Challenge 2

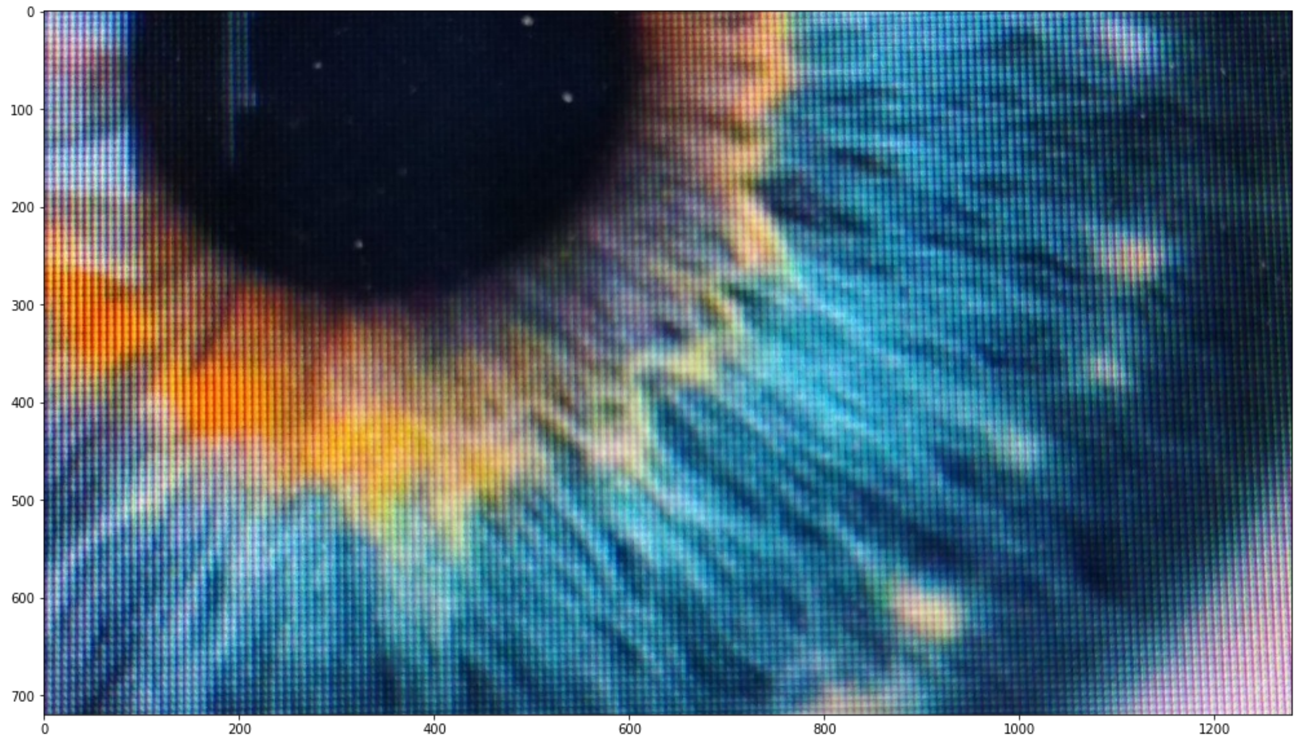
Xabier Oyanguren Asua 1456628

We first import the close take of the laptop screen with a colorful image.

In [7]:

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

image = cv2.imread("eye.jpeg")
image = np.flip(image, axis=2) # change from bgr to rgb
print(image.shape)
fig, ax = plt.subplots(1,1, figsize=(30,10))
ax.imshow(image)
plt.show()
```

```
(720, 1280, 3)
```

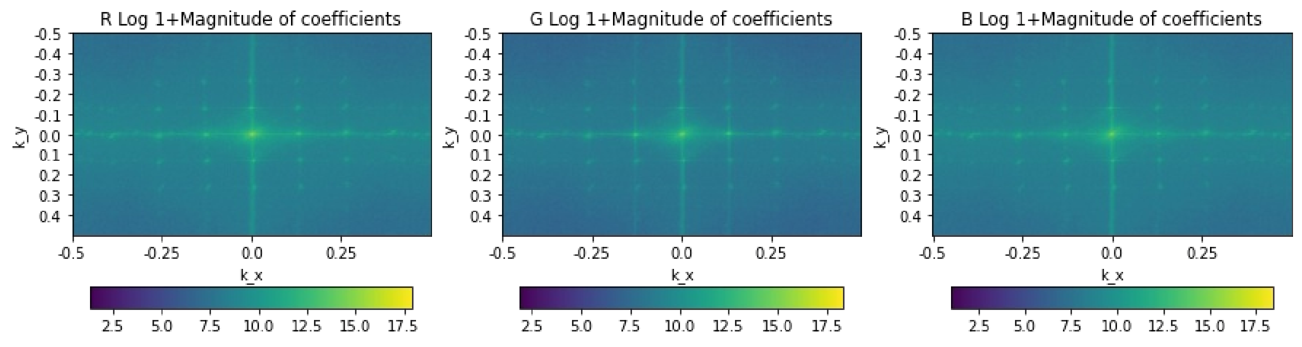


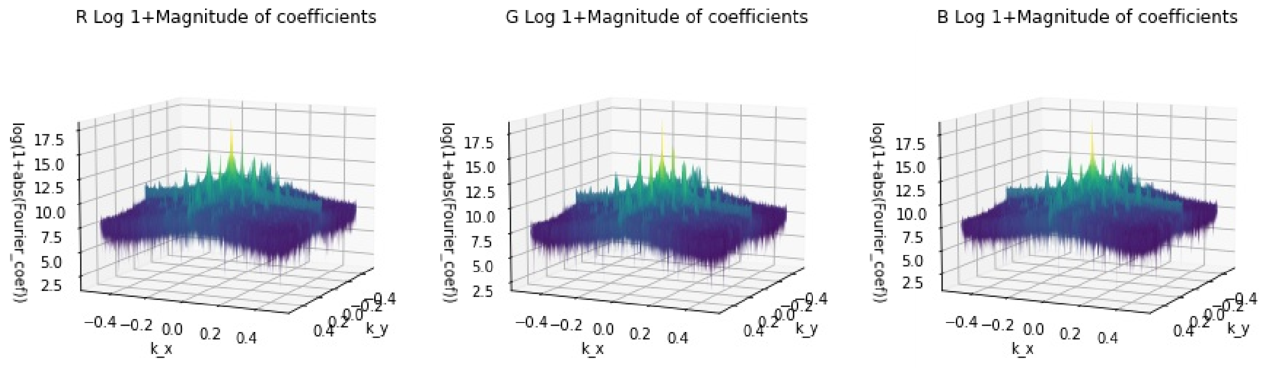
We can clearly see high frequency periodic artefacts probably due to the discrete nature of the monitor's light sources.

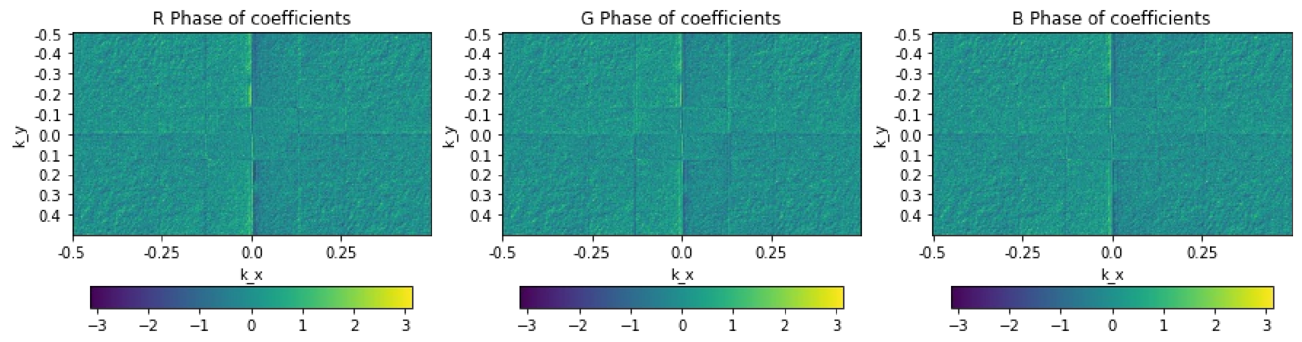
We convert it to Fourier space and look at the frequency distribution of the magnitude of the Fourier coefficients.

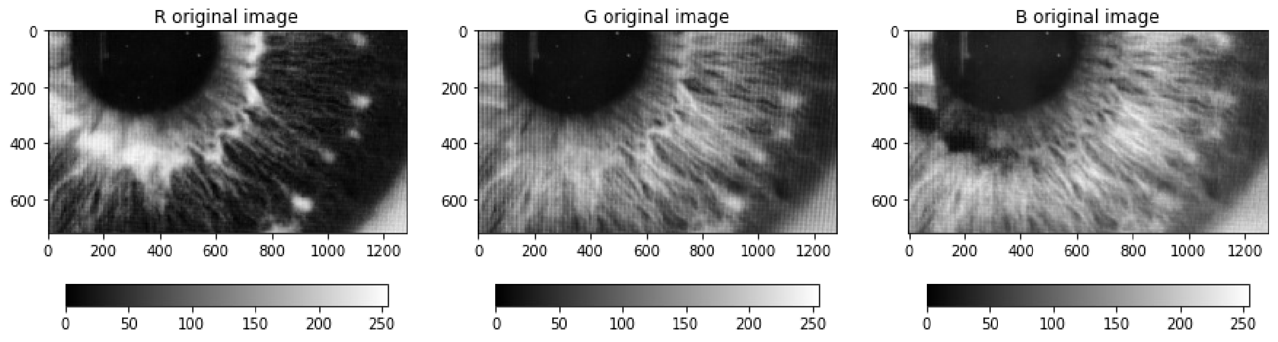
In [2]:

```python
from scipy.fft import fft2, fftshift, ifft2
from scipy.signal import convolve2d

from scipy.fft import fft2, ifft2, rfft2, fftshift

sample_ratex, sample_ratey =1, 1 # discrete samples of the underlying function per pixel in x and y

# number of total signal points and information channels
Ny, Nx, C = image.shape # h, w, C

# sample spacing is sample_rate smaples per second-> pixels between samples is:
Tx, Ty = 1.0 / sample_ratex, 1.0 / sample_ratey

pixelsx, pixelsy = np.linspace(0.0, Nx*Tx, Nx), np.linspace(0.0, Ny*Ty, Ny)
frecsx, frecsy = np.linspace(-1.0/(2.0*Tx), 1.0/(2.0*Ty), Nx), np.linspace(-1.0/(2.0*Ty), 1.0/(2.0*Ty), Ny)


coefs = fft2(image, axes=(0,1))
coefs = fftshift(coefs, axes=(0,1))


def plot_fourier2d(image, coefs, frecsx, frecsy, plot3d_resolution=0.3):
    Ny, Nx, C = image.shape # h, w, C
    fig, ax = plt.subplots(4,C, figsize=(C*5, 20))
    ffx, ffy = np.meshgrid(frecsx, frecsy)
    if C==1:
        cols = ['Gray']
    elif C==3:
        cols = ['R', 'G', 'B']
    else:
        cols = ['Channel'+str(i) for i in range(C)]
    for c, col in enumerate(cols):
        cmap1 = ax[0, c].imshow(np.log(1+np.abs(coefs[:,:,c])))
        ax[0,c].set_xticks(np.arange(0,Nx,Nx//4), np.round(frecsx[::Nx//4], 2))
        ax[0,c].set_yticks(np.arange(0,Ny,Ny//10), np.round(frecsy[::Ny//10], 2))
        ax[0,c].set_title(f"{col} Log 1+Magnitude of coefficients")
        ax[0,c].set_xlabel("k_x")
        ax[0,c].set_ylabel("k_y")

        ax[1,c].set_visible(False)
        axp = fig.add_subplot(int(f"{4}{C}{C+c+1}"), projection='3d')
        axp.plot_surface(ffx, ffy, np.log(1+np.abs(coefs[:,:,c])),
                         rcount=int(Ny*plot3d_resolution), ccount=int(Nx*plot3d_resolution), cmap='viridis') # rst
ride=1, cstride=1, linewidth=0
        axp.set_xlabel('k_y')
        axp.set_ylabel('k_x')
        axp.set_zlabel('log(1+abs(Fourier_coef))')
        #ax.set_zlim(-0.078*np.max(im), np.max(im))
        axp.set_title(f"{col} Log 1+Magnitude of coefficients")
        axp.view_init(10, 25)

        cmap2 = ax[2,c].imshow(np.arctan2(np.imag(coefs[:,:,c]), np.real(coefs[:,:,c])))
        ax[2,c].set_xticks(np.arange(0,Nx,Nx//4), np.round(frecsx[::Nx//4], 2))
        ax[2,c].set_yticks(np.arange(0,Ny,Ny//10), np.round(frecsy[::Ny//10], 2))
        ax[2,c].set_title(f"{col} Phase of coefficients")
        ax[2,c].set_xlabel("k_x")
        ax[2,c].set_ylabel("k_y")
        cmap3 = ax[3,c].imshow(image[:,:,c], cmap='gray')
        ax[3,c].set_title(f"{col} original image")
        fig.colorbar(cmap1,ax=ax[0,c], orientation='horizontal',shrink=0.9, aspect=15)
        fig.colorbar(cmap2,ax=ax[2,c], orientation='horizontal',shrink=0.9, aspect=15)
        fig.colorbar(cmap3,ax=ax[3,c], orientation='horizontal',shrink=0.9, aspect=15)

    plt.show()

plot_fourier2d(image, coefs, frecsx, frecsy)
```









We readily see in the three channels (columns) that there are high frequency components (high magnitude (k_x, k_y) vectors with high intensity), specially paying attention to the plots in the first row (the log magnitudes of the Fourier coefficients). In particular we can see a periodic high frequency structure, that as we will prove with their removal represent the artifious pixel grid of the image.

We can build a mask for the Fourier space coefficients that makes zero any frequency component with a magnitude higher than say 0.1, which seems in the log magnitude plots (first row of previous image) to be a good enough boundary to preserve the inner components but remove all the periodic peaks that are presumably the artifacts we wish to remove.

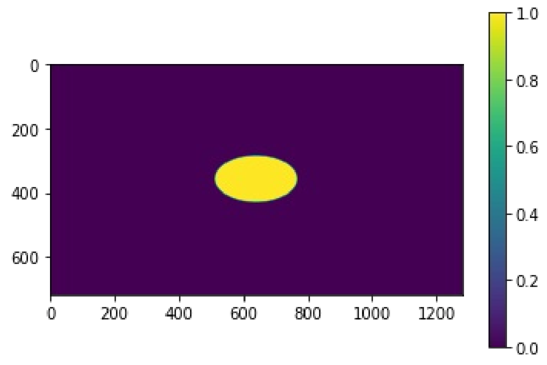
In [3]:

```
max_freq = 0.1

ffx, ffy = np.meshgrid(frecsx, frecsy)
filtered_coefs = np.zeros(coefs.shape, dtype=coefs.dtype)
mask = np.sqrt(ffx**2+ffy**2)<max_freq
filtered_coefs[mask, :] = coefs[mask, :]
corrected_image = np.abs(ifft2(fftshift(filtered_coefs, axes=(0,1)), axes=(0,1)))
corrected_image =  (np.iinfo(image.dtype).max*(corrected_image/corrected_image.max())).astype(image.dtype)
```

This is the mask we used:

In [4]:

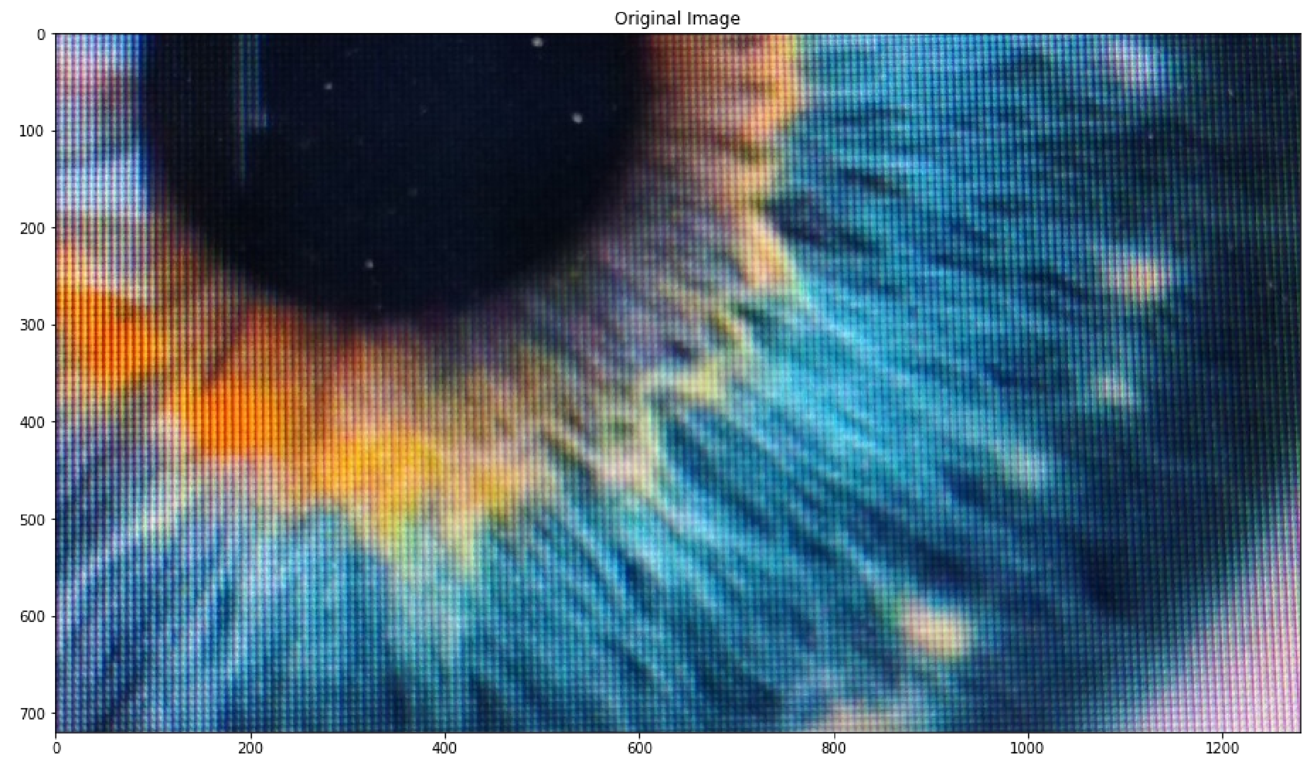
```
cmap = plt.imshow(mask)
plt.colorbar(cmap)
plt.show()
```

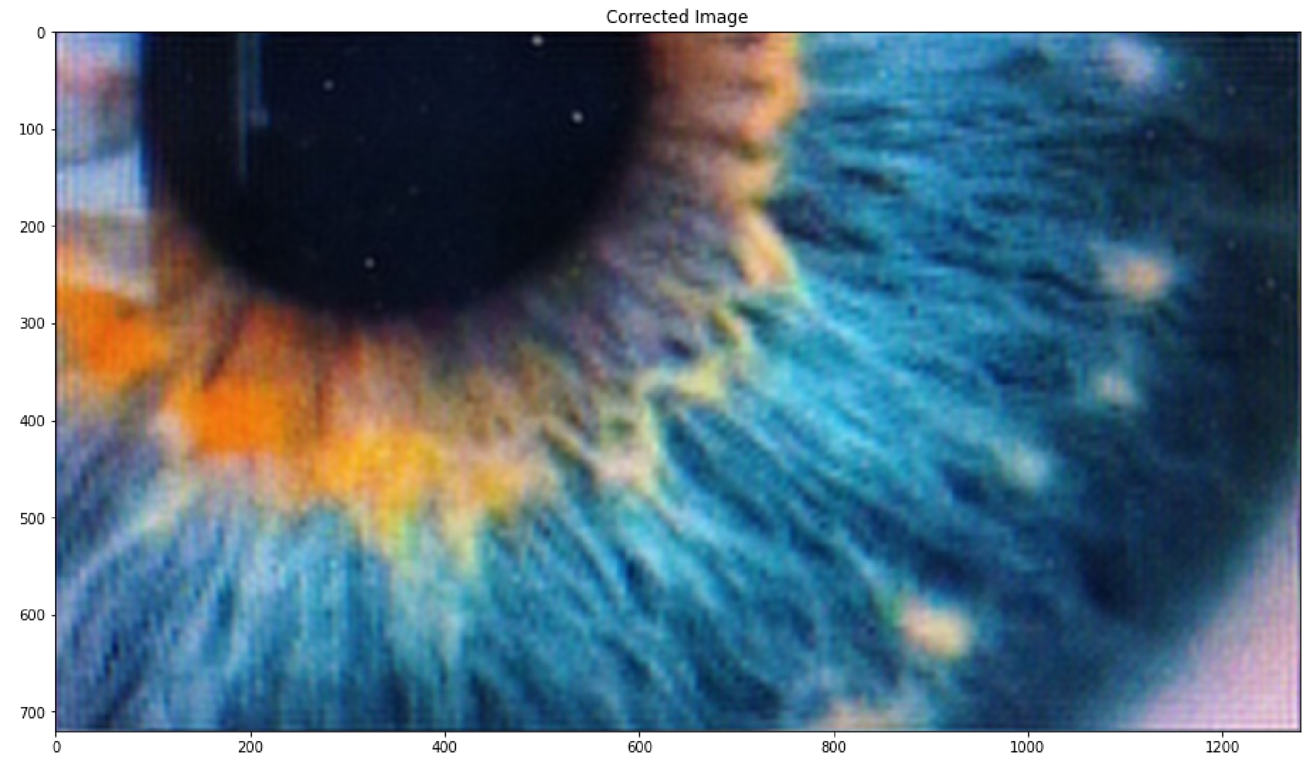


And this is the result:

In [5]:

```
fig, axs = plt.subplots(2, 1, figsize=(30, 20))
axs[0].imshow(image)
axs[0].set_title("Original Image")
axs[1].imshow(corrected_image)
axs[1].set_title("Corrected Image")
plt.show()
```



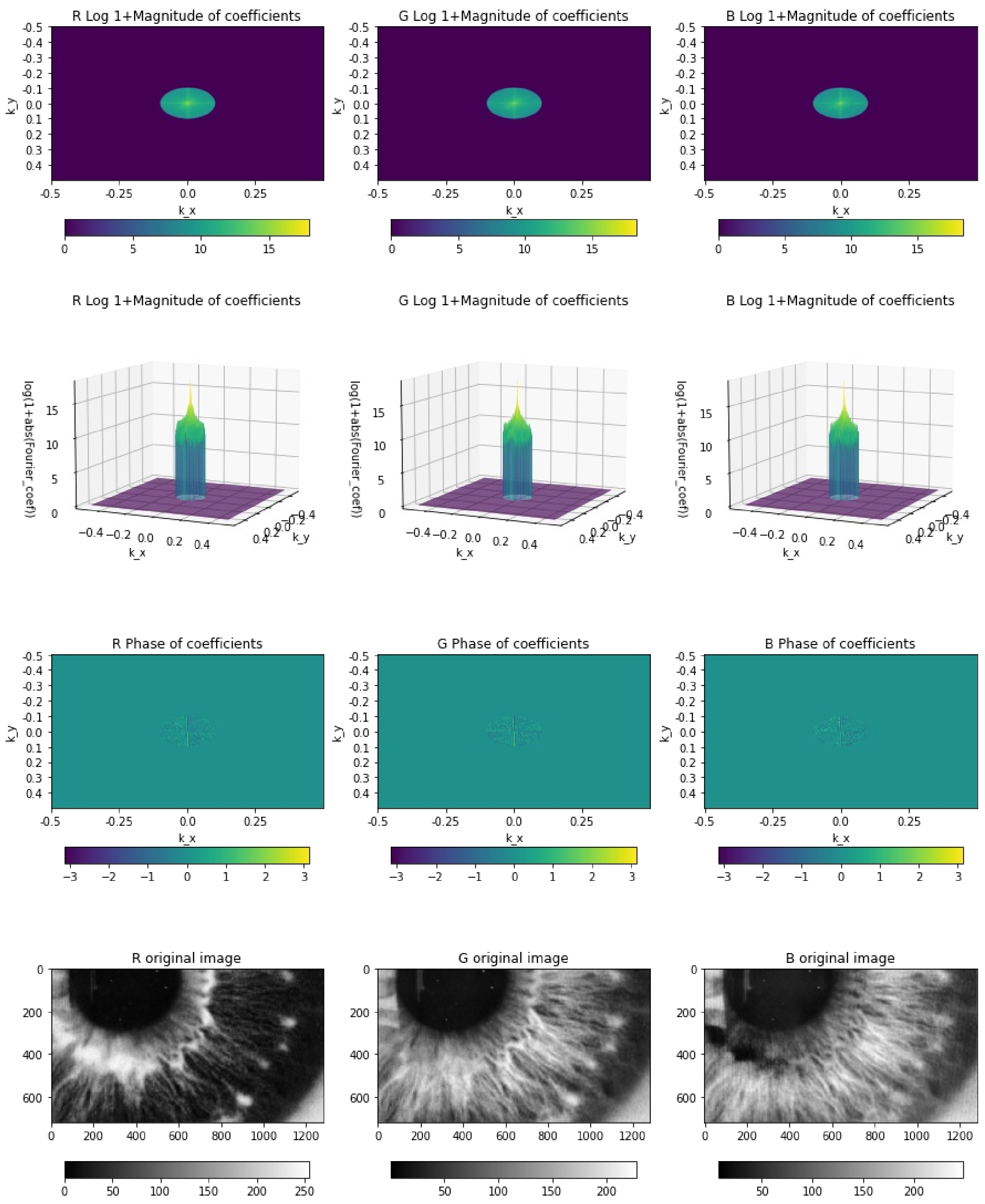


Effectively, we have succesfully removed all the "grid like" structure caused by the pixels!

We can now plot how the coefficients of the Fourier transform of the corrected image were left after the mask, to confirm it was a ciruclar mask with radious 0.1.

In [6]:

```
plot_fourier2d(corrected_image, filtered_coefs, frecsx, frecsy)
```


R Log 1+Magnitude of coefficients
G Log 1+Magnitude of coefficients
B Log 1+Magnitude of coefficients
k_x
k_y
log(1+abs(Fourier_coef))
R Phase of coefficients
G Phase of coefficients
B Phase of coefficients
R original image
G original image
B original image