ESTEGANOGRAFIA REVERSÍVEL E MARCAS D'ÁGUA DE AUTENTICAÇÃO REVERSÍVEIS PARA IMAGENS BINÁRIAS

Sergio Vicente Denser Pamboukian*
Hae Yong Kim**

Resumo

Esteganografia é uma técnica utilizada para inserir uma sequência de *bits* em uma imagem hospedeira com pequena deterioração visual e com meios para extraí-la posteriormente. A esteganografia reversível permite, além disso, recuperar exatamente a imagem original. Várias técnicas de esteganografia reversível têm sido desenvolvidas, mas muito poucas são apropriadas para imagens binárias. Este artigo propõe uma técnica de esteganografia reversível para imagens binárias. A técnica proposta é então utilizada para autenticar de forma reversível imagens binárias incluindo textos, desenhos e meio-tom.

Palavras-chave: Esteganografia, imagem binária, esteganografia reversível.

* Universidade Presbiteriana Mackenzie (UPM).
** Universidade de São Paulo (USP).

1 INTRODUÇÃO

Esteganografia é uma técnica utilizada para inserir uma sequência de *bits* em uma imagem hospedeira, sem degradar sua aparência e com meios para extraí-la posteriormente. A maioria das técnicas esteganográficas modifica e distorce o sinal hospedeiro para inserir a informação adicional. Essa distorção normalmente é pequena, mas irreversível.

Técnicas esteganográficas reversíveis inserem os *bits* de informação modificando o sinal hospedeiro, mas permitem a exata recuperação (sem perda) da imagem original após a extração da informação inserida. Em vários campos, como judicial, militar, médico e de pesquisa astronômica, a recuperação da imagem original sem perda é essencial. Alguns autores (AWRANGJEB; KANKANHALLI, 2004; CELIK et al., 2002) classificam as técnicas de esteganografia reversível em dois tipos:

1. O primeiro tipo (HONSINGER et al., 2001; FRIDRICH; GOLJAN; DU, 2001) faz uso de técnicas de sobreposição de sinais. Nessas técnicas, um sinal correspondente à informação a ser inserida é sobreposto (adicionado) ao sinal hospedeiro. Na decodificação, a informação escondida é detectada e o sinal sobreposto é removido (subtraído) para restaurar o sinal original. Essas técnicas utilizam aritmética de módulo para evitar erros de *overflow* e *underflow* que podem causar o aparecimento de artefatos do tipo sal e pimenta[1] na imagem e, em geral, oferecem capacidade de armazenamento de informação muito limitada.
2. No segundo tipo (AWRANGJEB; KANKANHALLI, 2004; CELIK et al., 2002, 2005; FRIDRICH; GOLJAN; DU, 2002; TIAN, 2003), algumas porções do sinal hospedeiro são sobrescritas pela informação inserida. Dois tipos de informação podem ser inseridos: dados comprimidos referentes à porção sobrescrita (para permitir a recuperação do sinal original) e a informação que realmente se deseja inserir (carga útil). Durante a decodificação, a informação escondida é extraída, a carga útil é recuperada, e os dados comprimidos são utilizados para restaurar o sinal original. Essas técnicas não causam ruídos do tipo sal e pimenta, porque as porções modificadas são usualmente os *bits* menos significativos ou os coeficientes *wavelet* de alta frequência que não causam distorção perceptível. Essas técnicas, em geral, oferecem maior capacidade de armazenamento do que as do primeiro tipo.

1 Pontos brancos (sal) ou pretos (pimenta) dispersos pela imagem.

Entre as técnicas esteganográficas reversíveis existentes, muito poucas são adequadas para imagens binárias (preto e branco). Este artigo propõe uma técnica esteganográfica reversível do segundo tipo para imagens binárias, chamada *Reversible Data hiding by Template ranking with symmetrical Central pixels* – RDTC (esteganografia reversível por ranqueamento de *templates* com *pixels* centrais simétricos). Com base nessa técnica, é proposto também um método para autenticar de forma reversível imagens e documentos. Existem dois grandes desafios para o projeto de uma técnica esteganográfica reversível do segundo tipo para imagens binárias:

1. O primeiro é o desenvolvimento de uma técnica que permita localizar precisamente os *pixels* que serão modificados tanto na inserção quanto na extração, para que a imagem original possa ser recuperada. Algumas técnicas não têm essa propriedade. Considere, por exemplo, uma técnica esteganográfica em que a imagem é subdividida em blocos, e um *bit* é inserido em cada bloco, invertendo (se necessário) o *pixel* de menor visibilidade (a operação de inversão consiste em transformar um *pixel* branco em preto ou vice-versa). Os blocos com quantidade par de *pixels* pretos possuem o *bit* 0 inserido e os blocos com quantidade ímpar de *pixels* pretos possuem o *bit* 1 inserido. Nessa técnica, a imagem original não pode ser recuperada mesmo se a paridade original dos *pixels* pretos for conhecida, pois o *pixel* invertido dentro do bloco não pode ser localizado. Uma solução é encontrar uma técnica reversível adequada que tenha essa propriedade e convertê-la em uma técnica reversível. Outra solução, que necessita de mais tempo e pesquisa, é desenvolver uma técnica totalmente nova.
2. O segundo é obter uma eficiente compressão da porção que será sobrescrita pela informação inserida. Essa porção é tipicamente pequena, não possui estrutura e suas amostras são distribuídas quase uniformemente e não correlacionadas umas com as outras. Portanto, a compressão direta desses dados resulta em uma baixa capacidade de armazenamento sem perda. Porém, se o restante da imagem for utilizado como informação adicional, um ganho significante pode ser obtido na compressão (CELIK et al., 2002). Em imagens em tons de cinza, a escolha do algoritmo de compressão parece não ser crítica, porque existe espaço suficiente para armazenar as informações (*bits* menos significativos, por exemplo). Awrangjeb e Kankanhalli (2004) utilizam codificação aritmética, LZW[2] e JBIG[3] para compressão sem perda. Celik et al. (2002) utilizam uma versão adaptada da técnica Calic[4]. Na esteganografia reversível para imagens binárias, ao contrário, muitos algoritmos de compressão baseados em

2 LZW é um algoritmo de compressão de dados desenvolvido por Lempel, Ziv e Welch.
3 JBIG é um padrão de compressão de imagens binárias sem perda.
4 Calic é um padrão de compressão de imagens binárias sem perda, adaptativo e baseado em contexto.

redundância e dicionários não são efetivos. A RDTC utiliza o código de Golomb para comprimir os erros de previsão dos *pixels* de baixa visibilidade para obter espaço para esconder a informação desejada.

2 TÉCNICA ESTEGANOGRÁFICA PWLC

Até o momento em que este estudo foi realizado, nossas pesquisas indicavam que apenas uma técnica para inserir informações de forma reversível em imagens binárias havia sido proposta, a *pair-wise logical computation* PWLC (TSAI et al., 2004, 2005). Porém, parece que algumas vezes essa técnica não consegue extrair corretamente as informações e falha na recuperação perfeita da imagem original.

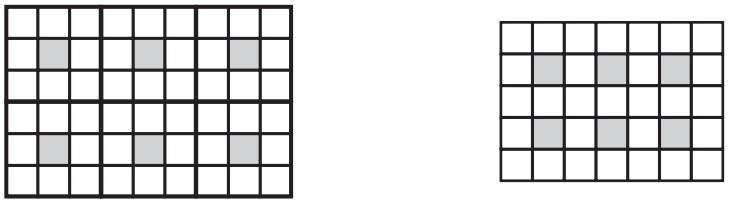
A técnica PWLC não utiliza sobreposição de sinais ou compressão de dados. Ela escaneia a imagem hospedeira em alguma ordem (por exemplo, de cima para baixo, da esquerda para a direita). Apenas sequências "000000" ou "111111" localizadas próximas aos limites da imagem são escolhidas para esconder informações. A sequência "000000" se tornará "001000" se o *bit* 0 for inserido, e "001100" se o *bit* 1 for inserido. Similarmente a sequência "111111" se tornará "110111" se o *bit* 0 for inserido, e "110011" se o *bit* 1 for inserido.

Tsai et al. (2004, 2005), porém, não descrevem claramente como identificar os *pixels* no processo de extração. Os limites da imagem podem ser alterados pela inserção da informação. Além disso, vamos supor que uma sequência "001000" (localizada perto do limite da imagem) seja encontrada na imagem marcada. Esses autores não descrevem como distinguir entre uma sequência "001000" não marcada e uma sequência original "000000" que se tornou "001000" com a inserção do *bit* 0.

3 TÉCNICA ESTEGANOGRÁFICA DHTC

A técnica esteganográfica proposta neste artigo (RDTC) é baseada em uma técnica esteganográfica não reversível para imagens binárias denominada *Data Hiding by Template ranking with symmetrical Central pixels* – DHTC (esteganografia por ranqueamento de *templates* com *pixels* centrais simétricos) (KIM, 2005). A DHTC inverte apenas *pixels* de baixa visibilidade para inserir as informações, e, consequentemente, as imagens marcadas por essa técnica têm excelente qualidade visual e não apresentam ruídos do tipo sal e pimenta.

No algoritmo de inserção da DHTC:

1. Divide-se a imagem hospedeira Z em uma sequência *v* de blocos não sobrepostos (por exemplo, 3 × 3) como pode ser visto na Figura 1a. Apenas os *pixels* centrais dos blocos *v* (que aparecem hachurados na Figura 1a) poderão ser invertidos para a inserção da informação. Uma sobreposição parcial, que aumenta a capacidade de armazenamento, também pode ser feita, sem sobrepor os *pixels* centrais (Figura 1b).
2. 

(a) (b)

Figura 1 Imagem dividida em blocos 3 × 3: (a) blocos não sobrepostos e (b) blocos parcialmente sobrepostos.

Fonte: Elaborada pelos autores.

3. Coloca-se a sequência *v* em ordem crescente utilizando a nota de impacto visual como chave primária e números pseudoaleatórios sem repetição como chave secundária. A chave primária classifica os *pixels* centrais de acordo com a sua "visibilidade". A Figura 2 ilustra um ranqueamento com todas as combinações possíveis de *templates* 3 × 3, listados em ordem crescente de visibilidade de seus *pixels* centrais. Nessa figura, *pixels* hachurados podem ser brancos ou pretos (note que todos os *pixels* centrais estão hachurados). A nota de um determinado bloco é aquela correspondente ao *template* de menor impacto. Espelhos, rotações e reversos de cada template têm a mesma nota. Para assegurar a viabilidade de reconstrução de *v* na etapa de extração de dados, dois *templates* que diferem apenas pela cor do *pixel* central devem ter exatamente a mesma nota de impacto visual. O ranqueamento de visibilidade pode ser modificado e *templates* mais largos podem ser usados de forma a minimizar alguma distorção específica que possa ser percebida. A chave secundária é utilizada para impedir que as informações sejam inseridas apenas na parte superior da imagem.
4. Os *n* primeiros *pixels* centrais do vetor *v* ordenado são os escolhidos para carregar a informação. A partir de agora, chamaremos esses *pixels* de *data bearing pixels* (DBPs). Inserir *n bits* de informação invertendo, se necessário, a cor dos DBPs.

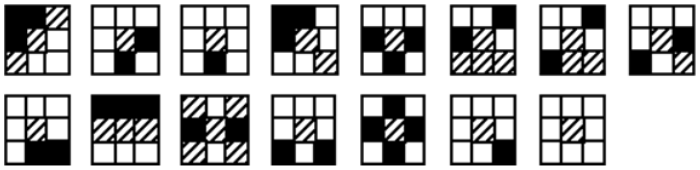


Figura 2 Ranqueamento de *templates* 3 × 3 com *pixels* centrais simétricos em ordem crescente de impacto visual. *Pixels* hachurados podem ser brancos ou pretos.

Fonte: Elaborada pelos autores.

Para extrair as informações escondidas, exatamente a mesma sequência v deve ser reconstruída e ordenada. Então, os n primeiros *pixels* centrais são os DBPs, e seus valores, a informação escondida.

Na técnica DHTC, a posição exata dos DBPs é conhecida em ambas as etapas de inserção e extração. Essa propriedade torna possível transformar DHTC em uma técnica esteganográfica reversível para imagens binárias. Para isso, os valores originais dos DBPs devem ser comprimidos, anexados aos *bits* que se deseja esconder (carga útil), e armazenados nos próprios DBPs.

4 AS TÉCNICAS PROPOSTAS

Este artigo propõe uma técnica esteganográfica reversível para imagens binárias chamada RDTC e uma marca d'água de autenticação frágil reversível chamada *Reversible Authentication watermarking by Template ranking with symmetrical Central pixels* – RATC (marca d'água de autenticação reversível por ranqueamento de *templates* com *pixels* centrais simétricos).

4.1 A técnica esteganográfica reversível proposta

A técnica esteganográfica reversível para imagens binárias proposta é baseada na técnica DHTC descrita anteriormente. Na técnica RDTC, dois tipos de informação devem ser inseridos na imagem hospedeira: dados comprimidos que permitirão a recuperação exata da imagem original e as informações que se deseja esconder (carga útil). Ou seja, os valores originais dos n primeiros *pixels* devem ser comprimidos para gerar espaço para a inserção da carga útil.

Existem algumas dificuldades para comprimir os valores originais dos DBPs. Muitos algoritmos de compressão baseados em redundância e dicionários não funcionam, porque normalmente a quantidade de *bits* a ser comprimida é muito pequena. Além

disso, não existe um modo de prever o valor do próximo *bit* baseado no valor do *bit* anterior, porque esses *bits* correspondem a *pixels* dispersos por toda a imagem.

A solução encontrada foi comprimir a previsão de erro dos valores dos DBPs (utilizando os *pixels* vizinhos como informação adicional) em vez de comprimir diretamente os valores dos DBPs. Foram testados dois esquemas de previsão:

1. Um *pixel* pode ser da mesma cor ou de cor diferente da maioria de seus *pixels* vizinhos. Vamos assumir que a primeira hipótese é mais provável que a segunda. Seja b o número de *pixels* vizinhos pretos de um DBP (usando *templates* 3 × 3, um DBP tem 8 *pixels* vizinhos). A previsão é correta (representada por 0) se a cor do DBP original é preta e $b > 4$, ou se a cor é branca e $b \leq 4$. Caso contrário, a previsão é incorreta (representada por 1). Se essa previsão for razoável, o valor previsto e o valor verdadeiro deverão ser idênticos com probabilidade maior do que 50%. Como o valor 0 é armazenado quando a previsão é correta e o valor 1 quando é incorreta, subsequências de 0 (zeros) serão mais longas (na maioria dos casos) do que subsequências de 1 (um).
2. Também foi testado um esquema mais elaborado de previsão. Construiu-se uma tabela com 256 elementos (todas as possíveis combinações de 8 *pixels* vizinhos), e, utilizando imagens binárias típicas, foram determinadas as cores mais prováveis para o *pixel* central em cada combinação de vizinhança.

Surpreendentemente, os dois esquemas de previsão alcançaram resultados muito parecidos.

A sequência de erros de previsão consiste normalmente em longos segmentos de 0 (zeros) separados por curtos segmentos de 1 (um), porque um 0 ocorre com alta probabilidade p e um 1 ocorre com baixa probabilidade 1-p. O código de Golomb (que será explicado na próxima seção) é um bom algoritmo de compressão para esse tipo de sequência. Como os vizinhos dos DBPs não são modificados durante a inserção, a previsão pode ser reconstruída na extração. O vetor de erros de previsão (0 e 1), juntos com as vizinhanças dos DBPs, permite a recuperação dos valores originais dos DBPs.

O algoritmo de inserção do RDTC:

1. Divide-se a imagem hospedeira Z em uma sequência v de blocos parcialmente sobrepostos.
2. Ordena-se a sequência v de forma crescente utilizando a nota de impacto visual como chave primária, a quantidade de *pixels* pretos ao redor do *pixel* central de cada bloco como chave secundária e números pseudoaleatórios sem repetição como chave terciária.
3. Estimam-se o menor comprimento n de DBPs capaz de armazenar o cabeçalho (tamanho h), o vetor de previsão de erros comprimido (tamanho w) e a carga útil (tamanho p), de forma que $n \geq h+w+p$. Devem-se tentar iterativa-

mente diferentes valores de *n*, até obter o menor valor de n que satisfaça essa desigualdade.

4. Devem-se inserir o cabeçalho (valores de *n*, *w*, *p* e o parâmetro *m* do código de Golomb), o vetor de previsão de erros comprimido e a carga útil invertendo, se necessário, os *pixels* centrais dos n primeiros blocos do vetor ordenado *v*.

Para extrair a carga útil e recuperar a imagem original, a sequência *v* de blocos 3 × 3 deve ser reconstruída e ordenada. Então, os dados são extraídos dos *n* primeiros *pixels* centrais de *v*. O vetor comprimido que contém as previsões de erro é descomprimido e utilizado para restaurar a imagem original.

Os dados são inseridos no início de *v*, porque essa parte possui os *pixels* centrais com menor visibilidade. Porém, para obter uma maior capacidade de armazenamento (sacrificando a qualidade visual), podemos escanear o vetor *v* em busca de um segmento que permita melhor compressão. Os *pixels* do início do vetor têm menor visibilidade, mas não podem ser previstos com precisão, porque normalmente possuem uma quantidade similar de vizinhos pretos e brancos (pois são *pixels* no limite da imagem). Ao avançar no vetor, encontramos *pixels* que podem ser previstos com maior precisão, mais com maior visibilidade. Nesse caso, a posição inicial onde as informações serão inseridas deve ser armazenada no início do vetor *v*. Dessa forma, é necessário verificar o que é mais importante, manter a qualidade visual da imagem ou aumentar a sua capacidade de armazenamento. A Tabela 1 mostra a quantidade *n* de DBPs necessária para armazenar 128 *bits* de carga útil e 37 *bits* de cabeçalho no início de *v* (melhor qualidade) e em um segmento que permite a melhor compressão. Neste último caso, os valores de 176 *pixels* podem ser comprimidos para apenas 11 *bits*. A Tabela 1 também mostra a quantidade de *bits w* utilizada para armazenar os *pixels* comprimidos, a quantidade *n-w* de *bits* livres para inserção de informação (cabeçalho e carga útil) e a quantidade máxima de *pixels* σ que podem ser inseridos em cada imagem de forma reversível. Note que, em alguns casos, até 20% dos *pixels* da imagem podem ser usados para armazenar informações de forma reversível (como no exemplo de texto escaneado a 300 dpi na Tabela 1).

TABELA 1

Inserção reversível em diferentes imagens em tamanho carta

Descrição da imagem	Tamanho (*pixels*)	Melhor qualidade			Melhor compressão			σ
		n	*w*	*n-w*	*n*	*w*	*n-w*	
Texto gerado por computador	1.275 × 1.650	336	146	190	176	11	165	401.600
Texto escaneado a 150 dpi	1.275 × 1.650	432	265	167	176	11	165	214.032
Texto escaneado a 300 dpi	2.384 × 3.194	496	325	171	176	11	165	1.543.680

Fonte: Elaborada pelos autores.

4.2 O código de Golomb

Como foi visto na última subseção, uma sequência de erros de previsão consiste tipicamente em longos segmentos de 0 (zeros) separados por curtos segmentos de 1 (um). Um método eficiente para comprimir esse tipo de informação é o código de Golomb (1966; SALOMON, 2004). Alguns outros métodos baseados no código de Golomb (como LOCO-I, FELICS e JPEG-LS) também parecem ser eficientes, porém não foram testados em nossos estudos.

O código de Golomb é utilizado para codificar sequências de 0 (zeros) e 1 (um), onde 0 ocorre com probabilidade p e 1 ocorre com probabilidade 1-p. Essa sequência é considerada como um número inteiro não negativo n. O código de Golomb depende da escolha do parâmetro inteiro m, com $m \geq 2$, e tem a melhor compressão quando m é calculado pela equação (1).

$$m = \left\lceil -\frac{\log_2 (1 + p)}{\log_2 p} \right\rceil \tag{1}$$

Para computar o código referente ao inteiro não negativo n, três quantidades q, r e c são computadas conforme as equações (2), (3) e (4):

$$q = \left\lfloor \frac{n}{m} \right\rfloor \tag{2}$$

$$r = n - qm \tag{3}$$

$$c = \lceil \log_2 m \rceil \tag{4}$$

Então, o código é construído em duas partes: a primeira é o valor de q codificado em unário, e a segunda, o valor binário de r codificado de forma especial. Os primeiros 2^c-m valores de r são codificados como inteiros sem sinal em c-1 *bits* cada e o restante é codificado em c *bits* cada. O caso onde m é uma potência de 2 é especial, pois não necessita da codificação dos c-1 *bits*, e é conhecido como código de Rice. Para decodificar o código de Golomb, os valores de q e r são utilizados para reconstruir n (n=r+qm). Maiores detalhes podem ser vistos em Golomb (1966) e Salomon (2004).

4.3 Marca d'água de autenticação reversível

Uma marca d'água de autenticação frágil reversível pode ser facilmente construída utilizando RDTC. Essa técnica será chamada de *Reversible Authentication watermarking by Template ranking with symmetrical Central pixels* – RATC (marca d'água de

autenticação reversível por ranqueamento de *templates* com *pixels* centrais simétricos). A RATC pode detectar qualquer alteração na imagem, mesmo em um único *pixel*. Ela pode trabalhar com cifras de chave secreta ou chave pública/privada.

A versão de chave pública/privada da RATC tem o seguinte algoritmo de inserção:

1. Dada uma imagem binária Z a ser autenticada, deve-se computar o índice de integridade de Z utilizando uma função *hash* de mão única $H = H(Z)$. Deve-se criptografar o índice de integridade H utilizando a chave privada para obter a assinatura digital S.
2. Deve-se inserir S em Z utilizando RDTC para obter a imagem marcada Z'.

No algoritmo de verificação da RATC:

1. Dada uma imagem marcada Z', deve-se extrair a assinatura digital S e decriptografá-la utilizando a chave pública, para obter o índice de integridade extraído E.
2. Deve-se extrair o vetor de previsões de erro, descomprimi-lo e restaurar a imagem original Z. Recalcula-se a função *hash*, para obter o índice de integridade de verificação $C = H(Z)$.
3. Se o índice de integridade extraído E e o índice de integridade de verificação C forem o mesmo, a imagem será autêntica. Caso contrário, a imagem foi modificada.

Essa técnica pode detectar qualquer alteração na imagem, mesmo a inversão de um único *pixel*, porque todos os *pixels* da imagem são utilizados para computar o índice de integridade de verificação C. Então, qualquer alteração dos *pixels* utilizados para computar C é detectada, pois altera o valor de C (a probabilidade de uma alteração desse tipo não ser detectada é de 2^{-n}, onde n é o número de bits da função de *hash*), e qualquer alteração nos DBPs também pode ser detectada, pois altera a assinatura digital armazenada usada para computar o índice de integridade extraído E.

5 RESULTADOS EXPERIMENTAIS

A técnica RATC foi testada para autenticar imagens binárias de diferentes tipos e tamanhos (textos escaneados, textos gerados por computador, meios-tons, desenhos, ruídos aleatórios etc.) inserindo 128 *bits* de forma reversível. Os 128 *bits* são suficientes para armazenar um código de autenticação de mensagem utilizado na autenticação de imagens com chave secreta. A Tabela 1 mostra a quantidade máxima de *bits* que

podem ser armazenados de forma reversível em imagens que representam documentos no tamanho carta. A Tabela 2 mostra que, em média, apenas 406 *pixels* de baixa visibilidade são comprimidos para gerar espaço suficiente para armazenar 128 *bits* de dados e 37 *bits* de cabeçalho (excluindo as imagens que representam ruídos aleatórios). Nessa tabela, "não" significa que a inserção não foi possível.

TABELA 2

Inserção de 128 *bits* de dados e 37 *bits* de cabeçalho em diferentes tipos de imagens

Imagem	Descrição	Tamanho	n	w	n-w
lena0	Difusão de erro	512 × 512	400	233	167
lena2	Ordered dithering	512 × 512	208	41	167
fides	Texto gerado por computador	1.275 × 1.650	336	146	190
persuas	Texto escaneado a 150 dpi	1.275 × 1.650	368	203	165
toip300	Texto escaneado a 300 dpi	2.384 × 3.194	528	355	173
toip300b	Porção da imagem toip300	1.094 × 414	464	288	176
toip300c	Porção da imagem toip300	1.092 × 1.664	560	385	175
toip400	Texto escaneado a 400 dpi	3.179 × 4.259	464	290	174
abc	Texto pequeno gerado por computador	91 × 58	368	200	168
pag1	Texto minúsculo gerado por computador	64 × 56	368	193	175
noise10	10% *pixels* pretos aleatórios	300 × 300	400	227	173
noise20	20% *pixels* pretos aleatórios	300 × 300	880	711	169
noise30	30% *pixels* pretos aleatórios	300 × 300	3824	3654	170
noise40	40% *pixels* pretos aleatórios	300 × 300	Não	Não	Não

Fonte: Elaborada pelos autores.

As imagens marcadas têm excelente qualidade visual, pois apenas *pixels* de baixa visibilidade são modificados. Em aplicações típicas de autenticação de documentos, uma página em tamanho carta é escaneada a 300 dpi e autenticada. A Figura 3 ilustra a qualidade visual de uma imagem (com 2.384 × 3.194 *pixels)* nessa situação, autenticada com 128 *bits*. Note que apenas *pixels* de baixa visibilidade no limite da imagem foram modificados. As figuras 4, 5, 6 e 7 ilustram diferentes tipos de pequenas imagens, cada uma com 300 × 300 *pixels*, autenticadas utilizando RATC com 128 *bits* escondidos, para mostrar a qualidade visual de imagens autenticadas em situações desfavoráveis. Em todos os casos, as imagens recuperadas são idênticas às originais. Em cada imagem, *n pixels* foram comprimidos, produzindo *w pixels* comprimidos e *n-w bits* livres onde os códigos de autenticação foram inseridos. Em média, *n*/2 *pixels* foram invertidos.

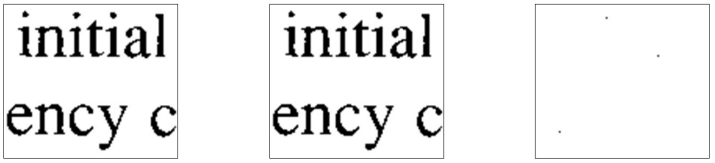


Figura 3 Imagem autenticada de forma reversível: (a) parte da imagem original, (b) parte da imagem autenticada e (c) *pixel*s modificados.

Fonte: Elaborada pelos autores.

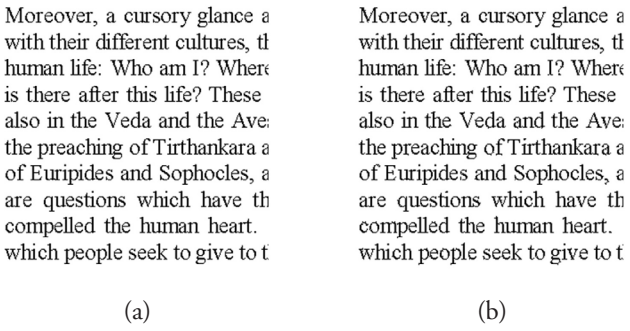


Figura 4 Documento gerado por computador: (a) imagem original e (b) imagem autenticada com $n = 464$, $w = 283$ e $n\text{-}w = 181$.

Fonte: Elaborada pelos autores.

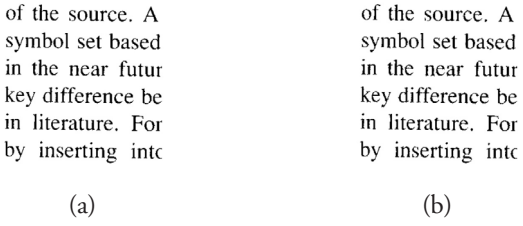
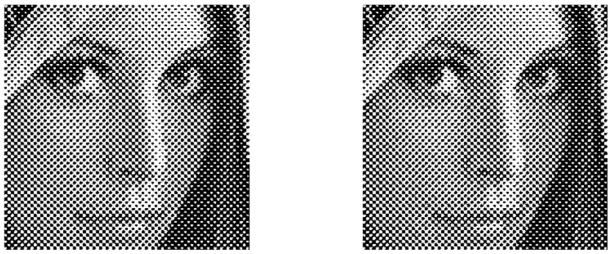


Figura 5 Documento escaneado a 300 dpi: (a) imagem original e (b) imagem autenticada com $n = 432$, $w = 250$ e $n\text{-}w = 182$.

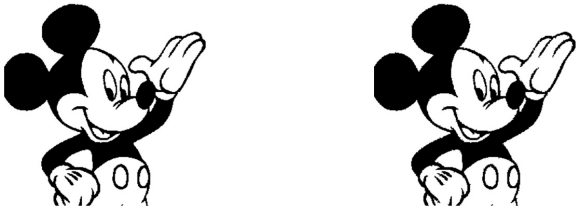
Fonte: Elaborada pelos autores.



(a) (b)

Figura 6 Imagem meio-tom para impressoras laserjet: (a) imagem original e (b) imagem autenticada com *n* = 432, *w* = 254 e *n*-*w* = 178.

Fonte: Elaborada pelos autores.



(a) (b)

Figura 7 Imagem em baixa resolução: (a) imagem original e (b) imagem autenticada com *n* = 432, *w* = 256 e *n*-*w* = 176.

Fonte: Elaborada pelos autores.

Apenas dois tipos de imagens não puderam ser autenticados: 1. imagens muito pequenas com menos de 50 × 50 *pixels*, pois não há espaço suficiente para inserir 128 *bits*; e 2. imagens que representam ruídos aleatórios com quantidades similares de *pixels* pretos e brancos (Figura 8), pois a previsão de erros é muito difícil. Entretanto, esses tipos de imagens não são muito usuais e podem ser ignorados para todos os propósitos práticos.

Figura 8 Parte de uma imagem 300×300 com 40% de *pixels* pretos aleatórios.

Fonte: Elaborada pelos autores.

O programa executável da técnica RATC está disponível em www.lps.usp.br/~hae/software/ratc.

6 CONCLUSÃO

Apresentou-se uma técnica esteganográfica reversível para imagens binárias, utilizada para autenticar imagens de forma reversível. Nessa técnica, erros de previsão de *pixels* de baixa visibilidade são comprimidos utilizando o código de Golomb para criar espaço para armazenar informações. Aplicou-se a técnica proposta em diversos tipos de imagens binárias, e, em média, apenas 406 *pixels* foram comprimidos para gerar espaço para armazenar 128 *bits* de informação. As imagens marcadas apresentaram excelente qualidade visual, uma vez que apenas *pixels* de baixa visibilidade são modificados no processo de inserção da marca. As imagens restauradas após a retirada da marca são idênticas às imagens originais.

DATA HIDING AND WATERMARKS REVERSIBLE AUTHENTICATION FOR BINARY IMAGES

Abstract

Data hiding is a technique used to embed a sequence of bits in a host image with small visual deterioration and the means to extract it afterwards. Reversible data hiding allows, in addition, recovering the original cover-image exactly. Several reversible data hiding techniques have been developed but very few of them are appropriate for binary images. This paper proposes a reversible data hiding technique for binary images. The proposed technique is then used to reversibly authenticate binary images, including texts, drawings and halftones.

Keywords: Data hiding, binary images, reversible data hiding.

REFERÊNCIAS

AWRANGJEB, M.; KANKANHALLI, M. S. Lossless watermarking considering the human visual system. *Int. Workshop on Digital Watermarking 2003, Lecture Notes in Computer Science*, v. 2939, p. 581-592, 2004.

CELIK, M. U. et al. Reversible data hiding. *Proc. IEEE Int. Conf. on Image Processing*, v. 2, p. 157-160, 2002.

_____. Lossless generalized-LSB data embedding. *IEEE Trans. Image Processing*, v. 14, n. 2, p. 253-266, 2005.

FRIDRICH, J.; GOLJAN, M.; DU, R. Invertible authentication. *Proc. SPIE Security and Watermarking of Multimedia Contents III*, San Jose, v. 3971, p. 197-208, 2001.

_____. Lossless data embedding – new paradigm in digital watermarking. *EURASIP Journal on Applied Signal Processing*, v. 2, p. 185-196, 2002.

GOLOMB, S. W. Run-length encodings. *IEEE Trans. Inform. Theory*, v. IT-12, p. 399-401, 1966.

HONSINGER, C. W. et al. *Lossless recovery of an original image containing embedded data*. US Patent #6,278,791, 2001.

KIM, H. Y. A new public-key authentication watermarking for binary document images resistant to parity attacks. *Proc. IEEE Int. Conf. on Image Processing*, Genoa, v. 2, p. 1074-1077, 2005.

SALOMON, D. *Data compression*: the complete reference. 3. ed. New York: Springer-Verlag, 2004. p. 57-64.

TIAN, J. Reversible data embedding using difference expansion. *IEEE Trans. Circuits Systems and Video Technology*, v. 13, n. 8, p. 890-896, 2003.

TSAI, C.-L. et al. Data hiding of binary images using pair-wise logical computation mechanism. *Proc. IEEE International Conference on Multimedia and Expo*, Taipei, v. 2, p. 951-954, 2004.

_____. Reversible data hiding and lossless reconstruction of binary images using pair-wise logical computation mechanism. *Pattern Recognition*, v. 38, p. 1993-2006, 2005.

Contato

Sergio Vicente Denser Pamboukian
e-mail: sergiop@mackenzie.br