada de modo análogo ao percurso em largura (pré-order) de sua árvore equivalente. Três campos informam o **endereço**, o **tipo** e o **valor** associado a cada nó. O campo de **endereço** fornece as características de tamanho e posição do bloco correspondente, pela indicação do nível e do código quaternário de seu primeiro pixel. O campo de **tipo** indica se o nó é **branco**, se contém um **segmento**, ou um **ponto**. O campo de **valor** em um nó **segmento**, informa as coordenadas das interseções do segmento com dois lados do bloco correspondente, dadas pelo deslocamento relativo ao lado oeste do bloco, quando o lado norte ou sul do bloco foi interceptado, e relativo ao lado sul se o segmento intercepta o lado oeste ou leste do bloco. No caso de nó do tipo ponto, o número de linhas que intercepta o bloco (pixel) associado é indicado no campo de valor.

O problema da codificação de interseções e vértices, que serão representados por nós do tipo ponto é atenuado, embora a geração de muitos nós no nível mais baixo, na vizinhança de um tal nó, permaneça. O seguimento de bordas é facilitado pela indicação do número de linhas que atingem um ponto, embora permaneçam ainda as dificuldades oriundas do fato de cada nó descrever apenas a parte do segmento que intercepta seu bloco associado, sem nenhuma informação explicita sobre o início ou fim do segmento que o contém. A representação não é exata, inserções e remoções podem causar inconsistências, a menos que algumas restrições sejam impostas. A Figura 4.5 mostra a configuração de blocos, representada pela árvore linear de bordas, para o mapa da Figura 4.2.

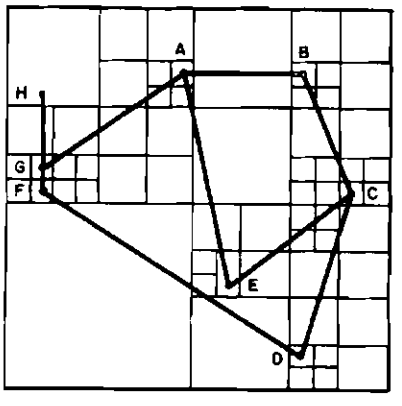

Fig. 4.5 - Árvore linear de bordas de linhas poligonais.

A fim de evitar erros de arredondamento causados pela alteração de pontos extremos de segmentos representados na estrutura, é assumido, nas operações de inserção e remoção, que segmentos são unidades atômicas indivisíveis. Não sendo, portanto, permitida a remoção ou inserção de partes de um segmento previamente inserido. Um segmento mais longo teria suas características alteradas após a remoção seguida de insersão.

Fixadas as restrições acima, as operações de inserção e remoção podem ser implementadas de um modo análogo ao utilizado para pontos e regiões. A **inserção** de um segmento em um bloco representado por um nó **segmento** causa a subdivisão do bloco em quadrantes e a geração de quatro nós associados aos quadrantes da região representada pelo nó original. A seguir, os dados do nó **segmento** encontrado são distribuídos entre os novos nós, podendo dar origem a até três nós **segmento** e os até três restantes serão **brancos**. A seguir o processo de inserção de um novo segmento prossegue recursivamente percorrendo os novos nós recém gerados. A **remoção** consiste em transformar em nós **brancos** cada nó **segmento** que contenha informação sobre o segmento a remover. Cada nó ponto têm a informação do número de segmentos que o intercepta atualizada, podendo transformar-se também em um nó **branco** se esse número tornar-se zero. Sempre que um nó **branco** é obtido nesse processo, a possibilidade de agrupamento com seu nós irmãos é verificada.

A fim de substituir a representação anterior no Sistema Geográfico de Informação onde havia sido aplicada, Samet et al. (1985) desenvolveram uma estrutura para representar linhas poligonais denominada **árvore de segmentos**, que satifaz os requisitos desejáveis descritos anteriormente e contorna algumas dificuldades apresentadas pela árvore de borbas na representação de coleções de feições lineares dadas por conjuntos de segmentos conexos de reta, em mapas poligonais (temáticos).

A estrutura utilizada distingue-se da árvore linear de bordas apenas quanto aos campos de tipo e valor de cada nó. O tipo de um nó pode ser **branco**, **vértice** ou **segmento**. Um nó **segmento** representa a parte de um único segmento de reta, que atravessa um certo bloco e cujos pontos extremos são exteriores a esse bloco. O valor, nesse

caso, indica as coordenadas do ponto inicial e final do segmento que a contém. O valor em um nó **vértice**, além das coordenadas do único vértice representado, indica os lados do bloco associado que são atravessadas por segmentos que o atingem.

A indicação das coordenadas dos pontos extremos do segmento inteiro em cada nó **segmento** e dos lados interceptados em cada nó **vértice**, permite uma representação exata e consistente do segmento. Raramente ocorrem nós **vértices** no nível mais baixo da estrutura. Dois nós na estrutura, que representem partes de um mesmo segmento, têm o mesmo conteúdo nos campos de valor. A Figura 4.6 mostra a configuração de blocos, dada pela árvore de segmentos, para o mapa da Figura 4.2.

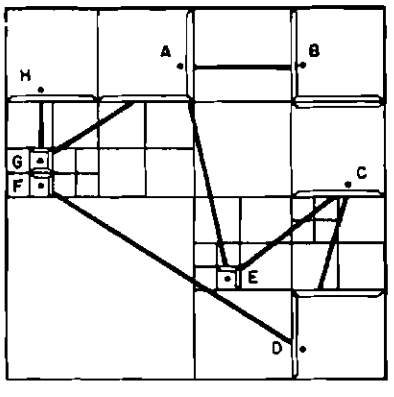

Fig. 4.6 - Árvore de segmentos de um mapa.

O procedimento para inserção de segmentos baseia-se no percurso em largura da árvore associada à imagem a fim de localizar o primeiro nó, cujo bloco associado intercepta o segmento a inserir. O tipo de atitude a tomar em seguida depende do tipo de nó encontrado e da posição relativa do segmento no bloco. Quanto à posição relativa, um segmento pode ter:

a) duas extremidades no bloco;

b) uma extremidade no bloco;

c) nenhuma extremidade no bloco.

Quando um nó **branco** é encontrado, a posição relativa do segmento a ser inserido determina no caso (a) que o nó encontrado deve ser substituído por quatro nós **brancos** associados aos quadrantes do bloco correspondente; no caso (b), que deve ser gerado um nó **vértice** na estrutura, com a indicação do lado que foi atravessada pelo segmento; No caso (c), que o nó **branco** deve ser transformado em um nó **segmento.**

Quando um nó **vértice** é encontrado, no caso (b), se o vértice representado coincide com a extremidade do segmento a ser inserido que está contida no bloco, o nó encontrado deve apenas ser atualizado quanto ao lado atravessado pelo segmento; no caso (c), se o vértice representado pertence ao segmento a ser inserido, o nó **vértice** encontrado deve apenas ser atualizado quanto aos lados atravessados pelo segmento. Nenhum novo nó é gerado em nenhum desses casos. Qualquer outra situação implica na geração de quatro nós **brancos** associados aos quadrantes do nó encontrado.

Quando um nó **segmento** é encontrado e o segmento a inserir é colinear com o segmento representado, se a posição relativa entre o segmento a inserir e o bloco atravessado pelo segmento representado se enquadra no caso (a), nenhuma alteração é feita sobre o nó; nos caso (b) e (c), pode ocorrer a alteração de uma ou duas das extremidades indicadas no campo de valor do nó encontrado. Quando os segmentos não são colineares, no caso (b), se apenas a extremidade contida no bloco pertence ao segmento representado ou, no caso (c), se os dois segmentos se

interceptam na região compreendida pelo bloco, o nó deve ser transformado em um nó **vértice**.

Qualquer outra combinação envolvendo a posição relativa entre o segmento a ser inserido e o conteúdo do nó encontrado, implica na geração de quatro nós **brancos**.

Quando no processo acima não ocorre a geração de novos nós na estrutura, caso não esteja concluida a inserção do segmento, o procedimento retorna recursivamente aos níveis superiores da árvore, a fim de localizar o próximo nó a ser confrontado com o segmento a inserir. Caso contrário, o procedimento reinsere o segmento ou vértice representado pelo nó terminal encontrado nos quatro nós **brancos** gerados e, só então segue recursivamente tentando inserir o novo segmento em cada nó resultante.

Para reinserir um nó **vértice**, cada segmento que atinge o vértice é reinserido independentemente nos quatro nós **brancos** gerados. Este processo dá origem a um nó **vértice** com o campo de valor idêntico ao do nó que foi substituído, o qual corresponde a um dos quadrantes do bloco original, e três ou mais outros, do tipo **branco** e **segmento**. A Figura 4.7 ilustra a inserção de um novo segmento na árvore de segmentos.

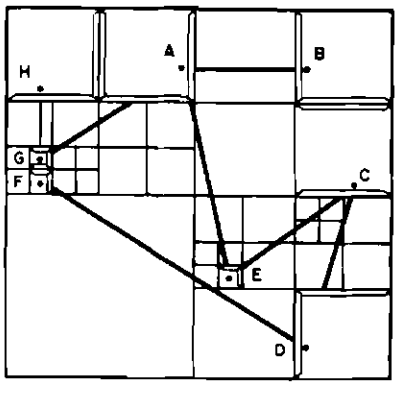

Fig. 4.7 - Inserção do segmento HA.

A remoção de um segmento consiste da repetição do mesmo processo de decomposição da imagem, utilizado na sua inserção, localizando e removendo toda informação que registra a presenca do segmento na estrutura. As atitudes a serem tomadas dependem do tipo de nó encontrado.

Se um nó **segmento** for encontrado, deve ser transformado em um do tipo **branco**. Se um nó **vértice** for encontrado, deve ser verificada sua pertinencia a algum outro segmento que não esteja sendo removido. Se os lados atravessados pelo segmento não forem atravessados por nenhum outro, então podem ser desmarcados. Se não restam lados atravessados no bloco associado ao nó, este deve ser transformado em um nó **branco**.

Eventualmente, a geração de um nó **branco** pode causar o agrupamento de quatro nós irmãos, o que resulta na geração de um nó **branco**, **segmento** ou **vértice**, em algum nível superior da estrutura. Agrupamentos ocorrem quando:

- Os quatro nós irmãos são **brancos**.
- 1, 2 ou 3 são nós **segmentos**, partes de um mesmo segmento enquanto, os restantes são **brancos**.
- 1 nó é **vértice**, e os restantes são **brancos** e/ou **segmentos** que interceptem o vértice representado.

No primeiro caso, deve ser gerado um nó **branco** associado ao bloco de nível imédiatamente inferior de subdivisão da imagem. No segundo caso, o nó resultante será do tipo **segmento**. No terceiro, um nó **vértice** deve ser gerado com a indicação dos lados atravessados. O nó gerado em cada caso substitui os anteriores, podendo também causar novos agrupamentos nos níveis anteriores da estrutura. A

Figura 4.8 mostra a configuração resultante da remoção de dois segmentos da estrutura da Figura 4.6.

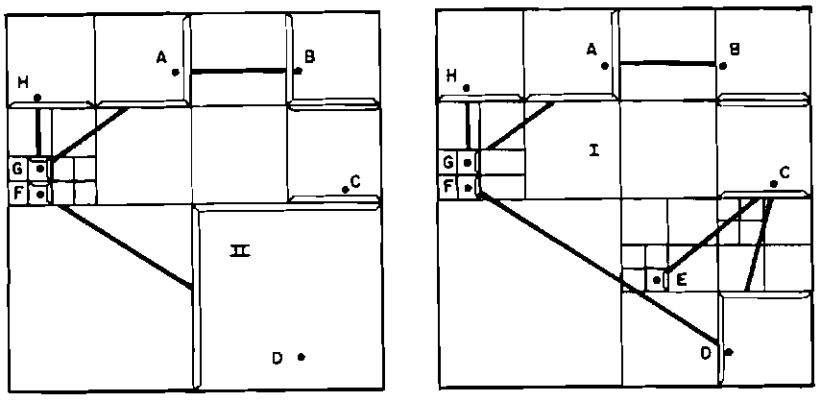

Fig. 4.8 - Remoção dos segmentos AE e CE.

4.3 - ÁRVORES QUATERNÁRIAS A PARTIR DE VETORES

Para demonstrar a viabilidade do modelamento de imagens dadas no formato vetorial por estruturas de árvore, foi também analisado e implementado, neste trabalho, um procedimento para inclusão de segmentos, baseado na árvore de segmentos delineada por Samet et al. (1985), que faz uso de estruturas convencionais de árvore. A sucessiva aplicação desse procedimento à lista de segmentos que descrevem linhas poligonais no formato vetorial define o procedimento de conversão entre as duas representações. A operação básica de remoção, embora necessária em aplicações onde deva ser permitida a atualização (correções e ajustes) de dados poligonais, não é discutida neste trabalho, o qual visa apenas demonstrar a conversão de imagens dadas no formato vetorial para a estrutura proposta.

O uso de estruturas convencionais de árvore acrescenta um "novo" tipo de nó na estrutura, o nó **cinza**. Sempre que possivel é utilizada a representação de pixels e

blocos por códigos quaternários, a fim de, por exemplo, determinar o caminho a ser percorrido na árvore, desde um certo nó **cinza** até algum de seus descendentes, usando operações primitivas sobre representações binárias.

Em seu trabalho, Samet et al. (1985) apresentam algumas atitudes tomadas em função da posição do segmento a ser inserido em relação ao bloco interceptado e ao segmento ou vértice representado pelo nó correspondente. Condições como a atomicidade de cada segmento simplificam a implementação, impedindo, por exemplo, que seja inserido (ou removido) um segmento contido em algum outro representado na estrutura. Essa restrição proíbe a representação, em uma mesma estrutura, de dois polígonos que compartilhem parte de sua borda. Na implementação proposta nesta seção, procurou-se permitir a inserção irrestrita de segmentos. Aqui, um nó **segmento** não representa apenas um segmento dado por suas extremidades, mas sim qualquer segmento que esteja contido entre essas extremidades.

Inicialmente é determinado o nó da estrutura associado ao menor bloco que contenha as extremidades do segmento a ser inserido. Para isto é determinado o código quaternário associado às coordenadas de suas extremidades, como descrito na Seção 1.3.2. A seguir os dígitos quaternários comuns aos dois códigos são utilizados para determinar o caminho a ser percorrido desde a raiz até o primeiro nó cujo bloco associado contém as extremidades do segmento a ser inserido. A operação de inserção é efetivamente iniciada neste nó. Quando um nó **cinza** é encontrado nesse processo, seus quatro filhos são percorridos recursivamente, a fim de localizar um nó terminal, **vértice**, **segmento**, ou **branco**, que intercepte o segmento a ser inserido.

Com relação ao bloco de imagem interceptado, é definido que um segmento **atravessa**, **atinge** ou pertence a este, conforme tenha nenhuma, duas, ou uma de suas extremidades nele contidas.

Com relação ao(s) segmento(s) representados pelo nó associado, é definido que um segmento **intercepta**, coincide ou estende algum segmento representado desde que, respectivamente: intercepte o vértice ou segmento representado; esteja contido em algum segmento representado; seja colinear ao segmento representado e contenha uma de suas extremidades.

As atitudes tomadas pelo procedimento para a inserção de um segmento na estrutura proposta são determinadas pela sua posição relativa ao bloco e a segmentos representados pelo nó correspondente. Assim, após a determinação do nó associado ao menor bloco que contém o segmento, três classes de atitude podem ser tomadas, dependendo do tipo de nó encontrado.

Quando é encontrado um nó **branco**, apenas a posição relativa ao bloco precisa ser considerada. Três atitudes podem ser tomadas, dependendo da posição do segmento em relação ao bloco associado:

a) se o segmento **atravessa** o bloco, o nó deve ser transformado em nó **segmento**;

b) se o segmento **atinge** o bloco, o nó deve ser transformado em um nó **vértice**;

c) se o segmento pertence ao bloco, o nó deve ser transformado em nó **cinza** com quatro nós **brancos** descendentes.

A Figura 4.9 ilustra as possíveis posições relativas entre um segmento e um bloco em uma operação de inserção.

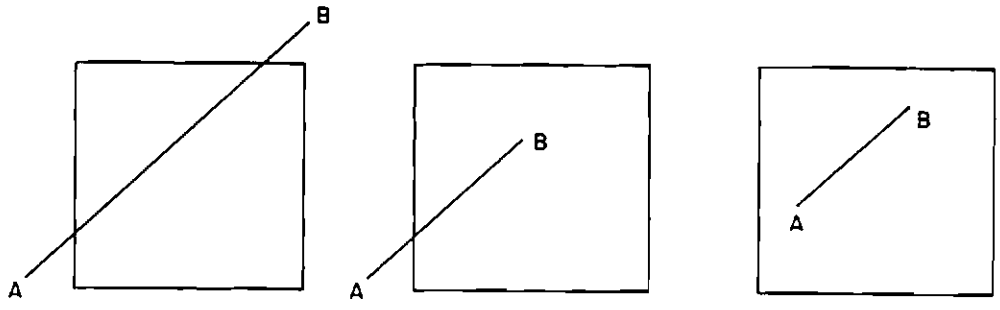

Fig. 4.9 - Posição relativa a um nó **branco**.

Quando um nó **vértice** é encontrado, as atitudes, dependentes da posição relativa ao bloco e ao(s) segmento(s) representado(s) pelo nó, a serem tomadas são:

a) Se o segmento **atravessa** o bloco e coincide com algum segmento representado, nenhuma atitude deve ser tomada.

b) Se o segmento **atravessa** o bloco e **intercepta** o vértice representado e, portanto a todos os segmentos que o atingem, o número de segmentos que atravessa dois lados deve ser atualizado.

c) Se o segmento **atravessa** o bloco e estende o único segmento nele representado, o nó deve ser transformado em um nó **segmento**.

d) Se o segmento **atinge** o bloco e **intercepta** o vértice na sua extremidade contida no bloco, o número de segmentos que cortam um lado deve ser atualizado.

e) Se o segmento **atinge** o bloco e coincide com algum segmento representado que contém sua extremidade no bloco, nenhuma atitude deve ser tomada.

f) Se o segmento **atinge** o bloco e estende o único segmento nele representado, duas atitudes podem ser tomadas:

- As coordenadas do vértice, dadas no campo de valor do nó encontrado, devem ser alteradas para informar as da extremidade do segmento a ser inserido contida no bloco.

- O nó deve ser transformado em um nó **segmento.**

Na Figura 4.10 são ilustradas as situações possíveis quanto a posição relativa, de um segmento, a um nó **vértice.** O bloco que contém o vértice é representado no plano inferior enquanto o segmento a ser inserido no superior; a superposição destes determina a situação resultante.

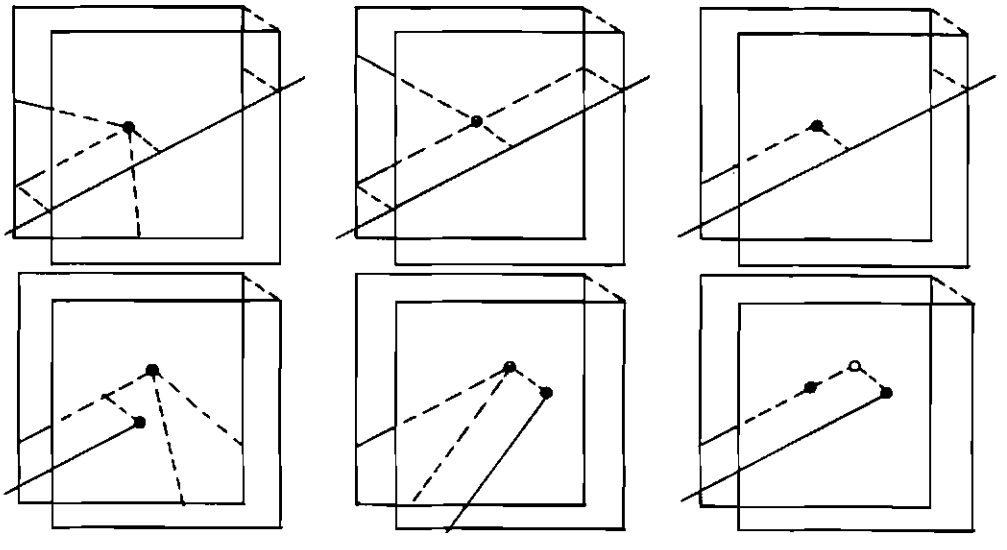

Fig. 4.10 - Posição relativa a um nó **vértice.**

Quando um nó **segmento** é encontrado, as atitudes, dependentes da posição com relação ao bloco e ao segmento nele representado, são:

a) Se o segmento a inserir **atravessa** o bloco e **intercepta** o que está nele representado, o nó associado deve ser transformado em um nó vértice, com as coordenadas da interseção dos segmentos e quatro cortes em seus lados indicados no campo de valor.

b) Se o segmento **atravessa** ou **atinge** o bloco, e coincide com o segmento representado, nenhuma atitude deve ser tomada.

c) Se o segmento **atravessa** ou **atinge** o bloco, e estende o segmento representado, as extremidades indicadas no campo de valor devem ser atualizadas.

d) Se o segmento **atinge** o bloco, **intercepta** o segmento nele representado na sua extremidade contida no bloco, então o nó deve ser transformado em nó **vértice**, indicando essa extremidade e três cortes em seus lados no campo de valor.

As situações que se pode abstrair dos itens acima são ilustradas na Figura 4.11, onde o plano inferior representa o bloco associado ao nó **segmento** encontrado.Qualquer situação que não corresponda a alguma dessas implica no particionamento do nó original em quatro nós filhos **brancos**, que serão em seguida tratados pelo procedimento.

Sempre que um nó for particionado no processo acima, seu conteúdo informativo deve ser distribuído pelos nós **brancos** gerados, que poderão dar origem a nós **vértice**

ou **segmento**. Essa operação será referida como a reinsersão do vértice ou segmento representado pelo nó original.

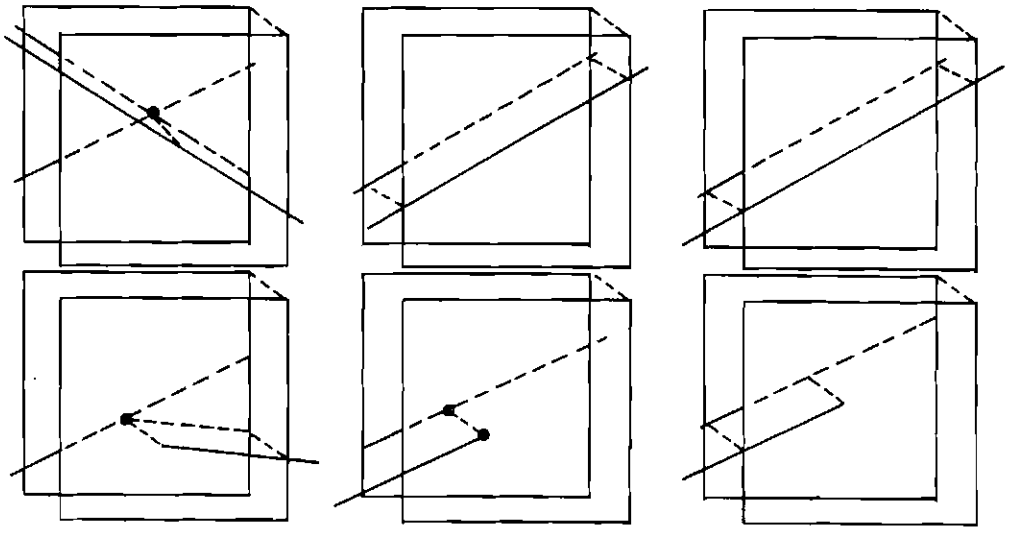

Fig. 4.11 - Posição relativa a um nó segmento.

CAPÍTULO 5

CONCLUSÕES

O propósito desta pesquisa foi analisar a viabilidade da utilização de estruturas de árvores quaternárias para representação e manipulação de imagens gráficas, abordando o problema da representação, pesquisa de adjacência, rotação por ângulos retos e operações de conjunto. Como resultados foram desenvolvidos e implementados programas em ambiente de microcomputador IBM-PC e sistema operacional DOS, em linguagem PASCAL. Nenhum recurso gráfico especial foi utilizado, além do oferecido pela placa CGA, com resolução 320x200.

Os procedimentos apresentados servem como ferramentas básicas para o desenvolvimento de aplicações como: o armazenamento e manipulação de imagens gráficas em sistemas que dependam de informações baseadas em áreas de imagens digitais; a implementação de linguagens de acesso a dados relativos a regiões e feições lineares em cartografia e sistemas geográficos de informação; o tratamento de imagens em multi resolução em visão computacional; e muitas outras em computação gráfica e áreas afins.

Usualmente imagens raster são manipuladas por procedimentos que a examinam linha por linha, entretanto as suas propriedades de inteêsse não são, em geral, "lineares" nesse sentido; outros aspectos geométricos, topológicos e estruturais estão quase sempre envolvidos. A representação por árvores reflete muitos desses aspectos, facilitando a pesquisa, que é baseada em áreas, e não linhas de imagens. Outra vantagem oferecida provém do fato de que vários procedimentos tem sua eficiência dependente do número de nós na estrutura e não do número de pixels em um bloco representado.

Nos procedimentos implementados foram usadas estruturas convencionais de árvore, embora, sempre que possível, a fim de melhorar a eficiência pela diminuição do número de chamadas recursivas e do uso de operações elementares sobre representações binárias, procurou-se explorar o conceito de código quaternário na representação de pixels e blocos.

Na determinação do nó adjacente a um certo nó numa direção dada, é proposto o uso da função que determina o menor nó adjacente, que seja maior ou igual ao nó atual, que pode ser um nó **cinza**. neste caso um procedimento recursivo é utilizado em seguida para a determinação do nó terminal descendente do nó **cinza** encontrado, que satisfaça alguma condição adicional de interêsse para a pesquisa.

É conveniente o uso, sempre que possível, na geração da árvore de uma imagem, de sua representação matricial como ponto de partida. O procedimento implementado para a geração a partir da representação raster demonstrou grande consumo de memória e tempo, pois o exame de cada pixel dá origem à criação de um nó, sendo necessária a condensação da estrutura resultante a fim de obter uma árvore quaternária mínima que represente a imagem original, o que é obtido diretamente se a imagem original é dada no formato matricial.

Pode-se acompanhar a visualização de uma imagem $2^n x 2^n$, a partir de sua árvore quaternária, segundo a ordenação do plano de imagem induzida pelo percurso em pré-ordem da estrutura. Enquanto isso, numa representação raster da mesma imagem, 2^n linhas com 2^n pixels são vizualizadas uma a uma. Para o caso de árvores quaternárias tudo se passa como se uma única "linha" de comprimento 2^{2n} fosse transmitida para o video desde o seu primeiro pixel,

que corresponde ao canto superior esquerdo da representação matricial, até o pixel associado ao canto inferior direito.

A conversão da árvore quaternária de uma imagem para a representação raster equivalente exige a pesquisa de nós terminais adjacentes que determinam segmentos que compõem cada linha gerada. Como após a geração de cada um desses segmentos são conhecidas as coordenadas do primeiro pixel do próximo segmento a ser gerado, estas são utilizadas na proposta deste trabalho para a determinação do código quaternário associado a este pixel. A partir daí, os dígitos quaternários locacionais desse código são usado para determinar diretamente o caminho a percorrer na estrutura até o nó procurado. Apenas operações elementares sobre a representação binária do código são utilizadas, evitando-se chamadas recursivas e permitindo maior eficiencia do procedimento implementado.

A eficiência das operaçôes de manipulação de imagens, implementadas neste trabalho, dependem básicamente do número de nós em sua(s) árvore(s) representativa(s). Percorrer ponteiros corresponde a endereçar posições de memória do modo mais rápido, de modo que se o total de nós em uma árvore quaternária de imagem for menor que o total de pixels de sua matriz correspondente, no caso das operações de conjunto; ou que o total de linhas dessa matriz, no caso da rotação por ângulos retos, então pode-se prever que a representação proposta torne-se competitiva com as representações matricial e raster, em uso corrente, em uma vasta classe de aplicações.

Na representação de feições lineares, o campo de valor em um nó terminal informa as extremidades do segmento representado; ou as coordenadas de um vértice com a indicação dos lados do bloco correspondente que são atravessados por segmentos que o atingem. Na implementação

proposta, o número de cortes em cada lado é efetivamente armazenado em cada nó **vértice**, embora apenas a informação de que um lado é ou não atravessado seja necessária. Em termos de consumo de memória esta estrutura pode não ser competitiva com a representação vetorial para certos fins, como o cáculo de propriedades geométricas. O que incentiva o seu uso é permitir que procedimentos de manipulação possam ser desenvolvidos de um modo independente do tipo de dado armazenado, facilitando operações envolvendo dados lineares e regionais.

O procedimento para inserção de segmentos na estrutura proposta, faz uso do conceito de código quaternário para determinar a intercesão de segmentos de reta em lados de blocos associados a nós **vértices**, bem como na determinação, pelo exame dos digitos comuns nos códigos quaternários associados às extremidades de um certo segmento, do menor bloco que o contenha. Procurou-se ainda permitir a inserção irrestrita de novos segmentos à estrutura, mesmo quando estes contenham ou pertençam a outros previamente inseridos. Neste enfoque, cada nó **segmento** na estrutura representa toda a classe dos segmentos menores nele contidos, de um modo exato e consistente.

A fim de evitar erros de arredondamento em cálculos sobre representações numéricas em ponto flutuante, é desejável que valores reais, como o coeficiente angular de uma reta ou as coordenadas da interseção entre segmentos, sejam dados sob a forma de pares ordenados de números inteiros (relativamente primos) que representem a expressão fracionária mínima desses valores, evitando operações numéricas em ponto flutuante.

Uma desvantagem da representação por árvores é sua grande variação por deslocamentos. Um mesmo objeto

pode assumir diversas representações, conforme a posição que esteja na imagem. Por exemplo, uma região homogênea quadrada $2^m x 2^m$, pode ser representada por um único ou até 2^m nós.

Embora não tenha sido objetivo do trabalho, ficou evidenciada a capacidade de compressão de imagens oferecida pela representação de imagens por árvores. Uma grande economia de memória pode ser conseguida se for observado que os nós intermediários dispensam o armazenamento do campo de valor, enquanto em nós-folha o armazenamento de ponteiros para os filhos pode ser omitidos. Além disso, no caso de imagens binárias pode-se representar apenas os nós **brancos** ou **pretos**, isto é, apenas o "fundo" ou a "figura" necessita ser representada. Além disso, no caso de árvores quaternárias lineares a representação de todos os ponteiros e dos nós **cinza** é, em geral dispensável.

A Figura 5.1 mostra uma imagem 128x128 que consiste de 4 regiões homogêneas distintas e 4 outras que representam cada uma de suas regiões (ou temas).

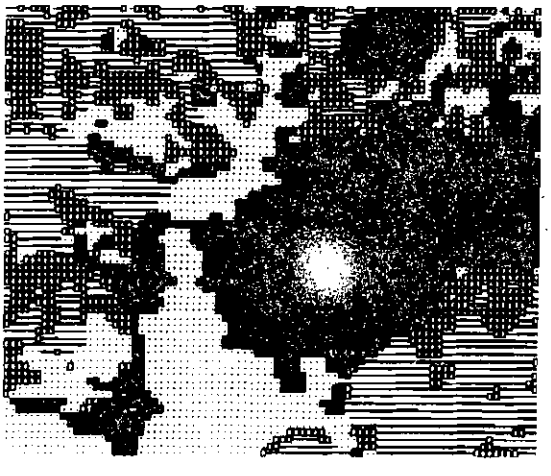

Fig. 5.1 - Imagem com quatro regiões distintas.

Para a representação matricial de uma imagem desse tipo, 16364 bytes seriam necessários, de modo que a estrutura de árvore pode ser uma alternativa viável para a representação no caso de imagens contendo regiões homogêneas sufientemente extensas, o que é comum em aplicações de computação gráfica. Na Figura 5.2, a região (1) exige 1034 nós **cinzas**, 1364 nós **brancos** e **1739 pretos**, em sua representação por árvore; a região (2), 776 **cinzas**, 1033 **brancos** e 1296 **pretos**; a região (3), 872 **cinzas**, 1141 **brancos** e 1476 **pretos**; finalmente, a região (4) exige 776 **cinzas**, 978 **brancos** e 1126 **pretos**.

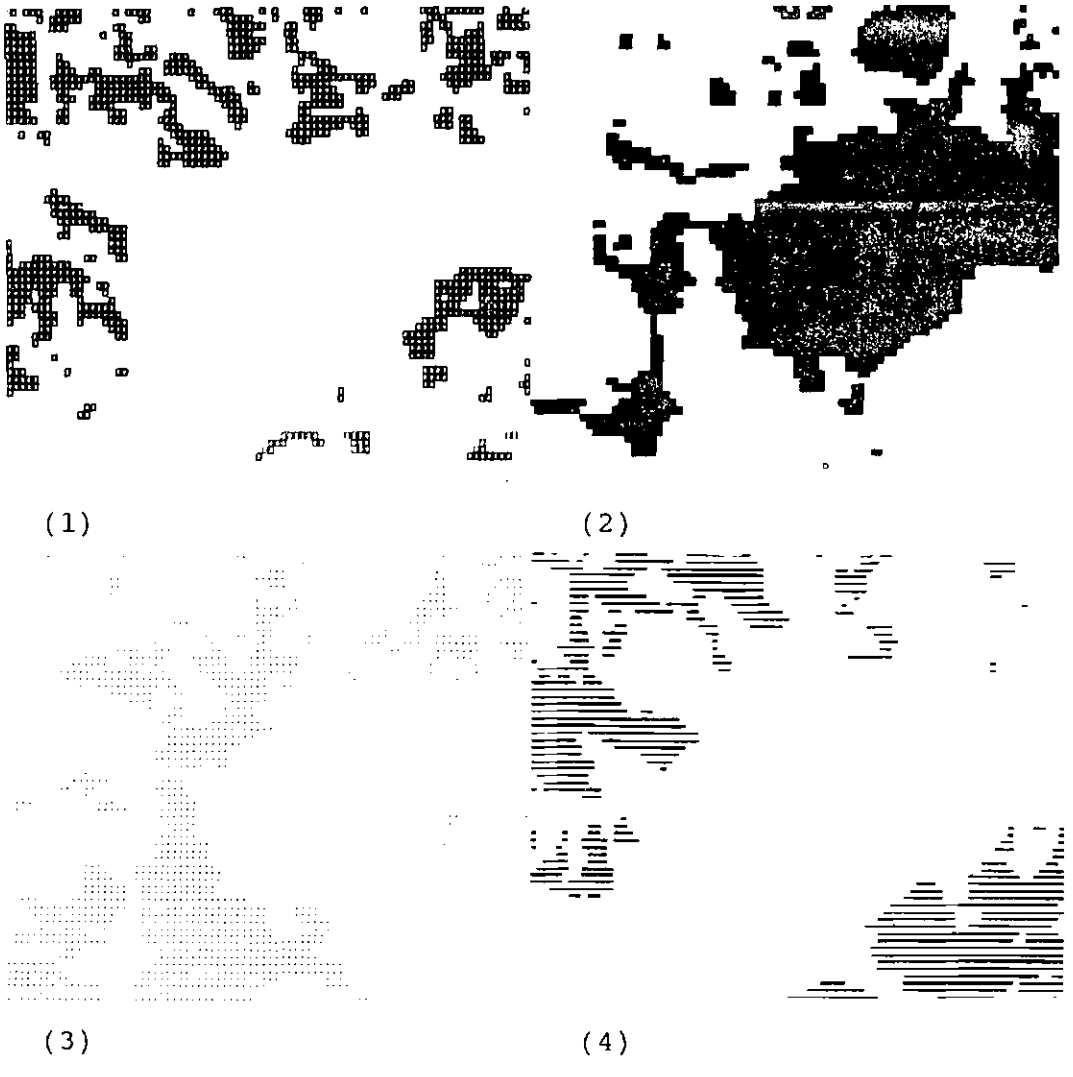

Fig. 5.2 - Regiões representadas isoladamente.

À medida em que mais regiões de interêsse sejam destacads numa imagem, o esquema de representação por árvore das imagens binárias determinadas por cada região isoladamente pode tornar-se inviável, de modo que outras alternativas precisam ser aplicadas nesse caso.

Se o critério de homogeneidade usado para a geração de nós em uma árvore quaternária for fixado em função, não da cor, mas sim da variação de cor entre os pixels examinados, torna-se possível representar, em uma única estrutura de árvore, todas as regiões de interêsse de uma imagem. Ainda assim, para imagens em escala de cinza a aplicação desses princípios raramente corresponderia a expressiva economia de memória. Uma alternativa para representar essa classe de imagem, que pode contornar o problema do consumo de memória, é obtido através da identificação da imagem a tantas imagens binárias quantos sejam os dígitos binários necessários para representar o total de níveis de cinza identificados em seus pixels. Neste enfoque, que sinaliza uma possível extensão para este trabalho, por exemplo, uma imagem onde possam ocorrer 256 níveis de cinza seria identificada a 8 imagens binárias representadas por árvores quaternárias; a primeira seria determinada pelos valores associados aos dígitos mais significativos dos bytes que informam o nível de cinza em cada pixel da imagem, a oitava árvore corresponderia à imagem binária determinada pelos dígitos (bits) menos significativos de cada byte.

Finalmente, deve ser ressaltado que estruturas de árvore contituem um tema vastamente explorado em computação, o que fornece recursos teóricos e práticos para fundamentar novas pesquisas e aplicações em computação gráfica como o modelamento geométrico de sólidos por estruturas de árvore que extendem os conceitos

desenvolvidos para árvores quaternárias para o caso tri-dimensional, conhecidas na literatura por "octrees".

REFERÊNCIAS BIBLIOGRÁFICAS

AHUJA, N. On Approaches to Poligonal Decomposition for Hierarchical Image Representation. **Computer Vision Graphics and Image Processing**, 24(2):200-214, Nov. 1983.

AMERICAN NATIONAL STANDARDS INSTITUTE COMMITTEE X3H31 (ANSI). American National Standards for the Functional Specification of theProgrammer's Hierarchical Interactive Graphics System (PHIGS). New York, 1985. (ANSI Standard X3H31/85-05 X3H3/85-21).

ATKINSONS, H.H.; GARGANTINI, I. Counting Regions, Holes, and Their Nesting Level in Time Proportional to the Border. **Computer Vision Graphics and Image Processing**, 29(2):196-215, Feb. 1985.

BAUER, M.A. Set Operations on Linear Quadtrees. **Computer Vision, Graphics, and Image Processing**, 29(2):248-258, Feb. 1985.

DYER, C. Computing the Euler Number of an Image from its Quadtree. **Computer Graphics and Image Processing**, 13(3):270-276, July 1980.

GARGANTINI, I. An Effective Way to Represent Quadtrees. **Communicatons of ACM**, 25(12):905-910, Dec. 1982.

HARMON, L.D. The Recognition of Faces. **Scientific American**, 229(5):70-82, Nov. 1973.

HUNTER, G.; STEIGLITZ, K. Operations on Images Using Quadtrees. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 1(2):145-153, Apr. 1979a.

_____Linear Transformation of Pictures Represented by Quadtrees, **Computer Graphics and Image Processing**, 10(3):289-296, July 1979b.

KLINGER, A. Patterns and search statistics. In: RUSTAGI, J.S., ed. **Optimizing Methods in Statistics**. New York, Academic, 1971. p. 303-337.

KLINGER, A.; DYER, C. Experiments on Picture Representation Using Regular Decomposition. **Computer Graphics and Image Processing**, 5(1):68-105, Mar. 1976.

KLINGER, A.; RHODES,M.L. Organization and Access of Image Data by Areas. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 1(1):50-60, Jan. 1979.

KNUTH, D.E. The art of computer programming. Fundamental algorithms, Menlo Park, CA, Addisson Wesley, 1973.

MASCARENHAS, M.; VELASCO, F. **Processamento digital de imagens.** São Paulo, Editora da Universidade de São Paulo, 1984.

MORTON, G.M. A Computer Oriented Geodetic Data Base and a New Technique in File Sequencing. Ottawa, Canada IBM, Mar. 1966.

PEUQUET, D.J. Data Structures for a Knowledge-based Geographic Information Sistem. In: INTERNATIONAL SYMPOSIUM ON SPATIAL DATA HANDLING, Zurich, Geographisches Institute/Abteilung Cartographie /EDV, 1984, p. 372-391.

RHODES, M.L.; KLINGER, A. Modifying Graphics Images. In: RUSTAGI, J.S., ed. **Optimizing methods in statistics.** New York, Academic, 1976. p 385-411.

ROSENFELD, A. Quadtrees and Pyramids for Pattern Recognition and Image Processing. In: COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, San Francisco, CA, June 19-23, 1980. Proceedings. Los Angeles, CA, IEEE Computer Society, 1980, p. 524-529.

SAMET, H. Region Representation: Quadtrees from Boundary Codes. **Communications of ACM**, 23(3):163-170, Mar. 1980a.

_____Region Representation: Quadtree From Binary Array. **Computer Graphics and Image Processing**, 13(1):88-93, May 1980b.

_____An Algorithm for Converting Raster to Quadtree. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 3(1):93-95, Jan. 1981a.

_____Connected Component Labeling Using Quadtrees. **Journal of ACM**, 28(3):487-501, July 1981b.

_____Computing Perimeters of Images Represented by Quadtrees. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 3(6):683-687, Nov. 1981c.

SAMET, H. Neighbor Finding Techniques for Images Represented by Quadtrees. **Computer Graphics and Image Processing**, 18(1):37-57, Jan. 1982.

_____Algorithms for the Convertion of Quadtrees to Raster. **Computer Vision Graphics, and Image Processing**, 26(1):1-16, Apr. 1984a.

_____The Quadtree and Related Hierarchical Data Structures. **Computing Surveys**, 16(2):187-260, June 1984b.

_____A Top-Down Quadtree Traversal Algorithm. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 7(1):95-98, Jan. 1985.

SAMET, H.; SHAFFER, A. A Model for the Analysis of Neighbor Finding in Pointer-Based Quadtrees. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 7(6):717-720, Nov. 1985.

SAMET, H.; TAMMINEN, M. Computing Geometric Properties of Images Represented by Linear Quadtrees. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, 7(2):229-240, Mar. 1985.

SAMET, H.; WEBBER, R.E. On Encoding Boundaries with Quadtrees. **IEEE Transactions on Pattern Analisys and Machine Intelligence**, 6(3):365-369, May 1981.

SAMET, H.; DYER, C.; ROSENFELD, A. Region Representation: Boundary Codes from Quadtrees. **Communications of ACM**, 23(3):171-179, Mar. 1980.

SAMET, H.; SHAFFER, C.A.; WEBBER, R.E. The Segment Quadtree: A Linear Quadtree-based Representation for Linear Features.In: COMPUTER SOCIETY CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION, San Francisco, CA, June 19-23, 1985. Proceedings. Los Angeles, CA, IEEE Computer Society, 1985, p. 524-529.

SAMET, H.; ROSENFELD, A.; SHAFFER, C.A.; WEBBER, R. A Geographic Information System Using Quadtrees. **Pattern Recognition**, 17(6):647-656, Dec. 1984.

SCHNEIER, M. Two Hierarquical Linear Feature Representations: Edge Pyramids and Edge Quadtree. **Computer Graphics and Image Processing**, 17(3):211-224, 1981a.

SCHNEIER, M. Calculation of Geometric Properties Using Quadtrees. **Computer Graphics and Image Processing**, 16(3):296-303, July 1981b.

TANIMOTO, S.L.; PAVLIDS, T. A Hierarquical Data Structure for Picture Processing. **Computer Graphics and Image Processing**, 4(2):104-119, June 1975.

WHITE, R.G. Compressing Image Data With Quadtrees. **Dr. Dobb's Journal**, 12(3):16-25, Mar. 1987.