

Trabalho de Processamento de Imagens

Atividade 2 - Esteganografia

Objetivo

O objetivo dessa atividade é explorar os conceitos de cores e de codificação e decodificação utilizados nos formatos de imagens.

Problema

O termo **esteganografia**¹ deriva da junção das palavras gregas *estegano* que significa “esconder ou mascarar”, e *grafia*, que significa “escrita”. Portanto, esteganografia pode ser compreendida como a arte de esconder informações, tornando-as ocultas, assim como a criptografia. O objetivo desta técnica é que esses dados não sejam percebidos por terceiros; ou seja, a presença de mensagens escondidas dentro de arquivos é simplesmente desconhecida. Somente o receptor da mensagem tem conhecimento de sua existência, assim como da maneira como extraí-la.

Apesar de parecer, a esteganografia e a criptografia são duas áreas com objetivos bastante diferentes. Enquanto o segundo tem o propósito de impedir que as pessoas saibam o conteúdo de uma mensagem, o primeiro se baseia em evitar que as pessoas saibam que a mensagem escondida existe. Ou seja, na criptografia, os receptores sabem da existência das mensagens, porém não conseguem, a princípio, lê-las; a esteganografia tenta fazer com que os receptores não percebam que há uma mensagem naquele meio (imagem, texto, etc.).

Existem várias formas de esconder mensagens ou arquivos em imagens. Uma das técnicas possíveis é modificar alguns bits (menos relevantes) dos pixels da imagem, com as informações que se quer esconder. Por exemplo, a modificação do bit menos significativo de uma banda (red) de um pixel de uma imagem colorida não é perceptível ao olho humano. A esteganoanálise dessa modificação, no entanto, não é difícil de se realizar a partir da comparação do arquivo original e do arquivo modificado. O ruído, calculado como a diferença desses dois arquivos é a mensagem codificada.

O objetivo desse trabalho é completar o código disponível nesta atividade para realizar a decodificação de um arquivo que foi esteganografado na imagem.

Descrição

1. Um arquivo está escondido na imagem .PPM. O arquivo escondido está codificado nos pixels da imagem da seguinte forma:
 - o binário dos códigos ASCII do nome original do arquivo (finalizado pelo caracter zero)
 - o binário do tamanho do arquivo (em quatro bytes)
 - e os bytes do arquivo.

¹https://www.gta.ufrj.br/grad/09_1/versao-final/stegano/introducao.html

- | | | | | | |
|-------------------|-------------------|-------------------|----------------------------|----------------------------|----------------------------|
| $r_1 \ g_1 \ b_1$ | $r_2 \ g_2 \ b_2$ | $r_3 \ g_3 \ b_3$ | $r_4 \ g_4 \ b_4$ | $r_5 \ g_5 \ b_5$ | $r_6 \ g_6 \ b_6$ |
| $r_7 \ g_7 \ b_7$ | $r_8 \ g_8 \ b_8$ | $r_9 \ g_9 \ b_9$ | $r_{10} \ g_{10} \ b_{10}$ | $r_{11} \ g_{11} \ b_{11}$ | $r_{12} \ g_{12} \ b_{12}$ |

R_1	g_1	b_1	r_2	G_2	b_2	r_3	g_3	B_3	R_4	g_4	b_4	r_5	G_5	b_5	r_6	g_6	B_6
R_7	g_7	b_7	r_8	G_8	b_8	r_9	g_9	b_9	r_{10}	g_{10}	b_{10}	r_{11}	g_{11}	b_{11}	r_{12}	g_{12}	b_{12}

- $R_1 = r_1 \ \& \ 0xFE$, codifica o bit zero (0)
- $G_2 = g_2 \mid 0x01$, codifica o bit um (1)
- $B_3 = b_3 \ \& \ 0xFE$, codifica o bit zero (0)
- $R_4 = r_4 \ \& \ 0xFE$, codifica o bit zero (0)
- $G_5 = g_5 \ \& \ 0xFE$, codifica o bit zero (0)
- $B_6 = b_6 \ \& \ 0xFE$, codifica o bit zero (0)
- $R_7 = r_7 \ \& \ 0xFE$, codifica o bit zero (0)
- $G_8 = g_8 \mid 0x01$, codifica o bita um (1)

- ## Exemplo

[illegible][illegible]

Para incluir o arquivo de nome 'A.txt', com os seguintes 2 bytes:

1 42

Cuja codificação (nome, tamanho e os bytes) em binário é:

```
1 A(41) 01000001 .(2E) 00101110
2 t(74) 01110100 x(78) 01111000 t(74) 01110100 (0) 00000000 // nome
3 (0) 00000000 (0) 00000000 (0) 00000000 (2) 00000010 // tamanho
4 4(34) 00110100 2(32) 00110010 // bytes
```

6. Pode-se observar ainda que, considerando esse esquema de codificação, para que um arquivo possa ser escondido numa imagem, o número de pixels da imagem deve ser suficiente para a codificação do arquivo. Sejam:

- TN (Tamanho do Nome) = o número de letras do nome do arquivo somado de um (caracter zero no fim do nome);
- TA (Tamanho do arquivo) = quatro bytes usados para codificar o tamanho do arquivo;
- NB (Número de bytes) = número de bytes do arquivo a ser ocultado;
- nlinhas = número de linhas da imagem;
- ncolunas = número de colunas da imagem.

Assim, o arquivo só pode ser esteganografado na imagem se:

$$8 \times (TN + TA + NA) \leq nlinhas \times ncolunas$$

7. No exemplo anterior, $TN = 6$, $TA = 4$, $NB = 2$, o que soma $12 * 8$ pixels que foram afetados, ou seja, 96 pixels da imagem com 100 pixels (10 linhas e 10 colunas) foram modificados para codificar o arquivo.
8. Considerando esse esquema de codificação do arquivo na imagem, a sua tarefa é desenvolver o **decodificador**. O código a ser completado está disponível junto com a atividade. Este programa deve receber o nome do arquivo da imagem, extrair e gerar (em disco) o arquivo que está escondido.

Entrega

1. Incluir um comentário no cabeçalho de cada programa fonte com o seguinte formato:

```
1 /*
2 *          UNIFAL – Universidade Federal de Alfenas.
3 *          BACHARELADO EM CIENCIA DA COMPUTACAO.
4 * Trabalho...: Esteganografia
5 * Disciplina: Processamento de Imagens
6 * Professor.: Luiz Eduardo da Silva
7 * Aluno.....: Fulano da Silva
8 * Data.....: 99/99/9999
9 *          */
```

2. O projeto deverá incluir um arquivo MAKEFILE para construção da ferramenta de decodificação do arquivo que está esteganografado (Já disponível nos arquivos da atividade)

3. O programa deverá ser chamado em linha de comando da seguinte forma:

```
1 ./decode <nome-arquivo-imagem>[.ppm]
2
3 por exemplo:
4 ./decode porto
```

4. O programa deverá gerar o arquivo que está codificado na imagem, na mesma pasta onde o decodificador for executado.