Projeto 4: Esteganografia

.

MC920 - Introdução ao Processamento Digital de Imagem (MC920 / MO443) 2S2019
Professor: Hélio Pedrini

Giovani Nascimento Pereira
giovani.x.pereira@gmail.com
168609

I. Definição do problema

Esteganografia é uma forma de transmissão de dados codificados em imagens.

A. Objetivo

O objetivo deste trabalho é de implementar métodos de esteganografia, codificando mensagens em imagens e depois sendo capaz de recuperá-las.

B. Execução do projeto

Para executar os projeto, existem 2 scripts:

- codificar.py
- decodificar.py

Para executar os scripts são utilizados os comandos:

Onde *input_image* é o caminho da imagem de entrada, na qual será codificada a mensagem. *input_text* é o caminho do texto a ser codificado. *output_text* é o caminho do texto de saída gerado na decodificação. *bits* é qual bit menos significativo será alterado para a codificação da mensagem. E onde *output_image* é o caminho da imagem gerada na codificação.

A única *flag* disponível para a execução é a *-d* que habilita que informação de debug seja imprimida na saída padrão. Isso inclui etapas intermediárias do processo de conversão de caracteres em uma string binárias para a codificação e do processo reverso também.

II. Codificação

A. Conversão do texto em binário

Para ser inserido na imagem, o texto de entrada deve ser convertido em uma sequência de bits que representa o mesmo texto. Isso foi feito, para cada caracter, convertendo para seu valor na tabela ASCII. Com esse valor numérico, ele foi transformado numa string que representava o mesmo valor mas em codificação binária.

Por exemplo, o caracter a, equivale a 97 na tabela ASCII. Com esse valor, convertemos 97 em binário 01100001.

É importante que as strings que representam os caracteres tenham exatamente 8 caracteres, isso garante consistência no nosso processo e facilita a decodificação.

Esse processo é repetido para todo caracter do texto de entrada, e concatenado com o texto em binário final a ser codificado.

Ao final, também foi adicionado um caracter que não pertencia ao texto de entrada, chamado de *caracter de parada*. Este caracter é uma sequência de 8 zeros 00000000. Na tabela ASCII este caracter de valor 0 não representa um caracter visível, e no nosso caso, pode ser usado para demarcar o fim do arquivo a ser lido.

Caso não houvesse um caracter de parada, ficaria mais difícil definir durante a decodificação quando a mensagem teria acabado, podendo tentar decodificar a imagem toda durante o processo e não apenas a parte que contém a mensagem desejada.

B. Codificação na imagem

Para inserir a mensagem, já representada em uma string binária, na imagem, é iterado sobre a imagem pixel a pixel e em cada canal de cor RGB da mesma (nessa ordem), modificando o bit menos significativo escolhido com o valor da string desejada.

Por exemplo próximo valor a ser inserido é 1 no bit menos significativo, e a imagem tem um pixel preto, que indica que todo canal de cor RGB tem valor mínimo, zero. Então o valor do pixel, representado em binário é 00000000, adicionando o valor desejado tem-se 00000001.

E este processo é repetido para todos os caracteres da mensagem a ser codificada.

III. Decodificação

A. Leitura da Imagem

A imagem é aberta da mesma forma e o bit no qual a mensagem deve estar codificada é apresentado.

Com essas informações, a fim de extrair a mensagem da imagem, basta iterar sobre todos os pixels na ordem correta dos canais de cor RBG e ir *acumulando* o valor dos bits menos significativos.

B. Conversão em texto

O acumulado de todos os bits extraídos dos pixels da imagem é a sequência de bits que pode corresponder a mensagem desejada. Para descobrir tanto, é iterado sobre esse conjunto de bits de 8 em 8, pois cada caracter é representado em 8 bits.

Um conjunto de 8 bits representa um valor numérico, então podeos converte-lo em um numero, e depois disso pegar o valor equivalente na tabela ASCII.

Fazendo esse processo corretamente é possível extrair uma mensgaem que tenha sido codificada na imagem.

Além disso, esse processo pode parar se o caracter de parada, que tem valor numérico 0, for encontrado.

IV. Resultados

As imagens a seguir foram geradas codificando um texto de 20.764 caracteres. O texto escolhido foi um *Lorem Ipsum* genérico, e foi aplicado na imagem do *baboon.png*.

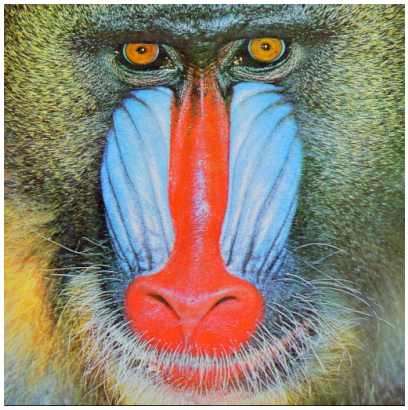

Figura 1. imagem original do baboon

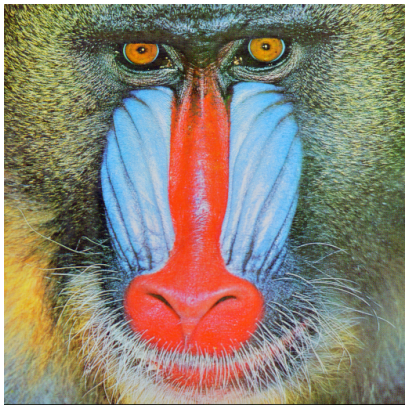

Figura 2. imagem do baboon com o texto modelo esteganografado no bit menos significativo

A imagem original do baboon está em formato png e tem dimensões 512 x 512 pixels.

Isso significa que tem 262.144 pixels no total, e como tem 3 canais de cor, é capaz de abrigar até três vezes a quatidade de pixels em bits.

Como cada caracter ocupa 8 bits, é possível codificar mensagens de até 98.304 caracteres nessa imagem. Isso significa que o texto utilizado de 20.764 caracteres pode ser todo codificado na imagem e ocupa 21.1% dos pixels.

V. Análises

É possível notar quer as imagens finais geradas depois do processo de codificação das mensagens são muito semelhantes às imagens originais.

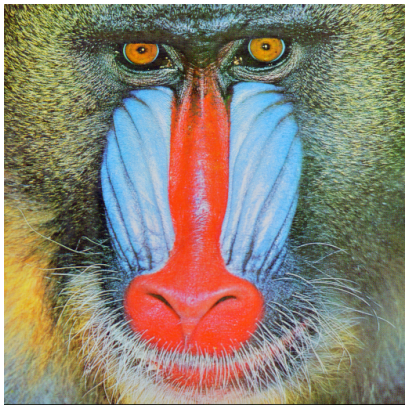

Figura 3. imagem do baboon com o texto modelo esteganografado no segundo bit menos significativo

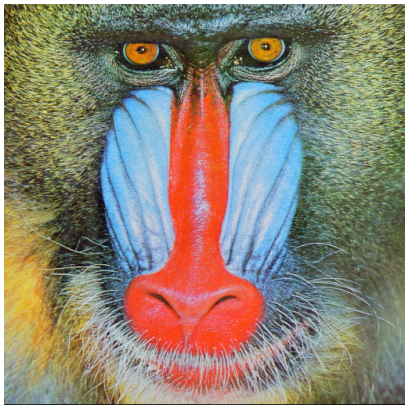

Figura 4. imagem do baboon com o texto modelo esteganografado no terceiro bit menos significativo

Isso se deve pelo fato que os valores modificados na imagens se referem aos bits menos significativos das mesmas, assim as mudanças em níveis de cor são pouco refletidas. Isso mesmo até para os textos grandes.

Contudo, com uma análise mais profunda dos valores dos pixels, é possível notar sim que há uma mudança na cor dos pixels modificados. Isso pode ser observado na imagem a seguir:

Mesmo havendo mudanças, elas são sutis, e ao observar a imagem como um tudo não é possível notar diferenças.

Dessa forma, uma mensagem esteganografada numa imagem colorida é muito difícil de ser percebida, apesar de se tratar de um processo relativamente simples de codificação.

Se observarmos novamente a imagem 5, os 6 primeiros pixels da imagem são capazes de armazenar 18 bits (3 cada pixel). Com isso, conseguem armazenar apenas 2 carateres por completo.

Vamos analisar o primeiro caracter codificado, a letra **E**.

E equivale a 69 na tabela ASCII e 01000101 em binário. Esse foi o valor gravado nos bits dos 3 primeiros pixels da imagem, nos 3 ultimos bits menos significativos para cada item.

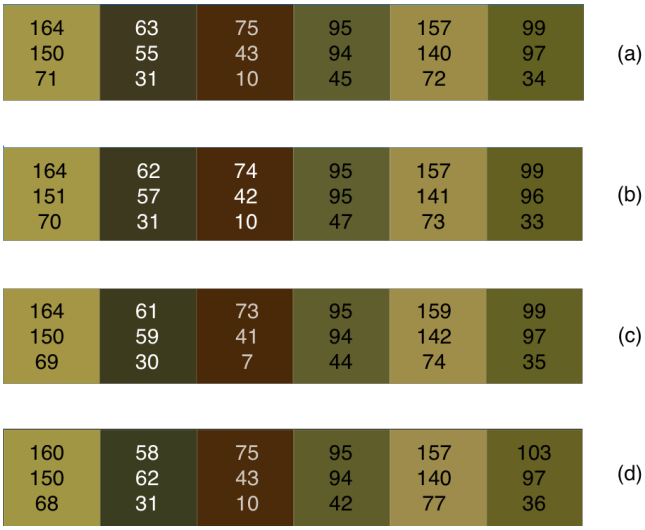

Figura 5. Esquematização dos 6 primeiros pixels da imagem *baboon* com os níveis de cor explicitados em cada pixel, na ordem R G e B. a - Imagem original. b - imagem aplicada esteganografia para o bit menos significativo. c - imagem aplicada esteganografia para o segundo bit menos significativo. d - imagem aplicada esteganografia para o terceioro bit menos significativo. O texto codificado na imagem foi *"Eu sei o que vocês fizeram no verão passado."*

VI. Conclusão

Através do resultados obtidos no projeto é possível dizer que o experimento foi satisfatório. De maneira geral foi possível codificar mensagens em imagens coloridas (.png) e depois obter as mesmas mensagens através apenas das imagens codificadas.

Referências

[1] Helio Pedrini. Trabalho 4. Introdução ao Processamento Digital de Imagem (MC920 / MO443), 2019.

[2] SRGB. disponível em https://en.wikipedia.org/wiki/SRGB. Acesso 11 de setembro de 2019.