Universidade Federal de Alfenas - UNIFAL-MG
Bacharelado em Ciência da Computação
Prof. Luiz Eduardo da Silva

Trabalho de Processamento de Imagens
Atividade 2 - Esteganografia

Objetivo

O objetivo desse 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.

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

2. Os bits (zero ou um) das representações binárias estão armazenados, modificando as bandas R,G e B dos pixels em sequência. Por exemplo, considerando a imagem colorida:

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

Seja 'A', a primeira letra do nome do arquivo (cujo código ASCII binário é 01000001). Na codificação do arquivo, a letra 'A' modifica as seguintes bandas dos pixels iniciais (identificados em letras maiúsculas e com cores).

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

Onde:

- $R_1 = r_1\ \&\ 0xFE$, codifica o bit zero (0)
- $G_2 = g_2\ |\ 0x01$, 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)
- $G_8 = g_8\ |\ 0x01$, codifica o bita um (1)

3. A operação $R_1 = r_1\ \&\ 0xFE$, que executa um e-bit-a-bit com a máscara $0xFE$, tem o efeito de transformar o último bit da banda R para zero.

4. De forma simétrica, a operação $G_2 = g_2\ |\ 0x01$, que executa um ou-bit-a-bit com a máscara $0x01$, tem o efeito de transformar o último bit da banda G para um e assim sucessivamente.

5. Depois da codificação do nome do arquivo, os quatro bytes seguintes definem a quantidade de bytes do arquivo, e dessa forma, a quantidade de pixels que estão modificados no restante da imagem.

Exemplo

Arquivo original:

```
P3
#Created with GIMP
10 10
255
0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
0 0 0  140 26 26  197 40 40  197 40 40  140 26 26  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
0 0 0  197 40 40  209 43 43  209 43 43  197 40 40  40 140 26  59 197 40  59 197 40  40 140 26  0 0 0
0 0 0  197 40 40  209 43 43  166 53 144  85 61 197  61 197 85  63 209 43  63 209 43  59 197 40  0 0 0
0 0 0  140 26 26  197 40 40  85 61 197  43 63 209  61 198 90  63 209 43  63 209 43  59 197 40  0 0 0
0 0 0  0 0 0  0 0 0  40 59 197  43 63 209  53 147 166  61 197 85  59 197 40  40 140 26  0 0 0
0 0 0  0 0 0  0 0 0  26 40 140  40 59 197  40 59 197  26 40 140  0 0 0  0 0 0  0 0 0
0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
```

Arquivo modificado:

```
P3
# CREATOR: Image Processing using C-Ansi - ByDu
#
10 10
255
0 0 0  0 1 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 1 0  0 0 0  0 0 0
0 1 0  0 0 0  1 0 0  0 1 0  0 0 1  0 0 0  0 0 0  0 0 1  1 0 0  0 1 0
0 0 0  141 26 26  197 40 40  197 40 40  140 26 26  0 1 0  0 0 1  1 0 0  0 1 0  0 0 0
0 0 0  197 40 40  209 43 42  209 43 43  197 41 40  40 140 27  58 197 40  59 197 40  40 140 26  0 0 0
0 0 0  197 40 40  208 43 43  166 52 144  85 61 196  60 197 85  63 208 43  63 209 42  58 197 40  0 0 0
0 0 0  140 26 26  197 40 40  85 61 196  42 63 209  61 198 90  63 209 42  62 209 43  59 196 40  0 0 0
0 0 0  0 0 0  0 0 0  40 59 197  43 62 209  53 147 166  60 197 85  59 196 40  40 140 26  0 0 0
0 0 0  0 0 0  0 0 0  26 40 140  40 59 196  40 59 197  26 40 140  0 0 0  1 0 0  0 0 0
0 0 0  0 0 0  0 1 0  0 0 1  0 0 0  0 1 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
1 0 0  0 1 0  0 0 0  0 0 0  0 1 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0  0 0 0
```

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

```
42
```

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

```
A(41) 01000001 .(2E) 00101110
t(74) 01110100 x(78) 01111000 t(74) 01110100 (0) 00000000 // nome
(0) 00000000 (0) 00000000 (0) 00000000 (2) 00000010       // tamanho
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:

```
/*-------------------------------------------------------------------
 *            UNIFAL - Universidade Federal de Alfenas.
 *              BACHARELADO EM CIENCIA DA COMPUTACAO.
 * Trabalho..: Esteganografia
 * Disciplina: Processamento de Imagens
 * Professor.: Luiz Eduardo da Silva
 * Aluno.....: Fulano da Silva
 * Data......: 99/99/9999
 *-------------------------------------------------------------------*/
```

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:

```
./decode <nome_arquivo_imagem>[.ppm]

por exemplo:
./decode porto
```

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