PRACTICAL - 3

Name: Shivanshu Anant Suryakar

PRN: 1841048

Batch: B3

Class: L.Y Computer Engineering

Aim: Program for digital Image Processing using Fourier Transform in Python

---

Theory:

2D Fourier transform

Let f(x,y)f(x,y) denote an M×NM×N image, for x=0,1,2,…,M−1x=0,1,2,…,M−1 (columns) and y=0,1,2,…,N−1y=0,1,2,…,N−1 (rows).
The 2D **discrete Fourier Transform** (DFT) of ff, denoted by F(m,n)F(m,n), is given by
F(m,n)=1MN∑x=0M−1∑y=0N−1f(x,y)exp(−2πi(xMm+yNn)),F(m,n)=1MN∑x=0M−1∑y=0N−1f(x,y)exp (−2πi(xMm+yNn)),

for m=0,1,2,…,M−1m=0,1,2,…,M−1 and n=0,1,2,…,N−1n=0,1,2,…,N−1.
The M×NM×N rectangular region defined for (m,n)(m,n) is called the **frequency domain**, and the values of F(m,n)F(m,n) are called the **Fourier coefficients**.

The **inverse discrete Fourier transform** is given by
f(x,y)=MN∑m=0M−1∑n=0N−1F(m,n)exp(2πi(mMx+nNy)),f(x,y)=MN∑m=0M−1∑n=0N−1F(m,n)exp (2πi(mMx+nNy)),

for x=0,1,2,…,M−1x=0,1,2,…,M−1 and y=0,1,2,…,N−1y=0,1,2,…,N−1.
Remark. There are other ways of normalizing the DFT and its inverse through the factors 1/MN1/MN and MNMN. The one we have chosen is the usual in Python.
Even if f(x,y)f(x,y) is real, its transform F(m,n)F(m,n) is in general a complex function. For visualization, we use the **power spectrum**,
P(m,n)=F(m,n)F¯(m,n),P(m,n)=F(m,n)F¯(m,n),

where $\bar{F}\bar{F}$ denotes the complex conjugate of FF.

Property: If the image is periodic, that is, it takes the same values in the top and bottom edges, and in the left and right edges, then, the DFT is also periodic, and the power spectrum is symmetric about the origin:
$P(m,n)=P(-m,-n).P(m,n)=P(-m,-n).$

If the image is not periodic, then FF may have discontinuities. This is known as **the Gibbs phenomenon.**
Thanks to the above property, we may represent the DFT in a square centered at (0,0)(0,0), which is more convenient for visualization porposes.

The DFT and its inverse are obtained in practice using a **fast Fourier Transform**. **In Matlab**, this is done using the command |fft2|: |F=fft2(f)|.
To compute the power spectrum, we use the **Matlab** function |abs|: |P=abs(F)^2|. If we want to move the origing of the transform to the center of the frequency rectangle, we use |Fc=fftshift(F)|. Finally, if we want to enhance the result, we use a loglog scale.

Application: image compression

An image is recovered from the set of coefficients FF by using the inverse Fourier Transform. As we have seen, in general, not all the coefficients have the same value. In order to use only the most important, we may select those that are larger in modulus than some given *threshold* TT, i.e. we set to zero all the coefficients but those contained in the following set:
$ST=\{0\le m\le M-1,0\le n\le N-1:|F(m,n)|\ge T\}.ST=\{0\le m\le M-1,0\le n\le N-1:|F(m,n)|\ge T\}.$

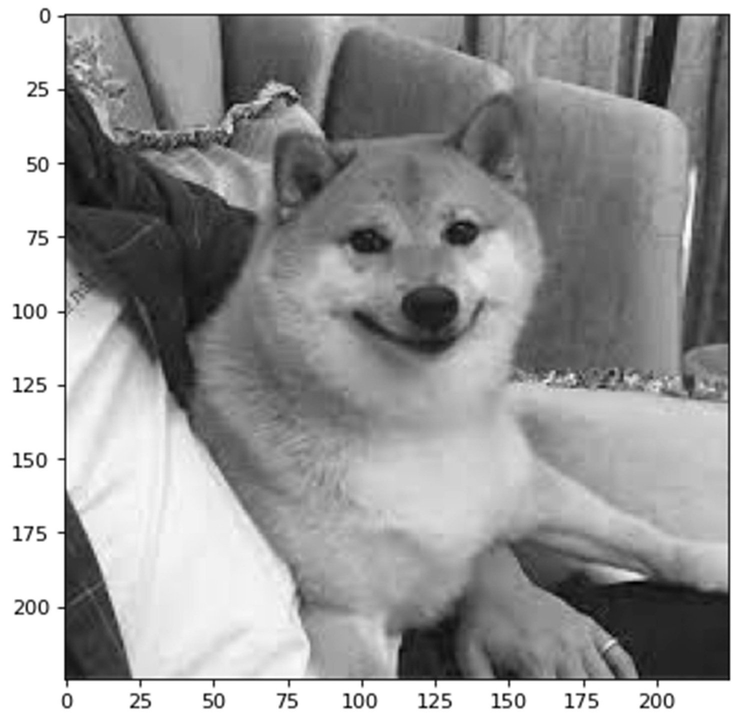
Let us denote by $g(T)g(T)$ the number of indices, (m,n)(m,n), in STST. One one hand, for $T=0T=0$ we have $g(0)=MNg(0)=MN$, since $|F(m,n)||F(m,n)|$ is always non-negative. On the other hand, for $T>\max|F(m,n)|T>\max|F(m,n)|$ we obviously have $g(T)=0g(T)=0$. Observe that gg is a decreasing function of TT. We construct here a plot of gg using the command |nnz|, which gives the number of non zero elements of a matrix.

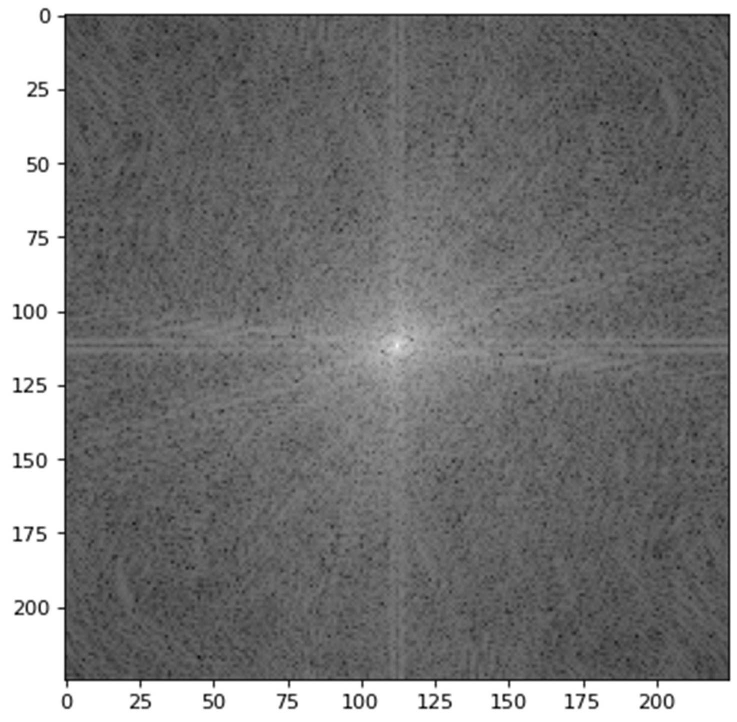
Requirements:

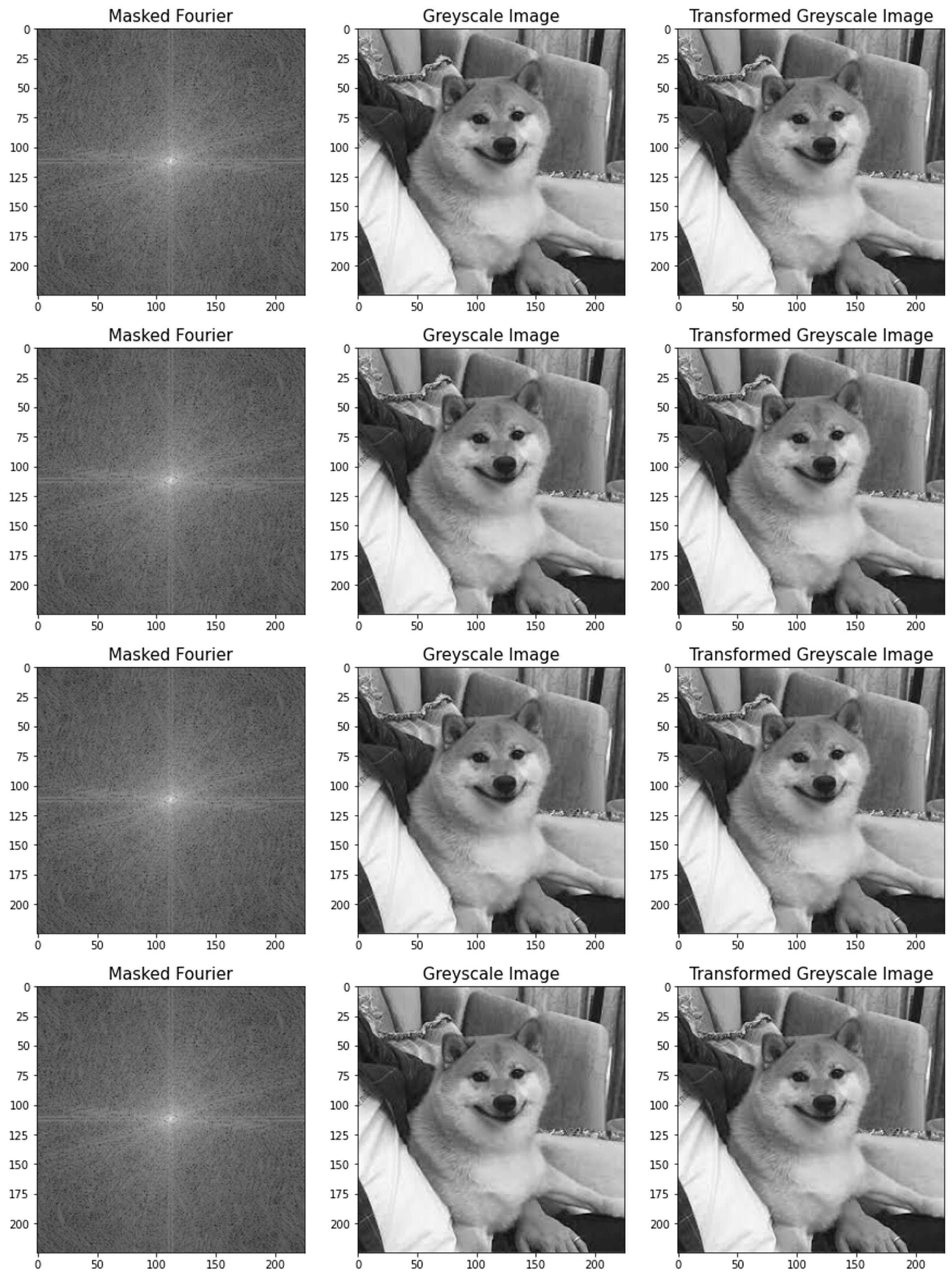
- numpy

- matplotlib
- skimage

Output:






Masked Fourier
Greyscale Image
Transformed Greyscale Image
Masked Fourier
Greyscale Image
Transformed Greyscale Image
Masked Fourier
Greyscale Image
Transformed Greyscale Image
Masked Fourier
Greyscale Image
Transformed Greyscale Image

Conclusion:

In this program we have studied digital Image Processing using Fourier Transform in Python