UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

CAESAR RALF FRANZ HOPPEN

Esteganografia e Marcas D'água: A busca por algoritmos robustos.

Trabalho de Conclusão apresentado como
requisito parcial para a obtenção do grau de
Bacharel em Ciência da Computação

Prof. Dr. Raul Weber
Orientador

Porto Alegre, julho de 2010

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Franz Hoppen, Caesar Ralf

Esteganografia e Marcas D'água: A busca por algoritmos robustos. / Caesar Ralf Franz Hoppen. – Porto Alegre: Graduação em Ciência da Computação da UFRGS, 2010.

42 f.: il.

Trabalho de Conclusão (bacharelado) – Universidade Federal do Rio Grande do Sul. Curso de Bacharelado em Ciência da Computação, Porto Alegre, BR–RS, 2010. Orientador: Raul Weber.

1. UFRGS. 2. Esteganografia. 3. Segurança. 4. Marcas d'água. 5. Espalhamento espectral. 6. Robustez. 7. DCT. 8. Manipulação de imagem. I. Weber, Raul. II. Título.

AGRADECIMENTOS

Agradeço a todos meus amigos de peito, que sempre me deram apoio não importando o quão insana fosse minha empreitada.

Agradeço a minha família, por ser minha base e ter me ensinado que estudo e esforço é tudo.

Agradeço a UFRGS, por ter me dado o conhecimento, sem o qual nada seria hoje.

Agradeço ao meu orientador, Raul Weber, por ter me aceitado e ajudado com este trabalho ao qual entrego finalmente com tanto gosto.

Agradeço principalmente as seguintes pessoas:

Laércio Pilla, por ser um grande amigo, incentivador e sempre estar lá para ajudar sem pedir nada de volta.

José Antônio Salini Ferreira, por ser também um ótimo amigo e uma das pessoas que eu mais respeito na Informática. Quando não encontrava um caminho para o trabalho, nossas discussões sempre me guiavam a uma solução.

E finalmente a Lucilene Cobalchini, por ser uma pessoa tão especial na minha vida, tendo acompanhado toda minha vida acadêmica direta e indiretamente, assim como ter puxado minha orelha quando descobria que eu não estava me concentrando neste trabalho.

SUMÁRIO

LISTA DE ABREVIATURAS E SIGLAS

DCT	*Discrete Cosine Transform*
LSB	*Least-Significant Bit*
RGB	*Red, Green and Blue*
ARGB	*Alpha, Red, Green and Blue*
Y'CbCr	*Luminance, Red Chroma and Blue Chroma*
HVS	*Human Visual System*
JVM	*Java Virtual Machine*
SSIS	*Spread Spectrum Image Steganography*
HTML	*HyperText Markup Language*

LISTA DE FIGURAS

RESUMO

O ser humano sempre sentiu necessidade de privacidade. Em um mundo que a informação pode ser vista e/ou interceptada por todos, métodos que garantem a privacidade de nossa comunicação são criados a todo momento. Um dos ramos mais explorados é chamado de *Esteganografia*. Neste trabalho de conclusão iremos demonstrar dois algoritmos de Esteganografia para imagens que utilizam-se da anonimidade da informação para criar mensagens robustas. O primeiro é considerado o algoritmo mais simples, e insere cada bit da mensagem no bit menos significativo (LSB) dos pixels de uma imagem. O segundo, utiliza-se do domínio frequência e subamostragem da imagem original. No fim, o propósito dos dois será o mesmo: Servir como métodos para inserção de Marca D'água. Uma análise mostrará que não poderemos usar o primeiro algoritmo, pois além de ser muito frágil, teremos mudanças visuais significativas para aumentar sua robustez. No entanto, resultados mostrarão que o segundo método além de ser bastante invisível, garante robustez contra distúrbios altos na imagem, como compressão JPEG.

Palavras-chave: UFRGS, esteganografia, segurança, marcas d'água, espalhamento espectral, robustez, DCT, manipulação de imagem.

ABSTRACT

The human being always felt the need for privacy. In a world where information can be viewed and/or interpreted by everyone, methods that can ensure the privacy of our communications are created every moment. One branch of methods which are increasingly being exploited is called *Steganography*. This work will show two Steganography algorithms for images that uses information anonymity to create robust messages. The first one is considered the most simple algorithm, and adds each message bit to the least significant bit (LSB) of pixels of an image. The second, subsamples the original imagem and uses it frequency domain. Ultimately, the purpose of both is the same: To serve as methods for inserting watermarks. An analysis will show that we can not use the first algorithm, because it's fragile and in order to increase its robustness we will have significant visual changes in the original recipient. However, results show that the second method besides being quite invisible, guarantees high robustness against disturbances in the image, like JPEG compression.

Keywords: Security, Watermark, Steganography, DCT, Image Manipulation, Spread Spectrum.

1 INTRODUÇÃO

Spion é um super espião disfarçado no território inimigo. Sua missão é complicada: obter informações privilegiadas e as enviar para seu Quartel General (QG) utilizando os meios de comunicação abertos sem levantar qualquer tipo de suspeitas. Infelizmente, isto acaba por destruir sua idéia inicial de utilizar *Criptografia* (SIM 94), pois ao ver uma mensagem confusa sendo enviada, os agentes do inimigo provavelmente desconfiariam de algo. Desanimado, Spion decide descansar e retira uma nota do dinheiro local para comprar algo para comer, quando algo chama-lhe a atenção. Ao colocar o dinheiro contra um recepiente de luz, uma marca escondida é revelada! Como não pensou nisso antes?! A solução para seu problema foi encontrada: *Esteganografia*.

Esteganografia (Do Grego *steganos*, "protegido", e *gráphein*, "escrita") é o meio de guardar informação de forma que esconda a existência da mesma. Diferente da Criptografia (Do Grego *kryptós*, "escondido", e *gráphein*, "escrita"), que visa transformar a informação de forma que ela se torne ilegível (*encriptação*) para todos aqueles que desconheçam o método para transformá-la de volta a forma original (*decriptação*), a Esteganografia utiliza-se da obscuridade para atingir a segurança dos dados. Ou seja, a principal diferença entre estas duas metodologias, e uma das grandes incoveniências da criptografia, é o conhecimento da existência de informação secreta. Para o caso de Spion, passar dados criptografados para seu QG através de um dos canais de comunicação monitorados resultaria na interceptação dos mesmos pelo inimigo. Porém, se ele embutir a informação secreta dentro de um recepiente aparentemente inocente, digamos uma foto sua aproveitando as 'férias' no território inimigo, quando esta for interceptada não levantará suspeitas. *O objetivo da esteganografia não é de prevenir os outros de saberem que há uma informação escondida — é de prevenir que os outros sequer saibam que a informação existe (ART 2001)*, o que nos introduz a um novo problema: como iremos esconder os dados?

Existem diferentes técnicas que podem ser utilizadas para conseguirmos estaganografar nossos dados, assim como diferentes aplicações para cada uma. Uma delas e a mais estudada é a de *Marcas D'água*. Uma Marca D'água nada mais é que uma informação contida em outra visando carregar dados sobre a mesma. Um bom exemplo é o da marca escondida em notas de dinheiro, que tem por objetivo identificar uma nota unicamente, de forma que impeça falsificações. Além, ela não altera a visualização da nota, já que é visível somente em condições especiais (no caso, somente quando colocamos a nota contra um recepiente de luz). Com a popularização da internet podemos encontrar vários usos para tal técnica, entretanto, normalmente, o principal foco é o de identificação de direitos autorais de arquivos com fins de combater à pirataria. Assim, encontramos nos últimos anos um crescimento contínuo de artigos científicos sobre Esteganografia e Marcas D'água. A Figura 1.1 ilustra este acontecimento do período de 1994 a 2004. A

necessidade de técnicas para adicionar segurança aos dados sendo trafegados é interminável e a escolha de um bom algoritmo depende do que priorizamos como importante. Para Marcas D'água, o que desejamos é que ela se mantenha invisível, assim como permaneça no arquivo que foi inserida, mesmo que este seja manipulado. Ou seja, desejamos um algoritmo que torne nossa marca em uma Marca D'água *robusta*.

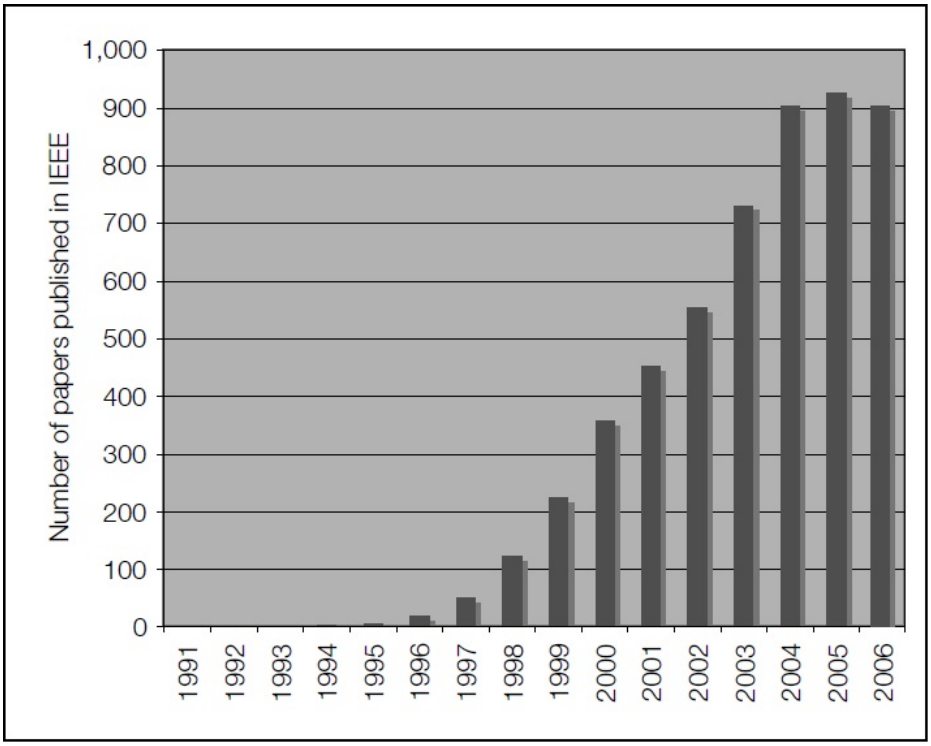


Figura 1.1: Número de artigos sobre Esteganografia e Marcas D'água de 1991 a 2004 (COX 2007).

1.1 Estrutura do Trabalho

Este trabalho apresenta dois algoritmos de Esteganografia implementados pelo autor com foco em Marcas D'água em imagens. Num primeiro momento realizar-se-á uma análise do algoritmo mais simples existente de forma a mostrar os atributos principais dos algoritmos de Esteganografia. Posteriormente, apresenta-se um algoritmo criado pensando na robustez, no qual realizaremos testes para verificar tal propriedade. A terminologia a ser adotada será a seguinte: A informação a ser inserida será chamada de *marca* no caso específico de Marcas D'água e *mensagem* para todos os outros casos de Esteganografia. O destino da marca/mensagem será chamado de *recepiente*. O recepiente antes da inserção da marca/mensagem será chamado de *recepiente original*.

Um pouco da história e da situação da Esteganografia e Marcas D'água são apresentados no Capítulo 2. Capítulo 3 contém os dois algoritmos citados no parágrafo anterior, assim como um pequeno estudo sobre as classificações dos algoritmos esteganográficos do mundo digital. Posteriormente, Capítulo 4 recupera o algoritmo mais robusto apresentado no Capítulo anterior, realiza simulações e analisa os resultados das mesmas de forma a comprovar a robustez do método utilizado. Finalmente, o Capítulo 5 conclui o trabalho demonstrando suas contribuições e apresentando sugestões para trabalhos futuros.

2 ESTEGANOGRAFIA E MARCAS D'ÁGUA

Este Capítulo tratará sobre Esteganografia e Marcas D'água em geral. Na Seção 2.1 iremos falar sobre como tudo começou, a história da Esteganografia e Marcas D'água e em que ponto as duas técnicas começaram a se correlacionar. Seção 2.2 iremos falar aonde as duas técnicas se encontram hoje e como estão sendo utilizadas.

2.1 Um Pouco de História

Na introdução mostramos como a Esteganografia funciona no mundo digital, porém seu nascimento data de muito tempo atrás. O ser humano sempre teve e sempre terá necessidade de esconder informações (KAH 96). O primeiro relato de uso de métodos esteganográficos vem da Grécia, aonde Herodotus descreve em seu livro de histórias (HER 98) que Histiaeus, desejando enviar uma mensagem para instigar a revolta contra o rei para Aristagoras, raspa a cabeça de um dos seus escravos mais confiáveis, tatua a mensagem em seu escalpo, e espera seu cabelo crescer de volta. Assim que o cabelo cresce, ele envia o escravo para Miletus com uma única ordem — de dizer para Aristagoras raspar e examinar sua cabeça. Outro método também descrito no livro de Herodotus foi o utilizado por Demeratus para avisar Esparta que Xerxes pretendia invadir a Grécia. Ele retirou a cera de uma placa de escrita e escreveu a mensagem diretamente na madeira. Posteriormente ele reinseriu a cera na placa, de forma que ela parecesse inutilizada. Desta forma, observadores desavisados imaginavam somente ver uma placa sem qualquer informação inserida.

Esteganografia Linguística, também conhecida como *acróstico*, se resume em esconder informação em textos seguindo algum tipo de padrão, por exemplo, formando a mensangem com a primeira letra de cada palavra do texto chamariz. Foi o método mais amplamente utilizado, já que a sua dificuldade é baixa comparada as outras técnicas. Ao longo da história podemos encontrar diversas publicações que fizeram o utensílio de tal técnica. Uma das mais famosas é o último capítulo do livro "Através do Espelho"' de Lewis Carroll, o qual se prestarmos atenção na primeira letra de cada linha do poema, descobriremos um acróstico indicando o verdadeiro nome de Alice, *Alice Pleasance Liddell*.

> **A** boat, beneath a sunny sky
> **L**ingering onward dreamily
> **I**n an evening of July -
>
> **C**hildren three that nestle near,
> **E**ager eye and willing ear,

Pleased a simple tale to hear -

Long has paled that sunny sky:
Echoes fade and memories die:
Autumn frosts have slain July.

Still she haunts me, phantomwise,
Alice moving under skies
Never seen by waking eyes.

Children yet, the tale to hear,
Eager eye and willing ear,
Lovingly shall nestle near.

In a Wonderland they lie,
Dreaming as the days go by,
Dreaming as the summers die:

Ever drifting down the stream -
Lingering in the golden gleam -
Life, what is it but a dream?

Outra forma de acróstico que surgiu posteriormente é o chamado *Cardan Grille*, no qual é utilizado uma grade que identifica os pedaços de palavras a serem usados para formar a mensagem secreta. Porém, este método acabou por não ser tão utilizado pela dificuldade que empunha na época de ter de se acertar corretamente a posição das palavras, assim como a necessidade da grade, que caso fosse perdida acarretaria na perda da mensagem também. A Figura 2.1 mostra um exemplo de Esteganografia utilizado este método.

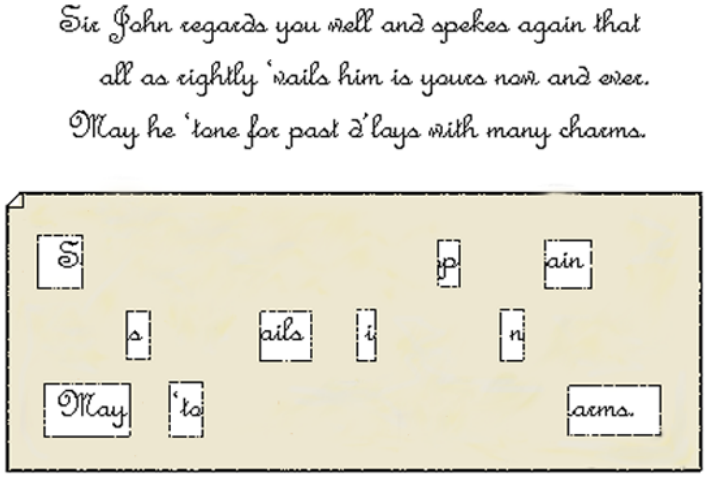


Figura 2.1: Mensagem escondida utilizando Cardan Grille.

Certamente, o uso de Esteganografia não escaparia às guerras. Durante a primeira guerra mundial, os alemães utilizaram-se dos *micropontos*, aonde a informação era diminuída de tal forma que se conseguia esconder desde imagens até uma página de texto completo em um ponto não muito maior do que um criado por uma máquina de escrever.

A eficiência do método foi tal que os Aliados descobriram estas informações somente em 1941. Além dessa, podemos citar a *tinta invisível*, onde uma carta de aparência inocente é escrita e entre suas linhas, ou em qualquer parte branca da mesma, era escrita a informação secreta utilizando-se da tinta especial, e.g., leite, e posteriormente revelada através de uma técnica específica para a tinta utilizada, no caso citado, aquecendo-se a carta.

A utilização de Marcas D'água é muito mais recente que Esteganografia[1]. Acredita-se que as primeiras foram criadas em 1282 na Itália, aonde utilizava-se arames durante a criação do papel criando-se padrões transparentes nas áreas aonde eram usados. Este estilo de marca é o que podemos chamar de *marca perceptível*, pois não há a necessidade de escondê-la. Apesar de ter surgido nesta época, foi somente por volta do século 18 que a Marca D'água assumiu o papel de registro, de forma a identificar a origem, intenção, ou mesmo autoria do objeto marcado. Variantes do método do arame foram criadas e são utilizadas até hoje. Para confecção das notas de 20 doláres, é utilizado um molde com pequeno relevo que cria um padrão único as notas.

Mais adiante, o primeiro sistema que se assemelha ao que temos hoje de Marca D'água digital foi criado por Emil Hermbrooke, aonde um sinal em código morse era inserido em uma música. Este é o primeira sistema de marca que podemos classificar como *marca imperceptível*, já que sua existência é invisível aos meios comuns. Com isso, apesar de ter sido utilizado na época para direitos autorais, esta técnica pode ser vista como o primeiro sistema de Marca D'água com possibilidade Esteganográfica, já que a podemos inserir qualquer tipo de informação no sinal da música.

2.2 A Era Digital

Apesar do número extenso de técnicas esteganográficas surgidas desde o começo da história humana, é agora, na era digital, que o estudo de Esteganografia ganhou maior aprofundamento. Enviamos toda hora arquivos pela Internet e muitas vezes sentimos a necessidade de privacidade, ou mesmo de marcar o que é nosso. A Criptografia serve aos seus propósitos, porém não conseguimos atingir um bom nível de privacidade quando as pessoas sabem que estamos compartilhando algo secreto. Dado tempo suficiente, um atacante pode descobrir o que estávamos compartilhando e com a tecnologia sempre em expansão, não estamos tão longe de ter super computadores domésticos que consigam quebrar boa parte dos algoritmos existentes. A Esteganografia entra neste ponto em conjunto a Criptografia, de forma a aumentar a segurança com a idéia de obscurecer o envio de informação secreta.

O Centro de Pesquisa e Análise de Esteganografia[2] estima que existem mais de 700 diferentes aplicativos de Esteganografia digital espalhados pela Internet. Nos últimos anos houve um grande crescimento de estudo de algoritmos. Acredita-se que um dos fatores deste acontecimento é relacionado à suspeita de que a comunicação sobre o ataque terrorista de 11 de Setembro foi realizada utilizando Esteganografia. Apesar de fazer mais de 8 anos desde o ataque, nada foi comprovado até o momento. Este é o grande fator crucial da Esteganografia na era digital. Você pode monitorar toda a rede, mas você nunca sabe o que é realmente um recepiente inocente e o que pode conter mensagens. *Tudo* é um recepiente em potencial. Desde pacotes IPs (MUR 2005), arquivos multimídia, ordem das tags em um arquivo de linguagem de marcagem como XML e HTML, entre muitos

[1] Por falta de material confiável disponível, muito da descrição sobre a história de Marcas D'Água descrita aqui é uma referência ao apresentado em (COX 2007)

[2] http://sarc-wv.com/

outros. Mesmo que você descubra que um arquivo contém uma mensagem, você ainda vai ter que descobrir de que forma ele se encontra lá. Os usos de Esteganografia podem variar entre um leque de fins legais e ilegais. Nesta monografia, iremos focar em como inserir uma Marca D'água em imagens.

3 MÉTODOS ESTEGANOGRÁFICOS

Este capítulo passa de forma mais especifícia os métodos esteganográficos para o mundo digital. Seção 3.1 apresenta como podemos classificar os algoritmos de Esteganografia, assim como introduz o conceito dos atributos principais relacionado a cada um deles. Seção 3.2 apresenta como uma imagem é representada no mundo digital, assim como introduz o algoritmo esteganográfico para imagens mais simples. Seção 3.3 fala um pouco mais sobre classificação, mas já começamos a entrar nos algoritmos focados em robustez. Seção 3.4 fala sobre os algoritmos que utilizam o domínio frequência e as Seções 3.5 e 3.6 introduzem um algoritmo robusto baseado em tudo que falamos anteriormente no Capítulo.

3.1 Classificação

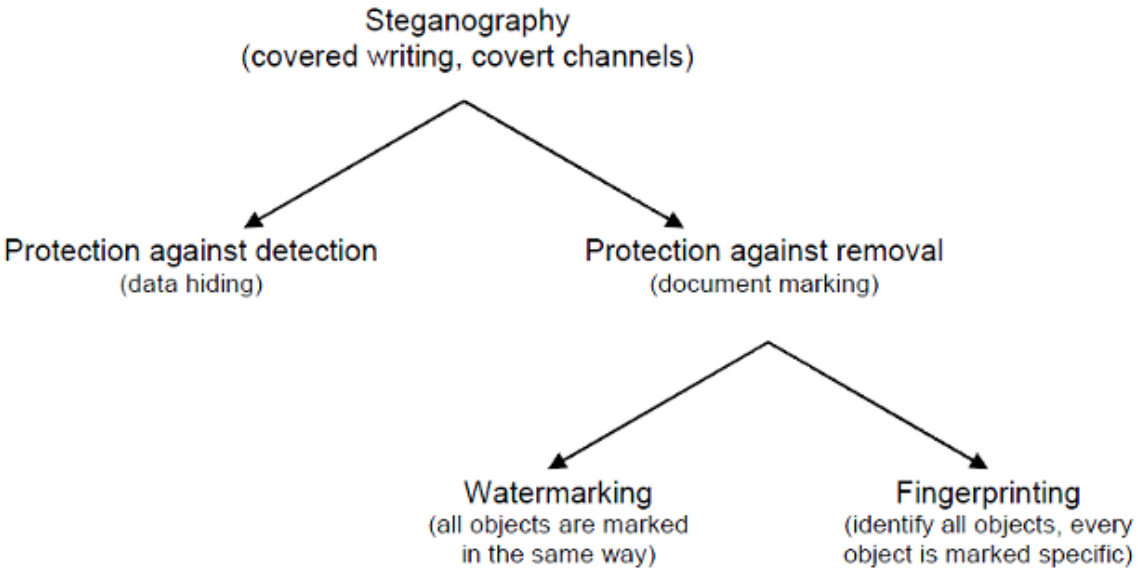


Figura 3.1: Tipos de Esteganografia.

Esteganografia pode ser classificada em dois métodos principais. *Linguística*, aonde utilizamos propriedades do recepiente para adicionarmos uma mensagem, por exemplo, os pixels de uma imagem ou a primeira letra de cada palavra; ou *técnica*, aonde o recepiente é utilizado somente como módulo "físico"para a mensagem, por exemplo, tinta invísivel ou tatuagem no escalpo. As variações dos algoritmos de Esteganografia repre-

sentados na Figura 3.1[1] utilizam-se de uma destas duas técnicas para atingir seu objetivo e são detalhados abaixo.

O primeiro, algoritmos de **proteção contra detecção** *(protection against detection)*, focam na invisibilidade da mensagem, assim como capacidade de carga do recepiente. Podemos incluir aqui o exemplo de Spion citado no Capítulo 1, necessitando enviar informações confidenciais sem ser descoberto. O segundo, algoritmos de **proteção contra remoção** *(protection against removal)*, são os que podemos relacionar com o exemplo das marcas em dinheiro, aonde a invisibilidade da mesma é bem menos importante que sua robustez. Este mesmo se divide novamente em duas categorias que podemos entender com os seguintes exemplos.

Imagine um autor que deseja que suas obras sejam sempre identificadas como suas. O que ele necessita é uma marca única que o identifique e que a mesma seja inserida em cada uma de suas criações. A isto damos o nome de **Marca D'água** *(watermark)*. Agora pense no caso também mostrado no Capítulo 1, das marcas em dinheiro. Caso utilizássemos uma Marca D'água para todas as notas, uma pessoa mal intencionada poderia apagar uma nota de 1 dinheiro e imprimir em cima uma de 100. Como não há identificação única para cada variação de nota, ela irá passar despercebida e a falsificação seria bem sucedida. Agora, adicionando-se uma marca única para cada nota, o que chamamos de **impressão digital** *(fingerprint)*, uma pessoa ao receber a nota falsificada descobriria o problema verificando a marca da mesma.

De fato, baseado no que já foi dito até o momento, podemos caracterizar a Esteganografia em três principais atributos (PRO 2003):

Capacidade: Quantidade de informação que conseguimos inserir dentro do recepiente original.

Invisibilidade(segurança): Fidelidade do recepiente esteganografado em relação ao recepiente original.

Robustez: O quanto que a informação inserida consegue sobreviver a modificações realizadas no seu recepiente.

Um algoritmo ideal possuiria os três atributos no valor máximo possível, entretanto em qualquer sistema do mundo real tal caraterística não consegue ser alcançada. Quando vamos implementar um algoritmo de Esteganografia devemos levar em conta o que desejamos que ele realize de acordo com o que é mais importante segundo os atributos apresentados acima. No caso da Marca D'água, é muito mais importante que o algoritmo ofereça robustez do que permita que insiramos o máximo possível de informação, ou mesmo que não seja detectado (apesar de ser um atributo desejável). Para termos uma idéia melhor do que está sendo afirmando, apresentamos a seguir um algoritmo de Esteganografia digital.

[1] Steganography and Digital Watermarking:
http://www.cs.bham.ac.uk/ mdr/teaching/modules03/security/students/SS5/Steganography.pdf

(a) Lenna, tamanho original: 512x512 (b) Baboon, tamanho original: 256x256

Figura 3.2: Lenna e Baboon original.

3.2 Um algoritmo não tão robusto

Como dito no Capítulo 1, iremos focar em algoritmos de Esteganografia para Marcas D'água em imagens. Para entedermos melhor o que vai ser discutido a partir daqui, necessitamos entender como um arquivo de imagem normalmente é representado em um sistema digital.

Segundo (GON 2008), uma imagem pode ser representada por uma função do tipo $f(x, y)$, onde x e y são coordenadas de um plano espacial e (x, y) é a *intensidade* da luminosidade e/ou cor naquele dado ponto. Em um imagem digital, o valor de cada ponto do plano é chamado de *pixel*. Tipicamente, cada pixel é representado por um valor de bits múltiplo de 8 e são organizados de acordo com um *espaço de cor* desejado. O espaço mais comumente utilizado é o **ARGB** *(Alpha Red Green Blue)*, aonde é usado um valor de 32 bits separado em 4 bytes. Cada byte representa a luminosidade (Alpha) ou uma das cores primárias (vermelho, verde, azul—RGB) daquele ponto da imagem. A luminosidade normalmente é associada a *opacidade*, e por ser representado por 8 bits assume valores de 0 a 255. Um valor 0 representa uma imagem transparente, i.e., invisível, e 255 representa uma imagem totalmente opaca, que é o que estamos acostumados a ver no mundo real. Os bytes de cor assumem também valores de 0 a 255 e representam a intensidade da mesma. Em HyperText Markup Language (HTML), estes bits são representados normalmente por uma sequência de 6 dígitos hexadecimais precedidos do símbolo $\#$, aonde cada par representa uma das cores primárias. Logo, se quisermos representar a cor verde pura, podemos utilizar a notação #00FF00. Com este sistema, conseguimos representar o total de 2^{24} $(16.777.216)$ cores diferentes, porém o olho humano não consegue distinguir a diferença visual entre todas elas.

O algoritmo de Esteganografia mais simples existente é o de *inserção no bit menos significativo*, mais popularmente conhecido como **LSB** (*Least Significant Bit*). O grande truque do algoritmo é explorar a deficiência citada no paragráfo anterior e inserirmos cada bit da mensagem no bit menos significativo de cada byte do pixel da imagem. Imagine a situação onde temos três pixels paralelos de cores verde, amarela, e vermelha como na Figura 3.3.

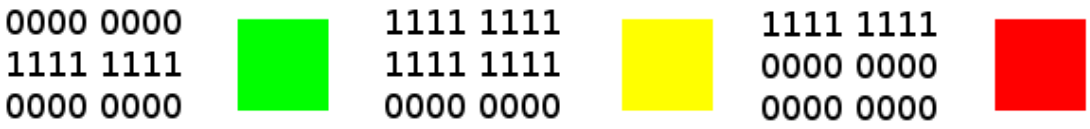


Figura 3.3: #00FF00, #FFFF00, #FF0000 representados visualmente

Se adicionar a letra "I"(código ASCII hexadecimal 49, em binário $0100\ 1001$) no bit menos significativo destes pixels obteremos um resultado como o apresentado na Figura 3.4.

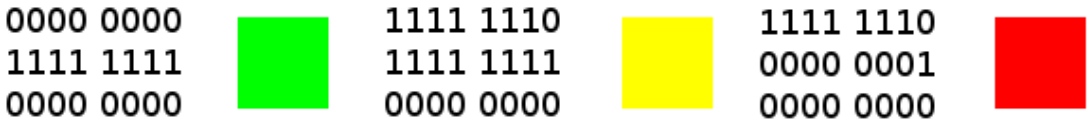


Figura 3.4: #00FF00, #FFFF00, #FF0000 após inserção LSB do valor hexa 49.

A mudança visual para o olho humano é imperceptível, entretanto os pixels agora carregam consigo a letra "I", comprovando a invisibilidade do algoritmo. Porém, ele é pouco robusto e possui baixa capacidade. Basta que alteremos todos os LSB novamente para que a mensagem seja perdida, e a quantidade de bits que podemos inserir é dada por

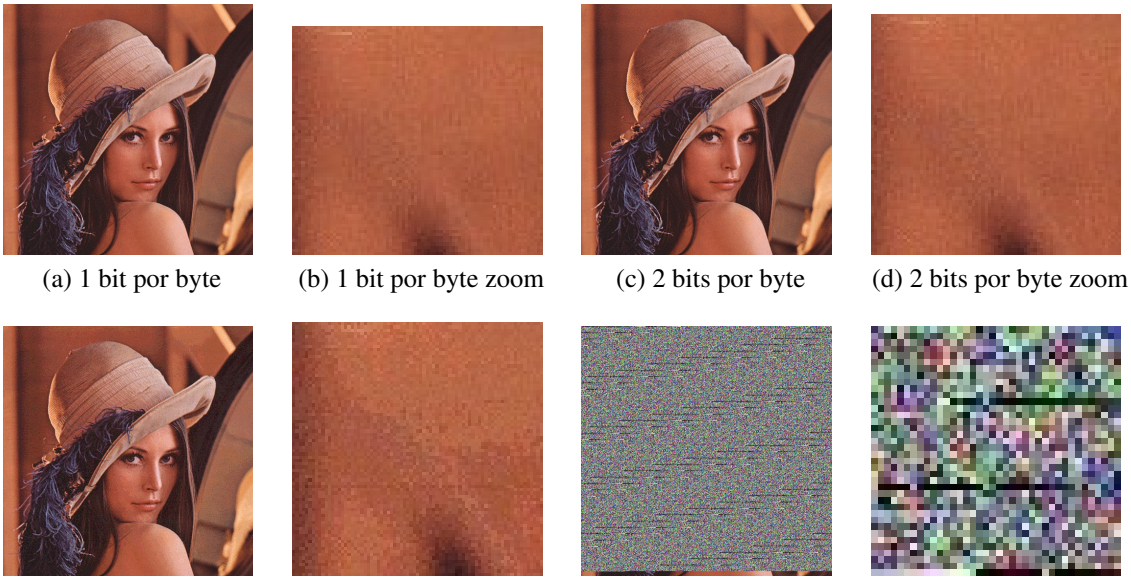
$$LARGURA \times ALTURA \times 3 \tag{3.1}$$

sendo 3 a constante igual ao número de cores primárias de uma imagem. Para resolver estes dois problemas, poderíamos extender o número de bits que inserimos em cada byte do pixel (capacidade dada pela fórmula 3.2), contudo tal propriedade só pode ser sustentada até um certo nível. Quando começamos a alterar muitos bits por pixel ganhamos robustez e capacidade, visto que será necessário um número maior de alterações para remover os bits de mensagem dos pixels, entretanto, a perda na invisibilidade é drástica, já que a imagem começa a apresentar alterações visuais.

$$LARGURA \times ALTURA \times 3 \times NR\_BITS\_PER\_BYTE \tag{3.2}$$

Para demonstrar esta propriedade, vamos pensar num exemplo básico. Iremos utilizar as clássicas imagens de Lenna e Baboon, apresentadas nas Figuras 3.2a e 3.2b, respectivamente, e o programa criado nesta monografia para inserir a segunda Figura na primeira usando LSB. O programa será apresentado por completo posteriormente, mas por enquanto basta sabermos que ele é chamado *Caesar's Mark* e permite que utilizemos o algoritmo LSB escolhendo inserir 1, 2, 4 ou 8 bits por byte de pixel de imagens de qualquer tipo lossless[2] registrados na Java Virtual Machine (**JVM**) de sua máquina[3]. Além disso, para termos uma robustez maior ainda, o algoritmo repete a mensagem no recepiente por um número máximo de vezes que ele consiga armazenar. A Figura 3.5 apresenta o resultado da execução do programa para inserção da mensagem com todos os

[2]Que não possuem perda de informação durante (de)compressão da imagem
[3]Por padrão, toda JVM possui suporte a imagens do tipo jpg, bmp, wbmp, png e gif



(a) 1 bit por byte (b) 1 bit por byte zoom (c) 2 bits por byte (d) 2 bits por byte zoom

(e) 4 bits por byte (f) 4 bits por byte zoom (g) 8 bits por byte (h) 8 bits por byte zoom

Figura 3.5: Lenna (Figura 3.2a) após inserção de Baboon (Figura 3.2b) como mensagem utilizando números diferentes de bits por byte do pixel.

números permitidos de bits. Aqui conseguimos enxergar bem o que haviamos afirmado anteriormente. Assim que começamos a aumentar o número de bits, começamos a notar perturbações cada vez maiores na imagem (apesar de ainda quase imperceptíveis), até chegar ao ponto em que ela acaba por ser totalmente sobreescrita e gerar uma imagem como a representada na Figura 3.5g. Apesar de ser a mais robusta, um algoritmo o qual troquemos todo os bytes do pixel é inaceitável, pois foge do escopo aonde desejamos a obscuridade da mensagem. Assim como um algoritmo que apesar de invísivel, é facilmente destruído por quase todos os tipos de modificações na imagem. Necessitamos descobrir um algoritmo que ofereça robustez e invisibilidade em níveis adequados para conseguirmos chegar no nosso objetivo final que é a inserção de uma Marca D'água.

3.3 A busca por um bom algoritmo robusto

Diferentes tipos de algoritmos foram criados desde que a Esteganografia entrou no mundo digital[(MOR 2000), (YOU 2008), (KER 2005), (GUA 2004), etc].O artigo (ROC 2008) oferece uma boa separação das técnicas de Esteganografia existentes, assim como maneiras de aumentar a segurança e, em consequência, a robustez de um bom algoritmo. O que se percebe é que as técnicas mais utilizadas são as baseadas em *espalhamento espectral* (**Spread Spectrum Image Steganography — SSIS**).

A idéia básica por trás dessas técnicas é transformar a imagem do domínio espacial para o domínio de frequência, inserir a marca e então fazer o caminho de volta. Consequentemente, a marca estará espalhada por toda a imagem, pois um coeficiente modificado no domínio frequência tem seu valor espalhado por todos os pontos do domínio espacial. A técnica funciona perfeitamente, pois não há perdas[4] durante a transformação de um domínio para outro. Uma boa analogia é dada por Richard G. Baldwin[5], aonde

[4]Desconsiderando possíveis perdas por arredondamento.

[5]http://www.dickbaldwin.com

podemos comparar esta transformação com a da água: "Se esfriarmos a água o suficiente, iremos transformá-la para o estado sólido. Se a esquentarmos, ela voltará a ficar líquida. A mesma informação é representada nos dois casos, porém de formas diferentes."

Para realizarmos a transformação, são utilizados principalmente dois métodos: *Transformação de Cosseno Discreta* (**Discrete Cosine Transform — DCT**) ou *Transformação Rápida de Fourier* (**Fast Fourier Transform — FFT**). As duas técnicas funcionam de formas parecidas. A transformação é definida de forma a mapear os dados espaciais correlacionados em coeficientes não-correlacionados. Ambos são utilizadas para diferentes aplicações, entretanto DCT é o mais popular, visto que possui computação muito mais simples e maior efetividade para compressão de dados multimídia. De fato, ele é utilizado no algoritmo de compressão de imagens *Joint Photographic Experts Group* (**JPEG**) (GON 2008) e é a ferramenta necessária para definir o que pode ser perdido de informação. Para conseguirmos chegar em um algoritmo robusto, devemos entender o funcionamento do DCT na compressão JPEG.

A imagem é divida em cores e blocos de 8 por 8 (64 pixels). Em cada bloco é aplicado DCT (representada na Figura 3.6), resultando num bloco de coeficientes, onde o primeiro é conhecido como *DC* e representa o valor médio. Todos os outros coeficientes são conhecidos como *AC*. Aqui entra a parte importante: na maior parte das imagens os valores com sinais mais fortes ficam presentes nas *baixas frequências*, que se encontram perto do coeficiente DC na parte superior à esquerda do bloco transformado. Isto é, se os alterarmos de forma drástica, provavelmente teremos mudanças visuais significativas na imagem. Em contraposição, os coeficientes de mais *alta frequência*—encontrados mais abaixo e a direita do bloco—possuem uma força muito menor na imagem. A compressão leva em conta esta propriedade e aplica uma *matriz de quantização* aonde normalmente os coeficientes de alta frequência acabam por serem zerados, enquanto os outros permanecem em sua maioria inalterados. Logo, se desejamos inserir mensagens que sobrevivam a compressões, e que tenham uma alta resiliência a modificações, será necessário criar um algoritmo que consiga *inserir uma marca nos coeficientes de baixa frequência do dominio frequência*.

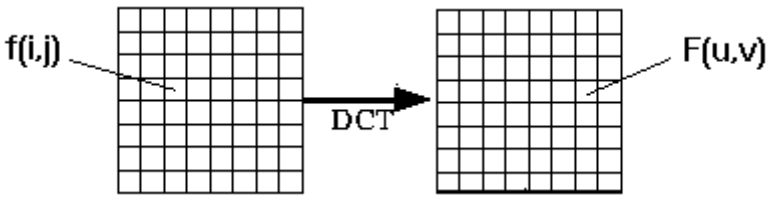


Figura 3.6: Transformação de um bloco 8x8 usando DCT.

3.4 Usando o domínio frequência

Muitos algoritmos baseados em DCT foram propostos levando em conta o que discutimos até o momento. Entretanto, a maioria deles necessita do recepiente original para a extração da marca inserida. Esta propriedade é considerada indesejada, pois um agressor obtendo uma cópia deste recepiente, pode utilizá-la para descobrir cópias marcadas, ou pior ainda, usar o recepiente inalterado e marcá-lo como se fosse seu.

No artigo (CHU 2003) é proposto um esquema baseado em *subamostra* para passar

por cima deste impedimento. A idéia é relativamente simples. Suponha I denota o recepiente original de tamanho Wx Y, $\{(x, y) \| x \in W \div 2, y \in H \div 2\}$, e a partir de I são geradas 4 subamostras $\{I_i, i \in (1, 2, 3, 4)\}$ utilizando-se a seguinte fórmula:

$$I_1 = 2x \times 2y \tag{3.3}$$
$$I_2 = 2x \times (2y + 1) \tag{3.4}$$
$$I_3 = (2x + 1) \times 2y \tag{3.5}$$
$$I_4 = (2x + 1) \times (2y + 1) \tag{3.6}$$

O que (CHU 2003) assume é que ao calcularmos o DCT dessas 4 subamostras, obteremos resultados com valores aproximados, i.e., $\{C_i = DCT(I_i) \| i \in (1, 2, 3, 4)\}$ temos $\{C_i \approx C_j \| i, j \in (1, 2, 3, 4), i \neq j\}$. As alterações na imagem normalmente afetarão muito pouco esta propriedades e as diferenças permancerão constantes, e.g., se o DCT de C_1 tiver um valor maior que C_2, após alterações o valor de C_1 normalmente continuará maior que C_2. Baseado neste fato e no algoritmo proposto por (LU 2006), utilizaremos uma técnica conhecida como *seleção de coeficientes*.

A técnica consiste em selecionar k primeiros coeficientes de uma imagem de acordo com um padrão e inserir a marca nestes coeficientes seguindo um função f que leva em conta um valor de *força* normalmente denominado como α. Valores altos de α garantem uma maior robustez, mas diminuem a invisibilidade da marca. Juntamos está técnica com subamostras e obteremos um algoritmo pouco invasivo e de alta robustez.

3.5 Inserção da Marca

Porque o algoritmo é pouco invasivo? Porque a nossa inserção de bits de marca será realizada baseada na comparação dos coeficientes das subamostras. Para melhor entendermos, vamos seguir como uma marca será inserida passo a passo.

3.5.1 Em que cor inserir a marca?

Do que foi explicado até o momento, o leitor pode imaginar que desejamos calcular o DCT de uma imagem para inserirmos a marca desejada. Entretanto, para chegarmos a tal objetivo devemos escolher em qual das cores primárias iremos realizar a transformação. Segundo as características do *sistema visual humano* (**Human Visual System—HVS**), o olho é muito mais sensitivo a cor verde, i.e., inserir a marca no espectro verde da imagem nos garantiria maior robustez, pois algoritmos como JPEG levam em conta tal propriedade. Se inserirmos no espectro azul, garantiríamos maior invisibilidade, pois o HVS é muito pouco sensível a esta cor. Todavia, cada imagem possui características únicas e ao escolhermos uma cor única poderíamos acabar por perder invisibilidade/robustez por que a imagem possui pouco dela. Por exemplo, se utilizássemos o espectro verde e inseríssimos a marca em uma imagem de uma floresta obteríamos uma grande robustez/invisibilidade. Porém, se esta foto fosse do oceano, correríamos o risco da marca se tornar mais visível e pouco robusta. Levando este fator em conta, decidimos que a melhor maneira de garantirmos uma boa robustez e invisibilidade seria espalhando a marca por todas as cores de maneira relativamente igual. Para isto, convertemos a imagem do formato RGB para *Luma, Red Diference Chroma and Blue Diference Chroma* ($Y'C_BC_R$).

$Y'C_BC_R$ não é considerado um formato absoluto de cor, mas sim uma maneira de codificarmos o formato RGB. Pense no exemplo dado na Seção 3.3 sobre a água. A

mudança entre RGB e $Y'C_BC_R$ pode ser visto da mesma maneira, aonde o primeiro enxerga a imagem de acordo com as cores primárias, enquanto o segundo leva em conta a luminosidade (luma) e a diferença das cores vermelha (C_R) e azul (C_B) em comparação a esta luminosidade. A Equação 3.9 mostra como convertemos de um formato para o outro.

$$Y = ((0.299 \times R) + (0.587 \times G) + (0.114 \times B)) \tag{3.7}$$
$$C_B = (128 + (-0.169 \times R) + (-0.331 \times G) + (0.5 \times B)) \tag{3.8}$$
$$C_R = (128 + (0.5 \times R) + (-0.419 \times G) + (-0.081 \times B)) \tag{3.9}$$

A Figura 3.7[6] mostra a Figura 3.2a separada nos 3 aspectos do formato $Y'C_BC_R$. Nota-se que Y' contém as partes mais significativas para o olho humano, a luminosidade da imagem. De fato, este formato é usado para que TVs em preto e branco ainda consigam processar o que deve ser apresentado, podendo ignorar os chromas vermelho e azul da transmissão. Utilizaremos esta propriedade do Luma, de carregar a parte importante para o HVS, para inserir nossa marca de forma garantir que ela seja robusta e esteja bem espalhada entre todas as cores.

(a) Canal Y' (b) Canal Cb (c) Canal Cr

Figura 3.7: Lenna (Figura 3.2a) separada nos três canais $Y'C_BC_R$.

3.5.2 Inserindo a Marca

O sistema de inserção da marca é apresentado na Figura 3.8. Logo, dada uma imagem em RGB I e uma Marca D'Água m, podemos inserir a marca seguindo os passos abaixo:

1. Convertemos a imagem I do espaço de cor RGB para $Y'C_BC_R$ gerando uma imagem I'.

2. Decompomos I' seguindo a equação 3.6, de forma a gerar 4 subamostras: I_1, I_2, I_3, I_4.

3. Dada uma chave secreta, geramos um array de seleção aleatória de coeficientes S com tamanho igual ao número de bits da Marca D'água m, onde para cada bit b de m temos

$$\{S_b = \{i, j\} \| i, j \in (1, 2, 3, 4), i \neq j\} \tag{3.10}$$

[6]Separação realizada utilizando ImageMagick (www.imagemagick.org)

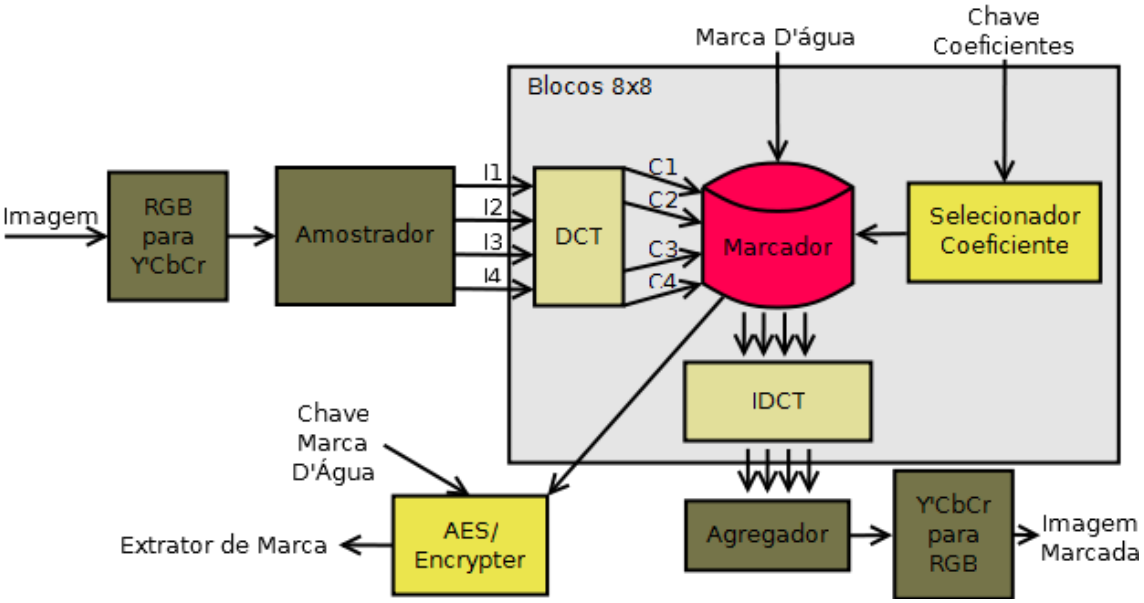


Figura 3.8: Diagrama de inserção da Marca D'água.

4. Para k bits da Marca D'água, onde $\{k, I_i \| i \in (1,2,3,4), 0 \leq k < MIN(sizeof(m), 63)\}$ extraímos um bloco de 8x8 bits das subamostras e entramos como dados para o segundo passo:

   (a) Para cada bloco calculamos o seu DCT, gerando blocos coeficientes $\{C_i \| i \in (1,2,3,4)\}$.

   (b) Organizamos cada bloco C_i em um array Z_i de 64 coeficientes em ordem de importância visual, i.e., coeficientes com sinal mais forte primeiro. Figura 3.9 apresenta como esta ordem é obtida[7].

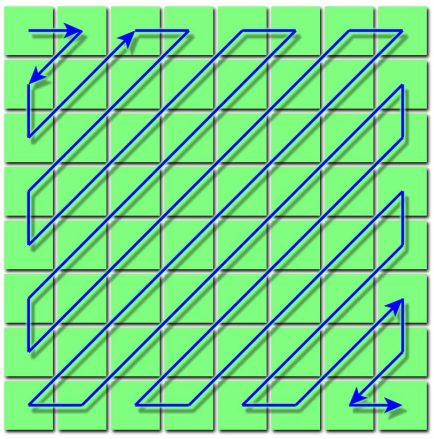


Figura 3.9: Ordem zig-zag de coeficientes.

   (c) Para $\{n \| 0 \leq n < k\}$ selecionamos o $n$ésimo bit da marca ainda não inserido m_n, e o $n$ésimo elemento do array de seleção de coeficientes $S_n = (i, j)$.

[7] O mesmo bloco é usado no algoritmo de compressão JPEG para determinação quais são os coeficientes de maior importância.

Como $\{(i,j) \| i,j \in (1,2,3,4), i \neq j\}$ obtemos a referência de quais subamostras devemos recuperar os coeficientes a serem utilizados. Com esta informação, selecionamos os coeficientes da posição $n+1$[8] das subamostras onde

$$V_i = \begin{cases} Z_j, m_n = 1 \wedge Z_i < Z_j \\ Z_i, otherwise \end{cases} \tag{3.11}$$

$$V_j = \begin{cases} Z_i, m_n = 0 \wedge Z_i > Z_j \\ Z_j, otherwise \end{cases} \tag{3.12}$$

(d) Tendo V_i e V_j, calculamos

$$V = |V_i| + |V_j| \tag{3.13}$$
$$D = V_i - V_j \tag{3.14}$$

e finalmente a inserção, caso $\left|\frac{D}{V}\right| \leq \beta$ seja verdade, ajustamos os coeficientes da seguinte forma

$$V'_a = V_i + \alpha(2m_b - 1)|\frac{V}{2}| \tag{3.15}$$
$$V'_b = V_j - \alpha(2m_b - 1)|\frac{V}{2}| \tag{3.16}$$

Após ajuste, reinserimos os coeficientes de volta às subamostras de onde as recuperamos. Como citado anteriormente na Seção 3.4, α representa a força com que inserimos a marca dentro da imagem. Se aumentarmos seu valor, teremos maior robustez, mas menor invisibilidade. O valor β representa um limite desejado de diferença entre os coeficientes das subamostras.

(e) Calculamos o DCT inverso dos blocos utilizando os novos valores de coeficientes. Se ainda não inserimos todos os bits da marca, selecionamos novos k bits a partir da posição do último bit inserido, extraímos novos blocos das subamostras e repetimos os passos a partir de 4a.

5. Finalmente, agregamos todas as subamostras em uma imagem I'' e a convertemos de volta para o espaço de cor RGB, obtendo-se um imagem final marcada M.

6. Salvamos o array de seleção de coeficientes em um arquivo. Realizamos isto por causa das trocas realizadas durante a escolha de qual subamostra deveríamos obter os coeficientes. Onde, o que realmente desejávamos nas equações 3.11 e 3.12, era a relação direta de V_i com Z_i e V_j com Z_j.

   Enfim, para garantir maior segurança da marca, codificamos o arquivo gerado com Criptografia *Advance Encryption Standard* (**AES**) de 128 bits[9].

[8] Ignoramos o coeficiente DC, pois alterá-lo pode acarretar em mudanças visuais significativas.

[9] A escolha por 128 bits e não uma Criptografia mais forte foi realizada por causa da política de alguns países em relação a chaves mais fortes.

Com isso, obtemos um algoritmo que consegue inserir uma Marca D'água com o total de bits igual a

$$\frac{(W \times H)}{63} \times k \tag{3.17}$$

onde k é o número de bits inseridos por bloco da imagem, chegando no máximo a 63 bits por bloco (64 coeficientes menos o DC), W é a largura, e H a altura da imagem.

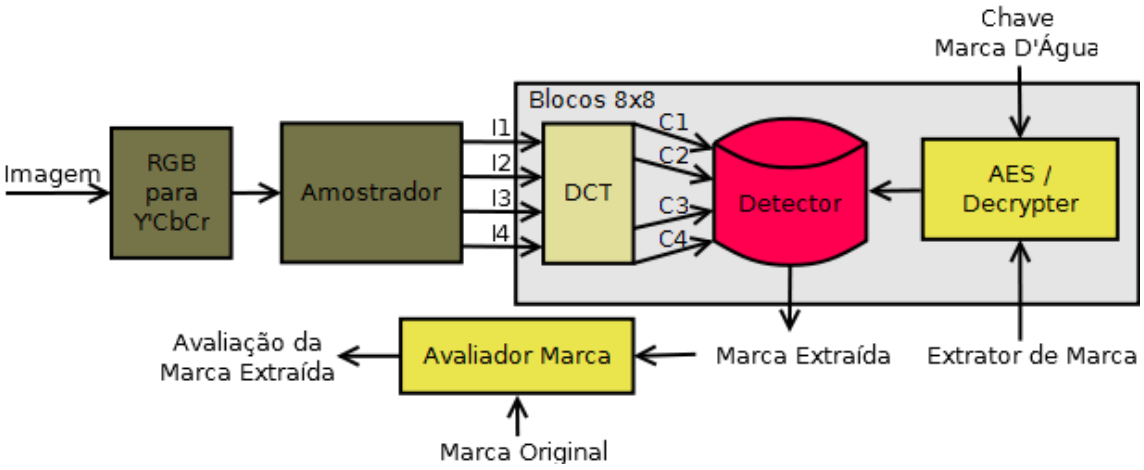


Figura 3.10: Diagrama de detecção da Marca D'água.

3.6 Detecção da Marca

A imagem marcada pode ser modificada intencionalmente ou não. Por causa deste tipo de situação é que criamos um algoritmo robusto que já não espera que consigamos recuperar todos os bits inseridos, mas que garante que não haverá remoção completa de todos. O sistema de detecção de marca é representado na Figura 3.10.

A extração da marca é praticamente igual ao sistema de inserção. Ainda necessitamos transformar a imagem do espaço de cor RGB para $Y'C_BC_R$, extrair blocos, calcular o DCT e organizá-los em ordem zig-zag. Porém, como não iremos reinserir a marca, mas simplesmente detectá-la, não há necessidade de realizarmos o passo inverso. Além disso, necessitamos do arquivo de extração de marca gerado no passo de inserção e a senha utilizada para encriptá-lo. Ele irá abstrair a criação do array de seleção de coeficientes. Com isto, conseguimos extrair a marca bit por bit seguindo a equação

$$m_n = \begin{cases} 1, S_i \geq S_j \\ 0, otherwise \end{cases} \tag{3.18}$$

onde m_n representa o $n$ésimo bit da Marca D'água, e S_i e S_j os coeficientes aonde supostamente deveria se encontrar este bit. Com isto temos uma marca extraída. Se desejarmos ainda avaliar a relação desta marca com a originalmente inserida, pode-se comparar as duas através de um avaliador. Atualmente, ele retorna a porcentagem de bits iguais entre as duas marcas.

4 RESULTADOS EXPERIMENTAIS

Neste Capítulo iremos apresentar uma série de testes e análises do algoritmo robusto apresentado no Capítulo anterior nas Seções 3.5 e 3.6. Na Seção 4.1 iremos demonstrar de forma geral o programa criado para este trabalho de conclusão. Seção 4.2 começamos a realizar testes de ataques à imagem marcada e fazemos uma análise da robustez do algoritmo.

4.1 Caesar's Mark

Caesar's Mark é o programa criado para esta monografia. Seu nome é baseado na famosa *Cifra de Caesar*[1], um dos primeiros sistemas conhecidos de Criptografia, e o nome do autor Caesar Ralf.

o programa foi criado utilizando-se a tecnologia Java, atualmente na versão 1.6, e suas bibliotecas de manipulação de imagem. Atualmente, o código do programa está liberado utilizando a licença *GNU General Public License v3*[2] e está sendo hospedado no repositório público de códigos do google[3]. Ainda não possui interface visual, mas pode ser utilizado através de linha de comando. Entre os comandos mais importantes podemos destacar:

-s,–steganography <stegonagraphy_type>: Qual algoritmo de Esteganografia desejamos usar. Atualmente pode se escolher entre LSB e DCT.

-u,–unapply: Se desejamos extrair um marca. Caso omitido, iremos aplicar Esteganografia.

-e,–embedFile <file_path>: Caminho para marca/mensagem que desejamos inserir no recepiente.

-m,–msg <message>: Texto que desejamos inserir como marca/mensagem no recepiente.

-o,–output <file_path>: Caminho para onde desejamos salvar o resultado da aplicação/ extração da marca/mensagem.

-p,–plainFile <file_path>: Caminho para o recepiente original/marcado, dependendo se estamos aplicando/extraindo marca/mensagem.

[1] http://www.cs.trincoll.edu/ crypto/historical/caesar.html
[2] http://www.gnu.org/licenses/gpl.html
[3] http://code.google.com/p/wmsteganography/

-b,–bits_per_block <number>: Indica número de bits que se deseja por bloco/byte. LSB [padrão = 1, permitido = (1, 2, 4, 8)]. DCT [padrão = 4, permitido = (0..63)]

-c,–dct_coefficient_key <number>: Chave para criação do array de seleção de coeficientes. Se omitido, será usado tempo referente ao momento de criação da marca/ mensagem.

-d,–dct_wm_extraction <file_path>: Caminho para salvar o arquivo de extração de marca gerado após a execução do DCT. Se omitido, será usado o mesmo da marca original com a extensão ".wdct".

-k,–dct_key <string>: Chave para ser usada para criptografar o arquivo de extração de marca.

-a,–dct_strength <number>: Valor que indica a força da inserção do algoritmo DCT. Se omitido, o valor 0.05 é usado.

4.2 Simulação

Para testar a robustez da marca utilizando o algoritmo DCT, se usará a Figura 3.2a de Lenna como recepiente. A Figura binária 4.1b de tamanho 48x48 será utilizada como Marca D'água a ser inserida. O programa permite a escolha de quase todos os parâmetros para inserção da marca, mas para todas simulações apresentadas neste Capítulo, caso não seja especificado, será utilizado um valor de $\alpha = 0.05$ e 4 bits da marca por bloco. Por padrão, o valor de β é sempre igual a 0.05. A Figura 4.1a mostra a imagem Lenna após a inserção da marca.

(a) Lenna marcada.

(b) Marca D'água original

Figura 4.1: Lenna (Figura 3.2a) após a inserção da marca (Figura 4.1b) usando DCT.

Visualmente a imagem marcada não é diferente da original. O olho humano não consegue perceber as poucas diferenças introduzidas pela marcação. Uma melhor avaliação do recepiente da marca pode ser feita utilizando a métrica conhecida como *relação do pico de sinal-ruído* (**PSNR**) muito utilizada para comparação de imagens comprimidas com a original. O valor retornado de PSNR é dado em decibéis (dB), e quando este é

maior que 30 afirma-se que a compressão resultou em uma imagem de boa qualidade e visualmente equivalente à original. Quanto maior o valor do PSNR, maior fidelidade a imagem possui. O PSNR de Lenna após a marcação é igual à 40.741, um valor ótimo confirmado pela invisibilidade da marca e semelhança da mesma com a original. Entretanto, a marca realmente está lá?

Um agente mal intencionado pode atacar uma imagem em que acredita haver uma marca de diferentes maneiras. As técnicas mais amplamente utilizadas na Internet são conhecidas como *ataques processamento de sinal*. Elas podem ser divididas em várias categorias, porém uma das técnicas mais utilizadas é a de compressão. A idéia é simples e consiste em utilizar quaisquer algoritmos de compressão do tipo lossy no provável recepiente da marca, e.g., salvar a imagem JPEG com uma qualidade menor. Com isto, o atacante espera que a marca seja completamente destruída. Para muitos algoritmos, isto acaba ocorrendo por causa da grande variação e perda dos dados ocorridas na imagem. Por consequência, um bom sistema esteganográfico deve ser criado tendo em mente que não poderá garantir a recuperação completa de todos os dados de uma marca. Portanto, para este trabalho irá se considerar o aspecto visual da marca extraída, assim como um requisito mínimo de 65% bits corretos para afirmarmos que uma dada marca se encontrava realmente contida no recepiente.

Além do ataque de compressão, os dois ataques mais comumente utilizados são os de *ruído branco Gaussiano* e *Salt and Pepper*. Estes dois podem ocorrer de forma passiva, por problemas da rede em que o recepiente está passando aonde valores de pixels são trocados, ou interferência no sinal causando adição de informação desnecessária.

A Figura 4.2 mostra marcas extraídas após diferentes ataques de processamento de sinal junto com a quantidade de bits iguais à marca original.

(a) 98% bits iguais

(b) 79% bits iguais

(c) 71% bits iguais

(d) 68% bits iguais

(e) 93% bits iguais.

(f) 97% bits iguais.

Figura 4.2: Marcas D'águas extraídas após ataques de compressão. (4.2a) Qualidade 100%. (4.2b) Qualidade 80%. (4.2c) Qualidade 40%. (4.2d) Qualidade 10%. (4.2e) Salt and Pepper 0.01. (4.2f) Ruído branco gaussiano 0.1.

De acordo com o visto na Figura, o algoritmo se mostrou muito robusto contra este tipo de ataques. Com um recepiente com qualidade igual a 10%, o programa conseguiu recuperar uma marca visualmente comparável a original e com o equivalente a 70% de bits corretos. Podemos atribuir esta robustez à escolha de um valor baixo de bits por bloco, resultando no uso de coeficientes de baixa frequência, que como visto anteriormente no Seção 3.4 permanecem praticamente inalterados durante a quantização dos mesmos. Caso tivéssemos escolhidos números maiores de bits por bloco, obteríamos uma maior capaci-

dade, porém a robustez da marca começaria a decair exponencialmente. Figura 4.3 mostra o resultado da extração em imagens com 40% de qualidade da original, porém variando-se o número de bits por bloco.

(a) 8 bits de marca por bloco.

(b) 24 bits de marca por bloco.

(c) 40 bits de marca por bloco

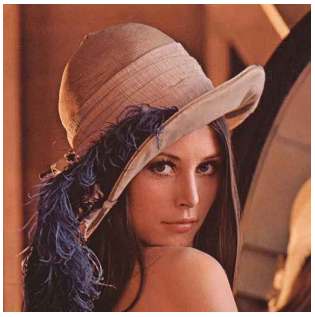
(d) 56 bits de marca por bloco

Figura 4.3: Marcas D'águas extraídas após ataque de compressão de 40% qualidade, com número de bits por bloco variável.

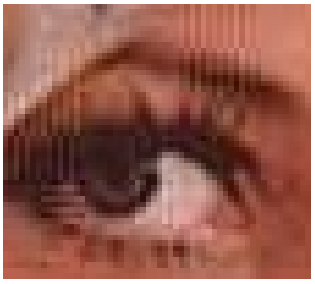
Averigua-se que a robustez da marca decai muito. Com 8 e 24 bits por bloco ainda consegue-se discernir a marca, porém a partir de 40 bits o resultado faz com que a imagem pareça ter sido gerada a partir de uma escolha aleatória de bits. Como dito anteriormente, isso acontece porque a marca entrou na área de coeficientes de mais alta frequência, que durante a compressão acabam por serem zerados. Para resolver parte do problema, pode-se aumentar a força em que a marca está inserida. A Figura 4.4 mostra a imagem marcada assim como o resultado de extração da marca de 4 bits por bloco do recepiente com 40% da qualidade do original, mas com valores de $alpha$ variados.

Verifica-se que a robustez aumentou, porém a invisibilidade da marca diminui bastante. Tanto na primeira, quanto na segunda Figura de Lenna pode-se notar "blocos"na maior parte da onde a marca foi inserida. Isto acontece porque se está aumentando a diferença entre os coeficientes, causando uma variação maior nos valores dos pixels. Por um ganho de aproximadamente 2%, acredita-se que este aumento de robustez não vale a degradação gerada no recepiente. Outra propriedade interessante é que se a força for aumentada para valores muito altos, a variação nos pixels será tanta que irá se acabar degradando demais a imagem. Consequentemente, boa parte da marca já será perdida na inserção, visto que a degradação acaba por "zerar"os pixels, retirando a diferença entre eles, que é a base deste método.

Diferentes formas de ruídos podem ser inseridas em uma imagem para destruir o conteúdo previamente inserido nela. Normalmente, tais ruídos afetam muito pouco uma imagem visualmente, porém inserem um bom número de alterações na estrutura da mesma. Uma forma bastante utilizada de inserir ruído é a utilização do mesmo método esteganográfico para inserção de marcas diferentes no mesmo recipiente. Desta forma, acredita-se que somente a última marca irá sobreviver ao ataque, destruindo as inseridas anteriormente. Pense no caso do algoritmo LSB, como todas marcas inseridas afetam os mesmos bits, fatalmente somente a última marca irá sobreviver. Realizou-se o mesmo tipo de ataque ao algoritmo com as marcas ilustradas na Figura 4.5, inserindo em uma imagem



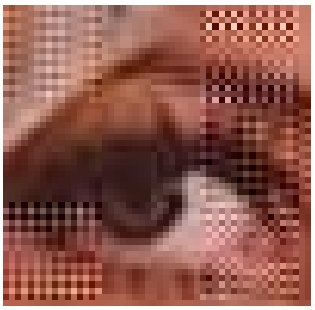
(a) Lenna para $\alpha = 0.5$



(b) Zoom $\alpha = 0.5$

(c) Marca para $\alpha = 0.5$

(d) Lenna para $\alpha = 2$



(e) Zoom $\alpha = 2$

(f) Marca para $\alpha = 2$

Figura 4.4: Lenna após inserção de marcas com valores de α variados e 40% qualidade original. (4.4a) ilustra Lenna marcada com $\alpha = 0.5$ ($10\times$ valor padrão). (4.4b) é um zoom da Figura 4.4a. (4.4c) é a marca extraída com $\alpha = 0.5$ com 74% bits corretos. (4.4d) ilustra Lenna marcada com $\alpha = 2$ ($40\times$ valor padrão). (4.4e) é um zoom da Figura 4.4d. (4.4f) é a marca extraída com $\alpha = 2$ e 73% dos bits corretos.

bitmap com já marcada com a Figura 4.1b.

O sucesso obtido para extração das marcas inseridas em ordem de foi de 96%, 97%, 97%, e 99%. Logo, pode-se concluir que o algoritmo mostrou uma ótima robustez contra a inserção de diferentes marcas, já que a extração de todas elas foi realizada com sucesso.

Por fim, comportamento semelhante é esperado em diferentes imagens, visto que os ataques simulados aqui as afetam da mesma maneira. Consequentemente, o uso de diferentes figuras acarreta somente em variações desconsideráveis para este estudo de ganho e/ou perda de robutes, invisibilidade e/ou capacidade.

(a) Lena após inserção das 4 marcas.

(b) Marca 1.

(c) Marca 2.

(d) Marca 3.

Figura 4.5: Marcas a serem inseridas como forma de ruído.

5 CONCLUSÃO

Este trabalho foi desenvolvido com o intuito de se aprender como podemos criar um algoritmo esteganográfico robusto para imagens. Para tanto, foram implementados dois algoritmos bastante diferentes em técnicas. O primeiro, conhecido como LSB, foi utilizado num primeiro momento para facilitar o entendimento dos 3 principais atributos da Esteganografia—Capacidade, Invisibilidade, e Robustez—e como eles se relacionam. Verificou-se que o LSB apesar de oferecer alta invisibilidade, não consegue chegar num nível de robustez satisfatório, pois removê-lo é relativamente simples.

Então, começou-se um pequeno estudo sobre algoritmos focados em robustez, aonde averiguou-se que a maior parte das técnicas robustas são baseadas em espalhamento espectral. Existem diferentes formas de utilizá-las, porém a escolhida foi a baseada em seleção de coeficientes, utilizando-se de DCT para geração dos mesmos. Entretanto, tal método possui uma falha grave: a necessidade da imagem original para detecção da marca. Para resolver tal problema, extendeu-se o método para utilizar subamostragem. Além disso, havia a escolha de qual canal de cor a marca deveria ser inserida. Para evitar a seleção estática de uma cor, optou-se em transformar a imagem do espaço de cor RGB para $Y'C_BC_R$ e inserir a mensagem no canal luma (Y'), onde a mesma acabaria se espalhando por todas as cores na transformação de volta. No final, obtemos um algoritmo que necessita de uma chave para gerar o array de seleção de coeficientes, e uma para encriptação do arquivo de extração da marca, aumentando um pouco a segurança do sistema.

A extração da marca é relativamente simples, necessitando somente do arquivo de extração e a chave para decriptá-lo. Os testes e simulações mostraram resultados ótimos referentes a esta detecção. A imagem marcada possui um PSNR alto, atendendo requisito de fidelidade à imagem original, e invisibilidade da marca. A marca demonstrou resiliência e alta robustez contra os ataques de compressão e processamento de sinal, onde conseguimos recuperá-la mesmo após a diminuição da qualidade do recepiente para 10%. Tal característica é altamente desejada, visto que estes tipos de ataques são os mais comumente utilizados.

Por fim, acredita-se que o trabalho atingiu o seu objetivo inicial na implementação de um algoritmo robusto para imagens. O programa criado pode ser utilizado para fins didáticos, onde a flexibilidade de escolha de parâmetros para os dois algoritmos implementados permite que um aluno consiga experenciar os ganhos e perdas dos atributos esteganográficos. Além disso, interessados podem contribuir com o código que se encontra, a partir da submissão deste trabalho, aberto com a Licença Pública Geral GNU versão 3 no domínio de códigos do google[1].

Extensões do trabalho incluem a implementação e análise de outros algoritmos ro-

[1] http://code.google.com/p/wmsteganography/

bustos, e.g., baseados em Wavelets, e robustos a diferentes tipos de ataques, e.g., ataques geométricos (rotação, redimensionamento). Atualmente, através de uma medição amadora verificou-se que o programa em geral leva menos de 5 segundos para inserção de uma marca em imagens de tamanho 512x512. Logo, outra possibilidade inclui extender a arquitetura sequencial do programa em uma paralela, de forma a aumentar ainda mais a velocidade de execução, garantindo o ganho em escalabilidade do mesmo com o aumento do tamanho do recepiente da marca.

REFERÊNCIAS

[ART 2001] ARTZ, D. Digital steganography: hiding data within data. **IEEE Internet Computing**, Los Alamitos, CA, USA, v.5, p.75–80, 2001.

[CHU 2003] CHU, W. Dct-based image watermarking using subsampling. **Multimedia, IEEE Transactions**, v.5, n.1, p.34 – 38, 2003.

[COX 2007] COX, I. et al. **Digital watermarking and steganography**. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2007.

[GON 2008] GONZALEZ, R. C.; WOODS, R. E. **Digital image processing**. Upper Saddle River, NJ, USA: Pearson Education Inc., New Jersey, 2008.

[GUA 2004] GUANNAN, Z.; SHUXUN, W.; GUIJUN, N. A blind watermarking algorithm based on dwt for color image. In: COMMUNICATIONS, 2004 AND THE 5TH INTERNATIONAL SYMPOSIUM ON MULTI-DIMENSIONAL MOBILE COMMUNICATIONS PROCEEDINGS. THE 2004 JOINT CONFERENCE OF THE 10TH ASIA-PACIFIC CONFERENCE ON, 2004. **Anais...** [S.l.: s.n.], 2004. v.2, p.634 – 638 vol.2.

[HER 98] HERODOTUS; WATERFIELD, R. **The history / herodotus; translated by robin waterfield**. Oxford, NY, USA: Oxford University Press Inc., New York, 1998.

[KAH 96] KAHN, D. The history of steganography. In: FIRST INTERNATIONAL WORKSHOP ON INFORMATION HIDING, 1996, London, UK. **Proceedings...** Springer-Verlag, 1996. p.1–5.

[KER 2005] KERMANI, Z.; JAMZAD, M. A robust steganography algorithm based on texture similarity using gabor filter. In: SIGNAL PROCESSING AND INFORMATION TECHNOLOGY, 2005. PROCEEDINGS OF THE FIFTH IEEE INTERNATIONAL SYMPOSIUM ON, 2005. **Anais...** [S.l.: s.n.], 2005. p.578 –582.

[LU 2006] LU, W.; LU, H.; CHUNG, F.-L. Robust digital image watermarking based on subsampling. **Applied Mathematics and Computation**, v.181, n.2, p.886 – 893, 2006.

[MOR 2000] MORA-JIMENEZ, I.; NAVIA-VAZQUEZ, A. A new spread spectrum watermarking method with self-synchronization capabilities. In: IMAGE PROCESSING, 2000. PROCEEDINGS. 2000 INTERNATIONAL CONFERENCE ON, 2000. **Anais...** [S.l.: s.n.], 2000. v.1, p.415 –418 vol.1.

[MUR 2005] MURDOCH, S. J.; LEWIS, S. Embedding covert channels into tcp/ip. In: INFORMATION HIDING: 7TH INTERNATIONAL WORKSHOP, VOLUME 3727 OF LNCS, 2005. **Anais...** [S.l.: s.n.], 2005. p.247–261.

[PRO 2003] PROVOS, N.; HONEYMAN, P. Hide and seek: an introduction to steganography. **Security & Privacy Magazine, IEEE**, v.1, n.3, p.32–44, 2003.

[ROC 2008] ROCHA, A.; GOLDENSTEIN, S. Steganography and steganalysis in digital multimedia: hype or hallelujah? **Revista de Informatica Teórica e Aplicada (RITA)**, v.15, n.1, p.83 –110, 2008.

[SIM 94] SIMMONS, G. J. **Contemporary cryptology**: the science of information integrity. Piscataway, NJ, USA: IEEE Computer Society, 1994.

[YOU 2008] YOUAIL, R.; SAMAWI, V.; KADHIM, A.-K.-R. Combining a spread spectrum technique with error-correction code to design an immune stegosystem. In: ANTI-COUNTERFEITING, SECURITY AND IDENTIFICATION, 2008. ASID 2008. 2ND INTERNATIONAL CONFERENCE ON, 2008. **Anais...** [S.l.: s.n.], 2008. p.245 –248.