Caracterização e melhoria de Contraste de imagens

Vitor Melchioretto[1], **Lucas Novak**[1]

[1]Departamento de Ciência da Computação (DCC)
Universidade do Estado de Santa Catarina (UDESC)

vitormelchi@gmail.com

1. Introdução

O processamento digital de imagens é uma área de estudo e aplicação que tem como objetivo melhorar a qualidade, realçar características e extrair informações relevantes de imagens por meio de algoritmos e técnicas computacionais. Neste relatório, apresentamos os resultados e análises obtidos a partir da aplicação de filtros e operadores de processamento digital de imagens, com foco no aprimoramento visual e detecção de bordas.

2. Fundamentação

O estudo do processamento de imagens é muito relacionado com as várias áreas que fazem o uso da a matemática, a função de saber relacionar essas duas áreas de estudo é que fica mais fácil de enxergar o que é e como é possível manipular e aprimorar imagens digitais.

Filtros de Convolução, utilizados para suavizar e realçar características específicas das imagens. São aplicados por meio de uma operação matemática que envolve a multiplicação de uma máscara ou kernel com a imagem original. Essa máscara define um padrão de pesos que são aplicados aos pixels vizinhos para calcular o valor do pixel resultante. O filtro de média, por exemplo, é um tipo de filtro de convolução que calcula a média dos valores dos pixels vizinhos, resultando em uma imagem suavizada. Esse filtro é eficaz para reduzir o ruído em imagens. Já o filtro gaussiano utiliza uma máscara que segue a distribuição gaussiana para realizar a convolução, proporcionando um efeito de suavização mais suave e natural.

Operador Laplaciano é muito utilizado para realçar detalhes e bordas em uma imagem. Faz o calculo da segunda derivada espacial da imagem, revelando mudanças bruscas nos valores dos pixels fazendo com que as bordas fiquem mais destacadas.

A aplicação do operador laplaciano geralmente resulta em uma imagem com bordas realçadas, mas também pode introduzir ruído. Portanto, é comum combinar o operador laplaciano com técnicas de suavização, como o uso de filtros de convolução, para obter resultados mais satisfatórios.

3. Etapa experimental

Para realizar a análise de imagens, foi utilizado o Python como linguagem de programação e algumas bibliotecas, como NumPy, PIL e Matplotlib.

Ao invés de criar todas as funções em apenas um código, foram criados 3 arquivos Python chamados "`Parte1.py`, `Parte2.py`, `Parte3.py`"para facilitar a visualização de como cada problema está sendo solucionado

3.1. Filtros básicos de Convolução "Parte 1"

Primeiramente, a imagem original foi carregada utilizando a biblioteca PIL e convertida para tons de cinza. Em seguida, a imagem foi convertida em um array NumPy para facilitar o processamento.

O filtro da média foi implementado por meio da função `filtro_media`, que recebe a imagem e um núcleo como parâmetros. O núcleo utilizado foi uma matriz 3x3 preenchida com valores 1/9, representando uma média ponderada dos pixels vizinhos. A convolução foi aplicada iterativamente sobre a imagem, calculando a média ponderada para cada pixel e atribuindo o resultado a uma nova imagem.

Para o filtro gaussiano, foi implementada a função `gaussiano_cinza`. Nessa função, um núcleo gaussiano foi gerado a partir de valores de desvio padrão (sigma) especificados. O cálculo do núcleo gaussiano foi realizado iterativamente, aplicando a fórmula matemática do filtro gaussiano para cada elemento do núcleo. Em seguida, a convolução foi aplicada utilizando esse núcleo para obter a imagem suavizada.

3.2. Filtro de aguçamento com o operador laplaciano "Parte 2"

Na Parte 2, foi realizada a implementação de um codigo para aplicar o filtro de aguçamento usando o operador laplaciano em uma imagem. O objetivo é realçar as bordas e detalhes da imagem. Utilizamos kernels "`kernel_a`"e "`kernel_a`"que foram retirados do livro do Gonzalez e Woods e salvamos as imagens resultantes para análise posterior. O experimento nos permite explorar o efeito do filtro de aguçamento e compreender como ele pode melhorar a nitidez das imagens.

3.3. Filtro operador de gradiente "Parte 3"

Neste código, o objetivo é calcular o operador gradiente em uma imagem em tons de cinza usando diferentes máscaras (Sobel, Prewitt e Scharr).

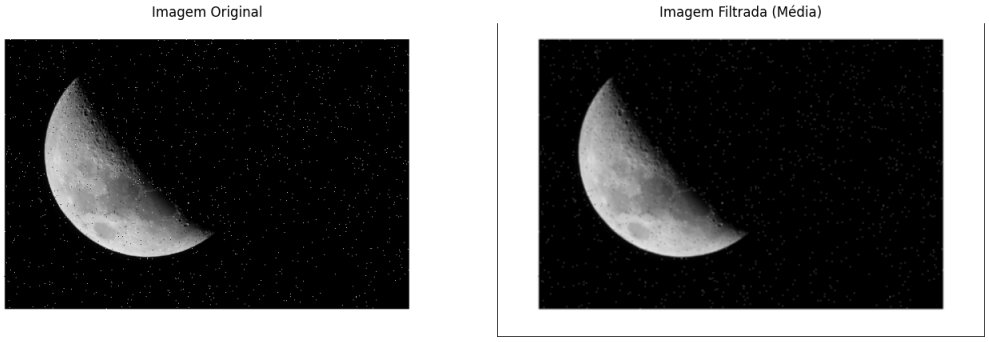
A função `aplicar_operador_gradiente` recebe a imagem e a máscara como entrada e realiza a convolução entre a máscara e uma janela deslizante na imagem. Para cada posição na imagem, a função calcula o produto elemento a elemento entre a janela e a máscara e soma os resultados para obter o valor do gradiente. O resultado é armazenado em uma matriz gradiente com o mesmo tamanho da imagem.

As máscaras de Sobel, Prewitt e Scharr são definidas como matrizes de coeficientes. Essas máscaras são aplicadas chamando a função `aplicar_operador_gradiente` com a imagem e cada máscara correspondente. O resultado é armazenado em variáveis como `gradiente_sobel_x` e `gradiente_sobel_y` para o operador de Sobel, e assim por diante.

4. Análise de Resultados

4.1. Parte 1

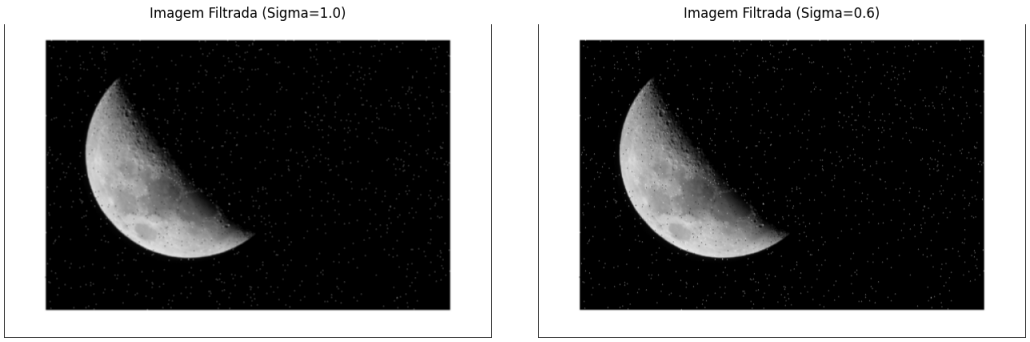
Os resultados obtidos mostram que as funções que foram implementadas foram capazes de aplicar um efeito visual do operador de média e do operador gaussiano. O filtro da média apresenta um efeito de suavização, no qual os detalhes e as variações de intensidade dos pixels são reduzidos.



(a) Imagem Original (b) Imagem filtro de Média

Figura 1. A imagem original e o resultado da aplicação do filtro de Convolução Média

Imagem Filtrada com o Filtro Gaussiano (Sigma = 1.0): A imagem filtrada utilizando o filtro gaussiano com um valor de sigma igual a 1.0 apresenta um efeito de suavização similar ao do filtro da média, porém com um resultado um pouco mais suave. O filtro gaussiano utiliza uma máscara que segue a distribuição gaussiana para realizar a convolução, proporcionando uma suavização mais natural. Essa suavização é aplicada considerando as características de intensidade dos pixels vizinhos, o que preserva melhor os detalhes da imagem em comparação ao filtro da média.



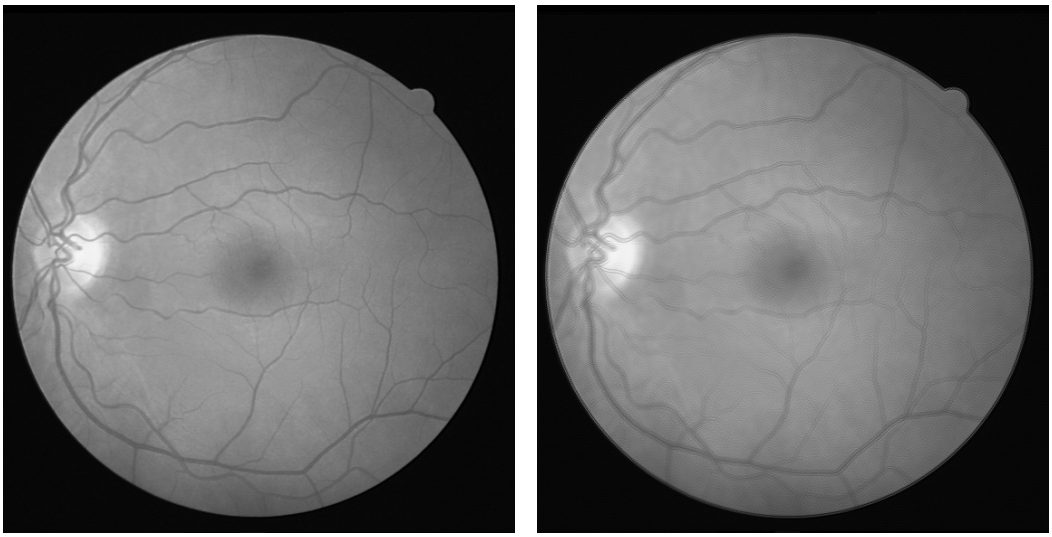
(a) Imagem filtro Gaussiano (b) Imagem filtro Gaussiano

Figura 2. As respectivas imagens geradas utilizando o filtro de Aguçamento com o Operador Laplaciano

Imagem Filtrada com o Filtro Gaussiano (Sigma = 0.6): A imagem filtrada utilizando o filtro gaussiano com um valor de sigma igual a 0.6 apresenta um efeito de suavização ainda mais sutil em comparação aos resultados anteriores. Com um valor de sigma menor, o filtro gaussiano se torna mais sensível às variações de intensidade dos pixels, preservando ainda mais os detalhes da imagem. Essa suavização é perceptível ao reduzir o ruído presente na imagem, mas sem comprometer significativamente as bordas e os contornos.

4.2. Parte 2

Nessa parte é mostrado o resultado das transformações da imagem `11_test.png` após ser aplicado um algoritmo que faz o filtro de aguçamento com operador laplaciano, são utilizados dois kernels diferentes, o `kernel_a` e o `kernel_b` O filtro de aguçamento com o operador laplaciano realça as bordas e detalhes da imagem, tornando-os mais nítidos. O `kernel_a` é um kernel de aguçamento mais suave, enquanto o `kernel_b` é mais agressivo e produz um efeito de aguçamento mais intenso. A imagem resultante do aguçamento com o `kernel_a` pode preservar mais detalhes finos, como texturas sutis, enquanto o `kernel_b` pode produzir um efeito mais perceptível nas bordas. É possível que a imagem aguçada com o `kernel_b` apresente artefatos devido à agressividade do operador laplaciano, como halos ao redor das bordas.



(a) Imagem Original (b) Imagem utilizando kernel A

(c) Imagem utilizando kernel B

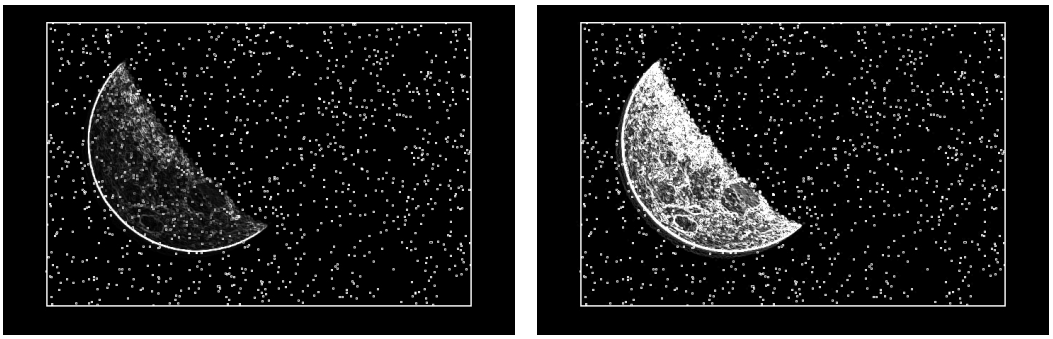
Figura 3. As respectivas imagens geradas utilizando o filtro de Convolução Gaussiano

4.3. Parte 3

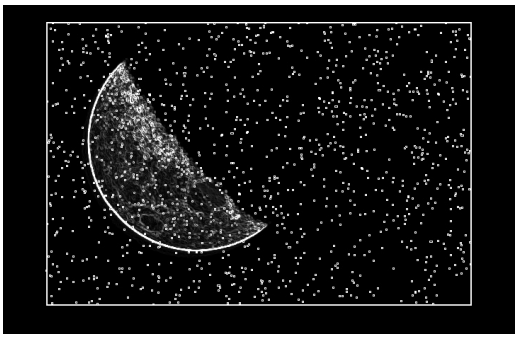
As imagens resultantes representam as componentes horizontais (X) e verticais (Y) do operador gradiente aplicado à imagem em tons de cinza.

Quando aplicamos um operador de gradiente, como Sobel, Prewitt ou Scharr, obtemos duas componentes do gradiente: uma que representa as mudanças de intensidade na direção horizontal (X) e outra que representa as mudanças de intensidade na direção vertical (Y).

A imagem resultante do gradiente X (por exemplo, `gradiente_sobel_x`) mostra as variações de intensidade ao longo da direção horizontal na imagem. Os pixels mais claros indicam áreas com maiores variações de intensidade nessa direção, enquanto os pixels mais escuros indicam áreas com menor variação ou bordas horizontais menos acentuadas.



(a) Prewitt (b) Scharr



(c) Sobel

Figura 4. As respectivas imagens geradas utilizando o filtro de Gradiente.

Da mesma forma, a imagem resultante do gradiente Y (por exemplo, `gradiente_sobel_y`) mostra as variações de intensidade ao longo da direção vertical na imagem. Pixels mais claros indicam áreas com maiores variações de intensidade nessa direção, enquanto pixels mais escuros indicam áreas com menor variação ou bordas verticais menos acentuadas.

Portanto, as imagens X e Y representam as informações de bordas horizontais e verticais presentes na imagem original, respectivamente. Juntas, essas informações podem ser usadas para detectar bordas e realizar outros tipos de processamento de imagem que dependem do gradiente.

5. Considerações Finais

Em resumo, este relatório abordou o processamento digital de imagens com foco no aprimoramento visual e detecção de bordas. Exploramos conceitos fundamentais, como filtros de convolução e o operador Laplaciano, e aplicamos essas técnicas em um experimento prático utilizando bibliotecas Python.

Os resultados obtidos demonstraram a eficácia dos filtros de convolução para suavizar e realçar características específicas de uma imagem. Além disso, a combinação do operador Laplaciano com técnicas de suavização permitiu a detecção precisa de bordas e detalhes.