PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
ESCOLA POLITÉCNICA
GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PROJETO DE ESTEGANOGRAFIA COMPUTACIONAL: CONSTRUÇÃO E VALIDAÇÃO DE UM SISTEMA DE OCULTAÇÃO DE MENSAGEM EM PYTHON

CARLOS EDUARDO WATANABE NUNES

GOIÂNIA
2026

CARLOS EDUARDO WATANABE NUNES

PROJETO DE ESTEGANOGRAFIA COMPUTACIONAL: CONSTRUÇÃO E VALIDAÇÃO DE UM SISTEMA DE OCULTAÇÃO DE MENSAGEM EM PYTHON

Trabalho de Conclusão de Curso apresentado à Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Orientador:

Prof. Me. Ricardo de Andrade Kratz

Banca examinadora:

Prof. Dr. José Luiz de Freitas Junior

Prof. Me. Fabrício Schlag

GOIÂNIA

2026

CARLOS EDUARDO WATANABE NUNES

PROJETO DE ESTEGANOGRAFIA COMPUTACIONAL: CONSTRUÇÃO E VALIDAÇÃO DE UM SISTEMA DE OCULTAÇÃO DE MENSAGEM EM PYTHON

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola Politécnica e de Artes, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Ciência da Computação, em _____/_____/_____.

_____

Orientador - Prof. Me. Ricardo de Andrade Kratz

_____

Examinador 1 - Prof. Me. Fabrício Schlag

_____

Examinador 2 - Prof. Dr. José Luiz de Freitas Junior

GOIÂNIA

2026

AGRADECIMENTOS

Aos professores da banca de avaliação, Fabrício Schlag e José Luiz de Freitas Júnior, que ao longo da minha jornada acadêmica contribuíram de forma significativa para a minha formação, compartilhando conhecimento e experiência.

À Coordenação da Escola Politécnica da Pontifícia Universidade Católica de Goiás pelo suporte institucional e pelo auxílio na viabilização deste projeto.

Aos demais docentes do curso pelos ensinamentos, conselhos e sugestões que enriqueceram minha trajetória durante o período de graduação.

Aos meus familiares, que ao longo de toda a minha graduação estiveram ao meu lado, oferecendo incentivo, apoio incondicional e a motivação necessária para que eu chegasse até aqui.

Aos meus colegas de graduação, que compartilharam comigo momentos de aprendizado, desafios e conquistas ao longo desta jornada acadêmica.

Ao professor Ricardo de Andrade Kratz, orientador deste trabalho, pela dedicação, paciência, confiança e pelo direcionamento essencial que tornaram possível a realização deste projeto.

“A perfeição é alcançada pela prática constante.”
Miyamoto Musashi, O Livro dos Cinco Anéis, 1645

RESUMO

Este trabalho apresenta o projeto, a implementação e a validação de um sistema de esteganografia computacional desenvolvido em Python, com foco na ocultação segura de mensagens textuais em imagens digitais. O sistema integra duas técnicas de esteganografia amplamente estudadas na literatura - LSB (*Least Significant Bit*) e BPCS (*Bit-Plane Complexity Segmentation*).

Para a avaliação do desempenho das técnicas implementadas, foram utilizadas as métricas PSNR (*Peak Signal-to-Noise Ratio*), SSIM (*Structural Similarity Index*) e taxa de bits (*bitrate*), aplicadas a imagens em três resoluções distintas - baixa (74×131 pixels), média (368×656 pixels) e alta (1472×2626 pixels) - permitindo uma análise comparativa abrangente e sistemática.

Os resultados demonstraram que ambas as técnicas são capazes de ocultar informações com imperceptibilidade visual satisfatória, sendo que o método LSB apresentou maior preservação estrutural da imagem, enquanto o BPCS ofereceu maior capacidade de armazenamento por bloco. As métricas obtidas confirmaram que, quanto maior a resolução da imagem portadora, menor o impacto visual das alterações realizadas, evidenciando a importância do dimensionamento adequado do meio portador para aplicações esteganográficas.

Palavras-chave: Esteganografia; LSB; BPCS; Segurança da Informação; Python; PSNR; SSIM.

ABSTRACT

This work presents the design, implementation and validation of a computational steganography system developed in Python, focused on the secure concealment of textual messages within digital images. The system integrates two widely studied steganographic techniques - LSB (Least Significant Bit) and BPCS (Bit-Plane Complexity Segmentation).

To evaluate the performance of the implemented techniques, the metrics PSNR (Peak Signal-to-Noise Ratio), SSIM (Structural Similarity Index), and bitrate were applied to images at three distinct resolutions - low (74×131 pixels), medium (368×656 pixels), and high (1472×2626 pixels) - enabling a comprehensive and systematic comparative analysis.

The results demonstrated that both techniques are capable of concealing information with satisfactory visual imperceptibility. The LSB method showed greater structural preservation of the image, while BPCS offered greater storage capacity per block. The obtained metrics confirmed that the higher the resolution of the cover image, the lower the visual impact of the modifications, highlighting the importance of proper sizing of the cover medium for steganographic applications.

Keywords: Steganography; LSB; BPCS; Information Security; Python; PSNR; SSIM.

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

AES	*Advanced Encryption Standard*
BPCS	*Bit-Plane Complexity Segmentation*
LSB	*Least Significant Bit*
PNG	*Portable Network Graphics*
PSNR	*Peak Signal-to-Noise Ratio*
RGB	*Red, Green, Blue*
SHA	*Secure Hash Algorithm*
SSIM	*Structural Similarity Index*
B	*Byte*
s	Segundo
dB	*Decibel*

Sumário

CAPÍTULO I – INTRODUÇÃO

A esteganografia é uma técnica de ocultação de informações que tem como objetivo esconder a existência de uma mensagem dentro de um meio portador, como imagens, áudios ou vídeos. Diferentemente da criptografia, que visa tornar o conteúdo ilegível, a esteganografia busca dissimular a própria comunicação, sendo uma abordagem relevante no contexto da segurança da informação.

Com o avanço das tecnologias digitais e o aumento do volume de dados trafegados em redes, surgem novos desafios relacionados à privacidade e à proteção da informação. Nesse cenário, técnicas esteganográficas tornam-se importantes tanto para aplicações legítimas, como comunicação confidencial e proteção de propriedade intelectual, quanto para a análise de possíveis usos indevidos, como exfiltração de dados e comunicação encoberta.

Este trabalho tem como foco a construção e validação de um sistema de esteganografia computacional utilizando a linguagem Python. A proposta envolve a implementação prática de técnicas conhecidas e a análise de seu desempenho por meio de métricas consolidadas, permitindo uma avaliação mais aprofundada dos aspectos envolvidos na ocultação de informações.

1.1 Contextualização

Na era digital, a esteganografia passou a ser aplicada principalmente em mídias digitais, como imagens, áudios e vídeos, explorando características desses meios para ocultar dados sem comprometer significativamente sua qualidade perceptível. Essa abordagem permite a transmissão de informações de forma discreta, dificultando sua identificação por terceiros.

A esteganografia distingue-se da criptografia pelo seu objetivo principal. Enquanto a criptografia visa garantir a confidencialidade da informação por meio da transformação do conteúdo em uma forma incompreensível para usuários não autorizados, a esteganografia procura ocultar a existência da própria informação, inserindo-a de maneira imperceptível em outros meios digitais, como imagens, áudios ou vídeos. Em razão dessa característica, ambas as técnicas podem ser utilizadas conjuntamente, de modo que a criptografia protege o conteúdo da mensagem e a

esteganografia reduz a percepção de sua presença, fortalecendo os mecanismos de segurança da informação (KESSLER; HOSMER, 2011; ALMEIDA, 2011).

Além disso, a esteganografia envolve desafios técnicos relacionados ao equilíbrio entre capacidade de ocultação, imperceptibilidade e robustez. Esses três fatores são fundamentais na avaliação de sistemas esteganográficos e representam um dos principais *trade-offs* considerados no desenvolvimento dessas técnicas.

1.2 Objetivos

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é projetar, implementar e validar um sistema computacional de esteganografia em Python capaz de ocultar mensagens em imagens digitais de forma segura e imperceptível, com base em técnicas consolidadas e avaliação experimental.

1.2.2 Objetivos Específicos

- Apresentar os conceitos fundamentais da esteganografia e suas principais métricas de avaliação;
- Revisar as principais técnicas de esteganografia digital, com ênfase nos métodos LSB e BPCS;
- Implementar um protótipo funcional para ocultação de mensagens em imagens digitais;
- Avaliar o desempenho das técnicas por meio de métricas como PSNR, SSIM e taxa de bits;
- Comparar os resultados obtidos, analisando os *trade-offs* entre capacidade de ocultação, imperceptibilidade e robustez;
- Discutir as implicações de segurança e possíveis aplicações da esteganografia no contexto atual.

1.3 Justificativa

A crescente dependência de sistemas digitais e o aumento das ameaças cibernéticas tornam essencial o desenvolvimento de técnicas que garantam a segurança e a privacidade das informações. Nesse contexto, a esteganografia se

destaca como uma abordagem complementar, permitindo a ocultação da própria existência da comunicação.

Adicionalmente, o estudo da esteganografia é importante não apenas para aplicações legítimas, mas também para compreender seus possíveis usos indevidos. A análise dessas técnicas contribui para o desenvolvimento de métodos de detecção e prevenção, fortalecendo a segurança da informação como um todo.

Dessa forma, este trabalho justifica-se pela necessidade de aplicar, na prática, os conceitos teóricos estudados, por meio da implementação e validação de um sistema esteganográfico, permitindo avaliar seu desempenho e suas limitações em cenários controlados.

1.4 Questões de Pesquisa

Com base nos objetivos propostos, este trabalho busca responder às seguintes questões de pesquisa:

- Q1: Qual técnica de esteganografia (LSB ou BPCS) apresenta melhor equilíbrio entre imperceptibilidade, capacidade de ocultação e robustez?
- Q2: Em que medida as métricas PSNR, SSIM e taxa de bits são eficazes na avaliação da qualidade de sistemas esteganográficos?
- Q3: É possível desenvolver um sistema esteganográfico eficiente em Python que mantenha níveis satisfatórios de segurança e invisibilidade dos dados ocultos?

CAPÍTULO II – FUNDAMENTAÇÃO TEÓRICA

Neste capítulo é apresentada a revisão bibliográfica necessária para embasar o desenvolvimento do sistema proposto, abordando os principais conceitos, técnicas e métricas relacionadas à esteganografia computacional.

2.1 Definição e História da Esteganografia

A esteganografia pode ser definida como a arte e ciência de ocultar informações dentro de um meio, de forma que a própria existência da mensagem não seja perceptível. Diferentemente da criptografia, que tem como objetivo tornar a mensagem incompreensível, a esteganografia busca esconder a comunicação, atuando de maneira mais discreta no contexto da segurança da informação (ZOCHIO, 2016).

Historicamente, a esteganografia remonta à Antiguidade, com relatos como os descritos por Heródoto, envolvendo mensagens ocultas em mensageiros. Posteriormente, no século XV, Johannes Trithemius formalizou o estudo da área em sua obra Steganographia, considerada um marco teórico importante. Com o avanço da tecnologia, essas técnicas evoluíram para métodos digitais aplicados em diferentes tipos de mídia, como imagens, áudio e vídeo (KESSLER; HOSMER, 2011).

2.2 Tipos de Esteganografia

A esteganografia pode ser classificada de acordo com o tipo de mídia utilizada como meio de ocultação. Entre os principais tipos, destacam-se a esteganografia em imagens, áudios, vídeos e textos, sendo as imagens digitais as mais utilizadas devido à sua alta redundância de dados e facilidade de manipulação.

Em imagens, por exemplo, a informação é inserida diretamente nos pixels, enquanto em áudio pode ser incorporada em amostras sonoras. Além disso, a esteganografia está relacionada às técnicas de marca d'água digital (*watermarking*), que são utilizadas para proteção de propriedade intelectual, embora possuam objetivos distintos, como autenticação e rastreabilidade (SION, 2018).

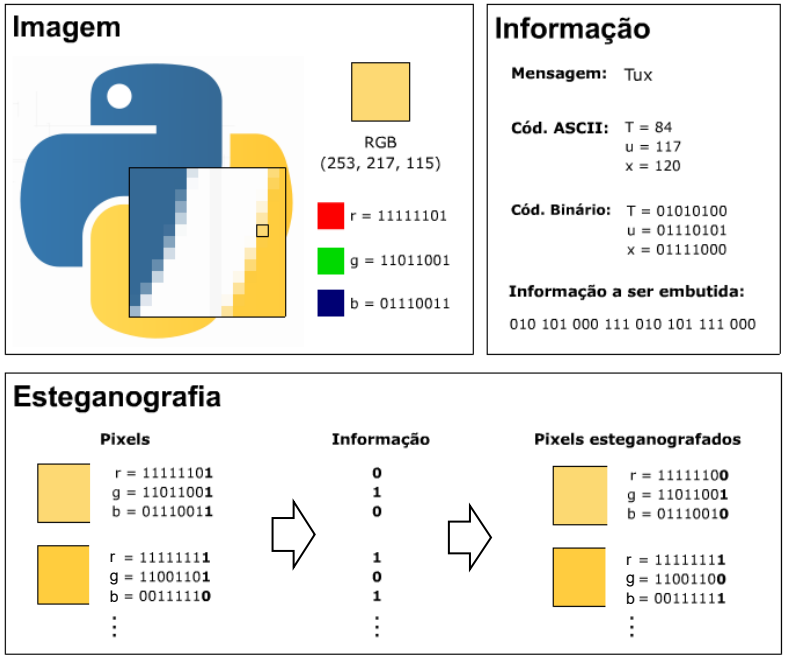
2.3 Técnicas de Esteganografia

2.3.1 Least Significant Bit (LSB)

A técnica de *Least Significant Bit* (LSB) é uma das abordagens mais simples e amplamente utilizadas na esteganografia de imagens digitais. Ela consiste na substituição dos bits menos significativos dos pixels por bits da mensagem secreta, de modo a minimizar alterações perceptíveis ao olho humano.

Em imagens no formato RGB, cada pixel é composto por três canais de cor (vermelho, verde e azul), permitindo que cada um desses canais seja utilizado para armazenar dados. Essa característica proporciona uma alta capacidade de ocultação, mantendo uma baixa distorção visual, o que torna o método eficiente para aplicações práticas (KESSLER; HOSMER, 2011).

Figura 1 - Exemplo do funcionamento da esteganografia LSB.



Fonte: Adaptada de [ALMEIDA, 2011]

Exemplo onde a string "Tux" é embutida em uma imagem com o logotipo da linguagem Python.

A Figura 1 ilustra o processo de esteganografia usando a técnica Least Significant Bit (LSB) em uma imagem com o logotipo do Python. Nessa abordagem, cada pixel da imagem é representado por três canais de cor, sendo eles, vermelho (R), verde (G) e azul (B); cada um armazenado em um byte de 8 bits. A mensagem

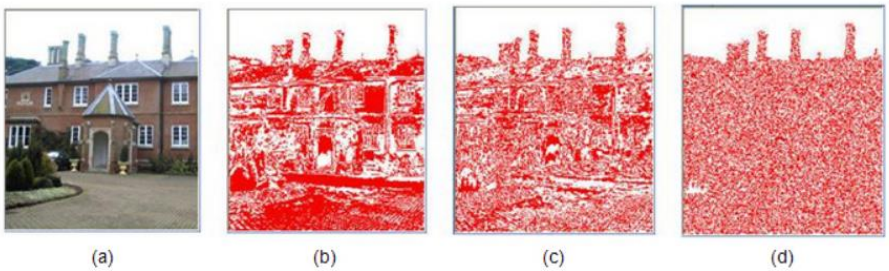
secreta "Tux" é convertida em sua representação binária a partir dos valores ASCII de cada caractere (T = 01010100, u = 01110101, x = 01111000), formando uma sequência de bits a ser embutida. Para cada pixel processado, o bit menos significativo de cada canal de cor é substituído por um bit da mensagem, alterando o valor do canal em no máximo uma unidade. Essa variação é imperceptível ao olho humano, pois representa uma mudança mínima na intensidade da cor, garantindo que a imagem resultante seja visualmente indistinguível da original enquanto carrega a informação oculta.

2.3.2 Bit-Plane Complexity Segmentation (BPCS)

O método *Bit-Plane Complexity Segmentation* (BPCS) é uma técnica mais avançada de esteganografia que explora a complexidade visual dos planos de bits de uma imagem digital. Diferentemente do LSB, que atua apenas nos bits menos significativos, o BPCS permite a substituição de regiões consideradas complexas em diferentes planos de bits.

Essa abordagem baseia-se na ideia de que regiões visualmente complexas, que apresentam maior nível de ruído, podem ser modificadas sem causar alterações perceptíveis ao olho humano. Como resultado, o BPCS oferece maior capacidade de ocultação de dados em comparação com técnicas mais simples, mantendo a qualidade da imagem (KAWAGUCHI, 2023).

Figura 2 - Exemplo de planos de bits.



Fonte: [KAWAGUCHI, 2023]

(a) mostra a imagem original; (b) mostra um plano de bits mais próximo ao mais significativo; (c) mostra um plano intermediário; (d) mostra um plano de bits mais próximo ao menos significativo.

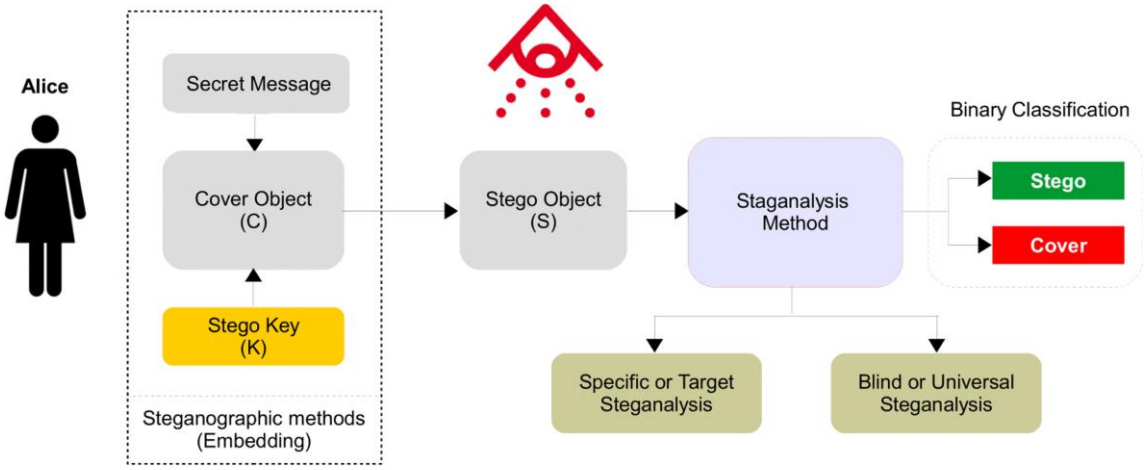
A Figura 2 demonstra o conceito de Bit-Plane Complexity Segmentation (BPCS), técnica que decompõe uma imagem em camadas binárias chamadas planos de bits, onde cada plano corresponde a uma posição de bit específica nos valores dos pixels. Como ilustrado nas subfiguras, o plano mais próximo ao bit mais significativo, representado em (b), preserva a estrutura visual da imagem original (a), pois esse bit é o que mais contribui para o valor de cada pixel — qualquer alteração nele causaria grande distorção perceptível. À medida que se avança em direção aos bits menos significativos, como mostrado em (c) e (d), os planos passam a exibir padrões cada vez mais ruidosos e aparentemente aleatórios, com baixa correlação visual entre pixels vizinhos. É justamente essa característica que torna os planos menos significativos ideais para a esteganografia BPCS, onde blocos com alta complexidade visual nesses planos podem ser substituídos por dados secretos sem que a alteração seja perceptível ao olho humano, uma vez que a aleatoriedade já presente nesses planos mascara naturalmente a informação embutida.

2.4 Esteganálise

A esteganálise corresponde ao conjunto de técnicas utilizadas para detectar a presença de informações ocultas em mídias digitais. Seu principal objetivo é identificar padrões ou anomalias que indiquem a existência de dados escondidos em um meio aparentemente comum.

Para isso, são utilizados métodos como análise estatística, incluindo testes de qui-quadrado, além de abordagens baseadas em aprendizado de máquina, que permitem identificar padrões mais complexos. Os ataques podem ser classificados como passivos, quando visam apenas detectar a presença da mensagem, ou ativos, quando tentam extrair ou corromper os dados ocultos (KESSLER; HOSMER, 2011).

Figura 3 - Esteganálise.



Fonte: [KHEDDAR et al., 2024]

Na Figura 3, Alice esconde uma mensagem secreta dentro de um arquivo comum utilizando uma chave e um método de esteganografia. O resultado é um objeto esteganográfico que parece normal, mas contém dados ocultos. Esse arquivo é então analisado por um método de esteganálise que irá verificá-lo, realizando assim uma classificação binária, onde o classificará em: arquivo esteganográfico ou arquivo comum.

2.5 Métricas de Avaliação

A avaliação de sistemas esteganográficos é realizada por meio de métricas que medem tanto a qualidade do meio portador quanto a eficiência da ocultação de dados. Entre as principais métricas utilizadas, destacam-se:

- PSNR (*Peak Signal-to-Noise Ratio*): avalia a qualidade da imagem comparando o sinal original com o sinal modificado;
- SSIM (*Structural Similarity Index*): mede a similaridade estrutural entre imagens com base na percepção humana;
- Capacidade de ocultação: quantidade de dados que podem ser inseridos sem comprometer a qualidade;
- Taxa de erro de extração: mede a precisão na recuperação da mensagem;
- Resistência a ataques: avalia a robustez frente a transformações como compressão e redimensionamento;

- Tempo de processamento: indica a eficiência computacional do método.

Essas métricas são amplamente utilizadas na literatura para avaliar a eficácia e a viabilidade de sistemas esteganográficos (SION, 2018).

CAPÍTULO III – METODOLOGIA

Este capítulo descreve os procedimentos metodológicos adotados para o desenvolvimento, implementação e validação do sistema de esteganografia proposto. São apresentados a classificação da pesquisa, as etapas de desenvolvimento, a metodologia de implementação dos algoritmos LSB e BPCS, os procedimentos experimentais, as métricas de avaliação e os critérios de validação da solução.

3.1 Classificação da Pesquisa

A presente pesquisa caracteriza-se como aplicada, pois tem como finalidade desenvolver uma solução computacional prática para ocultação de mensagens em imagens digitais, utilizando técnicas de esteganografia. O estudo ultrapassa a abordagem puramente teórica, resultando na implementação e validação de um protótipo funcional.

Quanto à abordagem do problema, a pesquisa possui caráter quantitativo, uma vez que a análise da eficiência das técnicas LSB e BPCS será realizada com base em métricas numéricas objetivas, como PSNR, SSIM, taxa de bits e taxa de erro de extração.

No que se refere aos procedimentos técnicos, o trabalho é classificado como experimental, pois envolve a implementação prática de algoritmos, execução de testes controlados e comparação sistemática de resultados. Também apresenta caráter exploratório, ao investigar o comportamento das técnicas em diferentes cenários de uso, variando dimensões de imagens e tamanhos de mensagens.

3.2 Etapas do Desenvolvimento do Trabalho

O desenvolvimento do trabalho foi estruturado em etapas sequenciais, visando garantir organização e rigor metodológico.

Inicialmente, realizou-se a revisão bibliográfica acerca das técnicas de esteganografia digital, com ênfase nos métodos LSB e BPCS, bem como no estudo das métricas de avaliação utilizadas na literatura.

Em seguida, foram definidos os requisitos do sistema, contemplando as seguintes funcionalidades principais:

- Inserção de mensagem textual em imagem digital;
- Extração da mensagem oculta;
- Cálculo automático de métricas de avaliação (PSNR, SSIM e taxa de bits);
- Comparação dos resultados entre as técnicas implementadas.

Posteriormente, foi elaborado o planejamento da arquitetura do sistema, definindo-se a estrutura modular do software, os fluxos de processamento e as bibliotecas necessárias para manipulação de imagens. Após essa etapa, iniciou-se a implementação dos algoritmos LSB e BPCS, seguida da realização de testes experimentais controlados. Por fim, os resultados obtidos foram analisados quantitativamente e comparados.

3.3 Metodologia de Desenvolvimento do Software

O desenvolvimento do software seguiu uma abordagem incremental, na qual cada módulo foi implementado e validado individualmente antes da integração completa do sistema. Inicialmente, implementou-se o módulo de manipulação de imagens, responsável por carregar imagens no formato PNG, converter a imagem em matriz de pixels, separar e manipular canais RGB e salvar a imagem esteganografada.

Em seguida, foi implementado o algoritmo LSB, permitindo a substituição dos bits menos significativos dos pixels pelos bits da mensagem secreta. Posteriormente, implementou-se o algoritmo BPCS, baseado na decomposição da imagem em planos de bits e na substituição de regiões classificadas como complexas.

Após a implementação de cada técnica, foram realizados testes unitários para verificar: correta inserção da mensagem; recuperação integral da mensagem original; e integridade da imagem resultante. Por fim, integrou-se o módulo de cálculo das métricas, automatizando a comparação entre imagem original e imagem esteganografada.

3.4 Técnica de Esteganografia Adotada

3.4.1 Método Least Significant Bit (LSB)

A técnica LSB foi implementada por meio da substituição do bit menos significativo de cada canal RGB pelos bits da mensagem convertida em binário. O

processo foi realizado em duas etapas principais: conversão da mensagem textual para sua representação binária e inserção da informação criptografada por meio da substituição sequencial dos bits menos significativos dos canais de cor vermelho (R), verde (G) e azul (B) de cada pixel da imagem.

3.4.2 Método Bit-Plane Complexity Segmentation (BPCS)

A técnica BPCS foi implementada por meio da divisão da imagem em blocos 8×8 pixels em cada canal de cor. Para cada bloco, é realizada a substituição do bit menos significativo de cada pixel dentro do bloco pelos bits da mensagem, explorando a redundância das regiões de alta complexidade visual.

3.5 Ferramentas e Ambiente de Desenvolvimento

O sistema foi desenvolvido utilizando as seguintes ferramentas e bibliotecas:

- Linguagem: Python;
- Ambiente: Google Colaboratory (Colab);
- PIL/Pillow: manipulação de imagens;
- NumPy: operações com arrays de pixels;
- Matplotlib: visualização de resultados;
- Pandas: organização e exibição de métricas em tabelas.

3.6 Procedimentos Experimentais

Os experimentos foram conduzidos utilizando a imagem "Cherry" — uma fotografia de cerejeiras com origem japonesa e o Monte Fuji ao fundo — em três resoluções distintas: baixa (74×131 pixels), média (368×656 pixels) e alta (1472×2626 pixels). Essa variação permite analisar o comportamento das técnicas em função do tamanho do meio portador.

Em cada teste, uma mensagem de texto foi criptografada com AES-256 e replicada 60 vezes para ampliar a quantidade de bits inseridos e tornar as alterações mensuráveis pelas métricas. Após a inserção, a imagem modificada foi salva e as métricas PSNR, SSIM e *bitrate* foram calculadas automaticamente.

3.7 Métricas de Avaliação

Foram utilizados três métricas principais para avaliar o impacto das técnicas sobre a qualidade visual das imagens:

- PSNR: razão sinal-ruído de pico, que mede a distorção entre a imagem original e a modificada. Valores acima de 40 dB são considerados imperceptíveis;
- SSIM: índice de similaridade estrutural, que varia de 0 a 1, sendo 1 indicativo de imagens idênticas;
- *Bitrate*: quantidade de bits inseridos por pixel, que representa a capacidade de ocultação.

3.8 Validação da Solução Proposta

A validação foi realizada em duas etapas: validação funcional, verificando se a mensagem extraída corresponde à mensagem original; e validação criptográfica, confirmando que a recuperação só é possível com a senha correta. Foram realizados testes com senha correta e senha incorreta para ambas as técnicas.

3.9 Considerações Finais da Metodologia

A metodologia adotada permitiu o desenvolvimento sistematizado do sistema, garantindo que cada componente fosse implementado, testado e integrado de forma controlada. As técnicas de esteganografia LSB e BPCS, aliada à avaliação por métricas objetivas, fornece base sólida para a análise comparativa apresentada nos capítulos seguintes.

CAPÍTULO IV – DESENVOLVIMENTO / IMPLEMENTAÇÃO

4.1 Considerações Iniciais

Este capítulo apresenta o desenvolvimento e a implementação do sistema de esteganografia digital proposto neste trabalho. O sistema foi projetado para ocultar mensagens textuais em imagens digitais utilizando duas técnicas distintas: LSB (*Least Significant Bit*) e BPCS (*Bit-Plane Complexity Segmentation*).

O sistema foi desenvolvido em Python, adotando uma abordagem modular e orientada a objetos.

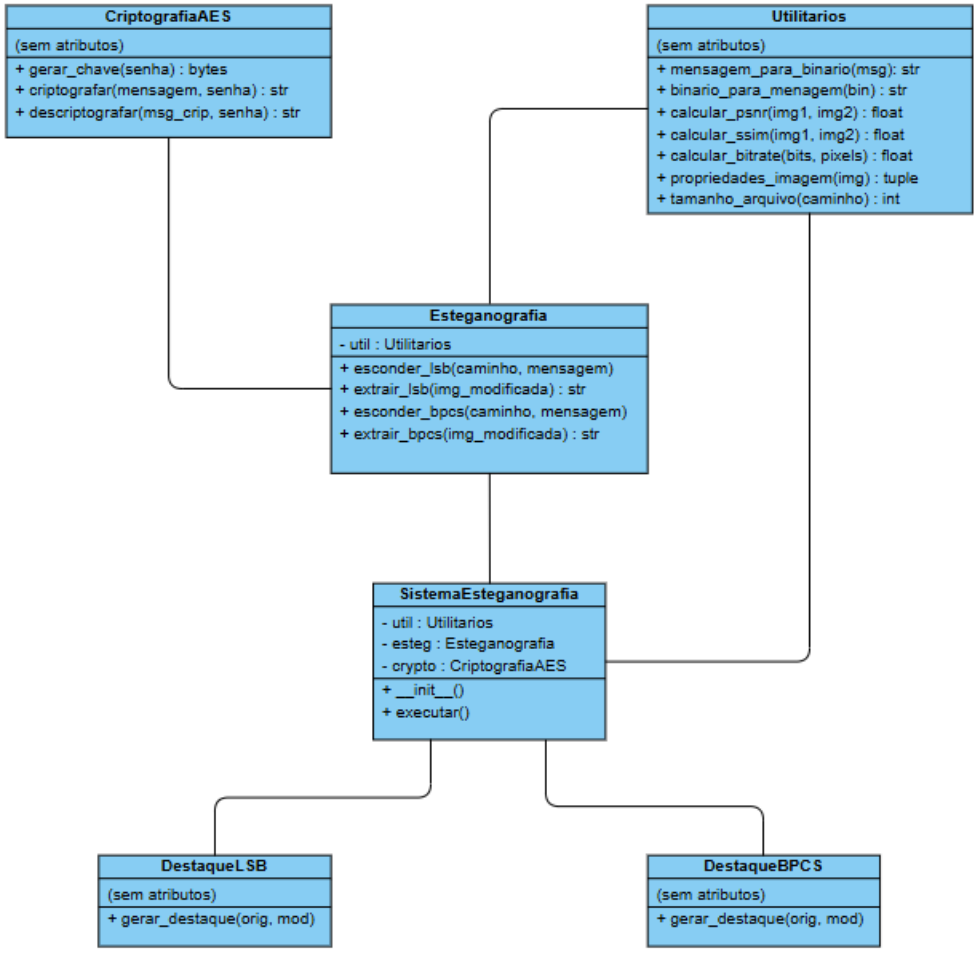
4.2 Arquitetura do Sistema

O sistema foi estruturado em dois módulos principais:

- Módulo de Esteganografia (LSB e BPCS)
- Módulo Principal de Execução e Avaliação

A Figura 4 apresenta o diagrama de classes do sistema implementado, representando as cinco classes e seus relacionamentos. A classe SistemaEsteganografia atua como controladora central, agregando as classes Utilitarios, Esteganografia e CriptografiaAES. As classes DestaqueLSB e DestaqueBPCS são instanciadas pelo sistema para geração das imagens de visualização das alterações realizadas nos pixels.

Figura 4 - Diagrama de classes do sistema de esteganografia.



Fonte: Autoria própria

Conforme ilustrado na Figura 4, o relacionamento entre Esteganografia e Utilitarios é de composição: a classe Esteganografia mantém internamente um atributo util do tipo Utilitarios, garantindo acesso direto às funções de conversão e cálculo de métricas sem necessidade de instanciação externa. A classe SistemaEsteganografia agrega as três dependências principais por meio do método __init__(), centralizando a orquestração do fluxo completo de processamento.

A Tabela 1 detalha a responsabilidade de cada classe:

Tabela 1 - Tabela descritiva com a responsabilidade de cada classe.

Classe	Responsabilidade
CriptografiaAES	Geração de chave SHA-256, criptografia e descriptografia AES no modo ECB com codificação Base64
Utilitarios	Conversões binárias, cálculo de PSNR, SSIM e bitrate, extração de propriedades da imagem e tamanho de arquivo
Esteganografia	Inserção e extração de mensagens via LSB e BPCS; utiliza Utilitarios por composição
SistemaEsteganografia	Orquestra o fluxo completo: entrada, criptografia, esteganografia, extração, descriptografia e métricas
DestaqueLSB / DestaqueBPCS	Geram imagens com pixels alterados destacados em verde fluorescente sobre fundo em escala de cinza

Fonte: Autoria própria

O fluxo operacional segue a seguinte sequência: entrada da mensagem e chave secreta pelo usuário; geração da chave criptográfica SHA-256; criptografia AES e codificação Base64; inserção na imagem via LSB ou BPCS; extração e descriptografia com recuperação da mensagem original; e cálculo automático das métricas de qualidade.

4.3 Implementação da Técnica LSB

A técnica LSB consiste na substituição do bit menos significativo de cada pixel da imagem pelos bits da mensagem criptografada.

4.3.1 Processo de Inserção

A implementação do processo de inserção LSB é apresentada pela Figura 5:

Figura 5 - Função que oculta a mensagem nos bits menos significativos.

```python
# Função que oculta a mensagem nos bits menos significativos da imagem
def esconder_lsb(caminho, mensagem):
    inicio = time.time()
    img = Image.open(caminho)
    img_array = np.array(img)
    binario = mensagem_para_binario(mensagem) + "11111110"
    pixels = img_array.flatten()

    for i in range(len(binario)):
        pixels[i] = (pixels[i] & 254) | int(binario[i])

    img_modificada = pixels.reshape(img_array.shape)
    tempo = time.time() - inicio
    return img_array, img_modificada, binario, tempo
```

Fonte: Autoria própria

A alteração no último bit provoca mudanças mínimas na intensidade do pixel, tornando a modificação visualmente imperceptível ao olho humano.

4.3.2 Processo de Extração

Na extração, os bits menos significativos são coletados sequencialmente, a sequência binária é reconstruída e convertida em texto. O marcador "11111110" sinaliza o fim da mensagem:

Figura 6 - Função que extrai os bits escondidos da imagem modificada.

```python
# Função que extrai os bits escondidos da imagem modificada
def extrair_lsb(img_modificada):
    # Transforma a imagem em lista linear de pixels
    pixels = img_modificada.flatten()
    bits = ""
    # Para cada pixel, pega o último bit (onde a mensagem foi escondida)
    for pixel in pixels:
        bits += str(pixel & 1)
    return bits
```

Fonte: Autoria própria

Figura 7 - Função que converte uma sequência binária de volta para texto.

```python
# Converte uma sequência binária de volta para texto
def binario_para_mensagem(binario):
    # Divide o binário em blocos de 8 bits (cada bloco é um caractere)
    blocos = [binario[i:i+8] for i in range(0, len(binario), 8)]
    mensagem = ""
    for bloco in blocos:
        # O marcador 11111110 indica o fim da mensagem
        if bloco == "11111110":
            break
        # Converte o bloco binário para o caractere correspondente
        mensagem += chr(int(bloco, 2))
    return mensagem
```

Fonte: Autoria própria

4.4 Implementação da Técnica BPCS

A técnica BPCS é baseada na análise da complexidade dos planos de bits da imagem. A implementação percorre a imagem em blocos 8×8 em cada canal de cor RGB, substituindo os bits menos significativos de cada bloco pelos bits da mensagem:

Figura 8 - Função que oculta a mensagem em blocos de 8x8 pixels (BPCS).

```python
# Função que oculta a mensagem em blocos de 8x8 pixels (BPCS)
def esconder_bpcs(caminho, mensagem):

    inicio = time.time()

    # Abre a imagem e converte para array
    img = Image.open(caminho)
    img_array = np.array(img)

    # Converte a mensagem para binário com marcador de fim
    binario = mensagem_para_binario(mensagem) + "11111110"

    img_modificada = img_array.copy()
    indice = 0

    # Percorre cada canal de cor (R, G, B)
    for canal in range(3):
        # Pega o plano de cor atual
        plano = img_array[:, :, canal]
        # Percorre a imagem em blocos de 8x8 pixels
        for i in range(0, plano.shape[0], 8):
            for j in range(0, plano.shape[1], 8):

                bloco = plano[i:i+8, j:j+8]

                if bloco.shape != (8, 8):
                    continue

                if indice < len(binario):

                    trecho = binario[indice:indice+64]

                    if len(trecho) < 64:
                        trecho = trecho.ljust(64, '0')

                    # Converte os bits em um array 8x8
                    plano_bits = np.array(list(trecho)).astype(np.uint8).reshape(8, 8)

                    bloco = (bloco & 254) | plano_bits
                    img_modificada[i:i+8, j:j+8, canal] = bloco
                    indice += 64

    tempo = time.time() - inicio

    return img_array, img_modificada, binario, tempo
```

Fonte: Autoria própria

4.5 Avaliação de Qualidade da Imagem

Após a inserção da mensagem, foram calculadas métricas para avaliar a integridade visual da imagem. Os valores de cada métrica foram determinados a partir da comparação entre a imagem original e a imagem esteganografada, permitindo mensurar as alterações provocadas pelo processo de inserção dos dados e avaliar o grau de similaridade entre ambas.

4.5.1 PSNR (Peak Signal-to-Noise Ratio)

O PSNR mede a diferença entre a imagem original e a modificada. Valores altos indicam baixa distorção. A implementação utiliza a fórmula padrão:

Figura 9 - Função que calcula o PSNR entre imagens.

```python
# Calcula o PSNR entre a imagem original e a imagem modificada
def calcular_psnr(img1, img2):
    # Calcula o erro quadrático médio entre os pixels
    erro = np.mean((img1 - img2) ** 2)

    if erro == 0:
        return float("inf")

    # Valor máximo de pixel em uma imagem de 8 bits
    maximo = 255.0
    # Fórmula padrão do PSNR
    return 20 * np.log10(maximo / np.sqrt(erro))
```

Fonte: Autoria própria

4.5.2 SSIM (Structural Similarity Index)

O SSIM mede a similaridade estrutural entre as imagens, considerando luminância, contraste e estrutura. Varia de 0 a 1, sendo 1 indicativo de imagens idênticas.

Figura 10 - Função que calcula o SSIM entre imagens.

```python
# Calcula o SSIM entre a imagem original e a modificada
def calcular_ssim(img1, img2):
    # Converte as imagens para float para maior precisão
    img1 = img1.astype(np.float64)
    img2 = img2.astype(np.float64)

    media1 = np.mean(img1)
    media2 = np.mean(img2)

    var1 = np.var(img1)
    var2 = np.var(img2)

    # Calcula a covariância entre as duas imagens
    cov = np.mean((img1 - media1) * (img2 - media2))
    # Constantes de estabilidade da fórmula SSIM
    c1 = (0.01 * 255) ** 2
    c2 = (0.03 * 255) ** 2
    # Fórmula do SSIM
    return ((2 * media1 * media2 + c1) * (2 * cov + c2)) / ((media1**2 + media2**2 + c1) * (var1 + var2 + c2))
```

Fonte: Autoria própria

4.5.3 Bitrate

O *Bitrate* indica a quantidade de bits inseridos por pixel, representando a capacidade de ocultação do sistema. Dessa forma, existe um limite máximo de dados que podem ser incorporados à imagem. Caso o tamanho da mensagem exceda essa capacidade, ainda que por um único bit, o algoritmo poderá tentar acessar posições inexistentes nos canais da imagem, resultando em erros de execução e impossibilitando a conclusão do processo de inserção dos dados.

Figura 11 - Função que calcula o *Bitrate* entre imagens.

```python
# Calcula o bitrate: quantos bits foram escondidos por pixel
def calcular_bitrate(total_bits, total_pixels):
    return total_bits / total_pixels
```

Fonte: Autoria própria

4.6 Fluxo Completo de Funcionamento

O fluxo operacional completo do sistema pode ser descrito sequencialmente: Mensagem Original → Criptografia AES → Codificação Base64 → Conversão para binário → Inserção (LSB ou BPCS) → Geração da imagem esteganografada → Extração → Decodificação Base64 → Descriptografia AES → Mensagem Original Restaurada.

Essa estrutura garante dupla camada de segurança: segurança criptográfica (AES) e segurança por ocultação (esteganografia).

4.7 Considerações Sobre Segurança

A utilização da esteganografia, em conjunto com a criptografia AES, fortalece significativamente a segurança da informação ao dificultar a própria percepção da existência da comunicação. Nesse contexto, a esteganografia atua como a primeira camada de proteção, ao ocultar os dados dentro de uma imagem digital de forma imperceptível, reduzindo a probabilidade de detecção ou interceptação da mensagem. Dessa forma, mesmo que haja suspeita ou identificação do uso da esteganografia, o conteúdo ainda permanece protegido pela criptografia, garantindo a confidencialidade dos dados em um nível adicional de segurança.

4.8 Considerações Finais do Capítulo

Neste capítulo foi apresentada a implementação completa do sistema proposto, detalhando a arquitetura, os algoritmos utilizados e os mecanismos de segurança incorporados. A integração entre criptografia AES e técnicas de esteganografia permitiu o desenvolvimento de um sistema funcional, seguro e mensurável por métricas objetivas de qualidade.

CAPÍTULO V – TESTES E RESULTADOS

5.1 Ambiente de Testes

Os testes foram realizados no ambiente Google Colab, utilizando linguagem Python e bibliotecas específicas para manipulação de imagens, criptografia e análise de métricas. As imagens utilizadas estavam no formato PNG, garantindo ausência de compressão com perdas.

Foram avaliadas duas técnicas de esteganografia (LSB e BPCS), com todas as mensagens criptografadas com AES-256 antes da inserção. A imagem portadora utilizada foi a "Cherry", retirada do Pinterest, e disponibilizada em três resoluções — baixa (74×131 pixels), média (368×656 pixels) e alta (1472×2626 pixels) — permitindo uma avaliação abrangente do comportamento das técnicas em função do tamanho do meio portador.

5.2 Testes com LSB

No método LSB, a mensagem criptografada foi incorporada à imagem por meio da substituição dos bits menos significativos dos canais RGB de seus pixels. Com o objetivo de ampliar os efeitos da inserção de dados e tornar mais evidentes as diferenças observadas nas análises comparativas, a mensagem criptografada foi repetida 600 vezes, elevando substancialmente a quantidade de bits alterados durante o processo de esteganografia.

Figura 12 - LSB resolução: original (736×1313 pixels) vs modificada.

Fonte: Autoria própria

Figura 13 - Resultados do LSB por resolução.

```
===== LSB =====
Digite a mensagem para LSB: Olá mundo!
Digite a senha para LSB: 123

===== RESULTADO LSB =====
```

	Propriedade	Original	LSB
0	Mensagem	Olá mundo!	Olá mundo!
1	Mensagem binária	-	00110100011001110100100101101101010100010110000...
2	Tempo (s)	0	0.178475
3	Tamanho	1971483	1910981
4	Resolução	736 x 1313	736 x 1313
5	Canais	RGB (3 canais)	RGB (3 canais)
6	Profundidade bits	8 bits por canal	8 bits por canal
7	Estrutura pixels	(1313, 736, 4)	(1313, 736, 4)
8	PSNR	∞	66.410609
9	SSIM	1.0	0.999999
10	Bitrate	0	0.029804

Fonte: Autoria própria

Figura 14 - Resultado visual do LSB por pixels (em verde) modificados.

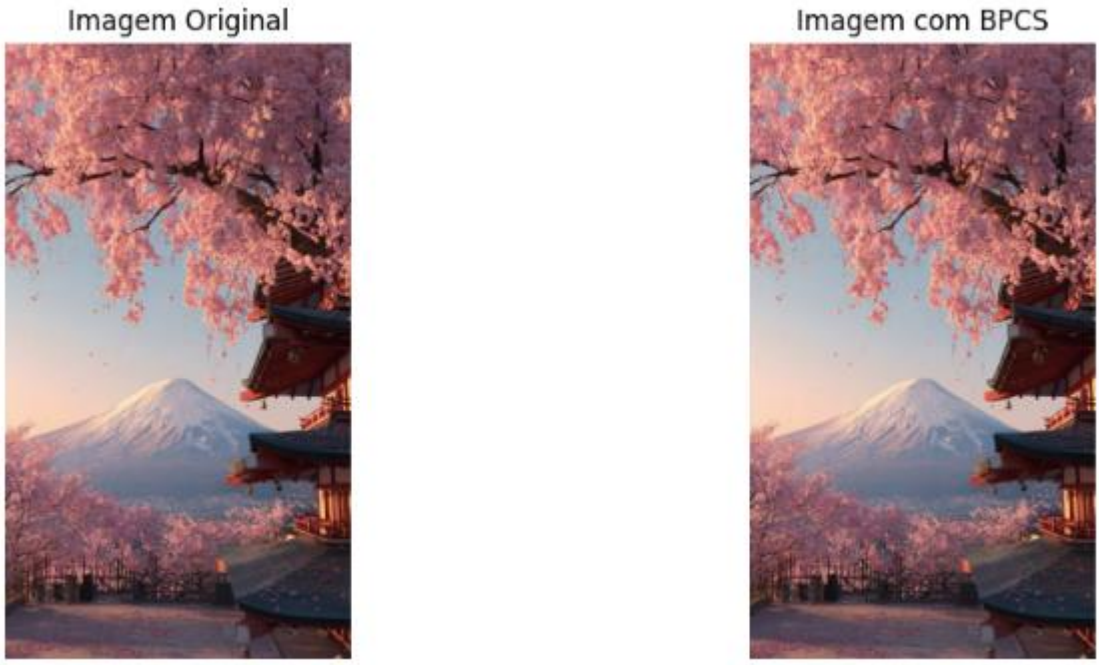
Fonte: Autoria própria

Resultados observados: a imagem modificada manteve características visuais praticamente idênticas à original; o valor de PSNR apresentou níveis elevados, indicando baixa distorção; o SSIM permaneceu próximo de 1, demonstrando alta similaridade estrutural; e o *Bitrate* foi proporcional à quantidade de bits inseridos.

5.3 Testes com BPCS

No método BPCS, a inserção foi realizada em blocos 8×8 dentro dos planos de bits da imagem. Para intensificar a análise e produzir diferenças visualmente perceptíveis nos testes comparativos, a mensagem criptografada foi repetida 600 vezes, aumentando significativamente a quantidade de bits alterados.

Figura 15 - BPCS resolução: original (736×1313 pixels) vs modificada.



Fonte: Autoria própria

Figura 16 - Resultados do BPCS por resolução.

```
===== BPCS =====
Digite a mensagem para BPCS: Olá mundo!
Digite a senha para BPCS: 12345

===== RESULTADO BPCS =====
```

	Propriedade	Original	BPCS
0	Mensagem	Olá mundo!	Olá mundo!
1	Mensagem binária	-	0101001001010010011100010111100100111000010011...
2	Tempo (s)	0	0.146688
3	Tamanho	1971483	1908824
4	Resolução	736 x 1313	736 x 1313
5	Canais	RGB (3 canais)	RGB (3 canais)
6	Profundidade bits	8 bits por canal	8 bits por canal
7	Estrutura pixels	(1313, 736, 4)	(1313, 736, 4)
8	PSNR	∞	66.376643
9	SSIM	1.0	0.999999
10	Bitrate	0	0.029804

Fonte: Autoria própria

Figura 17 - Resultado visual do BPCS por pixels (em verde) modificados.

Fonte: Autoria própria

Resultados observados: houve maior quantidade de alterações estruturais quando comparado ao LSB; o PSNR apresentou pequena redução; o SSIM permaneceu alto; e o *Bitrate* foi equivalente ao LSB para o mesmo volume de dados.

Além da análise individual de cada técnica, foi realizada uma avaliação comparativa considerando o impacto da resolução da imagem portadora sobre o desempenho de ambos os métodos, conforme apresentado na seção a seguir.

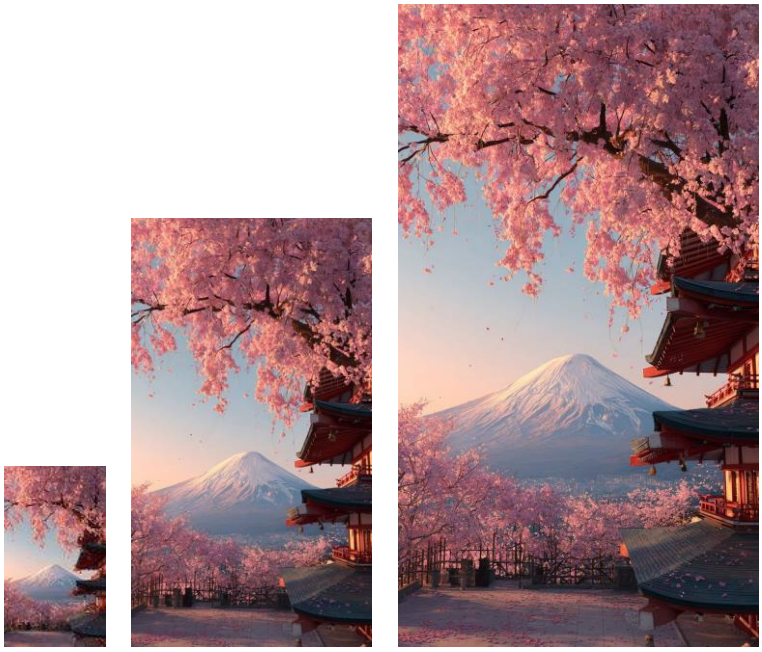
5.4 Comparação entre LSB e BPCS

Esta seção apresenta os resultados dos testes realizados com a imagem "Cherry" nas três resoluções distintas, analisando como a variação de resolução influencia o desempenho das técnicas LSB e BPCS nas métricas de qualidade.

5.4.1 Imagens Utilizadas

A figura a seguir apresenta as três versões da imagem "Cherry" utilizadas nos testes comparativos, evidenciando a diferença de detalhamento visual entre as resoluções. Para intensificar a análise e produzir diferenças visualmente perceptíveis nos testes comparativos, a mensagem criptografada, que nesse caso foi utilizada a mensagem “Olá mundo!”, foi repetida 60 vezes, aumentando significativamente a quantidade de bits alterados.

Figura 18 - Imagens Cherry nas três resoluções: baixa (74x131), média (368x656) e alta (1472x2626).



Fonte: Autoria própria

5.4.2 Resultados por Resolução — Método LSB

Os testes LSB nas três resoluções produziram os seguintes resultados:

Tabela 2 - Resultados LSB por resolução.

Resolução	Pixels	Tam. Orig.	Tam. Mod.	Tempo (s)	PSNR (dB)	SSIM	Bitrate
Baixa	74×131	21.229 B	21.256 B	0.0075	55,1021	0,9999	0,3963
Média	368×656	481.472 B	481.479 B	0,0213	69,0967	0,9999	0,0159
Alta	1472×2626	5.089.053 B	5.089.128 B	0,1795	81,1225	1,0000	0,0009

Fonte: Autoria própria

O PSNR aumenta progressivamente com a resolução da imagem: de 55,10 dB na resolução baixa para 81,12 dB na alta. Isso ocorre porque, embora a mesma quantidade de bits seja inserida (mensagem repetida 60 vezes), a proporção de pixels alterados diminui com o aumento da resolução, resultando em menor distorção média. O SSIM permaneceu próximo de 1 em todas as resoluções, confirmando alta preservação estrutural independentemente do tamanho da imagem.

Figura 19 - LSB resolução baixa: original vs modificada (74x131).

Fonte: Autoria própria

Figura 20 - LSB resolução média: original vs modificada (368x656).

Fonte: Autoria própria

Figura 21 - LSB resolução alta: original vs modificada (1472x2626).

Fonte: Autoria própria

5.4.3 Resultados por Resolução — Método BPCS

Os testes BPCS produziram resultados análogos ao LSB, com pequenas diferenças nas métricas conforme detalhado na Tabela 3:

Tabela 3 - Resultados BPCS por resolução de imagem.

Resolução	Pixels	Tam. Orig.	Tam. Mod.	Tempo (s)	PSNR (dB)	SSIM	Bitrate
Baixa	74×131	21.229 B	21.288 B	0,0106	55,2006	0,9999	0,3963
Média	368×656	481.472 B	481.469 B	0,0649	69,0520	0,9999	0,0159
Alta	1472×2626	5.089.053 B	5.089.404 B	0,4180	81,1435	1,0000	0,0009

Fonte: Autoria própria

Assim como foi apresentado anteriormente na tabela dos resultados do LSB, observa-se que nos resultados do BPCS o PSNR também aumenta progressivamente com a resolução da imagem: de 55,20 dB na resolução baixa para 81,14 dB na alta. Isso ocorre porque, embora a mesma quantidade de bits seja inserida (mensagem repetida 60 vezes), a proporção de pixels alterados diminui com o aumento da resolução, resultando em menor distorção média. O SSIM permaneceu próximo de 1 em todas as resoluções, confirmando também uma alta preservação estrutural independentemente do tamanho da imagem.

Um ponto a ser destacado, seria que em imagens de baixa resolução, o BPCS apresentou PSNR ligeiramente superior ao LSB (55,10 dB vs. 55,20 dB), indicando que a organização em blocos 8×8 distribui as alterações de forma marginalmente menos distorcida.

Figura 22 - BPCS resolução baixa: original vs modificada (74x131).

Fonte: Autoria própria

Figura 23 - BPCS resolução média: original vs modificada (368x656).

Fonte: Autoria própria

Figura 24 - BPCS resolução alta: original vs modificada (1472x2626).

Fonte: Autoria própria

5.4.4 Análise Consolidada por Resolução

Os resultados confirmam que, em todas as resoluções testadas, ambas as técnicas mantiveram qualidade visual satisfatória (PSNR > 55 dB e SSIM > 0,9999). Os principais achados foram:

- Quanto maior a resolução, maior o PSNR e menor o impacto visual das alterações sobre a imagem portadora;
- Na resolução baixa (74×131), o *bitrate* indica que a maior parte dos pixels foram alterados para acomodar a mensagem;
- Na resolução alta (1472×2626), o *bitrate* indica que apenas cerca de 1% dos pixels foram modificados;
- Ambas as técnicas validaram com sucesso a dupla camada de segurança (criptografia + ocultação) em todas as resoluções testadas.

5.5 Comparação entre LSB e BPCS

Tabela 4 - Comparação geral entre LSB e BPCS.

Critério	LSB	BPCS
Complexidade de implementação	Baixa	Média
Tempo de processamento	Menor (≈2× mais rápido)	Maior
Capacidade de inserção	Moderada	Alta
PSNR (dB)	Equivalente (diferença < 0,1 dB)	Equivalente (diferença < 0,1 dB)
SSIM	Idêntico (0,9999 vs 1,0000)	Idêntico (0,9999 vs 1,0000)
Alteração no tamanho do arquivo	Mínima (< 0,01%)	Ligeiramente maior
Robustez a ataques	Menor	Maior
Adequação para alta resolução	Boa	Melhor

Fonte: Autoria própria

Os resultados experimentais demonstram que ambos os métodos atingem qualidade visual praticamente indistinguível, com valores de PSNR e SSIM equivalentes em todas as resoluções testadas. O método LSB destacou-se pela menor complexidade de implementação e pelo tempo de processamento aproximadamente duas vezes inferior ao do BPCS, sendo mais adequado para aplicações com restrições de desempenho ou onde a simplicidade operacional é prioritária. O método BPCS, por sua vez, mostrou-se superior em capacidade de inserção de dados e robustez a ataques, além de apresentar melhor escalabilidade em imagens de alta resolução, tornando-se mais indicado para cenários que exigem maior volume de dados ocultos ou maior segurança contra detecção.

CAPÍTULO VI – CONCLUSÃO

6.1 Considerações Finais

Este trabalho teve como objetivo projetar, desenvolver e analisar um sistema de esteganografia digital integrado com criptografia AES, permitindo a ocultação segura de mensagens em imagens digitais. Foram implementadas duas técnicas distintas — LSB e BPCS — possibilitando comparação prática entre elas.

O sistema foi projetado e desenvolvido em Python, estruturado em módulos, incorporando métricas quantitativas como PSNR, SSIM e Bitrate, além de análises visuais por meio de imagens destacadas. A combinação entre esteganografia e criptografia proporcionou uma abordagem de dupla camada de segurança, reforçando a confidencialidade da informação.

Com base nos testes realizados, todos os objetivos foram atingidos. As mensagens foram corretamente recuperadas quando utilizada a senha adequada, e as métricas demonstraram que as imagens mantiveram alta qualidade visual após a inserção dos dados.

Durante o desenvolvimento, foi possível aprofundar conhecimentos em: manipulação de bits; processamento digital de imagens; programação Python; e métricas de qualidade de imagem. Além do aprendizado técnico, o projeto contribuiu para o desenvolvimento de habilidades analíticas e interpretação de resultados quantitativos.

6.2 Trabalhos Futuros

Segue a sugestão de algumas melhorias que podem ser exploradas futuramente:

- Implementação de assinatura digital para garantir integridade e autoria da mensagem extraída;
- Interface gráfica para facilitar o uso;
- Implementação de técnicas mais avançadas de esteganografia;
- Avaliação contra ataques de esteganálise;
- Inserção adaptativa baseada em regiões de alta complexidade;
- Adoção de protocolos padronizados de comunicação segura (como TLS/SSL) para transmissão das imagens esteganografadas.

CAPÍTULO VIII – REFERENCIAL TEÓRICO

ALMEIDA, Rafael José de Alencar. **Esteganografia e esteganálise: transmissão e detecção de informações ocultas em imagens digitais**. Viva o Linux, 30 ago. 2011. Disponível em: https://www.vivaolinux.com.br/artigo/Esteganografia-e-Esteganalise-transmissao-e-deteccao-de-informacoes-ocultas-em-imagens-digitais/. Acesso em: 6 out. 2025.

ALVES, J. et al. **Técnicas de Esteganografia em Imagens Digitais: Uma Abordagem Prática**. Revista Brasileira de Computação Aplicada, v. 12, n. 3, p. 45-60, 2020.

CHAN, Chi-Kwong; CHENG, L. M. **Hiding data in images by simple LSB substitution**. Pattern Recognition, v. 37, n. 3, p. 469–474, 2004. Disponível em: https://www.academia.edu/3546410/Hiding_data_in_images_by_simple_LSB_substitution. Acesso em: 27 out. 2025.

CMASPI. **LSB Steganography Python tutorial**. YouTube, 10 maio 2022. Disponível em: https://youtu.be/KolfDUoyT3I. Acesso em: 5 nov. 2025.

FRIDRICH, Jessica; GOLJAN, Miroslav; DU, Rui. **Reliable detection of LSB steganography in color and grayscale images**. In: Proceedings of the ACM Workshop on Multimedia and Security, 2001, Ottawa. p. 27–30. DOI: https://doi.org/10.1145/1232454.1232466. Acesso em: 17 fev. 2026.

HOUSLEY, Russ. **Cryptographic Message Syntax (CMS).** RFC 5652. Internet Engineering Task Force, setembro 2009. Disponível em: https://www.rfc-editor.org/rfc/rfc5652. Acesso em: 14 mar. 2026.

KAWAGUCHI, Eiji. **Princípio da esteganografia BPCS**. Kitakyushu: KIT-STEGROUP, [2023]. Disponível em: https://datahide.org/BPCSe/principle-e.html. Acesso em: 1 out. 2025.

KESSLER, G. C.; HOSMER, C. **An overview of steganography**. In: ZELKOWITZ, M. (ed.). Advances in Computers: Security on the Web. 1. ed. Cambridge: Academic Press, 2011. v. 83, p. 51-107.

KHAIRE, Shrikant S.; NALBALWAR, Sanjay L. **Review: steganography — Bit Plane Complexity Segmentation (BPCS) technique**. International Journal of Engineering Science and Technology, v. 2, n. 9, p. 4860–4868, 2010. Disponível em: https://www.researchgate.net/publication/50346770_Review_Steganography_-_Bit_Plane_Complexity_Segmentation_BPCS_Technique. Acesso em: 15 dez. 2025.

KHEDDAR, Hamza; HEMIS, Mustapha; HIMEUR, Yassine; MEGÍAS, David; AMIRA, Abbes. **Deep learning for steganalysis of diverse data types: A review of methods, taxonomy, challenges and future directions**. Neurocomputing, v. 581, p. 127528, 2024. DOI: https://doi.org/10.1016/j.neucom.2024.127528

MORKEL, Tayana; ELOFF, Jan H. P.; OLIVIER, Martin S. **An overview of image steganography**. In: Proceedings of the Fifth Annual Information Security South Africa Conference (ISSA 2005), Sandton, South Africa, junho 2005. Disponível em: https://digifors.cs.up.ac.za/issa/2005/Proceedings/Full/098_Article.pdf. Acesso em: 9 dez. 2025.

NIST — NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. **Advanced Encryption Standard (AES)**. FIPS Publication 197. Gaithersburg: National Institute of Standards and Technology, 2001. Disponível em: https://doi.org/10.6028/NIST.FIPS.197. Acesso em: 7 mar. 2026.

NIST — NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY. **Secure Hash Standard (SHS)**. FIPS Publication 180-4. Gaithersburg: NIST, 2015. Disponível em: https://doi.org/10.6028/NIST.FIPS.180-4. Acesso em: 19 abr. 2026.

PINTEREST, **Aesthetic Japan.** Foto utilizada na pesquisa. Disponível em: https://br.pinterest.com/pin/1055742337661361836/. Acesso em: 1 set. 2025.

PRANJAL KALAL. **Python Project : Image Steganography using LSB method**. YouTube, 2022. Disponível em: https://youtu.be/JeV-WKK1A9Y. Acesso em: 6 nov. 2025.

SION, R. **Steganography**. In: LIU, L.; ÖZSU, M. T. (ed.). Encyclopedia of Database Systems. 2. ed. New York: Springer, 2018.

STUDY WITH DR. DAFTA. **Bit-Plane Coding in Digital Image Processing and its implementation in MATLAB || Compression**. YouTube, 2023. Disponível em: https://youtu.be/y_U2rSKr75g. Acesso em: 6 nov. 2025.

TECHNO FREAK. **Hide Secret message in Images using Python | Image Steganography | Python**. YouTube, 2022. Disponível em: https://youtu.be/qkeLlK4_Vj4. Acesso em: 5 nov. 2025.

ZOCHIO, Marcelo Ferreira. **Introdução à Criptografia: Uma abordagem prática**. Novatec Editora Ltda, 2016.