Trabalho de Processamento de Imagem

Pedro Henrique do Nascimento Correa
Juliana Nogueira Peixoto Minerva Nascimento de Jesus

November 13, 2024

Abstract

Este relatório demonstra a implementação de uma aplicação em Python para carregar e processar imagens, com o foco na aplicação de diferentes filtros e operações morfológicas. A ferramenta permite aos usuários carregar uma imagem e aplicar filtros passa-baixa, como o filtro Gaussiano e o filtro de média, e filtros passa-alta, como o filtro Laplaciano e de Sobel, para destacar bordas e detalhes. Além disso, foram implementadas operações morfológicas como dilatação, erosão, fechamento e abertura, bem como técnicas de segmentação via limiar global e métodos Otsu. O aplicativo permite uma interface interativa que permite aos usuários visualizar os efeitos de cada operação em tempo real.

1 Introdução

O objetivo desse projeto é criar uma aplicação em Python que permita ao usuário explorar diversas técnicas de processamento de imagens por meio de uma interface gráfica intuitiva. As técnicas implementadas são amplamente utilizadas em processamento de imagens, seja para melhorar a visualização, extrair ou ocultar informações. Esta ferramenta foi desenvolvida com o objetivo de fornecer uma base prática e visual dos efeitos e utilidades desses filtros e operações em imagens digitais.

O processamento de imagens é uma área fundamental na computação visual e em diversas aplicações práticas, como reconhecimento de padrões e análise de imagens. Neste projeto, foram desenvolvidos filtros passa-baixa, que suaviam a imagem, e dos filtros passa-alta, que realçam detalhes e bordas.

As operações morfológicas, como a dilatação e erosão, permitem manipular as formas dentro da image, enquanto as técnicas de segmentação destacam áreas de interesse para análise mais aprofundada.

2 Técnicas Utilizadas

No desenvolvimento deste projeto, implementamos uma série de técnicas de processamento de imagens, classificadas em filtros de passa-baixa, filtros de passa-alta, operações morfológicas e métodos de segmentação. Abaixo estão descritas as técnicas, junto com suas representações matemáticas e pontos de amostra de imagens processadas.

2.1 Filtros de Passa-Baixa

Os filtros de passa-baixa suavizam a imagem, reduzindo detalhes e ruídos ao eliminar altas frequências e preservar áreas homogêneas.

Filtro de Média Este filtro suaviza a imagem ao calcular a média dos valores dos pixels em uma janela de tamanho específico. Com isso, os pixels não significativos são removidos e a imagem fica com um efeito mais suave:

$$I'(x,y) = \frac{1}{m \times n} \sum_{i=-a}^{a} \sum_{j=-b}^{b} I(x+i, y+j) \tag{1}$$

onde $I'(x,y)$ é o valor suavizado do pixel, $m \times n$ é o tamanho do kernel, e $I(x+i, y+j)$ são os pixels vizinhos. A aplicação dessa fórmula, de forma visual, pode ser vista na Figura 1.

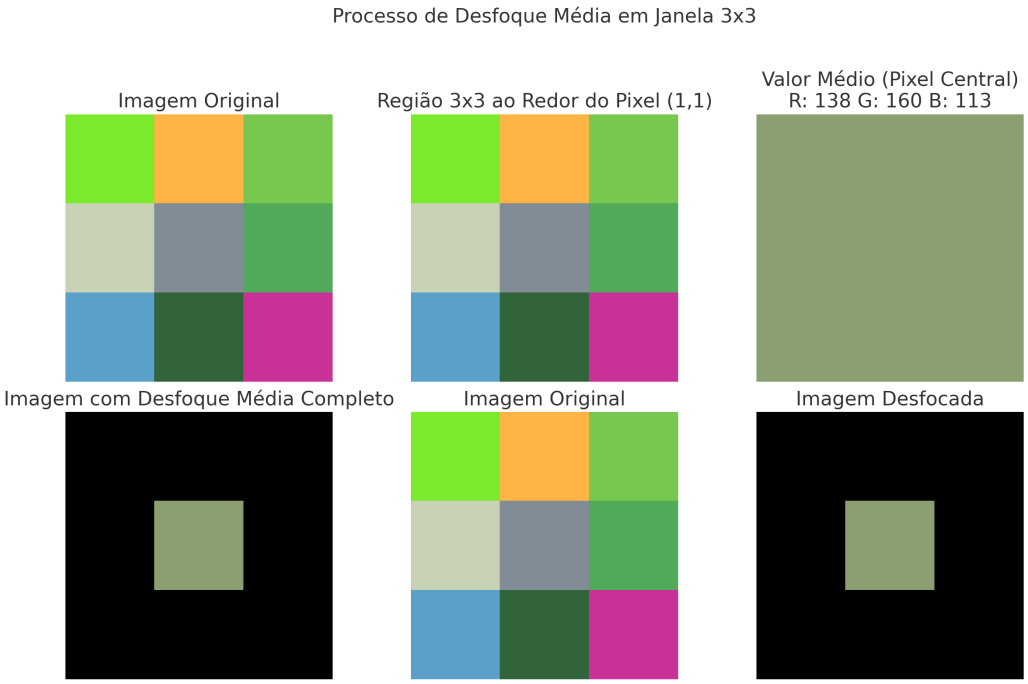

Figure 1: Imagem com Filtro de Média aplicado.

Filtro Gaussiano O filtro Gaussiano aplica uma suavização controlada usando uma função Gaussiana para pesar os pixels vizinhos, com os pixels vizinhos tendo um peso maior:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2}$$

onde $G(x, y)$ é o valor do filtro no ponto (x, y), e σ controla o grau de desfoque. A aplicação dessa fórmula, de forma visual, pode ser vista na Figura 2.

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

Figure 2: Kernel gausiano com os pesos dos pixels.

2.2 Filtros de Passa-Alta

Os filtros de passa-alta realçam detalhes e bordas na imagem, eliminando baixas frequências e preservando altas frequências. Os exemplos dessa seção irão usar uma imagem simples de um martelo (Figura 3) para mostrar com clareza o objetivo de cada filtro.

Figure 3: Imagem de um martelo.

Filtro Laplaciano Este filtro realça bordas usando a segunda derivada da intensidade do pixel:

$$\nabla^2 I(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{3}$$

Ao aplicar esse filtro na imagem do martelo (Figura 3), temos o seguinte resultado:

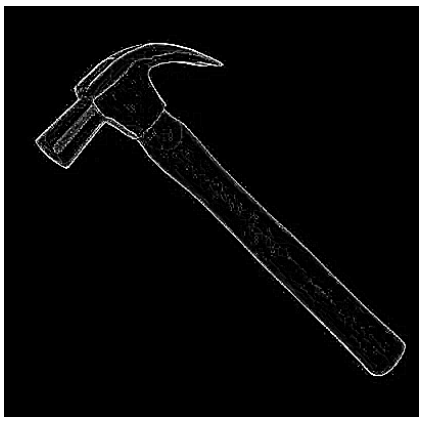

Figure 4: Imagem com Filtro Laplaciano aplicado.

Filtro Sobel O filtro Sobel calcula a derivada da intensidade do pixel nas direções horizontal (S_x) e vertical (S_y):

$$S_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix}, \quad S_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{bmatrix} \tag{4}$$

A magnitude do gradiente é dada por:

$$G = \sqrt{(G_x)^2 + (G_y)^2} \tag{5}$$

Ao aplicar esse filtro na imagem do martelo (Figura 3), temos o seguinte resultado:

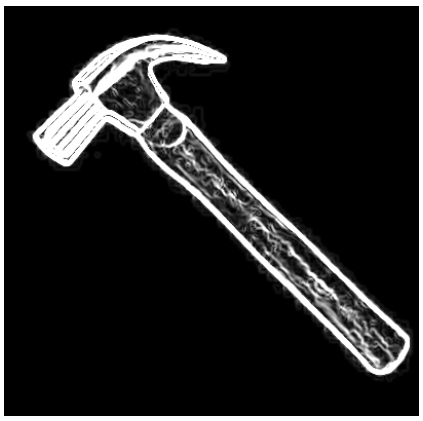

Figure 5: Imagem com Filtro Sobel aplicado.

2.3 Operações Morfológicas

As operações morfológicas manipulam a estrutura dos objetos em imagens binárias. As operações implementadas incluem dilatação, erosão, fechamento e abertura.

Dilatação A dilatação expande regiões brancas da imagem, aumentando os objetos e preenchendo pequenos buracos da imagem.

Erosão A erosão reduz as regiões brancas da imagem, encolhe os objetos e elimina pequenos detalhes.

Fechamento O fechamento é a operação de dilatação seguida da operação de erosão. Com isso, é possível preencher pequenas lacunas ou eliminar buracos em objetos sem perder de forma significativa sua forma.

Abertura A abertura é a operação de erosão seguida da operação de dilatação. Com isso, é possível suavizar as bordas em objetos sem perder de forma significativa sua forma.

2.4 Segmentação

A segmentação separa áreas de interesse na imagem. Foram utilizadas duas técnicas:

Threshold Global Define um valor de threshold global para a binarização de forma iterativa, com base na média das regiões de pixels acima e abaixo do threshold atual.

$$T = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} I(i,j) \tag{6}$$

Threshold de Otsu Este método implementa o método de Otsu, que calcula de forma automatica um valor de limiar ótimo paa separar a imagem em duas classes. Com isso, é possível minimizar a variância intra-classe e maximizar a variância entre classes.

3 Sistema

O sistema foi desenvolvido em Python, utilizando a biblioteca CustomTkinter para criar uma interface gráfica de usuário (GUI). A interface permite o carregamento de imagens, aplicação de filtros e a configuração da intensidade dos mesmos. Além disso, é utilizado o Numpy para facilitar os calculos a aplicação das fórmulas nas imagens.

4 Conclusão

O sistema fornece uma interface intuitiva para processamento de imagens, permitindo aplicar filtros e operações morfolóficas de foma simples e cus-

tomizavel. A implementação em Python permite uma base prática, de fácil entendimento e com baixa curva de aprendizado.