Luiz Henrique Botega Cervantes BI3008479

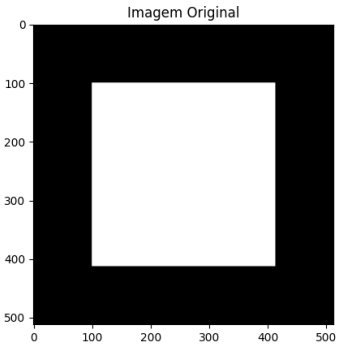
Filtragem no domínio da frequência

```python
import numpy as np
import cv2
import matplotlib.pyplot as plt

# Função para calcular e exibir o espectro de Fourier (amplitudes)
def plot_amplitude_spectrum(image):
    fft_image = np.fft.fft2(image)
    amplitude_spectrum = np.abs(fft_image)
    plt.imshow(np.log1p(amplitude_spectrum), cmap='gray')
    plt.title('Espectro de Fourier (Amplitudes)')
    plt.colorbar()
    plt.show()

# Função para calcular e exibir o espectro de Fourier (fases)
def plot_phase_spectrum(image):
    fft_image = np.fft.fft2(image)
    phase_spectrum = np.angle(fft_image)
    plt.imshow(phase_spectrum, cmap='gray')
    plt.title('Espectro de Fourier (Fases)')
    plt.colorbar()
    plt.show()

# Função para calcular e exibir o espectro de Fourier centralizado
def plot_shifted_amplitude_spectrum(image):
    fft_image = np.fft.fft2(image)
    fft_shifted = np.fft.fftshift(fft_image)
    amplitude_spectrum_shifted = np.abs(fft_shifted)
    plt.imshow(np.log1p(amplitude_spectrum_shifted), cmap='gray')
    plt.title('Espectro de Fourier Centralizado (Amplitudes)')
    plt.colorbar()
    plt.show()

# a) Crie e visualize uma imagem simples - quadrado branco sobre fundo
preto
image = np.zeros((512, 512), dtype=np.uint8)
cv2.rectangle(image, (100, 100), (412, 412), 255, -1)
plt.imshow(image, cmap='gray')
plt.title('Imagem Original')
plt.show()
```

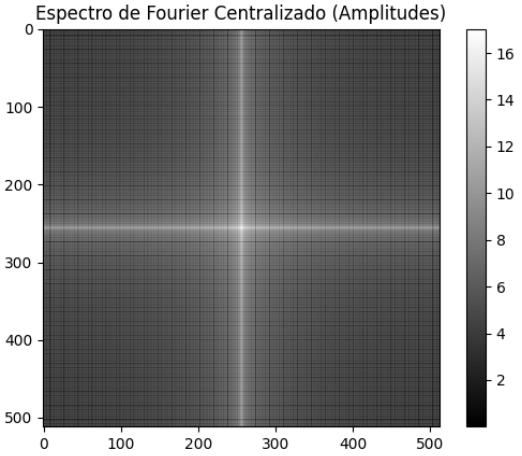
```
# b) Calcular e visualizar o espectro de Fourier (amplitudes)
plot_amplitude_spectrum(image)

# c) Calcular e visualizar o espectro de Fourier (fases)
plot_phase_spectrum(image)

# d) Obter e visualizar o espectro de Fourier centralizado
plot_shifted_amplitude_spectrum(image)

# e) Aplicar uma rotação de 40° no quadrado e repetir os passos b-d
angle = 40
rotated_image = cv2.warpAffine(image, cv2.getRotationMatrix2D((256,
256), angle, 1), (512, 512))
plt.imshow(rotated_image, cmap='gray')
plt.title('Imagem Rotacionada')
plt.show()

plot_amplitude_spectrum(rotated_image)
plot_phase_spectrum(rotated_image)
plot_shifted_amplitude_spectrum(rotated_image)

# f) Aplicar uma translação nos eixos x e y no quadrado e repetir os
passos b-d
translation_x = 50
translation_y = 30
translated_image = np.roll(image, (translation_x, translation_y),
axis=(1, 0))
plt.imshow(translated_image, cmap='gray')
plt.title('Imagem Transladada')
plt.show()

plot_amplitude_spectrum(translated_image)
plot_phase_spectrum(translated_image)
plot_shifted_amplitude_spectrum(translated_image)

# g) Aplicar um zoom na imagem e repetir os passos b-d
zoomed_image = cv2.resize(image, (256, 256))
plt.imshow(zoomed_image, cmap='gray')
plt.title('Imagem Ampliada/Reduzida')
plt.show()

plot_amplitude_spectrum(zoomed_image)
```

```
plot_phase_spectrum(zoomed_image)
plot_shifted_amplitude_spectrum(zoomed_image)
```

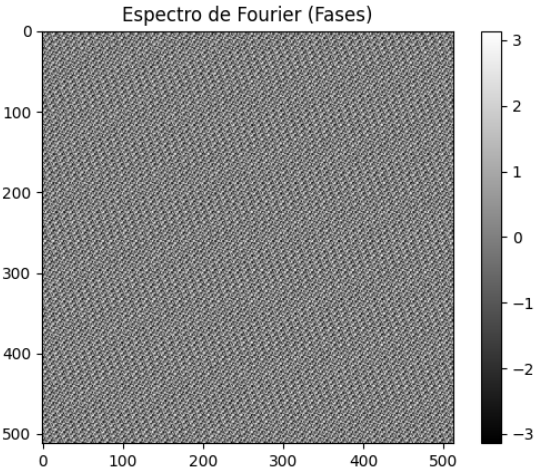
1. a)



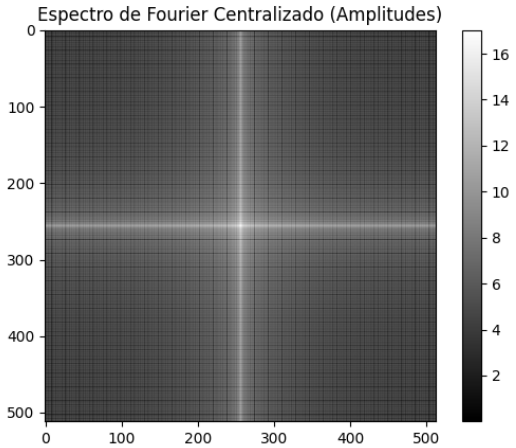
b)



c)



d)


Espectro de Fourier Centralizado (Amplitudes)

e)

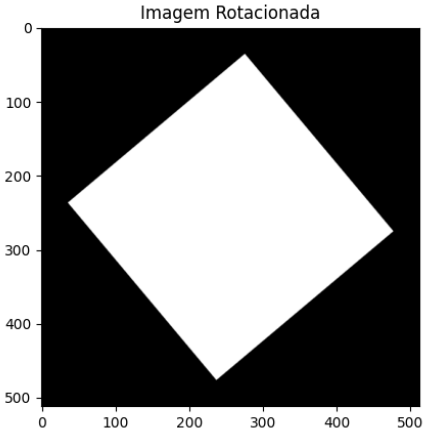
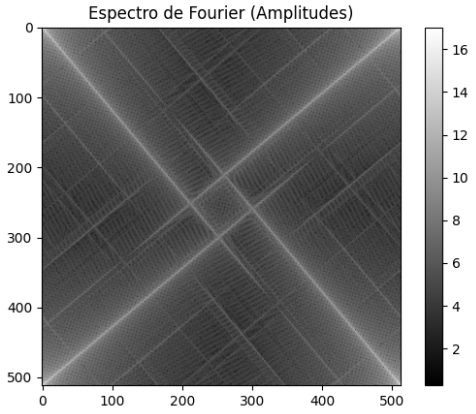
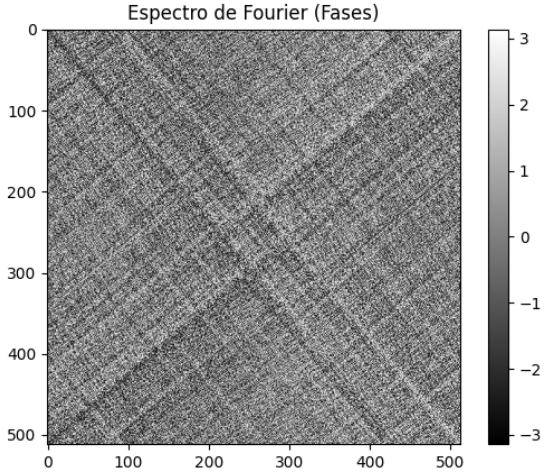
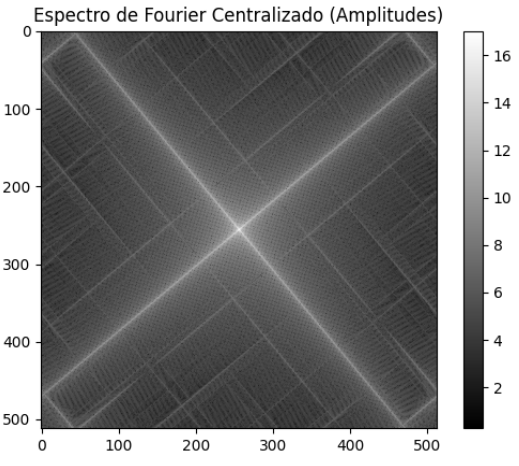


Imagem Rotacionada

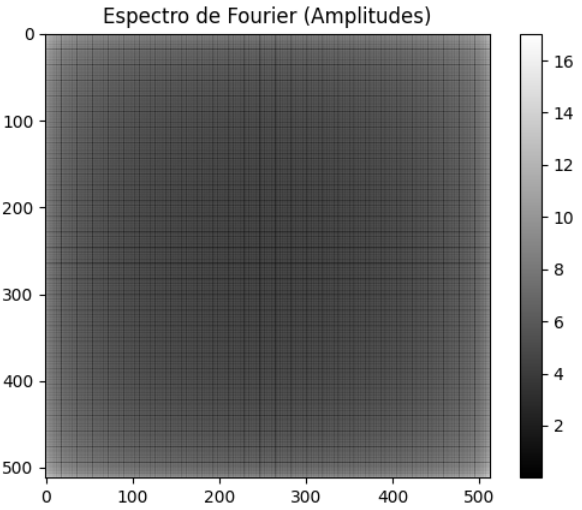

Espectro de Fourier (Amplitudes)

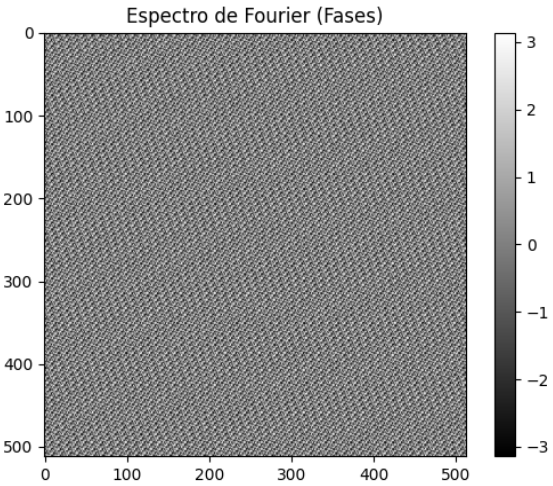

Espectro de Fourier (Fases)

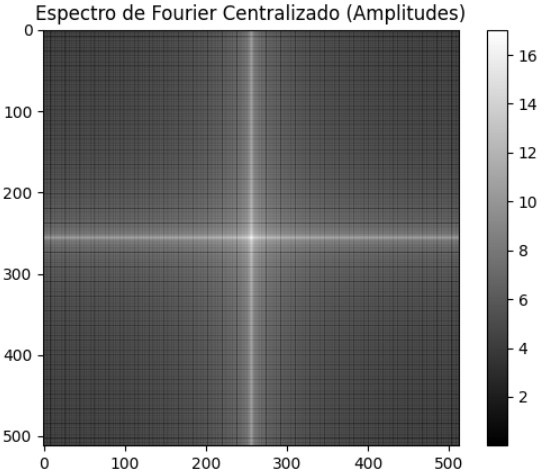

Espectro de Fourier Centralizado (Amplitudes)

f)


Imagem Transladada


Espectro de Fourier (Amplitudes)


Espectro de Fourier (Fases)


Espectro de Fourier Centralizado (Amplitudes)

g)

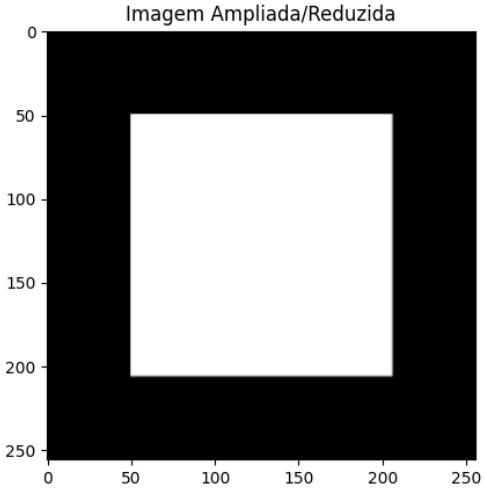
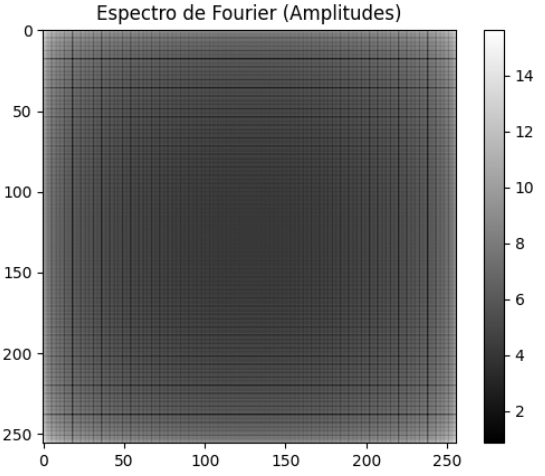
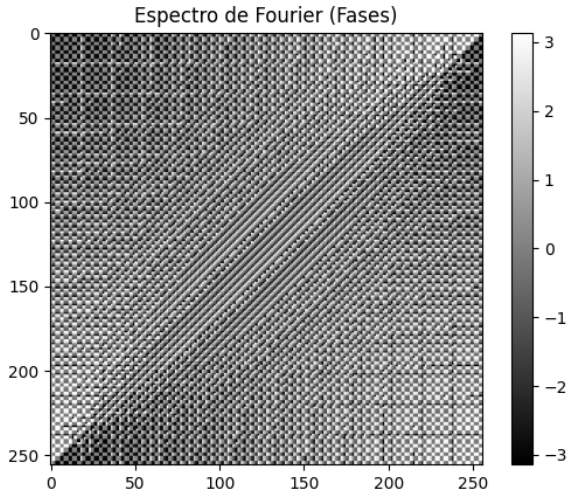
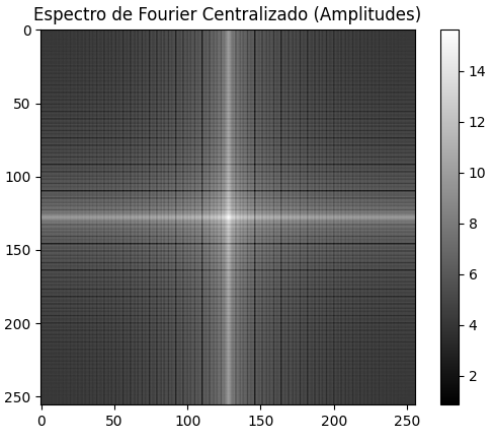


Imagem Ampliada/Reduzida


Espectro de Fourier (Amplitudes)





h) Aplicar as mudanças alteram a direção dos raios do espectro e aplicar o zoom aumenta a espessura dos traços.

2)

```
import numpy as np
import matplotlib.pyplot as plt
import cv2

# Carregar a imagem
imagem = cv2.imread('biel.png', cv2.IMREAD_GRAYSCALE)

# Calcular o espectro de Fourier da imagem
espectro_fourier = np.fft.fft2(imagem)
espectro_fourier_centralizado = np.fft.fftshift(espectro_fourier)

# Dimensões da imagem
altura, largura = imagem.shape

# Criar grades de frequência para a aplicação dos filtros
x, y = np.meshgrid(np.arange(-largura/2, largura/2),
np.arange(-altura/2, altura/2))
raio = np.sqrt(x**2 + y**2)

# Definir frequência de corte para os filtros
frequencia_corte = 40

# Filtro passa-baixa ideal
filtro_ideal = (raio <= frequencia_corte).astype(float)
```

```
# Filtro Butterworth
ordem_butterworth = 3
filtro_butterworth = 1 / (1 + (raio / frequencia_corte)**(2 *
ordem_butterworth))

# Filtro gaussiano
sigma = 40
filtro_gaussiano = np.exp(-raio**2 / (2 * (sigma**2)))

# Aplicar os filtros ao espectro de Fourier
espectro_filtrado_ideal = espectro_fourier_centralizado * filtro_ideal
espectro_filtrado_butterworth = espectro_fourier_centralizado *
filtro_butterworth
espectro_filtrado_gaussiano = espectro_fourier_centralizado *
filtro_gaussiano

# Transformada inversa de Fourier para obter as imagens filtradas
imagem_filtrada_ideal =
np.abs(np.fft.ifft2(np.fft.ifftshift(espectro_filtrado_ideal)))
imagem_filtrada_butterworth =
np.abs(np.fft.ifft2(np.fft.ifftshift(espectro_filtrado_butterworth)))
imagem_filtrada_gaussiano =
np.abs(np.fft.ifft2(np.fft.ifftshift(espectro_filtrado_gaussiano)))

# Exibir as imagens
plt.figure(figsize=(12, 12))

# Imagem original
plt.subplot(3, 4, 1)
plt.imshow(imagem, cmap='gray')
plt.title('Imagem Original')

# Espectro de Fourier
plt.subplot(3, 4, 2)
plt.imshow(np.log(np.abs(espectro_fourier_centralizado) + 1),
cmap='gray')
plt.title('Espectro de Fourier')

# Filtro Passa-Baixa Ideal
plt.subplot(3, 4, 3)
plt.imshow(filtro_ideal, cmap='gray')
plt.title('Filtro Passa-Baixa Ideal')
```

```
# Imagem Resultante após Filtro Ideal
plt.subplot(3, 4, 4)
plt.imshow(imagem_filtrada_ideal, cmap='gray')
plt.title('Imagem após Filtro Ideal')

# Filtro Butterworth
plt.subplot(3, 4, 7)
plt.imshow(filtro_butterworth, cmap='gray')
plt.title('Filtro Butterworth')

# Imagem Resultante após Filtro Butterworth
plt.subplot(3, 4, 8)
plt.imshow(imagem_filtrada_butterworth, cmap='gray')
plt.title('Imagem após Filtro Butterworth')

# Filtro Gaussiano
plt.subplot(3, 4, 11)
plt.imshow(filtro_gaussiano, cmap='gray')
plt.title('Filtro Gaussiano')

# Imagem Resultante após Filtro Gaussiano
plt.subplot(3, 4, 12)
plt.imshow(imagem_filtrada_gaussiano, cmap='gray')
plt.title('Imagem após Filtro Gaussiano')

plt.tight_layout()
plt.show()
```

a)

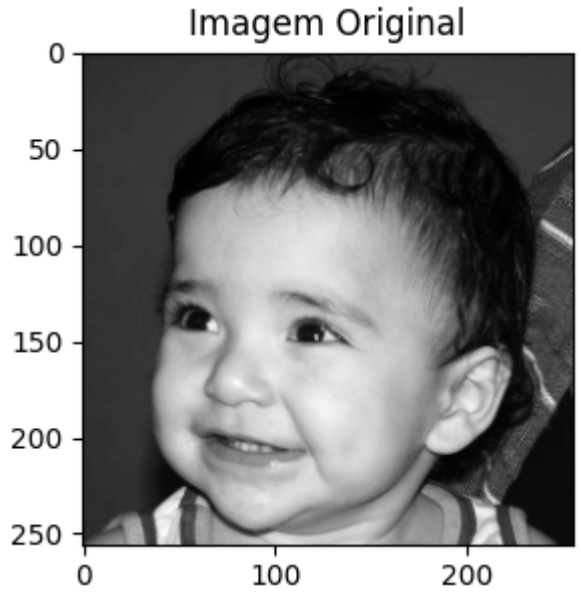
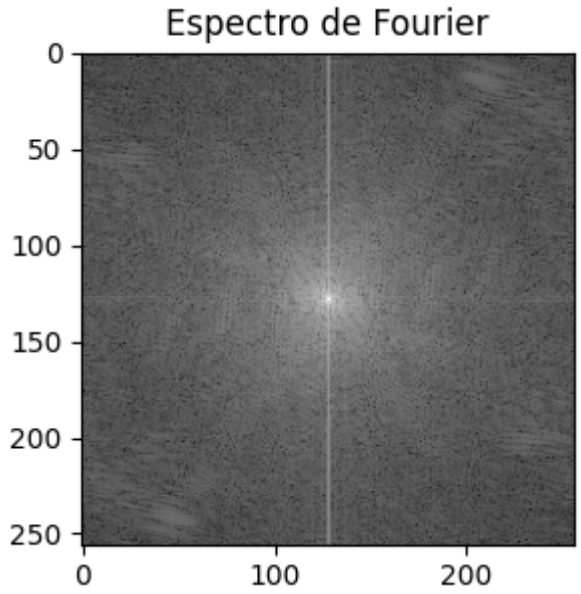
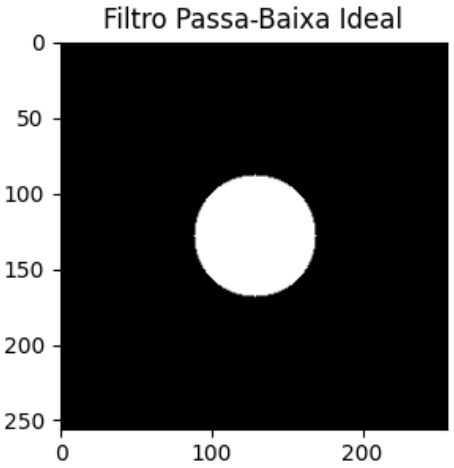


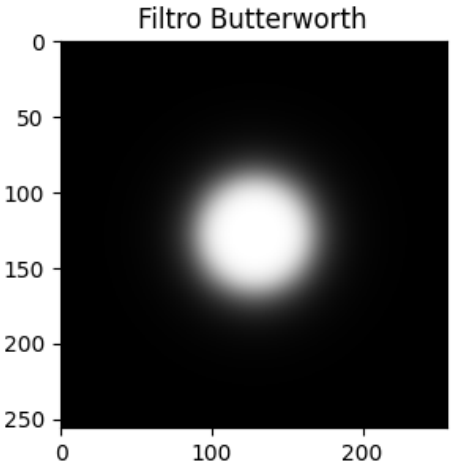
Imagem Original
0
50
100
150
200
250
0
100
200

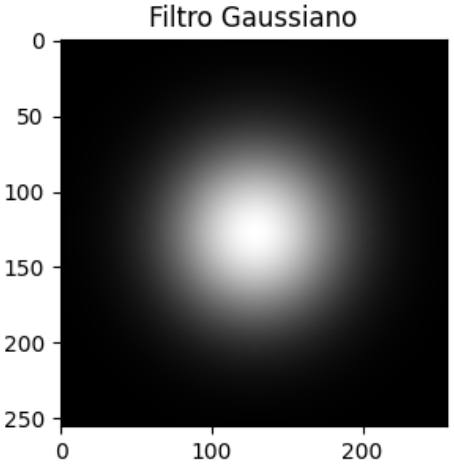
b)


Espectro de Fourier
0
50
100
150
200
250
0
100
200

c)


Filtro Passa-Baixa Ideal
0
50
100
150
200
250
0
100
200


Filtro Butterworth
0
50
100
150
200
250
0
100
200


Filtro Gaussiano
0
50
100
150
200
250
0
100
200

d)

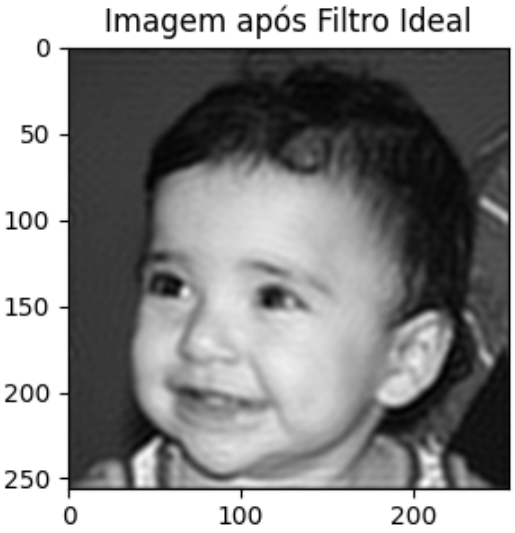


Imagem após Filtro Ideal
0
50
100
150
200
250
0
100
200

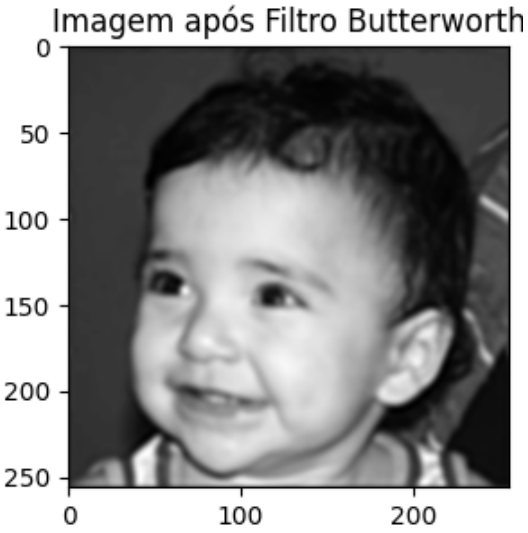


Imagem após Filtro Butterworth
0
50
100
150
200
250
0
100
200

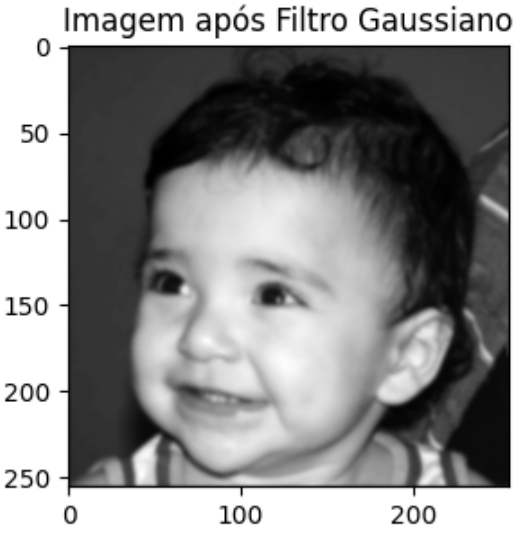


Imagem após Filtro Gaussiano
0
50
100
150
200
250
0
100
200

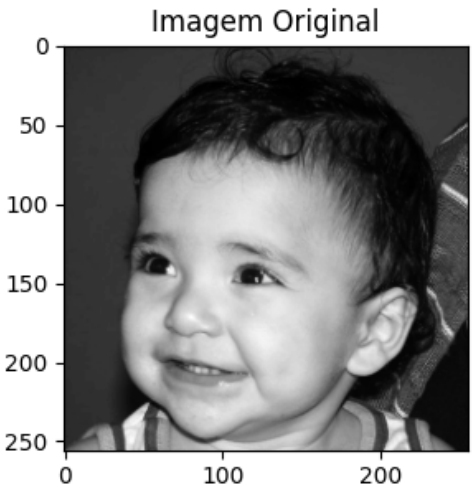
3)

```
import numpy as np
import matplotlib.pyplot as plt
import cv2

# Carregar a imagem
imagem = cv2.imread('biel.png', cv2.IMREAD_GRAYSCALE)

# Calcular o espectro de Fourier da imagem
espectro_fourier = np.fft.fft2(imagem)
espectro_fourier_centralizado = np.fft.fftshift(espectro_fourier)

# Dimensões da imagem
altura, largura = imagem.shape

# Criar grades de frequência para a aplicação dos filtros
x, y = np.meshgrid(np.arange(-largura/2, largura/2),
np.arange(-altura/2, altura/2))
raio = np.sqrt(x**2 + y**2)

# Definir frequência de corte para os filtros passa-altas
frequencia_corte = 40

# Filtro passa-alta ideal (inverso do passa-baixa ideal)
filtro_alta = (raio > frequencia_corte).astype(float)

# Filtro Butterworth passa-alta (inverso do passa-baixa Butterworth)
ordem_butterworth = 3
filtro_butterworth_alta = 1 - (1 / (1 + (raio / frequencia_corte)**(2 *
ordem_butterworth)))

# Filtro gaussiano passa-alta (inverso do passa-baixa gaussiano)
sigma = 40
filtro_gaussiano_alta = 1 - np.exp(-raio**2 / (2 * (sigma**2)))

# Aplicar os filtros ao espectro de Fourier
espectro_filtrado_alta = espectro_fourier_centralizado * filtro_alta
espectro_filtrado_butterworth_alta = espectro_fourier_centralizado *
filtro_butterworth_alta
espectro_filtrado_gaussiano_alta = espectro_fourier_centralizado *
filtro_gaussiano_alta

# Transformada inversa de Fourier para obter as imagens filtradas
```

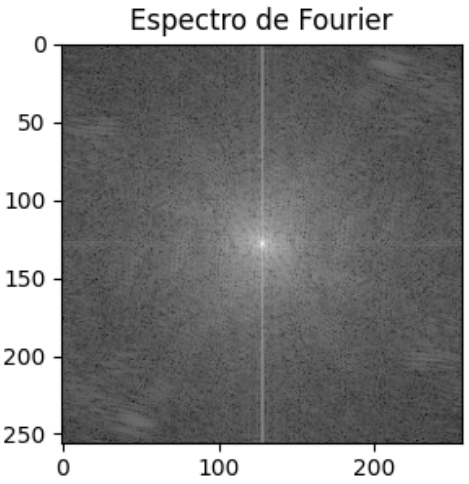
```
imagem_filtrada_alta =
np.abs(np.fft.ifft2(np.fft.ifftshift(espectro_filtrado_alta)))
imagem_filtrada_butterworth_alta =
np.abs(np.fft.ifft2(np.fft.ifftshift(espectro_filtrado_butterworth_alta
)))
imagem_filtrada_gaussiano_alta =
np.abs(np.fft.ifft2(np.fft.ifftshift(espectro_filtrado_gaussiano_alta))
)

# Exibir as imagens
plt.figure(figsize=(12, 12))

# Imagem original
plt.subplot(3, 4, 1)
plt.imshow(imagem, cmap='gray')
plt.title('Imagem Original')

# Espectro de Fourier
plt.subplot(3, 4, 2)
plt.imshow(np.log(np.abs(espectro_fourier_centralizado) + 1),
cmap='gray')
plt.title('Espectro de Fourier')

# Filtro Passa-Alta Ideal
plt.subplot(3, 4, 3)
plt.imshow(filtro_alta, cmap='gray')
plt.title('Filtro Passa-Alta Ideal')

# Imagem Resultante após Filtro Passa-Alta Ideal
plt.subplot(3, 4, 4)
plt.imshow(imagem_filtrada_alta, cmap='gray')
plt.title('Imagem após Filtro Passa-Alta Ideal')

# Filtro Butterworth Passa-Alta
plt.subplot(3, 4, 7)
plt.imshow(filtro_butterworth_alta, cmap='gray')
plt.title('Filtro Butterworth Passa-Alta')

# Imagem Resultante após Filtro Butterworth Passa-Alta
plt.subplot(3, 4, 8)
plt.imshow(imagem_filtrada_butterworth_alta, cmap='gray')
plt.title('Imagem após Filtro Butterworth Passa-Alta')
```

```
# Filtro Gaussiano Passa-Alta
plt.subplot(3, 4, 11)
plt.imshow(filtro_gaussiano_alta, cmap='gray')
plt.title('Filtro Gaussiano Passa-Alta')

# Imagem Resultante após Filtro Gaussiano Passa-Alta
plt.subplot(3, 4, 12)
plt.imshow(imagem_filtrada_gaussiano_alta, cmap='gray')
plt.title('Imagem após Filtro Gaussiano Passa-Alta')

plt.tight_layout()
plt.show()
```

a)

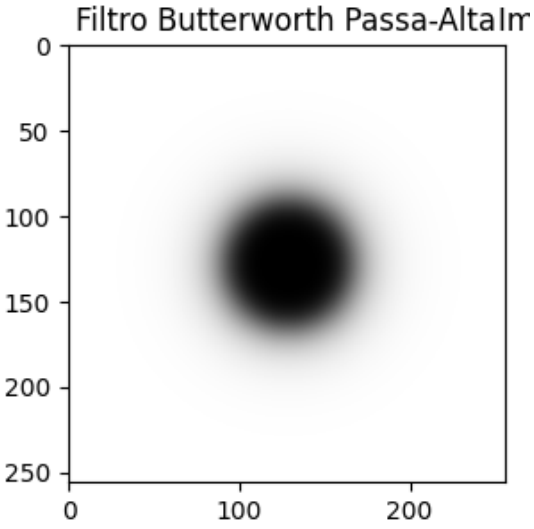


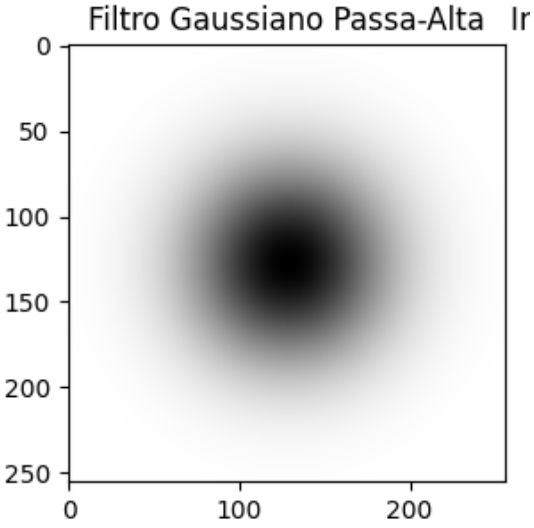
b)



c)


Filtro Passa-Alta Ideal
0
50
100
150
200
250
0
100
200


Filtro Butterworth Passa-AltaIm
0
50
100
150
200
250
0
100
200


Filtro Gaussiano Passa-Alta Ir
0
50
100
150
200
250
0
100
200

d)

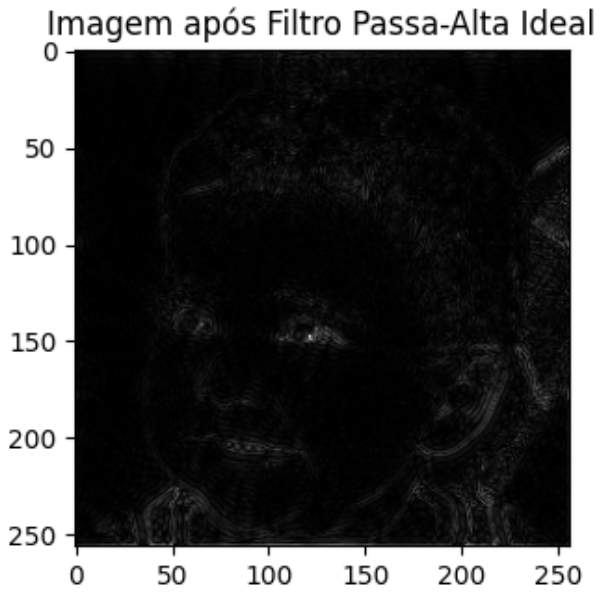
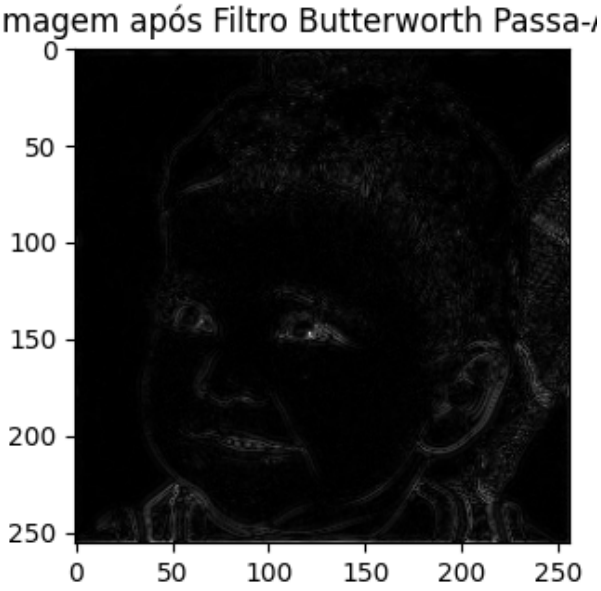


Imagem após Filtro Passa-Alta Ideal
0
50
100
150
200
250
0
50
100
150
200
250


magem após Filtro Butterworth Passa-
0
50
100
150
200
250
0
50
100
150
200
250

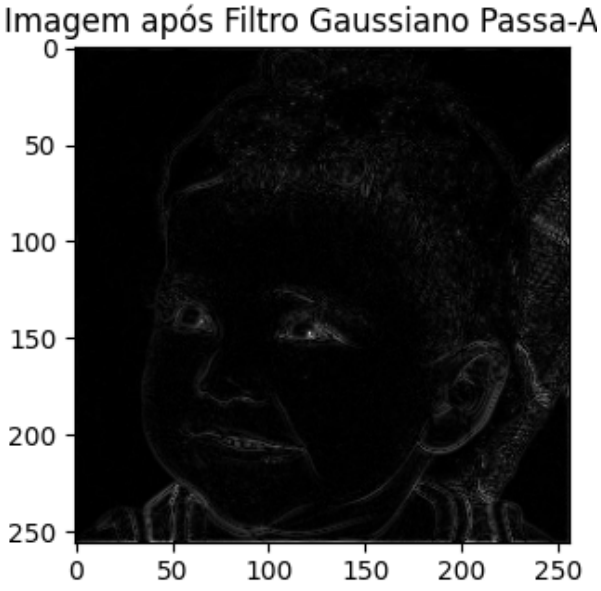
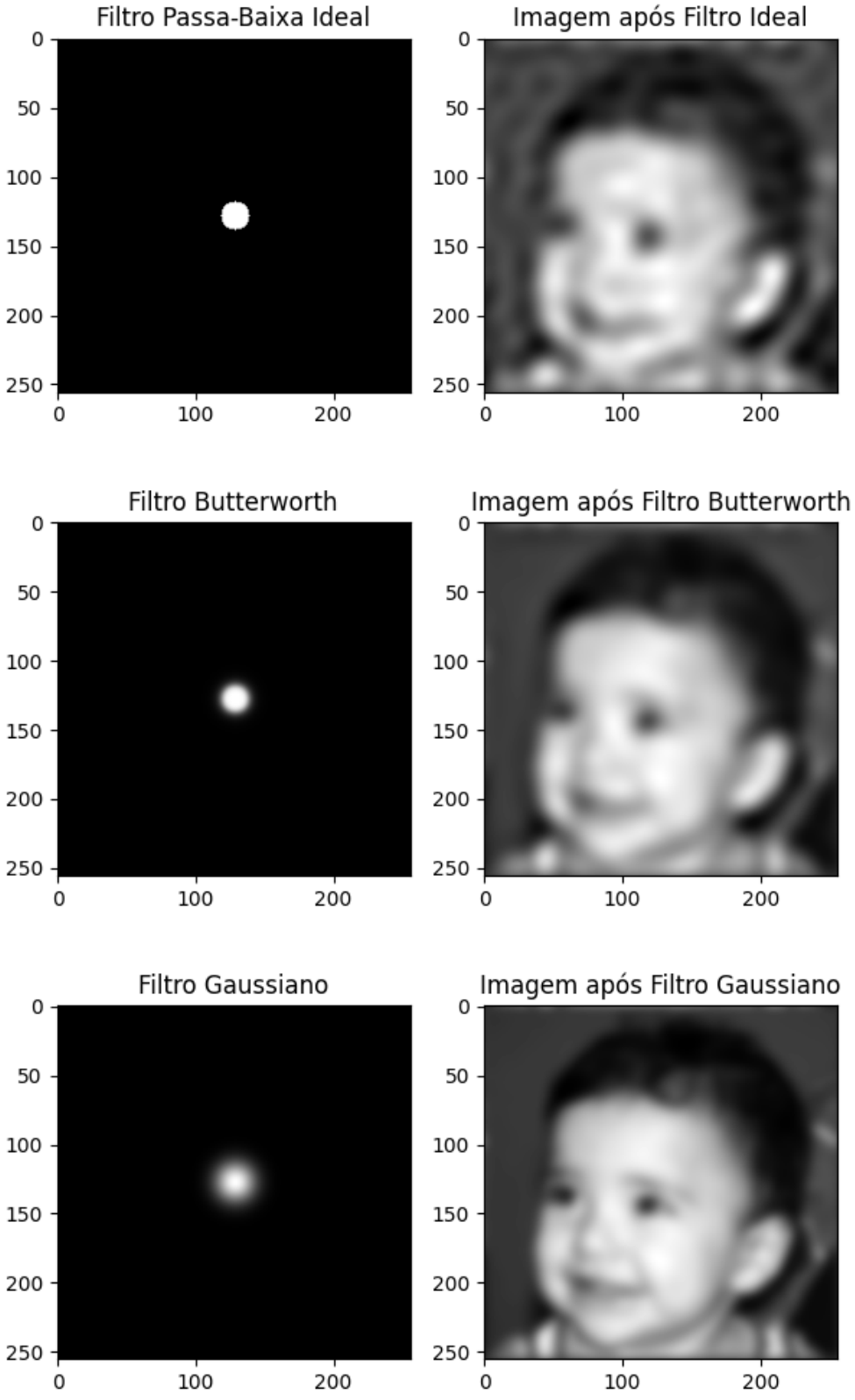


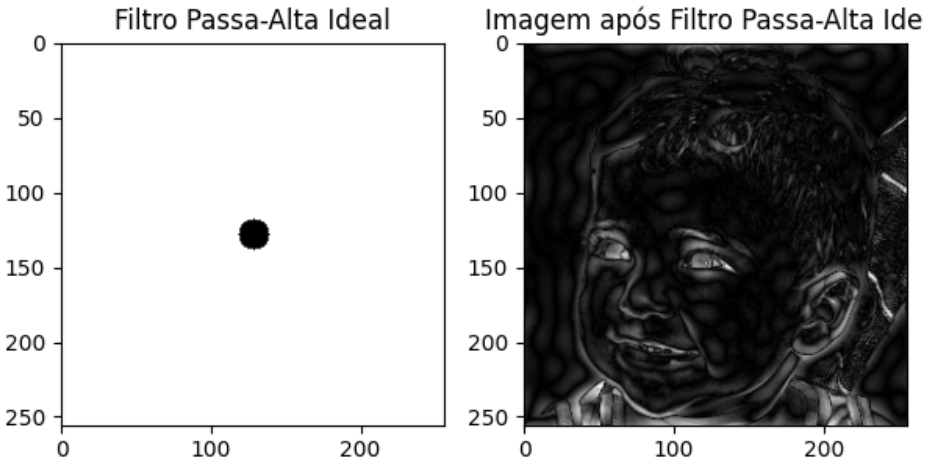
Imagem após Filtro Gaussiano Passa-A
0
50
100
150
200
250
0
50
100
150
200
250

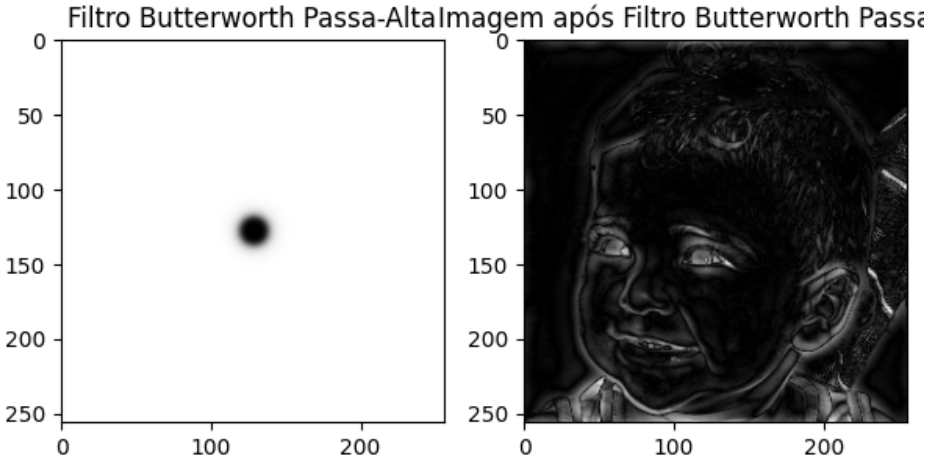
4)

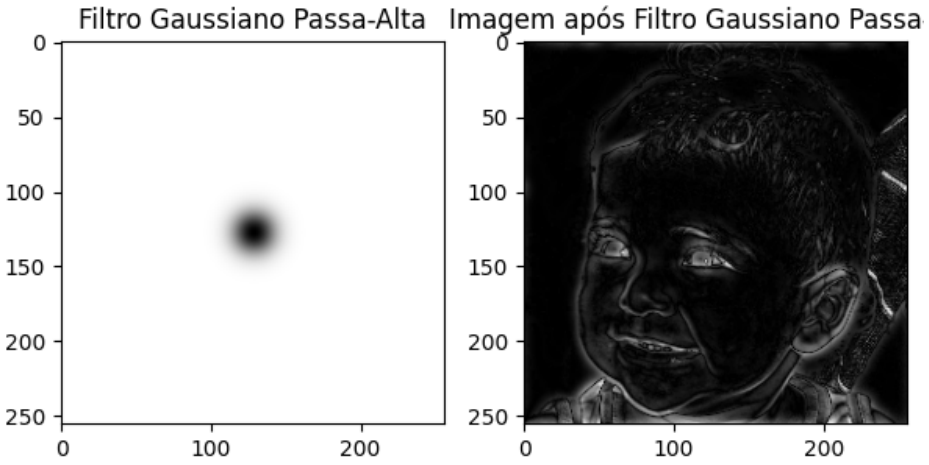


Ao diminuir a frequência de corte dos filtros, nota-se uma grande alteração em todos os filtros, as imagens ficaram mais desfocadas

5)

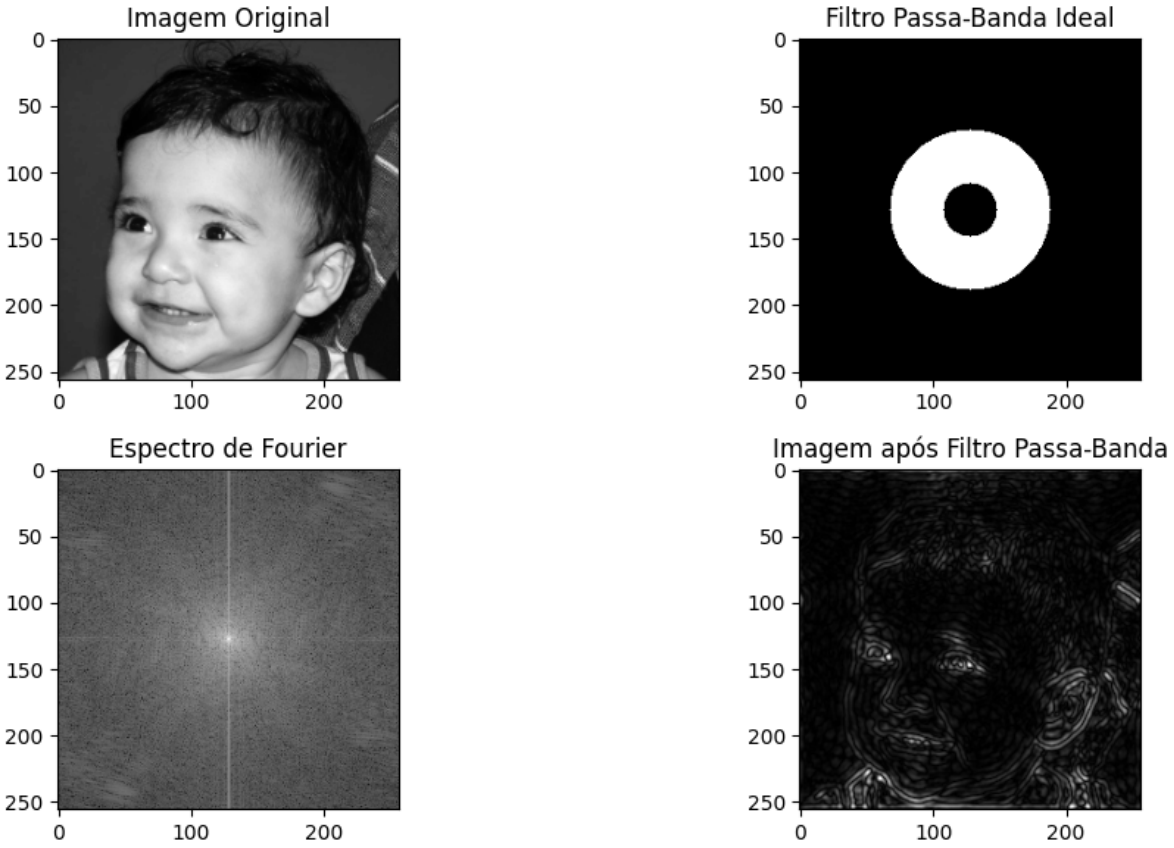






Para o filtro passa alta, podemos perceber que temos mais detalhes em frequencias de corte menores.

6) O filtro passa banda se apresenta da seguinte maneira em uma imagem:



Ele permite que frequências nem muito altas e nem muito baixas sejam consideradas, resultado nas figuras acima.