Ex2_part5

May 3, 2022

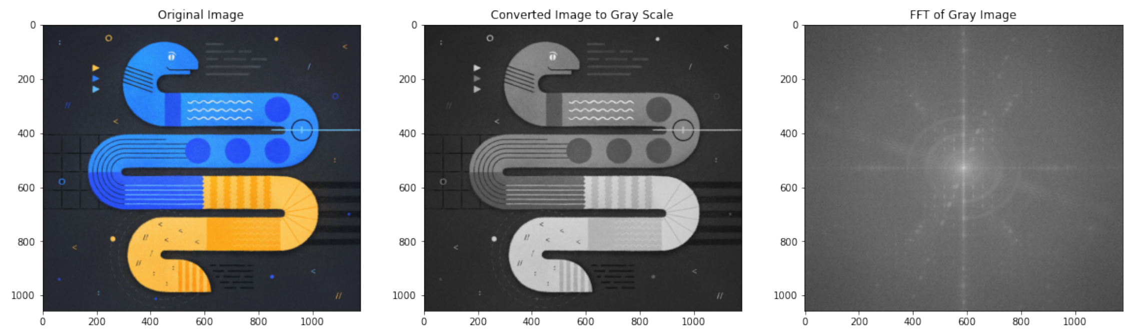
```
[ ]: # Done by Munther Odeh and Timo Marks
     import numpy as np
     from matplotlib import pyplot as plt
     from PIL import Image
     from skimage.color import rgb2gray
```

```
[ ]: # Function for converting image to gray scale -> fft of gray image -> Plot
     def fft_of_image(filepath):
         fig = plt.figure(figsize=(20, 20))
         img = plt.imread(filepath)
         gray_img = rgb2gray(img)
         fig1 = fig.add_subplot(1,3,1)
         fig1.imshow(img)
         fig1.title.set_text('Original Image')

         fig2 = fig.add_subplot(1,3,2)
         fig2.imshow(gray_img, cmap="gray")
         fig2.title.set_text('Converted Image to Gray Scale')

         # Fouriertransformation
         gray_img_fft = np.fft.fftshift(np.fft.fft2(gray_img))
         fig3 = fig.add_subplot(1,3,3)
         fig3.imshow(np.log(abs(gray_img_fft)), cmap="gray")
         fig3.title.set_text('FFT of Gray Image')
```

```
[ ]: fft_of_image('python-hero.jpg')
```



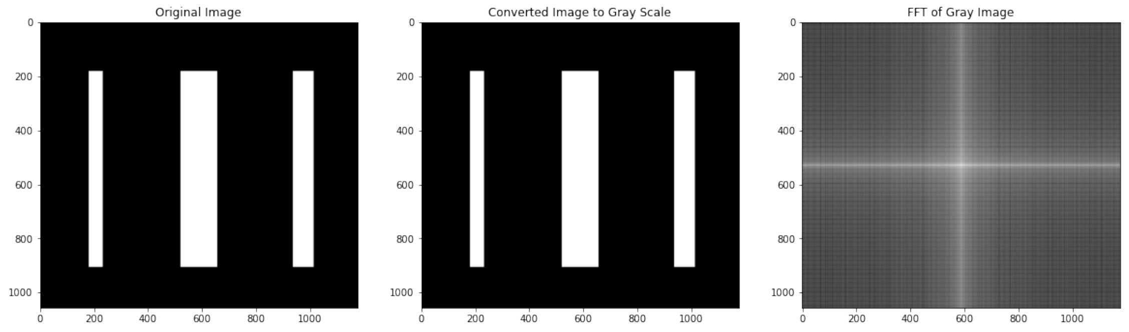
The plots show the absolute value of the frequency components in the image

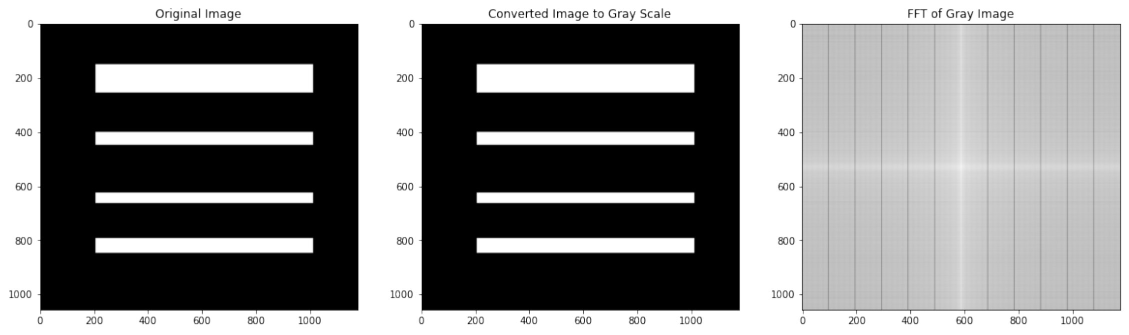
The Python image has a rather complex fourier transformation with bright lines on the main horizontal and vertical axis and a very bright sport in the middle. But we can also see some circular shapes and other radial lines to the outside.

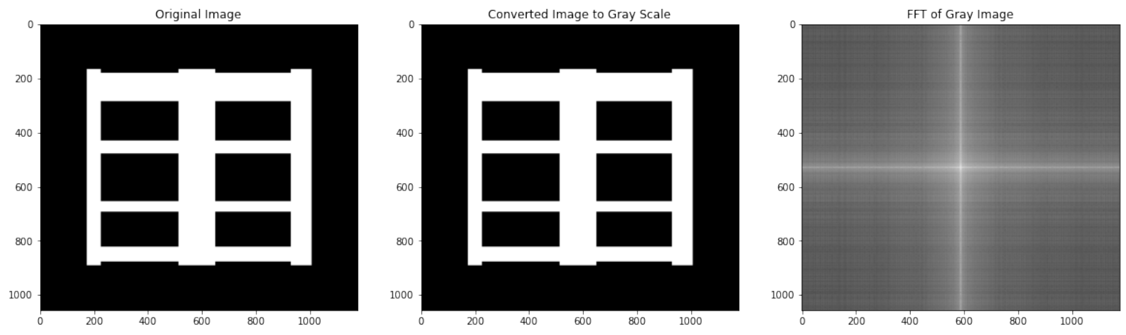
The fourier transformations of the less complex images (with vertical and horizontal white rectangles) have a distinct grid like pattern. In the fourier transformation of the vertical rectangles, we have some vertical dark lines, which means that these frequencies do not occur in the image. They are always a multiple of one another. If we combine both images (Both.png) we also see a combination of both fourier transformations.

One thing to be noted: The vertical/horizontal images have more higher frequency components than the Python image because we need more high frequency components to "create" the sharp edges of the rectangles in the image. We also have these sharp edges/high frequency components in the Python image but not as pronounced.

[ ]:
```
fft_of_image('Horizontal.png')
fft_of_image('Vertical.png')
fft_of_image('Both.png')
```






Original Image
Converted Image to Gray Scale
FFT of Gray Image
0
200
400
600
800
1000