

 <p>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA SÃO PAULO Campus Birigui</p>	<p>INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA Campus Birigui</p> <p>Bacharelado em Engenharia de Computação</p>	
<p>Disciplina: Processamento Digital de Imagens</p>	<p>Filtragem Frequência</p>	
<p>Professor: Prof. Dr. Murilo Vargas da Silva</p>	<p>Data: 18/09/2023</p>	
<p>Nome do aluno: Leonardo Reneres dos Santos</p>	<p>Prontuário: BI3009131</p>	

Questões:

1. Calcule e visualize o espectro de uma imagem 512x512 pixels:

- a) crie e visualize uma imagem simples – quadrado branco sobre fundo preto;
- b) calcular e visualizar seu espectro de Fourier (amplitudes);
- c) calcular e visualizar seu espectro de Fourier (fases);
- d) obter e visualizar seu espectro de Fourier centralizado;
- e) Aplique uma rotação de 40° no quadrado e repita os passo b-d;
- f) Aplique uma translação nos eixos x e y no quadrado e repita os passo b-d;
- g) Aplique um zoom na imagem e repita os passo b-d;
- h) Explique o que acontece com a transformada de Fourier quando é aplicado a rotação, translação e zoom.

2. Crie filtros passa-baixa do tipo ideal, butterworth e gaussiano e aplique-o às imagens disponibilizadas. Visualize o seguinte:

- a) a imagem inicial;
- b) a imagem de cada filtro;
- c) a imagem resultante após aplicação de cada filtro.

3. Crie um filtro passa-alta do tipo ideal, butterworth e gaussiano e aplique-o às imagens disponibilizadas. Visualize os mesmos dados da tarefa anterior:

- a) a imagem inicial;
- b) a imagem de cada filtro;
- c) a imagem resultante após aplicação de cada filtro.

4. Varie o parâmetro de frequência de corte no filtro passa-baixa criado na tarefa 2. Por exemplo, tome valores de D_0 iguais a 0,01, 0,05, 0,5. A imagem inicial é igual à anterior. Visualize as imagens dos filtros e as imagens resultantes. Explique os resultados.

5. Efetue o mesmo que se pede no item 4, mas use o filtro passa-alta em vez do filtro passa-baixa.

6. Além dos filtros passa-baixa e passa-alta também existe o filtro passa-banda? Explique seu funcionamento e aplique um filtro passa-banda na imagem.

Respostas:

1:

Código:

```
import cv2

import numpy as np

import matplotlib.pyplot as plt
```

```
def plotar(im1,n):

    f = np.fft.fft2(im1)

    fshift = np.fft.fftshift(f)

    magnitude_spectrum = 20*np.log(np.abs(fshift))

    mag = 20*np.log(np.abs(f))

    fase = np.angle(f)

    plt.subplot(221),plt.imshow(im1, cmap = 'gray')

    plt.title(n), plt.xticks([]), plt.yticks([])

    plt.subplot(222),plt.imshow(mag, cmap = 'gray')

    plt.title('Magnitude Spectrum não centralizado'), plt.xticks([]),
plt.yticks([])

    plt.subplot(223),plt.imshow(fase, cmap = 'gray')

    plt.title('Fase não centralizado'), plt.xticks([]), plt.yticks([])
```

```
plt.subplot(224),plt.imshow(magnitude_spectrum, cmap = 'gray')

plt.title('Magnitude Spectrum'), plt.xticks([]), plt.yticks([])


plt.show()


def main():

    # Crie uma imagem preta

    imagem = np.zeros((512, 512, 3), dtype = "uint8")

    # Desenhe um quadrado branco

    cv2.rectangle(imagem, (106, 106), (406, 406), (255, 255, 255), -1)

    im = cv2.cvtColor(imagem, cv2.COLOR_BGR2GRAY)

    plotar(im,"Quadrado")

    while(True):

        print("Selecione a opção: 1: rotação, 2= translação, 3=zoom ou
5 = sair")
```

```

m = int(input())

(altura, largura) = im.shape[:2]

if(m == 1 ):

    centro = (largura / 2, altura / 2)

    matriz = cv2.getRotationMatrix2D(centro, 40, 1.0)

    plotar(cv2.warpAffine(im, matriz, (largura,
altura)), "Rotação 40 graus")

elif(m==2):

    desloc_x = largura / 6

    desloc_y = altura / 6

    matriz = np.float32([[1, 0, desloc_x], [0, 1, desloc_y]])

    plotar(cv2.warpAffine(im, matriz, (largura,
altura)), "Translação")

elif(m==3):

    zoom = 1.5

    imzoom= cv2.resize(im, None, fx=zoom, fy=zoom,
interpolation=cv2.INTER_LINEAR)

    plotar(imzoom, "Zoom")

elif(m==5):

    break

else:

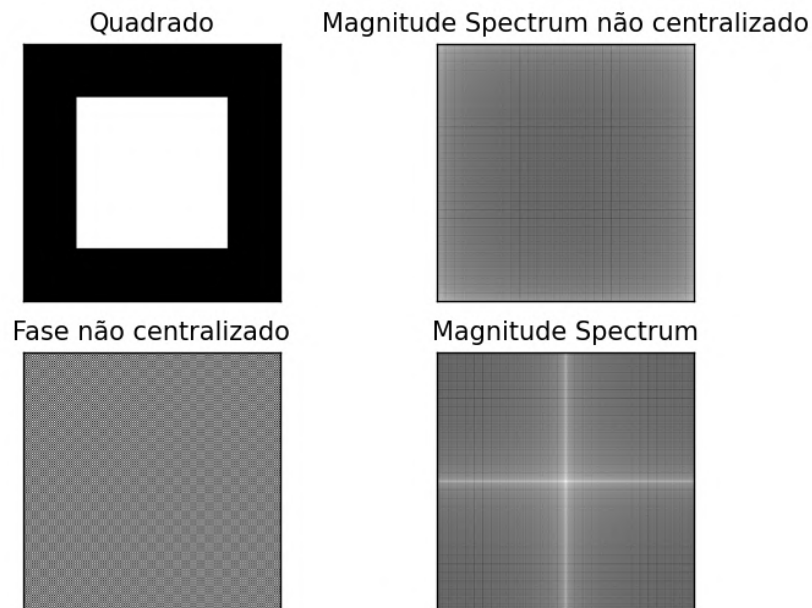
    break

if __name__ == "__main__":

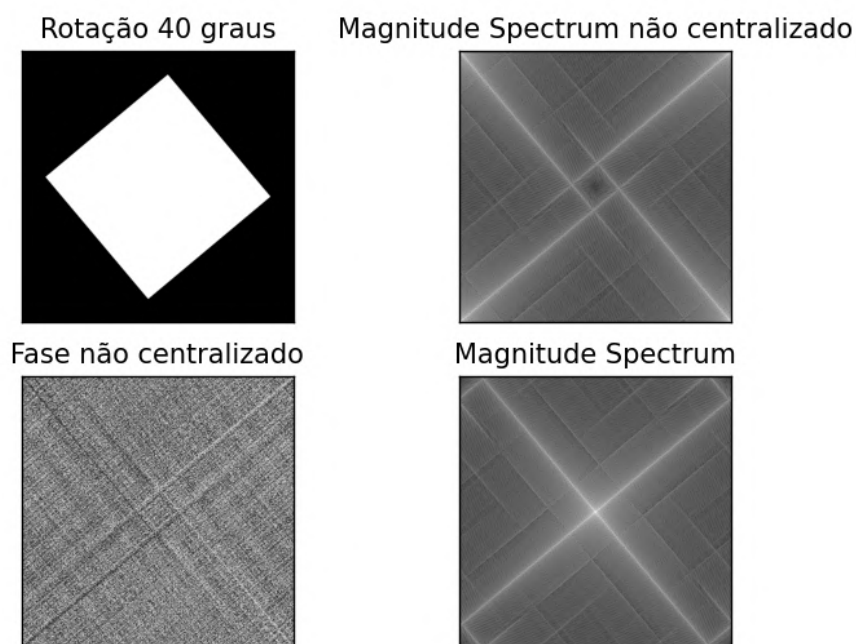
    main()

```

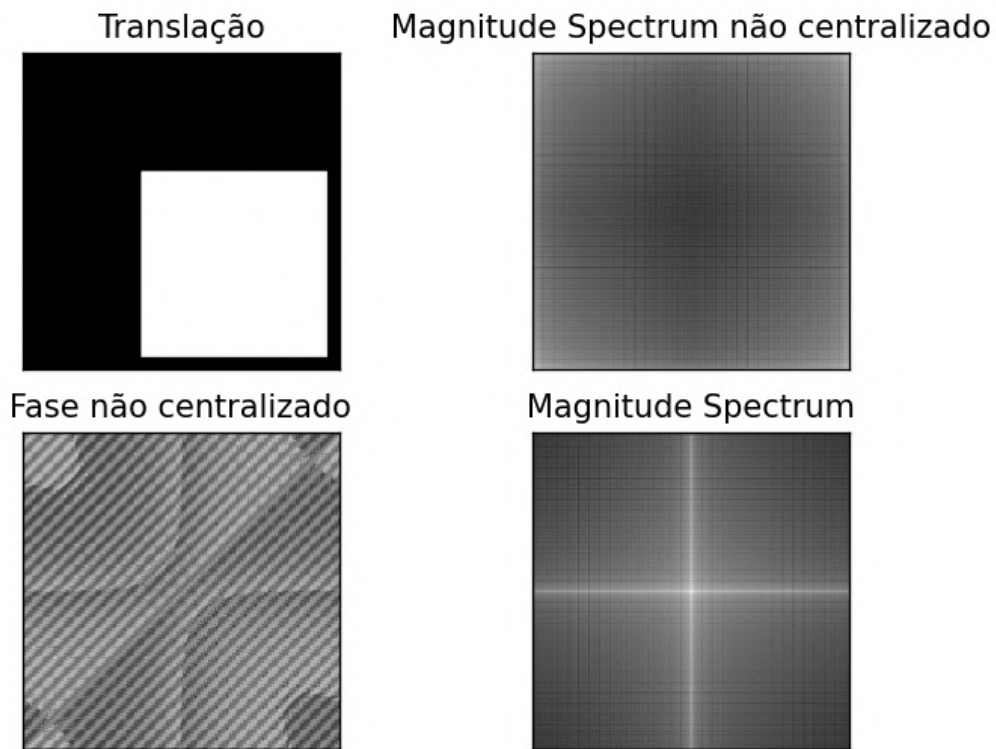
- crie e visualize uma imagem simples – quadrado branco sobre fundo preto;
- calcular e visualizar seu espectro de Fourier (amplitudes);
- calcular e visualizar seu espectro de Fourier (fases);
- obter e visualizar seu espectro de Fourier centralizado;



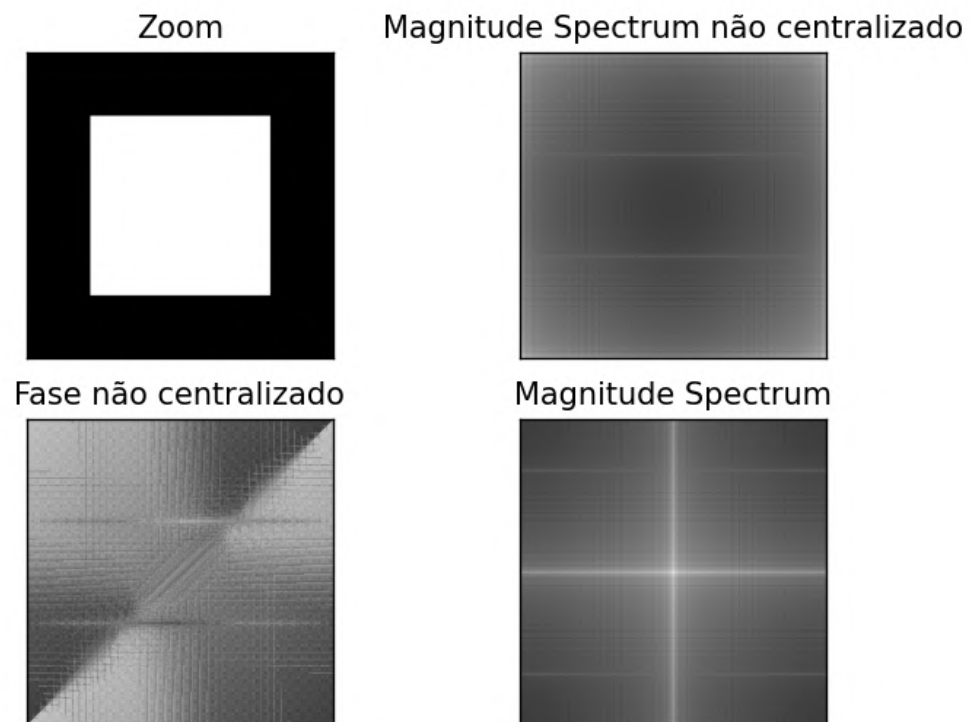
- Aplice uma rotação de 40° no quadrado e repita os passo b-d;



f) Aplique uma translação nos eixos x e y no quadrado e repita os passo b-d;



g) Aplique um zoom na imagem e repita os passo b-d;



h) Explique o que acontece com a transformada de Fourier quando é aplicado a rotação, translação e zoom.

1. Rotação: A rotação de uma imagem não tem um efeito direto e simples no domínio da frequência. a rotação resulta em uma mudança correspondente no espectro de frequência.
2. Translação: Mudança de fase no domínio da frequência, mas não altera o espectro de magnitude. Isso significa que se você mover uma imagem, a Transformada de Fourier da imagem deslocada terá a mesma magnitude da Transformada de Fourier original, mas a fase será diferente.
3. Zoom (Escala): O zoom em uma imagem tem o efeito de compressão ou expansão no domínio da frequência. Isso significa que se você ampliar uma imagem, a Transformada de Fourier da imagem ampliada será comprimida em relação à Transformada de Fourier original.

2 e 3:

Código:

```
import cv2 as cv

import numpy as np

import matplotlib.pyplot as plt

from math import sqrt

from math import exp


def inverse_fourier_transform(fshift):

    f_ishift = np.fft.ifftshift(fshift)

    im1_back = np.fft.ifft2(f_ishift)

    im1_back = np.abs(im1_back)

    return im1_back


def distance(point1, point2):
```



```

        return sqrt((point1[0]-point2[0])**2 + (point1[1]-point2[1])**2)

def idealFilterLP(D0,imgShape):

    base = np.zeros(imgShape[:2])

    rows, cols = imgShape[:2]

    center = (rows/2,cols/2)

    for x in range(cols):

        for y in range(rows):

            if distance((y,x),center) < D0:

                base[y,x] = 1

    return base

def idealFilterHP(D0,imgShape):

    base = np.ones(imgShape[:2])

    rows, cols = imgShape[:2]

    center = (rows/2,cols/2)

    for x in range(cols):

        for y in range(rows):

            if distance((y,x),center) < D0:

                base[y,x] = 0

    return base

def butterworthLP(D0,imgShape,n):

    base = np.zeros(imgShape[:2])

    rows, cols = imgShape[:2]

```

```

        center = (rows/2,cols/2)

        for x in range(cols):

            for y in range(rows):

                base[y,x] = 1/(1+(distance((y,x),center)/D0)**(2*n))

    return base

def butterworthHP(D0,imgShape,n):

    base = np.zeros(imgShape[:2])

    rows, cols = imgShape[:2]

    center = (rows/2,cols/2)

    for x in range(cols):

        for y in range(rows):

            base[y,x] = 1-1/(1+(distance((y,x),center)/D0)**(2*n))

    return base

def gaussianLP(D0,imgShape):

    base = np.zeros(imgShape[:2])

    rows, cols = imgShape[:2]

    center = (rows/2,cols/2)

    for x in range(cols):

        for y in range(rows):

            base[y,x] = exp(((distance((y,x),center)**2)/(2*(D0**2))))

    return base

def gaussianHP(D0,imgShape):

```

```

base = np.zeros(imgShape[:2])

rows, cols = imgShape[:2]

center = (rows/2,cols/2)

for x in range(cols):

    for y in range(rows):

        base[y,x] = 1 -
exp(((distance((y,x),center)**2)/(2*(D0**2))))

    return base

def plotarPassaAlta(im1):

    f = np.fft.fft2(im1)

    fshift = np.fft.fftshift(f)

    magnitude_spectrum = 20*np.log(np.abs(fshift))

    fase = np.angle(fshift)

    PassaaltaIdeal= idealFilterHP(50,fshift.shape)

    PassaaltaButterworth= butterworthHP(50,fshift.shape,2)

    PassaltaGaussiano= gaussianHP(50,fshift.shape)

    ImAltaGaussiano = fshift*PassaltaGaussiano

    ImAltaIdeal = fshift*PassaaltaIdeal

    ImAltaButterworth = fshift*PassaaltaButterworth

    ImAltaIdeal =
inverse_fourier_transform(np.fft.ifftshift(ImAltaIdeal))

```

```
ImAltaButterworth =  
inverse_fourier_transform(np.fft.ifftshift(ImAltaButterworth))  
  
ImAltaGaussiano =  
inverse_fourier_transform(np.fft.ifftshift(ImAltaGaussiano))  
  
  
  
plt.subplot(331),plt.imshow(im1, cmap = 'gray')  
  
plt.title("Original"), plt.xticks([]), plt.yticks([])  
  
  
  
plt.subplot(332),plt.imshow(magnitude_spectrum, cmap = 'gray')  
  
plt.title('Magnitude'), plt.xticks([]), plt.yticks([])  
  
  
  
plt.subplot(333),plt.imshow(fase, cmap = 'gray')  
  
plt.title('Fase'), plt.xticks([]), plt.yticks([])  
  
  
  
plt.subplot(334),plt.imshow(PassaaltaIdeal, cmap = 'gray')  
  
plt.title('Passa Baixa Ideal'), plt.xticks([]), plt.yticks([])  
  
  
  
plt.subplot(335),plt.imshow(PassaaltaButterworth, cmap = 'gray')  
  
plt.title('Passa Baixa Butterworth'), plt.xticks([]),  
plt.yticks([])  
  
  
  
plt.subplot(336),plt.imshow(PassaltaGaussiano, cmap = 'gray')  
  
plt.title('Passa Baixa Gaussiano'), plt.xticks([]), plt.yticks([])
```

```

plt.subplot(337),plt.imshow(ImAltaIdeal, cmap = 'gray')

plt.title('Imagem Passa Baixa Ideal'), plt.xticks([]),
plt.yticks([])

plt.subplot(338),plt.imshow(ImAltaButterworth, cmap = 'gray')

plt.title('Imagem Passa Baixa Butterworth'), plt.xticks([]),
plt.yticks([])

plt.subplot(339),plt.imshow(ImAltaGaussiano, cmap = 'gray')

plt.title('Imagem Passa Baixa Gaussiano'), plt.xticks([]),
plt.yticks([])

plt.show()

def plotarpassabaixa(im1):

    f = np.fft.fft2(im1)

    fshift = np.fft.fftshift(f)

    magnitude_spectrum = 20*np.log(np.abs(fshift))

    fase = np.angle(fshift)

    PassabaixaIdeal= idealFilterLP(50,fshift.shape)

    PassabaixaButterworth= butterworthLP(50,fshift.shape,2)

```

```
PassabaixaGaussiano= gaussianLP(50,fshift.shape)

ImBaixaButterworth = fshift*PassabaixaButterworth

ImBaixaIdeal = fshift*PassabaixaIdeal

ImBaixaGaussiano = fshift*PassabaixaGaussiano


plt.subplot(331),plt.imshow(im1, cmap = 'gray')

plt.title("Original"), plt.xticks([]), plt.yticks([])


plt.subplot(332),plt.imshow(magnitude_spectrum, cmap = 'gray')

plt.title('Magnitude'), plt.xticks([]), plt.yticks([])


plt.subplot(333),plt.imshow(fase, cmap = 'gray')

plt.title('Fase'), plt.xticks([]), plt.yticks([])


plt.subplot(334),plt.imshow(PassabaixaIdeal, cmap = 'gray')

plt.title('Passa Baixa Ideal'), plt.xticks([]), plt.yticks([])


plt.subplot(335),plt.imshow(PassabaixaButterworth, cmap = 'gray')

plt.title('Passa Baixa Butterworth'), plt.xticks([]),
plt.yticks([])


plt.subplot(336),plt.imshow(PassabaixaGaussiano, cmap = 'gray')
```

```

plt.title('Passa Baixa Gaussiano'), plt.xticks([]), plt.yticks([])

plt.subplot(337),plt.imshow(np.real(ImBaixaIdeal), cmap = 'gray')

plt.title('Imagem Passa Baixa Ideal'), plt.xticks([]),
plt.yticks([])

plt.subplot(338),plt.imshow(np.real(ImBaixaButterworth), cmap =
'gray')

plt.title('Imagem Passa Baixa Butterworth'), plt.xticks([]),
plt.yticks([])

plt.subplot(339),plt.imshow(np.real(ImBaixaGaussiano), cmap =
'gray')

plt.title('Imagem Passa Baixa Gaussiano'), plt.xticks([]),
plt.yticks([])

plt.show()

def main():

    while(True):

        print("Selecione a imagem: 1 , 2, 3 ou 4")

        im = input()

        if im == '1':

```

```
        im1 =cv.imread("car.tif", cv.IMREAD_GRAYSCALE)

    elif im == '2':

        im1 = cv.imread('len_periodic_noise.png',
cv.IMREAD_GRAYSCALE)

    elif im == '3':

        im1 = cv.imread('periodic_noise.png', cv.IMREAD_GRAYSCALE)

    elif im == '4':

        im1 = cv.imread('newspaper_shot_woman.tif',
cv.IMREAD_GRAYSCALE)

    elif im == '5':

        break

    else:

        break

while(True):

    print("Selecione a opção: 1: passa baixa, 2= passa alta ou
3= sair")

    m = int(input())

    if(m ==1 ):

        plotarpassabaixa(im1)

    elif(m==2):

        plotarPassaAlta(im1)

    elif(m==3):

        break

    else:
```



```
break

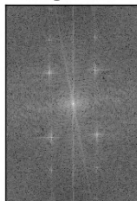
if __name__ == "__main__":
    main()
```

Passa baixa:

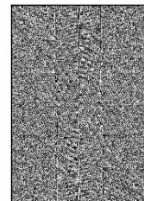
Original



Magnitude



Fase



Passa Baixa Ideal



Passa Baixa Butterworth



Passa Baixa Gaussiano



Imagem Passa Baixa Ideal



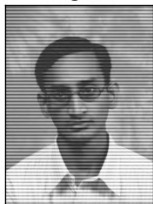
Imagem Passa Baixa Butterworth



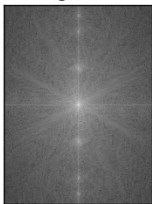
Imagem Passa Baixa Gaussiano



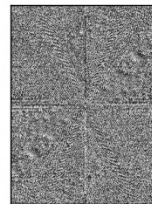
Original



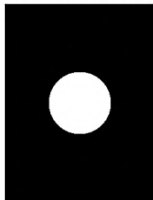
Magnitude



Fase



Passa Baixa Ideal



Passa Baixa Butterworth



Passa Baixa Gaussiano



Imagem Passa Baixa Ideal



Imagem Passa Baixa Butterworth



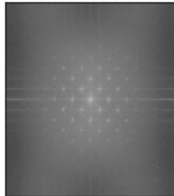
Imagem Passa Baixa Gaussiano



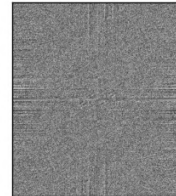
Original



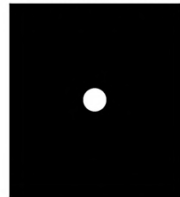
Magnitude



Fase



Passa Baixa Ideal



Passa Baixa Butterworth



Passa Baixa Gaussiano

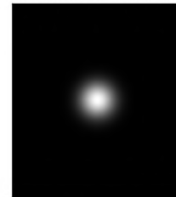


Imagem Passa Baixa Ideal



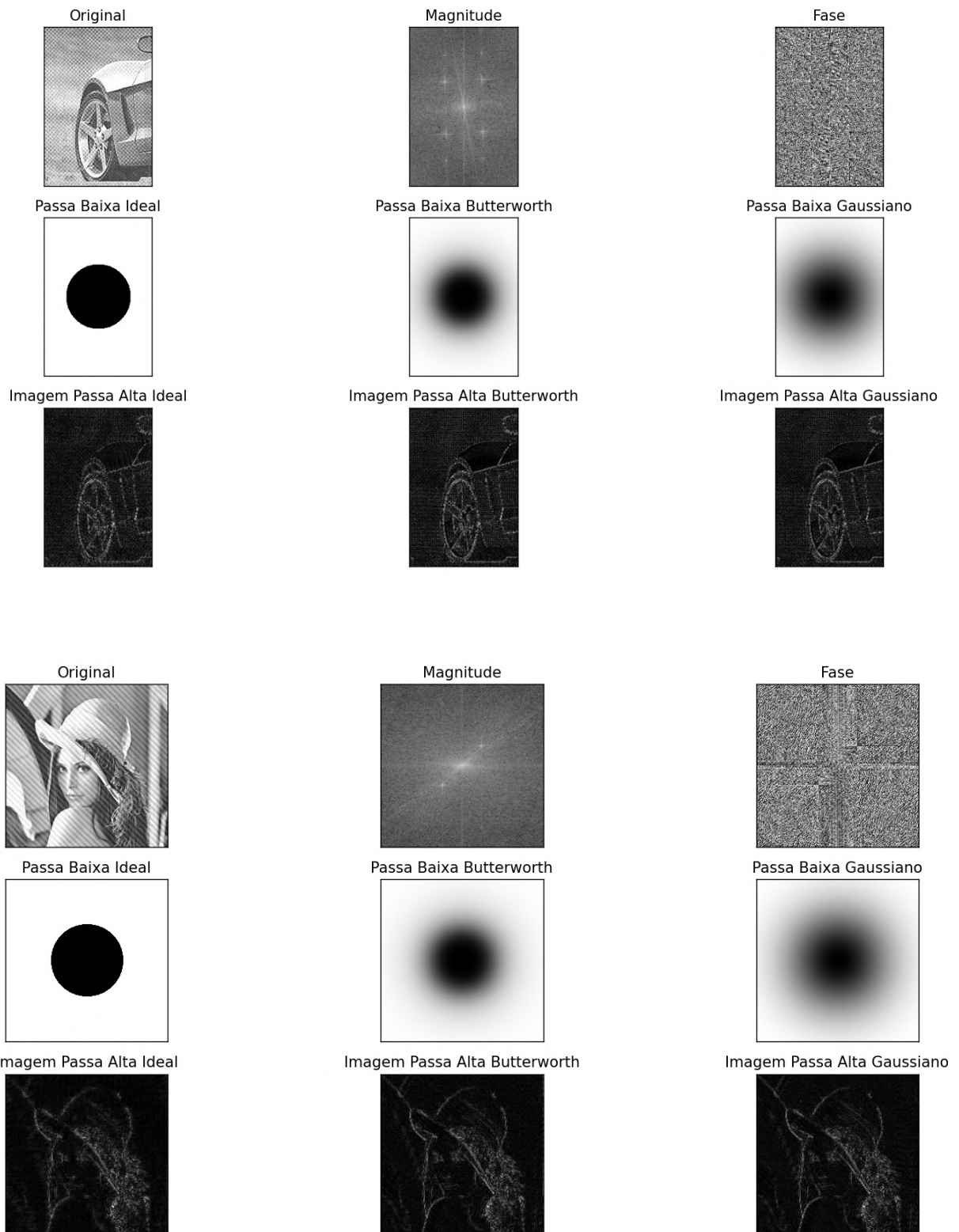
Imagem Passa Baixa Butterworth

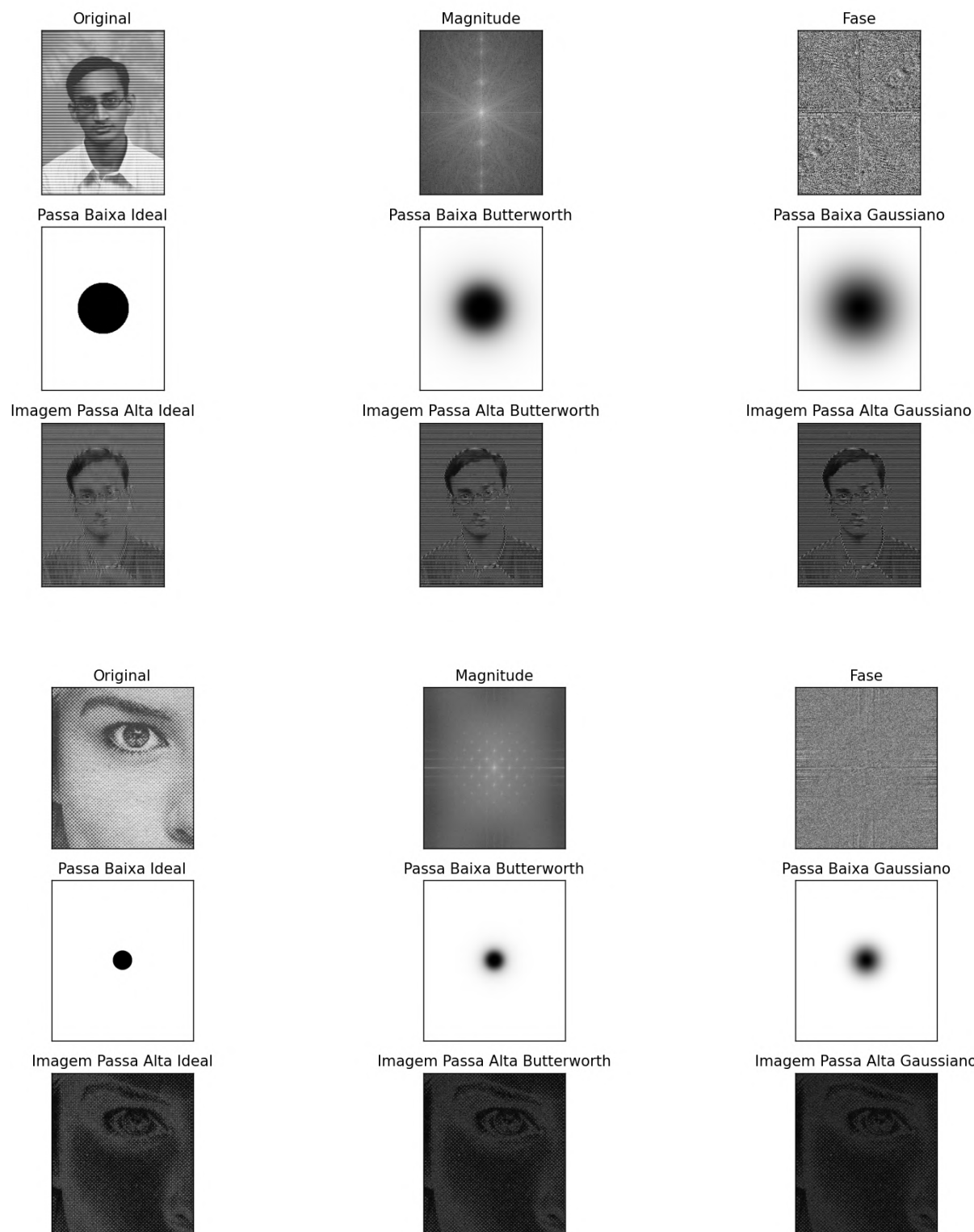


Imagem Passa Baixa Gaussiano



Passa alta:





4:

código:



```

def plotarpassabaixa(im1):

    f = np.fft.fft2(im1)

    fshift = np.fft.fftshift(f)

    magnitude_spectrum = 20*np.log(np.abs(fshift))

    fase = np.angle(fshift)

    PassabaixaIdeal= idealFilterLP(50,fshift.shape)

    # PassabaixaButterworth= butterworthLP(50,fshift.shape,2)

    # PassabaixaGaussiano= gaussianLP(50,fshift.shape)

    ImBaixaIdeal = fshift*PassabaixaIdeal

    # ImBaixaButterworth = fshift*PassabaixaButterworth

    # ImBaixaGaussiano = fshift*PassabaixaGaussiano

    ImBaixaIdeal =
inverse_fourier_transform(np.fft.ifftshift(ImBaixaIdeal))

    # ImBaixaButterworth =
inverse_fourier_transform(np.fft.ifftshift(ImBaixaButterworth))

    # ImBaixaGaussiano =
inverse_fourier_transform(np.fft.ifftshift(ImBaixaGaussiano))

```

```
plt.subplot(231),plt.imshow(im1, cmap = 'gray')

plt.title("Original"), plt.xticks([]), plt.yticks([])


plt.subplot(232),plt.imshow(magnitude_spectrum, cmap = 'gray')

plt.title('Magnitude'), plt.xticks([]), plt.yticks([])


plt.subplot(233),plt.imshow(fase, cmap = 'gray')

plt.title('Fase'), plt.xticks([]), plt.yticks([])


plt.subplot(234),plt.imshow(PassabaixaIdeal, cmap = 'gray')

plt.title('Passa Baixa Ideal'), plt.xticks([]), plt.yticks([])


# plt.subplot(335),plt.imshow(PassabaixaButterworth, cmap = 'gray')

# plt.title('Passa Baixa Butterworth'), plt.xticks([]),
plt.yticks([])


# plt.subplot(336),plt.imshow(PassabaixaGaussiano, cmap = 'gray')

# plt.title('Passa Baixa Gaussiano'), plt.xticks([]),
plt.yticks([])


plt.subplot(235),plt.imshow(np.real(ImBaixaIdeal), cmap = 'gray')

plt.title('Imagem Passa Baixa Ideal'), plt.xticks([]),
plt.yticks([])
```

```

        # plt.subplot(338),plt.imshow(np.real(ImBaixaButterworth), cmap =
'gray')

        # plt.title('Imagem Passa Baixa Butterworth'), plt.xticks([]),
plt.yticks([])


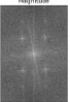



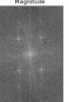



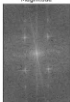






        # plt.subplot(339),plt.imshow(np.real(ImBaixaGaussiano), cmap =
'gray')

        # plt.title('Imagem Passa Baixa Gaussiano'), plt.xticks([]),
plt.yticks([])

plt.show()

```

alterando a chamada da função `idelFilterLP`

D0 = 0,01	D0 =0,05	D0 =0,5	teste adicional D0 =20
   	   	   	   

5:

Código:

```

def plotarPassaAlta(im1):

    f = np.fft.fft2(im1)

    fshift = np.fft.fftshift(f)

```

```

magnitude_spectrum = 20*np.log(np.abs(fshift))

#fase = np.angle(fshift)

PassaaltaIdeal= idealFilterHP(50,fshift.shape)

# PassaaltaButterworth= butterworthHP(50,fshift.shape,2)

# PassaltaGaussiano= gaussianHP(50,fshift.shape)

ImAltaIdeal = fshift*PassaaltaIdeal

# ImAltaGaussiano = fshift*PassaltaGaussiano

# ImAltaButterworth = fshift*PassaaltaButterworth


ImAltaIdeal =
inverse_fourier_transform(np.fft.ifftshift(ImAltaIdeal))

# ImAltaButterworth =
inverse_fourier_transform(np.fft.ifftshift(ImAltaButterworth))

# ImAltaGaussiano =
inverse_fourier_transform(np.fft.ifftshift(ImAltaGaussiano))


plt.subplot(221),plt.imshow(im1, cmap = 'gray')

plt.title("Original"), plt.xticks([]), plt.yticks([])


plt.subplot(222),plt.imshow(magnitude_spectrum, cmap = 'gray')

plt.title('Magnitude'), plt.xticks([]), plt.yticks([])

```



```
# plt.subplot(333),plt.imshow(fase, cmap = 'gray')

# plt.title('Fase'), plt.xticks([]), plt.yticks([])


plt.subplot(223),plt.imshow(PassaaltaIdeal, cmap = 'gray')

plt.title('Passa Alta Ideal'), plt.xticks([]), plt.yticks([])


# plt.subplot(335),plt.imshow(PassaaltaButterworth, cmap = 'gray')

# plt.title('Passa Alta Butterworth'), plt.xticks([]),
plt.yticks([])


# plt.subplot(336),plt.imshow(PassaltaGaussiano, cmap = 'gray')

# plt.title('Passa Alta Gaussiano'), plt.xticks([]), plt.yticks([])


plt.subplot(224),plt.imshow(ImAltaIdeal, cmap = 'gray')

plt.title('Imagem Passa Alta Ideal'), plt.xticks([]),
plt.yticks([])


# plt.subplot(338),plt.imshow(ImAltaButterworth, cmap = 'gray')

















# plt.title('Imagem Passa Alta Butterworth'), plt.xticks([]),
plt.yticks([])


# plt.subplot(339),plt.imshow(ImAltaGaussiano, cmap = 'gray')

# plt.title('Imagem Passa Alta Gaussiano'), plt.xticks([]),
plt.yticks([])


plt.show()
```

Alterando a chamada da função idealFilterHP

D0=0,01	D0=0,05	D0 =0,5	teste adicional D0=20
<div><div>Original</div><div>Passa Alta Ideal</div></div> <div><div>Magnitude</div><div>Imagem Passa Alta Ideal</div></div>	<div><div>Original</div><div>Passa Alta Ideal</div></div> <div><div>Magnitude</div><div>Imagem Passa Alta Ideal</div></div>	<div><div>Original</div><div>Passa Alta Ideal</div></div> <div><div>Magnitude</div><div>Imagem Passa Alta Ideal</div></div>	<div><div>Original</div><div>Passa Alta Ideal</div></div> <div><div>Magnitude</div><div>Imagem Passa Alta Ideal</div></div>

6:

Um filtro passa-faixa (ou passa-banda) é um filtro que permite a passagem das frequências de uma certa faixa e rejeita (atenua) as frequências fora dessa faixa. Ele pode ser usado para limitar a largura de banda do sinal de saída em transmissores e receptores sem fio, ou para eliminar sinais indesejáveis, por exemplo ruídos.

Imagens:

Original



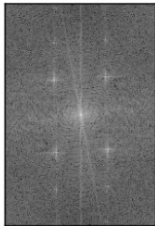
Passa Baixa Ideal



magem Passa Faixa Ideal



Magnitude



Passa Alta Ideal



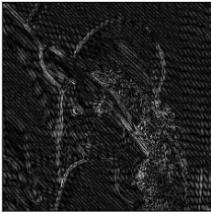
Original



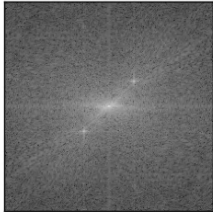
Passa Baixa Ideal



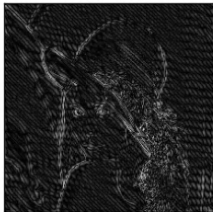
Imagem Passa Faixa Ideal



Magnitude



Passa Alta Ideal



Original



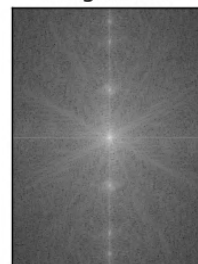
Passa Baixa Ideal



Imagem Passa Faixa Ideal



Magnitude



Passa Alta Ideal



Original



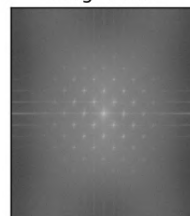
Passa Baixa Ideal



Imagem Passa Faixa Ideal



Magnitude



Passa Alta Ideal

