Engenharia de Computação

Processamento Digital de Imagem

Trabalho 4

Helena Tavares

Junho de 2022

1.

O objetivo é apresentar duas máscaras passa baixa e o resultado após a aplicação na imagem proposta.

O grupo de filtros passa baixa escolhidos foram filtro de Média utilizando uma máscara 3x3 e o filtro Gaussiano, também com máscara 3x3.

1.1 Filtro passa-baixa de média

Multiplica-se os pesos das máscaras pelos pixels da imagem, e divide-se o resultado pelo número total dos pixels da máscara. O valor do pixel central da imagem é substituído pela média dos valores dos pixels vizinhos.

Esse filtro reduz a variação dos níveis de cinza da imagem e suaviza seu contraste. Os pixels de valores mais altos que o valor dos seus vizinhos, são diminuídos e assim os detalhes locais são perdidos.

A multiplicação entre os pesos, é uma imagem levemente desfocada.

Esse filtro é representado, matematicamente, da seguinte forma:

$$g(i,j) = \frac{1}{MN} \sum_{m=1}^{M} \sum_{n=1}^{N} f(m,n)$$

Já que esse filtro reduz as altas frequências de brilho da imagem, ele é usado para remover ruídos e uniformizar os valores de brilho da imagem.

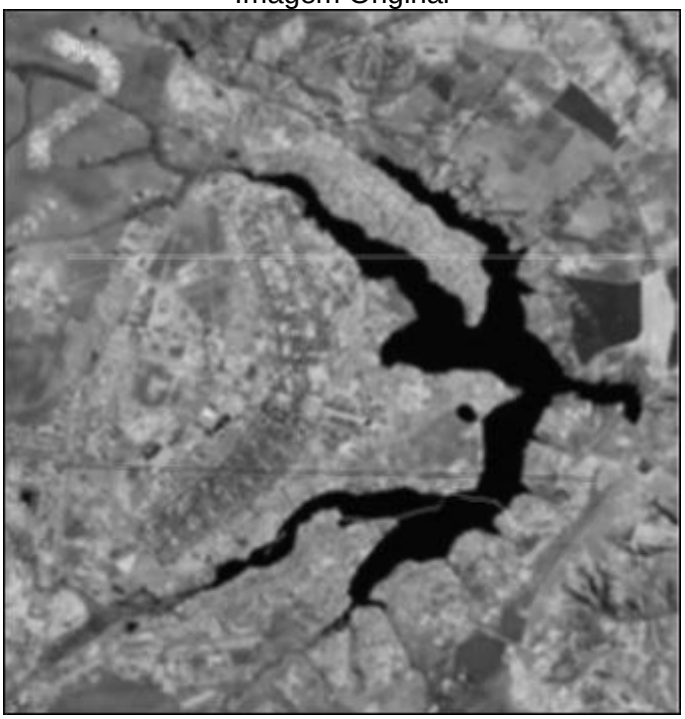
O tamanho da máscara está relacionado ao nível de suavização e a resolução espacial da imagem. Filtros maiores que 9x9 não são tão utilizados.

Abaixo a máscara utilizada nesse filtro, a imagem original e o resultado da imagem respectivamente.

$$Z = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Máscara 3x3 utilizada

Imagem Original



Filtro passa baixa de Média aplicado

1.2. Filtro passa baixa Gaussiano

O filtro Gaussiano, assim como o filtro de Média, é passa baixa e utilizado para reduzir ruídos dos componentes de alta frequência. Permite borrar uma região da imagem, ou ela por completo. É implementado com uma máscara simétrica de tamanho ímpar, e passa por cada pixel da região que interessa realizar o “borramento”.

O filtro de suavização Gaussiano é baseado em uma aproximação digital da função gaussiana. O Filtro Gaussiano em 1D é descrito por:

$$G(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-x^2}{2\sigma^2}}$$

Já o Filtro Gaussiano em 2D é dado por:

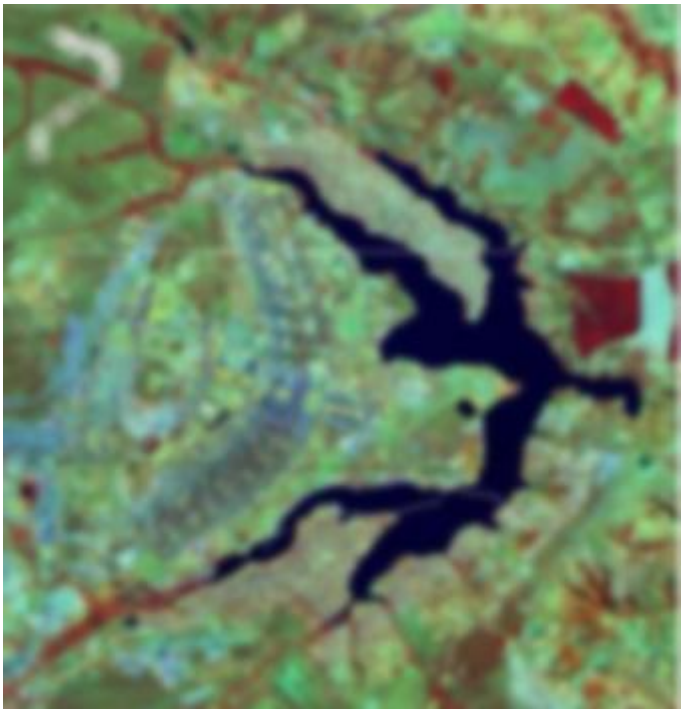
$$G(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

Abaixo a máscara utilizada nesse filtro, a imagem original e o resultado da imagem respectivamente.

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

Máscara 3x3 utilizada

Imagem original



Filtro passa baixa Gaussiano aplicado

O filtro Gaussiano suaviza a imagem de forma muito semelhante ao filtro de média, o resultado será mais suave quanto maior o desvio padrão da Gaussiana utilizada.

2.

O objetivo é apresentar duas máscaras passa alta e o resultado após a aplicação na imagem proposta.

O grupo de filtros passa alta escolhidos foram filtro Laplace utilizando uma máscara 3x3 e o filtro Sobel, também com máscara 3x3.

Filtros passa alta, não alteram os componentes de alta frequência da transformada de Fourrier, já os de baixa frequência são removidos. Esse funcionamento enfatiza os detalhes finos da imagem.

2.1. Filtro passa alta Laplace

O filtro Laplaciano é passa alta, baseado em operador de segunda derivada é usado para identificar bordas nas imagens (áreas de mudança rápida). Esse filtro não necessita processamento individual vertical e horizontal.

A entrada do operador geralmente está no modo escala de cinza.

Abaixo a máscara utilizada nesse filtro, a imagem original e o resultado da imagem respectivamente.

$$Z=\begin{bmatrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{bmatrix}$$

Imagem original

Filtro passa alta Laplace aplicado

2.2. Filtro passa alta Sobel

Assim como o filtro Laplaciano, esse filtro também é passa alta e utilizado para detecção de bordas. Em termos técnicos, consiste num operador que calcula diferenças finitas, dando uma aproximação do gradiente da intensidade dos pixels da imagem.

Utiliza derivada de primeira ordem e necessita um processamento separado horizontal e verticalmente.

Para calcular um Laplaciano, você precisará calcular primeiro duas derivadas, chamadas derivadas de Sobel , cada uma das quais leva em consideração as variações do gradiente em uma determinada direção: uma horizontal, a outra vertical.

Derivada horizontal de Sobel (Sobel x) : É obtida através da convolução da imagem com uma matriz denominada kernel que sempre tem tamanho ímpar. O kernel com tamanho 3 é o caso mais simples.

Derivada vertical de Sobel (Sobel y) : É obtida através da convolução da imagem com uma matriz denominada kernel que sempre tem tamanho ímpar. O kernel com tamanho 3 é o caso mais simples.

A convolução é calculada pelo seguinte método: a imagem representa a matriz da imagem original e o filtro é a matriz do kernel.

$$Z=\begin{bmatrix} 0 & -2 & 0 \\ -2 & 11 & -2 \\ 0 & -2 & 0 \end{bmatrix}$$

Fator = 11 - 2- 2- 2- 2- 2 = 3
Offset = 0
Soma ponderada = 124 * 0 + 19 * (- 2) + 110 * (- 2) + 53 * 11 + 44 * (- 2) + 19 * 0 + 60 * (- 2) + 100 * 0 = 117
O [4,2] = (117/3) + 0 = 39

Portanto, no final, para obter o Laplaciano (aproximação), precisaremos combinar os dois resultados anteriores (Sobelx e Sobely) e armazená-lo no laplaciano

Abaixo a máscara utilizada nesse filtro, a imagem original e o resultado da imagem respectivamente.

$$Z=\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

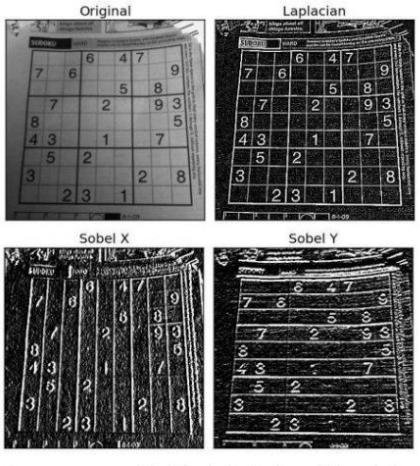
Imagem original

Filtro passa alta Sobel aplicado

A principal diferença entre esses dois filtros é que o Laplaciano não necessita que a imagem seja processada individualmente na horizontal e na vertical. Outra diferença é o Sobel utiliza primeira derivada, e o Laplaciano segunda derivada.

O Sobel funciona melhor para imagens com ruído, ainda que o Laplaciano também seja capaz de detectar as bordas, porém, a clareza da imagem pode ficar comprometida. Na ausência de ruído, o Laplace apresenta resultado muito superior ao Sobel.

Abaixo um exemplo clássico para demonstrar a principal diferença entre os dois:



3.

Aplicação de um filtro de média 5x5 na imagem abaixo:

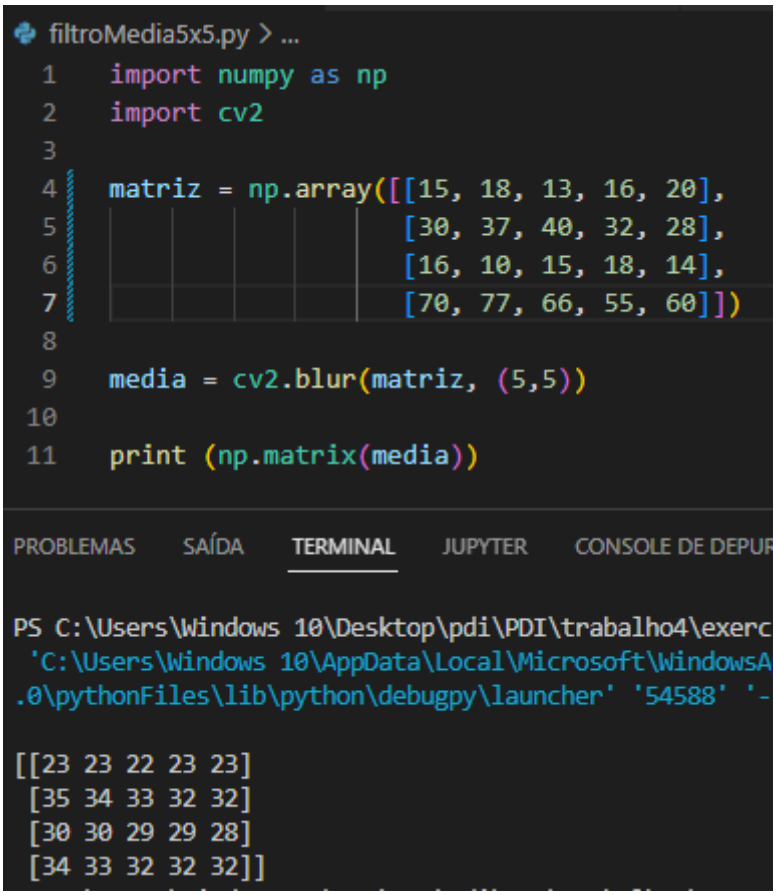
$$X = \begin{matrix} 15 & 18 & 13 & 16 & 20 \\ 30 & 37 & 40 & 32 & 28 \\ 16 & 10 & 15 & 18 & 14 \\ 70 & 77 & 66 & 55 & 60 \end{matrix}$$

Matriz de média 5x5 aplicada:

$$\frac{1}{25}\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

A masca apresentada é referente ao filtro de média quadrada, que é do grupo passa baixa. A matriz é toda preenchida por números um, multiplicada pela base da matriz e dividida pela quantidade de linhas versus colunas da mesma. É o processo de divisão que realiza o efeito de borramento da imagem e varia de acordo com o número de linhas e colunas da máscara.

O resultado foi o seguinte:



Conforme aumenta o tamanho da máscara, maior é o efeito de borramento na imagem.

4. Implementação do filtro Mediana

```python
import cv2
import numpy as np

img = cv2.imread('chile.jpg', 0)
m, n = img.shape

img_new = np.zeros([m, n])

for i in range(1, m-1):
    for j in range(1, n-1):
        temp = [img[i-1, j-1],
                img[i-1, j],
                img[i-1, j + 1],
                img[i, j-1],
                img[i, j],
                img[i, j + 1],
                img[i + 1, j-1],
                img[i + 1, j],
                img[i + 1, j + 1]]
        temp = sorted(temp)
        img_new[i, j]= temp[4]

img_new = img_new.astype(np.uint8)
cv2.imwrite('filtroMediana.jpg', img_new)
```

Filtro Mediana

Este filtro é do grupo passa baixas, porém, é um filtro não linear. Substitui a intensidade de cada pixel pela mediana das intensidades na vizinhança do pixel.

Esse filtro coloca em ordem crescente ou decrescente a intensidade dos pixels dentro da área da máscara e aloca ao pixel da imagem correspondente à posição central da máscara. O valor da intensidade do pixel que corresponde à posição intermediária do respectivo intervalo ordenado.

O filtro de mediana remove ruídos do tipo sal e pimenta e preserva o contorno dos detalhes da imagem de forma mais eficiente que os filtros de média e gaussiano.

5.

Aplicação de um filtro passa alta 3x3 na imagem abaixo:

$$X = \begin{matrix} 15 & 18 & 13 & 16 & 20 \\ 30 & 37 & 40 & 32 & 28 \\ 16 & 10 & 15 & 18 & 14 \\ 70 & 77 & 66 & 55 & 60 \end{matrix}$$

O filtro escolhido para a aplicação foi o laplaciano, que faz parte do grupo passa alta. Esse grupo busca destacar as bordas da imagem e as colocar em evidencia. Filtro Laplaciano é um dos mais simples do grupo.

As máscaras para esse filtro que tem o centro negativo, removem as bordas exteriores e as máscaras com centro positivo, removem as bordas interiores.

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad \left| \frac{1}{8} \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix} \right| \qquad \left| \frac{1}{4} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} \right| \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Como o filtro Laplaciano é linear, existem máscaras que já combinam as duas operações realce + reconstrução do fundo da imagem. São eles:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & -5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

A matriz utilizada foi a padrão da função do Python cv2.laplacian, limitada no tamanho 3x3, conforme pedido no exercício.

O resultado foi o seguinte:

filtroPassaalta3x3.py > ...

```python
import numpy as np
import cv2

matriz = np.array([[15, 18, 13, 16, 20],
                   [30, 37, 40, 32, 28],
                   [16, 10, 15, 18, 14],
                   [70, 77, 66, 55, 60]])

matriz = matriz.astype(np.uint8)

lap = cv2.Laplacian(matriz, cv2.CV_64F,(3,3))
lap = np.uint8(np.absolute(lap))

print (np.matrix(lap))
```

PROBLEMAS SAÍDA TERMINAL JUPYTER CONSOLE DE DEPURAÇÃO

```
PS C:\Users\Windows 10\Desktop\pdi\PDI\trabalho4\exercicios3
 'C:\Users\Windows 10\AppData\Local\Microsoft\WindowsApps\py
.0\pythonFiles\lib\python\debugpy\launcher' '54614' '--' 'c:
.py'
[[ 36  30  62  33   8]
 [ 15  50  63  26  14]
 [ 56 105  74  44  68]
 [ 94 152 102  58 102]]
```