UM ESTUDO SOBRE ARQUIVOS VETORIAIS PARA A VISUALIZAÇÃO DE MAPAS NA *WEB*

CARLA CRISTINA FONSECA FERREIRA[1] MARCELO GATTASS[1] LUIZ HENRIQUE DE FIGUEIREDO[2,1]

[1]TECGRAF – GRUPO DE TECNOLOGIA EM COMPUTAÇÃO GRÁFICA, DEPARTAMENTO DE INFORMÁTICA, PUC–RIO
RUA MARQUÊS DE SÃO VICENTE, 225, 22453-900 RIO DE JANEIRO, RJ, BRASIL
TELEFONE: (021) 512-5984 FAX: (021) 259-2232
{CARLA, GATTASS, LHF}@TECGRAF.PUC-RIO.BR

[2]LNCC – LABORATÓRIO NACIONAL DE COMPUTAÇÃO CIENTÍFICA
AVENIDA GETÚLIO VARGAS, 333, 25651-070 PETRÓPOLIS, RJ, BRASIL

RESUMO

Este trabalho apresenta um estudo sobre a transmissão de mapas na *World Wide Web*. Mapas são representações aproximadas da superfície terrestre, que projetam cada ponto do globo em uma superfície plana. A característica marcante das figuras que representam mapas é a presença de linhas poligonais com um grande número de pontos. Os mapas, representados nos formatos vetoriais atuais, ocupam arquivos muito grandes para serem transmitidos pela *web*. Atualmente, a utilização de formatos de imagens na *web* é comum, isto é, primitivas gráficas do tipo linhas, textos e áreas preenchidas são transformadas em uma matriz de *pixels*. Porém, estes formatos geram figuras que têm baixa resolução e não podem ser escaladas.

Este trabalho propõe estratégias para reduzir o tamanho dos arquivos através da restrição da precisão dos pontos, da eliminação de pontos repetidos e da codificação eficiente das coordenadas. São mostrados exemplos reais para avaliar as estratégias propostas e são feitas recomendações baseadas nesses exemplos.

ABSTRACT

This paper presents a study on the transmission of maps through the World Wide Web. Maps are approximate representations of the Earth's surface in which points of the globe are projected on a plane. An important feature of figures that represent maps is that they have polygonal lines with a large number of points. With current vector formats these points generate files that are too large to be transmitted through the web. The use of image formats transform graphic primitives like lines, texts and filled areas in a matrix of pixels. Although common in the web today, these formats only provide poor resolution figures that do not scale well.

This paper proposes strategies to reduce the size of the files by restricting the precision of the points, eliminating repeated points, and performing an efficient coding of the coordinates. Real examples are shown to evaluate the proposed strategies, and recommendations are made based on these examples.

INTRODUÇÃO

Sistemas de Informação Geográfica, ou SIGs, são sistemas de informação construídos para armazenar, analisar e manipular dados geográficos, ou seja, dados que representam objetos e fenômenos em que a localização geográfica é uma característica inerente e indispensável para tratá-los. Dados geográficos são coletados a partir de diversas fontes e armazenados via de regra nos chamados *bancos de dados geográficos* [1]. SIGs têm ocupado um papel cada vez mais importante em diversas atividades humanas e a Internet é um veículo fundamental para a divulgação dessas informações.

A visualização de mapas geográficos na *World Wide Web* não encontra um suporte adequado na atual tecnologia de arquivos de figuras. Na maioria dos servidores de *web* que apresentam informações geográficas, os mapas são transmitidos como imagens, ou seja, há a transformação de suas primitivas gráficas do tipo linhas, textos e áreas preenchidas em uma matriz de *pixels*.

Normalmente, as imagens de mapas são transmitidas no formato GIF (*Compuserve Graphics Interchange Format*) [2], que possui compressão e gera arquivos pequenos o suficiente para trafegar na rede sem muita demora, desde que o tamanho da imagem seja reduzido. Porém, a apresentação de mapas como imagens de pequena resolução não é satisfatória. Na resolução 4096×4096, por exemplo, a imagem no formato GIF do mapa da Cobertura Vegetal no Brasil (Figura 1) possui 438 *Kbytes* de tamanho. Este tamanho torna inviável o tráfego na Internet atual para aplicações interativas. Se as informações vetoriais originais do mapa forem preservadas, o arquivo pode ser menor e manter a mesma qualidade.

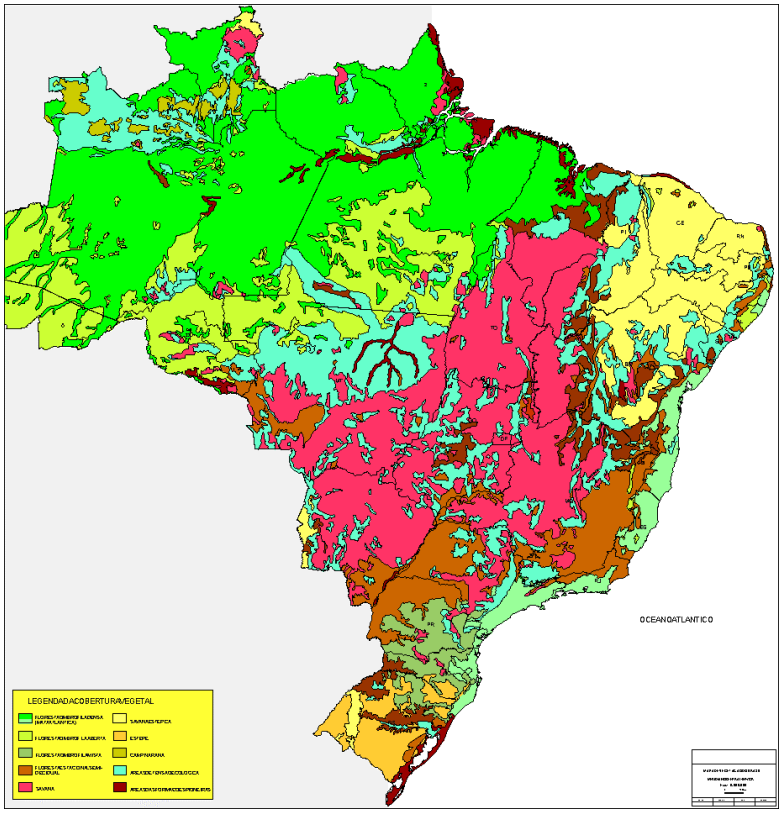


Figura 1: Mapa da Cobertura Vegetal do Brasil

Para contornar essa dificuldade, os sistemas atuais fornecem as imagens na resolução de uma parte de uma janela de um monitor, tipicamente menor que algo em torno de 640×480. Nessa resolução, as imagens de mapas ficam com tamanhos variando tipicamente entre 5 e 30 *Kbytes*. Quando o usuário deseja ver algum detalhe é necessário acessar novamente o servidor para buscar uma nova imagem com o mapa na nova escala. Alguns sistemas trabalham com determinadas escalas pré-definidas, enquanto outros regeram a imagem a partir da base de dados vetorial na escala especificada pelo usuário. Em ambos os casos, a interatividade do sistema pode ficar seriamente comprometida, pois a nova imagem deve ser reenviada através da rede, que geralmente tem baixa velocidade.

Outro problema importante decorrente da apresentação de mapas como imagens é a dificuldade da identificação de entidades. A extensão de uma entidade geográfica, como um rio ou uma estrada, é mais facilmente identificável através da seleção das primitivas gráficas que as compõem do que por áreas de imagens.

O W3C (*World Wide Web Consortium*) [3] é um consórcio fundado em outubro de 1994 com o propósito de direcionar a *web* para seu potencial completo através do desenvolvimento de protocolos comuns que promovam sua

evolução e assegurem sua interoperabilidade. Trata-se de um consórcio industrial internacional, conjuntamente dirigido por: *Massachusetts Institute of Technology, Laboratory for Computer Science* (MIT/LCS) nos Estados Unidos; *Institut National de Recherche en Informatique et Automatique* (INRIA) na Europa; e *Keio University Shonan Fujisawa Campus* (SFC) no Japão. O W3C estabeleceu alguns requisitos [4] que gráficos que podem ser escalados devem atender para que sejam utilizáveis na *web*. Os principais requisitos podem ser divididos em dois grupos: facilidades gráficas e interação. As facilidades gráficas, entre outras coisas, definem que os gráficos devem ser vetoriais e devem possuir elementos curvos, texto, seleção de fonte de texto, modo *truecolor* (sem restrição a cores indexadas), transparência, controle de fim de linha, níveis de detalhe e dados *raster*. A interação inclui *zoom*, *pan*, seleção de elementos únicos, agrupamento de elementos em estruturas semânticas, *menus* ativados por seleção, *links* para outras vistas e outras figuras no mesmo arquivo, *links* para meios externos (URL) e acesso através de meios externos.

O formato padrão ISO de armazenamento de figuras é o CGM (*Computer Graphics Metafile*) [5]. Ocorre, entretanto, que o tamanho dos arquivos das figuras armazenadas depende fundamentalmente do número de primitivas gráficas neles contidas e da precisão do sistema de coordenadas utilizado. A codificação de mapas nesse formato resulta em arquivos muito grandes, bem maiores que os respectivos GIFs ou JPEGs.

Recentemente, diversos formatos vetoriais novos foram propostos para reduzir o tamanho dos arquivos gerados. Entre eles, destacam-se o DWF (*Drawing Web Format*) [6] da *Autodesk* e o PDF (*Portable Document Format*) [7] da *Adobe*. Porém, esses formatos não foram concebidos para tratar especificamente de mapas, que possuem características especiais. Mapas são representações aproximadas da superfície terrestre que projetam cada ponto do globo terrestre em uma superfície plana. A característica marcante das figuras que representam mapas é a presença de linhas poligonais com um grande número de pontos.

O problema de reduzir o tamanho do arquivo que descreve um mapa, requer, necessariamente, uma investigação de como armazenar poligonais longas em um espaço pequeno e este é um foco importante deste trabalho. De certa maneira, esse foco está relacionado com o assunto de *generalização* da área de SIGs. Generalização é um assunto reconhecido como importante para a apresentação de mapas e consiste na criação de uma representação simplificada dos dados. Dettori e Puppo [8] procuram sintetizar e formalizar matematicamente as operações que transformam as representações de mapas em outras mais simples e eficazes para serem usadas na comunicação visual.

Na cartografia digital, as poligonais são simplificadas segundo critérios geométricos, como o de Douglas-Peucker [9], de acordo com uma tolerância geralmente definida em função da escala da carta. No processo de envio de figuras pela *web* para a apresentação de mapas, entretanto, o critério de simplificação é mais naturalmente definido em função da resolução da superfície de visualização do dispositivo utilizado (*canvas* ou janelas em impressoras). Além disso, a simplificação dos dados para a apresentação deve incluir um processo de quantização de coordenadas, visando reduzir o tamanho final do arquivo.

Este trabalho propõe uma estratégia para reduzir substancialmente o tamanho dos arquivos que representam os mapas, eliminando informações desnecessárias e redundantes, sem perda visual na figura gerada. A primeira informação desnecessária é a precisão exagerada das coordenadas dos pontos, considerando que a figura é apenas para apresentação. Na maioria dos sistemas de informação, não se transmite a figura para que, sobre seus dados, sejam feitas análises numéricas precisas. Os requisitos de precisão nas coordenadas dos gráficos de apresentação adotados neste trabalho são os seguintes:

- as coordenadas devem possibilitar a exibição da figura em uma tela de monitor convencional, com um fator de *zoom* de 400%, sem erros de posicionamento dos vértices;
- fatores de *zoom* acima de 400% podem ou acarretar aproximações no posicionamento dos vértices ou solicitar ao servidor de *web* novas coordenadas através do *plug-in* responsável pela tarefa;
- a figura deve poder ser impressa ou inserida em documentos, de modo que não haja perda de qualidade visual, diferentemente do que acontece com as imagens.

Para tanto, é proposto um processo de codificação com compressão. É importante notar que este processo não inclui a simplificação cartográfica dos dados, ou seja, supõe-se que os dados foram simplificados antes de serem submetidos ao processo em questão. Resumindo, o processo de codificação com compressão proposto possui os seguintes passos básicos:

(a) Quantização: imersão da figura no espaço cartesiano dos inteiros positivos Z_+^2, escalando-se as coordenadas dos pontos para que seu valor mínimo seja zero e seu valor máximo M seja definido pela aplicação.
(b) Simplificação: eliminação dos segmentos de tamanho zero resultantes do passo anterior.
(c) Codificação: codificação das primitivas da figura em um formato compacto.
(d) Compressão: compressão através de algoritmos clássicos de compressão sem perda.

A descompressão proposta possui apenas dois passos seqüenciais:

(a) Descompressão.
(b) Interpretação dos comandos.

DESENVOLVIMENTO

Na *quantização*, é feita a conversão das coordenadas originais do mapa para o espaço cartesiano discreto dos inteiros positivos Z_+^*. O valor máximo de cada coordenada é igual à resolução em *pixels* da imagem, que é especificada em função da necessidade da apresentação.

Nessa conversão, muitos pontos do mapa podem corresponder ao mesmo ponto no sistema inteiro. Quando uma poligonal possui pontos consecutivos iguais no sistema de coordenadas inteiras, apenas um ponto é considerado, eliminando-se segmentos de tamanho zero. Logo, a *simplificação* elimina as informações que se tornam redundantes.

A *codificação* compacta consiste em armazenar as poligonais do mapa da forma mais compacta possível. Os pontos da poligonal podem possuir coordenadas de três tipos: absolutas, relativas por incremento e relativas por direção quantizada. A estratégia geral é simples: todos os pontos são sempre armazenados no menor tamanho possível.

Nas coordenadas absolutas, um ponto é armazenado de acordo com a resolução da quantização. Um ponto absoluto é composto por um *header*, pelas coordenadas *x* e *y* e por um *footer* (Figura 2). Há mais de um tamanho de ponto com coordenadas absolutas; por isso, é necessário ter um *header* e um *footer* para cada ponto. O objetivo do *header* é indicar qual o tamanho da codificação do ponto absoluto: enorme, grande, normal ou pequeno. O *footer* indica o que há depois do ponto corrente: um ponto absoluto, um ponto relativo por incremento, uma cadeia de *bits* ou o fim da poligonal. Tanto o *header* quanto o *footer* ocupam dois *bits* cada e seus possíveis valores podem ser vistos na Tabela 1 e na Tabela 2, respectivamente.

Cada coordenada absoluta pode ocupar 10, 14, 18 ou 30 *bits*, portanto um ponto absoluto pode ter 3, 4, 5 ou 8 *bytes* (Tabela 3), respectivamente. As coordenadas dos pontos absolutos nunca possuem valores negativos, logo os valores das coordenadas de um ponto absoluto enorme, por exemplo, variam de 0 a $2^{30}-1 \approx 10^{10}$. Por isso, a maior resolução admitida nesta proposta é $2^{30} \times 2^{30}$, isto é, a maior figura criada pode ter tamanho $2^{30} \times 2^{30}$.

Os pontos com coordenadas absolutas podem aparecer isolados na poligonal, isto é, os pontos anterior e posterior ao ponto absoluto corrente podem ou não ser pontos absolutos. O primeiro ponto de cada poligonal é sempre um ponto absoluto, assim o conceito de posição corrente fica restrito a cada poligonal.

header	coordenada x	coordenada y	*footer*

Figura 2: Formato dos pontos absolutos e dos pontos relativos por incremento

Header	Tamanho
00	Enorme
01	Grande
10	Normal
11	Pequeno

Tabela 1: Valores para o *heade*

Footer	Próximo
00	Fim de poligonal
01	Ponto com coordenadas absolutas
10	Ponto com coordenadas relativas por incremento
11	Cadeia de *bits*

Tabela 2: Valores para o *footer*

Ponto absoluto	*Header* (*bits*)	Coord. x (*bits*)	Coord. y (*bits*)	*Footer* (*bits*)	**Total (*bytes*)**
Enorme	2	30	30	2	**8**
Grande	2	18	18	2	**5**
Normal	2	14	14	2	**4**
Pequeno	2	10	10	2	**3**

Tabela 3: Tamanho dos pontos absolutos

Nas coordenadas relativas por incremento, o valor das coordenadas armazenado no arquivo é igual à diferença das coordenadas do ponto atual com relação às coordenadas do ponto anterior (no sistema de coordenadas da quantização). Um ponto relativo por incremento tem a mesma composição de um ponto absoluto, isto é, possui um *header*, as coordenadas *x* e *y* e um *footer* (Figura 2). Da mesma forma que nos pontos com coordenadas absolutas, há mais de um tamanho de ponto com coordenadas relativas por incremento. As funções do *header* e do *footer* são as

mesmas dos pontos com coordenadas absolutas, sendo que o *header* indica qual o tamanho do ponto com coordenadas relativas por incremento que será analisado a seguir. O *header* e o *footer* continuam ocupando dois *bits* cada e seus possíveis valores são iguais aos dos pontos absolutos, mostrados na Tabela 1 e na Tabela 2, respectivamente. Cada coordenada relativa por incremento pode ocupar 2, 6, 10 ou 14 *bits*, logo um ponto relativo por incremento pode ter 1, 2, 3 ou 4 *bytes* (Tabela 4), respectivamente, de acordo com o tamanho do segmento. As coordenadas podem assumir valores negativos; no caso de um ponto relativo por incremento enorme, por exemplo, podem variar de -2^{13} a $2^{13}-1$, isto é, de −8192 a 8191.

Um ponto relativo por incremento também pode aparecer isolado na poligonal, ou seja, os pontos anterior e posterior ao ponto relativo por incremento corrente podem ou não ser pontos relativos por incremento. Os pontos absolutos normal e pequeno e os pontos relativos por incremento enorme e grande possuem o mesmo tamanho, respectivamente. Nesse caso, o ponto corrente sempre é considerado ponto relativo, a não ser que seja o primeiro ponto da poligonal.

Ponto relativo por incremento	*Header* (*bits*)	Coord. x (*bits*)	Coord. y (*bits*)	*Footer* (*bits*)	**Total (*bytes*)**
Enorme	2	14	14	2	**4**
Grande	2	10	10	2	**3**
Normal	2	6	6	2	**2**
Pequeno	2	2	2	2	**1**

Tabela 4: Tamanho dos pontos relativos por incremento

Uma cadeia de *pixels* é composta por um conjunto de pontos adjacentes no sentido 8-conexo na grade de $\dot{Z}_+^2$. Nesse trabalho, as cadeias de *pixels* são codificadas como cadeias de *bits* (*Freeman chain codes*) [10].

Um ponto com coordenadas relativas por direção quantizada sempre pertence a uma cadeia de *bits*. Cada cadeia de *bits* possui no mínimo duas direções e termina com um código de marcação de fim de cadeia (6 *bits*) seguido por um *footer* (2 *bits*). Como a cadeia de *bits* pode ser interna a uma poligonal, é necessária a presença de um *footer* (Tabela 1). O *header* não é necessário, porque as cadeias de *bits* só podem ter uma formação, isto é, não há diferentes formatos dessas cadeias. As direções são calculadas segundo a rosa dos ventos (Figura 3), ocupando 3 *bits* cada. Por exemplo, supondo o segmento P_iP_{i+1} (Figura 3), o ponto inicial do segmento é colocado na origem da rosa dos ventos e de acordo com a localização do ponto P_{i+1} verifica-se qual a direção correspondente. Neste caso, a direção corresponde é a nordeste. Os valores das direções são apresentados na Tabela 5.

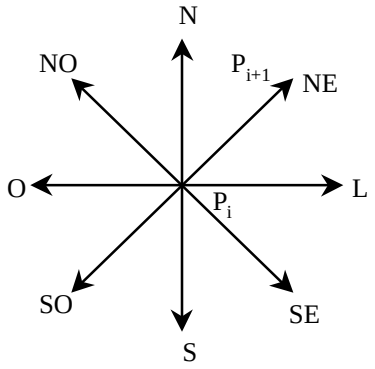


Figura 3: Direções

NO	N	NE		3	2	1		011	010	001
O	P_i	L	=	4	P_i	0	=	100	P_i	000
SO	S	SE		5	6	7		101	110	111

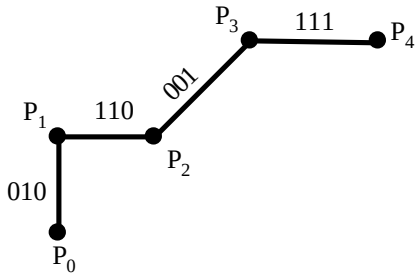
Tabela 5: Valores das direções na cadeia de *bits*

A primeira direção de cada cadeia de *bits* é armazenada segundo a rosa dos ventos. Da segunda direção em diante, é feita a relação da direção atual com a sua anterior. Essa relação é realizada verificando-se quantas direções têm que ser percorridas da direção anterior para chegar à direção atual, seguindo-se sempre o sentido anti-horário. Por exemplo, supondo que a direção anterior é norte e a atual é leste, o número de direções percorridas da direção norte até a direção leste é igual a 6, logo a direção atual relativa à anterior vale 6. O objetivo dessa codificação relativa é aumentar a ocorrência das direções leste em detrimento das oeste. Com isso, espera-se que o algoritmo de compressão

da fase posterior possa ser mais eficiente e cria-se um código improvável (oeste) que pode ser utilizado para codificar o fim da cadeia.

A marcação de fim da cadeia de *bits* é composta por duas direções oeste seguidas. Uma direção oeste relativa corresponde a uma volta de 180°. Por exemplo, um segmento é composto pelos pontos P_1 e P_2 e o próximo segmento é P_2P_1. A direção relativa do segmento P_2P_1 em relação ao seu anterior é oeste. Na codificação proposta neste trabalho, duas direções oeste não podem ocorrer no meio de uma cadeia.

Um exemplo de uma cadeia de *pixels* codificada como uma cadeia de *bits* pode ser visto na Figura 4. A primeira direção da cadeia de *bits* não pode ser relativa, pois não há uma direção anterior. Desse modo, a primeira direção da cadeia é a direção norte (P_0P_1). As direções seguintes são sempre relativas às suas anteriores. A direção P_1P_2 é leste segundo a rosa dos ventos, porém esta deve ser relativa à direção de P_0P_1. Desse modo, da direção norte até a direção leste são percorridas 6 direções, logo a segunda direção da cadeia de *bits* é 110. O procedimento é o mesmo até a última direção (P_3P_4). Depois da última direção (111), acrescenta-se à cadeia a marcação de fim, composta pelas duas direções oeste seguidas (direções sublinhadas). No fim da cadeia, é colocado o *footer* indicando o que está a seguir.



010, 110, 001, 111, <u>100</u>, <u>100</u>, *footer*

Figura 4: Exemplo de cadeia de *bits*

O algoritmo de Bresenham [11] também trabalha com direções e poderia ter sido utilizado na codificação de todos os segmentos de uma poligonal. Assim, a codificação de um segmento seria feita com 1 ou mais códigos de direção. Quando a distância entre pontos fosse 1, o segmento seria uma direção, quando a distância fosse maior, o segmento seria *rasterizado*, gerando uma cadeia de direções.

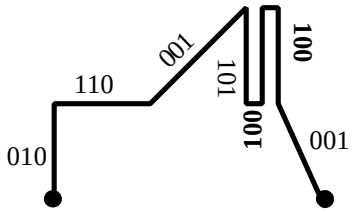
Como cada direção ocupa 3 *bits*, uma linha horizontal com tamanho 10, por exemplo, seria codificada com 30 *bits* (≈ 4 *bytes*), utilizando-se o algoritmo de Bresenham. Isto sem considerar nenhuma marcação de início ou de fim. Essa linha é muito melhor armazenada em um ponto relativo por incremento normal (2 *bytes*). Desse modo, descartou-se nesse trabalho a idéia de *rasterizar* segmentos.

A cadeia de *pixels* não possui um tamanho fixo, pois não é possível saber quantas direções esta vai possuir. Forma-se uma cadeia de *bits* a partir de uma cadeia de *pixels* sempre que o valor absoluto da diferença entre as coordenadas quantizadas *x* e *y* de dois pontos consecutivos em relação aos seus anteriores for menor ou igual a 1. Como cada direção da cadeia ocupa três *bits*, uma cadeia de *pixels* é criada com no mínimo duas direções (total de 2 *bytes*), pois caso contrário é mais econômico guardar as coordenadas como um ponto relativo por incremento pequeno (1 *byte*).

Os casos em que no meio de uma cadeia de *bits* ocorrem direções oeste consecutivas podem ser confundidos com a marcação de fim de cadeia e precisam ser eliminados. Esta eliminação, além de diminuir o tamanho do arquivo, não causa perda de informação, desde que certos cuidados sejam tomados. Esses cuidados são diferentes para números pares e ímpares de direções oeste internas consecutivas.

Um número par de direções oeste consecutivas não causa nenhum problema, pois um número 2*n* de direções oeste indica que o mesmo segmento da poligonal foi percorrido 2*n*+1 vezes seguidas (1 direção qualquer mais 2*n* direções oeste), mas terminou no mesmo ponto. Nesse caso, nenhuma direção oeste é considerada. Apenas a primeira direção, que é diferente de oeste, é considerada. Com isso, 2*n* pontos relativos são descartados, o arquivo criado fica menor e sem informações repetidas. A Figura 5 ilustra essa situação, onde a direção qualquer é a sul (101) e ocorrem 2 direções oeste (100) que são eliminadas. Nessa figura, as direções sublinhadas indicam a marcação de fim de cadeia e as direções em negrito são as direções oeste no meio da cadeia.

Quando ocorre um número ímpar (2*n*+1) de direções oeste, apenas uma direção oeste é considerada. Dessa forma, o arquivo contém apenas a primeira direção, que pode ser qualquer uma diferente de oeste, e uma direção oeste. Como ocorre na primeira situação, o arquivo criado fica menor, pois 2*n* pontos relativos são descartados (Figura 6).

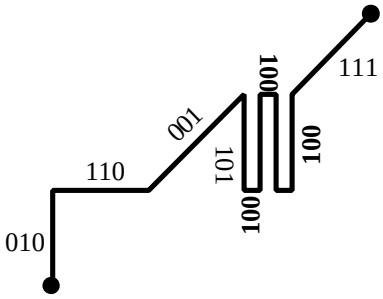


010, 110, 001, 101, **100**, **100**, 001, 100, 100 ☒

↓

010, 110, 001, 101, 001, 100, 100 ☑

Figura 5: Número par de direções oeste consecutivas no meio da cadeia de *bits*



010, 110, 001, 101, **100**, **100**, **100**, 111, 100, 100 ☒

↓

010, 110, 001, 101, **100**, 111, 100, 100 ☑

Figura 6: Número ímpar de direções oeste consecutivas no meio da cadeia de *bits*

Um número ímpar de direções oeste consecutivas causa um problema, quando a direção oeste considerada é a última direção de uma cadeia de *bits*, porque no final da cadeia ficam três direções oeste consecutivas: a considerada devido à seqüência ímpar de direções oeste e duas que marcam o fim de cadeia. O problema ocorre no momento de interpretação do arquivo, pois duas cadeias oeste indicam fim de cadeia. As duas primeiras direções serão interpretadas como fim de cadeia e a partir da terceira direção oeste, a leitura será errada, comprometendo a interpretação. Quatro opções para resolução deste problema foram analisadas (Figura 7):

(a) terminar a cadeia e substituir a última direção (oeste) por um ponto relativo por incremento pequeno;
(b) eliminar da cadeia a última direção diferente de oeste e as últimas direções oeste, perdendo o último ponto;
(c) aumentar o tamanho de cada direção para 4 *bits*, de modo a criar outra codificação de fim de cadeia;
(d) colocar no início de cada cadeia de *bits* o número de direções que esta possui.

A segunda opção eliminaria um segmento da poligonal original. Mesmo sendo um segmento pequeno, isto poderia ocorrer várias vezes em um único mapa, o que causaria a perda de algumas características originais. Por isso, este procedimento não foi considerado uma boa solução.

A terceira opção aumentaria o tamanho de todas as direções, logo também aumentaria bastante o tamanho final do arquivo, o que não é objetivo dessa proposta.

Sendo assim, das opções apresentadas, restam a primeira e a quarta a serem analisadas. Nos testes que serão apresentados adiante, foram realizadas algumas estatísticas. Em relação à primeira opção, uma estatística obtida foi o maior número de direções oeste que seriam substituídas por um ponto relativo por incremento pequeno. Sobre a quarta opção, obteve-se o tamanho da maior cadeia de *pixels*, ou seja, o maior número de direções que uma cadeia de *pixels* apresentou, considerando-se todas as cadeias de todos os arquivos utilizados nos testes, e o maior número de cadeias que um arquivo possuiu. Com estas estatísticas, foi possível observar qual das opções apresenta melhor resultado para os casos testes estudados.

Analisando a quarta opção, o maior número direções de uma cadeia de *bits*, sem considerar as direções que marcam o fim da cadeia, foi igual a 1063. Assim, seriam necessários no mínimo 11 *bits* no início de cada cadeia para guardar o número de direções que essa possui. No pior caso, um arquivo chegou a ter 78083 cadeias de *pixels*. Desse modo, seriam gastos 11 *bits* × 78083 cadeias = 858913 *bits* ≈ 107364 *bytes* ≈ 104 *Kbytes* a mais no arquivo final para colocar o número de direções no início de cada cadeia. Com a inserção do número de direções no início de cada cadeia

de *bits*, não seria mais necessário colocar as duas direções oeste que marcam o fim da mesma, ou seja, seriam diminuídos 6 *bits* × 78083 cadeias = 468498 *bits* ≈ 58562 *bytes* ≈ 57 *Kbytes*. Finalizando, o arquivo criado teria 104 *Kbytes* – 57 *Kbytes* = 47 *Kbytes* a mais, se a quarta opção fosse utilizada.

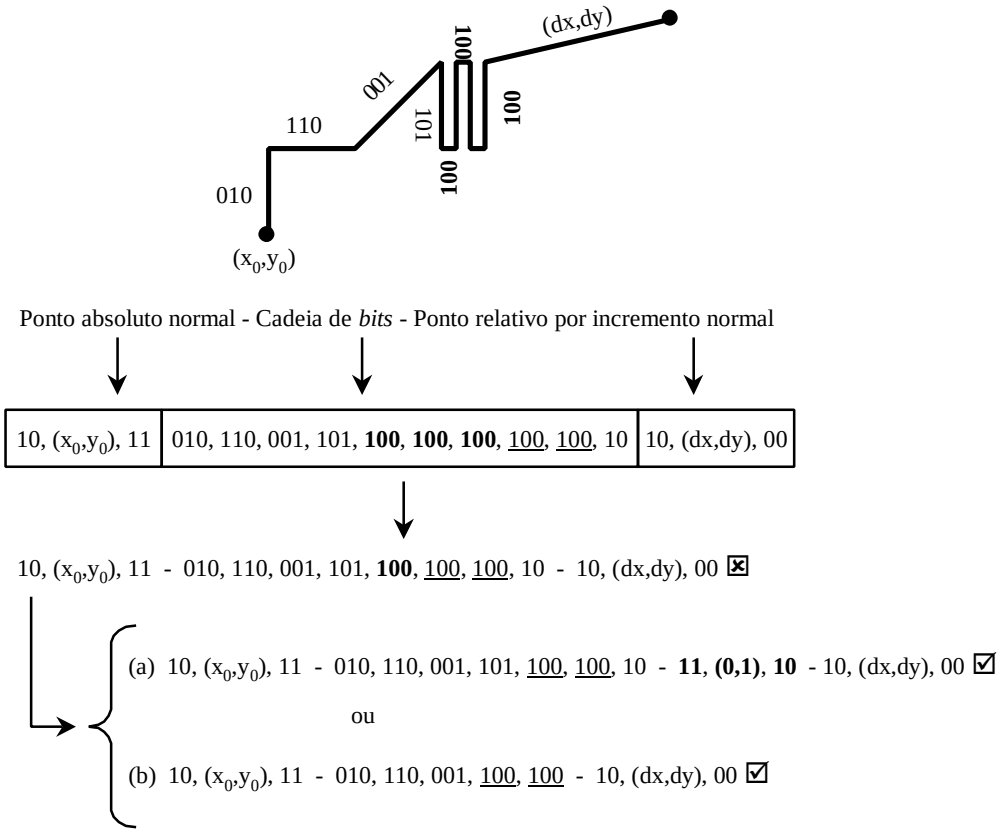


Figura 7: Soluções para um número ímpar de direções oeste consecutivas no final da cadeia de *bits*

Na primeira opção, o maior número de direções oeste convertidas em pontos relativos por incremento pequenos, que ocupam 1 *byte*, foi igual a 792. Logo, no pior caso da primeira opção, seriam utilizados 792 *bytes* a mais no arquivo criado, enquanto a quarta opção gastaria mais 47 *Kbytes* no arquivo final. Desse modo, concluiu-se que a melhor solução é a primeira. Uma análise mais detalhada da quarta opção mostra que mesmo utilizando uma codificação adaptativa para o número de pontos da poligonal, dificilmente essa geraria melhores resultados que a primeira opção.

Sendo assim, quando ocorre um número ímpar de direções oeste consecutivas no final de uma cadeia de *bits*, a cadeia termina na última direção diferente de oeste e uma direção oeste é colocada logo após a cadeia utilizando um ponto relativo por incremento pequeno.

A etapa de *compressão* consiste em aplicar o algoritmo LZO1X da biblioteca LZO (*Lempel-Ziv-Oberhumer*) [12] no arquivo gerado pela codificação compacta. A LZO é uma biblioteca de compressão e descompressão de dados em tempo real. Segundo os testes realizados, ela apresenta resultados tão bons quanto o algoritmo LZW, utilizado pelo formato GIF, mesmo porque ambos se baseiam na modelagem do dicionário. O LZO1X é o algoritmo de uso geral da LZO, isto é, o algoritmo que apresenta resultados melhores na maioria dos casos.

A etapa de descompressão é a primeira etapa executada na interpretação de um arquivo TWF. A segunda e última etapa é a decodificação. A descompressão é feita utilizando-se a função de descompressão relativa ao algoritmo LZO1X_1 da biblioteca LZO. Para decodificar um arquivo, basta seguir os passos da codificação de modo inverso, ou seja, os últimos passos da codificação são os primeiros da decodificação e assim por diante. A decodificação é a etapa inversa da codificação.

RESULTADOS

Nos testes, foram utilizados arquivos origem de dois tipos: BIN e CGM. O formato BIN é binário e cada poligonal é composta pelo número de pontos (*int*) que possui, seguido pelos pontos em coordenadas reais (*float*). O formato CGM utilizado é binário e com coordenadas inteiras de 32 *bits*. Os arquivos utilizados nos testes foram:

- Curvas de Nível de Madison [14] (Figura 8).
- Mapa de Municípios do Brasil [15] (Figura 9).
- Mapa da Cobertura Vegetal do Brasil (Figura 1).

O mapa original da Cobertura Vegetal do Brasil está no formato CGM, enquanto que os demais estão no formato BIN. Todos os arquivos BIN foram convertidos para CGM utilizando-se a biblioteca CD, porém o inverso não ocorreu, porque o formato BIN não é muito utilizado na transmissão de mapas. O mapa das Curvas de Nível de Madison é muito denso, por isso na Figura 8 é difícil visualizar com detalhes as curvas de nível.

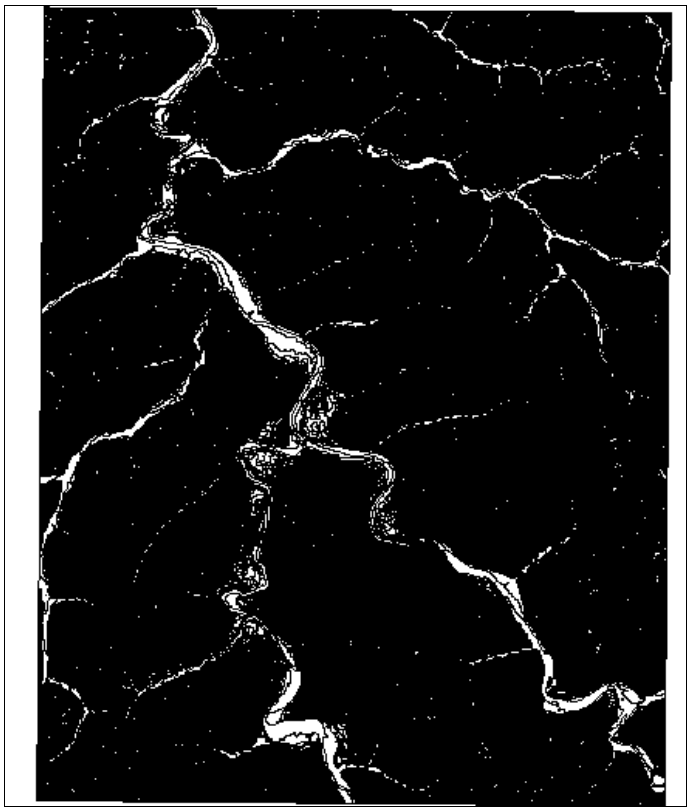


Figura 8: Curvas de Nível de Madison

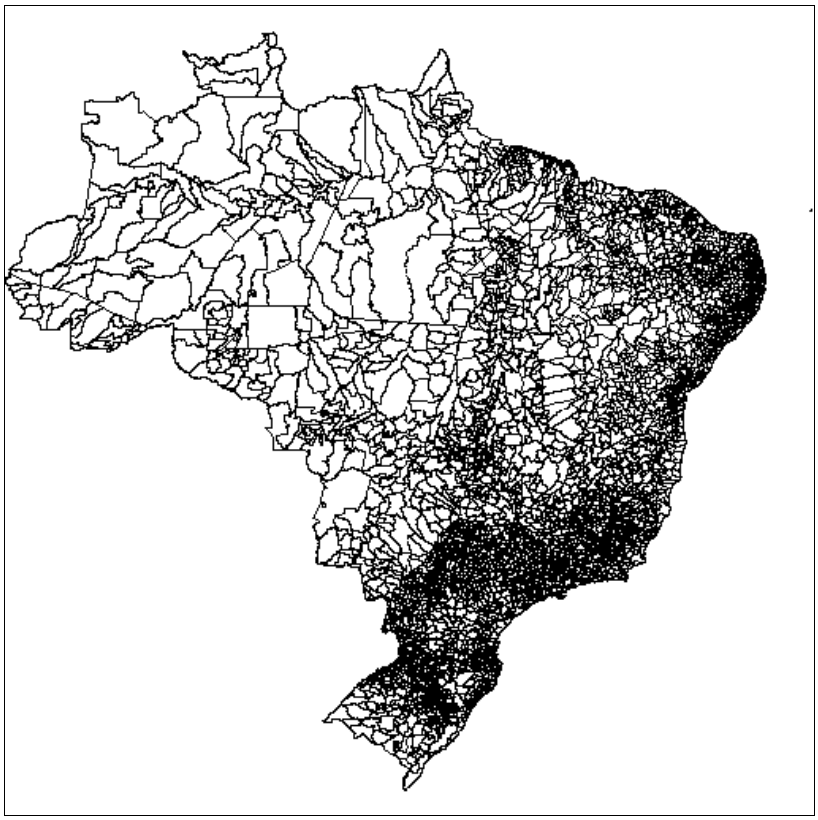


Figura 9: Mapa de Municípios do Brasil

Na Tabela 6, são apresentados alguns dados sobre esses arquivos, como o formato original, o tamanho, o número de pontos, o número de poligonais e a média de pontos por poligonal (número de pontos/número de poligonais). Estes dados são parâmetros para avaliação dos resultados que serão apresentados a seguir. Na escolha dos casos testes, houve a preocupação em trabalhar com arquivos de características diferentes, principalmente a média de pontos por poligonal.

Arquivo	Formato	Tamanho (Kb)	Número de pontos	Número de poligonais	Média de pontos por poligonal
Madison	BIN	5212	665674	2840	234
BR Municípios	BIN	5666	716933	16586	43
BR Vegetação	CGM	830	196977	2781	71

Tabela 6: Dados sobre os arquivos originais

A partir dos arquivos originais, foram geradas imagens e figuras nas resoluções 512×512, 1024×1024, 2048×2048 e 4096×4096 em diversos formatos: CDB (*Canvas Draw Binary Format* – uma implementação do esquema descrito acima) [13], CGM, DWF e GIF. Os formatos que não possuem compressão foram comprimidos com o algoritmo LZO1X. Os arquivos nos formatos GIF e CGM foram criados através das bibliotecas CD (*Canvas Draw*) Versão 3.6 [17] e IM – Biblioteca de Acesso a Arquivos de Imagens *Bitmaps* – Versão 2.2 [18], ambas desenvolvidas pelo TeCGraf – Grupo de Tecnologia em Computação Gráfica [16]. Os arquivos no formato DWF foram criados a partir dos respectivos DXF utilizando-se o *AutoCAD Release 14.0*. Os arquivos DXF também foram gerados utilizando a biblioteca CD.

São apresentados os resultados obtidos nas etapas de quantização, codificação e compressão. A quantização é comum a todos os formatos analisados, ou seja, para nivelar a comparação, todos os conversores utilizarão as mesmas figuras quantizadas. A compressão utilizando a biblioteca LZO também visa uniformizar a comparação e só é aplicada nos formatos que não realizam compressão.

Na quantização, os pontos dos arquivos originais são convertidos para um sistema inteiro, cujos valores variam de zero até a resolução do arquivo a ser gerado. Como já foi visto, quanto menor a resolução mais pontos irão coincidir e serão eliminados do arquivo resultante. O resultado da quantização consiste no número de pontos que foram realmente codificados em comparação com o número de pontos originais dos arquivos (Tabela 7 e Gráfico 1).

Arquivo	Núm. pontos originais	512x512		1024x1024		2048x2048		4096x4096	
		Núm. Pontos	% original	Núm. pontos	% original	Núm. pontos	% original	Núm. pontos	% original
Madison	665674	402919	61	551159	83	632607	95	658471	99
BR Municípios	716933	78840	11	133756	19	231710	32	383925	54
BR Vegetação	196977	93571	48	147996	75	178032	90	188622	96

Tabela 7: Quantização

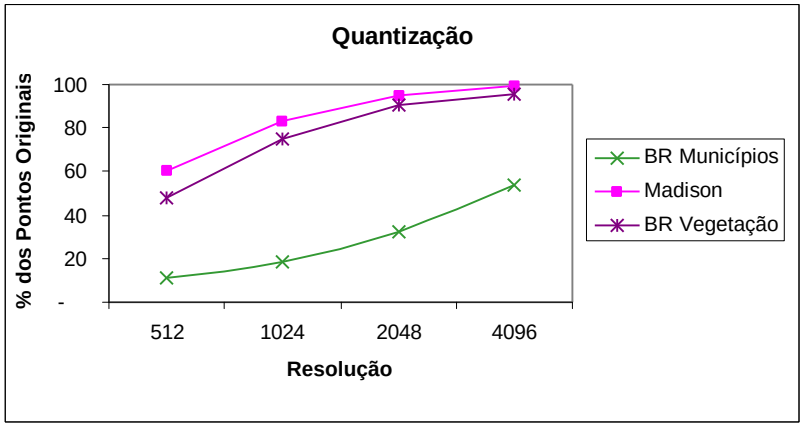


Gráfico 1: Quantização

Na Tabela 8, no Gráfico 2, no Gráfico 3 e no Gráfico 4 são apresentadas as comparações entre os tamanhos dos arquivos em vários formatos e em várias resoluções. Como citado anteriormente, os formatos que não possuem compressão foram comprimidos com o algoritmo LZOX_1 da biblioteca LZO. Os tamanhos finais dos arquivos depois de passarem pela compressão são comparados com o tamanho do arquivo GIF correspondente, isto é, verifica-se a qual porcentagem do arquivo GIF corresponde o tamanho do arquivo no formato que está sendo analisado.

	4096 x 4096				2048 x 2048				1024 x 1024				512 x 512			
Arquivo	CDB	CGM	DWF	GIF	CDB	CGM	DWF	GIF	CDB	CGM	DWF	GIF	CDB	CGM	DWF	GIF
Madison (Kb)	1233	2645	1221	1512	1037	2645	1013	486	662	2645	693	71	288	2645	432	8
Comp. LZO1X_1 (Kb)	1080	2476			826	2243			541	1924			263	1547		
(%) GIF	71	164	81	100	170	462	208	100	762	2710	976	100	3288	19338	5400	100
BR Municípios (Kb)	335	3055	521	337	225	3055	356	131	164	3055	237	48	138	3055	156	16
Comp. LZO1X_1 (Kb)	294	1612			182	1148			123	761			91	501		
(%) GIF	87	478	155	100	139	876	272	100	256	1585	494	100	569	3131	975	100
BR Vegetação (Kb)	378	839	311	438	300	839	260	172	141	839	177	67	84	839	112	24
Comp. LZO1X_1 (Kb)	277	634			208	569			111	480			63	373		
(%) GIF	63	145	71	100	121	331	151	100	166	716	264	100	263	1554	467	100

Tabela 8: Comparação dos formatos

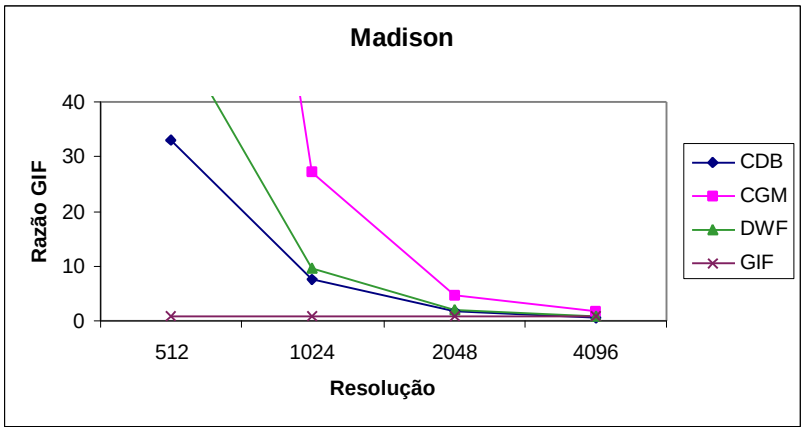


Gráfico 2: Madison - Razão dos formatos CDB, CGM e DWF em relação ao GIF

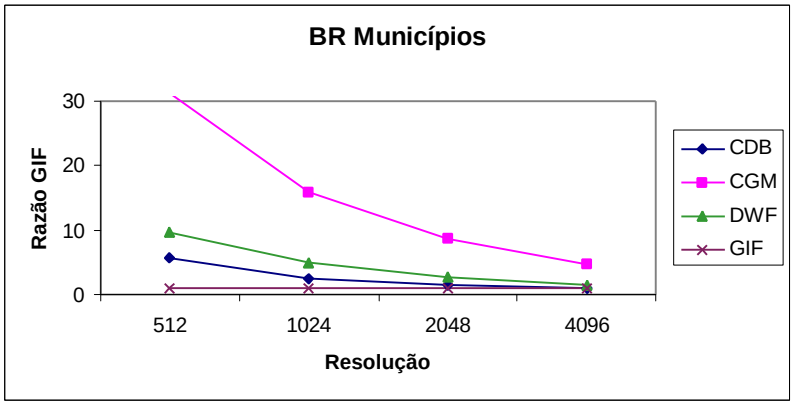


Gráfico 3: BR Municípios - Razão dos formatos CDB, CGM e DWF em relação ao GIF

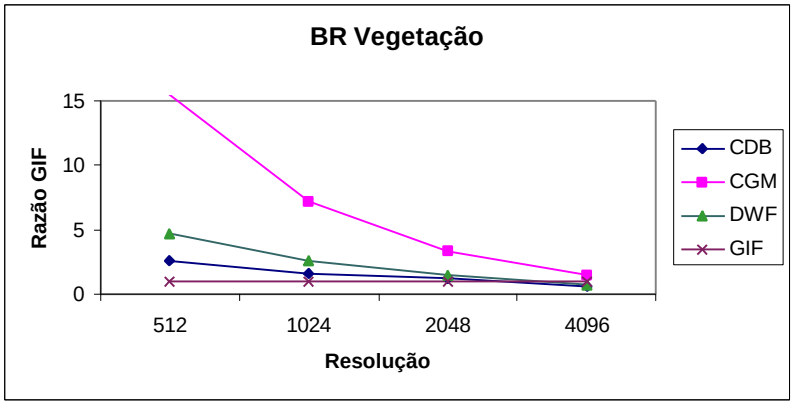


Gráfico 4: BR Vegetação - Razão dos formatos CDB, CGM e DWF em relação ao GIF

Como pôde ser observado, o CDB apresentou resultados melhores ou tão bons quanto os outros formatos. Quanto maior a resolução, melhor o resultado no CDB em relação aos arquivos GIF.

A seguir, são apresentadas análises da quantidade de cada tipo de ponto, para cada arquivo de origem, em cada uma das resoluções (Tabela 9, Tabela 10 e Tabela 11). Nestas análises também podem ser revistos os resultados da etapa de quantização: o número original de pontos de cada arquivo e o número de pontos gerados para cada resolução. São apresentados o número de pontos e de poligonais originais de cada arquivo e o número de pontos armazenados para cada arquivo em cada uma das resoluções, este último correspondente à coluna Pontos. As colunas AP, AN, AG e AE correspondem aos tipos de ponto absoluto pequeno, normal, grande e enorme, respectivamente. As colunas RP, RN, RG e RE correspondem aos tipos de ponto relativo por incremento pequeno, normal, grande e enorme, respectivamente. A coluna RDr indica o número de pontos relativos por direção quantizada e a DrO indica o número de pontos que foram eliminados, isto é, que não foram gravados porque correspondiam a direções oeste consecutivas em uma cadeia de *bits*.

Madison	Núm. Original de Pontos = 665674					Núm. Original de Poligonais = 2840					
Resolução	Pontos	AP	AN	AG	AE	RP	RN	RG	RE	RDr	DrO
4096x4096	658471	225	2615	0	0	46796	586493	596	0	21744	2
2048x2048	632607	699	2141	0	0	90770	421643	11	0	117323	20
1024x1024	551159	2840	0	0	0	69418	187949	0	0	290860	92
512x512	402919	2840	0	0	0	19806	42417	0	0	337680	176

Tabela 9: Madison - Tipos de pontos CDB

BR Municípios	Núm. Original de Poligonais = 16596					Núm. Original de Pontos = 716933					
Resolução	Pontos	AP	AN	AG	AE	RP	RN	RG	RE	RDr	DrO
4096x4096	383925	0	16596	0	0	11637	28069	221	0	322526	4876
2048x2048	231710	1263	15333	0	0	3727	8045	33	0	194567	8742
1024x1024	133756	16596	0	0	0	3209	3772	2	0	100549	9628
512x512	78840	16596	0	0	0	4638	1569	0	0	48365	7672

Tabela 10: BR Municípios - Tipos de pontos CDB

BR Vegetação	Núm. Original de Pontos = 196977					Núm. Original de Poligonais = 2781					
Resolução	Pontos	AP	AN	AG	AE	RP	RN	RG	RE	RDr	DrO
4096x4096	188622	26	2755	0	0	12552	159425	1427	19	12384	34
2048x2048	178032	225	2556	0	0	27848	98977	672	11	47655	88
1024x1024	147996	2781	0	0	0	7243	17515	226	9	120066	156
512x512	93571	2781	0	0	0	1610	3855	73	0	84906	346

Tabela 11: BR Vegetação - Tipos de pontos CDB

Como foi visto, a quantização diminui consideravelmente o número de pontos e grande parte dos pontos foram armazenados como pontos relativos por direção quantizada, o que indica uma melhoria em relação aos tipos de armazenamentos nos formatos vetoriais existentes. Para exemplificar, os dados da Tabela 10 também podem ser visualizados graficamente no Gráfico 5, no Gráfico 6 e no Gráfico 7.

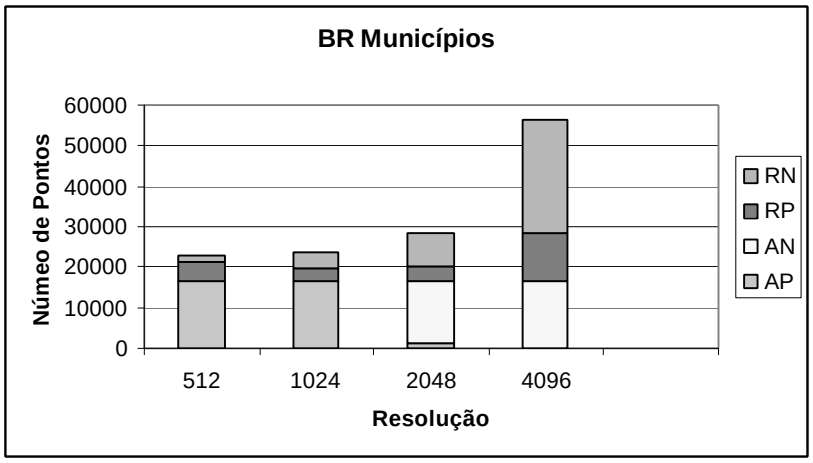


Gráfico 5: CDB - Pontos absolutos e relativos por incremento

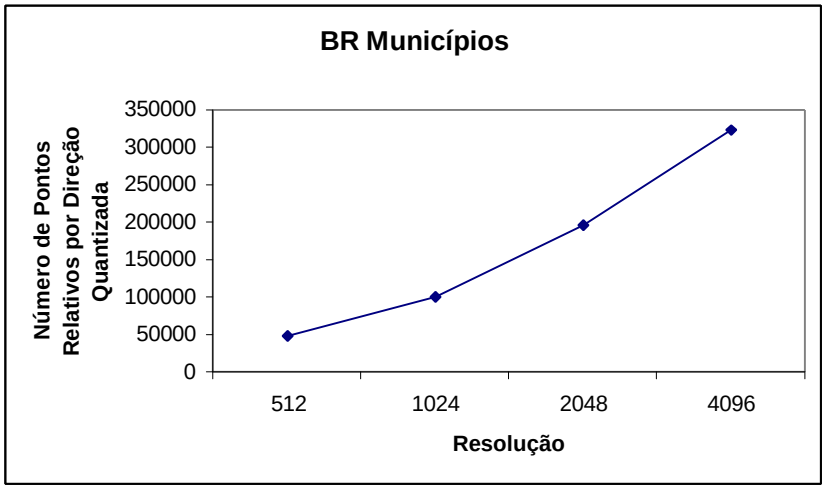


Gráfico 6: CDB - Pontos relativos por direção quantizada

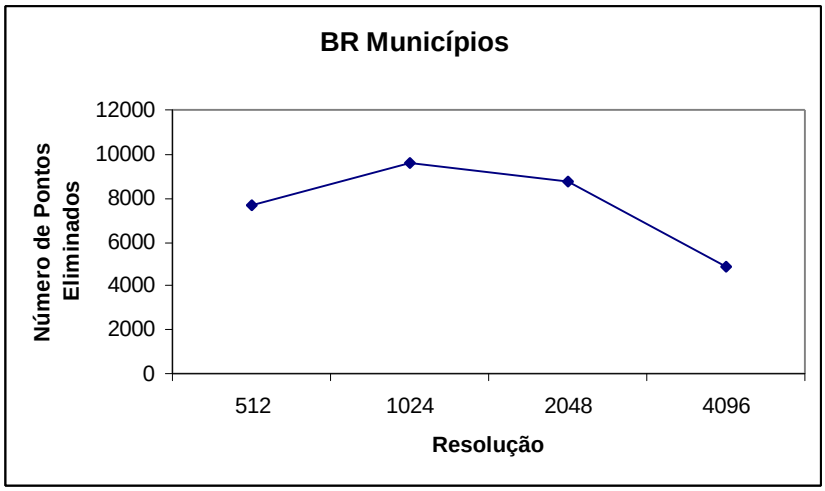


Gráfico 7: CDB – Pontos eliminados (direções oeste consecutivas)

O Gráfico 5mostra que com o aumento da resolução a quantidade de pontos absolutos pequenos no diminui e a quantidade de pontos absolutos normais aumenta, porém o número de pontos absolutos se mantém aparentemente constante. Isto ocorre porque o número de pontos absolutos deve ser no mínimo igual ao número de poligonais do arquivo. O número de pontos relativos por incremento também se mantém aparentemente constante, sendo que com o aumento da resolução aumenta o número de pontos relativos por incremento normais e diminui o número de pontos relativos por incremento pequenos.

No Mapa de Municípios do Brasil, as poligonais possuem muitos segmentos, apresentam muitos detalhes. Assim, quanto maior a resolução mais esses detalhes aparecem, favorecendo a codificação com pontos relativos por direção quantizada (Gráfico 6). O Gráfico 7 mostra que o número de pontos relativos por direção quantizada eliminados não possui uma correspondência direta com o aumento da resolução.

A seguir é apresentada uma avaliação do número médio de *bits* ocupado por cada ponto nos arquivos (Tabela 12 e Gráfico 8). Esta média é feita dividindo-se o tamanho ocupado por todos os pontos, sem compressão, pelo número total de pontos. O valor obtido é aproximadamente o número de *bits* por ponto.

	Madison				BR Municípios				Br Vegetação			
	4096 x 4096	2048 x 2048	1024 x 1024	512 x 512	4096 x 4096	2048 x 2048	1024 x 1024	512 x 512	4096 x 4096	2048 x 2048	1024 x 1024	512 x 512
Pontos (Kb)	1214	979	581	251	263	162	102	79	344	255	99	50
Núm. Pontos	658741	632607	551159	402919	383925	231710	133756	78840	188622	178032	147996	93571
Bits / Ponto	15.1	12.7	8.6	5.1	5.6	5.7	6.2	8.2	14.9	11.7	5.5	4.4

Tabela 12: CDB - Número médio de *bits* por ponto

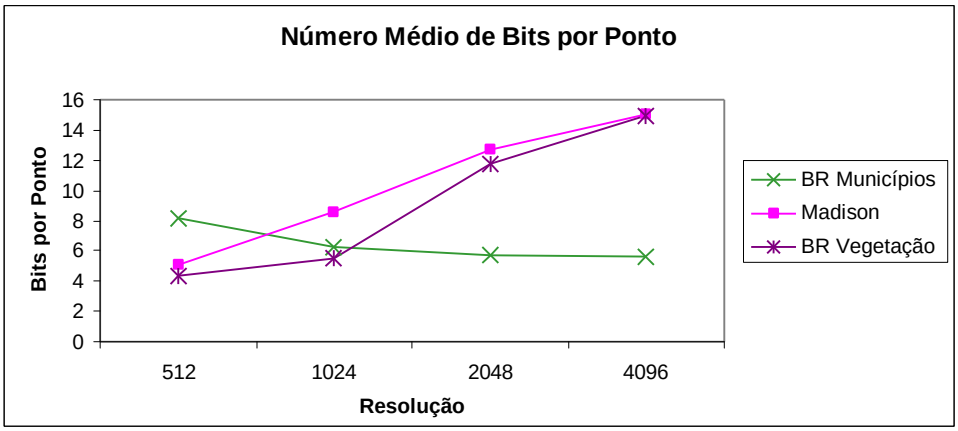


Gráfico 8: CDB - Número médio de *bits* por ponto

O arquivo BR Municípios apresentou uma diminuição no número médio de *bits* por ponto com o aumento da resolução (Gráfico 8), porque houve um maior número de pontos relativos por direção quantizada com o aumento da resolução (Gráfico 6). Para os demais arquivos, o número médio de *bits* por ponto aumenta com a resolução, sendo que o maior ponto armazenado ocupa menos de 2 *bytes*.

Para exemplificar, a Tabela 13 mostra o resultado em *Kbytes* de um estudo dos diversos algoritmos de compressão da biblioteca LZO. O Gráfico 9 mostra os mesmos resultados em termos de percentuais de compressão. Estes resultados indicam que não há um algoritmo que seja ótimo em todos os casos. Outra observação a ser feita é que os arquivos não sofrem uma grande compressão, o que indica que a codificação foi bem sucedida, conseguindo eliminar muitas redundâncias. Na tabela e no gráfico seguinte, 1A, 1B_1, 1B_5, 1B_9, 1C_1, 1C_5, 1C_9, 1F_1, 1X_1, 1X_1_1 e 1Y_1 correspondem aos algoritmos LZO1A, LZO1B_1, LZO1B_5, LZO1B_9, LZO1C_1, LZO1C_5, LZO1C_9, LZO1F_1, LZO1X_1, LZO1X_1_1 e LZO1Y_1, respectivamente.

	BR Municípios										
	1A	1B_1	1B_5	1B_9	1C_1	1C_5	1C_9	1F_1	1X_1	1X_1_11	1Y_1
4096 x 4096	294	292	288	286	292	287	285	294	294	301	300
2048 x 2048	184	183	180	179	183	180	178	183	182	187	186
1024 x 1024	129	128	126	126	128	126	125	124	123	127	126
512 x 512	101	100	99	98	100	98	97	93	91	94	93

Tabela 13: CDB – Compressão por vários algoritmos

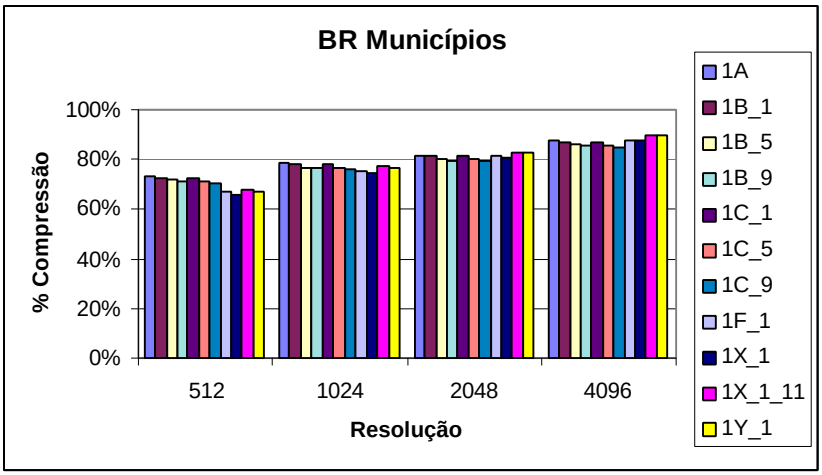


Gráfico 9: CDB – Compressão por vários algoritmos

CONCLUSÃO

Os formatos CDB, DWF e GIF apresentaram os melhores resultados. A proposta do CDB apresentou resultados melhores que os outros formatos vetoriais. Quanto maior a resolução, melhor o resultado do CDB em relação aos arquivos GIF.

O número médio de *bits* ocupado por cada ponto nos arquivos CDB, que é obtido dividindo-se o tamanho ocupado por todos os pontos (sem compressão) pelo número total de pontos gerados, foi sempre menor que 16. A compressão média obtida com a aplicação da biblioteca LZO nos arquivos CDB foi de 20%, evidenciando que a codificação das coordenadas é a responsável pela remoção da maior parte da redundância contida nos dados.

O foco deste trabalho é a codificação compacta de poligonais longas. Há, no entanto, outras questões que devem ser observadas, como: inclusão de dados *raster*; inclusão de *hyperlinks* com agrupamento de elementos em estruturas semânticas; textura e sobreposição com o plano alfa; criação de camadas (*layers*), controle de estilo de junção e fim de linha; e exibição progressiva.

Para o futuro planejamos realizar um estudo mais extenso sobre a compressão, para verificar se é possível obter um algoritmo melhor, talvez adaptativo aos dados de entrada. Além disso, a etapa de quantização deverá aumentar e incluir a simplificação cartográfica dos dados, a fim de tornar o formato mais genérico, eliminando o requisito que estabelece a simplificação prévia dos dados.

AGRADECIMENTOS

Agradecemos ao professor Gilberto Câmara Neto (INPE) pela contribuição na construção deste trabalho; ao CNPq pelo auxílio financeiro; à PETROBRAS e ao projeto FINEP/RECOPE/Modelagem, Imagem e Visualização pelo apoio ao TeCGraf/PUC-Rio. Esse trabalho é parte da dissertação de mestrado da primeira autora na PUC-Rio [13].

REFERÊNCIAS

1. CÂMARA, G., CASANOVA, M. A., HEMERLY, A. S., MAGALHÃES, G. C., MEDEIROS, C. M. B., Anatomia de Sistemas de Informação Geográfica, 10ª Escola de Computação, Campinas, 1996.
2. MURRAY, J. D., VANRYPER, W. *Encyclopedia of Graphics File Formats*. O'Reilly & Associates, Inc., 1994.
3. W3C: http://www.w3.org/
4. *W3C Scalable Graphics Requirements*: http://www.w3.org/Graphics/ScalableReq
5. *Information Technology - Computer Graphics - Metafile for the Storage and Transfer of Picture Description Information.* Part 1 - Functional Specification (ISO8632-1); Part 2 - Character Encoding (ISO8632-2); Part 3 - Binary Encoding (ISO8632-3); Part 4 - Clear Text Encoding (ISO8632-4), 1992.
6. *WHIP! and DWF - A Primer*: http://www.autodesk.com/products/whip/primer.htm
7. *Web Vector Format Comparison*: http://www.tailormade.com/WebFormatCompare.htm
8. PUPPO, E., DETTORI, G., *Towards a Formal Model for Multiresolution Spatial Maps*, in 4th International Simposium on Large Spatial Databases, pp. 152-169, 1995.
9. DOUGLAS, D. H., PEUCKER, T. K. Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature. *The Canadian Cartographer*, 1973, Volume 10, 2, pp. 112-122.
10. GONZALEZ, R. C., WOODS, R. E., *Digital Image Processing*, Addison-Wesley Publishing Company, 1992.
11. FOLEY, J. D., DAM, A., FEINER, S. K., HUGHES, J. F., *Computer Graphics: Principles and Practice*, Addison-Wesley Publishing Company, 1996.
12. OBERHUMER, M. F. X. J.: LZO: http://wildsau.idv.uni-linz.ac.at/mfx/lzo.html
13. FERREIRA, C. C. F., GATTASS, M., FIGUEIREDO, L. H. *Um Estudo sobre Arquivos Vetoriais para Visualização de Mapas na* Web. Dissertação de Mestrado, Departamento de Informática, PUC-Rio, 1998.
14. Curvas de Nível: ftp://spectrum.xerox.com/ds9/map/dlg/1:24,000/optional/.
15. Malha Municipal Digital do Brasil - Situação em 1991 e 1994, IBGE (Instituto Brasileiro de Geografia e Estatística), Base de dados em CD, ISBN: 85-240-0591-2.
16. TeCGraf: http://www.tecgraf.puc-rio.br/
17. CD – Canvas Draw – Uma Biblioteca Gráfica 2D: http://www.tecgraf.puc-rio.br/manuais/cd/
18. IM – Biblioteca de Acesso a Arquivos de Imagens *Bitmaps*: http://www.tecgraf.puc-rio.br/manuais/im/