

# SCC251 - Processamento de Imagens 2018

## Projeto Final Esteganografia BPCS

Relatório

Guilherme dos Santos Marcon  
NUSP 9293564

# Esteganografia BPCS

## Definições

- Imagem recipiente é a que esconderá a outra imagem.
- Imagem alvo é a que será escondida.
- Bloco é uma matriz 8x8 de números de 8 bits da imagem recipiente.
- Plano é uma matriz 8x8 de números binários.
- Informativo é quando um plano possui uma característica de formador de padrão ou forma, ele contribui para formar alguma parte padronizada da imagem.
- Complexo é quando um plano possui alternâncias vizinhas entre 0-1 suficientes para ser classificado como tendo características aleatórias, ou seja, não informativo.
- Conjugado é quando um plano não é complexo, então é realizado uma operação XOR bit a bit do plano com uma matriz padrão, deixando o plano complexo.
- Conjugation Map é um conjunto de dados que indica quais planos da imagem alvo inseridos no recipiente foi conjugado.

## Bit-Plane Complexity Segmentation

O tema do projeto é esteganografia, o ato de esconder informações em objetos ou data, de forma que sua detecção seja despercebida. Neste trabalho em si será utilizada o método Bit-Plane Complexity Segmentation, proposto por Eiji Kawaguchi and Richard O. Eason em 1998, para esconder uma imagem em outra imagem.

O método em si se aproveita de uma característica da visão humana, de focar no reconhecimento de padrões e formas, ignorando mudanças pequenas, ele faz isso segmentando a imagem recipiente em planos complexos e planos informativos, substituindo assim os planos complexos por informações da imagem a ser inserida.

## Obtendo as imagens

As imagens utilizadas foram retiradas do site [Pexels - Domínio Público](#), as imagens baixadas estavam no formato jpg, portanto foram convertidas para png no site [JPG to PNG](#) e a partir da conversão, as imagens foram reduzidas para testes com o programa [GIMP](#).

Todas as imagens utilizadas estão no repositório no [GitHub](#). Caso necessário, os links para o Pexels das imagens são:

- [Mountain](#)
- [Colorful Smoke](#)
- [Lion](#)

# Explicação detalhada

O método de **inserir uma imagem** em outra é composto dos seguintes passos:

- Transformar a imagem de Pure Binary Code para Canonical Gray Code, PBC é a codificação de bits normal, cada bit representando 2 elevado à sua significância, já o CGC codifica os bits em relação ao bit anterior, 0 sendo igualdade (esse bit é igual ao anterior) e 1 sendo diferença (esse bit é diferente do anterior). Essa transformação ajuda às mudanças serem ainda menos impactantes, já que um bloco será complexo apenas se houver muitas mudanças em relação ao anterior. Mais detalhes em [Two Binary Number Coding Systems](#).
- Preparar a imagem recipiente, criando mais uma dimensão na imagem para cada plano de bit, o intuito era otimizar, para não realizar bitshifts com tanta frequência, mas os testes mostraram que não fez diferença, então isso permanecerá por deixar o código mais claro.
- Inserir as dimensões da imagem alvo, é esperado que as dimensões das imagens sejam 3 inteiros de 32 bits, então são criados 2 planos 8x8 de binários para inserir esses inteiros. As primeiras 4 colunas do primeiro plano são preenchidas com valores aleatórios, as próximas 4 colunas com o primeiro número das dimensões, seguindo o raciocínio para o próximo plano. O detalhe é que as posições [0,0] e [0,1] do primeiro plano são usadas para representar se, respectivamente, o plano 0 e 1 foi conjugado.
- Inserir a imagem em si, parte mais custosa do process, como a imagem é composta de inteiros de 8 bits, para formar o plano 8x8, são utilizados os 8 próximos valores da imagem alvo. Se o plano gerado com esses 8 valores for informativo, ele é conjugado e é anotado no Conjugation Map que aquele plano foi conjugado, essa transformação de informativo para complexo é utilizada para que apenas planos complexos sejam inseridos no recipiente, já que trocar um plano complexo por outro causa mudanças relativamente imperceptíveis.
- Inserir o Conjugation Map, é necessário porque durante o processo de inserir a imagem alvo, é importante deixar o plano também complexo para inserção, já que não é o objetivo criar padrões artificiais na imagem recipiente, o objetivo é justamente o contrário, causar menos impacto possível. Os planos do Conjugation Map são compostos por uma coluna com valores aleatórios e sete colunas com os valores do mapa, na primeira posição dessa coluna de valores aleatórios é guardado se aquele plano foi conjugado ou não, novamente, conjugando para causar o menor impacto possível.
- Transformar de volta o recipiente de CGC para PBC.

Já o método para **recuperar a imagem** de um recipiente é composto pelos passos:

- Transformar o recipiente de PBC para CGC.
- Preparar a imagem recipiente, mesmo processo da inserção.
- Recuperar os dois blocos do tamanho da imagem alvo.
- Checar se é possível a imagem recipiente contar uma mensagem, isso é feito checando se nenhum dos números do tamanho é negativo e se o tamanho da imagem alvo é pelo menos menor que 60% da imagem recipiente. Mesmo se passar no teste, é possível ainda não existir uma imagem escondida, porém o algoritmo vai tentar recuperar mesmo assim.
- Recuperar todos os blocos complexos que correspondem à imagem alvo.
- Recuperar todos os blocos complexos que correspondem ao Conjugation Map.
- Aplicar a operação de conjugar nos blocos que o Conjugation Map marca como conjugados.
- Transformar os blocos recuperados na imagem alvo.

O método de **conjugar** é composto por uma operação XOR de um plano qualquer com o plano 8x8 cujas linhas se alternam entre '0 1 0 1 0 1 0 1' e '1 0 1 0 1 0 1 0'. Essa operação possui a propriedade de transformar um plano complexo em um plano informativo e vice-versa, e é utilizada para não inserir planos informativos na imagem recipiente.

# Resultados

Tabela com resultados, explicações:

- Coluna % concentra a porcentagem substituída da imagem recipiente.
- Coluna T mostra o tempo demorado em segundos.

#	Teste	Imagem Recipiente	Tamanho do Recipiente (bytes)	Imagem Alvo	Tamanho do Alvo (bytes)	%	T	RMSE	Mudança Visível?
1	hide12.in	100-colorful-smoke.png	4207048	12-colorful-smoke.png	647753	12	48	1.85	Não
2*	hide25.in	100-colorful-smoke.png	4207048	25-colorful-smoke.png	1380575	25	98	3.92	Sim*
3	hide40.in	25-colorful-smoke.png	1380575	10-colorful-smoke.png	552619	40	41	11.22	Sim
4	hide50.png	25-colorful-smoke.png	1380575	12-colorful-smoke.png	647753	48	---	---	---
5	hideInception1.png	100-lion.png	4489143	25-colorful-smoke.png	1380575	38	93	8.05	Não
6	hideInception2.png	100-mountain.png	21422985	Resultado do teste 5	5899266	14	252	1.86	Não

Outros testes relataram que quando a imagem recipiente possui uma grande extensão de planos pretos ou com cores mais escuras, qualquer inserção que chegue nos planos mais significativos vai gerar um resultado perceptível, como exemplo: o teste 2\* da tabela, mesmo que majoritariamente a imagem não possua mudança, é possível perceber uma leve deformação na região superior direita, onde as regiões escuras possuem comportamentos incomuns.

Outro ponto a observar é que dependendo dos métodos para conseguir as imagens png, ainda existirá uma certa compressão, e como a biblioteca “imageio” do Python não realiza nenhuma compressão ao salvar, uma mudança no tamanho da imagem pode ser percebida, como nos testes 5 e 6, o resultado do teste 5 possui um tamanho diferente que a do recipiente original.

Todos os testes de recuperação mostraram um RMSE de 0.0, significando que a imagem inserida e a recuperada são idênticas.

# Conclusão

O método é muito bom utilizando imagens sem compressão, consegue esconder uma quantidade de informação muito maior que o método dos bits menos significativos (LSB) e consegue recuperar a imagem inserida em um estado perfeito, mas é importante prestar atenção com alguns pontos:

- Pelo menos nessa implementação, ele é lento e não possui muitas brechas para otimização paralela, já que depende muito em achar um plano complexo para substituir.
- Achar uma boa imagem recipiente é importante, aquelas com uma grande porcentagem de regiões escuras não são boas, já que qualquer substituição num plano mais significativo causará uma propagação perceptível na linha daquele plano, o motivo é o Canonical Gray Code. Isso é verificado nos testes 2, 3 e 5, onde o recipiente do 5 é mais propício à receber uma imagem.

Fora esses pontos, é possível inserir uma imagem dentro de outra dentro de outra, como mostrado nos testes 5 e 6, a recuperação da primeira imagem inserida também é perfeita.