Marcas d'Água Visuais em Mapas Vetoriais

Douglas Aurélio Marques

Trabalho Final de Mestrado Profissional

Instituto de Computação
Universidade Estadual de Campinas

Marcas d'Água Visuais em Mapas Vetoriais

Douglas Aurélio Marques

Agosto de 2005

Banca Examinadora:

- Prof. Dr. Ricardo Dahab (Orientador)
  Instituto de Computação – UNICAMP
- Prof. Dr. Léo Pini Magalhães
  Faculdade de Engenharia Elétrica e de Computação - UNICAMP
- Prof. Dr. Neucimar J. Leite
  Instituto de Computação - UNICAMP
- Prof. Dr. Roberto de Alencar Lotufo
  Faculdade de Engenharia Elétrica e de Computação - UNICAMP
- Prof. Dr. Siome Klein Goldstein
  Instituto de Computação - UNICAMP

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP

Bibliotecário: Maria Júlia Milani Rodrigues – CRB8a / 2116

Marques, Douglas Aurélio

M348m Marcas d´água visuais em mapas vetoriais / Douglas Aurélio Marques -- Campinas, [S.P. :s.n.], 2005.

Orientador : Ricardo Dahab

Trabalho final (mestrado profissional) - Universidade Estadual de Campinas, Instituto de Computação.

1. Sistemas de informação geográfica. 2. Proteção de dados. 3. Processamento de imagens. I. Dahab, Ricardo. II. Universidade Estadual de Campinas. Instituto de Computação. III. Título.

Título em inglês: Visual watermarks in vector maps.

Palavras-chave em inglês (Keywords): 1. Geographic information systems. 2. Data protection. 3. Image processing.

Área de concentração: Segurança da Informação

Titulação: Mestre em Computação

Banca examinadora: Prof. Dr. Ricardo Dahab (IC-UNICAMP)
Prof. Dr. Léo Pini Magalhães (FEEC-UNICAMP)
Prof. Dr. Roberto de Alencar Lotufo (FEEC-UNICAMP)
Prof. Dr. Neucimar J. Leite (IC-UNICAMP)
Prof. Dr. Siome Klein Goldstein (IC-UNICAMP)

Data da defesa: 18/08/2005

Marcas d'Água Visuais em Mapas Vetoriais

Este exemplar corresponde à redação final do Trabalho Final devidamente corrigida e defendida por Douglas Aurélio Marques e aprovada pela Banca Examinadora.

Campinas, 18 de Agosto de 2005

Prof. Dr. Ricardo Dahab (Orientador)

Trabalho Final apresentado ao Instituto de Computação, UNICAMP, como requisito parcial para obtenção do título de Mestre em Computação na Área de Engenharia da Computação.

TERMO DE APROVAÇÃO

Trabalho Final Escrito defendido e aprovado em 18 de agosto de 2005, pela Banca Examinadora composta pelos Professores Doutores:

Prof. Dr. Léo Pini Magalhães
FEEC - UNICAMP

Prof. Dr. Roberto de Alencar Lotufo
FEEC - UNICAMP

Prof. Dr. Neucimar Jerônimo Leite
IC - UNICAMP

Prof. Dr. Siome Klein Goldenstein
IC - UNICAMP

Prof. Dr. Ricardo Dahab
IC - UNICAMP

Dedicatória

Dedico este trabalho à memória de meu querido irmão Denilson.

Agradecimentos

Agradeço à minha esposa Junia pelo apoio e incentivo.

Ao Professor Ricardo Dahab pela confiança em meu trabalho.

Ao Professor Neucimar J. Leite, com quem aprendi os fundamentos de processamento de imagens que me foram muito úteis neste trabalho.

Resumo

Com a proliferação dos sistemas de informação geográfica (GIS) e a disponibilidade crescente de dados espaciais, empresas de vários segmentos têm tido a oportunidade de criar bases de dados geográficas para uso próprio ou fornecimento de serviços. A representação de informações nessas bases de dados se dá através de mapas em formato vetorial, produzidos geralmente a um alto custo, mas de fácil reprodução. Este trabalho tem como objetivo a apresentação de um novo método para inserção de marcas d'água visuais em mapas vetoriais digitais, para combater a cópia e a distribuição ilegais desse tipo de mídia. Neste novo método, a marca d´água, formada pelos pixels de uma imagem, é inserida em um mapa através de deslocamentos controlados de suas coordenadas. A detecção da marca d´água em um mapa é feita com a extração da imagem inserida e sua comparação com a original. Vários experimentos práticos mostram que o método é bastante resistente a diversos tipos de ataque, como *cropping*, adição de coordenadas, alteração da ordem das entidades, inserção de ruído, transformações locais e globais e combinações desses ataques.

Abstract

The proliferation of geographic information systems (GIS) and the increasing availability of spatial data makes it possible to create geographic databases for a large variety of ends and services. The information in these databases is represented by digital vector maps which are expensive to produce, but easy to copy. In this work we present a new method for embedding visual watermarks into digital vector maps to counter illegal copying and distribution of these digital contents. In this new method, the watermark, formed by pixels of an image, is embedded within a map by controlled displacements of its coordinates. The detection of the watermark within a map is accomplished by extracting the embedded image and comparing it with the original one. Many experiments show that the method has good resiliency against attacks such as cropping, coordinate addition, entity order scrambling, random noise insertion, global and local transformations and some combinations of these attacks.

Lista de Figuras

Lista de Tabelas

Lista de Algoritmos

Lista de Abreviaturas e Siglas

2D	Bidimensional, em que as coordenadas são representadas por tuplas (x, y)
3D	Tridimensional, em que as coordenadas são representadas por tuplas (x, y, z)
BMP	*Bitmap*, formato digital de imagem.
CAD	*Computer Aided Design*, desenho assistido por computador
DGN	*Design*, formato vetorial CAD proprietário da Bentley, utilizado pelo Microstation
DWG	*Drawing*, formato vetorial digital proprietário da Autodesk, utilizado pelo AutoCAD
DXF	*Drawing Exchange Format*, formato vetorial digital aberto desenvolvido pela Autodesk para facilitar a abertura de arquivos do AutoCAD por outros programas de CAD
GIF	*Graphics Interchange Format*, formato digital de imagem
GIS	*Geographic Information System*, sistema de informação geográfica
GPS	*Global Positioning System*, sistema de posicionamento global
KNN	*K-Nearest Neighbors*, *K*-vizinhos mais próximos. Dados um conjunto de elementos e uma métrica de distância entre dois elementos do conjunto, um algoritmo de KNN encontra os K vizinhos mais próximos de um dado elemento do conjunto
MD	Marca d'água
MIF	*Map Info Data Interchange Format*, formato de mapa vetorial do MapInfo, que permite que informações genéricas sejam armazenadas juntamente com entidades gráficas
MSE	*Mean Square Error*, erro quadrático máximo
PGM	*Portable Grey Map*, formato de imagem *raster* em tons de cinza
PPM	*Point Pattern Matching*, casamento de padrões de pontos, é um algoritmo que analisa dois conjuntos de pontos e verifica a correspondência entre os pontos dos dois conjuntos
PSNT	*Peak Signal to Noise Ratio*, razão máxima sinal-ruído
RAWVec	*Raster Watermarks in Vector Maps*, ou Marcas d'Água *Raster* em Mapas Vetoriais. Novo método de marcas d'água apresentado neste trabalho
SHP	*Shapefile*, formato de mapa vetorial da ESRI, que armazena entidades geométricas e atributos associados
TD	Triangulação de Delaunay
TIFF	*Tagged Image File Format*, formato de imagem *raster* largamente utilizado que se tornou um padrão *de facto* para imagens de 32 bits
UTM	*Universal Transverse Mercator*, sistema de projeção cartográfica que projeta a superfície da Terra sobre um cilindro tangente a um meridiano

Lista de Símbolos

M	Mapa vetorial original a ser marcado
$\hat{M}$	Mapa marcado, contendo entidades marcadas em que suas coordenadas foram deslocadas pela adição de informações da marca d'água
t	Número total de pontos de $v(M)$, que é a representação por pontos do mapa marcado M
n	Número de linhas e de colunas das matrizes quadradas utilizadas pelos algoritmos
$\mathbf{A}_x$	Matriz quadrada em que são distribuídas as abscissas das coordenadas de $v(M)$
$\mathbf{A}_y$	Matriz quadrada em que são distribuídas as ordenadas das coordenadas de $v(M)$
$\mathbf{R}$	Marca d'água original, imagem *raster* a ser inserida em $v(M)$, $(l \times c)$
r_{ij}	Intensidade do pixel da linha i e coluna j da imagem R
$\mathbf{E}$	Marca d'água estendida $(n \times n)$, produzida estendendo-se ou reduzindo-se a imagem R para caber em uma matriz $n \times n$
I_{max}	Sendo r_{ij} o pixel de maior intensidade da imagem R, I_{max} é o valor de sua intensidade
I_{min}	Sendo r_{ij} o pixel de menor intensidade da imagem R, I_{min} é o valor de sua intensidade
I_m	Valor utilizado para normalizar as intensidades dos pixels de R ao gerar a imagem E
$\mathbf{B}_x$	Matriz quadrada em que são distribuídas as abscissas das coordenadas de $v(M)$ após serem marcadas
$\mathbf{B}_y$	Matriz quadrada em que são distribuídas as abscissas das coordenadas de $v(M)$ após serem marcadas
C	Constante real utilizada para controlar o deslocamento máximo ao inserir um pixel em uma coordenada
P	$P = v(\hat{M})$ é a representação por pontos do mapa marcado $\hat{M}$.
N	Mapa no qual se quer detectar a marca d'água. Corresponde ao mapa $\hat{M}$ após ter sofrido ataques
Q	$Q = v(N)$ é a representação por pontos do mapa N
κ	Número de vizinhos no algoritmo PPM
ρ	Probabilidade do match no algoritmo PPM
τ	Raio de tolerância de match no algoritmo PPM
$\mathbf{D}_x$	Imagem detectada nas abscissas dos pontos de $v(N)$
$\mathbf{D}_y$	Imagem detectada nas ordenadas dos pontos de $v(N)$
$\mathbf{D}$	Imagem detectada a partir da média de D_x e D_y
$\mathbf{S}$	Imagem de prova, calculada a partir da média das intensidades dos pixels que se repetem em D

Sumário

Capítulo 1 – Introdução

1.1. Motivação

Mapas vetoriais são conteúdos digitais que utilizam primitivas como pontos, linhas e polígonos para representar objetos no espaço geográfico. A crescente demanda por informações que incluam algum componente geográfico, como serviços de navegação por GPS (*Global Positioning System*), consulta de mapas pela internet e GIS (*Geographic Information Systems*) vem tornando esse tipo de mídia cada vez mais difundida. Mapas utilizados em bases de dados geográficas possuem um alto valor ([Voigt+04], [Giannoula+02]). Isso porque a criação desses mapas geralmente envolve recursos e processos dispendiosos, como levantamento de informações por equipes de campo, aquisição, análise e processamento de fotos aéreas ou imagens de satélite e vetorização. Além disso, são altos os custos para manter essas informações atualizadas [Voigt+02]. Ao mesmo tempo, por se tratar de uma mídia digital, a reprodução fiel desses mapas se torna trivial, possibilitando a distribuição de cópias ilegais. Conseqüentemente, como ocorre com outros tipos de mídia, existe uma crescente necessidade de métodos para determinar a autoria das informações. A inclusão nos mapas de marcas d'água digitais, compostas de informações que sejam imperceptíveis em seus respectivos domínios de aplicação, permite identificar sua autoria e combater sua distribuição e utilização ilegais.

Marcas d'água digitais, ou simplesmente marcas d'água (MD), têm sua origem na Esteganografia e seu objetivo é inserir[1] informações em conteúdos digitais de forma que estas não interfiram com seu uso pretendido ([Ohbuchi+00], [Sion02]). As informações inseridas podem ser utilizadas, por exemplo, para indicar autoria, garantir autenticidade e integridade das informações ou adicionar informações úteis para o tipo de mídia em questão.

Neste trabalho, propomos o método de Marcas d'Água *Raster*[2] em Mapas Vetoriais (RAWVec – *Raster Watermarks in Vector Maps*) para a inserção de MDs visuais em mapas vetoriais, de modo que sua autoria possa ser determinada após sua distribuição. Serão tratados os mapas 2D, mas as

[1] do inglês: *embed*
[2] Uma imagem *raster* é uma estrutura de dados que representa uma grade retangular de pixels.

idéias podem ser estendidas também para mapas 3D. Nosso novo método insere no mapa uma imagem *raster*, representada por uma matriz de pixels em tons de cinza, aplicando deslocamentos nas coordenadas das entidades[3] nele contidas. Os deslocamentos são aplicados de acordo com as intensidades dos pixels da imagem e podem ser controlados de forma a manter as coordenadas dentro de um raio de tolerância, observando-se a tolerância máxima de erro exigida no domínio de aplicação do mapa. Por exemplo, mapas utilizados para navegação urbana são mais tolerantes a erros nas coordenadas do que aqueles utilizados para localização de equipamentos em uma rede de gás enterrada, em que a localização precisa é crítica. A detecção da imagem inserida é feita utilizando-se o mapa original. A imagem detectada pode então ser comparada com a imagem inserida originalmente para a identificação da MD.

A comparação entre a imagem original e a imagem detectada pode ser dividida em duas etapas: a primeira utiliza análise estatística para definir a probabilidade de as duas imagens serem iguais (por exemplo, correlação [Praun+99]). A segunda utiliza a própria percepção visual humana para identificar similaridades entre as imagens. Tanto a comparação estatística quanto a visual devem gerar índices que possibilitem dizer se as imagens se correspondem ou não. Falso-positivos e falso-negativos podem ocorrer, como em outros métodos, mas a utilização de uma comparação visual além da estatística fornece subsídios para diminuir sua incidência.

Algumas vantagens do método são: (a) sua simplicidade, em que a inserção e a extração da MD são feitas através de simples adições e subtrações de matrizes e (b) sua resistência a diversos ataques devido ao fato de se utilizar do sistema visual humano na identificação da MD. Como será mostrado, ataques à MD são refletidos em mudanças nas intensidades dos pixels na imagem extraída. Após sofrer ataques, dentro de certos limites estudados neste trabalho, a imagem pode ainda ser reconhecida pelo olho humano. Podem ser aplicadas métricas de comparação de imagens para uma avaliação automática de similaridade. Também podem auxiliar no reconhecimento da imagem a análise de seu histograma para a aplicação de *thresholds*[4]

[3] Uma entidade em um mapa vetorial é a representação digital de um ente geométrico, como por exemplo um ponto ou um polígono.
[4] Um *threshold* é o limiar da intensidade de pixels de uma imagem, abaixo e acima do qual os pixels de uma imagem são considerados como tendo intensidade mínima e máxima respectivamente.

adequados e a aplicação de filtros, como os que eliminam ruídos, que detectam contornos em imagens ou que realizam a segmentação da imagem em busca de algum padrão ou forma.

1.2. Notação e Convenções

As seguintes convenções são utilizadas neste documento:

- Aparecem em itálico os termos no original em língua estrangeira; nomes de constantes, variáveis e funções; nomes de autores. Exemplos: *Geographic Information Systems*, $f(x)$.
- Aparecem em negrito os termos que mereçam destaque. Exemplo: **marcas d'água**.
- Nomes de matrizes, vetores e listas aparecem em letra maiúscula e em negrito. Exemplos: imagem **E**, lista **L**.
- Listagens de algoritmos aparecem em fonte diferenciada. Exemplo: `i = 1 até n`.
- Referências a outros trabalhos são indicadas entre colchetes, com o sobrenome do autor quando houver somente um, ou o sobrenome de um dos autores seguido de "+", quando houver mais de um. Por último, vem o ano de publicação com dois dígitos. Exemplo: [Ohbuchi+00].

1.3. Organização do Trabalho

No Capítulo 2, serão descritos os tipos de mídia em discussão neste artigo: imagens *raster* e mapas vetoriais, no contexto de MDs digitais. Também serão descritas algumas características e aplicações de MDs.

No Capítulo 3, são descritos alguns trabalhos que tratam de MDs digitais em vetores.

No Capítulo 4, trataremos do novo método de Marcas d'Água *Raster* em Mapas Vetoriais (RAWVec). Serão descritos os algoritmos para inserção e detecção de MDs em mapas vetoriais utilizando imagens *raster*. Também serão descritos os tipos de ataques mais comuns a MDs e o

comportamento do método diante de tais ataques. Ao final do capítulo, trataremos de algumas limitações do método.

Finalmente, no Capítulo 5, apresentamos nossas conclusões sobre os temas abordados neste trabalho e sugestões de melhorias que poderiam ser exploradas em um trabalho futuro

Capítulo 2 – Marcas d'Água Digitais

Neste capítulo serão descritas algumas características das imagens *raster* e dos mapas vetoriais, com o objetivo de destacar as vantagens destes últimos no armazenamento de informações, seu valor intrínseco e a conseqüente necessidade de marcar seu conteúdo. Em seguida, serão discutidas algumas aplicações das MDs e alguns de seus critérios de classificação.

2.1. Características das Imagens *Raster*

Imagens *raster* utilizam matrizes bidimensionais de pixels para representar informações visuais. A cada pixel está associada uma informação de cor ou de nível de cinza. Alguns formatos comuns deste tipo de mídia são BMP, GIF, TIFF e PGM.

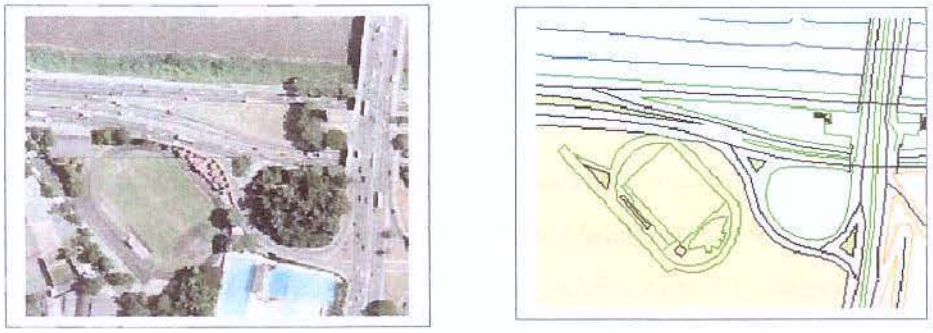


Figura 2.1- Exemplos de imagem *raster* e arquivo vetorial representando uma mesma informação

Uma característica importante das imagens *raster* é que, apesar de aparentemente mais ricas em informações visuais do que arquivos vetoriais, como ilustra a Figura 2.1, seu poder de representação é bem mais limitado do que o destes últimos, devido a características como:

- Representação das informações dependente da resolução da imagem.

- Possibilidade limitada de manipulação das informações, pois a seleção de itens que compõem a imagem deve ser feita selecionando-se todos os pixels que os representam, o que cria o problema da separação de tais pixels do resto da imagem (segmentação). Para isso, são necessários algoritmos de segmentação cuja eficácia depende de fatores como gradiente e contraste dos pixels da imagem.

- Perda de informações ocorrem tipicamente na aplicação de transformações, como ampliação e rotação por exemplo.

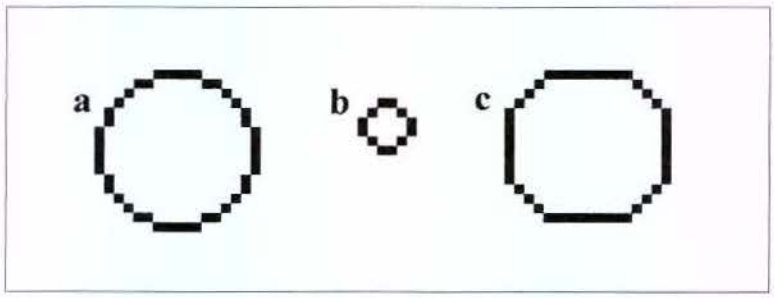


Figura 2.2 - Exemplos de representação limitada de imagens *raster*.

Como exemplo, na Figura 2.2, o "círculo" da esquerda (a) foi reduzido (b) e, em seguida, o "círculo" (b) foi ampliado (c) utilizando um software de edição de imagens. Devido à perda de informação no processo, a forma original não se mantém mesmo tendo sido feita a operação inversa.

Uma outra característica das imagens *raster* é que podem sofrer filtragens com objetivos diversos. Podemos destacar como exemplos os filtros para suavização, detecção de contorno e segmentação. A aplicação de filtros permite, por exemplo, recuperar informações que podem ter sido degradadas devido à inserção de ruído, ou ainda destacar determinadas características da imagem. A Figura 2.3 exemplifica a aplicação de um filtro em uma imagem.

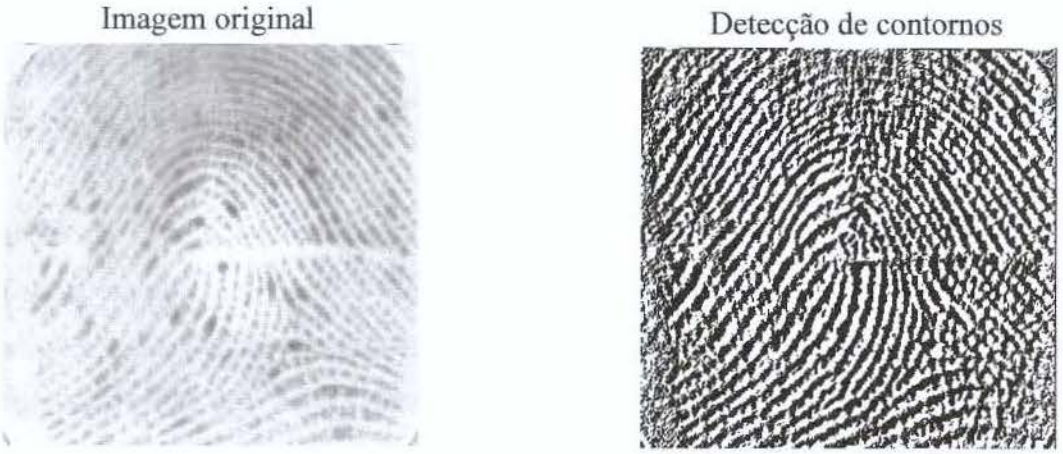


Figura 2.3 - Exemplo de aplicação de filtro para detecção de contornos.

2.2. Características dos Mapas Vetoriais

Arquivos de dados vetoriais representam as informações sob a forma de entidades matemáticas, como pontos, retas, polígonos, etc., em um sistema de coordenadas. Por exemplo, um segmento de reta pode ser representado por um par de coordenadas que, ao serem interpretadas por uma aplicação, geram uma linha no momento da exibição. A Figura 2.4 ilustra a representação de uma entidade em um arquivo vetorial.

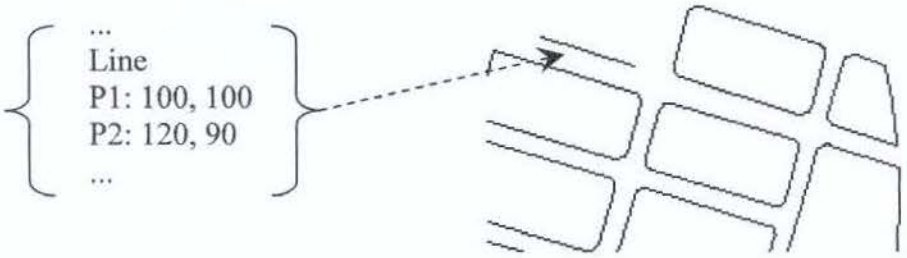


Figura 2.4 - Exemplo hipotético de representação interna de uma linha em um mapa vetorial.

Arquivos vetoriais são facilmente modificáveis e podem ser utilizados para organizar as informações em camadas, de modo que entidades possam se sobrepor umas às outras sem comprometimento das informações que representam. Transformações como rotação e ampliação

podem ser efetuadas sem perda de qualidade, embora também possam introduzir erros devido à limitação na precisão de representação das coordenadas. Nesses tipos de mídia, a precisão na representação das informações só depende da precisão numérica das coordenadas das entidades.

Um tipo particular de arquivo de dados vetoriais são os mapas vetoriais. São passíveis de uma organização interna que permite uma gama maior de aplicações do que mapas representados por imagens *raster*, como fotografias aéreas, por exemplo. Por possuírem dados vetoriais, podem ser alterados e atualizados muito mais facilmente que as imagens *raster*. Outra característica que determina o alto valor dos mapas vetoriais é que geralmente necessitam de intenso trabalho humano para serem produzidos e para serem mantidos atualizados.

Entidades típicas representadas em arquivos vetoriais são pontos, símbolos (representados por um ponto e um estilo de exibição associado), textos, linhas, poligonais, polígonos, diversos tipos de arcos e curvas, círculos, elipses, etc. Alguns exemplos de formatos de arquivos vetoriais utilizados no mercado são DWG, DXF, DGN, SHP, MIF, etc.

2.3. Marcas d'Água Digitais e Aplicações

Uma MD é um conjunto de informações inseridas em um conteúdo digital a ser distribuído. Um conteúdo em que tenha sido inserida uma MD é chamado de conteúdo marcado ou hospedeiro. Imagens *raster*, arquivos de áudio e vídeo são alguns dos conteúdos a que mais tipicamente se aplicam técnicas de MDs. Dentre as principais aplicações deste recurso, podemos destacar:

- Identificação de autoria: as MDs são úteis no rastreamento dos conteúdos digitais, permitindo ao detentor dos direitos ter evidências de que uma determinada informação é de sua autoria.

- Garantia de integridade ou autenticidade: neste caso, adulterações no conteúdo digital podem ser detectadas, pois provocam a destruição parcial ou total da MD.

- Inserção de informação extra, útil dentro do domínio de aplicação do conteúdo digital em questão. Por exemplo, *Sonnet et al.* introduzem o conceito de **Marca d'Água de Ilustração**[5], que é um tipo de MD utilizado para acrescentar detalhes às informações presentes em um conteúdo digital [Sonnet+03].

Tendo em vista que os conteúdos marcados podem ser utilizados para fins diversos, deve-se esperar que possam sofrer diversos tipos de alteração em sua utilização. Por exemplo, uma imagem pode ser ampliada ou processada de forma a aumentar seu contraste. As alterações sofridas pelo conteúdo marcado podem provocar alterações também na MD, ou mesmo destruí-la completamente de forma a não ser mais identificável. Essas alterações, intencionais ou não, são denominadas **ataques**. A necessidade de resistência ou não da MD a modificações no conteúdo marcado depende da aplicação à qual a marca d'água se destina.

2.4. Classificação de Marcas d'Água Digitais

Para um melhor entendimento das aplicações de marcas d'água, surge a necessidade de categorizá-las de acordo com determinados critérios. Serão descritos a seguir os critérios de transparência, robustez, necessidade do conteúdo original para detecção e domínio de inserção.

2.4.1. Transparência

Marcas d'água digitais devem ser imperceptíveis à verificação direta pelos usuários e, portanto, são ditas transparentes. Porém, em relação à similaridade entre os dados originais e os dados marcados, podem ser definidos dois tipos de transparência: funcional e perceptual [Ohbuchi+00]. Quando o conteúdo original e o marcado são indistinguíveis para os usuários humanos, a transparência é dita **perceptual**. Por outro lado, quando os conteúdos são indistinguíveis dentro de suas respectivas aplicações, a transparência é dita **funcional**. Por exemplo, um mapa vetorial marcado, em que a MD foi inserida através de perturbações nas coordenadas, pode ter transparência funcional devido ao fato de as perturbações estarem dentro do limite de erro

[5] no original: *Illustration Watermark*

aceitável para o mapa. Mas o mesmo mapa pode não ter transparência perceptual, por terem surgido ângulos perceptíveis em regiões que antes apareciam como curvas suaves.

O conceito de transparência está intimamente ligado ao conceito de **utilidade**[6] do conteúdo marcado [Sion02]. Qualquer marcação de conteúdo digital provoca alterações em suas propriedades originais. Dependendo do tipo de aplicação a que o conteúdo se destina, este só será **utilizável** se as alterações não ultrapassarem a tolerância exigida pelo domínio de sua aplicação.

2.4.2. Dificuldade de Remoção ou Robustez

Nesta categoria, as marcas d'água podem ser classificadas em frágeis e robustas [Ohbuchi+00]. Uma MD **robusta** é aquela projetada para resistir a alterações no conteúdo digital no qual está inserida. Uma aplicação de MDs robustas é na identificação de autoria. Neste caso, a prova de autoria é possível ao se provar que as informações extraídas do conteúdo exibem certas propriedades, conhecidas pelo detentor dos direitos, que tenham virtualmente nenhuma probabilidade de ocorrerem aleatoriamente. Para que cumpra seu propósito, é necessário garantir que a informação inserida resista a diversos tipos de modificações, sejam elas intencionais ou acidentais. As modificações, são chamadas de **ataques** e podem visar à destruição da marca d'água para que o conteúdo possa ser utilizado livremente.

Uma marca d'água **frágil** é aquela projetada para ser sensível a alterações no conteúdo no qual está inserida. Como tal, é útil na identificação da autenticidade de um conteúdo. Mesmo pequenas alterações no conteúdo devem necessariamente provocar a destruição parcial ou total da marca d'água, permitindo dessa forma a detecção de fraudes e adulterações.

---

[6] do original *usability*.

2.4.3. Uso do Conteúdo Original para Detecção

Nesta categoria, as marcas d'água que necessitam do conteúdo original para serem extraídas são ditas **privadas** e seu processo de extração é dito **não-cego**[7] [Ohbuchi+00]. Já as que podem ser extraídas sem a necessidade de utilização do conteúdo original são ditas **públicas** e seu processo de extração é dito **cego**[8].

2.4.4. Domínio de Inserção

A informação associada à MD pode ser inserida em um conteúdo digital no domínio da freqüência ou no domínio espacial [Wang+98]. No primeiro caso, a informação é inserida diretamente nos pixels ou coordenadas do conteúdo digital. No segundo caso, é extraída uma representação do conteúdo no domínio da freqüência, como por exemplo, utilizando análise espectral ([Ohbuchi+03], [Praun+99], [Voigt+04]), e nesta representação a informação é inserida. Por fim, a representação é reconvertida para o domínio espacial.

[7] do original *non-blind*
[8] do original *blind*

Capítulo 3 – Trabalhos Correlatos

Técnicas de inserção de marcas d'água em diversos tipos de mídia têm sido amplamente estudadas, mas bem menos estudado tem sido o tema de MDs em arquivos vetoriais, especialmente mapas [Giannoula+02]. De um modo geral, os métodos de inserção de MDs em arquivos vetoriais seguem a regra de inserir informações em parâmetros característicos do conteúdo. Em [Hartung+98], por exemplo, os autores tratam da inserção de MDs em arquivos de animação facial e os parâmetros são os relacionados à animação facial 3D. No caso de mapas vetoriais, os parâmetros são geralmente as coordenadas das entidades.

Métodos robustos são aqueles em que a MD deve necessariamente ser resistente a alguns tipos de ataque. Uma suposição utilizada com freqüência é que ataques à MD produzem alterações que também prejudicam a utilização do mapa, ou sua utilidade [Sion02]. Ou seja, quanto mais eficiente o ataque em termos de descaracterizar a MD, mais terá prejudicado os dados em que ela está inserida. Esta suposição, no caso de mapas vetoriais, está correta. Todo mapa de uso comercial possui um erro máximo tolerável nas coordenadas de suas entidades, o que funciona como um limitador para a intensidade do ataque. Acima de certo patamar de erro, o mapa perde seu valor por não mais corresponder a uma representação confiável do mundo real.

No que se refere à resistência a ataques de transformação, de um modo geral, os algoritmos para MDs em mapas vetoriais cobrem somente as transformações afins ou de similaridade ([Ohbuchi+02], [Ohbuchi+03] e [Voigt+04]). Estes algoritmos não trazem uma análise do ponto de vista de transformações mais complexas, como as que envolvem mudança de datum e de sistema de projeção, que são comuns no contexto de mapas. No presente trabalho, essas transformações complexas também não são tratadas pelos algoritmos, devendo ser identificadas e removidas do mapa antes da aplicação do método de detecção da MD.

Muitos algoritmos também não analisam as alterações na topologia das entidades após a inserção da marca d'água. Por exemplo, não tratam do problema de garantir que entidades fechadas, formando uma região, continuem fechadas após a inserção, ou entidades que se tocam continuem

se tocando (em [Praun+99] isso é discutido no que se refere a manter a conectividade das malhas triangulares tridimensionais ao inserir a MD). Da mesma forma, muitos não tratam dos problemas relativos a alteração de áreas, perímetros, comprimentos e distâncias relativas das entidades após a inserção da MD. Pode haver aplicações em que alterações nesses parâmetros não são toleradas e, neste caso, é necessário analisar a possibilidade de que a inserção da MD mantenha essas propriedades invariáveis, dentro de certa tolerância. Uma exceção é [Ohbuchi+00], cujo método preserva a forma e o tamanho de curvas paramétricas após a inserção da MD. O presente trabalho considera alguns tipos de relações topológicas, como será discutido no capítulo seguinte.

Utilizando inserção de marcas d'água em modelos de superfície, *Ohbuchi et al.* [Ohbuchi+00] desenvolvem uma técnica de inserção de informações em arquivos vetoriais compostos de malhas poligonais de objetos 3D. Inicialmente, os autores propõem autenticação como sendo uma operação sobre um tipo abstrato de dados multimídia, do qual arquivos de CAD fazem parte. Em seguida, apresentam alguns métodos para inserir MDs nesse tipo de dados com o objetivo de verificar autoria. Os métodos se aplicam a curvas e superfícies paramétricas e exploram alguns tipos de redundância que essas entidades costumam apresentar. As informações da marca d'água são inseridas nos parâmetros das entidades (coordenadas dos pontos de controle das curvas), levando em consideração duas abordagens: reparametrização e inserção de nós, as duas garantindo a preservação da forma das entidades. A reparametrização não é robusta. Já a inserção de nós é robusta desde que se considere que a forma das entidades deva se manter, ou seja, tentativas de remoção da marca d'água irão causar alteração na forma, impedindo a utilização dos dados em sua aplicação. Nos casos de desenhos mecânicos de precisão essa suposição é válida, mas para desenhos 3D para outros fins pode não ser o caso. Além disso, por considerarem somente arquivos compostos por modelos 3D, com curvas e superfícies paramétricas, tipicamente utilizados em desenho mecânico, os métodos não se aplicam diretamente a mapas, que geralmente possuem entidades mais simples.

Em [Praun+99] é descrito um método para inserção de marcas d'água em superfícies modeladas por malhas triangulares 3D. Os autores estendem a abordagem de *spread spectrum*, isto é, a inserção da marca d'água no domínio da freqüência, para a utilização de malhas tridimensionais.

Como se trata de malhas conexas, o método apresentado pelos autores trata de manter a conectividade das malhas após a inserção da MD. Não são analisados os ataques de transformações, deixando-os juntamente com outros ataques como uma possibilidade de pesquisa futura.

Em [Gou+04] é apresentado um método de marcas d'água em mapas vetoriais 2D representados por curvas *B-splines*. Inicialmente, as entidades do mapa são analisadas e sua representação por curvas é criada. Em seguida, as informações da MD são inseridas no domínio da freqüência dos pontos de controle das curvas. A detecção é feita comparando o mapa de prova e o mapa original, subtraindo os valores dos pontos de controle. A comparação entre a MD de prova e a MD original é feita através de correlação. Não é discutida a complexidade do método. É dito que o mapa de prova deve ser registrado com o original, ou seja, uma eventual transformação que tenha sofrido deve ser removida, mas a remoção é manual. Também assume-se que as entidades do mapa podem ser representadas sob a forma de curvas. O algoritmo não se aplica, portanto, a mapas compostos somente por símbolos pontuais, que foram um dos tipos de mapas que tratamos neste trabalho. A robustez é limitada à resistência a alguns ataques que mantenham fixo o número de pontos de controle das curvas, o que pode não ser o caso quando há mudança do formato do arquivo. Os autores também não tratam da manutenção da topologia das entidades do mapa.

Em [Voigt+02], os autores tratam especificamente do tema de marcas d'água em mapas 2D utilizados em GIS, com o objetivo de determinar autoria. A marca d'água, representada por seqüências pseudo-aleatórias (*pseudo-noise sequences* ou *PN-sequences*) é inserida dentro da faixa de tolerância das coordenadas do mapa. Para a detecção, é feita uma correlação entre a seqüência original e o mapa marcado e é verificada a probabilidade de que o mapa marcado seja realmente o original. Um tema importante que não foi tratado no artigo em questão é sua dependência em relação à ordem das entidades no mapa. O algoritmo necessita que as entidades apareçam no mapa marcado na mesma ordem em que aparecem no mapa original. Os autores se limitam a dizer que a resistência do algoritmo neste aspecto será estudada oportunamente. Também não é tratada a questão da manutenção da topologia.

Tratando de mapas utilizados em GIS, *Giannoula et al.* [Giannoula+02] trazem um método de marcas d'água em curvas de nível. Uma característica particular das curvas de nível é que o número de vértices em geral é grande e as curvas são poligonais fechadas. A inserção é feita no domínio da freqüência, modificando os coeficientes da transformada discreta de Fourier para cada poligonal. É interessante notar que o método para detecção da MD não necessita do mapa original, mas ainda necessita da MD original. O método é invariante a alguns ataques de transformação, em relação aos quais o cálculo da transformada discreta de Fourier das linhas é invariável. Também é dito que o método é invariante à "suavização" das linhas utilizando filtros da média e da mediana, embora não se deixe claro se o ataque trata de inserção de novos pontos para suavização ou modificação de pontos existentes.

Também tratando especificamente de marcas d'água em mapas vetoriais, *Ohbuchi et al.* [Ohbuchi+02] descrevem um algoritmo em que as informações são inseridas nos vértices das entidades do mapa, incluindo certa redundância para maior resistência a ruído aleatório. As informações são inseridas em setores do mapa, obtidos através de subdivisão por *quadtree*. A marca d'água é inserida em cada folha da subdivisão por *quadtree*, para aumentar a resistência a cropping. Não é discutida a manutenção de relações topológicas entre as entidades. Utilizam um algoritmo de *Point Pattern Matching* (*pose normalization*) para remover uma eventual transformação aplicada ao mapa de prova, embora não discuta o impacto de sua complexidade no método.

Em [Ohbuchi+03], as informações da MD são inseridas em uma representação dos vértices do mapa no domínio da freqüência, obtida através de análise espectral. Esta análise é feita com a criação de uma malha poligonal sobre as coordenadas do mapa e, em seguida, a extração do Laplaciano dessa malha. Ao ser feito o re-mapeamento para o domínio espacial, os vértices sofrem um deslocamento em função da perturbação inserida no domínio da freqüência, o que representa a MD inserida. O algoritmo é altamente complexo, necessitando a certa altura do cálculo dos auto-vetores para a matriz de vizinhança produzida pela Triangulação de Dalaunay envolvendo todos os vértices do mapa. Os autores também não tratam da manutenção de relações topológicas entre as entidades. No método apresentado pelos autores, é utilizado

também um algoritmo de *pose normalization* para remover uma eventual transformação aplicada ao mapa de prova, embora não se discuta o impacto de sua complexidade no método.

Em [Voigt+04], é apresentado um esquema de marcas d'água reversíveis em mapas 2D, isto é, em que tanto o mapa original quanto a MD original podem ser obtidos com precisão a partir do mapa marcado. A inserção é feita nos coeficientes mais altos da transformada discreta do co-seno e o erro máximo inserido no mapa pode ser previsto e controlado. A resistência a ataques não é tratada no artigo.

Em [Sonnet+03], também é apresentado um método para marcar arquivos vetoriais, mas utilizando o conceito de marca d'água de ilustração (*Illustration Watermark*) , que é uma marca d'água com o objetivo de agregar informação útil ao mapa e não de rastrear sua autoria. A informação extra funciona como uma camada a mais de informação, que pode ser utilizada pela aplicação que irá manipular o conteúdo digital marcado. Embora mapas vetoriais não tenham sido abordados pelos autores, a idéia poderia ser estendida para mapas vetoriais, em que a MD poderia ser, por exemplo, uma imagem de satélite da área representada pelo mapa.

Como veremos no próximo capítulo, a principal característica que diferencia nosso novo método de marcas d'água de outros métodos estudados é a utilização de imagens como MDs em mapas vetoriais. Isso permite a utilização de métricas para comparação de imagens na detecção da MD, além de permitir a aplicação de filtros na imagem detectada para facilitar sua comparação com a original. Outras características importantes deste método são a manutenção de algumas relações topológicas, a robustez a diversos tipos de ataque e a transparência funcional.

Capítulo 4 – Método de Marcas d'Água *Raster* em Mapas Vetoriais

Neste capítulo será apresentado um novo método de marcas d'água que desenvolvemos, chamado de Marcas d'Água *Raster* em Mapas Vetoriais (RAWVec – *Raster Watermarks in Vector Maps*). Neste método, uma marca d'água representada por uma imagem *raster* é inserida em um mapa vetorial, permitindo desta forma determinar sua autoria. A inserção é feita no domínio espacial, através de deslocamentos nas coordenadas das entidades do mapa. Neste método, a MD pode ser classificada como privada, robusta e com transparência funcional. Assume-se que os mapas vetoriais sejam representados em algum sistema de coordenadas em projeção retangular, como UTM (*Universal Transverse Mercator*), por exemplo. Assume-se também que o sistema de coordenadas seja bidimensional, embora o método possa ser estendido também a mapas tridimensionais. A Figura 4.1 fornece uma visão geral da aplicação do método RAWVec.

A idéia por trás da inserção da MD em um mapa digital é marcá-lo de modo que possa ser identificado no futuro. Um cenário típico que ilustra a utilização do método é o seguinte: o mapa original é marcado com a informação da MD, uma imagem *raster*, produzindo um mapa marcado. Tanto a informação da MD quanto o mapa original devem ser mantidos em sigilo. O mapa marcado é então tornado público. Um terceiro se apodera do mapa e, imaginando que este possua alguma MD, ataca as informações, mas tentando não prejudicar muito o próprio mapa, na esperança de destruir a MD mas ainda possuir um mapa útil ao final. Este mesmo terceiro passa então a utilizar livremente o mapa, dizendo agora que este é de sua autoria. Em algum momento futuro, o mapa suspeito, ou mapa de prova, é descoberto pelo verdadeiro autor. O mapa de prova é então analisado em conjunto com o mapa original, extraindo-se do primeiro uma nova imagem *raster*, ou marca d'água de prova. Esta é comparada com a MD original tanto estatisticamente quanto visualmente. O resultado da comparação irá dizer se as imagens se correspondem ou não e com que grau de certeza. A correspondência entre as duas imagens é utilizada para determinar se os mapas se correspondem ou não.

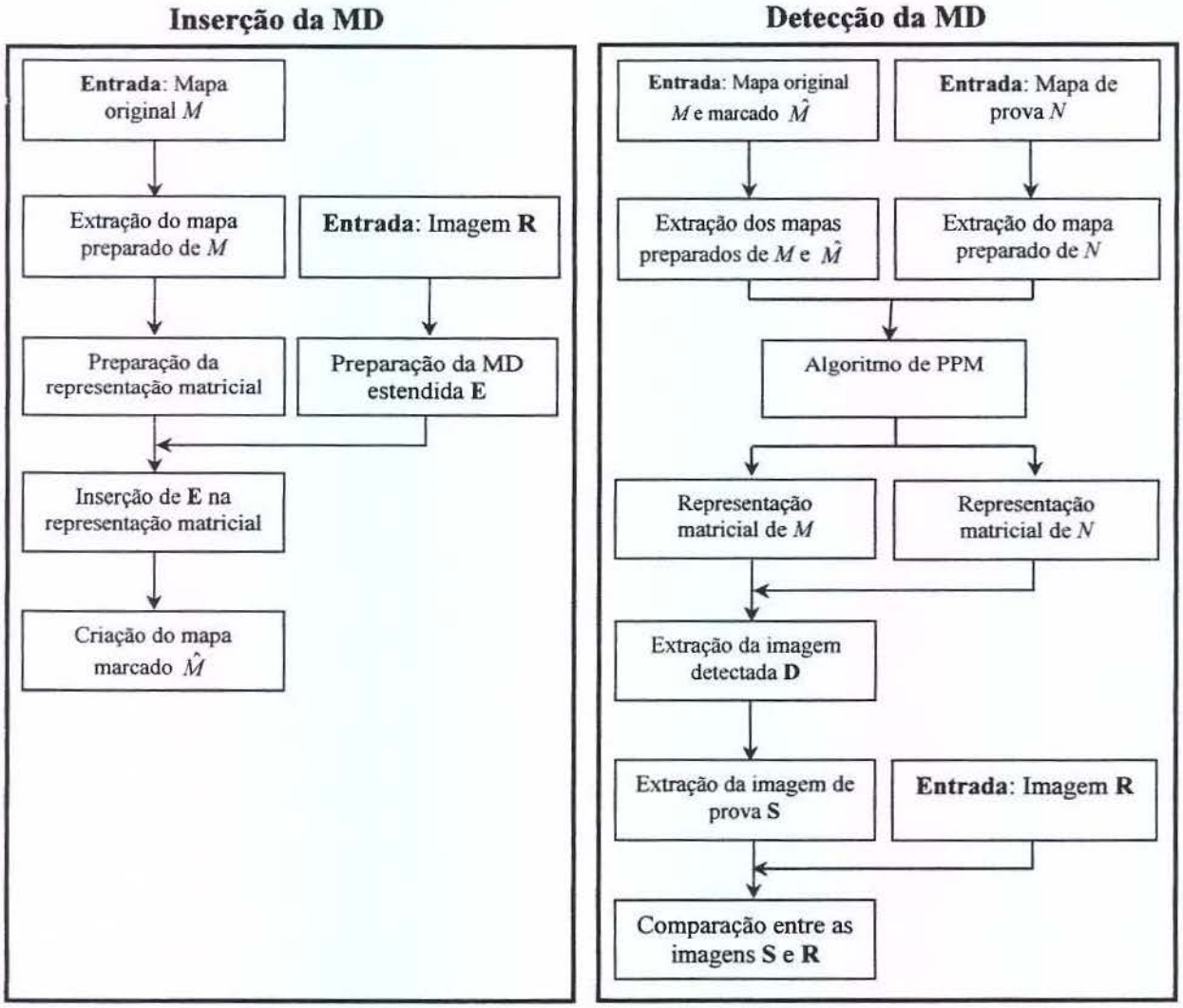


Figura 4.1– Representação das fases principais do método RAWVec.

4.1 Algoritmo para Inserção da Marca d'Água

Nesta seção, será discutida inicialmente a extração de uma representação apropriada do mapa, sobre a qual os algoritmos serão aplicados. Também será discutida a preparação da imagem *raster* que servirá como marca d'água. Em seguida, será descrito o algoritmo para a efetiva inserção da imagem no mapa vetorial.

4.1.1 Extração de uma Representação Simplificada do Mapa Vetorial

Um mapa vetorial M possui um conjunto de entidades geométricas representadas em um determinado formato digital. Independentemente do formato de representação dessas informações, necessitamos obter de M uma seqüência de coordenadas[9] a serem tratadas de modo a receberem a MD sob a forma de deslocamentos. Após o tratamento das coordenadas, as alterações se refletirão em reposicionamento das entidades, que serão copiadas para um novo mapa, a que chamaremos de mapa marcado, contendo a MD. Após a extração apropriada das coordenadas, os algoritmos ficam independentes do formato digital de representação. No método RAWVec, dividimos as entidades em três categorias:

- **Entidades pontuais**: são descritas geograficamente por um único ponto. Símbolos, pontos e textos se enquadram nesta categoria.
- **Entidades lineares**: são descritas geograficamente por uma seqüência de coordenadas. Linhas, poligonais e polígonos são exemplos desse tipo de entidade.
- **Entidades parametrizadas**: são descritas por parâmetros que definem seu posicionamento geográfico. Círculos, elipses, arcos, splines e outros tipos de curvas são exemplos típicos. Essas entidades possuem, além de outros parâmetros, um ou mais **pontos de controle**, que são utilizados para calcular sua forma e posição geográfica, muito embora tais pontos não necessariamente façam parte da própria geometria das entidades que descrevem. Por exemplo, uma representação típica do ponto de controle do círculo é seu centro, embora ele não faça parte da circunferência.

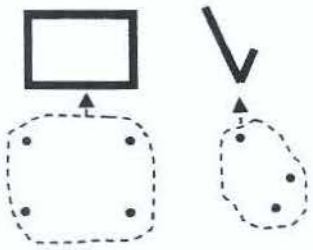


Figura 4.2 – Exemplo da representação por pontos das entidades de um mapa.

[9] Usaremos indistintamente os termos coordenada, ponto ou vértice quando se referirem às coordenadas das entidades do mapa.

Daremos a idéia básica por trás da extração das coordenadas de um mapa vetorial. Os detalhes dependem da forma como as entidades são representadas no formato digital do mapa. Devemos decompor o mapa M no que chamaremos de **mapa original preparado**, composto somente de pontos com coordenadas (x,y). Essa decomposição é ilustrada pela Figura 4.2. A representação por pontos será utilizada nas operações necessárias para a inserção da MD em M, para produzir ao final o **mapa marcado** $\hat{M}$. Seja $v(X)$ uma função que, aplicada a um mapa vetorial X, extrai dele o mapa preparado correspondente. Dado o mapa original M, onde as entidades possuem coordenadas 2D, extrai-se o mapa original preparado $v(M)$ sob a forma de pontos de coordenadas, conforme descrito no Algoritmo 4.1. A idéia básica do algoritmo é extrair em uma lista as coordenadas de cada uma das três categorias de entidades discutidas nesta seção.

```
Entrada: mapa vetorial M
Seja L = {}
INÍCIO
Para cada entidade i do mapa vetorial M, na ordem em que aparece:
      - Se i for uma entidade pontual: inserir sua coordenada diretamente
      no final de L;
      - Se i for uma entidade linear: inserir no final de L cada uma das
      coordenadas de seus vértices, na ordem em que aparecem;
      - Se i for uma entidade parametrizada: inserir no final de L seus
      pontos de controle ou, opcionalmente, as coordenadas resultantes da
      segmentação da entidade em uma seqüência finita de pontos, dentro
      de uma tolerância de representação.  Por exemplo, para um círculo
      podemos utilizar seu centro ou segmentá-lo em um número finito de
      pontos;

      Observação: Para cada coordenada inserida em L, devemos manter uma
      referência à entidade de M que a originou, de forma a ser possível
      recriar a entidade de M nas novas coordenadas ao produzir M̂.
Retornar a lista L
FIM.
```

Algoritmo 4.1 – Extração de uma lista de coordenadas de M

4.1.2 Criação de uma Representação Matricial do Mapa Vetorial

Seja t o número total de pontos em $v(M)$ e seja p_i o i-ésimo ponto de $v(M)$, com coordenadas (x_i, y_i). Dispomos a i-ésima coordenada x_i e y_i respectivamente em duas matrizes $\mathbf{A_x}$ e $\mathbf{A_y}$ de dimensões $n \times n$, onde:

$$n = \lfloor \sqrt{t} \rfloor \qquad (1)$$

$$(A_x)_{ij} = x_{n(i-1)+j}$$
$$(A_y)_{ij} = y_{n(i-1)+j} \qquad (2)$$

Os pares $[(A_x)_{ij}, (A_y)_{ij}]$ correspondem a coordenadas de $v(M)$.

$$\mathbf{A_x} = \begin{pmatrix} x_1 & x_2 & \cdots & x_n \\ x_{n+1} & x_{n+2} & \cdots & x_{2n} \\ x_{2n+1} & x_{2n+2} & \cdots & x_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ x_{(n-1)n+1} & x_{(n-1)n+2} & \cdots & x_n^2 \end{pmatrix} \quad \mathbf{A_y} = \begin{pmatrix} y_1 & y_2 & \cdots & y_n \\ y_n & y_{n+2} & \cdots & y_{2n} \\ y_{2n} & y_{2n+2} & \cdots & y_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ y_{(n-1)n+1} & y_{(n-1)n+2} & \cdots & y_n^2 \end{pmatrix} \qquad (3)$$

As matrizes $\mathbf{A_x}$ e $\mathbf{A_y}$, cujas estruturas estão detalhadas na equação (3), serão utilizadas na inserção da MD, como será descrito nas seções seguintes.

4.1.3 Tratamento de Algumas Relações Topológicas

A aplicação do método RAWVec em um mapa produzirá um novo mapa com alterações nas coordenadas das entidades do primeiro. É desejável, porém, que as novas entidades mantenham entre si algumas relações topológicas que possuíam originalmente.

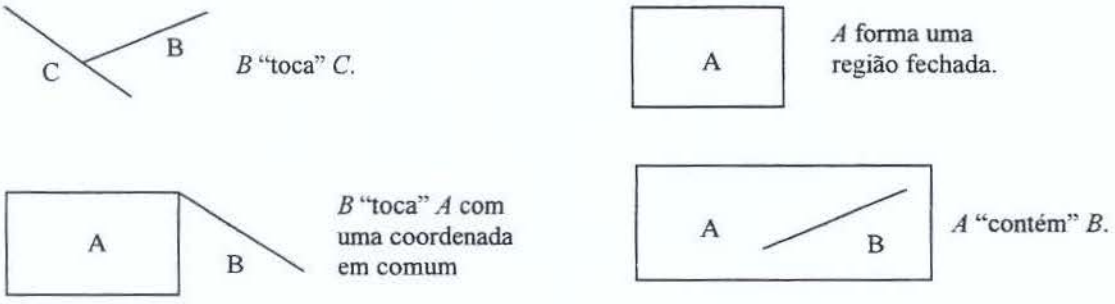


Figura 4.3 – Exemplos de relações topológicas

Um caso particular é o das entidades que compartilham coordenadas. Por exemplo, dois polígonos podem compartilhar uma aresta, o que corresponde a compartilharem dois vértices consecutivos. Outro exemplo é o de uma poligonal, em que o primeiro e o último pontos são

iguais. No contexto de mapas, é interessante manter estas mesmas propriedades após a inserção da MD nas coordenadas dessas entidades. A Figura 4.3 ilustra algumas relações topológicas.

Propomos aqui uma abordagem que trata das relações topológicas por compartilhamento de coordenadas, que é o caso mais simples de ser analisado.

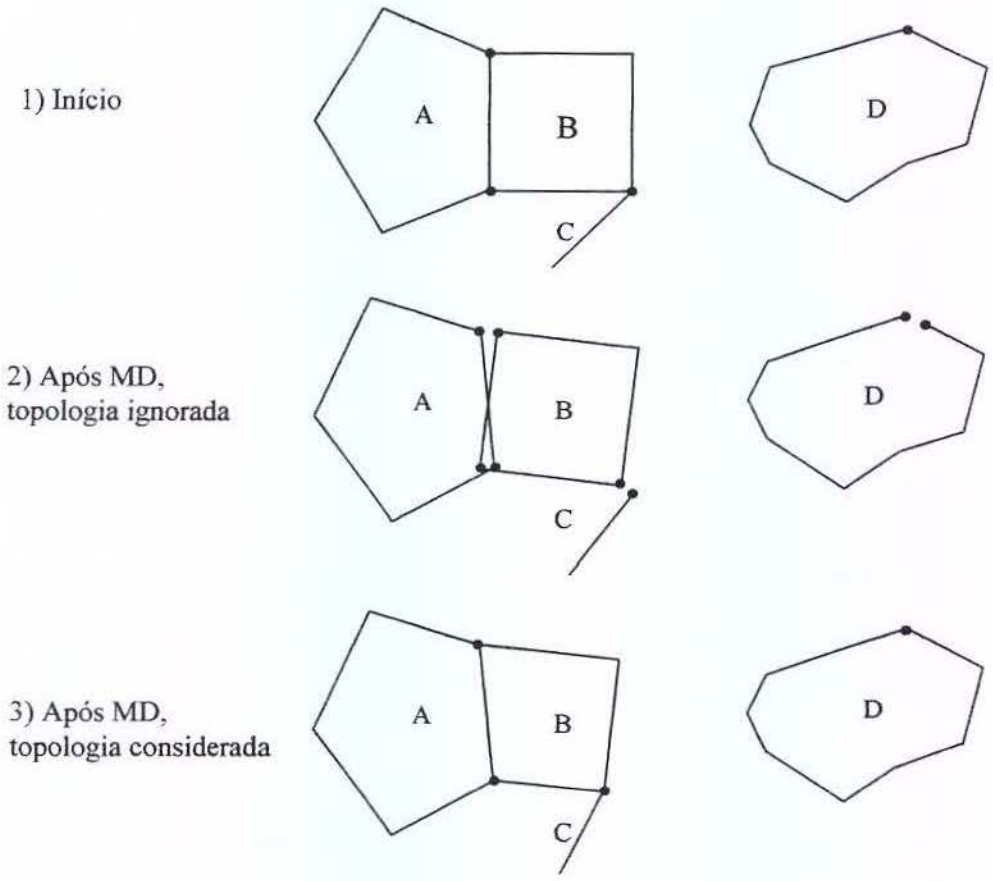


Figura 4.4 – Exemplos de inserção de MD levando ou não em conta a topologia.

A Figura 4.4 mostra alguns exemplos do tipo de relação topológica que trataremos. No primeiro caso, as entidades *A* e *B* compartilham inicialmente duas coordenadas, as entidades *B* e *C* compartilham uma só coordenada e a entidade *D* possui coordenadas inicial e final idênticas. O segundo caso mostra o que pode ocorrer caso as coordenadas sejam tratadas independentemente: coordenadas correspondentes podem ser alteradas de forma diferente ao inserir a MD. O terceiro caso ilustra a inserção da MD através de alterações iguais em coordenadas correspondentes.

Para levar em consideração essas relações topológicas, podemos considerar $v(M)$, criada através do Algoritmo 4.1, como sendo uma lista de coordenadas distintas, às quais as entidades originais se referem. Sempre que duas coordenadas de entidades de M forem consideradas idênticas, dentro de certa tolerância, elas farão referência às mesmas coordenadas de $v(M)$. Utilizando esta abordagem, garantimos que a alteração de uma coordenada se refletirá em todas as ocorrências daquela coordenada nas entidades de M. Ao gerar o mapa $\hat{M}$, cada entidade de M é recriada, mas considerando as coordenadas de $v(M)$. A Figura 4.5 ilustra a relação entre as coordenadas de $v(M)$ e as entidades de M.

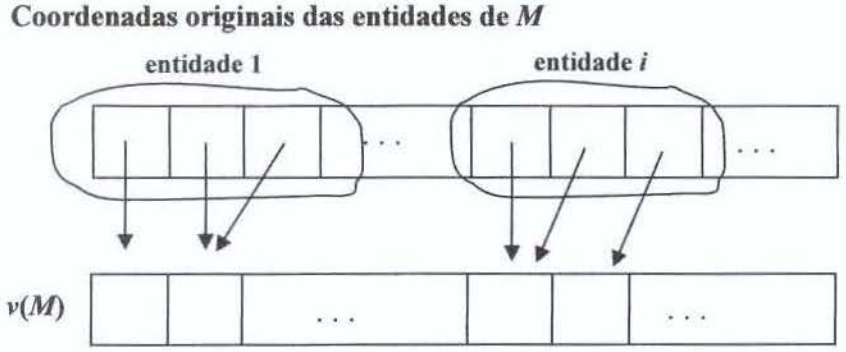


Figura 4.5 – $v(M)$ como uma lista de coordenadas distintas das entidades de M.

Para encontrar uma determinada coordenada na lista, pode-se utilizar a divisão da área do mapa em uma *quadtree* ou mesmo utilizar uma tabela *hash* de coordenadas para tornar as buscas mais eficientes.

4.1.4 Preparação da Imagem *Raster*

Seja **R** uma imagem *raster* com l linhas e c colunas a ser inserida como marca d'água no mapa M. Chamemos **R** de MD original, que é representada por uma matriz de pixels, em que o elemento r_{ij} é um número inteiro que representa a intensidade de cinza do pixel na linha i e coluna j. Vamos assumir que as intensidades dos pixels correspondam a valores de 0 a 255 (escala de cinza). Deve-se produzir a partir de **R** uma matriz quadrada de pixels **E**, a que chamaremos de **marca d'água estendida**, com n linhas e n colunas, onde n é o mesmo da equação (1). A imagem **E** pode ser produzida a partir da repetição de **R**, de acordo com o

Algoritmo 4.2 a seguir e com a Figura 4.6. Sejam I_{min} e I_{max} respectivamente os pixels de maior e menor intensidade de **R**. Podemos utilizar $I_m = (I_{min} + I_{max})/2$ para normalizar as intensidades dos pixels de **R** ao gerar **E**. Um determinado pixel r_{ij} de **R** apareceria em **E** em sua devida posição (Algoritmo 4.2), mas com o valor $r_{ij} - I_m$. Isto fará com que os deslocamentos das coordenadas sejam reduzidos, de modo que o deslocamento de maior amplitude seja $\pm(I_{max} - I_m)$. Uma alternativa seria escolher o valor de I_m como sendo o valor da intensidade que mais se repete na imagem **R**, neste caso contribuindo para que um menor número de pontos do mapa sofra deslocamento.

Entrada: $n = \lfloor \sqrt{t} \rfloor$ (t é o número do pontos de $v(M)$)
R (marca d'água, matriz $l \times c$)
E (marca d'água estendida, matriz $n \times n$ vazia)
INÍCIO
Sejam I_{max} e I_{min} o pixel de maior e o de menor intensidade de **R**, respectivamente. Seja $I_m = (I_{max} + I_{min})/2$
Para u variando de 1 até n
  Para v variando de 1 até n
    $i = [(u-1)\%l]+1$ (onde $a\%b$ é o resto da divisão de a por b)
    $j = [(v-1)\%c]+1$
    $e_{uv} = r_{ij} - I_m$, onde r_{ij} é o pixel de **R** na linha i e coluna j;
    (e_{uv} é o pixel de **E** na linha u e coluna v)
Retornar a imagem **E**.
FIM.

Algoritmo 4.2 – Criação de *E*, a marca d'água estendida.

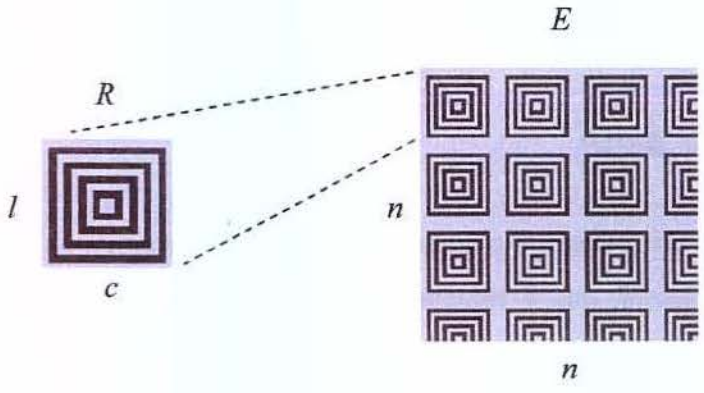


Figura 4.6 - Exemplo de preparação da marca d'água estendida.

A imagem **E**, obtida aplicando-se o Algoritmo 4.2 em **R**, corresponderá a múltiplas ocorrências de **R** (ou de parte de **R**), caso o número de linhas ou colunas de **R** seja menor que n, ou será

somente uma sub-região de **R**, caso o número de linhas e colunas de **R** seja maior que n. A Figura 4.6 ilustra a aplicação do algoritmo. A escolha de uma imagem **R** que se repita diversas vezes em **E** é muito interessante, pois insere redundância de informações, o que aumenta a resistência a ataques, como será discutido oportunamente. A carga de informações, ou *payload*, a ser inserida no mapa é limitada por n, que por sua vez é uma função de t, o número de pontos em $v(M)$.

4.1.5 Inserção da Imagem na Representação Matricial do Mapa

Inicialmente, seja $w(\mathbf{A})$ uma função que, aplicada a uma matriz quadrada **A** de elementos a_{ij} com n linhas e n colunas, produz uma matriz $\hat{\mathbf{A}}$ também $n \times n$, de elementos $\hat{a}_{ij} = a_{uv}$, onde $u = n - i + 1$ e $v = n - j + 1$.

$$\mathbf{A} = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix} \Rightarrow w(\mathbf{A}) = \begin{pmatrix} 16 & 15 & 14 & 13 \\ 12 & 11 & 10 & 9 \\ 8 & 7 & 6 & 5 \\ 4 & 3 & 2 & 1 \end{pmatrix}$$

Figura 4.7 – Exemplo de aplicação da função *w(X)* [*O(t)*]

Em outras palavras, $w(\mathbf{A})$ é a matriz que se obtém ao percorrer **A** no sentido anti-raster. A Figura 4.7 exemplifica a ordem dos elementos em **A** e os correspondentes em $w(\mathbf{A})$. Temos como resultado direto que $w(w(\mathbf{A}))=\mathbf{A}$.

De posse da matriz **E** ($n \times n$), e das matrizes $\mathbf{A_x}$ e $\mathbf{A_y}$, definidas nas seções anteriores, podemos produzir duas matrizes $\mathbf{B_x}$ e $\mathbf{B_y}$ conforme a equação (4) seguir:

$$\mathbf{B}_x = C\mathbf{E} + \mathbf{A}_x \qquad \mathbf{B}_y = Cw(\mathbf{E}) + \mathbf{A}_y \qquad (4)$$

onde C é uma constante real positiva. Sendo $(B_x)_{ij}$ e $(B_y)_{ij}$ elementos da linha i e coluna j de $\mathbf{B_x}$ e $\mathbf{B_y}$, respectivamente, temos que os pares $[(B_x)_{ij}, (B_y)_{ij}]$ correspondem às coordenadas originais de $v(M)$ já com as informações da marca d'água estendida **E** inseridas.

O Algoritmo 4.3 mostra como deve ser feita a inserção da MD estendida no mapa utilizando equação (4). Através do mapeamento entre as coordenadas de $v(M)$ (pares $[(A_x)_{ij}, (A_y)_{ij}]$) e as entidades que as originaram em M, o que foi necessário no Algoritmo 4.1, podemos recriar estas entidades utilizando como novas coordenadas os pares $[(B_x)_{ij}, (B_y)_{ij}]$. Vamos chamar o novo mapa criado desta maneira de mapa marcado $\hat{M}$, que já possui a MD e já pode ser distribuído. Neste caso, as matrizes $\mathbf{B_x}$ e $\mathbf{B_y}$ correspondem à representação matricial dos pontos de $\hat{M}$. A seqüência de pares correspondentes $[(B_x)_{ij}, (B_y)_{ij}]$, quando as matrizes são percorridas primeiro pelas linhas e depois pelas colunas, são as coordenadas da representação por pontos de $\hat{M}$, ou seja, $v(\hat{M})$.

```
Entrada: mapa vetorial M, imagem raster R, parâmetro C
INÍCIO
   Obter v(M) através do Algoritmo 4.1;  [O(t)]
   Obter E através do Algoritmo 4.2; [O(t)]
   Obter w(E); [O(t)]
   Seja t o número de pontos em v(M).  Seja n = ⌊√t⌋ .  Seja L uma lista
   vazia de coordenadas;
   Para cada i variando de 1 até n [O(t)]
          Para cada j variando de 1 até n
              Seja k = n × (i-1) + j
              Se k ≤ t, ou seja, existe o k-ésimo ponto em v(M)
                  Seja p o k-ésimo ponto de v(M)
                  (Ax)ij = px , (Ay)ij = py
   Bx = C × E + Ax;     [O(t)]
   By = C × w(E) + Ay; [O(t)]
   Para i variando de 1 até n  [O(t)]
          Para j variando de 1 até n
              Seja k = n × (i-1) + j
              Se k ≤ t, ou seja, existe o k-ésimo ponto em v(M)
                    adicionar o par [(Bx)ij, (By)ij] à lista L
   Criar um novo mapa vetorial M̂ como uma cópia de M; [O(t)]
   Para cada coordenada l de L, refletir seu valor na entidade
   correspondente de M̂;  [O(t)]
   Ao final, retornar M̂.
FIM
```

Algoritmo 4.3 – Inserção da MD visual em um mapa vetorial.

A utilização de $w(\mathbf{E})$ na matriz **By** permite que as duas ocorrências do mesmo pixel fiquem armazenadas em coordenadas diferentes, aumentando a resistência a ataques, como será discutido. Também poderíamos definir $w(\mathbf{E})$ como uma função que espalha aleatoriamente os

pixels de **E**, porém teríamos que armazenar o mapeamento feito de alguma forma para obter **E** a partir de $w(\mathbf{E})$, o que será necessário mais tarde na detecção da MD.

4.2 Algoritmo para Detecção da Marca d'Água

Dado um mapa N a ser testado, a que chamaremos de **mapa de prova**, queremos identificar nele uma imagem **S** chamada de **imagem de prova**. A comparação de **S** com a MD original R poderá indicar se N corresponde ao mapa marcado $\hat{M}$. Para a detecção da MD em N, é necessário possuir o mapa original M, o valor da constante C utilizada na inserção e a imagem original **R** (para comparação).

O primeiro passo é a extração, conforme descrita na Seção 4.1.1, da representação por pontos de $\hat{M}$ e N, respectivamente o **mapa marcado preparado** $P = v(\hat{M})$ e o **mapa de prova preparado** $Q = v(N)$, respectivamente com t e t' coordenadas. De posse dessas representações, P deve ser comparado com Q através de um algoritmo de *point pattern matching*, de forma que eventuais transformações que tenham sido aplicadas a P para produzir Q sejam identificadas e removidas, fazendo então a correspondência entre coordenadas de P e de Q. De fato, por definição, o i-ésimo ponto do mapa original preparado $v(M)$ e de P se correspondem, sendo que o segundo equivale ao primeiro deslocado pela MD. Da mesma forma, o i-ésimo ponto de $v(M)$ precisa corresponder a um j-ésimo de Q, não necessariamente para todo i ou todo j, para que, por fim, seja possível calcular uma imagem detectada de **S** diretamente utilizando a equação (4) e, através dela, verificar se corresponde à MD original **R**.

4.2.1 Point Pattern Matching

O mapa N pode ter sido produzido a partir de uma transformação sobre o mapa $\hat{M}$. É necessário então descobrir tal transformação e removê-la, aplicando sua transformação inversa ao mapa de prova preparado Q. Depois disso, teremos um conjunto de pontos de Q que terão correspondência com o mapa marcado preparado P. A importância de ser ter uma

correspondência entre os pontos de P e de Q está no fato de o método realizar operações sobre pontos correspondentes no mapa original preparado $v(M)$ e no mapa de prova preparado $v(N)$, para produzir os pixels de uma imagem. De fato, por definição, o i-ésimo ponto de $v(M)$ e de P se correspondem. Da mesma forma, o i-ésimo ponto de $v(M)$ precisa ter correspondência com um j-ésimo de Q, não necessariamente para todo i ou todo j. Quando algum ponto não possuir correspondência, o pixel relativo àquela posição não possuirá informação útil na imagem resultante. Exemplos de métodos que também necessitam de uma correspondência entre os pontos do mapa original e do mapa marcado são tratados por *Ohbuchi et al.* em [Ohbuchi+02] e [Ohbuchi+03].

O problema de *point pattern matching* (PPM) consiste em verificar se existe uma transformação que, ao ser aplicada a um certo conjunto de pontos, melhor o aproxima de um segundo conjunto de pontos. Em [Van Wamelen+99] o problema é descrito da seguinte maneira:

> [...] Sejam dois conjuntos de pontos P e Q, com $P = \{p_1, p_2, \ldots p_n\}$ e $Q = \{q_1, q_2, \ldots q_n\}$, onde p_i e q_j são pontos em $\mathbb{R}^2$. Vamos assumir que P e Q têm aproximadamente o mesmo número de pontos. Queremos encontrar uma transformação $T_{s,\theta,tx,ty}$ tal que $T(P)$ **corresponde**[10] a Q, onde tal correspondência será definida abaixo. Na transformação $T_{s,\theta,tx,ty}$, temos que s é um fator de escala, θ é um ângulo de rotação e t_x e t_y são translações ao longo dos eixos X e Y respectivamente. Isto é, para $(x,y) \in \mathbb{R}^2$, temos:
>
> $$T\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} t_x \\ t_y \end{pmatrix} + s\begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} \qquad (5)$$
>
> Para definir uma **correspondência**, assumimos que dois parâmetros são dados: a probabilidade de correspondência $\rho \in [0,1]$ e o tamanho da correspondência $t \in \mathbb{R}^+$. Dizemos então que $T_{s,\theta,tx,ty}(P)$ corresponde a Q se existe um subconjunto $P' \subset P$ com no mínimo ρt elementos (onde t é o número de pontos de P) tais que para cada $p \in P'$ temos $| T_{s,\theta,tx,ty}(p) - q| < \tau$ para algum $q \in Q$. [...] Estamos portanto assumindo que Q é uma transformação de similaridade de P distorcida localmente, com pontos extras e/ou faltantes. Nosso problema é recuperar a transformação a partir dos dois conjuntos de pontos. [...]

[10] do original: *matches*

No método RAWVec, é necessário implementar um algoritmo de PPM para remover eventuais transformações aplicadas a Q de modo a se obter uma correspondência entre os dois mapas. A Figura 4.8 ilustra a idéia básica de um algoritmo de PPM.

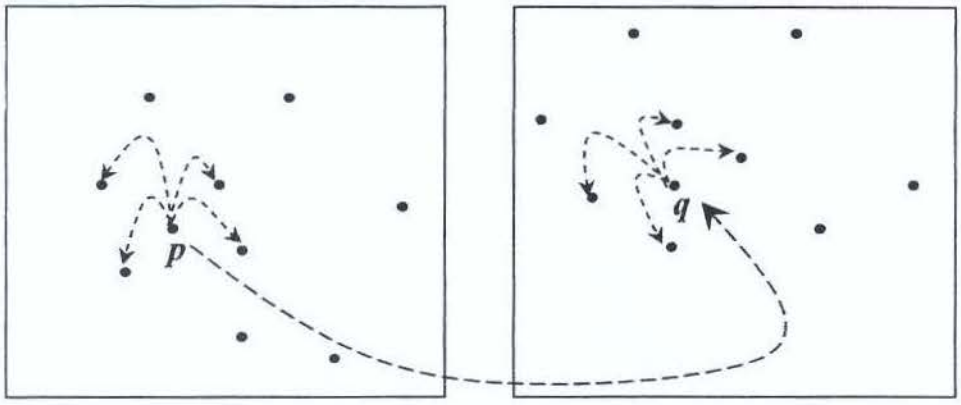


Figura 4.8 – Exemplo de *Point Pattern Matching*.

Utilizamos o Algoritmo 4.4, proposto por *Wamelen et al.* [Van Wamelen+99], que define parâmetros que permitem controlar a tolerância de posicionamento e o número mínimo de pontos que devem possuir correspondência nos conjuntos sendo comparados. Esse algoritmo faz a correspondência entre dois conjuntos de pontos P e Q e recebe os parâmetros κ, ρ e τ, onde $\boldsymbol{\kappa}$ é o número de vizinhos a ser considerado para cada ponto; $\boldsymbol{\rho}$ é a probabilidade do casamento entre P e Q, ou seja, o percentual mínimo de pontos dos dois conjuntos que devem casar; $\boldsymbol{\tau}$ é a distância máxima entre dois pontos para que sejam considerados correspondentes. Esses e outros parâmetros serão discutidos na Seção 4.2.2.

Laço Principal
Entrada: conjuntos de pontos P e Q, parâmetros κ, ρ e τ
1) **Preparação**
   a. Para cada ponto em P e Q, encontre os KNN, em ordem de proximidade;
   b. Sendo r o raio do menor círculo que envolve todas as coordenadas de Q e t' o número de pontos em Q, crie uma matriz bidimensional em que cada posição representa um quadrado no mapa de lado $r/\sqrt{t'}$. Cada posição deve possuir a lista das coordenadas que se encontram no quadrado correspondente. Esta matriz permitirá verificar rapidamente se existe um ponto em Q em uma dada coordenada.
2) **Busca da Transformação Global**
   Sejam t o número de pontos em P e t' o número em Q.
   Enquanto não for encontrado um casamento global
      Para cada i variando de 1 até t e j variando de 1 até t'
         Determinar se os KNN de p_i casam com os KNN de q_j, através do Algoritmo 4.8. Seja T a transformação sob a qual há o casamento. Se T puder ser refinada para um casamento global, retornar T e os pares de pontos de P e Q que se casam sob T.

Algoritmo 4.4 – Laço principal do algoritmo de PPM ([Van Wamelen+99]).

O método necessita da obtenção dos KNN (*K Nearest Neighbors*, ou K vizinhos mais próximos) de cada coordenada de um conjunto de pontos, o que é necessário como preparação para o Algoritmo 4.4. Para a obtenção dos KNN, inicialmente criamos uma **Triangulação de Delaunay** (TD) para o conjunto. *Boissonat* e *Teillaud* fonecem um algoritmo para cálculo da TD a partir de uma Árvore de Delaunay em $O(n \log n)$ em duas dimensões [Boissonnat+93].

Após a obtenção da TD, podemos aplicar o Algoritmo 4.5 para cada coordenada do conjunto. Esse algoritmo, listado em [CGAL 3.1], recebe um ponto p que faz parte da TD e retorna seus K vizinhos mais próximos em uma lista, em ordem crescente de sua distância até p. Utiliza uma fila de prioridades (*heap* [Williams64]), percorrendo a TD em largura a partir do ponto p. A prioridade para cada ponto é dada pelo valor negativo de sua distância até o ponto p.

```
Entrada: ponto p, Triangulação de Delaunay T, número de vizinhos K.
Auxiliares:
 . H, fila de prioridades (heap [Williams64]) vazia, em que cada item
será um ponto e sua prioridade será o oposto de sua distância até p.
 . Push(H, A), insere o ponto A na fila de prioridades H.
 . Pop(H), retira e retorna o primeiro da fila de prioridades H.
 . L, uma lista vazia de pontos.
INÍCIO
    Se o número de pontos em T for menor ou igual a K,
        Para cada ponto q de T, fazer push(H, q)
        Enquanto H não estiver vazia, inserir pop(H) em L
    Se o número de pontos em T for maior que K,
        Fazer push(H, p) e marcar p
        Enquanto K >= 0
            Sejam K = K - 1 e w = pop(H)
            Se w ≠ p, inserir w em L
            Para cada vizinho imediato v de w em T
                Se v não estiver marcado, marcar v e fazer push(H, v)
   Retornar L, que conterá os KNN de p em ordem crescente da distância
FIM
```

Algoritmo 4.5 – Obtenção dos KNN de um ponto, em ordem decrescente de proximidade, a partir de uma Triangulação de Delaunay ([CGAL 3.1]).

Ainda na fase de preparação do Algoritmo 4.4, os autores utilizam-se de uma matriz bidimensional em que cada posição representa no mapa um quadrado de lado $r/\sqrt{t'}$, onde r é o raio do menor círculo que contém todos os pontos de Q e t' é o número de pontos de Q. Cada posição da matriz deve guardar uma lista dos pontos de Q que estão contidos no quadrado correspondente.

O Algoritmo 4.6 mostra como verificar se há um ponto em Q próximo de um ponto p. Nele, dividem-se as coordenadas de p (p_x e p_y) pela largura do quadrado que representa cada posição da matriz ($r/\sqrt{t'}$), obtendo respectivamente i e j, que é uma posição, ou quadrado, da matriz. Verifica-se então se algum ponto dessa posição ou dos oito quadrados vizinhos, quando existirem, estão a uma distância de p menor que o valor de um parâmetro τ.

```
Entrada:
Uma matriz L de pontos descrita na preparação do Algoritmo 4.4.
O ponto p.  A distância τ a ser considerada
INÍCIO
  Seja l a largura de cada quadrado da matriz
  Seja u = p_x/l e v=p_y/l
  Para i variando de u-1 até u+1
      Para j variando de v-1 até v+1
          Se existir o quadrado L_ij e não estiver vazio
             Se algum ponto do quadrado L_ij está a uma distância menor que
             τ de p, retornar tal ponto
FIM
```

Algoritmo 4.6 – Verificando se há um ponto próximo de uma coordenada ([Van Wamelen+99]).

O Algoritmo 4.7 mostra como obter o ponto de Q mais próximo de um ponto p qualquer, caso exista. Utiliza a mesma idéia do Algoritmo 4.6, exceto que agora será procurado não um ponto qualquer, mas o ponto mais próximo de p.

```
Entrada:
Uma matriz L de pontos descrita na preparação do Algoritmo 4.4.
O ponto p
A distância τ a ser considerada
INÍCIO
  Seja d = τ, s = vazio, l = largura de cada quadrado da matriz
  Seja u = p_x/l e v=p_y/l
  Para i variando de u-1 até u+1
      Para j variando de v-1 até v+1
          Se existir o quadrado L_ij e não estiver vazio
             Se p está a uma distância menor que d das bordas do quadrado
             L_ij
                  Para cada ponto q do quadrado L_ij
                     Se q está a uma distância menor que d de p
                        s = q
                        d = distância de p até q
  Retornar s
FIM
```

Algoritmo 4.7 – Localizando o ponto mais próximo de uma coordenada ([Van Wamelen+99] modificado).

No Algoritmo 4.4, é necessário percorrer os pontos de P e de Q em busca de uma transformação T local que leva $p \in P$ até $q \in Q$ e um subconjunto dos vizinhos de P até os correspondentes em Q. De acordo com *Wamelen et al.* [Van Wamelen+99], a transformação única que leva dois pontos p e a de P até os pontos q e b de Q é dada pela equação (6) a seguir:

$$s = \frac{|\overrightarrow{qb}|}{|\overrightarrow{pa}|}$$

θ = ângulo de $\overrightarrow{pa}$ até $\overrightarrow{qb}$ (6)

$$t_x = q_x - p_x s\ cos(\theta) + p_y s\ sin(\theta)$$

$$t_y = q_y - p_x s\ sin(\theta) - p_y s\ cos(\theta)$$

O Algoritmo 4.8 mostra como uma transformação local T pode ser obtida. Dados um ponto p de P e um ponto q de Q, o algoritmo escolhe um vizinho de p e um de q que estejam tão distantes quanto possível de p e de q, respectivamente. Calcula-se então a transformação que leva p e seu vizinho até q e seu vizinho e verifica-se, com base nessa transformação, se um número suficiente de vizinhos de P e Q se casam. De acordo com *Wamelen et al.* [Van Wamelen+99]:

> [...] assuma que $\{a_1, a_2, ..., a_k\}$ são os KNN ordenados do ponto p de P (com a_1 o mais próximo, etc.) e $\{b_1, b_2, ..., b_k\}$ são os KNN ordenados do ponto q de Q. Então, para cada um dos k_2 pontos $a_{k-[k_3/2]-k_2+1}$ até $a_{k-[k_3/2]}$ calculamos a transformação de similaridade T que leva p até q e $a_{k-[k_3/2]-i}$, por sua vez, até cada um dos k_3 pontos b mais próximos de $b_{k-[k_3/2]-i}$ (isto é, b_{k-k_3+1-i} até b_{k-i}). Para cada uma dessas k_2 k_3 transformações T, verificamos se elas nos dão mais de $\rho(k\text{-}1)$ casamentos adicionais de vizinhos. Para cada T que satisfaça essa condição, verificamos se T pode ser refinada em uma transformação global. [...]

O mesmo algoritmo utiliza os parâmetros k_2 e k_3, as janelas de busca por pontos vizinhos, cujos valores recomendados por *Wamelen et al.* [Van Wamelen+99] são:

$$k_2 = \frac{\ln 0.05}{\ln(1-\rho)} \quad (7) \qquad k_3 = \frac{2\lambda\sqrt{k}}{\sqrt{\pi}} \quad (8)$$

Nas equações (7) e (8), λ é o *matching factor* e ρ é a probabilidade de casamento dos pontos de P e Q. Esses e outros parâmetros serão definidos na Seção 4.2.2.

```
Entrada:
Parâmetros k, k2 e k3
Sejam {a1,a2,..., ak} os k vizinhos mais próximos do ponto p em P.  Sejam
{b1,b2,..., bk} os k vizinhos mais próximos do ponto q em Q.
INÍCIO
  Para i variando de 0 até k2-1
      Para j variando de 0 até k3-1
          Seja La = {p} e Lb = {q}
          Seja u = k-i-k3/2
          Seja v = k-i-j
          Seja T a transformação única que leva p até q e au até bv
          Esta transformação é dada pela equação ( 6 )
          Para l variando de 1 até k
              Se existe um ponto q' em Q cuja distância até T(al) seja
              menor que t, inserir al em La e q' em Lb
          Se (número de pontos em La)-2 for maior que ρ(k-1),
          Então T é uma transformação local candidata.
              Se aplicando o Algoritmo 4.9 esta transformação local pode
              ser refinada para uma transformação global, retornar T, La
              e Lb.
FIM
```

Algoritmo 4.8 – Obtendo uma transformação candidata ([Van Wamelen+99]).

A Figura 4.9 ilustra a utilização dos KNN no algoritmo de PPM.

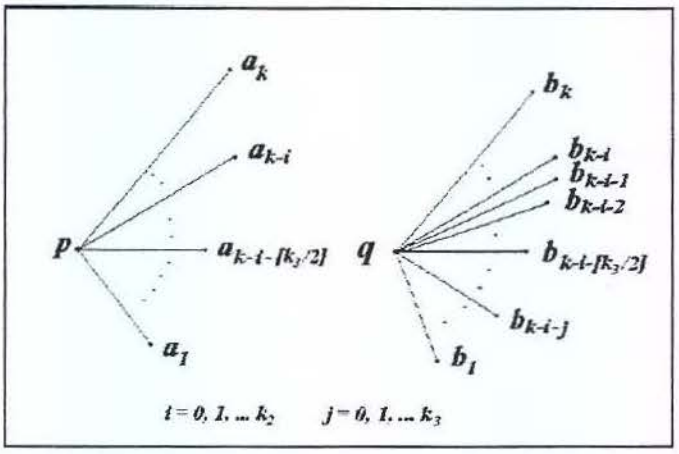


Figura 4.9 – Utilização dos KNN no PPM ([Van Wamelen+99])

A transformação local candidata T é calculada a partir da lista de vizinhos de um determinado ponto a de P e seu correspondente b em Q. O resultado são duas listas de pontos correspondentes: $\mathbf{L_a}$ de a e $\mathbf{L_b}$ de b. É verificado, então, se T se aplica globalmente, pelo Algoritmo 4.9. Para isso, é calculada a melhor transformação T que aplicada a $\mathbf{L_a}$ produz $\mathbf{L_b}$.

Sejam $\mathbf{L_{ax}}$ e $\mathbf{L_{ay}}$, respectivamente, as listas contendo as abscissas e ordenadas de $\mathbf{L_a}$ e sejam $\mathbf{L_{bx}}$ e $\mathbf{L_{by}}$, respectivamente, as listas contendo as abscissas e ordenadas de $\mathbf{L_b}$. Então, de acordo com *Wamelen et al.* [Van Wamelen+99], $\overline{T}$ pode ser calculada pela equação (17) a seguir, onde t_x e t_y são a translação, s é a escala e θ a rotação.

$$\mu_{ax} = \sum_{i=1}^{l} (L_{ax})_i \quad (9) \qquad \mu_{bx} = \sum_{i=1}^{l} (L_{bx})_i \quad (10)$$

$$\mu_{ay} = \sum_{i=1}^{l} (L_{ay})_i \quad (11) \qquad \mu_{by} = \sum_{i=1}^{l} (L_{by})_i \quad (12)$$

$$\mu_{a+b} = \sum_{i=1}^{l} (L_{ax})_i (L_{bx})_i + (L_{ay})_i (L_{by})_i \quad (13) \qquad \mu_{a-b} = \sum_{i=1}^{l} (L_{ax})_i (L_{by})_i - (L_{ay})_i (L_{bx})_i \quad (14)$$

$$\mu_a = \sum_{i=1}^{l} (L_{ax})_i^2 (L_{ay})_i^2 \quad (15)$$

$$\mu_d = l\mu_a - \mu_{ax}^2 - \mu_{ay}^2 \quad (16)$$

$$\overline{T} = \begin{pmatrix} t_x \\ t_y \\ s\cos\theta \\ s\sin\theta \end{pmatrix} = \frac{1}{\mu_d} \begin{pmatrix} \mu_a & 0 & -\mu_{ax} & \mu_{ay} \\ 0 & \mu_a & -\mu_{ay} & \mu_{ax} \\ -\mu_{ax} & -\mu_{ay} & l & 0 \\ \mu_{ay} & -\mu_{ax} & 0 & l \end{pmatrix} \begin{pmatrix} \mu_{bx} \\ \mu_{by} \\ \mu_{a+b} \\ \mu_{a-b} \end{pmatrix} \quad (17)$$

No Algoritmo 4.9, a equação (17) é aplicada inicialmente em $\mathbf{L_a}$ e $\mathbf{L_b}$. Para a transformação $\overline{T}$ encontrada, percorrem-se todos os pontos de P e Q em busca de pontos que se correspondam sob $\overline{T}$, produzindo novos em $\mathbf{L_a}$ e $\mathbf{L_b}$. Se o número mínimo de pontos foi encontrado, então o algoritmo atingiu seu objetivo. Senão, calcula-se um novo $\overline{T}$ com as novas listas e, com ele, geram-se novas listas $\mathbf{L_a}$ e $\mathbf{L_b}$, reiniciando o processo. O algoritmo termina sem sucesso quando não há progresso em uma dada iteração, ou seja, quando o número de pontos que se casam em duas iterações consecutivas não aumenta e o número mínimo de pontos não foi atingido.

```
Entrada:
As duas listas La e Lb de pontos que se casam sob T inicial.
Saída: Nova transformação T, listas de pontos correspondentes La e Lb.
INÍCIO
  Repita até o número de pontos correspondentes ser menor que pt
      Calcular uma nova transformação T a partir dos pontos
correspondentes de La e Lb através da equação ( 17 )
      Seja La = {} e Lb = {}
      Desmarque todos os pontos de Q
      Para i variando de 1 até t
         Se o Algoritmo 4.6 detectar algum ponto a uma distância menor
         que τ de T(pi) em Q
              Seja q ∈ Q o ponto mais próximo de T(pi) retornado pelo
              Algoritmo 4.7
              Se q não estiver marcado
                   Inserir pi em La e q em Lb
                   Marcar q
      Se La não aumentou de tamanho na última iteração
           Então T não se aplica globalmente
  Retornar T, La e Lb
FIM
```

Algoritmo 4.9 – Obtendo uma transformação global ([Van Wamelen+99]).

Os resultados do algoritmo são:

- a transformação T que, ao ser aplicada a Q, produz o mapa P com certa probabilidade, que é um dos parâmetros de entrada;

- as listas $\mathbf{L_a}$ e $\mathbf{L_b}$ de pares de pontos de P e de Q correspondentes.

A partir do resultado, aplica-se a transformação a Q, obtendo-se um novo Q que já pode ser utilizado na detecção da marca d'água.

4.2.2 Parâmetros do Algoritmo de Point Pattern Matching

O algoritmo de PPM utilizado possui alguns parâmetros que têm efeito na eficiência do algoritmo e em suas chances encontrar a transformação procurada. São eles:

- k, o número de vizinhos utilizados no cálculo dos KNN. O valor recomendado pelos autores para este parâmetro é $k \leq \frac{\ln t}{2(\rho - \lambda^2/4)^2}$. Porém, pode haver casos em que são necessários valores maiores de k. Isso porque k será o número de pontos utilizados originalmente na equação (17) para o cálculo de $\overline{T}$ na primeira iteração do Algoritmo 4.9. Para um valor pequeno de k, a transformação será calculada com base em um número pequeno de pontos, em uma área pequena do mapa, podendo ser bem diferente da melhor transformação, e o algoritmo pode não convergir.

- ρ, (onde $0 < \rho \leq 1$) a probabilidade de casamento entre P e Q. É o percentual mínimo de pontos que devem se corresponder. Nos casos em que P e Q têm quantidades diferentes de pontos, ρ será aplicada ao conjunto com o menor número de pontos. O número mínimo de pontos que devem ter correspondência no PPM é ρt.

- t é o número de pontos a ser considerado. De fato, t é um parâmetro obtido diretamente dos conjuntos P e Q e será o número de elementos do conjunto com o menor número de pontos.

- τ, a distância máxima a ser considerada na verificação da correspondência entre dois pontos. Dois pontos p e q se casam sob a transformação T caso $|T(p) - q| < \tau$. O valor recomendado pelos autores para τ é $\tau = \lambda \frac{r}{2\sqrt{t'}}$, onde r é o raio do menor círculo que contém todos os pontos de Q, t' é o número de pontos de Q e o termo λ é o **matching factor**, utilizado para controlar a amplitude do raio de τ e, dessa forma, controlar a tolerância na comparação entre dois pontos em busca de uma correspondência. De fato, podem ser necessários valores maiores que o recomendado para τ a fim de aumentar a tolerância do algoritmo a perturbações nas coordenadas.

4.2.3 Modificações no Algoritmo de Point Pattern Matching

O algoritmo de PPM proposto por *Wamelen et al.* [Van Wamelen+99], apesar de sua eficiência, pode-se tornar inviável para um número muito grande de pontos, principalmente no seu pior caso, em que os mapas não têm correspondência. Para cada um dos pontos de P, é necessário localizar um ponto em Q que possua alguns vizinhos que correspondam a vizinhos de P. Para cada um desses casos, se houver um casamento local, será necessário percorrer todos os pontos de P e Q novamente para verificar se há um casamento global.

É possível auxiliar o algoritmo utilizando uma abordagem parecida com a proposta em [Ohbuchi+03], em que o PPM se inicia através da escolha manual de um conjunto de entidades correspondentes nos dois mapas. Propomos passar como parâmetro ao algoritmo uma ou mais coordenadas de P e sua(s) correspondente(s) em Q. O algoritmo localiza então tais pontos nos dois mapas, calcula uma transformação inicial a partir de seus vizinhos e tenta encontrar uma transformação global a partir daí. Dessa forma, diminui-se o espaço de busca pela transformação local que leva alguns pontos de P até os correspondentes de Q, restando refiná-la em busca de uma transformação global. O Algoritmo 4.4 poderia então ser substituído pelo Algoritmo 4.10 a seguir.

Laço Principal
Entrada: conjuntos de pontos P e Q, ponto p de P e seu correspondente q de Q, parâmetros κ, ρ e τ
1) **Preparação**
   Como no Algoritmo 4.4.
2) **Busca da Transformação Global**
   Determinar se os KNN de p casam com os KNN de q, através do Algoritmo 4.8. Seja T a transformação sob a qual há o casamento. Se T puder ser refinada para um casamento global, retornar T e os pares de pontos de P e Q que se casam sob T.

Algoritmo 4.10 – Primeira alteração do Algoritmo 4.4 ([Van Wamelen+99], modificado).

Podemos ampliar esta abordagem e não utilizar vizinhos no cálculo da transformação local. Pares de pontos correspondentes podem ser identificados nos dois mapas e a transformação local pode ser calculada a partir deles. Neste caso, não é necessário o cálculo dos KNN para cada ponto na fase de preparação do Algoritmo 4.4. Esta alteração é mostrada no Algoritmo 4.11.

```
Laço Principal
Entrada: conjuntos de pontos P e Q, pontos p1 e p2 de P e seus
correspondentes q1 e q2 de Q, parâmetros ρ e τ
1) Preparação
      Somente o item b do Algoritmo 4.4.

2) Busca da Transformação Global
      Seja T a transformação que leva p1 e p2 até q1 e q2. Se T puder ser
      refinada para um casamento global, retornar T e os pares de pontos
      de P e Q que se casam sob T.
```

Algoritmo 4.11 – Segunda alteração do Algoritmo 4.4 ([Van Wamelen+99], modificado).

Outra alteração interessante é não parar o Algoritmo 4.9 ao encontrar o número mínimo de pontos necessários ρt. Poderíamos continuar com as iterações enquanto haja progresso, ou seja, enquanto novos pontos puderem ser acrescentados nas listas a cada refinamento da transformação. Isso permite a obtenção de um número maior de pontos correspondentes, o que se reflete em mais pixels úteis na imagem detectada em Q, ao invés de somente o mínimo necessário, indicado pela probabilidade ρ. Esta alteração aparece no Algoritmo 4.12.

```
Entrada:
As duas listas La e Lb de pontos que se casam sob T inicial.
INÍCIO
  Seja F = 0
  Repita até que F = 1
      Calcular uma nova transformação T a partir dos pontos
correspondentes de La e Lb através da equação ( 17 )
      Seja La = {} e Lb = {}
      Desmarque todos os pontos de Q
      Para i=1 até t
          Se o Algoritmo 4.6 detectar algum ponto a uma distância menor
          que τ de T(pi) em Q
                Seja q ∈ Q o ponto mais próximo de T(pi) retornado pelo
                Algoritmo 4.7
                Se q não estiver marcado
                     Inserir pi em La e inserir q em Lb
                     Marcar q
      Se La não aumentou de tamanho na última iteração, F = 1
  Se o número de pontos em La for menor que ρt, T não é transformação
        global
  Senão, retornar T, La e Lb
FIM
```

Algoritmo 4.12 – Alteração no Algoritmo 4.9 ([Van Wamelen+99] modificado).

4.2.4 Detecção da Marca d'Água

Uma vez tendo sido feita a correspondência entre os pontos de Q e P e tendo sido aplicada a Q a melhor transformação que o leva até P, devemos preparar as matrizes $\mathbf{A_x}$ e $\mathbf{A_y}$ a partir de $v(M)$ e as matrizes $\mathbf{B_x}$ e $\mathbf{B_y}$ a partir de Q, através da distribuição das coordenadas de seus pontos. Sabemos que, de acordo com o algoritmo de inserção da MD, o i-ésimo ponto de $v(M)$ corresponde ao i-ésimo ponto de $v(\hat{M})$. Visto que o valor de C utilizado na inserção da marca d'água é conhecido, a **imagem detectada D**, representada por uma matriz $n \times n$, pode ser obtida pelas equações (18) e (19).

$$\mathbf{D_x} = (\mathbf{B_x} - \mathbf{A_x}) / C \quad \text{e} \quad \mathbf{D_y} = w(\mathbf{B_y} - \mathbf{A_y}) / C \qquad (18)$$

$$\mathbf{D} = (\mathbf{D_x} + \mathbf{D_y}) / 2 \qquad (19)$$

As matrizes $\mathbf{B_x}$ e $\mathbf{B_y}$ devem ser preenchidas tendo como referência de posição dos elementos na equação (2) o conjunto de pontos P. As posições de $\mathbf{B_x}$ e $\mathbf{B_y}$ preenchidas são aquelas que possuem pontos de Q que têm correspondência com P no PPM. As posições sem correspondência recebem o valor zero. De fato, a imagem detectada **D** corresponderá à imagem estendida **E**, caso o mapa N corresponda a $\hat{M}$. O Algoritmo 4.13 obtém a imagem detectada **D** a partir de $v(\hat{M})$, a representação por pontos de $\hat{M}$. Também obtém a freqüência de repetição de cada pixel da imagem original **R** em **D**.

```
Entrada: v(M), P = v(M̂), L1 e L2 obtidas de P e Q, respectivamente, a
partir do algoritmo de PPM; c e l respectivamente o número de colunas e
linhas da imagem original R; Parâmetro C;

INÍCIO
   Seja t o número de pontos em P.  Seja n = ⌊√t⌋ ;
   Sejam Ax, Bx, Ay, By, Dx, Dy matrizes n × n nulas;
   Seja F uma matriz c × l vazia, onde Fij é a freqüência de repetição do
   pixel rij de R.
   Se c > n => c = n
   Se l > n => l = n
   Para cada i variando de 1 até n [O(t)]
      Para cada j variando de 1 até n
            Seja k = n × (i-1) + j
            Se k ≤ t, ou seja, existe o k-ésimo ponto em P
                Seja pk o k-ésimo ponto de P
                Se pk estiver em L1
                      Seja h o índice de pk em L1
                      Seja q o h-ésimo ponto em L2
                      Seja p o k-ésimo ponto de v(M)
                      (Ax)ij = px , (Ay)ij = py
                      (Bx)ij = qx , (By)ij = qy
                      Seja u = i % l e seja v = j % c
                      Fuv = Fuv + 1
   Dx = (Bx - Ax)/C    [O(t)]
   Dy = (w(By) - w(Ay))/C    [O(t)]
   D = (Dx + Dy)/2
   Retornar a imagem detectada D e a matriz F
FIM
```

Algoritmo 4.13 – Obtenção da imagem detectada D [$O(t)$]

Seja **S** a imagem de prova, calculada a partir da média da intensidade dos pixels que se repetem em **D**, onde s_{ij} representa o pixel da linha i e coluna j de **S**. A repetição de pixels irá ocorrer de acordo com o que foi discutido na Seção 4.1.4 (ver Figura 4.6, que ilustra a repetição da imagem **R** na imagem estendida). O Algoritmo 4.14 calcula a imagem de prova **S**.

Uma comparação visual entre as imagens **R** e **S** permite identificar similaridades entre elas. Quanto maiores as similaridades, maior a probabilidade de N corresponder ao mapa marcado $\hat{M}$. Ataques a $\hat{M}$ provocam, em última instância, alterações na imagem **S**, interpretadas como variações nas intensidades dos pixels e degradação da imagem pelo olho humano. Se $\hat{M}$ não sofreu nenhum ataque, então **S** e **R** serão idênticas. A possibilidade ou não de identificar **S** como sendo **R** depende somente da qualidade da imagem **S** do ponto de vista do olho humano.

Também é possível utilizar alguma métrica de comparação de imagens para auxiliar na análise, o que será tratado na Seção 4.4.

Entrada: c e l respectivamente o número de colunas e linhas da imagem original **R**; matrizes **D** e **F** do Algoritmo 4.13.

INÍCIO
Seja t o número de pontos em P. Seja $n = \lfloor \sqrt{t} \rfloor$
Se $c > n$ => $c = n$
Se $l > n$ => $l = n$

Sejam I_{max} e I_{min} o pixel de maior e o de menor intensidade de R, respectivamente. Seja $I_m = (I_{max} + I_{min})/2$;
Seja **S** a matriz vazia que representa a imagem de prova.

Para cada i variando de 1 até l [no máximo $O(t)$]
    Para cada j variando de 1 até c
        Se $F_{ij} > 0$, $S_{ij} = I_m + D_{ij}/F_{ij}$
        Senão, $S_{ij} = I_m$
Retornar a imagem **S**
FIM.

Algoritmo 4.14 – Obtenção da imagem de prova *S*, com base na média dos pixels correspondentes que se repetem na imagem detectada D. [*O*(*t*)]

A Figura 4.10 ilustra a comparação entre a imagem detectada **D** extensamente atacada por adição de ruído e a melhoria em sua definição ao calcular a imagem de prova **S**.

Imagem detectada **D**

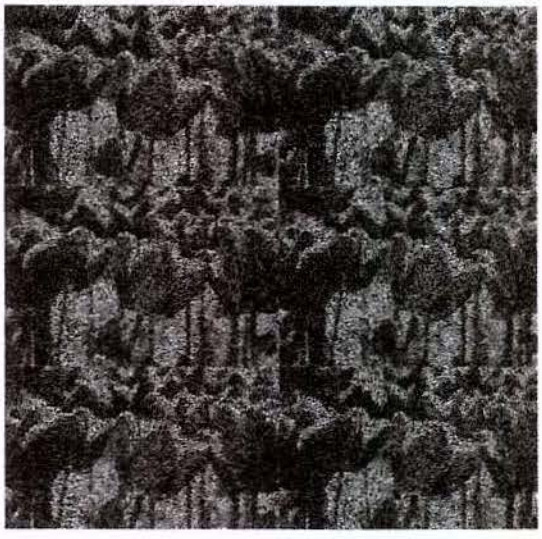


Imagem de prova **S**

Imagem original **R**

Figura 4.10 – Exemplo de extração da imagem detectada D (com ataque) e do cálculo da imagem de prova S.

4.2.5 O Papel da Redundância

Marcas d'água inseridas em conteúdos digitais estão sujeitas a ataques, que são alterações nestes conteúdos que podem perturbá-las ou mesmo destruí-las. Ataques serão tratados em detalhe na Seção 4.3.

No método RAWVec, a MD é inserida através da adição de deslocamentos nas coordenadas. De fato, a MD é inserida duas vezes, uma nas abscissas e outra nas ordenadas dos pontos do mapa. A imagem detectada **D** é então produzida através da média das imagens detectadas nas abscissas e nas ordenadas dos pontos de N, o que tende a diminuir o efeito de perturbações devidas a ataques.

Além disso, é também calculada a imagem de prova **S** a partir de **D** com base na repetição de pixels correspondentes em **D**. Quanto menor forem as dimensões da imagem original R em relação à imagem estendida **E** na inserção da MD, maior será a redundância e com isso maior será a resistência a ataques.

4.3 Tipos de Ataque

Neste capítulo serão descritos alguns tipos de ataque mais comuns que a MD pode sofrer e como nosso novo método RAWVec pode ser afetado por tais ataques. Ataques são alterações nos dados do mapa marcado $\hat{M}$, que deram origem ao mapa de prova N. Podem se refletir em alterações na MD, representada neste caso pela imagem de prova **S**. Como a MD é armazenada nas coordenadas dos vértices das entidades do mapa, os ataques relevantes são os que alteram as coordenadas de todo ou parte do mapa. Alguns tipos de ataque podem ocorrer acidentalmente. Outros ocorrem por necessidade do usuário do mapa ao realizar algum tipo de operação, como por exemplo, uma translação. Por fim, há ataques com o claro objetivo de descaracterizar o mapa original para que sua utilização ilegal não possa ser detectada facilmente.

4.3.1 Ataques de Transformação

Transformações como translações, rotações e ampliações, mudanças de projeção, de *datum* e de sistema de coordenadas são bastante comuns em se tratando de mapas. No caso de um mapa marcado, é possível tratar um ataque de transformação se for possível descobrir a transformação aplicada ao mapa, de forma a ser aplicada sua inversa. Transformações de similaridade podem ser detectadas através de PPM, conforme foi discutido na Seção 4.2.1. Já as transformações de mudança de projeção ou *datum* são combinações de transformações mais complexas e difíceis de serem identificadas automaticamente. Entretanto, mesmo sendo possível descobrir a transformação inversa a ser aplicada, há o problema da introdução de erros durante a aplicação da transformação e de sua inversa, acarretando em maiores dificuldades na detecção da MD atacada. A introdução de erros é causada pela limitação na representação interna das coordenadas.

Dos ataques de transformação, a redução do tamanho do mapa pode ser altamente prejudicial à marca d'água caso a redução implique em perda de precisão nas coordenadas. Apesar de ser uma operação reversível e em geral facilmente identificável pelo PPM, a redução pode ser tão drástica que a transformação não pode ser calculada com precisão devido à limitação na representação das coordenadas. Como exemplo desse tipo de problema, alguns formatos vetoriais armazenam as coordenadas como inteiros longos (é o caso do DGN, utilizado em nossos experimentos). Neste caso, há uma área máxima que o formato consegue representar e uma resolução máxima de representação, chegando a poucas casas decimais. Uma redução neste caso pode causar o arredondamento de coordenadas, acarretando em perturbação ou mesmo a destruição da imagem armazenada.

O método RAWVec resiste a ataques de transformação que possam ser tratados pelo algoritmo de PPM utilizado, dentro dos limites de precisão numérica. Perturbações na precisão que sejam maiores do que a tolerância do algoritmo podem fazer com que coordenadas sejam desprezadas, enquanto perturbações menores que a tolerância poderão provocar alterações nas intensidades dos pixels na imagem detectada.

No caso de um mapa que tenha sofrido transformações locais, há o problema de não existir uma transformação global, envolvendo todos ou a maioria dos pontos, que possa ser calculada pelo algoritmo de PPM. Um modo de tratar este problema é aplicar o algoritmo normalmente, desconsiderando as transformações locais. As coordenadas que sofrem essas transformações serão ignoradas (o problema se reduz ao de cropping, sendo necessário então reduzir o valor da probabilidade ρ). É possível também aplicar o algoritmo várias vezes, indicando a cada vez as entidades correspondentes em áreas diferentes do mapa. Pode-se então calcular a média das imagens resultantes para se obter uma que seja mais próxima da original.

Nos casos em que a transformação não é de similaridade, o algoritmo de PPM não se aplica e é necessário saber de antemão qual a transformação aplicada, de modo a calcular sua inversa e aplicá-la em N antes da extração da MD.

4.3.2 Ataques de Recorte (*cropping*)

Quando um mapa é distribuído, existe a possibilidade de seu usuário não necessitar de toda sua área ou de todas as informações nele contidas. Neste caso, o usuário poderia recortar parte do mapa e utilizar somente a área que lhe interessa, ou remover entidades ou coordenadas de acordo com algum critério. Ataques de *cropping* podem descaracterizar bastante o mapa, tornando difícil identificá-lo como sendo parte do original.

O algoritmo de PPM apresentado lida com *cropping* ignorando coordenadas que tenham sido removidas. Quanto maior a extensão do cropping, entretanto, menos coordenadas sobram para a detecção da marca d'água. Através da inserção da imagem **E** nas abscissas e da imagem $w(\mathbf{E})$ nas ordenadas, pode-se diminuir o efeito destrutivo do cropping sobre a imagem, pois a remoção de uma coordenada só implica na remoção de metade das ocorrências do pixel da MD. Por exemplo, se forem removidos poucos vértices aleatoriamente, há pequenas chances de que dois vértices removidos guardem o mesmo pixel.

A total ausência de um pixel implica em um "buraco" na marca d'água detectada, que precisará ser preenchido com algum valor de intensidade de pixel, como branco, preto ou o valor médio

das intensidades dos pixels, por exemplo. Mas só haverá ausência de pixels caso as duas coordenadas que armazenam um dado pixel forem removidas.

4.3.3 Ataques de Inserção de Entidades

O usuário de um mapa pode também fazer alterações em seu conteúdo adicionando informações que lhe sejam úteis. Dados um mapa marcado e um mapa de prova N, o algoritmo de PPM faz a correspondência entre as coordenadas dos dois mapas, ignorando coordenas adicionadas no segundo. O único cuidado é o de ajustar o valor da probabilidade ρ, visto que haverá pontos em Q que não terão correspondentes em P.

4.3.4 Ataques de Alteração na Ordem das Entidades

O RAWVec utiliza coordenadas correspondentes em dois mapas para calcular a MD. Entretanto, não é necessário que as entidades apareçam na mesma ordem nos dois mapas. De fato, a utilização do algoritmo de PPM permite que as entidades apareçam em qualquer ordem ou mesmo que haja entidades sem correspondência nos dois mapas.

4.3.5 Ataques de Inserção de Ruído

Este tipo de ataque geralmente é esperado de usuários que desejam destruir informações de MD presentes em um meio digital. Em mapas vetoriais, o ataque consiste em introduzir nas coordenadas do mapa algum tipo de ruído, seja aleatório ou de acordo com alguma distribuição de probabilidade, de modo a perturbar ao máximo qualquer informação de MD contida no mapa. Isso é feito somando valores a cada coordenada do mapa. A amplitude máxima do ruído é o valor máximo que pode ter sido somado a uma coordenada durante o ataque.

Os mapas vetoriais, dentro de seus domínios de aplicação, possuem um erro máximo tolerável nas coordenadas. Uma marca d'água resistente a ruído neste caso seria a que resiste a uma amplitude de ruído próxima do erro máximo tolerável. Podemos assumir que, acima deste limite,

o próprio processo de detecção de MD já não faz sentido, pois o mapa já não pode ser mais utilizado para o fim a que se destinava inicialmente.

4.3.6 Ataques de Mudança de Formato de Arquivo

Este tipo de ataque também decorre da utilização normal de um conteúdo digital. A conversão de formato de armazenamento pode ser necessária, por exemplo, para a utilização do conteúdo em outra aplicação. No caso de mapas vetoriais, há diversos formatos comerciais de arquivos. Muitos deles não são perfeitamente conversíveis, ocorrendo alterações na representação dos dados. A conversão entre formatos pode destruir a marca d'água armazenada ao modificar a representação interna do conteúdo. A representação das informações sob a forma de uma seqüência de pontos independente de formato pode diminuir tal efeito, principalmente no caso de entidades mais simples como pontos, linhas e polígonos. Porém, para entidades que não são conversíveis entre os formatos, pode haver perda ou alteração da informação, através da adição ou remoção de coordenadas.

O RAWVec trata desse problema através do algoritmo de PPM durante o processo de extração das informações, permitindo fazer a correspondência entre as coordenadas que se mantiveram. Os casos em que não há correspondência podem ser reduzidos a um ataque de cropping.

4.3.7 Ataques Combinados

A combinação de vários ataques pode tornar ainda mais difícil a detecção da marca d'água no mapa marcado. Por exemplo, ataques de remoção de objetos associados a ataques de transformação podem afetar o algoritmo de PPM, que necessita de uma certa quantidade mínima de pontos para realizar o cálculo da transformação aplicada ao mapa marcado. Ataques de inserção de ruídos associados a ataques de transformação tornam a detecção particularmente difícil. Isso porque o algoritmo de PPM pode não ser capaz de descobrir a transformação aplicada ao mapa devido aos erros adicionados a cada coordenada. Nesse caso, não é possível fazer a correspondência entre as coordenadas de P e de Q. Espera-se, entretanto, que se o erro

ficar abaixo do valor de τ, a aplicação do PPM deva funcionar bem. Mas se além desses ataques ainda houver cropping, não é possível garantir que haja um número suficiente de vizinhos correspondentes para o cálculo da transformação inicial do PPM.

4.4 Análise Teórica dos Algoritmos

4.4.1 Complexidade

As tabelas a seguir mostram as complexidades de alguns dos algoritmos utilizados neste trabalho. Não será explicitado o cálculo da complexidade do algoritmo de PPM, que aparece em detalhes em [Van Wamelen+99]. Aqui, vamos considerar que t é o número de pontos no mapa.

Tabela 4.1 - Complexidade dos algoritmos utilizados na inserção da MD

Algoritmo	Complexidade	Justificativa
Algoritmo 4.1	$O(t)$	Cada um dos pontos do mapa é extraído das entidades. Consideramos que a extração de um ponto é $O(1)$.
Algoritmo 4.2	$O(t)$	Para cada ponto do mapa, um ponto da imagem é escolhido. Consideramos que a escolha de um ponto da imagem é $O(1)$.
Algoritmo 4.3	$O(t)$	A maior complexidade de qualquer uma das operações listadas no algoritmo é $O(t)$.

Tabela 4.2– Complexidade dos algoritmos utilizados na detecção da MD.

Algoritmo	Complexidade	Justificativa
Algoritmo 4.5	$O(t \log t)$	A construção da Triangulação de Delaunay é $O(t \log t)$ [Boissonnat+93]. A obtenção dos k vizinhos de cada ponto do mapa é $O(kt \log t)$ de acordo com [Dickerson+96].
Algoritmo 4.13	$O(t)$	A maior complexidade de qualquer uma das operações listadas no algoritmo é $O(t)$.
Algoritmo 4.14	$O(t)$	A maior complexidade de qualquer uma das operações listadas no algoritmo é $O(t)$.
PPM	$O(t (\log t)^{3/2})$	Conforme demonstrado em [Van Wamelen+99].

Em resumo, na inserção da MD, a complexidade é $O(t)$. Na detecção, a complexidade dos algoritmos resulta em $O(t\ (\log t)^{3/2})$.

4.4.2 Tolerância

É possível controlar a perturbação inserida no mapa com base em uma tolerância predefinida. Sejam I_{min} e I_{max} respectivamente os pixels de maior e menor intensidade de **R** e seja $I_m = (I_{min} + I_{max})/2$. Sendo $I = (I_{max} - I_m)$, então o acréscimo máximo em uma coordenada x ou y de um vértice qualquer de M será $\pm CI$, o que equivale a dizer que um vértice qualquer do mapa sofrerá no máximo um deslocamento de:

$$\varepsilon = \pm CI\sqrt{2} \qquad (20)$$

A Figura 4.11 ilustra a amplitude máxima de deslocamento de uma coordenada.

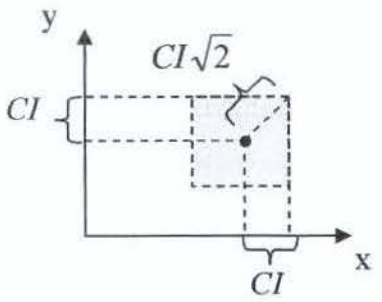


Figura 4.11 – Representação do deslocamento máximo em uma coordenada

O valor de C pode ser escolhido de forma a controlar a amplitude máxima da perturbação inserida nas coordenadas de M para produzir $\hat{M}$, de acordo com a tolerância máxima permitida para o mapa em sua aplicação. Dado que seja tolerável um erro máximo ε a ser introduzido nas coordenadas dos vértices do mapa, temos que

$$C = \frac{\varepsilon}{I\sqrt{2}} \qquad (21)$$

Por exemplo, se a maior intensidade de um pixel de R for 90 e a menor for 10, temos que $I_m = 50$ e o maior deslocamento I sofrido por qualquer ponto será $I = 90 - 50 = 40$. Se a unidade das coordenadas de M for o metro e se o erro máximo tolerável for um deslocamento de 0.25m, temos que é necessário um valor de C menor que 0.00442.

4.4.3 Comportamento Esperado para Cropping

Sendo l e c respectivamente o número de linhas e colunas de **R** e sabendo que cada pixel é inserido duas vezes (uma nas abscissas e uma nas ordenadas, em pontos diferentes) então o número médio de vezes que um pixel de **R** inserido no mapa se repete é de:

$$F = \begin{cases} 2\left\lfloor \frac{n^2}{lc} \right\rfloor, \text{ se } lc < n^2 \\ 2, \text{ se } lc \geq n^2 \end{cases} \qquad (22)$$

Seja $m < n^2$ o número de pontos que possuem pixel associado e que foram removidos do mapa marcado através deste tipo de ataque. Se considerarmos uma remoção aleatória, a probabilidade de que todas as ocorrências de um determinado pixel sejam removidas do mapa é a de que todas as suas ocorrências estejam nos m pontos removidos. Trata-se de uma distribuição hipergeométrica de probabilidade [Ross88], que modela o número total de sucessos em uma amostra de tamanho fixo sem repetições, da forma:

$$p = \frac{\binom{K}{i}\binom{M-K}{N-i}}{\binom{M}{N}} \qquad (23)$$

onde i é o número de sucessos, M é o tamanho da população, K é o número de itens com a característica desejada e N é o número de amostras coletadas.

No caso da probabilidade que desejamos calcular, o tamanho da população é n^2; o número de itens é o número médio de vezes que o pixel se repete, ou seja, F; o número de amostras coletadas é m e o número de sucessos também é F (todas as ocorrências do pixel).

$$p = \frac{\binom{F}{F}\binom{n^2 - F}{m - F}}{\binom{n^2}{m}} = \frac{\binom{n^2 - F}{m - F}}{\binom{n^2}{m}}, \text{ para } F \leq m \leq n \qquad (24)$$

Podemos analisar o comportamento desta função verificando como ela varia de acordo com a variação de F (com m fixo) ou m (com F fixo).

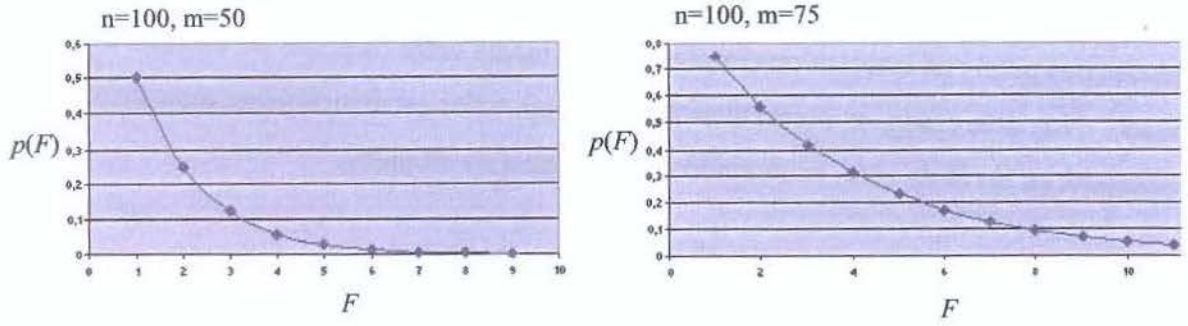


Figura 4.12 – Gráficos da probabilidade de remoção de um determinado pixel em função da freqüência média de ocorrências dos pixels.

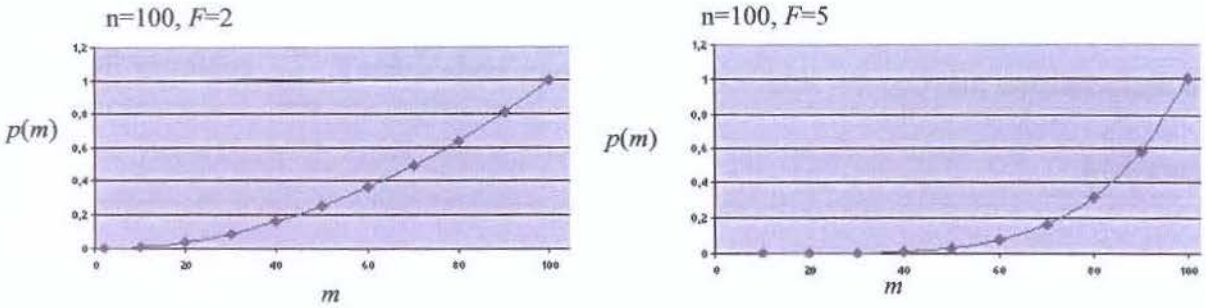


Figura 4.13 – Gráficos da probabilidade de remoção de um determinado pixel em função do número de pontos removidos.

A Figura 4.12 mostra a variação da probabilidade de remoção de um pixel em função de F, a freqüência média de cada pixel na imagem, mantendo-se fixo o valor de m. Quanto maior a freqüência média dos pixels na imagem, menor é a probabilidade de um determinado pixel ser removido da imagem ao serem removidos pontos da mesma. A Figura 4.13 mostra a variação da probabilidade de remoção de um pixel em função de m, o número de pontos removidos do mapa. Quanto maior o valor de m, maior a probabilidade de que um pixel seja removido. Os gráficos mostram também que quanto maior for o valor de F, maior é a resistência ao cropping. No

exemplo, para F=2, quando m é aproximadamente 50, temos p com valor próximo de 50%. No caso de F=5, ainda com m aproximadamente 50, temos p com valor menor que 5%. Na Seção 4.5, iremos verificar se esse comportamento ocorre com experimentos práticos.

A utilização de um pixel médio I_m que é subtraído da imagem inserida e somado à imagem extraída tem um papel interessante na resistência a cropping. Nas áreas em que todos os pixels são removidos da imagem, surgem na imagem extraída pixels de valor zero. Porém, o valor I_m é somado também a tais pixels. Como foi discutido na Seção 4.1.4, I_m pode ser o valor médio das intensidades dos pixels na imagem ou simplesmente a média entre os pixels de máxima e mínima intensidade. Nos dois casos, os pixels ausentes terão um valor estatisticamente mais próximo do valor esperado, ao invés de zero, aumentando a resistência ao cropping.

4.4.4 Comportamento Esperado para Ruído Aleatório

Neste tipo de ataque, a amplitude máxima do ruído, a, será a distância máxima que uma coordenada qualquer poderá ser deslocada após o ataque. Dada uma amplitude máxima de ruído a, o deslocamento máximo que uma coordenada x ou y pode sofrer é $\frac{a}{\sqrt{2}}$, conforme ilustrado na Figura 4.14.

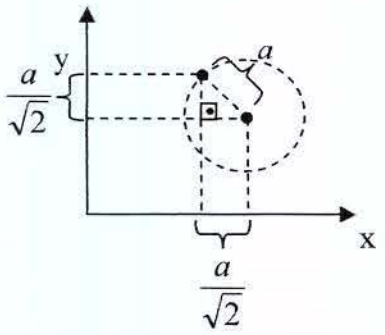


Figura 4.14 – Deslocamento máximo de x e y dado um deslocamento *a* de uma coordenada.

O menor ruído Δ que provoca alteração na MD é aquele que soma ou subtrai 1 de qualquer pixel, ou seja:

$$\varepsilon = CI\sqrt{2} \Rightarrow \varepsilon + \Delta = C(I+1)\sqrt{2} \Rightarrow \Delta = C\sqrt{2} \quad (25)$$

A máxima perturbação δ de um pixel qualquer da imagem **E** pode ser obtida de:

$$\varepsilon + a = C(I+\delta)\sqrt{2} \Rightarrow \delta = \frac{a}{C\sqrt{2}} \quad (26)$$

A conclusão é que, para garantir uma perturbação menor que 1 (valor mínimo de variação de cinza), precisamos ter $a < C\sqrt{2}$. Quando esse não for o caso, é necessário que cada pixel se repita diversas vezes para que os deslocamentos aleatórios tendam a se anular.

A inserção de ruído pode também ser analisada do ponto de vista do erro máximo suportado pelo mapa. Esse erro está relacionado à aplicação a que o mapa se destina. Um ataque tenta manter o mapa marcado na vizinhança de utilidade do mapa original não marcado, ou seja, a perturbação introduzida pelo ataque não pode exceder o erro máximo suportado pelo mapa em sua aplicação [Sion02]. Seja ε_{max} o erro máximo tolerado em um mapa e ε o erro máximo inserido na MD. Como a inserção da MD não pode inutilizar o mapa, temos que necessariamente $\varepsilon < \varepsilon_{max}$. Da mesma forma, podemos assumir que $\varepsilon + a < \varepsilon_{max}$, ou seja, a amplitude máxima de ruído a ser inserida no mapa em um ataque não deveria ultrapassar o erro máximo tolerável, pois nesse caso o mapa ficaria inutilizado e a própria detecção da MD perde o sentido.

4.4.5 Métricas para Comparação das Imagens

Métricas muito utilizadas para comparar imagens antes e após a aplicação de um algoritmo de compressão são o Erro Quadrático Médio (*Mean Square Error* ou MSE) e Razão Máxima Sinal-Ruído (*Peak Signal to Noise Ratio* ou PSNR). Podem ser utilizados também fora do âmbito de compressão para comparar uma imagem que sofreu algum processamento com sua correspondente original.

$$MSE = \frac{1}{lc}\sum_{i=1}^{c}\sum_{j=1}^{l}(r_{ij} - s_{ij})^2 \quad (27) \qquad PSNR = 10\log_{10}\sum_{i=1}^{c}\sum_{j=1}^{l}\frac{255^2}{(r_{ij} - s_{ij})} \quad (28)$$

Nas equações (27) e (28) [Cheng+05], l e c são respectivamente o número de linhas e colunas das imagens sendo comparadas. Quanto maior o valor de PSNR, mais parecidas são as imagens. Da mesma forma, quanto menor o valor de MSE, mais próximas são as imagens.

Outra métrica bastante utilizada para comparar duas imagens compostas por tons de cinza é o **coeficiente de correlação de Pearson**. Sejam duas imagens **R** e **S** e sejam r_{ij} e s_{ij} o pixel da linha i e coluna j de **R** e **S** respectivamente. Sejam $\overline{R}$ e $\overline{S}$ o valor médio das intensidades dos pixels de **R** e **S** respectivamente. O coeficiente de correlação de Pearson é definido por:

$$r = \frac{\sum_i\sum_j (r_{ij} - \overline{R})(s_{ij} - \overline{S})}{\sqrt{\left[\sum_i\sum_j (r_{ij} - \overline{R})^2\right]\left[\sum_i\sum_j (s_{ij} - \overline{S})^2\right]}} \quad (29)$$

Conforme discutido em [Yen+96], há certas limitações na utilização do coeficiente r na comparação de imagens, especialmente se pequenas diferenças precisam ser detectadas. O coeficiente r está diretamente relacionado à distribuição das intensidades dos pixels no histograma das imagens, implicando em baixa sensibilidade a pequenas diferenças. Quanto mais próximos os histogramas das imagens, mais próximo de 1 é o coeficiente. Pode ocorrer também o caso em que uma imagem é o negativo da outra, neste caso o coeficiente será -1. Ainda de acordo com [Yen+96], várias aplicações de segurança consideram satisfatórios na comparação de imagens supostamente idênticas coeficientes mínimos variando entre 0.30 e 0.85.

Nos experimentos exibidos neste trabalho utilizaremos, além da comparação visual das imagens **R** e **S**, o coeficiente de Pearson para compará-las. A baixa sensibilidade do coeficiente a pequenas variações é interessante no contexto de MD. A expectativa é que para se ter um coeficiente de valor baixo, implicando em imagens **R** e **S** com histogramas muito diferentes, o mapa precisa ter sido bastante atacado. Nos casos em que o número de linhas ou colunas da

imagem original **R** for maior que n, a comparação será feita somente com a área da imagem que foi efetivamente inserida no mapa.

4.5 Resultados Experimentais

Nesta seção serão descritos alguns experimentos aplicando o método RAWVec estudado neste trabalho. Os experimentos serão analisados tendo em vista os resultados esperados discutidos na Seção 4.4.3. Foi utilizado o formato PGM (*portable grey map*) para criar e manipular as imagens *raster* utilizadas. O formato dos arquivos vetoriais utilizados foi o DGN, formato CAD proprietário da Bentley, utilizado pelo Microstation™, que foi o software utilizado para visualizar e manipular os mapas. Todos os testes foram feitos com mapas 2D e os mapas correspondem a dados reais.

Como métricas para a comparação da imagem original **R** com a imagem de prova **S**, foram utilizados o coeficiente de correlação r e a análise visual da imagem, que resulta em um coeficiente de qualidade a que chamamos de h. Neste caso, a comparação será subjetiva. Caso seja possível a uma pessoa distinguir em **S** traços inequívocos da imagem **R**, então há uma identificação positiva. Para cada imagem **S**, serão apresentadas a cinco pessoas leigas a imagem **R** e a seguinte pergunta: "*Você diria com certa segurança que a imagem* **S** *é a mesma imagem* **R** *após sofrer algum processo de degradação?*". O valor de h será então o número de respostas positivas, podendo variar de 0 a 5.

4.5.1 Extração sem Ataques

Este experimento foi feito para demonstrar o funcionamento do RAWVec no caso trivial, em que é feita a inserção da MD e, em seguida, sua extração.

$I = 128$; $C = 0.0005$; $\varepsilon = 90$mm; $t = 42769$; $n = 206$

Imagem original
65×61, I = 128

Imagem de prova
$r = 1, h = 5$

Mapa original

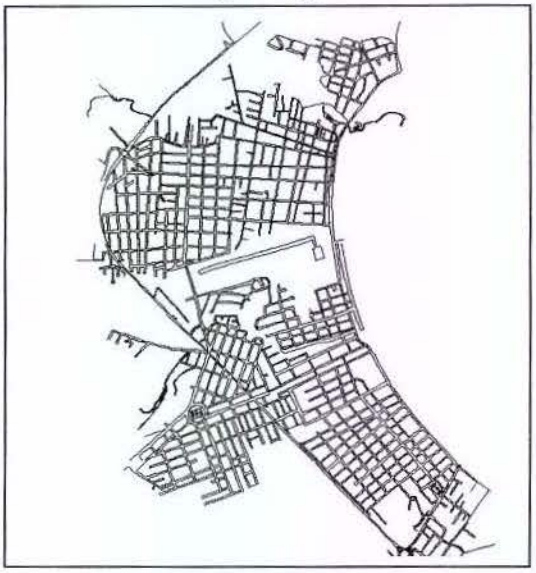


Figura 4.15 – Imagem e mapa originais e a imagem de prova após a inserção e extração.

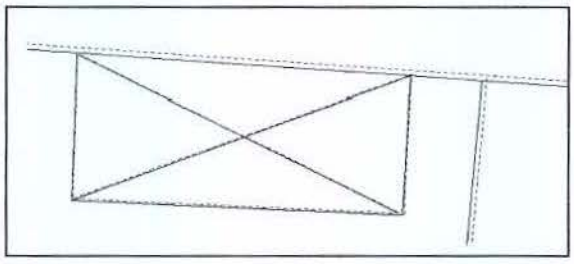


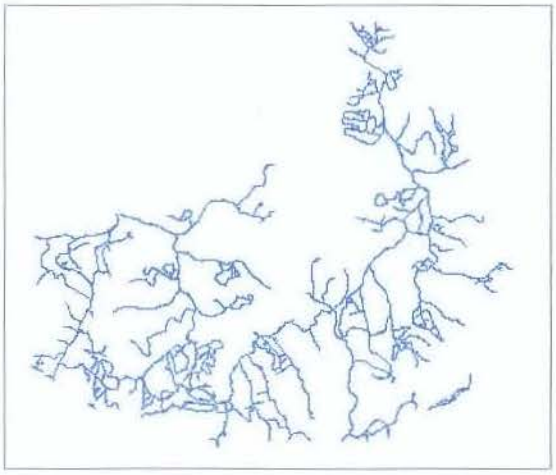
Figura 4.16 – Inserção da marca d'água provoca deslocamento máximo de 90mm.

Na Figura 4.15, a imagem de cima, à esquerda, corresponde à imagem **R** (marca d'água) a ser inserida no mapa M da direita. Após a inserção e a criação do mapa $\hat{M}$, sem nenhum ataque posterior, a imagem **S** (em baixo, à esquerda) foi extraída. O coeficiente r de correlação de *Pearson* teve valor 1, como esperado, indicando que **S** é idêntica a **R**. A análise visual resultou em um coeficiente de qualidade $h = 5$. A Figura 4.16 ilustra o deslocamento sofrido por uma entidade ao ser inserida a MD.

4.5.2 Extração pós Ataque Combinado

Neste experimento, foi aplicado um ataque combinado de inserção de ruído, *cropping*, ampliação, rotação e translação.

Imagem **R** pequena em relação ao mapa e que se repete 6 vezes em **E**.
$I = 128$; $C = 0.0005$; $\varepsilon = 90$mm; $t = 27044$; $n = 164$; F=12; imagem de 61×65 pixels.



Inserção de ruído de amplitude 45mm
Cropping: removidos 13563 pontos (49.8%)
Ampliação: 0.1 do original
Rotação: 232° em relação a (0,0)
Translação: (-2608.5, 2624.7)

Figura 4.17 – Mapa original (à esquerda) e após sofrer um ataque (à direita, ampliado)

Na Figura 4.17, o mapa da esquerda sofreu um ataque considerável, composto por uma redução para 10% do tamanho original, uma rotação de 232° em relação à origem (0,0), uma translação e a remoção de quase metade dos pontos originais.

Imagem extraída de X

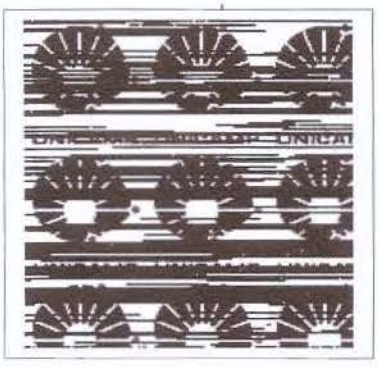


Imagem extraída de Y

Marca d'água original R

Imagem de prova S

Figura 4.18 – Comparação entre a imagem final extraída e a marca d'água inserida ($r = 0.99$, $h = 5$)

As imagens extraídas de X e de Y aparecem com diversas áreas sem informação (áreas escuras), correspondendo aos pontos ausentes devido ao cropping. Entretanto, a imagem final, criada através da média entre as imagens X e Y, é bem semelhante visualmente à imagem original, possuindo com ela uma alta correlação ($r = 0.99$). Isso acontece, como previsto, devido ao fato de a imagem se repetir diversas vezes no mapa e de se ter armazenado a imagem em X na seqüência natural das coordenadas do mapa e em Y na seqüência inversa. Assim, a ausência de um ponto do mapa só remove uma das ocorrências do pixel. Um pixel só é totalmente removido quando todas as coordenadas em que ele ocorre forem removidas. A análise visual obteve $h = 5$.

4.5.3 Extração com Ataque de Cropping

Em cada teste, um conjunto aleatório de entidades foi removido do mapa até atingir o percentual de remoção de pontos necessário. Para cada percentual de remoção testado foram executados cinco recortes (*cropping*) aleatórios e foi retirada a média dos valores do coeficiente r.

Os resultados mostraram uma alta resistência a cropping, permitindo uma identificação positiva da imagem mesmo após mais de 80% dos pontos terem sido removidos e mesmo para imagens que se repetem poucas vezes no mapa. Apesar da resistência, em alguns casos (acima de 90% de remoções) o algoritmo de PPM começou a falhar por não haver um número mínimo de vizinhos em P correspondentes a pontos de Q devido ao cropping. O Algoritmo 4.4 assume que as quantidades de pontos são aproximadamente iguais e pode falhar quando são muito diferentes, por não haver correspondência entre vizinhos. Nestes casos, foram dados valores altos para K ou foi aplicado o Algoritmo 4.11, que utiliza entidades correspondentes nos dois mapas, permitindo calcular uma transformação inicial que não necessite de vizinhos correspondentes.

- Imagem **R** de tamanho médio em relação ao mapa e que se repete 4 vezes em **E**.
  I=127; C=0.00056; e = 100mm; t = 27044; n=164; F = 8; imagem de 77×84 pixels.

Tabela 4.3– Ataque de cropping. A freqüência média de repetição dos pixels no mapa é F=8.

% de pontos removidos	Imagem extraída	r	h
0		1	5
40%		0.999	5
60%		0.978	5
80%		0.893	5

% de pontos removidos	Imagem extraída	r	h
85%		0.808	5
90%		0.709	4
95%		0.561	2
98%		0.280	0

A Tabela 4.3 mostra uma seqüência de ataques de cropping distintos em um mesmo mapa marcado. Cada ataque corresponde à remoção de um certo percentual de pontos do mapa.

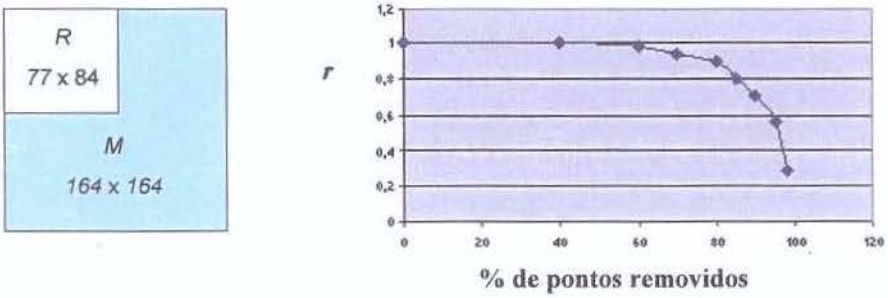


Figura 4.19 – Tamanho relativo de R e M e gráfico da Tabela 4.3

Na Figura 4.19 é mostrado o gráfico do percentual de pontos removidos do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na Tabela 4.3. Como esperado, quanto maior a quantidade de pontos removidos, menor é o valor de r. Porém, a relação não é linear: r começa diminuindo pouco com o número de pontos removidos, mostrando a alta resistência do método até próximo de 60%, mas depois a diminuição se acelera. Mesmo assim, com 80% de pontos removidos a imagem extraída ainda é praticamente idêntica à original. A alta resistência a cropping também se deve à repetição da imagem original no mapa. Na comparação visual, foi possível obter $h = 4$ mesmo para 90% de remoções (Figura 4.20). Para 98% das remoções, nenhum dos avaliadores conseguiu identificar a imagem de prova como sendo a imagem original.

Figura 4.20 – Mapa antes e depois da aplicação de cropping de 90%

- Imagem **R** pequena em relação ao mapa e que se repete 11 vezes em **E**.

  $I = 114$; $C = 0.00062$; $\varepsilon = 100$mm; $t = 42758$; $n = 206$; F=22; imagem de 60×60 pixels

Tabela 4.4– Ataque de cropping. A freqüência média de repetição dos pixels no mapa é F=22.

% de pontos removidos	Imagem extraída	r	h
0		1	5
30%		1	5
60%		1	5
80%		0.967	5
85%		0.912	5
90%		0.872	3
95%		0.741	2
98%		0.482	1

A Tabela 4.4 mostra uma seqüência de ataques de cropping distintos em um mesmo mapa marcado. Cada ataque corresponde à remoção de um certo número de pontos. A imagem original se repete um número relativamente grande de vezes no mapa.

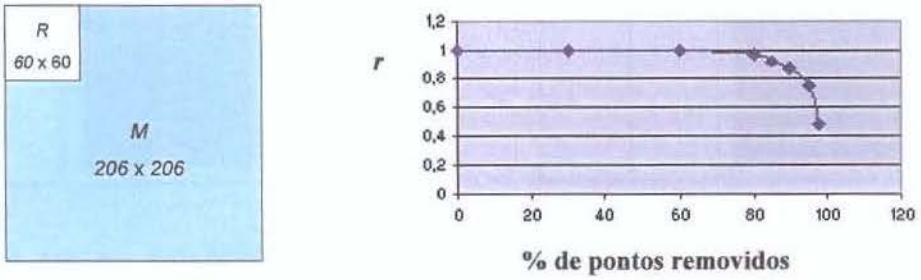


Figura 4.21 – Tamanho relativo de R e M e gráfico da Tabela 4.4

Na Figura 4.21 é mostrado o gráfico do percentual de pontos removidos do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na Tabela 4.4. O gráfico é similar ao da Figura 4.18, mas mostra que r praticamente não se altera até que se atinge cerca de 80% de remoções. Neste caso, com mais de 90% de remoções, a

imagem extraída apresenta ainda uma alta correlação, acima de 0.85, com a imagem original. Como esperado, uma freqüência maior de repetições da imagem no mapa aumenta a resistência do método a ataques de cropping. Na comparação visual, foi possível obter $h = 3$ para 90% de remoções. Mesmo para altíssimos 98% de remoções, um dos avaliadores ainda considerou que a imagem de prova e a original eram idênticas e, neste caso, a correlação ainda é mesmo relativamente alta ($r = 0.482$).

- Imagem **R** grande, que se repete somente uma vez em **E**.
  $I = 90$; $C = 0.00047$; $\varepsilon = 60$mm; $t = 28960$; $n = 170$; F=2; imagem de 170×170 pixels.

Tabela 4.5– Ataque de cropping. A freqüência média de repetição dos pixels no mapa é F=2.

% de pontos removidos	Imagem extraída	*r*	*h*
0		1	5
20%		0.976	5
40%		0.915	5
60%		0.796	4
80%		0.598	3
95%		0.310	0

A Tabela 4.5 mostra uma seqüência de ataques de cropping distintos em um mesmo mapa marcado. Cada ataque corresponde à remoção de um certo número de pontos. A imagem original se repete somente uma vez no mapa.

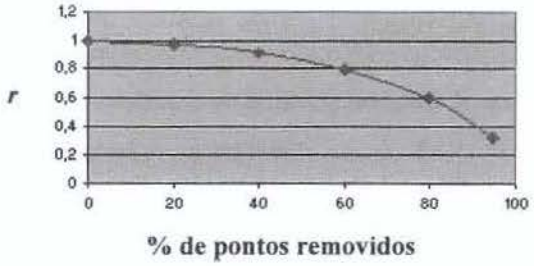


Figura 4.22 – Gráfico da Tabela 4.5

Na Figura 4.22 é mostrado o gráfico do percentual de pontos removidos do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na Tabela 4.5. O gráfico mostra que de r se afasta de 1 bem mais cedo que no gráfico da Figura 4.21. Como esperado, uma menor freqüência de repetições da imagem no mapa diminui a resistência do método a ataques de cropping. Mesmo assim, com 80% dos pontos removidos, ainda foi possível obter uma alta correlação ($r = 0.598$) e um coeficiente de qualidade $h = 3$.

4.5.4 Extração pós Ataque de Inserção de Ruído Aleatório

Em cada teste, ruídos de amplitudes crescentes foram aplicadas a um mapa marcado. Em cada caso, a imagem foi extraída e comparada com a original, obtendo-se r. Cada ataque foi efetuado cinco vezes, utilizando uma nova semente para inserir novamente o ruído aleatório a cada vez, e o valor médio de r foi obtido. A imagem mostrada em cada caso corresponde, das cinco imagens obtidas, à imagem cujo valor de r é o mais próximo do valor médio calculado.

- Imagem **R** com 170×170 pixels, que aparece somente uma vez em **E**.
  $I = 90$; $C = 0.00047$; $\varepsilon = 60$mm; $t = 28960$; $n = 170$; F=2.

Tabela 4.6– Ataque de inserção de ruído. A freqüência média de repetição dos pixels no mapa é F=2.

Ruído	Imagem extraída	r	δ	h
0		1	0	5
10mm		0.988	15	5
20mm		0.947	30	5

Ruído	Imagem extraída	r	δ	h
40mm		0.826	60	5
60mm		0.602	90	5
80mm		0.420	120	3

A Tabela 4.6 mostra uma seqüência de ataques distintos de inserção de ruído aleatório em um mesmo mapa marcado. Cada ataque corresponde à adição de ruído de uma certa amplitude máxima. A imagem original se repete somente uma vez no mapa.

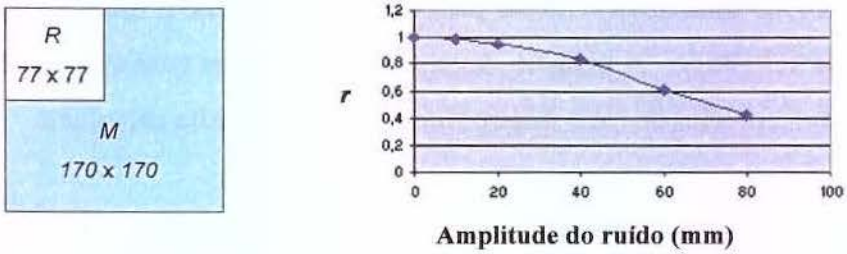


Figura 4.23 – Tamanho relativo de R e M e gráfico da Tabela 4.6

Na Figura 4.23 é mostrado o gráfico da amplitude máxima do ruído inserido do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na Tabela 4.6. O gráfico mostra uma variação lenta de r para ruídos de até cerca de 30mm de amplitude. A partir daí, a diminuição de r começa a decrescer mais rapidamente, de forma que r

é próximo de 0.6 para uma amplitude máxima de 60mm, que também é o deslocamento máximo inserido originalmente no mapa marcado. Neste caso, o coeficiente de qualidade ainda foi máximo, com $h = 5$. Como esperado, quanto maior a amplitude do ruído, menor é o valor de r.

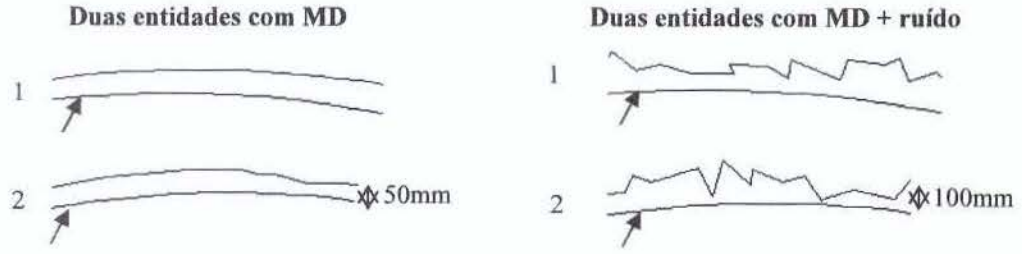


Figura 4.24 – Efeito da inserção de ruído de 80mm no mapa. As setas indicam como eram as entidades em *M*.

A Figura 4.24 mostra duas entidades (1 e 2) do mapa. A figura da esquerda mostra as entidades após a inserção da MD. A da direita mostra as mesmas entidades após a inserção de ruído de amplitude máxima 80mm. As setas indicam como eram as entidades no mapa original *M*. É interessante notar os padrões de "ziguezague" formados ao ser inserido ruído no mapa. A inserção de ruído pelo atacante deverá ser utilizada com critério, pois mesmo com amplitude baixa pode destruir a utilidade do mapa ao criar esses padrões ou mesmo laços nas entidades.

- Imagem **R** pequena, que se repete 4 vezes em **E**.

  $I = 83$; $C = 0.00051$; $\varepsilon = 60$mm; $t = 28960$; $n = 170$; F=8; imagem de 77×77 pixels.

Tabela 4.7– Ataque de inserção de ruído. A freqüência média de repetição dos pixels no mapa é F=8.

Ruído	Imagem extraída	r	δ	h
0		1	0	5
20mm		0.990	28	5
40mm		0.960	56	5
60mm		0.905	84	5
80mm		0.858	113	4
100mm		0.807	141	3
120mm		0.737	166	2
140mm		0.545	194	0

A Tabela 4.7 mostra uma seqüência de ataques distintos de inserção de ruído aleatório em um mesmo mapa marcado. Cada ataque corresponde à adição de ruído de uma certa amplitude máxima. A imagem original se repete quatro vezes no mapa.

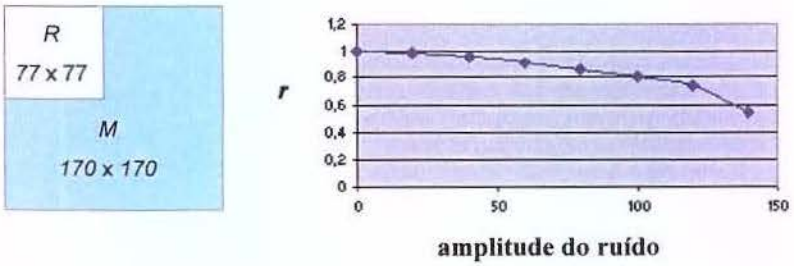


Figura 4.25 – Tamanho relativo de R e M e gráfico da Tabela 4.7

Na Figura 4.25 é mostrado o gráfico da amplitude máxima do ruído inserido do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na

Tabela 4.7. O gráfico mostra uma variação lenta de r para ruídos de até cerca de 40mm de amplitude. A partir daí, a diminuição de r começa a decrescer mais rapidamente, mas há uma maior resistência em relação ao gráfico da Figura 4.23, com r próximo de 0.8 para uma amplitude máxima de cerca de 100mm. Neste teste, a repetição da imagem fez com que a MD resistisse bem mais à inserção de ruído, o que pode ser notado pela alta correlação mesmo para ruído de 140mm. É interessante notar, entretanto, que o coeficiente de qualidade começa a diminuir já com 80mm de ruído. Provavelmente se deve ao fato de a imagem ser pequena e já ter originalmente uma baixa resolução.

A Figura 4.26 ilustra o caso em que foi inserido um ruído de amplitude máxima de 80mm, com uma correlação $r = 0.858$. Aplicamos então um filtro da mediana e um filtro da média, ambos de ordem 3. Nos dois casos, houve uma ligeira melhora na correlação entre a imagem original e a imagem processada, com $r = 0.875$ para o filtro da mediana e $r = 0.872$ para o filtro da média, ilustrando como é possível utilizar filtros na imagem extraída para auxiliar na comparação.

Figura 4.26 – Aplicação de filtro da mediana e da média para o caso de inserção de ruído de 80mm.

- Imagem **R** pequena, que se repete uma vez em **E**.

  $I = 98$; $C = 0.0014$; $\varepsilon = 200$mm; $t = 8917$; $n = 94$; F=8; imagem de 94×71 pixels.

Tabela 4.8– Ataque de inserção de ruído. A freqüência média de repetição dos pixels no mapa é F=2.

Ruído	Imagem extraída	r	δ	h
0		1	0	5
50mm		0.850	25	5
100mm		0.680	51	5
150mm		0.550	76	4
200mm		0.430	101	2
250mm		0.280	126	1

A Tabela 4.8 mostra uma seqüência de ataques distintos de inserção de ruído aleatório em um mesmo mapa marcado. Cada ataque corresponde à adição de ruído de uma certa amplitude máxima. A imagem original só aparece uma vez no mapa.

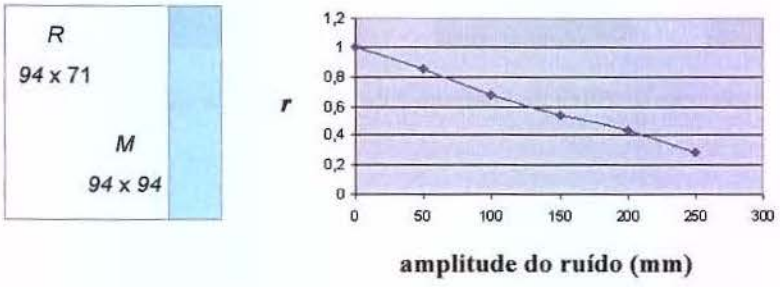


Figura 4.27 – Tamanho relativo de R e M e gráfico da Tabela 4.8

Na Figura 4.27 é mostrado o gráfico da amplitude máxima do ruído inserido do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na

Tabela 4.8. O gráfico mostra uma variação decrescente e praticamente linear de r. Para ruído de amplitude igual ou maior que o valor de ε, a correlação é menor que 0.5 e h é pequeno ($h \leq 2$).

- Imagem **R** pequena, que se repete onze vezes em **E**.
  $I = 98$; $C = 0.0014$; $\varepsilon = 200$mm; $t = 78633$; $n = 280$; F=22; imagem de 94×71 pixels.

Tabela 4.9– Ataque de inserção de ruído. A freqüência média de repetição dos pixels no mapa é F=2.

Ruído	Imagem extraída	r	δ	h
0		1	0	5
50mm		0.970	25	5
100mm		0.913	51	5
200mm		0.817	101	5
300mm		0.701	152	4
400mm		0.603	202	3

A Tabela 4.9 mostra uma seqüência de ataques distintos de inserção de ruído aleatório em um mesmo mapa marcado. Cada ataque corresponde à adição de ruído de uma certa amplitude máxima. Cada pixel da imagem original se repete em média 22 vezes no mapa. A imagem é a mesma da Tabela 4.8, mas o mapa é bem maior, com mais de 78 mil pontos.

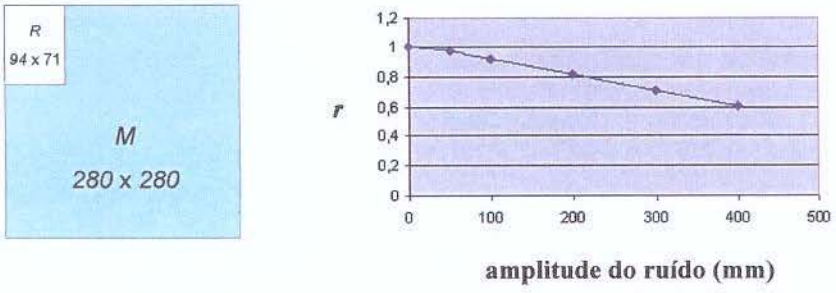


Figura 4.28 – Tamanho relativo de R e M e gráfico da Tabela 4.9

Na Figura 4.28 é mostrado o gráfico da amplitude máxima do ruído inserido do mapa N versus o coeficiente de correlação entre a MD original e as imagens extraídas, cujos valores estão na Tabela 4.9. O gráfico mostra uma variação decrescente e praticamente linear de r, mas desta vez, devido ao número de vezes que a imagem se repete, a correlação é bem alta ($r = 0.603$) mesmo para um valor de amplitude de ruído que é o dobro de ε. Neste teste, a resistência da MD foi bem maior que no anterior, mesmo utilizando a mesma imagem.

4.5.5 Extração pós Ataque de Redução Global

A resolução de representação das coordenadas utilizada nos testes é de 10^{-6}m.

- Imagem com 119×150 pixels que se repete uma vez em **E**;
  $I = 110$; $C = 0.00064$; $\varepsilon = 100$mm; $t = 33392$; $n = 182$; F=2

Tabela 4.10– Ataques de redução

escala	Imagem extraída	*r*	*h*
0.02		1	5
0.001		0.998	5
0.0004		0.989	5
0.0003		0.980	5

escala	Imagem extraída	*r*	*h*
0.0002		0.937	5
0.00015		0.873	5
0.0001		0.655	2
0.00007		0.428	2

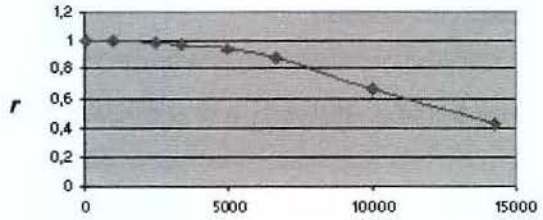


Figura 4.29 – Gráfico da Tabela 4.10

Em nossos testes de reduções progressivas, obtivemos $r = 1$ para todas as reduções de 1 até cerca de 0.01 do tamanho original. A partir deste ponto, o PPM do Algoritmo 4.4 começou a falhar com os valores sugeridos de k na Seção 4.2.3, pois as transformações locais não puderam convergir para transformações globais envolvendo um mínimo de 85% dos pontos, que foi a probabilidade ρ utilizada. O valor sugerido de k neste experimento, conforme discutido na Seção 4.2.2, foi de 5. Para reduções abaixo de 0.01, foi preciso utilizar valores diferenciados de k, variando de 10 até 200. De qualquer modo, quanto maior a redução, mais pontos foram deixados de fora no PPM, resultando em menos pixels na reconstrução da imagem.

Fizemos também os mesmos testes utilizando uma implementação do Algoritmo 4.11, que não utiliza KNN, mas sim pontos que sabidamente se correspondem nos dois mapas para calcular a transformação local a ser refinada em uma transformação global. Os resultados foram praticamente os mesmos que os obtidos na Tabela 4.10, com a diferença de que sempre foi possível encontrar a transformação global, embora o número de pontos que não se encaixaram na transformação também tenha sido crescente.

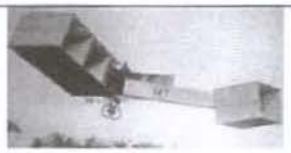
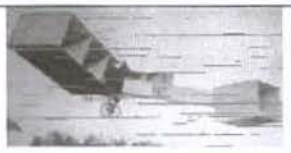
É interessante notar também que a imagem começa a se degradar mais rapidamente quando são atingidas reduções cada vez mais próximas do limite da representação das coordenadas do mapa, que é de 10^{-6}. É possível que reduções por um fator que seja uma potência de 2, mesmo quando próximas do limite de representação das coordenadas, não introduzam ou introduzam poucos erros, menores do que os verificados nos testes. Porém não testamos esse tipo de situação.

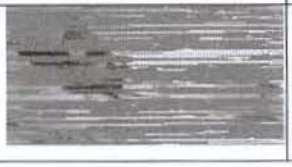
4.5.6 Extração pós Ataque de Transformação Local

Transformações locais são aquelas aplicadas em uma ou mais áreas do mapa. Podem ter sido aplicadas transformações distintas em áreas diferentes do mapa. Esse tipo de transformação tem um efeito parecido com o cropping, pois as coordenadas que sofrem a transformação, em geral, não poderão ser tratadas de maneira global pelo algoritmo de PPM e terão que ser ignoradas. Neste teste, foram aplicadas transformações locais em um determinado percentual de pontos do mapa. Os pontos foram escolhidos de áreas retangulares, cada qual contendo no máximo 10% dos pontos do mapa.

- Imagem com 100×196 pixels que se repete duas vezes em **E**;
  $I = 97$; $C = 0.00073$; $\varepsilon = 100$mm; $t = 42759$; $n = 206$; F=4

Tabela 4.11– Ataques de transformações locais.

pontos	Imagem extraída	*r*	*h*
0		1	5
31%		0.998	5
55%		0.942	5

pontos	Imagem extraída	*r*	*h*
78%		0.657	2
86%		0.584	2
92%		0.412	0

Na Tabela 4.11, foram aplicadas transformações distintas em diversas áreas do mesmo mapa.

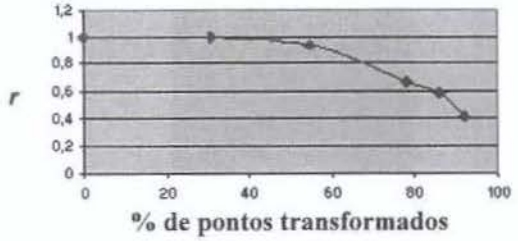


Figura 4.30 – Gráfico da Tabela 4.11

O gráfico da Figura 4.30 mostra a variação de r com o percentual de pontos do mapa envolvidos em transformações locais. Como nos casos de cropping, r começa a diminuir rapidamente após atingido um certo percentual de pontos transformados localmente e ignorados pelo PPM.

4.5.7 Extração pós Ataque de Mudança de Formato

- Imagem com 230×229 pixels que se repete uma vez em **E**;
  $I = 128$; $C = 0.00055$; $\varepsilon = 100$mm; $t = 56917$; $n = 238$; F=2.

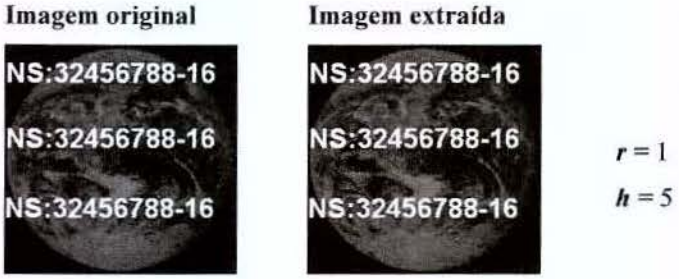


Figura 4.31 – Ataque de mudança de formato com imagem grande

Neste teste, o mapa marcado foi exportado para o formato DXF. Não houve perda de informações na conversão que prejudicasse a MD. Entretanto, observando as duas imagens extraídas de X e de Y mostradas na Figura 4.32, verificamos que aparecem atacadas, indicando que houve perda de informações. Entretanto, ao calcular a imagem **S** através das duas imagens, as informações perdidas de uma puderam ser resgatadas da outra.

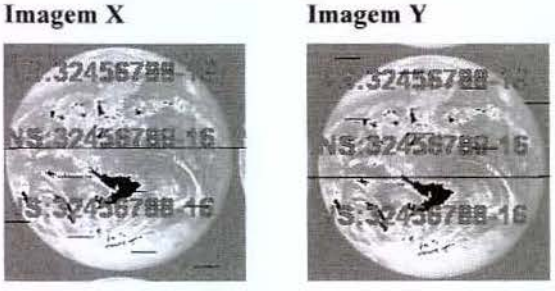


Figura 4.32 – Imagens extraídas de X e Y após o ataque de mudança de formato

- Imagem com 230×229 pixels que se repete uma vez em **E**;
  $I = 128$; $C = 0.00055$; $\varepsilon = 100$mm; $t = 56917$; $n = 238$; F=2.

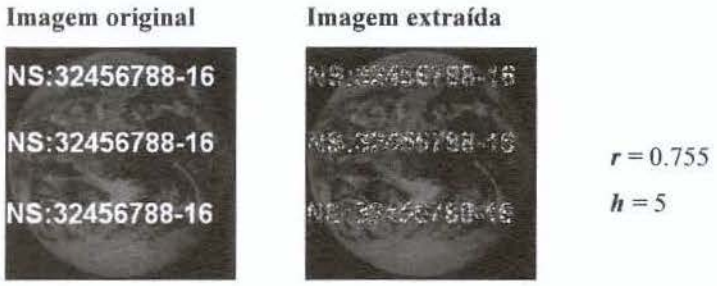


Figura 4.33 – Ataque de mudança de formato com imagem grande

Neste teste, o mapa marcado foi exportado para o formato DXF. Em seguida, o arquivo foi importado no GIS GE Smallworld™. Os dados foram então exportados do GE Smallworld para DXF e re-importados no Microstation, gerando o formato DGN. Houve perda de informação nas conversões, resultando em um valor de r diferente de 1. Verificando o número de pontos no mapa de prova e no mapa original, percebemos que são idênticos. Concluímos que a principal causa da diferença entre as imagens foi que no Smallworld o banco de dados estava configurado para representar coordenadas com uma resolução de 10^{-2}. Sendo assim, toda a informação a partir da terceira casa decimal foi arredondada durante a importação no Smallworld. Isso equivale a dizer que foi somado ou subtraído um valor máximo de 0.005 em cada coordenada. Como aproximação, pode-se dizer que o mapa sofreu um ataque de inserção de ruído aleatório de amplitude máxima de 5mm. Fizemos então este teste, ilustrado na Figura 4.34, em que ruído de amplitude máxima de 5mm foi inserido no mesmo mapa marcado. O teste foi executado cinco vezes e o valor médio de r foi calculado, sendo bem próximo do valor da Figura 4.33.

Figura 4.34 – Ataque de ruído aleatório de 5mm

4.6 Limitações e Melhorias

Nesta seção, discutiremos algumas limitações do método RAWVec e algumas melhorias que poderiam ser implementadas.

4.6.1 Transparência

No método RAWVec, a marca d'água possui baixa transparência perceptual pois, com uma aproximação (ou *zoom*) adequada é possível verificar que as perturbações nos vértices produzem às vezes formação de ângulos mais agudos do que a situação original. Mas as perturbações inseridas podem ser controladas pelo fator C, o que pode ser usado para minimizar esse tipo de ocorrência no mapa marcado. Espera-se também que as alterações nas coordenadas não sejam tão bruscas quanto as que ocorrem na inserção de ruído, pois a) as coordenadas de uma entidade aparecem em seqüência no mapa e são consideradas em seqüência pelo método; b) pixels vizinhos são inseridos no mapa em seqüência, donde coordenadas vizinhas recebem pixels vizinhos (exceto nas bordas da imagem, onde há uma descontinuidade); c) em uma imagem típica, a tendência geral é que pixels vizinhos tenham intensidades próximas, sem variações muito bruscas, exceto nas áreas de contorno. Decorre que coordenadas vizinhas têm maior probabilidade de receberem deslocamentos semelhantes, minimizando os casos de ângulos muito agudos.

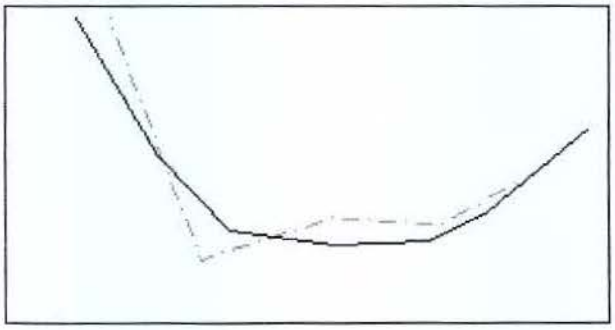


Figura 4.35 – Parte de uma entidade antes (contínua) e depois (pontilhada) da marcação.

Durante a inserção da MD, todos os vértices terão um deslocamento máximo conhecido e que pode ser mantido menor que uma tolerância máxima aceitável, de acordo com o tipo de aplicação a que o mapa se destina, o que caracteriza a transparência funcional. Mas esta é até certo ponto limitada no que diz respeito à topologia. Por exemplo, duas entidades que se tocam em um determinado ponto não necessariamente continuarão mantendo a mesma relação espacial após a inserção da marca d'água, exceto nos casos em que haja realmente um ponto em comum representado no mapa vetorial, conforme foi discutido na Seção 4.1.3. Esse tipo de relação, em que há compartilhamento de coordenadas, permite, por exemplo, que um polígono fechado permaneça fechado após a inserção da MD e que duas entidades que se tocam com uma coordenada comum continuem se tocando após a inserção. Porém, há outras relações que podem ser importantes em determinados contextos. Poderia ser necessário, por exemplo:

- Fazer com que entidades que se intersectam sem coordenadas em comum continuem se intersectando após a inserção. Por exemplo, dois segmentos de reta que se cruzam.
- Fazer com que entidades que possuam relação de continência mantenham a mesma relação após a inserção. Por exemplo, uma entidade no interior de um polígono.
- Manter o valor de áreas ou perímetros invariáveis após a inserção. Pode ser importante em mapas que forneçam informações para cobrança de impostos, em que as entidades representam imóveis e suas áreas ou perímetros são levados em conta.
- Manter a forma das entidades. Por exemplo, um retângulo pode se transformar em um polígono irregular após a inserção.
- Manter entidades válidas. Por exemplo, um retângulo estreito pode ficar degenerado após a inserção (auto-intersecção), dependendo da amplitude de deslocamento de seus vértices.
- Fazer com que entidades disjuntas se mantenham dessa forma após a inserção. Por exemplo, um polígono que não tenha nenhuma entidade em seu interior deve continuar assim após a inserção.

4.6.2 Carga de Informações (payload)

O *payload* é a quantidade de informações que podemos inserir em um método de marca d'água. No RAWVec, o payload é a quantidade máxima de pixels que podem ser inseridos em um mapa. Em um mapa M que possui t coordenadas em sua representação $v(M)$, e sendo $n = \lfloor \sqrt{t} \rfloor$, o payload é dado por n^2, o que pode ser pouco quando se trata de imagens. Um mapa relativamente pequeno, com 1000 pontos, por exemplo, suporta uma imagem de no máximo 31×31 pixels, que não comporta muita informação visual. Mas seria possível armazenar uma imagem com duas vezes mais pixels inserindo metade dos pixels em deslocamentos na coordenada X e metade na Y. Em contrapartida, seria perdida parte da redundância que dá maior resistência aos diversos tipos de ataques.

Também é possível aumentar o payload criando coordenadas extras redundantes no mapa. Pode-se fazê-lo, por exemplo, inserindo entre duas coordenadas que formam um segmento de reta, alguns pontos a mais que sejam colineares com os pontos das extremidades.

Outra possibilidade seria a inserção de dois ou mais pixels em cada coordenada, o que pode ser feito criando-se uma representação da imagem por "super-pixels" de dois ou mais pixels em seqüência. Por exemplo, se um pixel em escala de cinza é representado por 8 bits, um "super-pixel" composto por dois pixels pode ser representado por 16 bits. Isso implicaria em um deslocamento maior nas coordenadas para acomodar a informação extra, o que pode ser controlado diminuindo o valor de C. Porém, um valor muito pequeno para C que faça com que o deslocamento seja próximo do limite de representação numérica das coordenadas torna o método mais sensível a problemas de limitação na representação das coordenadas. Pode-se então diminuir a quantidade de bits que representa cada pixel utilizando um algoritmo como o de redundância mínima (Huffman), para gerar uma representação mais compacta da imagem, com menos bits por pixel e conseqüentemente menor deslocamento nas coordenadas.

4.6.3 Cálculo da Transformação

Da mesma forma que em [Ohbuchi+02] e [Ohbuchi+03], na detecção da MD, o RAWVec utiliza um algoritmo de PPM que considera somente transformações de similaridade, que podem ser descritas como uma combinação de mudança de escala, translação e rotação. Tal algoritmo tenta calcular a transformação fazendo o mapeamento dos dois conjuntos de pontos. O cálculo de uma transformação de similaridade é bastante simples neste caso. Porém, mapas vetoriais com freqüência podem sofrer transformações mais complexas, envolvendo mudança de sistema de projeção. Por exemplo, um modelo comum de transformações para mapas é o de Molodenksy, em que a transformação de coordenadas utiliza parâmetros como a diferença nos elipsóides de referência e nos *data*[11]. Nos casos em que o algoritmo de PPM proposto não se aplique pelo fato de a transformação que mapeia um conjunto de pontos no outro não seja de similaridade, e caso tal transformação seja conhecida, pode-se removê-la previamente utilizando alguma outra ferramenta antes de proceder à detecção da MD.

4.6.4 Mapas 3D

O RAWVec pode ser aplicado também a mapas tridimensionais. A maneira mais trivial de fazê-lo é simplesmente ignorando a coordenada Z nos pontos, mas mantendo-a armazenada. Também é possível utilizar a coordenada Z para armazenar informações da MD, o que aumentaria o *payload* do método caso parte dos pixels fossem armazenados da coordenada Z, ou a resistência a ataques, caso todos os pixels fossem armazenados de forma redundante em Z. A utilização da coordenada Z não impede que se aplique um algoritmo de PPM em 2D, como o proposto por [Van Wamelen+99], mas também pode-se utilizar um algoritmo de PPM em 3D.

4.6.5 Necessidade do Mapa Original para Detecção

O método RAWVec pode ser classificado como não-cego, por necessitar do conteúdo original para a detecção da MD. Esta necessidade pode ser uma desvantagem quando se leva em conta

[11] plural de *datum*, que é um ponto de referência associado a um modelo da forma da Terra, utilizado para calcular coordenadas.

que dois mapas devem ser processados durante a detecção e que é necessário manter o mapa original. Diversos métodos seguem esta linha, dentre os quais [Praun+99], [Ohbuchi+00], [Ohbuchi+02], [Ohbuchi+03] e [Gou+04].

4.6.6 Alteração das Propriedades do Mapa

Como foi discutido na Seção 4.1.3, são tratados pelo RAWVec os casos de manutenção da topologia por compartilhamento de coordenadas, dentro de certa tolerância. Entretanto, não é possível garantir que nenhuma outra relação entre as entidades se mantenha, o que pode ser uma limitação importante em determinadas situações. Por exemplo, uma entidade interna a um polígono não necessariamente continuará interna após a marcação. Outras propriedades das entidades, como perímetros, áreas, comprimentos e distâncias relativas também poderão se alterar na inserção da MD. Mas a escolha de um valor apropriado para C permite controlar as perturbações inseridas no mapa marcado e, dessa forma, minimizar tais alterações. Mesmo assim, é necessário analisar se é possível modificar o RAWVec para que uma ou mais dessas propriedades possam se manter após a marcação do mapa.

Capítulo 5 – Conclusão e Trabalhos Futuros

Neste trabalho, introduzimos um novo método para inserção de marcas d'água em mapas vetoriais denominado Marcas d'Água *Raster* em Mapas Vetoriais (RAWVec). Este novo método insere em um mapa vetorial uma matriz, tipicamente uma imagem *raster*, através de deslocamentos controlados das coordenadas das entidades do mapa, produzindo um mapa marcado que pode, então, ser distribuído. Para determinar se um determinado mapa de prova corresponde a uma cópia distribuída, realiza-se a detecção da marca d'água utilizando o mapa original e o mapa de prova e extraindo deste último uma imagem de prova, comparando-a então com a imagem original. A vantagem de se ter uma marca d'água representada por uma imagem é que parte da resistência a ataques fica por conta da avançada capacidade visual humana em detectar semelhanças entre imagens. Em um mapa marcado, os ataques importantes são aqueles que provocam alterações ou perdas de coordenadas. Alterações nas coordenadas podem provocar alterações nos valores dos pixels na imagem detectada. Ausência de coordenadas pode acarretar perdas de pixels na imagem detectada. Mesmo após ataques, ainda é possível que a imagem seja reconhecida.

A utilização de uma métrica permite uma análise inicial da proximidade da imagem de prova com a imagem original. No caso do coeficiente de correlação de Pierson (r), tipicamente existirá uma correlação muito alta entre as imagens se o valor for superior a 0.5, por exemplo. Assim, após a identificação da correlação, pode-se então fazer a análise visual para concluir se as imagens se correspondem.

Pudemos verificar, através de vários experimentos práticos, que o RAWVec é robusto quanto à inserção de ruído aleatório, cropping, adição de entidades, troca da ordem de ocorrência das entidades no mapa, transformações de similaridade globais e locais, mudança de formato do arquivo digital e combinações desses ataques. Como esperado, há uma relação direta entre a freqüência média dos pixels no mapa e a resistência a ataques. Quanto menor a imagem inserida em relação ao número de pontos do mapa, maior é a resistência a ataques. A amplitude máxima do deslocamento inserido em uma coordenada é controlada pelo parâmetro C, real e positivo.

Este parâmetro é uma função do erro máximo a ser inserido nas coordenadas do mapa, de acordo com a equação (21). Quanto menor o valor deste parâmetro, menor será a perturbação inserida nas coordenadas. Entretanto, deve-se levar em consideração a precisão na representação das coordenadas do mapa ao ser escolhido o valor de C. Em alguns de nossos experimentos utilizando representação inteira das coordenadas, tivemos uma resolução de 10^{-6} e, neste caso, valores de C que introduzem perturbações muito próximas deste limite tornaram a MD mais sensível a erros devido à limitação na representação numérica.

Nos experimentos, percebemos uma notável resistência a cropping, mesmo para imagens grandes em relação ao número de pontos do mapa. Houve casos em que a imagem de prova ainda pôde ser facilmente reconhecida, mesmo após a remoção aleatória de mais de 90% dos pontos do mapa. Em todos os casos apresentados, a imagem de prova pôde ser reconhecida mesmo após a remoção de cerca de 60% dos pontos do mapa.

Na inserção de ruído aleatório, mesmo imagens com baixa freqüência de repetição de pixels tiveram boa resistência ($r > 0.5$ e $h > 2$) para amplitudes máximas de ruído até próximas do deslocamento máximo ε inserido nas coordenadas através da MD. Temos que, para ruídos de amplitude ε, o deslocamento máximo de uma coordenada qualquer em relação à sua posição original é 2ε. Isto pode indicar que um valor bom para ε seja a metade do erro máximo permitido para o mapa, de modo que, se for feito um ataque de ruído de amplitude maior que ε, o mapa ficaria inutilizado e a MD não precisaria mais resistir. Entretanto, a relação entre ε e o ruído máximo suportável pela MD precisa ser melhor investigada. Mas ficou claro que os ataques de inserção de ruído prejudicam o aspecto das entidades, especialmente as que possuem coordenadas muito próximas, criando padrões de ziguezague ou mesmo laços. Em geral, essas características não são aceitáveis em mapas comerciais, o que indica que a inserção de ruídos de amplitude muito alta não é um ataque útil.

Ao ser analisada a resistência de um método de MD, é necessário ter em mente a aplicação do conteúdo digital em questão e até onde a MD deve resistir antes que o conteúdo fique inutilizado. Quando tratamos de mapas, estes possuem em geral um erro máximo associado, para que possam ser utilizados em seus domínios de aplicação. Mapas cujas coordenadas excedem tal erro não

podem ser utilizados em determinadas aplicações que necessitam de informações mais precisas. Tipicamente a faixa de erro aceitável pode ir de poucos centímetros até alguns metros. Mostramos que é possível inserir a MD através de deslocamentos das coordenadas, mas mantendo-as dentro do raio de tolerância necessário. Uma conclusão importante é que, além outros fatores que levam à robustez do RAWVec, quanto maior o deslocamento inserido com a MD, menor será a margem que os usuários têm para ataques ao mapa. De fato, a situação ideal é a em que o deslocamento máximo inserido pela MD equivale ao deslocamento máximo suportado pelo mapa. Porém, isso pode diminuir o valor do mapa por estar no limite do erro permitido e por prejudicar a transparência perceptual dos dados. Há um claro compromisso entre a qualidade dos dados e a utilização de uma MD efetiva.

5.1 Trabalhos Futuros

Existem ainda vários aspectos do nosso novo método que poderiam ser melhor explorados em trabalhos futuros. Nesta seção, discutiremos alguns desses aspectos.

É importante compreender melhor por que determinadas imagens oferecem maior resistência a ataques do que outras e até que ponto isso também depende de características do próprio mapa. Neste trabalho estudamos somente o efeito do tamanho relativo da imagem, mas seria interessante levar em conta também outros parâmetros da imagem, como histograma, contraste, gradiente, etc.

Também consideramos neste trabalho somente um tipo específico de relação topológica, que leva em conta o compartilhamento de coordenadas, conforme discutimos nas seções 4.1.3 e 4.6.1. Porém é preciso estudar como manter outras relações topológicas intactas ao inserir a MD no mapa, aumentando a transparência funcional do método. Poderíamos, por exemplo, estudar um modo de se analisar a imagem em conjunto com o mapa para se determinar que alterações devem ser feitas na imagem, antes de inseri-la no mapa, de modo que determinadas características do mapa sejam mantidas após a inserção, inclusive a topologia. Com base nessa análise, seria feito então um pré-processamento da imagem para que então fosse inserida no mapa.

Conforme discutimos na Seção 4.6.2, nosso método possui um *payload* relativamente baixo, pois a imagem deve ter no máximo t pixels, onde t é o número de pontos do mapa. Poderíamos aumentar o *payload*, criando representações mais compactas da imagem a ser inserida ou inserindo mais de um pixel na mesma coordenada do mapa. Mas seria necessário estudar o impacto dessas alterações na robustez, visto que o pressuposto do método é que os ataques afetam diretamente os pixels da imagem. Nessas novas situações, os pixels seriam afetados indiretamente.

Como já foi discutido, seria interessante estudar se é possível fazer com que o método passe a ser público, ou seja, que não seja necessário o mapa original para a detecção. Neste caso, não seria mais necessária a utilização de PPM, pois o próprio mapa de prova já possuiria toda a informação necessária para a detecção. A não utilização de PPM seria uma grande vantagem, visto que o algoritmo pode ser bastante custoso em seu pior caso. Também seria necessário estudar o impacto na robustez, visto que um atacante poderia agora ser capaz de descobrir a MD e removê-la completamente do mapa, o que não acontece quando o método é privado, quando é impossível inferir qualquer coisa sobre a MD sem conhecer a imagem ou o mapa originais.

Também poderia ser interessante ter o método público como uma de suas variações, para ser utilizado como MD de ilustração (*illustration watermark*), conforme descrito em [Sonnet+03]. Poderíamos então inserir uma imagem com informações úteis a respeito do mapa, como por exemplo uma fotografia aérea ou uma imagem de satélite que se sobreponha ao mapa. Neste caso, o objetivo da inserção da MD não seria mais a determinação de autoria e não seria mais necessária a preocupação com ataques maliciosos. Seria necessário, porém, estender o método para suportar imagens coloridas e com maior definição. Conseqüentemente, seria preciso aumentar o *payload*, também para suportar imagens maiores.

Como o RAWVec se baseia na comparação das coordenadas do mapa de prova e do mapa marcado, um ataque efetivo contra o método seria aquele que refaz as entidades do mapa mantendo sua forma mas alterando completamente as coordenadas. Esse tipo de ataque seria possível em determinadas entidades parametrizadas, como curvas paramétricas, por exemplo, em

que a mesma curva pode ser gerada a partir de diversas configurações de pontos de controle. Em [Ohbuchi+00] é descrito um método de MDs que utiliza reparametrização de um tipo específico de curva, que altera seus pontos de controle, mantendo entretanto sua forma original exata. Porém, em mapas contendo somente entidades mais simples, como pontos e segmentos de reta, não seria possível esse tipo de ataque, pois só existe uma maneira de representar um ponto e, no caso de um segmento de reta, que em um mapa vetorial é representado pelos dois pontos das extremidades, não é possível manter a mesma forma da entidade quando se alteram os pontos. Sendo assim, para prevenir esse tipo de ataque, é interessante utilizar entidades mais simples, o que de fato acontece na maioria dos mapas vetoriais, ou ainda converter entidades complexas em entidades mais simples.

Capítulo 6 – Referências

[Boissonnat+93] Boissonnat, Jean-Daniel; Teillaud, Monique; "On the Randomized Construction of the Delaunay Tree"; Theoretical Computer Science, 112, 1993, p. 339-354.

[CGAL 3.1] Computational Geometry Algorithms Library, release 3.1, "CGAL::nearest_neighbors", "nearest_neighbor_delaunay_2.h", www.cgal.org

[Cheng+05] Cheng, Liang; Bossi, Stefano; Mohapatra, Shivajit; El Zarki, Magda; Venkatasubramanian, Nalini; Dutt, Nikil; "Quality Adapted Backlight Scaling (QABS) for Video Streaming to Mobile Handheld Devices"; IEEE/IEE/LNCS 4th International Conference on Networking (ICN 2005), 2005; Disponível em: <citeseer.ist.psu.edu/722345.html>.

[Dickerson+96] Dickerson, M. T.; Eppstein; "Algorithms for Proximity Problems in Higher Dimensions"; Computational Geometry: Theory and Applications 5, 1996, p. 277-291.

[Giannoula+02] Giannoula, A.; Nikolaidis, N.; Pitas, I.; "Watermarking of Sets of Polygonal Lines using Fusion Techniques", in Proc. of IEEE International Conference on Multimedia and Expo 2002 (ICME2002), Laussane, Switzerland, 26-29 August 2002.

[Gou+04] Gou, Hongmei; Wu, Min; "Data Hiding in Curves for Collusion-Resistant Digital Fingerprinting"; IEEE International Conference on Image Processing (ICIP'04), Singapore, 24-27 Oct. 2004.

[Hartung+98] Hartung, Frank; Eisert, Peter; Girod, Bernd; "Digital Watermarking of MPEG-4 Facial Animation Parameters"; Computer & Graphics, Special Issue on Data Security in Image Communication, v. 22, n. 4, Elsevier Science, 1988, p. 425-435.

[Ohbuchi+00] Ohbuchi, Ryutarou; Masuda, Hiroshi; "Managing CAD Data as a Multimedia Data Type Using Digital Watermark", IFIP WG 5.2, Fourth International Workshop on Knowledge Intensive CAD (KIC-4), Parma, Italy, 2000.

[Ohbuchi+02] Ohbuchi, Ryutarou; Ueda, Hiroo; Endoh, Shuh; "Robust Watermarking of Vector Digital Maps" – in Proc. IEEE Conference on Multimedia and Expo 2002 (ICME 2002), Lausanne, Swistzerland, August 26-29, 2002.

[Ohbuchi+03] Ohbuchi, Ryutarou; Ueda, Hiroo; Endoh, Shuh; "Watermarking 2D Vector Maps in the Mesh-Spectral Domain" – International Conference on Shape Modeling and Applications, Seoul, Korea, May 12-15, 2003.

[Praun+99] Praun, Emil; Hoppe, Hugues; Finkelstein, Adam; "Robust Mesh Watermarking"; Proceedings of SIGGRAPH 99, August 1999, p. 49-56.

[Ross88] Ross, Sheldon; "A First Course in Probability (3rd edition)"; Macmillan Publishing Co., Inc.; 422 pages; 1988.

[Sion02] Sion, Radu; "Power: A Metric for Evaluating Watermarking Algorithms"; International Conference on Information Technology: Coding and Computing; Las Vegas, NV, April 08-10, 2002.

[Sonnet+03] Sonnet, Henry; Isenberg, Tobias; Dittmann, Jana; Strothotte, Thomas; "Illustration Watermarks for Vector Graphics"; 11th Pacific Conference on Computer Graphics and Applications (PG '03); Canmore, Canada, October 08-10, 2003.

[Van Wamelen+99] Van Wamelen, P. B.; Li, Z.; Iyengar, S.S.; "A Fast Expected Time Algorithm for the Point Pattern Matching Problem"; Tech. report 1999-28, Lousiana State Univ., Dept. of Mathematics, 1999.

[Voigt+02] Voigt, Michael; Busch, Christoph; "Watermarking 2D Vector Data for Geographical Information Systems"; Security and Watermarking of Multimedia Contents IV (Proceedings of SPIE Volume 4675); San Jose, CA; 2002; p. 621-628.

[Voigt+04] Voigt, Michael; Yang, Bian; Busch, Christoph; "Reversible Watermarking of 2D-Vector Data", Proceedings of the 2004 multimedia and security workshop on Multimedia and security; Magdeburg, Germany, 2004, p.160-165.

[Wang+98] Wang, Houng-Jyh Mike; Su, Po-Chyi; Kuo, C. Jay; "Wavelet-based Digital Image Watermarking"; Opt. Express 3, 491-496, 1998.

[Williams64] Williams, J. W. J.; "Algorithm 232"; Communications of the ACM; 7(6), p. 347-348, 1964.

[Yen+96] Yen, Eugene K.; Johnston, G. Roger; "The Ineffectiveness of the Correlation Coefficient for Image Comparisons"; Report LAUR-96-2474, Los Alamos National Laboratory, 1996.