Introdução sobre Estruturas de Dados Espaciais e Árvore de Quadrante

Henrique Dantas Pighini 21.1.4025
Caio Silas de Araujo Amaro 21.1.4111
Arthur Henrique Santos Celestino 21.1.4019
Marcella Silveira Campos 21.1.4008
Mauro Lucio Afonso Paulino dos Santos Filho 21.1.4002

Dados Escalares vs Dados Espaciais

Dados Escalares

1. As variáveis escalares são usadas para representar objetos de dados de tamanho fixo individuais, tais como inteiros e ponteiros.

2. Exemplos:

   Tipos primitivos: int, char, bool.

   Campo Estado de uma Struct pessoa.

Inscrição	Nota	Estado	Cidade	Curso
00467354	47,8	MG	VIÇOSA	ADMINISTRAÇÃO
00085820	52,0	MG	UBERLÂNDIA	DIREITO
00015022	51,0	MG	ALFENAS	ENGENHARIA CIVIL
00403068	8,0	MG	VARGINHA	ENGENHARIA QUÍMICA
00130230	36,3	MG	UBERABA	MEDICINA VETERINÁRIA

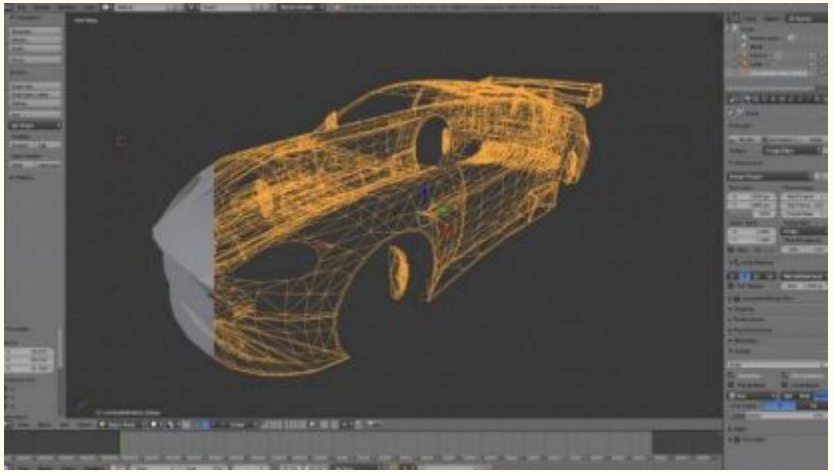
Dados Espaciais

1. Dado espacial é qualquer tipo de dado que descreve fenômenos aos quais esteja associada alguma dimensão espacial.

2. Exemplos:

   Latitude, Longitude.

   Posição de um ponto em um modelo 3d.



Latitude	Longitude
-20º 45' 14"	-42º 52' 55"
-18° 55' 07''	-48° 16' 38''
-21° 25' 45''	-45° 56' 50'
-21° 33' 05''	-45° 25' 49''
-19° 44' 54''	-47° 55' 55''

Dados Escalares + Dados Espaciais

Inscrição	Nota	Estado	Cidade	Curso
00467354	47,8	MG	VIÇOSA	ADMINISTRAÇÃO
00085820	52,0	MG	UBERLÂNDIA	DIREITO
00015022	51,0	MG	ALFENAS	ENGENHARIA CIVIL
00403068	8,0	MG	VARGINHA	ENGENHARIA QUÍMICA
00130230	36,3	MG	UBERABA	MEDICINA VETERINÁRIA

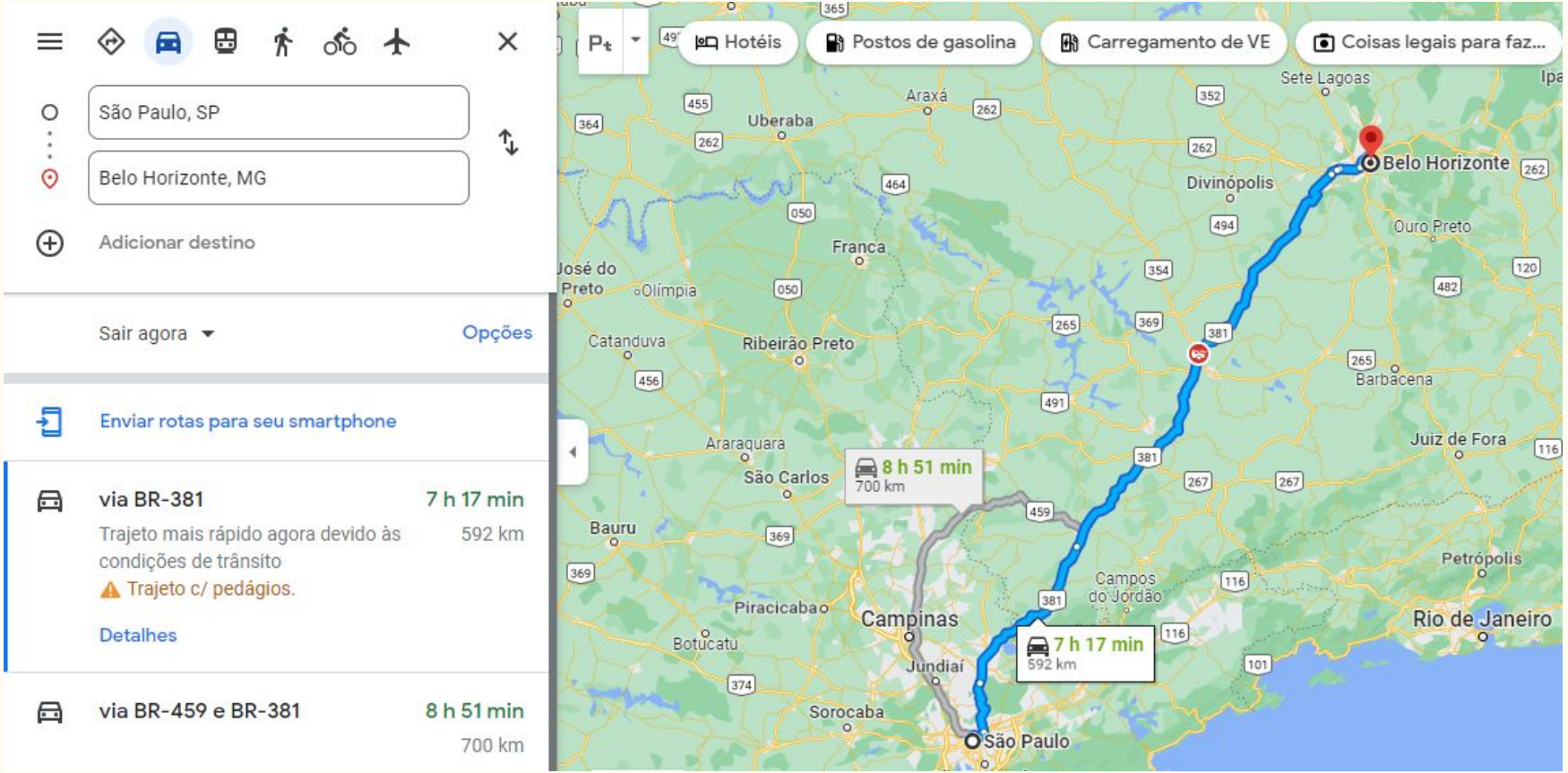
Qual a média dos alunos que moram próximo de Ouro Preto?

Quais alunos moram no máximo a até 100 km de Uberaba?

Dados Escalares + Dados Espaciais

	Dados Escalares				Dados Espaciais	
Inscrição	**Nota**	**Estado**	**Cidade**	**Curso**	**Latitude**	**Longitude**
00467354	47,8	MG	VIÇOSA	ADMINISTRAÇÃO	-20º 45' 14"	-42º 52' 55"
00085820	52,0	MG	UBERLÂNDIA	DIREITO	-18° 55' 07"	-48° 16' 38"
00015022	51,0	MG	ALFENAS	ENGENHARIA CIVIL	-21° 25' 45"	-45° 56' 50'
00403068	8,0	MG	VARGINHA	ENGENHARIA QUÍMICA	-21° 33' 05"	-45° 25' 49"
00130230	36,3	MG	UBERABA	MEDICINA VETERINÁRIA	-19° 44' 54"	-47° 55' 55"

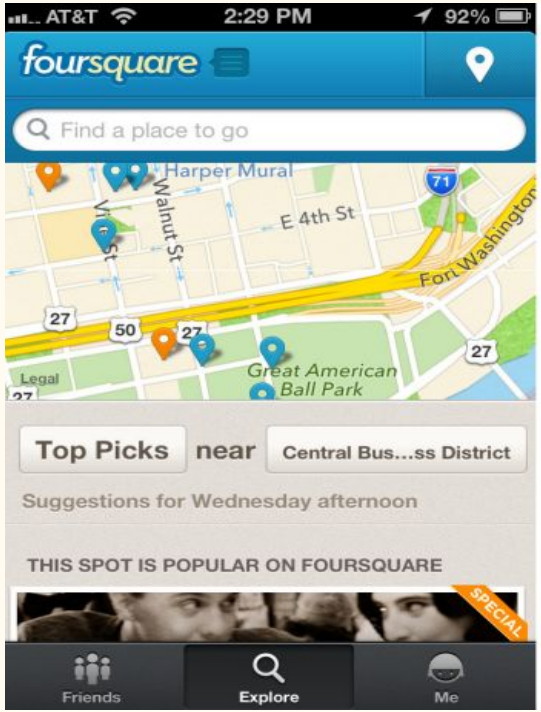
Dados Escalares + Dados Espaciais

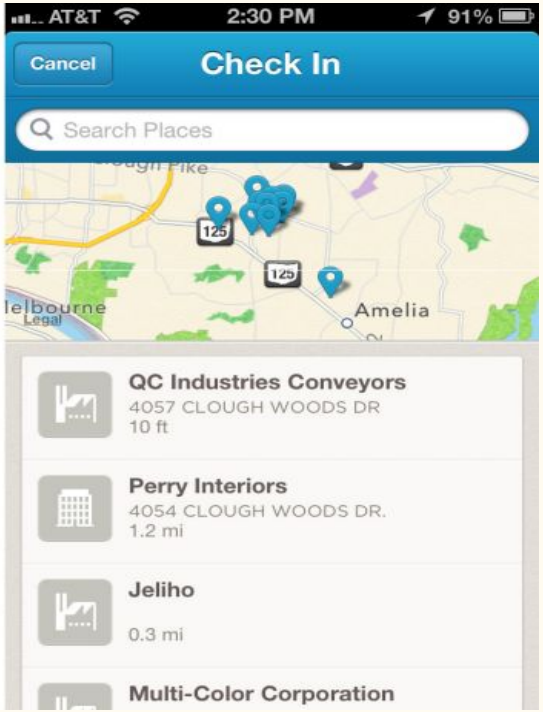


Tipos centrais de aplicação de Dados Espaciais

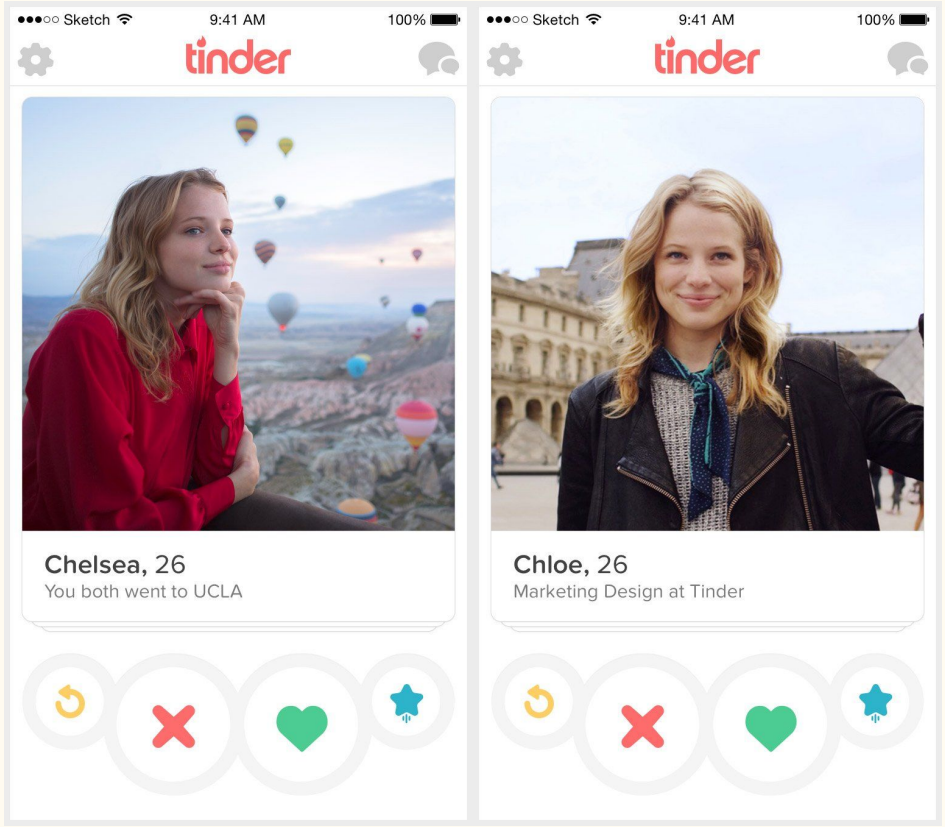
1. Dados geográficos ou georreferenciados.
2. Sistema CAD(Desenho guiado por computador)
3. Bancos de dados espaciais.

Utilização dos Dados Espaciais



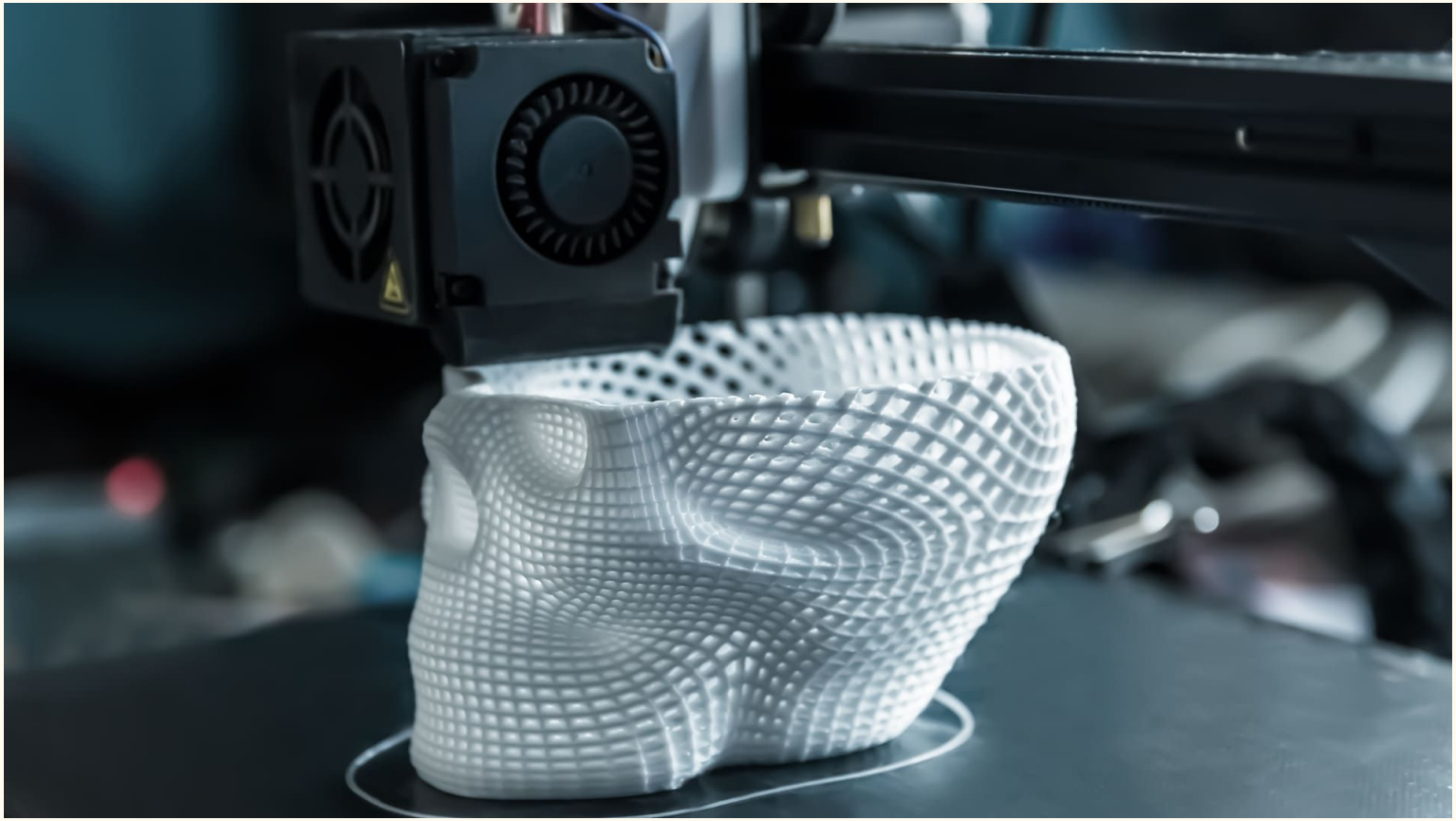


Utilização dos Dados Espaciais

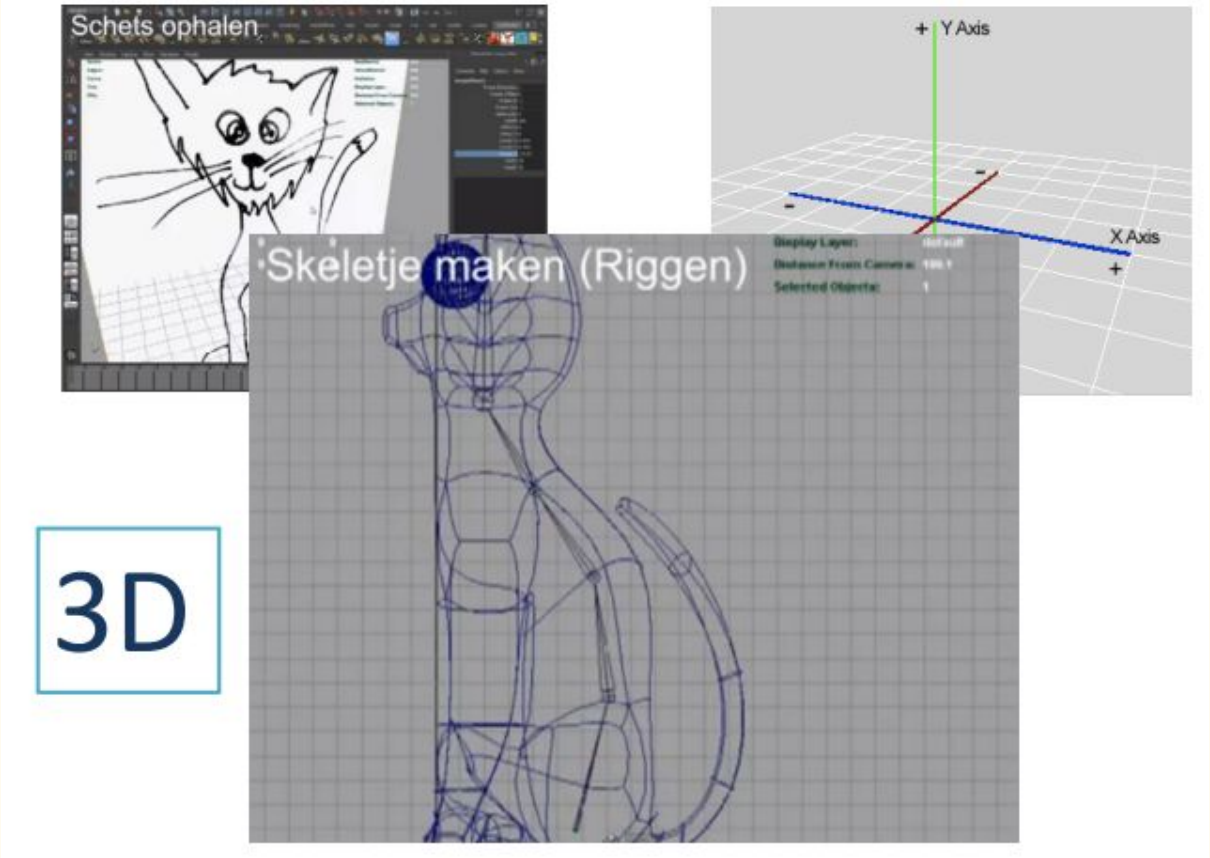


Apenas mapeamento e localização ??

Utilização dos Dados Espaciais



Utilização dos Dados Espaciais



Banco de Dados

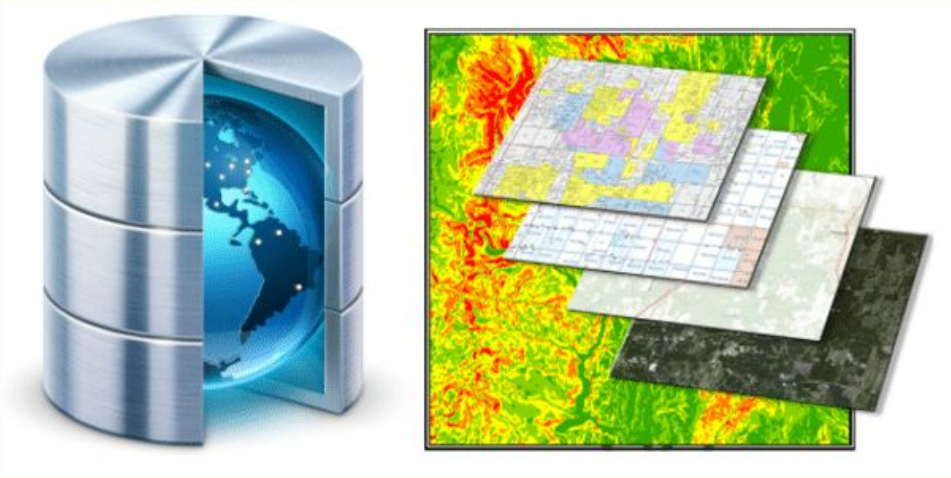
Banco de Dados

- Coleção de dados estruturada.
    - Representada por tabelas com campos.
- Armazenadas eletronicamente.
    - Em nuvens ou servidores.
- Vários tipos de BD

Banco de Dados Espaciais

1. Focados em dados espaciais.
   a. coordenadas, CEPs, pontos

2. Capaz de gerar e calcular dados geométricos
   a. polígonos, área
3. Amplas maneiras de utilizar o Banco de Dados
   a. • SIG (Cartografia)
   b. • CAD (Computer-Aided Design)
   c. Bancos de dados espaciais de aplicações Web (Foursquare, Tinder, etc).



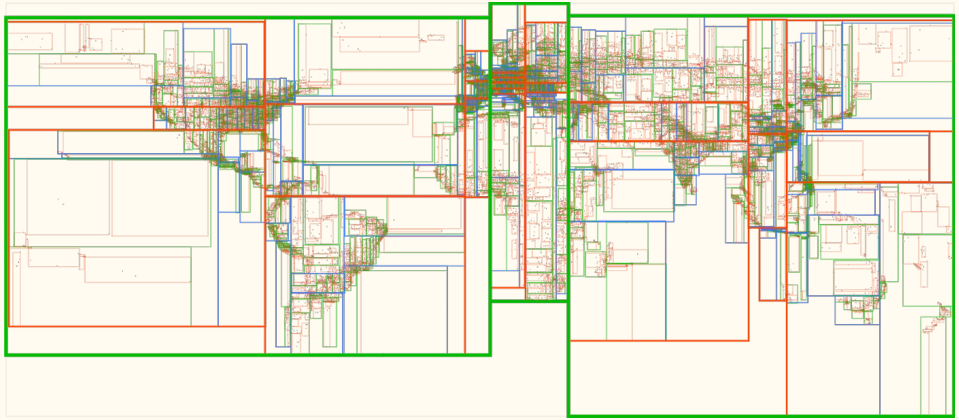
SIG - Sistema de Informação Geográfica

- Método de manipulação de dados Geográfica.
  - Várias maneiras de manusear os dados.

- Utilizado em GPS e APPs que necessitam de localização.

Estrutura de Dados Espaciais

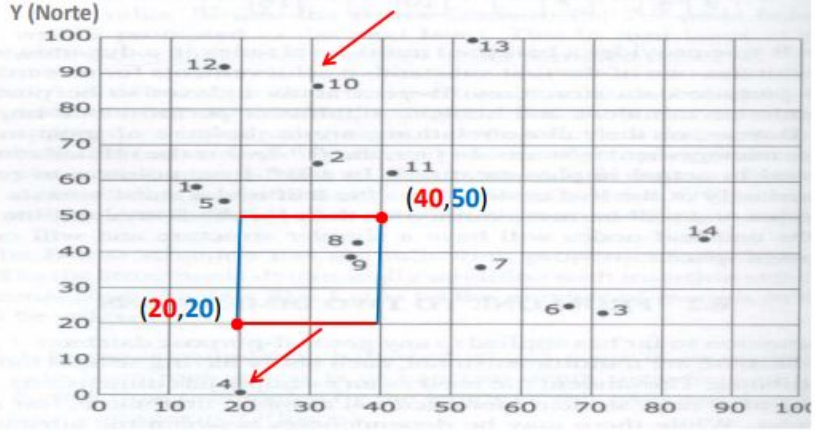
- Conjunto de tudo apresentado.
  - Dados espaciais que representam um espaço.
  - Armazenados em um banco de dados.
  - Sendo implementados por um método espacial.
- Métodos:
  - QuadTree
  - Grid
  - Árvore kD
  - Árvore R



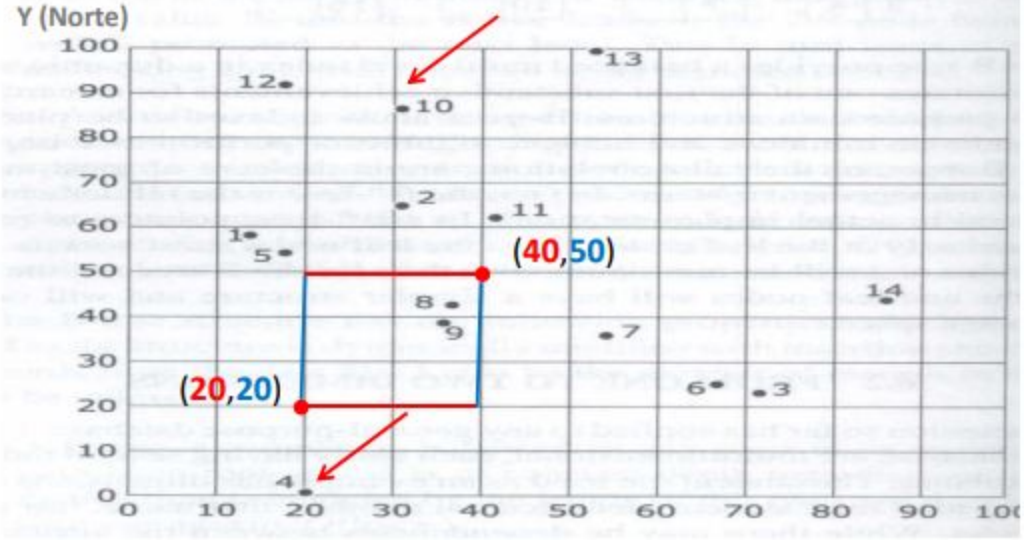
Estrutura de Dados Espaciais

- É possível realizar manipulações a partir das estruturas.
  - Medir distâncias, perímetro, áreas
  - Calcular a conectividade e o caminho mais curto entre dois pontos
  - Analisar pontos e linhas dentro de um polígono
  - Realizar buscas por região (intervalo)

Id	Local	X (Leste)	Y (Norte)
1	Newcastle Museum	14	58
2	Waterworld	31	65
3	Gladstone Pottery Museum	74	23
4	Trentham Gardens	20	00
5	New Victoria Theater	18	55
6	Beswick Pottery	66	25
7	Coalport Pottery	54	36
8	Spode Pottery	37	43
9	Minton Pottery	36	39
10	Royal Doulton Pottery	31	87
11	City Museum	41	62
12	Westport Lake	17	92
13	Ford Green Hall	53	99
14	Park Hall Country Park	86	44



Id	Local	X (Leste)	Y (Norte)
1	Newcastle Museum	14	58
2	Waterworld	31	65
3	Gladstone Pottery Museum	74	23
4	Trentham Gardens	20	00
5	New Victoria Theater	18	55
6	Beswick Pottery	66	25
7	Coalport Pottery	54	36
8	Spode Pottery	37	43
9	Minton Pottery	36	39
10	Royal Doulton Pottery	31	87
11	City Museum	41	62
12	Westport Lake	17	92
13	Ford Green Hall	53	99
14	Park Hall Country Park	86	44

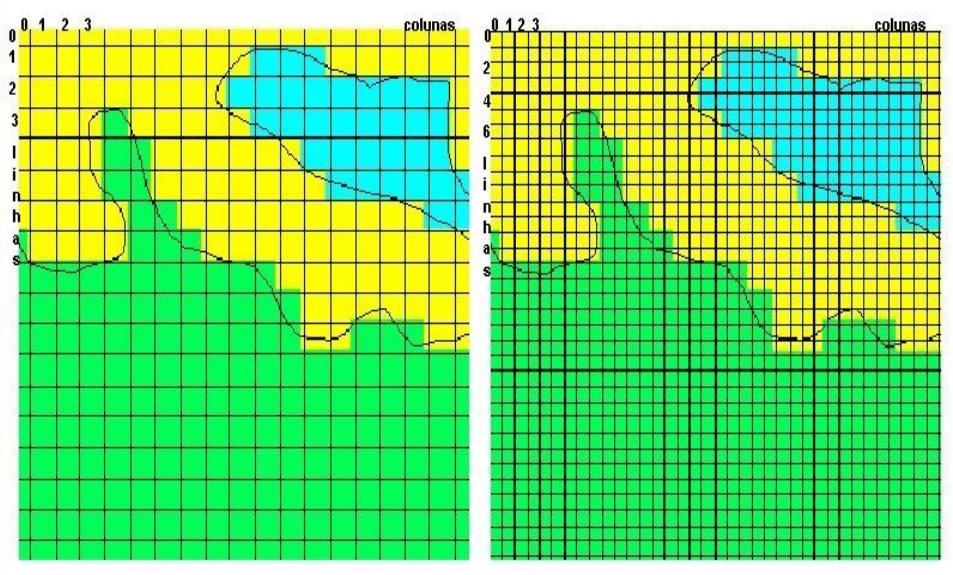


Dados Vetoriais

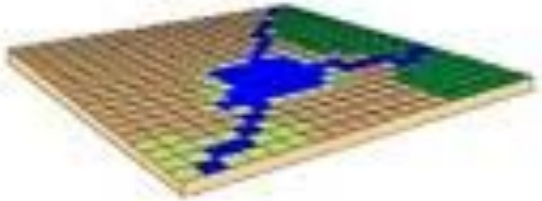
O dado Vetorial constitui uma maneira de representar elementos do mundo real dentro do ambiente SIG.

Dados Matriciais

O dado Matricial é composto por linhas e colunas de pixels, onde cada pixel representa uma região do mundo real.

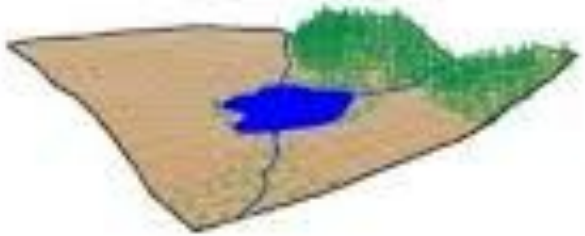


Dados Matriciais

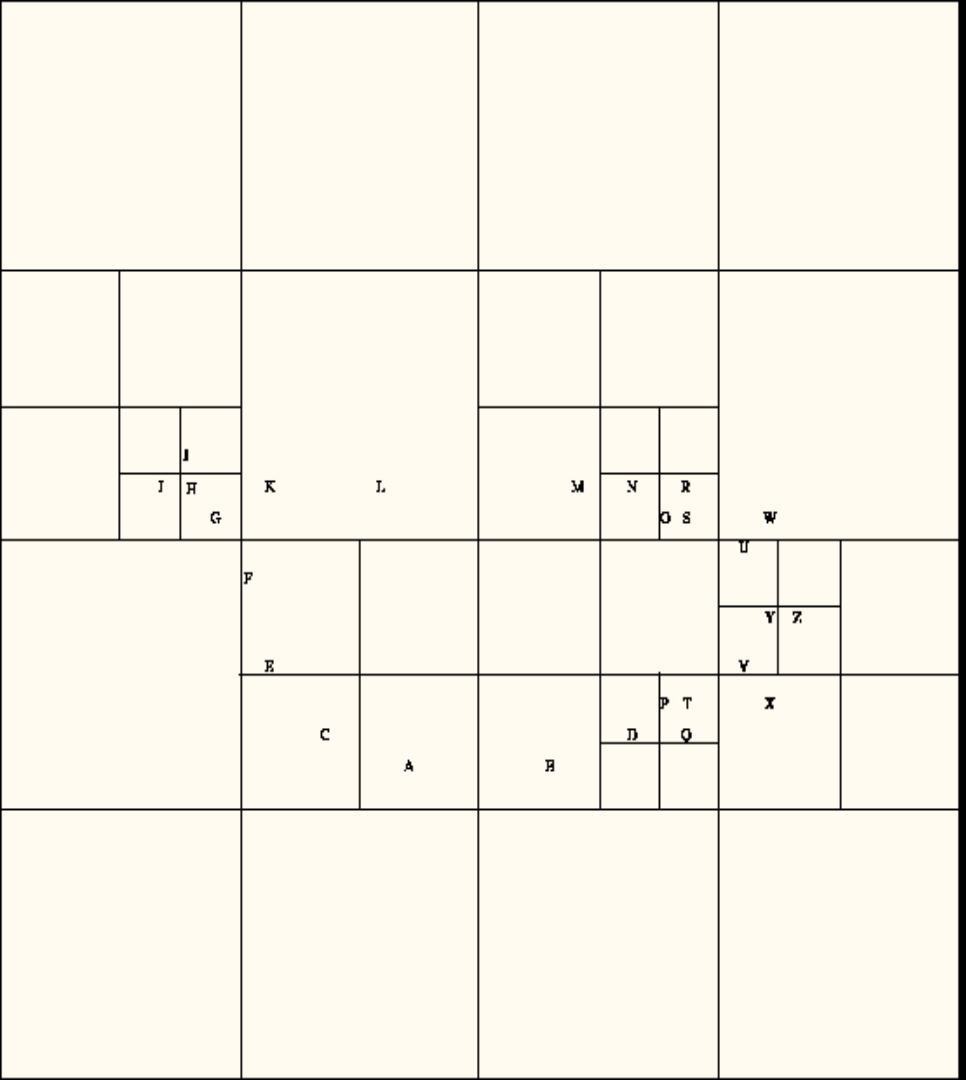


Dados Vetoriais

Mundo Real

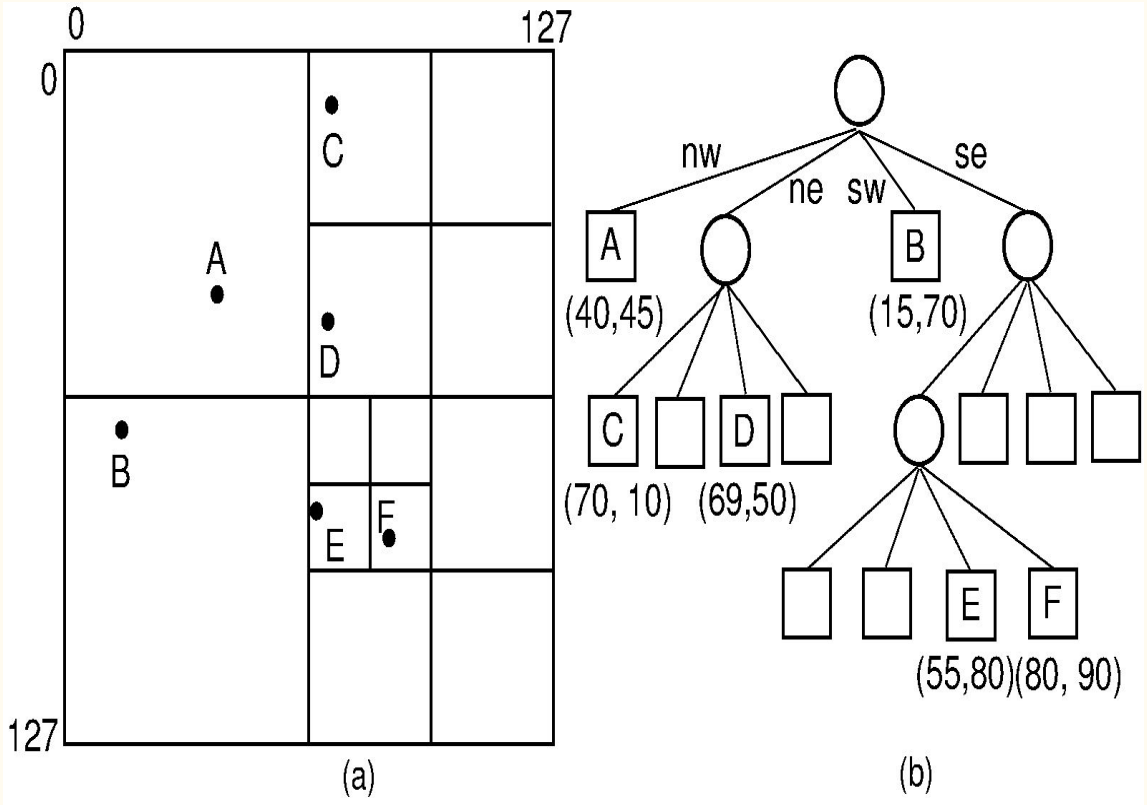


O que é a Árvore Quadrante



- É uma estrutura de dados usada para codificar imagens.
- Cada imagem é dividida em quatro quadrantes.
- Cada quadrante pode ser dividido recursivamente em mais quatro quadrantes e assim por diante

1. O espaço bidimensional é dividido em 4 caixas.
2. Se uma caixa tem um ou mais pontos nela, construa um objeto filho, armazenando nele o espaço bidimensional da caixa
3. Se uma caixa não possui pontos, não crie um filho para ela
4. Faça recursão para todos os filhos.



Aplicações:

- Tratamento de fotografias (por exemplo, retirar olhos vermelhos de fotos)
- Ecografias (identifica tumores pela cor da imagem)
- Games (detectando se algum projétil atingiu certo local)
- Compressão de imagens
- Videochamadas (mandando apenas o que foi alterado na imagem)

Vantagens:

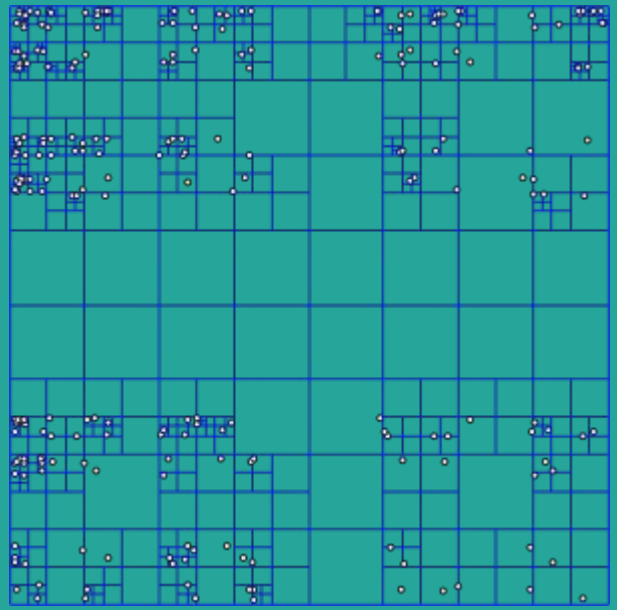
- Útil na compactação de imagens
- Tem estrutura enxuta (gasta-se pouco tempo para chegar aos filhos em geral)
- Não necessita de balancear a cada inserção, fazendo cada inserção não afetar o desempenho da árvore.
- Pode ser usada para realizar mudanças (como rotações) na imagem com facilidade.

Desvantagens:

- Em imagens complexas, com muitas cores e detalhes, a árvore pode ficar muito grande e complexa, se tornando por vezes até maior que a imagem original.
- Devido ao problema acima, também pode haver alto gasto de CPU
- Somente imagens bidimensionais podem ser tratadas com ela.

Há mais de um tipo de Árvore Quadrante. Dentre eles, temos:

-> Region
-> Point
-> Point-Region
-> Edge
-> Polygon Map
-> Compressed



Region QuadTree

A Region QuadTree é usada para representar uma partição do espaço em duas dimensões, quebrando a região em quatro quadrantes iguais, sub-quadrantes, e então cada nó folha consiste em informação correspondente a uma sub-região. Cada nó na árvore ou é associado com exatamente quatro filhos ou nenhum.

Uma Region QuadTree com uma altura **n** pode ser implementada para representar uma imagem que consiste de **2nx2n** pixels, em que cada pixel tem o valor de 0 ou 1. O nó raiz pode ser usado para representar uma região inteira da imagem. Se os pixels em qualquer região não são inteiramente 0 ou 1, ela é subdividida. Nessa aplicação, cada nó folha é usado para representar um bloco de pixels que são ou todos 0 ou todos 1.

Point QuadTree

A Point QuadTree é uma adaptação de uma árvore binária implementada para representar uma informação de um ponto 2-dimensional. Características de todas QuadTrees são compartilhadas pela Point QuadTree.

Elas são geralmente muito eficientes em comparar informações de pontos 2-dimensional, usualmente executado em tempo de O(log n). As Point QuadTrees são uma menção valiosa para a completude, mas as árvores k-d as superam como ferramentas para busca binária generalizada.

Point-Region QuadTree

Na Point-Region QuadTree (também referida como PR QuadTree) cada nó tem exatamente quatro filhos ou é uma folha. Isto é, A PR QuadTree é uma árvore de modelo 4-ária. A PR QuadTree representa uma coleção de informações de pontos de duas dimensões, decompondo a região que contém os pontos em quatro sub quadrantes, e assim por diante, até que nenhum nó folha tenha mais que um único ponto. Em outras palavras, se uma região contém zero ou um ponto, então ela é representada por uma PR QuadTree com um único nó folha. Se a região contém mais que um único ponto, então a região se divide em quatro quadrantes iguais.

Quadtree edge

Quadtree edge são usados para armazenar linhas em vez de pontos. As curvas são aproximadas e subdivide as células em uma resolução muito fina, especificamente até que haja um segmento de linha única por célula. Perto de cantos/vértices, os quadtree edge continuam se dividindo até atingirem seu nível máximo de decomposição.

A decomposição quadtree é uma técnica adequada para detecção de bordas porque há uma diferença distinta entre bordas e pixels vizinhos. Se a decomposição quadtree é realizada sobre as imagens, as folhas do quadtree ou o nível acima das folhas representarão uma intensidade máxima desses pixels.

Polygon map

O quadtree de mapa poligonal (ou PM Quadtree) é uma variação de quadtree que é usado para armazenar coleções de polígonos que podem ser degenerados (o que significa que eles têm vértices isolados ou bordas).

Uma grande diferença entre quadtrees pm e quadtrees de borda é que a célula em consideração não é subdividida se os segmentos se encontram em um vértice na célula.

Compressed

Usada para compressão de imagem, funciona dividindo recursivamente a imagem em quatro subespaços com cada um segurando a cor RGB média e o erro,determinando a cor para seus subespaços. O limiar é definido com base nesse erro e ajuda a árvore a determinar se um nó deve ser dividido mais ou não.

Pseudo-Código

```
struct XY
{
    float x;
    float y;
}

struct AABB
{
    XY center;
    float halfDimension;

    function __construct(XY center, float halfDimension) {...}
    function containsPoint(XY point) {...}
    function intersectsAABB(AABB other) {...}
}
```

Classe Principal

```
class QuadTree
{
    // Arbitrary constant to indicate how many elements can be stored in this quad tree node
    constant int QT_NODE_CAPACITY = 4;

    // Axis-aligned bounding box stored as a center with half-dimensions
    // to represent the boundaries of this quad tree
    AABB boundary;

    // Points in this quad tree node
    Array of XY [size = QT_NODE_CAPACITY] points;

    // Children
    QuadTree* northWest, northEast, southWest, southEast;

    // Methods
    function __construct(AABB _boundary) {...}
    function insert(XY p) {...}
    function subdivide() {...} // create four children that fully divide this quad into four quads of equal area
    function queryRange(AABB range) {...}
}
```

Função de Inserção (pt 1)

```
class QuadTree
{
    ...
    // Insert a point into the QuadTree
    function insert(XY p){
        // Ignore objects that do not belong in this quad tree
        if (!boundary.containsPoint(p))
            return false; // object cannot be added

        // If there is space in this quad tree and if doesn't have subdivisions, add the object here
        if (points.size < QT_NODE_CAPACITY && northWest == null){
            points.append(p);
            return true;
        }

        // Otherwise, subdivide and then add the point to whichever node will accept it
        if (northWest == null)
            subdivide();
        // We have to add the points/data contained in this quad array to the new quads if we only want
        // the last node to hold the data

***continua***
```

Função de Inserção (pt 2)

```
***continua***
        if (northWest->insert(p)) return true;
        if (northEast->insert(p)) return true;
        if (southWest->insert(p)) return true;
        if (southEast->insert(p)) return true;

        // Otherwise, the point cannot be inserted for some unknown reason (this should never happen)
        return false;
    }
} (CLASS BRACKET)
```

Função de Pesquisa (pt 1)

```
class QuadTree
{
    ...

    // Find all points that appear within a range
    function queryRange(AABB range)
    {
        // Prepare an array of results
        Array of XY pointsInRange;

        // Automatically abort if the range does not intersect this quad
        if (!boundary.intersectsAABB(range))
            return pointsInRange; // empty list

        // Check objects at this quad level
        for (int p = 0; p < points.size; p++)
        {
            if (range.containsPoint(points[p]))
                pointsInRange.append(points[p]);
        }
```

continua

Função de Pesquisa (pt 2)

```
***continua***
        // Terminate here, if there are no children
        if (northWest == null)
            return pointsInRange;

        // Otherwise, add the points from the children
        pointsInRange.appendArray(northWest->queryRange(range));
        pointsInRange.appendArray(northEast->queryRange(range));
        pointsInRange.appendArray(southWest->queryRange(range));
        pointsInRange.appendArray(southEast->queryRange(range));

        return pointsInRange;
    }
} (CLASS BRACKET)
```

Bibliografia:

- Quadtree. Disponível em: https://en.wikipedia.org/wiki/Quadtree. Acesso em 23/10.
- Estruturas de Dados Espaciais. Dísponivel em: https://moodlepresencial.ufop.br/pluginfile.php/1204661/mod_resource/content/3/ApresentacaoDadosEspaciaisLeandro.pdf. Acesso em 23/10.
- Quadtree. Disponível em: https://www.geeksforgeeks.org/quad-tree/. Acesso em 23/10
- Region QuadTree. Disponível em: https://www.tutorialspoint.com/region-quadtrees-in-data-structure. Acesso em 23/10
- Point QuadTree. Disponível em: https://www.tutorialspoint.com/point-quadtrees-in-data-structure. Acesso em 23/10
-Quadtree edge. Disponível em : Região Quad-Tree Base de detecção de borda para imagens médicas - PMC (nih.gov). Acesso em 24/10
-Quadtree compression, Disponível em: Página de Vercy . Acesso em 24/10
-Dados espaciais. Disponível em: http://www.csr.ufmg.br/geoprocessamento/publicacoes/Modelagem%20de%20dados%20geografico.PDF. Acesso em 24/10
-Estrutura de dados espaciais. Disponível em: http://www.decom.ufop.br/guilherme/BCC203/geral/ed2_introducao-estruturas-dados-espaciais_josiane.pdf. Acesso em 24/10