Mc920 - Trabalho 4

Felipe Santana Dias - 215775

1. ESPECIFICAÇÃO DOS PROBLEMAS

O objetivo desse trabalho é implementar um algoritmo de esteganografia em imagens digitais e analisar os resultados obtidos.

2. ENTRADA DE DADOS

O código desenvolvido para a codificação e decodificação da imagem foi desenvolvido em python através das bibliotecas OpenCV e NumPy em dois arquivo chamados '*codificar.py' e* 'de*codificar.py'.* O programa foi projetado para imagens coloridas RGB no formato PNG.

Para executar o código é necessário fornecer algumas informações no momento da requisição em ambos os casos. Para cada um dos código é necessário as seguintes execuções:

```
python codificar.py imagem_entrada.png texto_entrada.txt plano_bits imagem_saida.png
python decodificar.py imagem_saida.png plano_bits texto_saida.txt
```

As análises feitas neste relatório estão baseadas nas observações realizadas sobre a imagem 'baboon.png', disponibilizada pelo professor e presente na pasta raiz juntamente com o código e as imagens resultantes.

Após o processamento, os arquivos gerados serão salvos na pasta raiz respeitando as nomenclaturas passadas na linha de comando.

3. CÓDIGO E DECISÕES TOMADAS

3.1 Algoritmo - codificar.py

Inicialmente é realizada a leitura do arquivo *.txt* contendo a mensagem a ser transmitida. Esse conteúdo é enviado para a função *textToASCII,* a qual realiza a conversão de cada caracter para o decimal correspondente na tabela ASCII e posteriormente para o valor em binário. Ao final temos como resposta uma string de 0s e 1s correspondentes à toda mensagem. Ao final, utilizamos o número binário correspondente ao * como critério de parada. É importante ressaltar que como estamos usando a codificação UTF-8, palavras acentuadas irão acarretar em erros de execução.

A imagem foi carregada através da biblioteca OpenCV e convertida em uma NumPy Array. Com todos os parâmetros presente, primeiramente na função *imgCamada* geramos a imagem do plano de bits selecionado antes da aplicação da esteganografia, resultando no arquivo na pasta raiz cujo título é 'Camada - Sem mensagem - imagem_saida.png'.

Em seguida, é utilizada a função *msgInsertion* para realizar o processo de esteganografia e inserir a mensagem na imagem. Para isso, foi utilizado 3 laços encaixados que percorrem as camadas RGB da imagem enquanto a mensagem ainda não foi inserida completamente. A troca do bit da imagem pelo bit da mensagem é feita na função *trocarBit* e é aplicada todas as vezes durante o laço.

Após feito esse processo, é utilizada novamente a função *imgCamada* para gerar a imagem do plano de bits escolhido, resultando em um arquivo na pasta raiz cujo título é 'Camada - Com mensagem - imagem_saida.png'.

3.2 Algoritmo - decodificar.py

Para decifrar a mensagem inserida na imagem, o algoritmo de *decodificar.py* utiliza a função *decodificador* para realizar essa tarefa. Com 3 loops encaixados percorrendo a imagem, esse código utiliza a função *selecionarPlano* para encontrar os bits relativos à mensagem em cada pixel da imagem. Esse processo é feito até ser encontrado número binário '00101010', correspondente ao caracter '*", nosso critério de parada..

Após reconhecer a mensagem, é realizada a transformação de *string* para *int* e convertido para os caracteres correspondentes na tabela ASCII. Dessa maneira, a mensagem decodificada é salva no arquivo *.txt* especificado na linha de comando.

3.3 Arquivos para análise

Como nosso objetivo é analisar os resultados que o algoritmo de esteganografia gera, utilizamos unicamente a imagem 'baboon.png' como base e realizamos testes em textos menores, com 5 palavras, e textos maiores, com 100.000 palavras. Também foram feitos testes alterando os bits em que a mensagem seria ocultada.

4. SAÍDA DE DADOS

Ao final da execução é gerada todos os arquivos especificados na linha de comando, adicionados daqueles já discutidos anteriormente que auxiliam na compreensão dos resultados.

5. RESULTADOS

Para discutir os resultados obtidos utilizaremos a imagem *'baboon.png'* disponibilizada no site da disciplina, aplicando a esteganografia nos textos menores e maiores, variando também o bit utilizado.

5.1 Mensagem pequena

No caso em que utilizamos uma mensagem pequena com 5 palavras (*"Esse eh um texto pequeno."*), obtivemos resultados extremamente satisfatórios quanto à discrição da mensagem na imagem.

Em todos os casos que realizamos a esteganografia, isto é, nos planos de Bit 0 (figura 1) e 7 (figura 2), a mensagem é indetectável ao olho humano tanto na imagem final quanto nas imagens dos planos de bit separados.

Figura 1 - Mensagem pequena e plano Bit 0, da esquerda para a direita: Plano de bits sem mensagem, imagem com mensagem, plano de bits com mensagem.

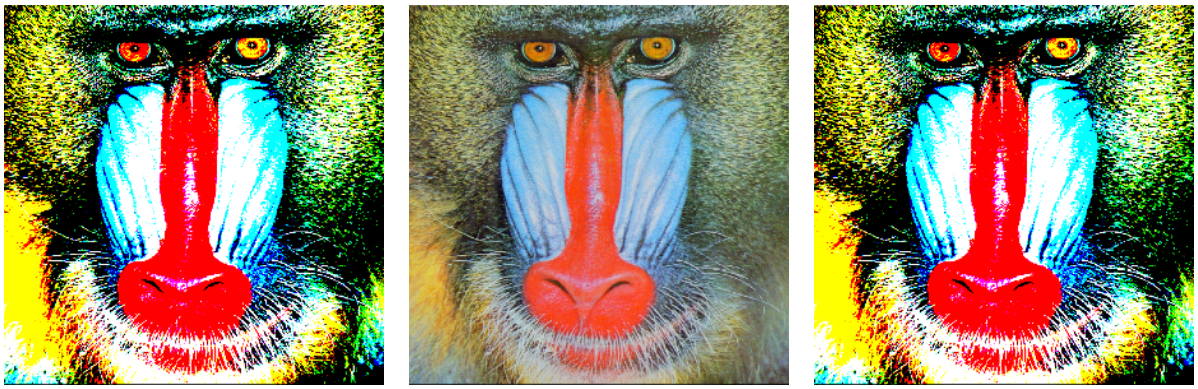

Figura 2 - Mensagem pequena e plano Bit 7, da esquerda para a direita: Plano de bits sem mensagem, imagem com mensagem, plano de bits com mensagem.

5.2 Mensagem grande

No caso em que utilizamos uma mensagem grande com 100.000 palavras, obtivemos resultados mais expressivos quanto à discrição da mensagem na imagem.

Ao ocultar a mensagem em bits menos significativos, isto é, nos planos de Bit 0 e Bit 2, obtivemos resultados semelhantes. Analisando a imagem final gerada, não é possível perceber visualmente nenhuma grande diferença que indique a modificação realizada na imagem (figura 3 e 4), contanto é totalmente perceptível a manipulação realizada ao visualizar as camadas de bits separadamente no qual os pixels alterados assumem uma organização retilínea quando comparado à mesma camada sem a mensagem.

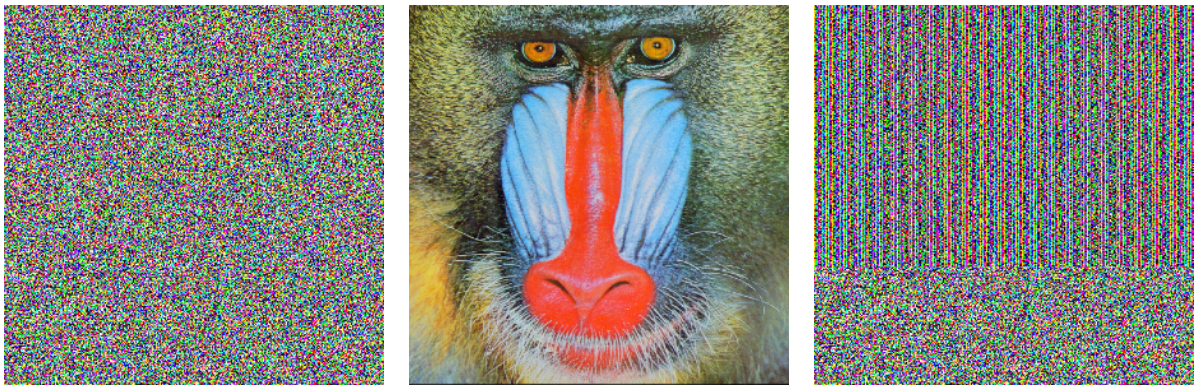

Figura 3 - Mensagem grande e plano Bit 0, da esquerda para a direita: Plano de bits sem mensagem, imagem com mensagem, plano de bits com mensagem.

Figura 4 - Mensagem grande e plano Bit 2, da esquerda para a direita: Plano de bits sem mensagem, imagem com mensagem, plano de bits com mensagem.

Já no caso em que ocultamos a mensagem no bit mais significativo, isto é, no Bit 7, fica claro que a imagem sofreu alterações (figura 5). Por possuir papel mais relevante na definição das cores, a imagem final apresentou mudanças drásticas na região em que se encontrava a mensagem, apresentando além da coloração diferente, o mesmo padrão retilíneo observado anteriormente nas camadas de bits separadas.

Além da imagem final apresentar alterações, ao observarmos a separadamente a camada de Bit 7 vemos que a figura do babuíno é substituída pelo padrão formado pela mensagem.

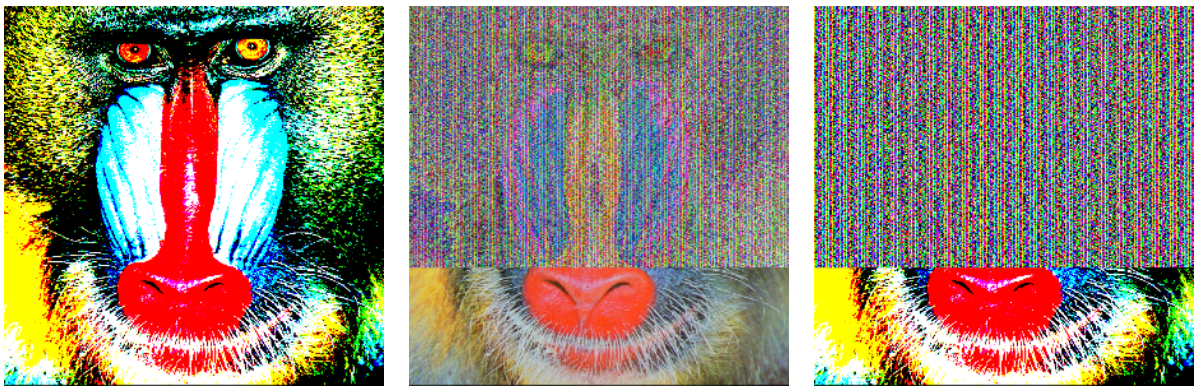

Figura 5 - Mensagem grande e plano Bit 7, da esquerda para a direita: Plano de bits sem mensagem, imagem com mensagem, plano de bits com mensagem.

6. CONCLUSÃO

Observando os resultados obtidos, podemos concluir que ocultar informações em bits menos significativos é altamente eficiente mesmo para mensagens muito grandes, sendo possível perceber a existência delas através da análise separada de cada camada de Bits. Já quando a mensagem é pequena, há grandes chances de não ser percebida a sua codificação.