Name: NGUYEN TRONG NGHIA

ID no: 217163

Homework submit:

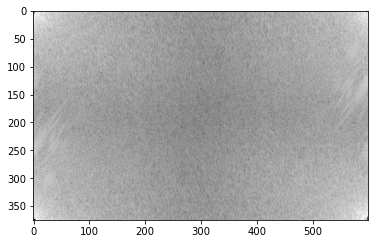
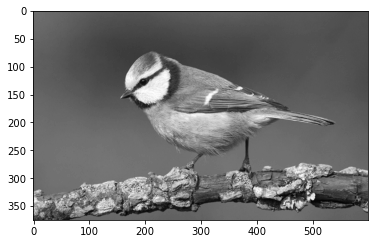
1. Select a gray scale image and present FFT results (before and after FFT Shift)

Choosing image:



**Fourier Transform:** Fourier Transformation transforms our image information - gray scaled pixels into frequencies. Because the digital image is discrete due to its incoherent pixels, we have two decision for Fourier transform: Discrete Fourier Transformation (DFT) and Fast Fourier transform (FFT).

* In this case, I choose FFT because it’s much faster, with the Numpy function **np.fft.fft2** for 2-D Fast Fourier transform
* Then, for visualizing, as we know, the Fourier transform result have two path: Spectrum and Phase angle. We usually use Spectrum for display
* Code:
* Result:

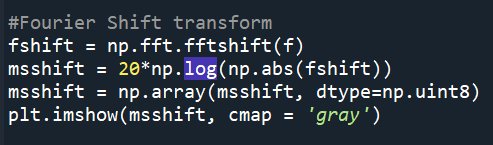


Magnitude spectrum

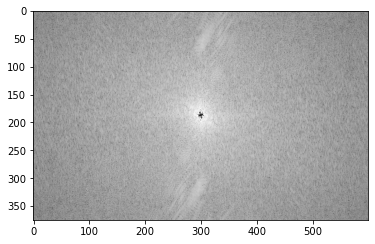
Original image

* The white area in the spectrum image show the high power of frequency. The corners in the spectrum image represent low frequencies.

**FFT Shift:** I could basically understand that FFT shift method will shift the zero-frequency component to the center of the spectrum which makes the spectrum image more visible without losing any piece of information also help us implement high/low-pass filter easily. I also use function **np.fft.fftshift** of the Numpy library for the output frequency f from FFT process.

* Code:

-

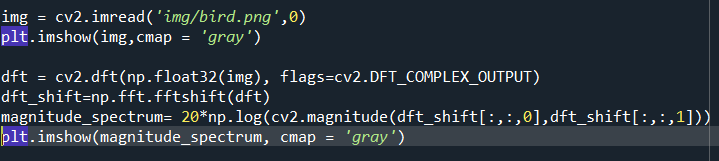
* Result:

Spectrum of Shift FFT process

* As we see, opposite to the FFT process, the Shift FFT process makes the low frequencies focus to the center and the high frequencies to the corner.

1. The select image is the same image as task 1

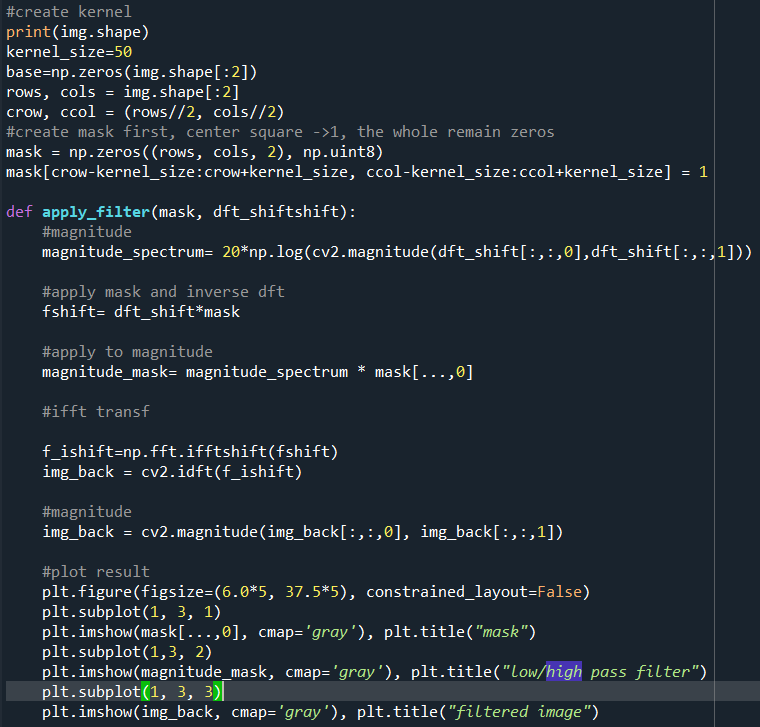
* First, read image and perform FFT, shift FFT:



* In this task, we will do the filter in frequency domain so I firstly want to:

+ Create a kernel to apply with all filter method on the frequency domain

+ Create a function to apply the filter and plot the results visually for all case. Code:



1. **Low Pass Filter:** In the kernel setting, I set the mask as low pass filter which only allow low frequencies to pass through so It only keep the low frequencies. We can test the function with lowpass Filter.

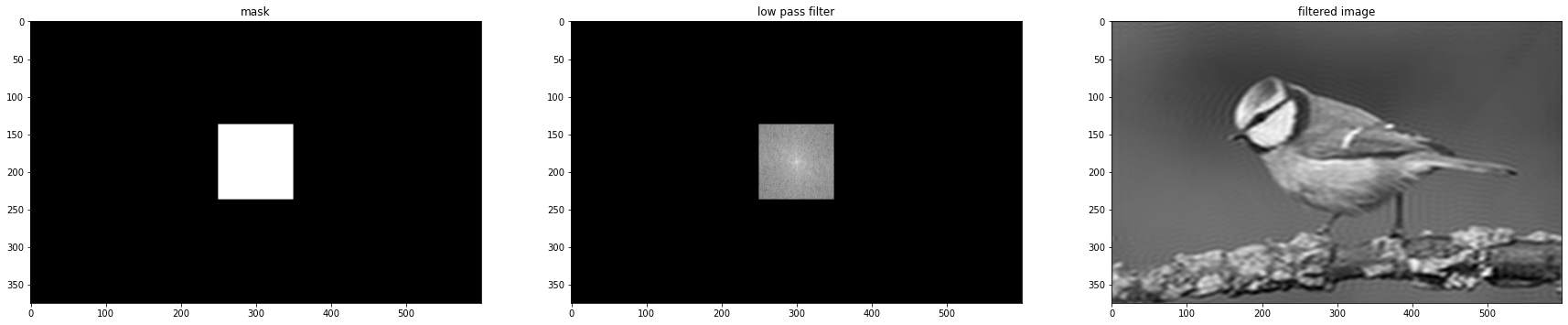
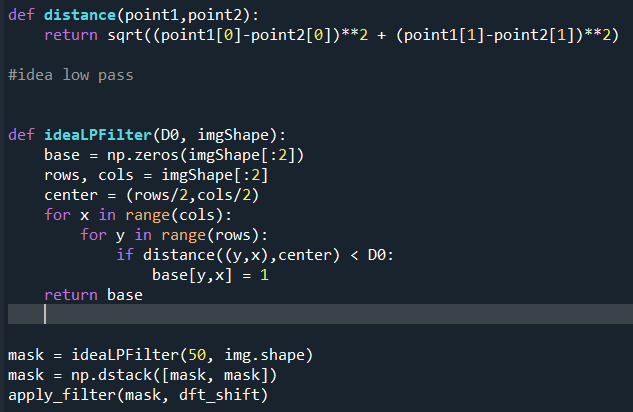
* Result:

Figure 1: **Low pass filter (LPF) result.** As we see, the output of low pass filter only allow low frequencies to pass through so the high frequencies contents such as noises are blocked. As a result, this make processed image has less noisy pixels. in images.

**Ideal Low Pass Filter:** This method will define a threshold D0. According to the Formula (1), low pass filter result H(u,v) equals to 1 under the threshold and equals to 0 when above the threshold.

**Formular (1):** Formula for ideal low pass filter where D₀ is a positive constant and D(u, v) is the distance between a point (u, v) in the frequency domain and the center of the frequency rectangle

* Code:



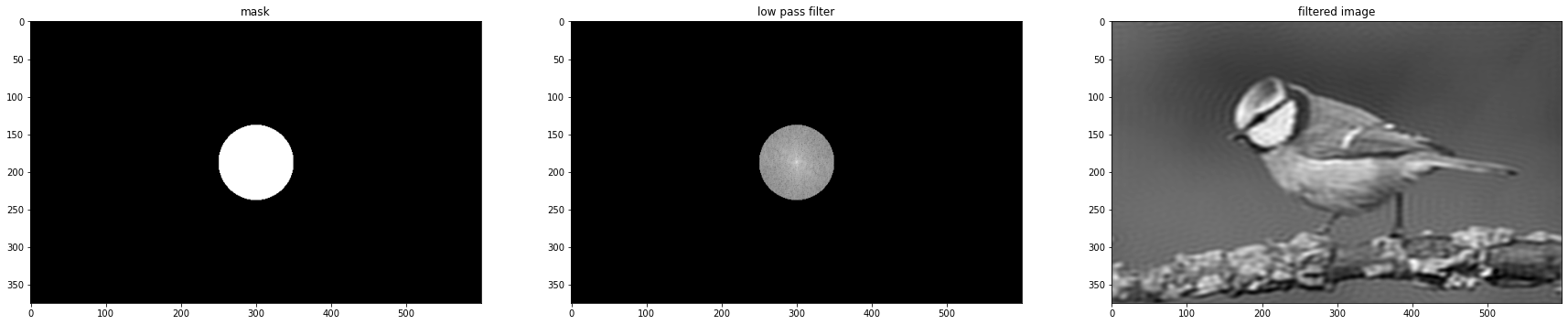
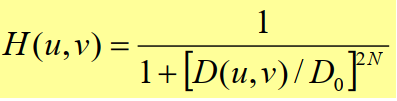
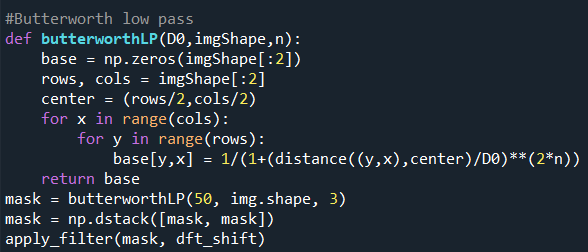
* Result (set D0 = 50):

Figure 1: **Ideal Low pass filter (ILPF) result.** The obtained result from this method help to reduce the ringing effect of the input image compare to the **LPF** method. However, It still remained a lot of noise because this filter blocks all information that is outside of certain radius from origin point. As a result, some information will be discontinued sharply without any smooth out.

**Butterworth Low Pass Filter (BLPF):** As same as ILPF, this method aim to reduce ringing effect but Its formula provides a parameter **N** which affects the clearness of the cutoff between passed and filtered frequencies

**Formular (2):** Formula for Butterworth low pass filter where D0 is a positive constant and D(u, v) is the distance between a point (u, v) in the frequency domain and the center of the frequency rectangle

* Code:

-

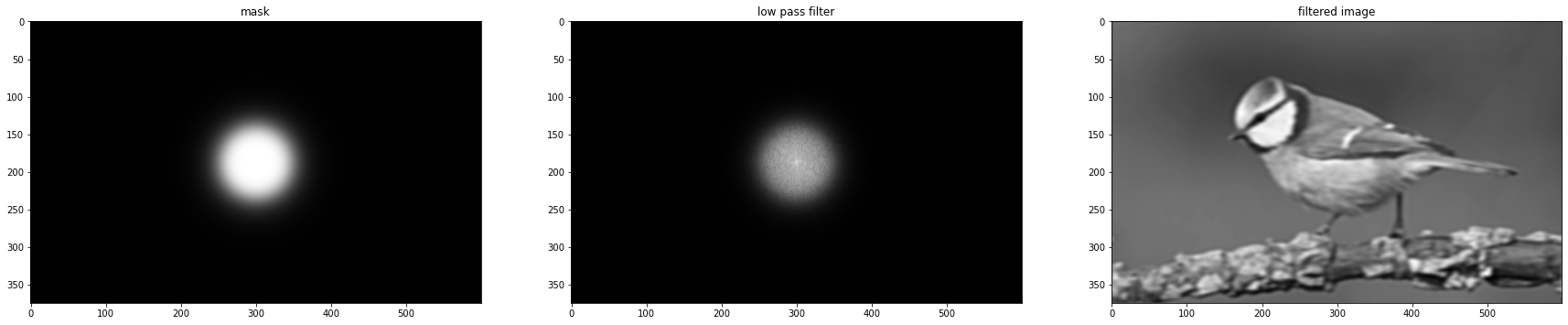
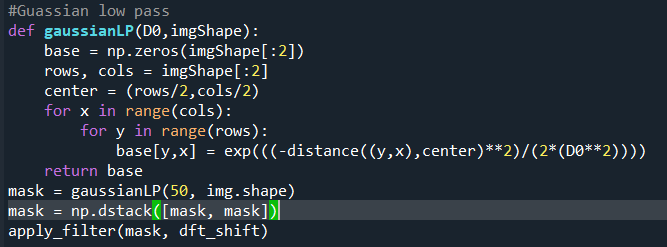
* Result(D0 = 50, n = 3):

Figure 2: **Butterworth Low pass filter (ILPF) result.** By this method, the ringing effect has been significantly reduced, almost disappear from the result image.

**Gaussian low pass filter:** Gaussian filter is a smoother cutoff version than Butterworth. The cutoff between passed and filtered frequencies is very blurry which leads to smoother processed images.

**Formular (3):** Formula for Gaussian low pass filter where D₀ is a positive constant and D(u, v) is the distance between a point (u, v) in the frequency domain and the center of the frequency rectangle.

* Code:



* Result (D0 = 50):

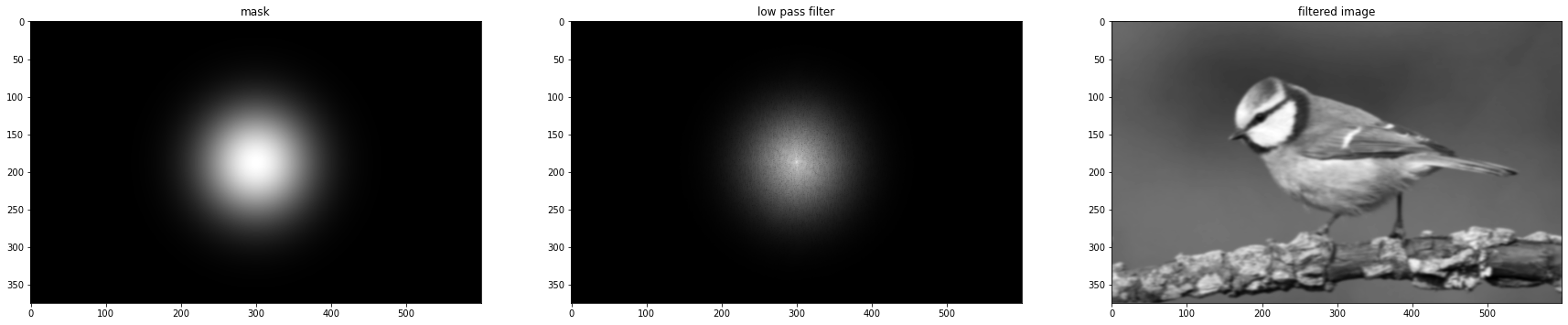
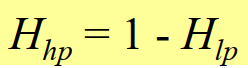
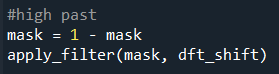


Figure 3: **Guassian Low pass filter (ILPF) result.** Gaussian filter are better smoothly blocking information that is outside of certain radius from origin point which makes image more smoothly with less distortion.

1. **High Pass Filter**

Opposite to low pass filter, high pass filter (HPF) only allow high frequencies to pass through. Therefore, the result of this process captures the edges in image which could be used to sharpen the original image with proper overlap calculation. This will enhance sharpness in original image making edges more clear.

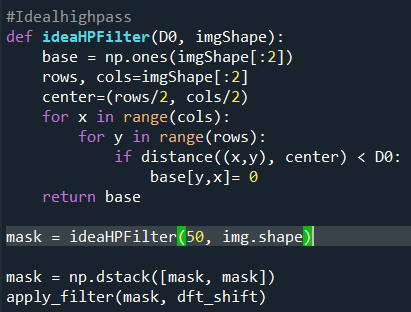
High pass filter formula. We can configure the parament to apply to the kernel for display high pass filter by console :



**Idea High Pass Filter (IHPF)**: idea high pass filter has H(u, v) equals to 0 under threshold, and H(u, v) equals to 1 when above the threshold.

****

* Code:



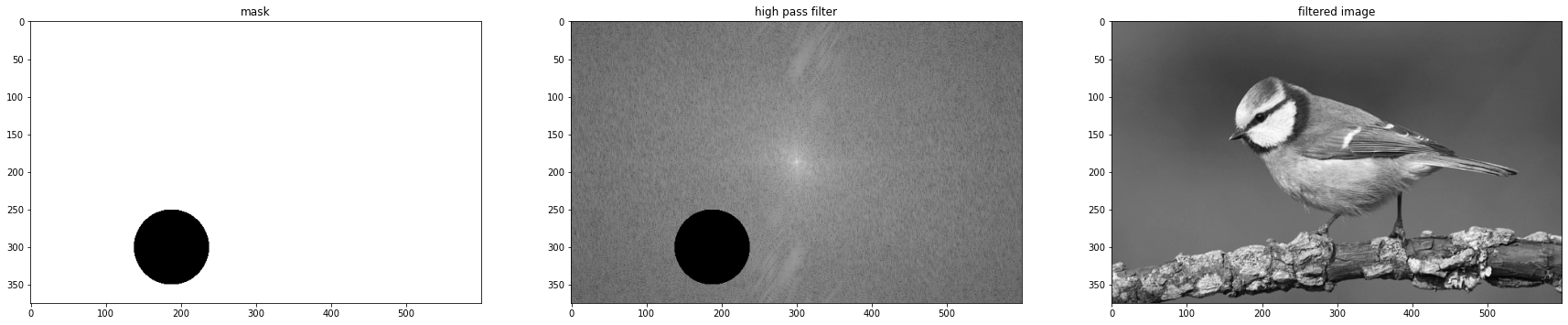
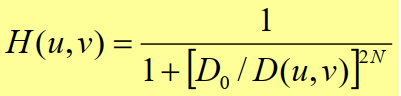
* Result (D0 = 50):

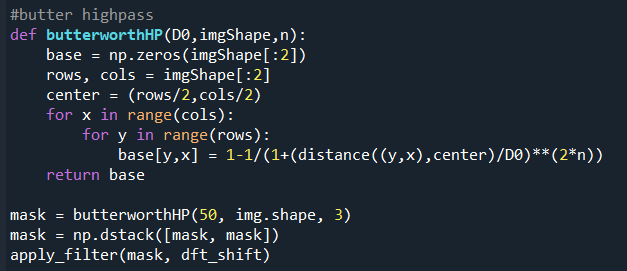
Figure 4: **Ideal High pass filter (IHPF) result.** The filter try to make the image sharpened but the result is too bad.

**Butterworth High Pass Filter:** As same as BLPF, this method also use a parameter N.



**Formular (4):** Formula for Butterworth high pass filter where D₀ is a positive constant and D(u, v) is the distance between a point (u, v) in the frequency domain and the center of the frequency rectangle

* Code:



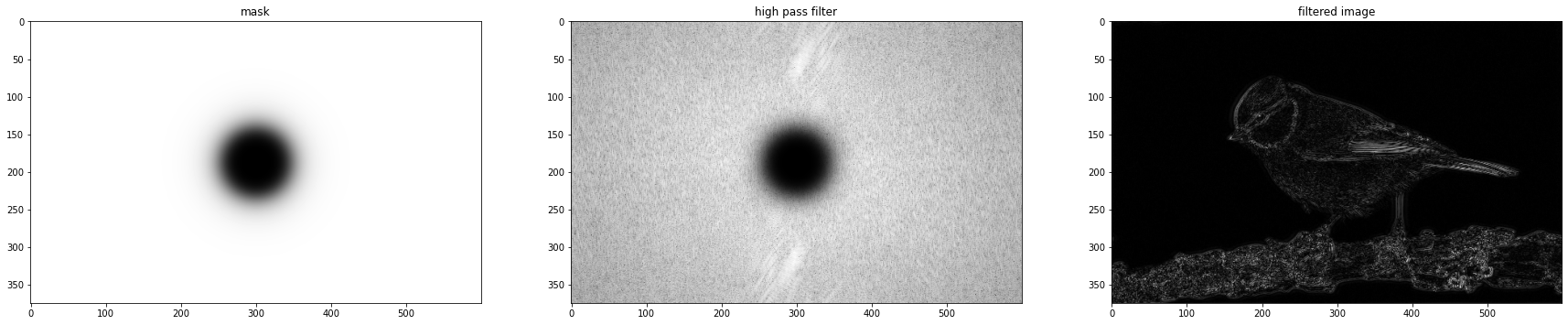
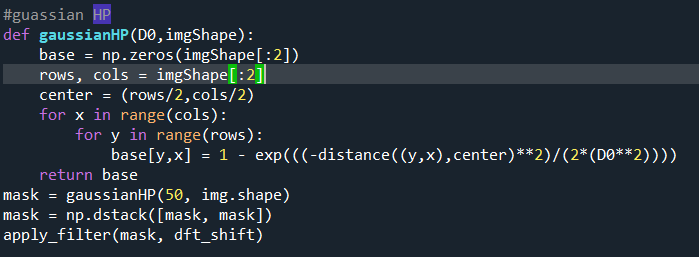
* Result:

Figure 5: **Butterworth High pass filter (IHPF) result.** This result seem to be better than the ideal method’s result thanks to smoothly blocking information that is outside of certain radius from origin point which makes image more smoothly with less distortion.

 **Gaussian High Pass Filter:**

**Formula (5): Formula for Gaussian high pass filter**

* Code:



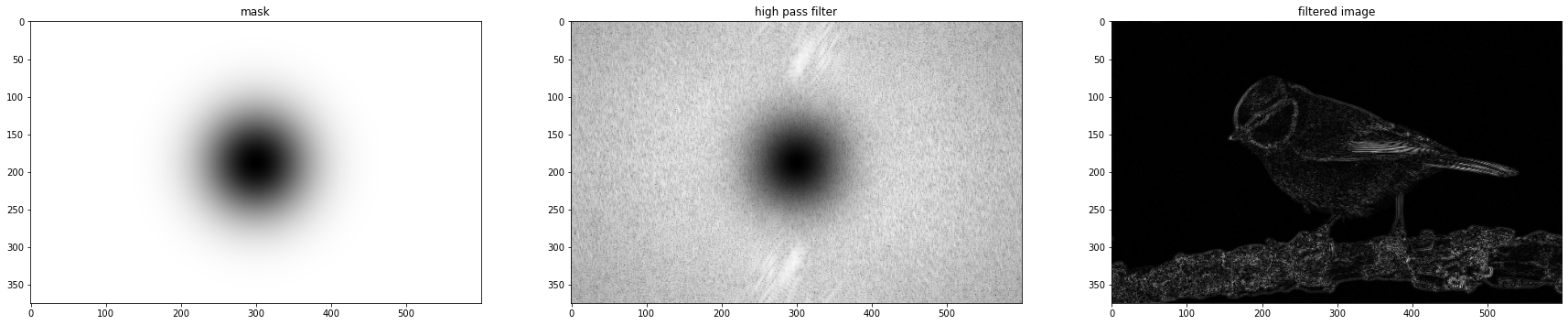
* Result:

Figure 6: **Gaussian High pass filter (GHPF) result.** It could be notice that this filter performed better than other two filters due to low distortion. There are a lot of distortions in an ideal filter result when compares to a Butterworth filter and a Gaussian filter.

Full code:

