Relatório 1

April 17, 2019

1 Trabalho 1

MC920 João Vitor Araki Gonçalves (176353)

1.1 Introdução

O objetivo desse trabalho é analisar o efeito do processo de convolução de diferentes filtros, como um filtro Laplaciano, um filtro passa baixa e os efeitos do filtro de Sobel. Também será analisado os efeitos da aplicação de um filtro Laplaciano sobre o espectro de Fourrier da imagem.

1.2 O Programa

O programa foi implementado em python 3.7.3 utilizando as seguintes bibliotecas: * skimage: Leitura e plot das imagens * matplotlib: Plot das imagens * numpy: Calculo do DFT das imagens e calculos vetorizados * opencv2: Convolução e normalização das imagens

1.2.1 Execução

O programa pode ser executado pela linha de comando: *python main.py*

O programa irá usar a imagem pré definida (butterfly.png) e irá executar os filtros na ordem proposta. No caso: 1. Laplassiano 2. Passa baixa 3. Sobel vertical 4. Sobel horizontal 5. Sobel combinado 6. Espectro de fourrier da imagem

Então serão exibidos espectros de fourrier filtrados pelo filtro Gaussiano seguidas das imagens resultantes com frequências de corte crescente (10, 20, 30, 40, 50, 60)

Todas as imagens utilizadas estão na raiz do projeto, a imagem sendo filtrada pode ser alterada mudando a linha:

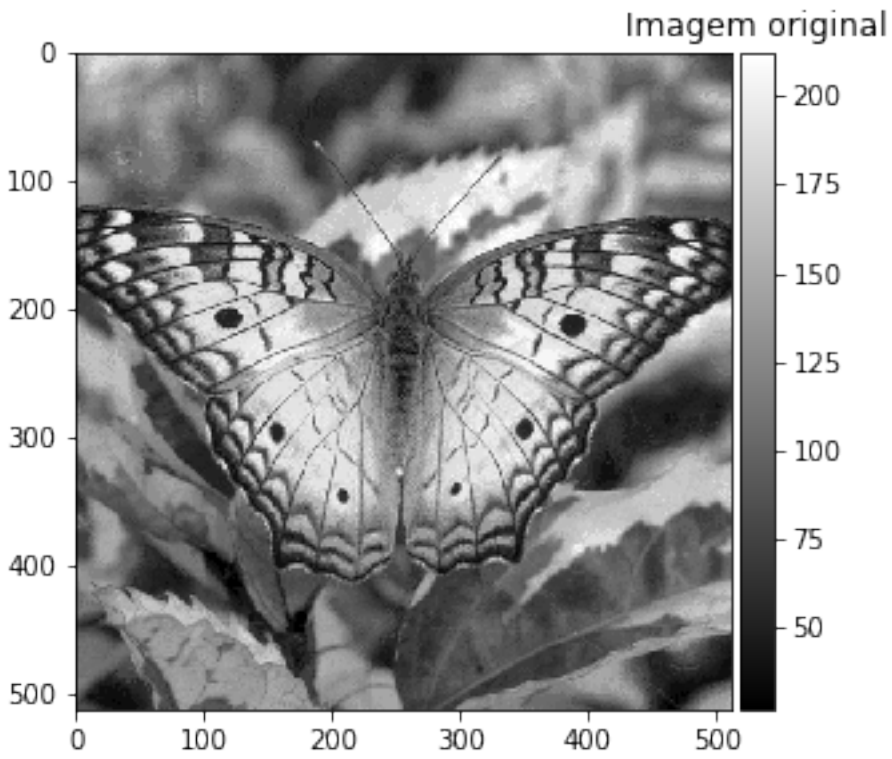
```
filename = os.path.join('./', 'butterfly.png')
```

para utilizar a imagen desejada

1.3 Processo e Resultados

1.3.1 Leitura da Imagem

A imagem é lida pelo método `skimage.io.imread` que lê a imagem e à guarda uma matriz numpy com valores de 0 à 255 para os níveis de cinza.



1.3.2 Filtragem no domínio espacial

Para a filtragem no domínio espacial, precisamos de kernels para realizar as convoluções na imagem para obter o resultado da filtragem. As diferentes máscaras foram criadas como vetores numpy, como a biblioteca opencv necessita para realizar as convoluções: Filtro Laplaciano passa alta:

```
In [2]: filter_h1

Out[2]: array([[ 0.,  0., -1.,  0.,  0.],
               [ 0., -1., -2., -1.,  0.],
               [-1., -2., 16., -2., -1.],
               [ 0., -1., -2., -1.,  0.],
               [ 0.,  0., -1.,  0.,  0.]])
```

Filtro passa baixa

```
In [3]: filter_h2

Out[3]: array([[0.00390625, 0.015625  , 0.0234375 , 0.015625  , 0.00390625],
               [0.015625  , 0.0625    , 0.09375   , 0.0625    , 0.015625  ],
               [0.0234375 , 0.09375   , 0.140625  , 0.09375   , 0.015625  ],
               [0.015625  , 0.0625    , 0.09375   , 0.0625    , 0.015625  ],
               [0.00390625, 0.015625  , 0.0234375 , 0.015625  , 0.00390625]])
```

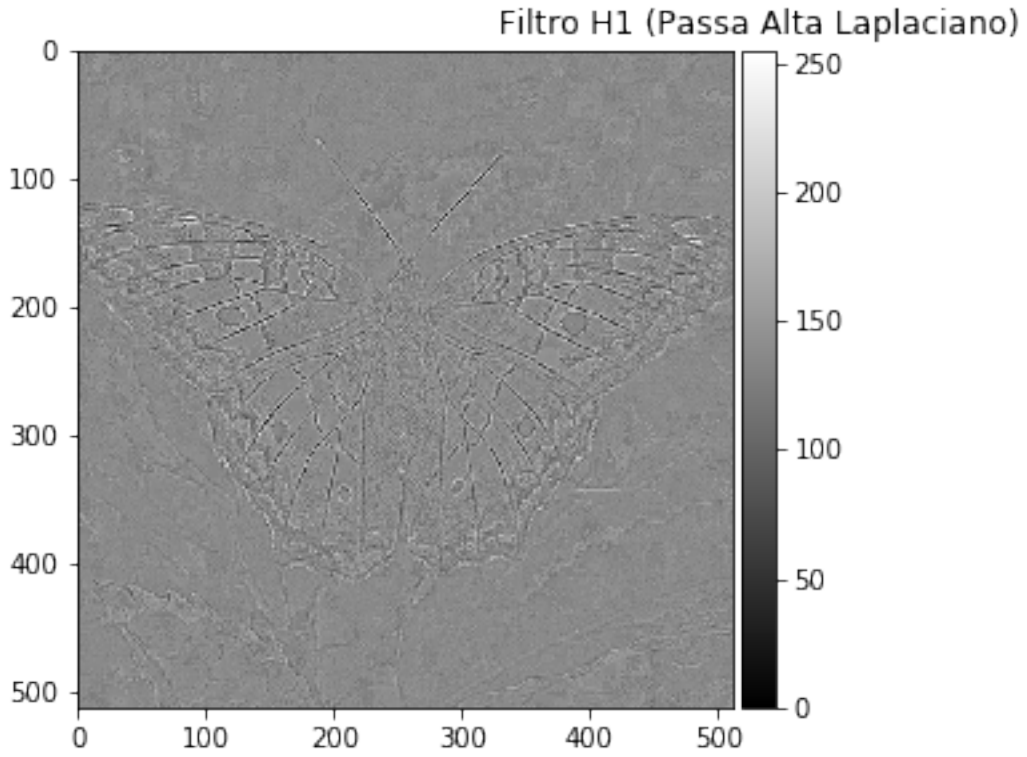
Filtro passa alta de sobel vertical

```
In [4]: filter_h3

Out[4]: array([[-1.,  0.,  1.],
               [-2.,  0.,  2.],
               [-1.,  0.,  1.]])
```

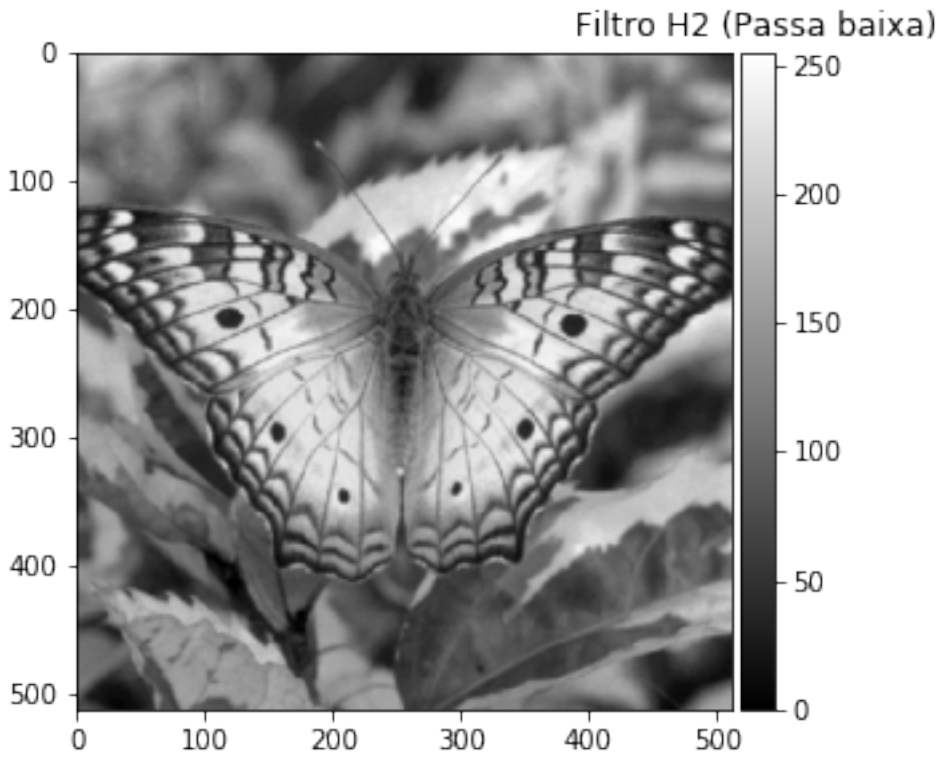
Filtro passa alta de sobel horizontal

```
In [5]: filter_h4

Out[5]: array([[-1., -2., -1.],
               [ 0.,  0.,  0.],
               [ 1.,  2.,  1.]])
```

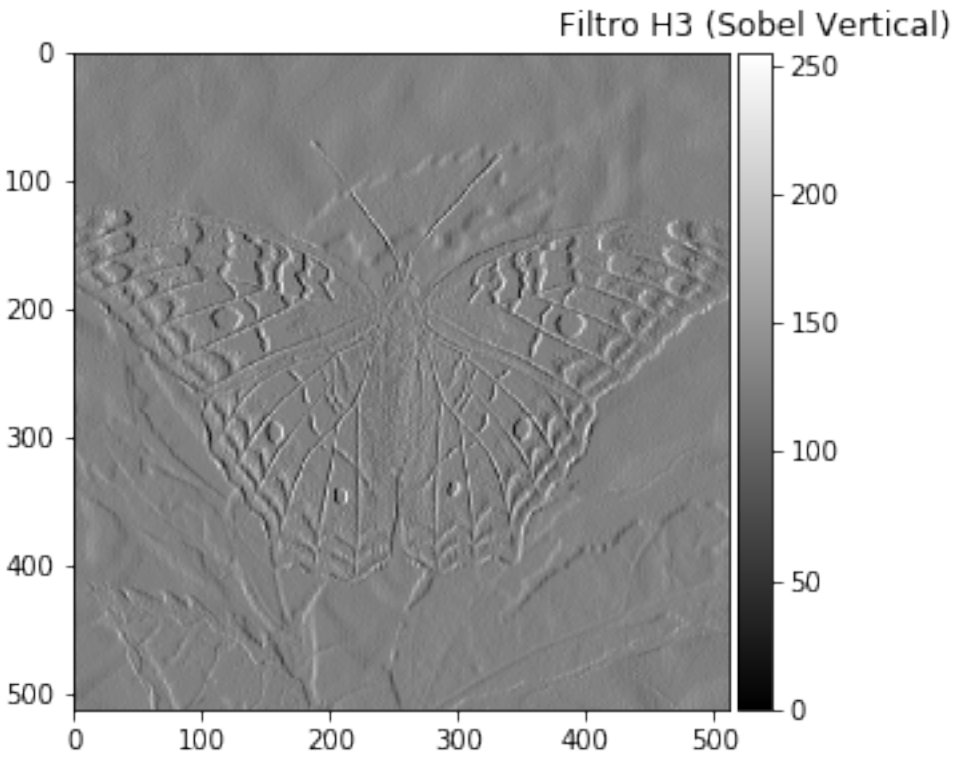
Para a realização das convoluções, foi utilizado o método `cv2.filter2D`, com parâmetros da imagem, do kernel e -1 para o nível de profundidade para ela ser igual à da imagem original, após cada convolução, foi utilizado o método `cv2.normalize` para normalizar o resultado para o espaço de 0 à 255



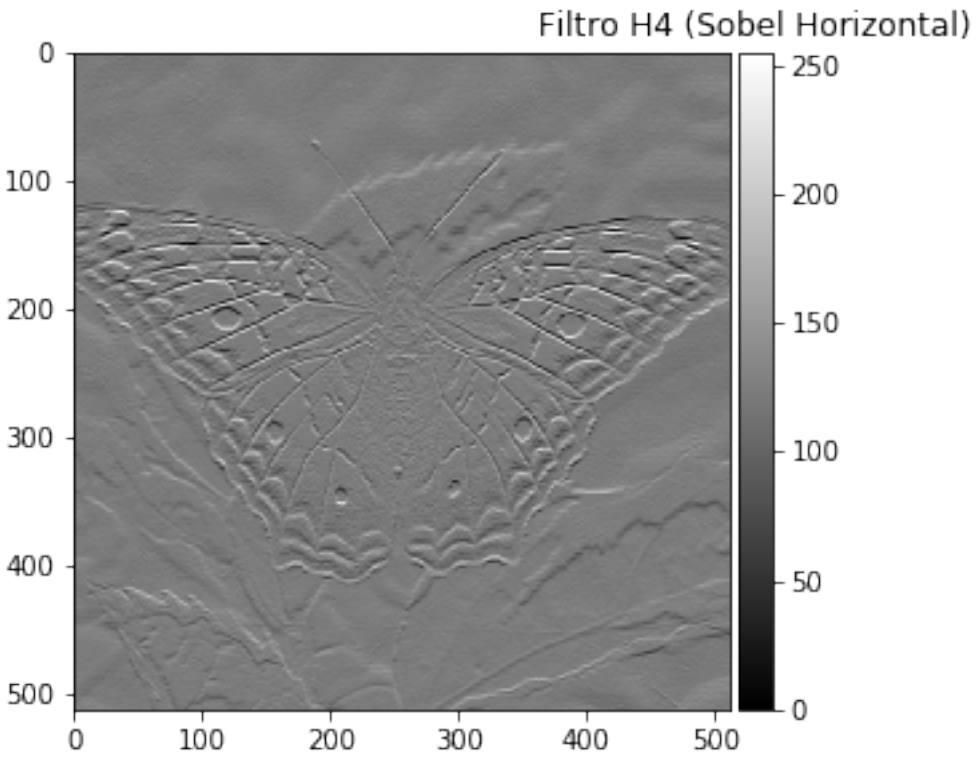
No reultado do primeiro filtro (Laplaciano) é possível observar sua caracterísca de ser bastante sensível a ruídos, mas também é possível perceber um detaque nas bordas da borboleta, que era o efeito esperado de realce de bordas.



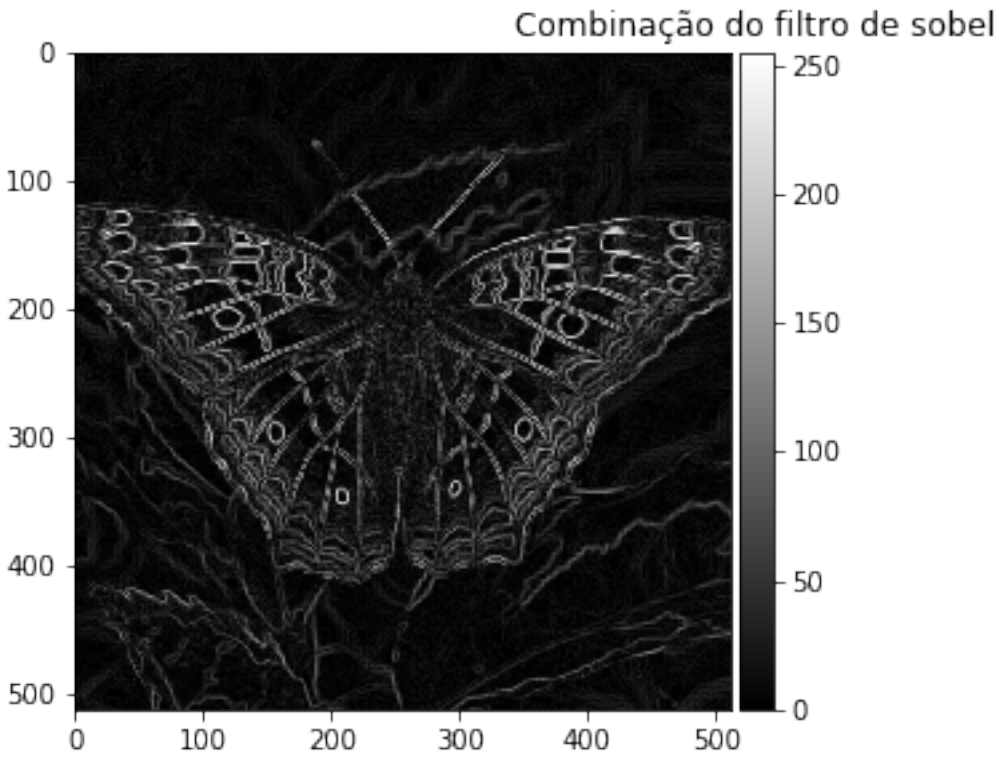
Na execução do filtro passa baixa é possível observar uma clara suavização da imagem, como esperado.



No resultado do filtro de sobel vertical, é possível notar um destaque nas bordas da esquerda e direita da imagem, ou seja das bordas verticais, como se era esperado.



No resultado do filtro de sobel vertical, é possível notar um destaque nas bordas superiores e inferiores da imagem, ou seja das bordas horizontais, como se era esperado.



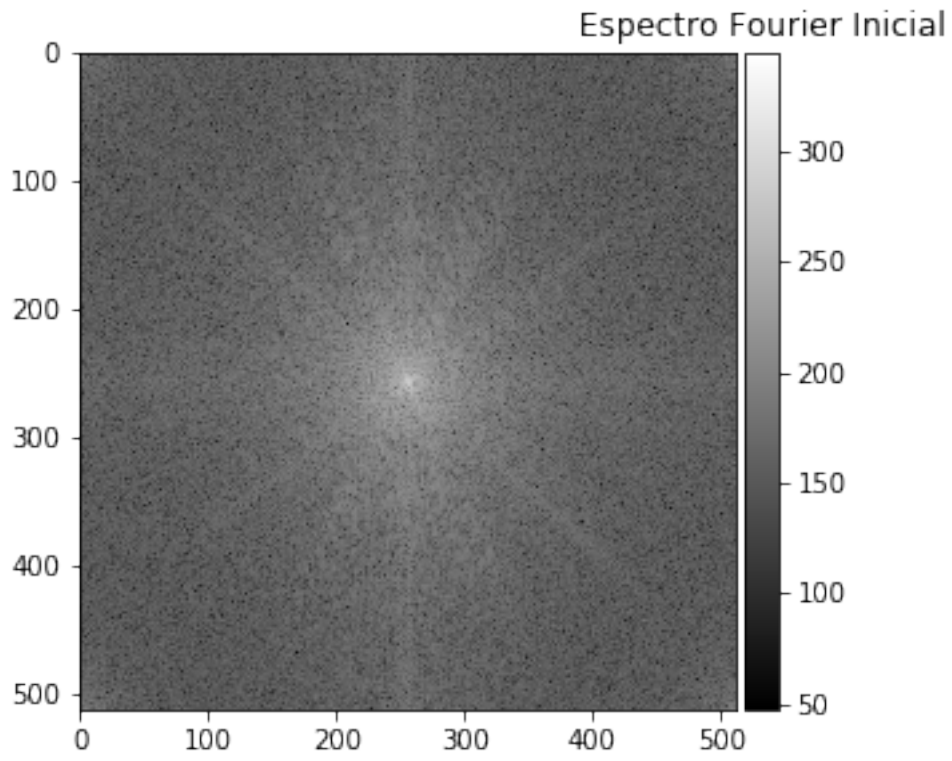
A combinação dos resultados do filtro de sobel foi realizada pela expressão nos resultados das filtragens de h3 e h4

$$\sqrt{h_3^2 + h_4^2}$$

No resultado final, é possível observar um realce de todas as bordas da imagem, que é esperado, já que o filtro de sobel é usualmente utilizado para detecção de bordas.

1.3.3 Filtragem no Domínio de Frequências

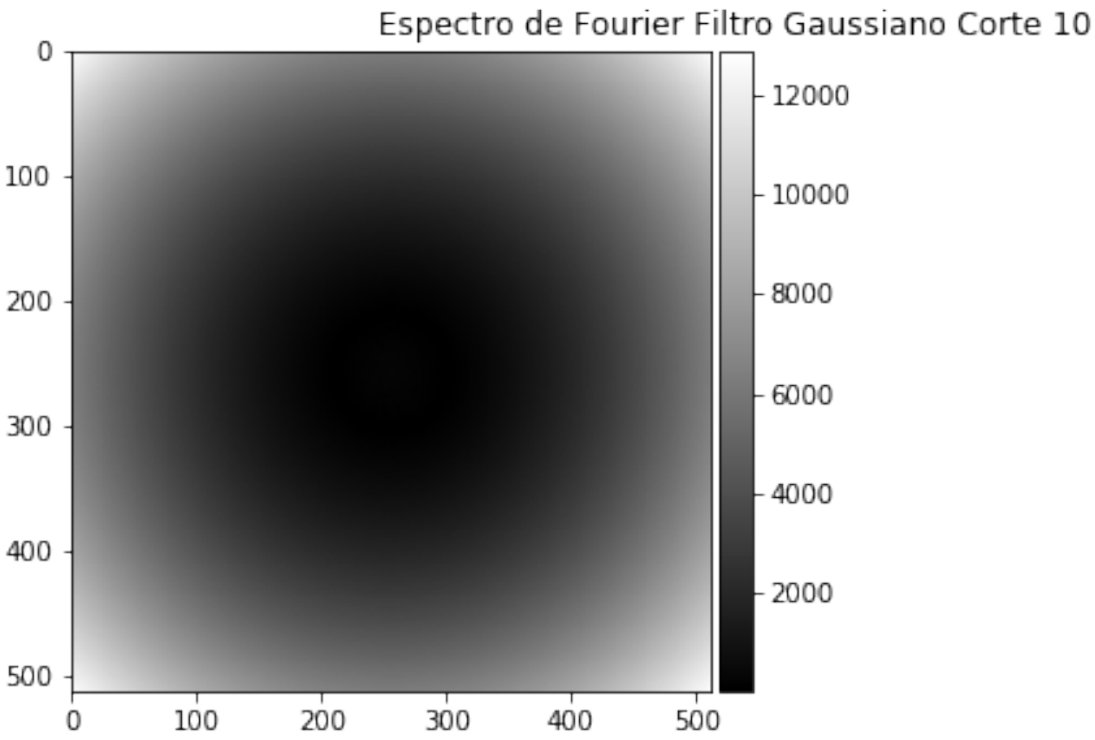
Para realizar filtragens no domínio de frequências, primeiramente precisamos transformar a imagem para o domínio de frequência por meio de uma transformada discreta de fourier (DFT). Para isso foi utilizado o método `np.fft.fft2` e para transladar a componente zero para o centro do espectro, foi utilizado o método `np.fft.fftshift`



Foi criado um filtro gaussiano do tamanho da imagem, e deslocado para o centro do espectro por meio da fórmula

$$\exp \frac{-(D(u,v) - (\frac{H}{2}, \frac{W}{2}))^2}{(2d_0^2)}$$

Subtraindo o valor de D(u,v) pela metade do tamnho da imagem para posicionar o filtro no centro da imagem, onde foi posicionada a componente 0 do espectro de fourier. Para aplicar o filtro, é aplicado o produto com o espectro de Fourier $F(u,v)H(u,v)$ Para voltar para o domínio espacial, foi utilizado o método `np.fft.ifft2()` que aplica o DFT inverso, e foi tirado pego o valor absoluto para converter de valores complexos.


Espectro de Fourier Filtro Gaussiano Corte 10
0
100
200
300
400
500
12000
10000
8000
6000
4000
2000

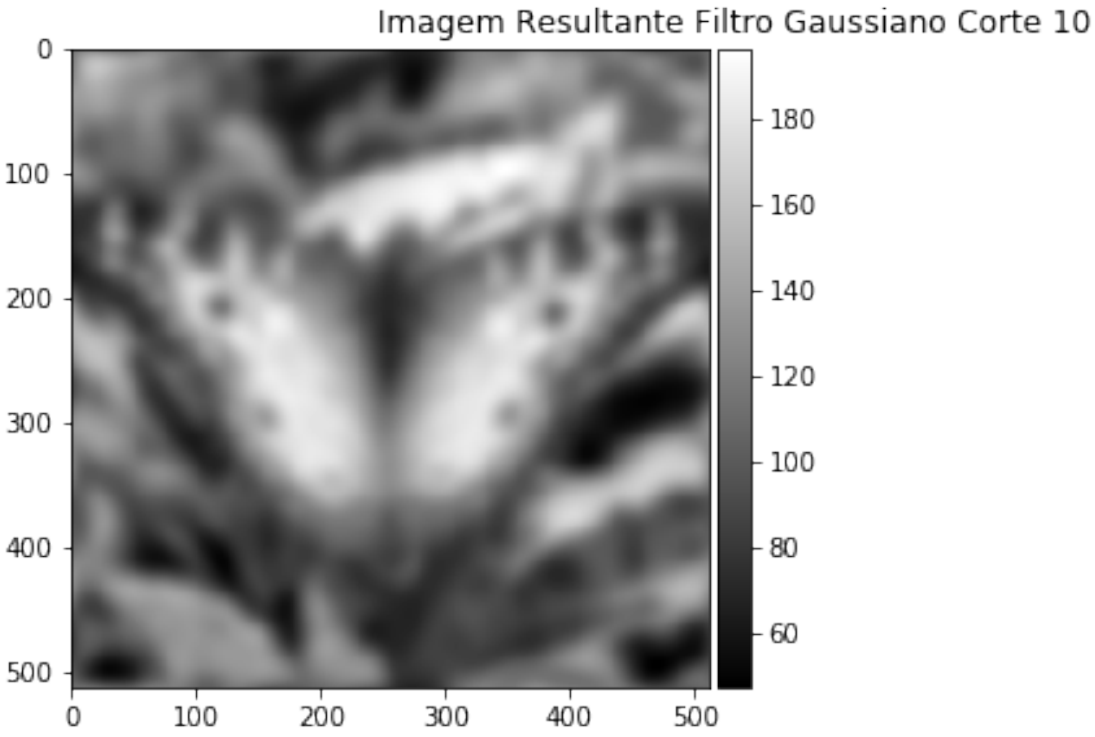
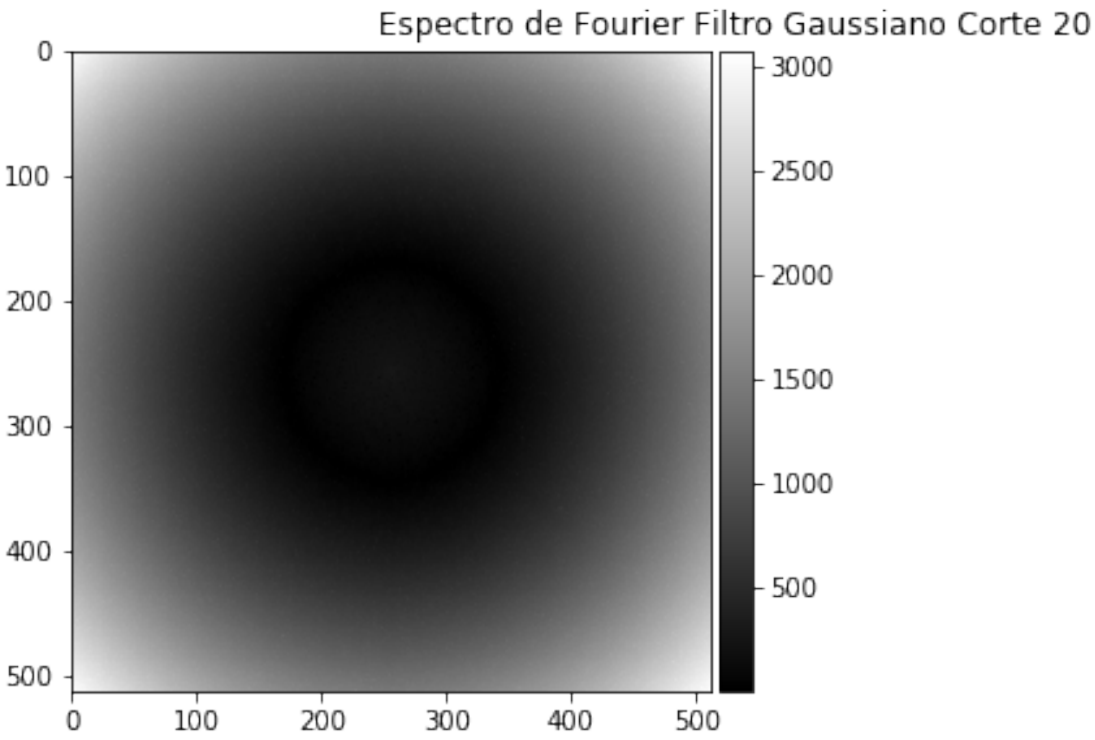


Imagem Resultante Filtro Gaussiano Corte 10
0
100
200
300
400
500
180
160
140
120
100
80
60


Espectro de Fourier Filtro Gaussiano Corte 20
0
100
200
300
400
500
3000
2500
2000
1500
1000
500

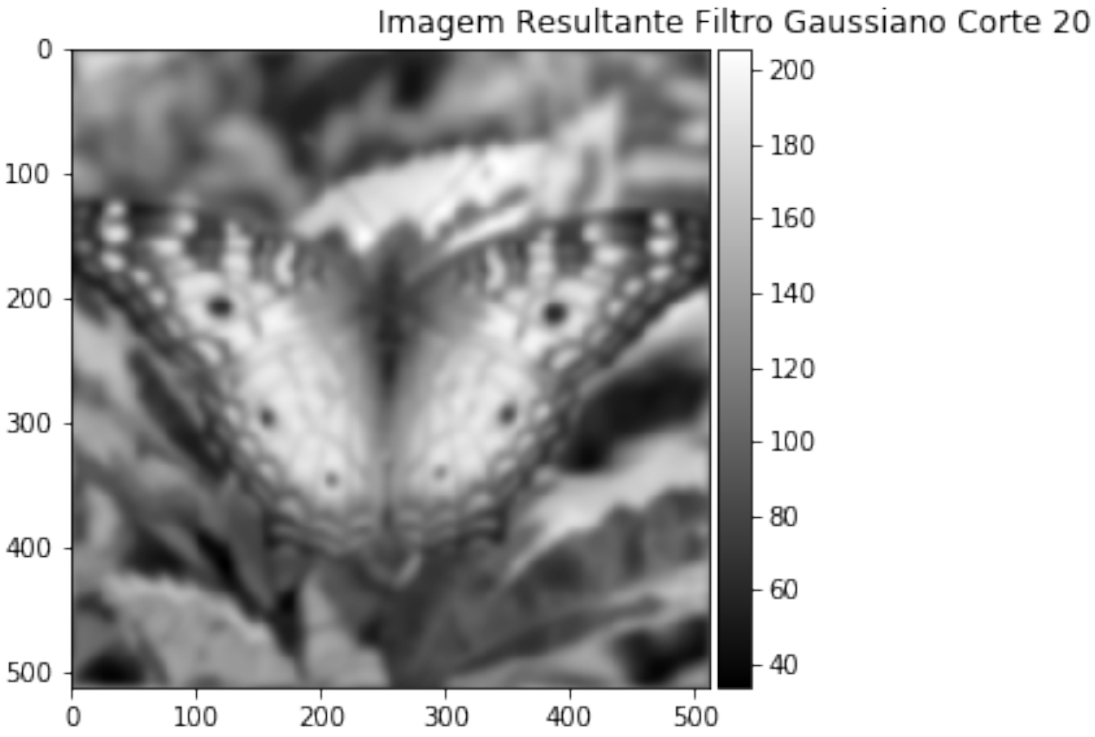
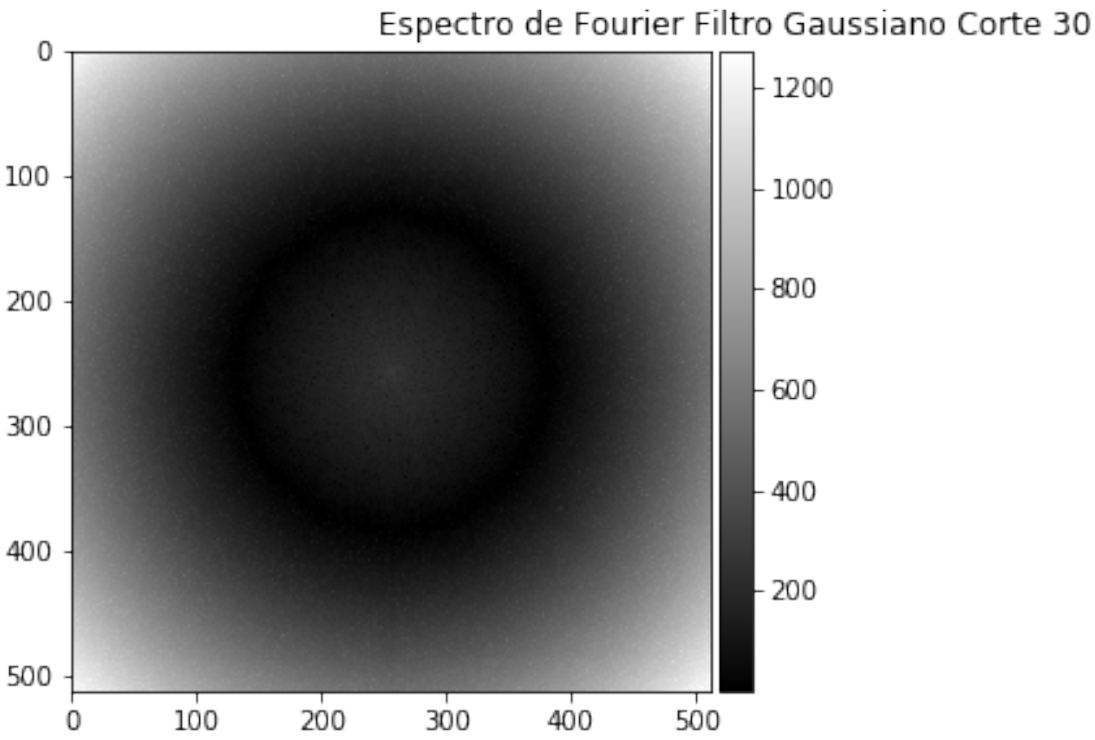


Imagem Resultante Filtro Gaussiano Corte 20
0
100
200
300
400
500
200
180
160
140
120
100
80
60
40


Espectro de Fourier Filtro Gaussiano Corte 30
0
100
200
300
400
500
1200
1000
800
600
400
200

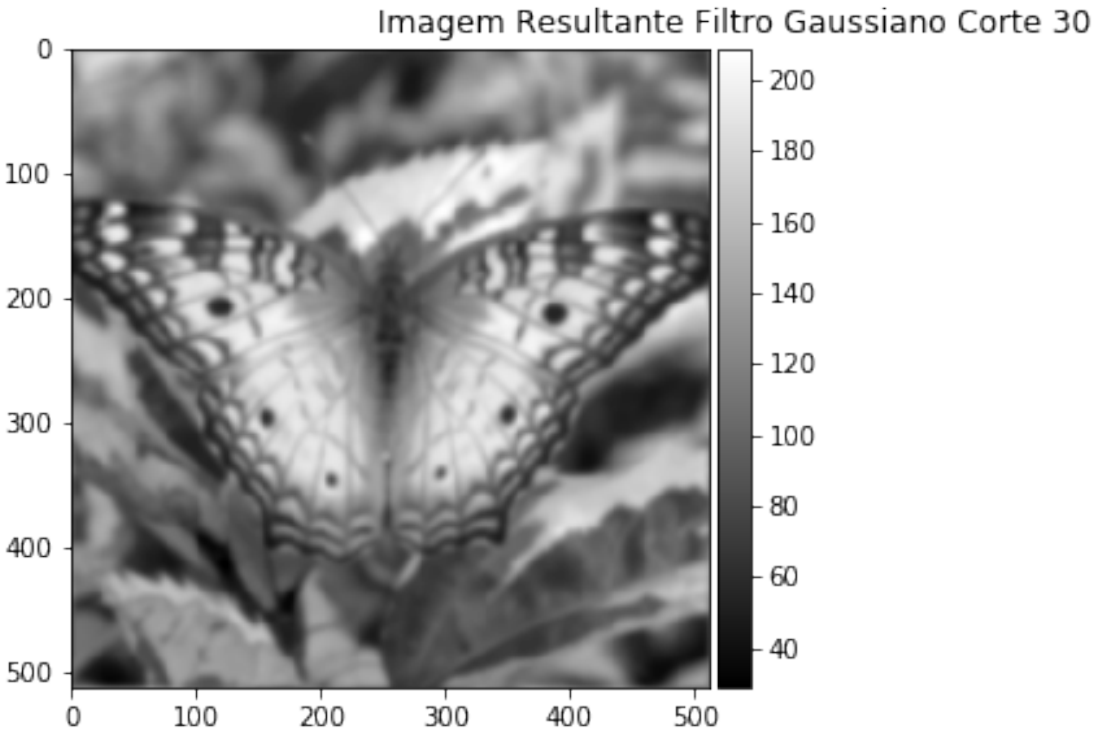
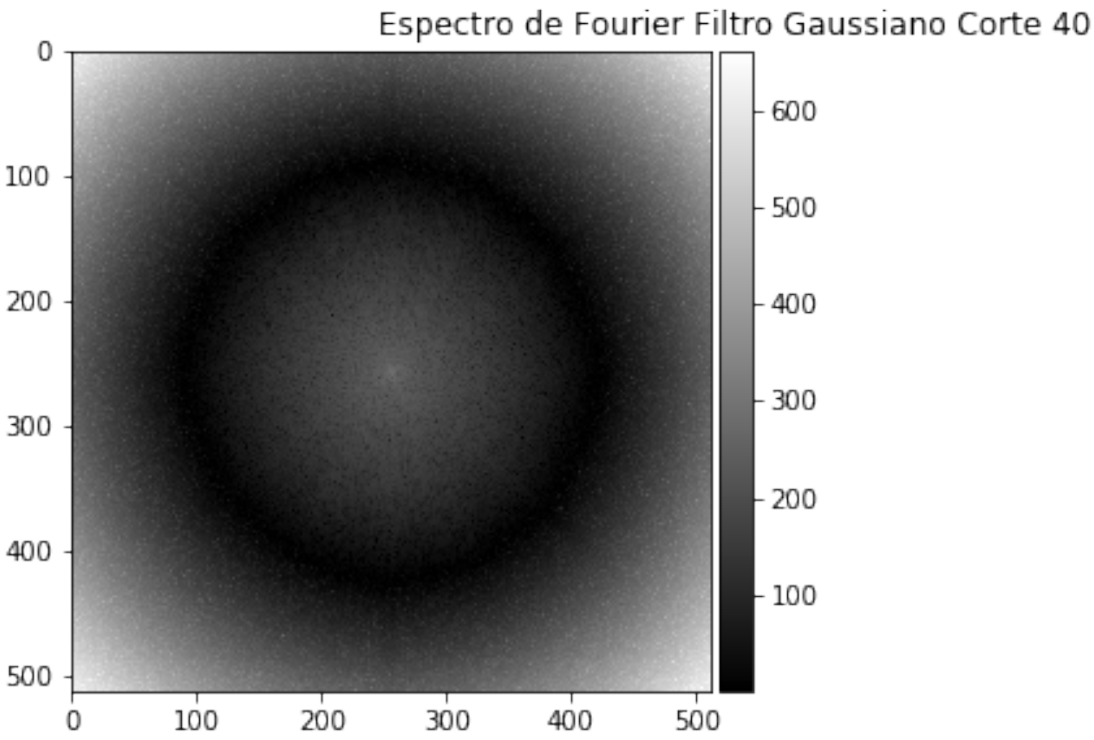


Imagem Resultante Filtro Gaussiano Corte 30
0
100
200
300
400
500
200
180
160
140
120
100
80
60
40


Espectro de Fourier Filtro Gaussiano Corte 40
0
100
200
300
400
500
600
500
400
300
200
100
0
100
200
300
400
500

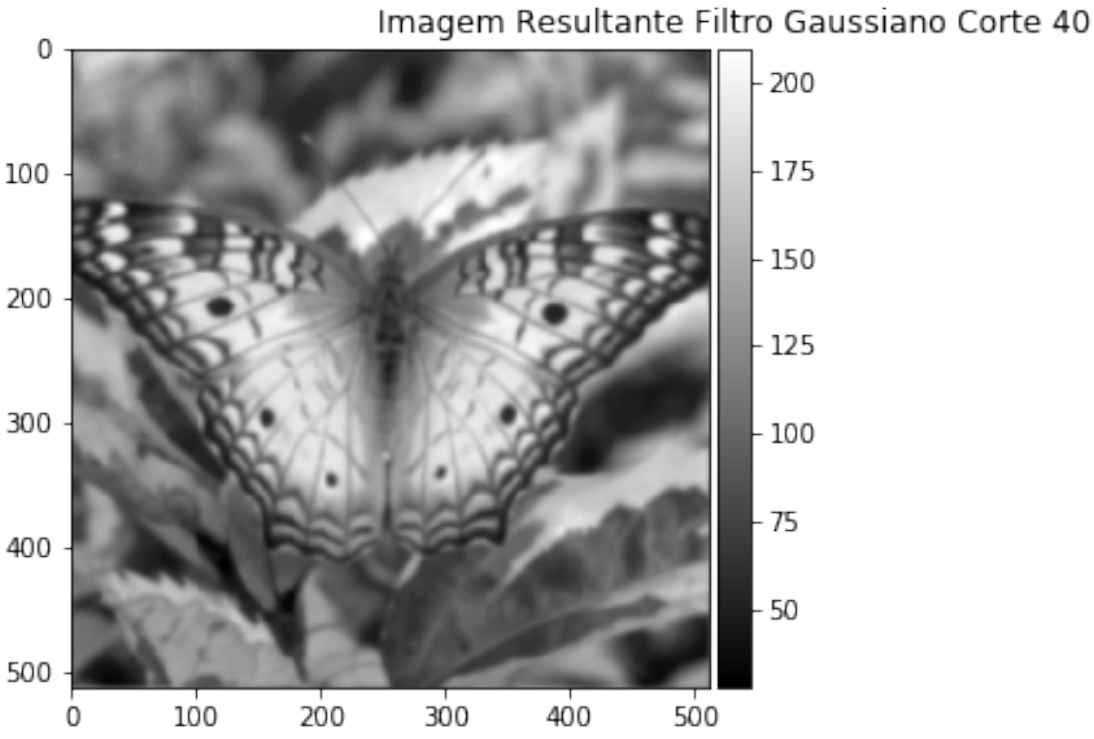
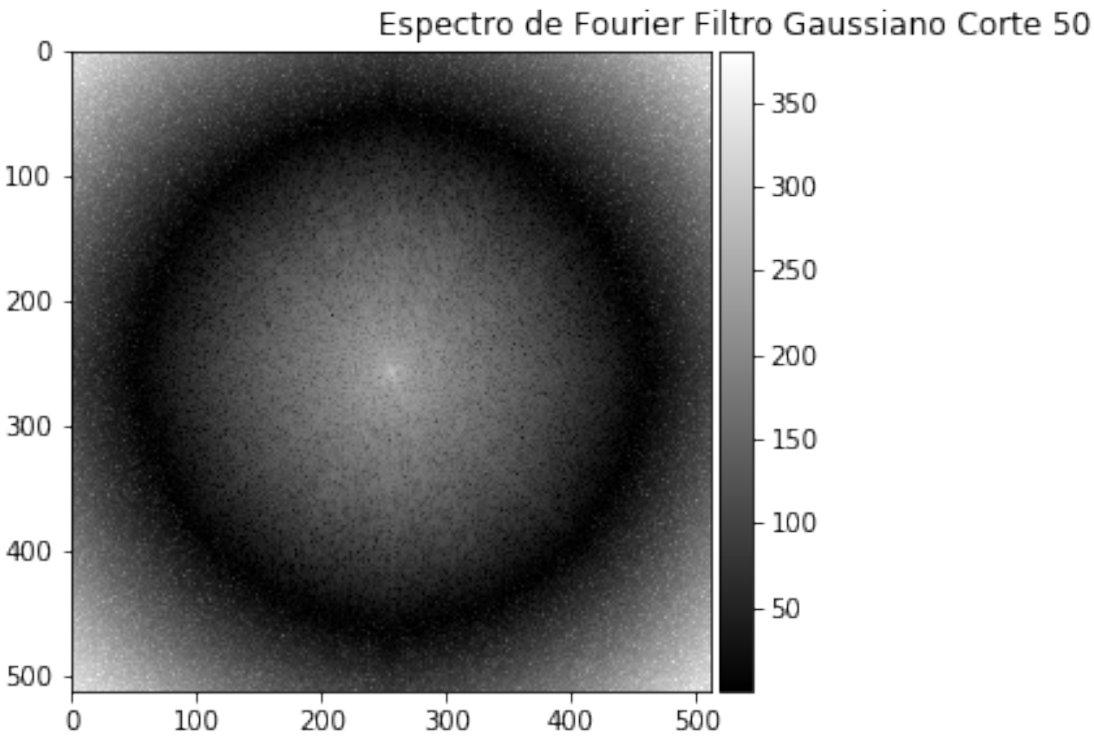


Imagem Resultante Filtro Gaussiano Corte 40
0
100
200
300
400
500
200
175
150
125
100
75
50
0
100
200
300
400
500


Espectro de Fourier Filtro Gaussiano Corte 50
0
100
200
300
400
500
350
300
250
200
150
100
50

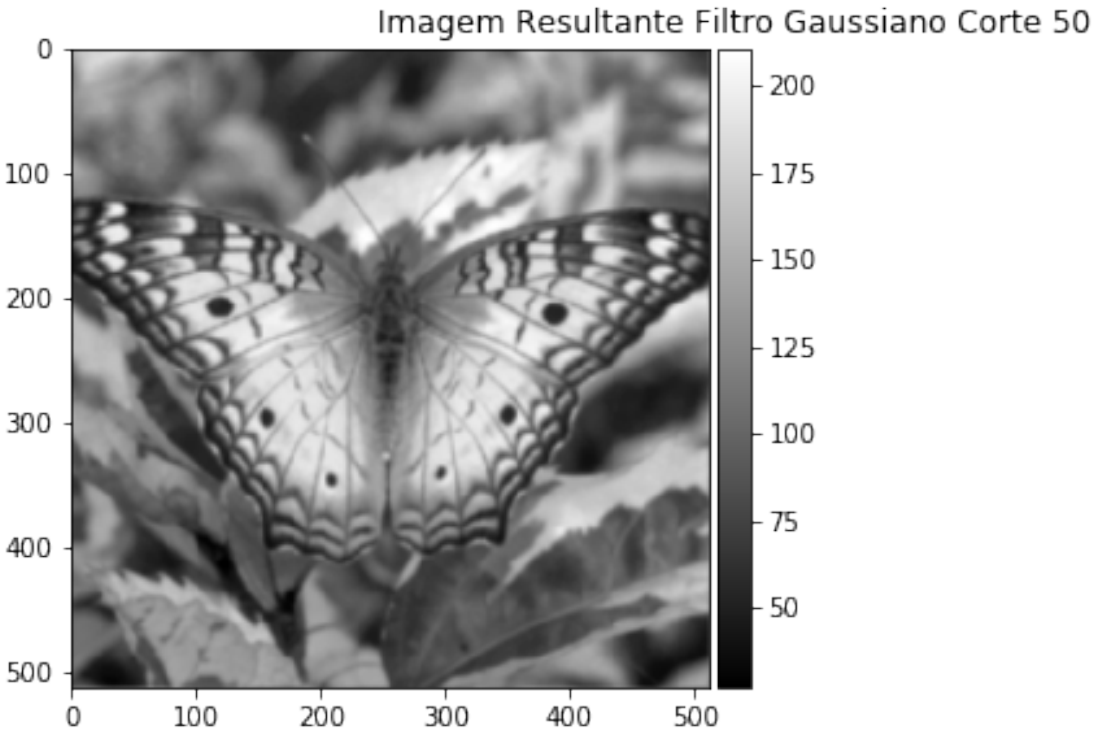
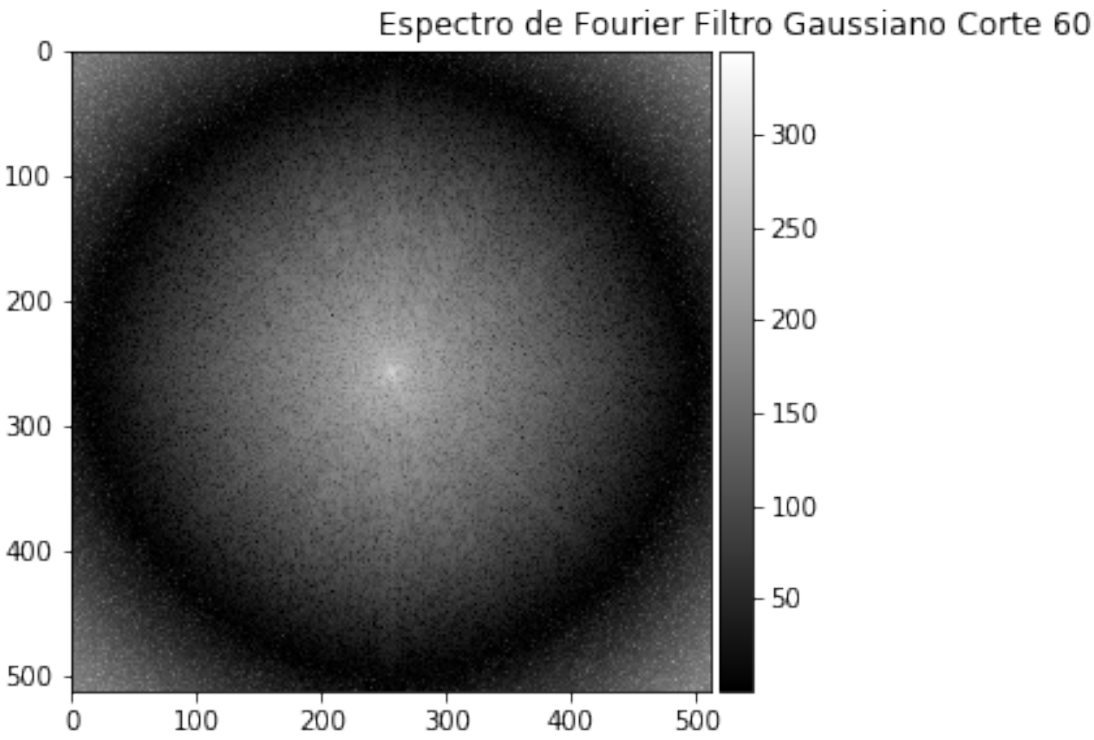


Imagem Resultante Filtro Gaussiano Corte 50
0
100
200
300
400
500
200
175
150
125
100
75
50


Espectro de Fourier Filtro Gaussiano Corte 60
0
100
200
300
400
500
0
100
200
300
400
500
300
250
200
150
100
50

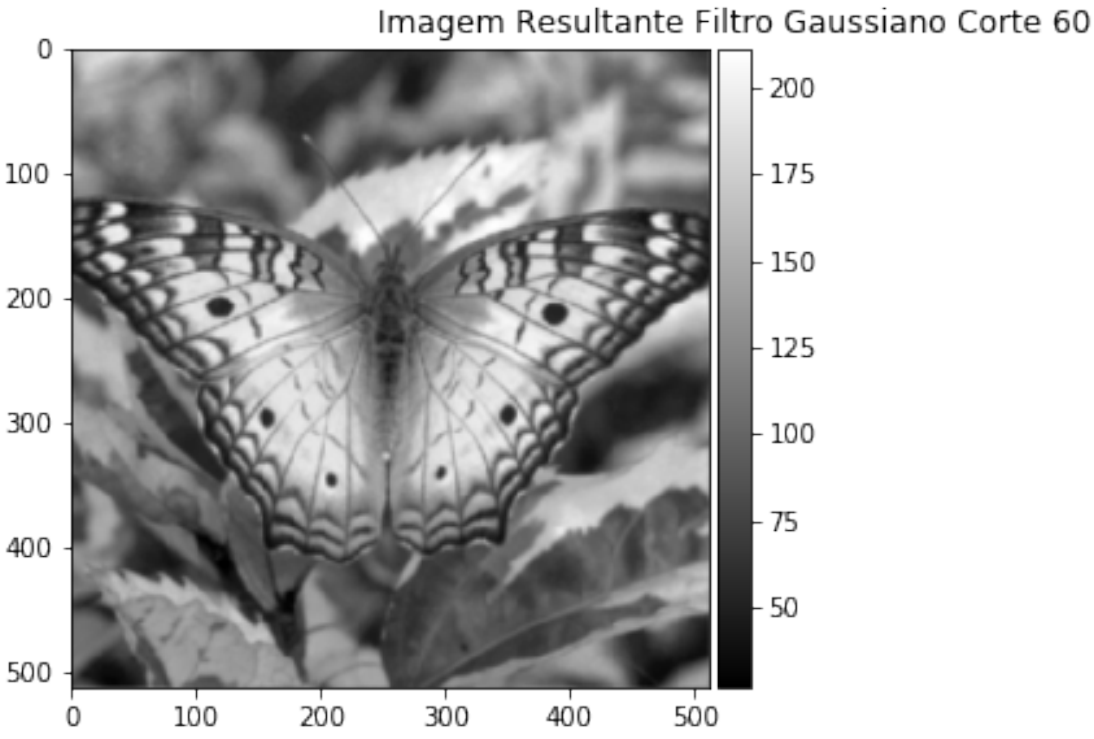


Imagem Resultante Filtro Gaussiano Corte 60
0
100
200
300
400
500
0
100
200
300
400
500
200
175
150
125
100
75
50

Como é possível observar dos resultados, quanto maior é a frequência de corte D_0, menos das altas frequências do espectro são removídas, portanto menor é o grau de suavização. Portanto podemos observar que a imagem se torna cada vez mais nítida conforme o valor de D_0 aumenta.

1.4 Conclusão

Nesse trabalho aprendemos o efeito de diferentes tipos de filtragem, tanto em domínio espacial quanto no domínio de frequências. Pudemos verificar o efeito de suavização do filtro Gaussiano, a sensibilidade à ruídos do filtro Laplaciano e a propriedade de identificação de bordas do filtro de Sobel.