PHASE SHIFT PROFILOMETRY

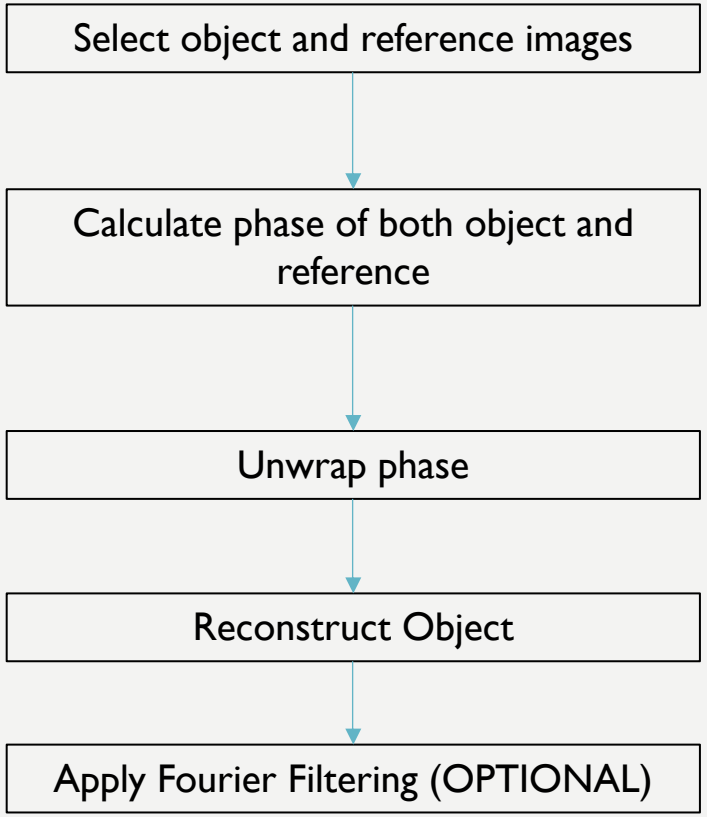
ACTIVITY 08

PHYSICS 301

KENNETH M. LEO

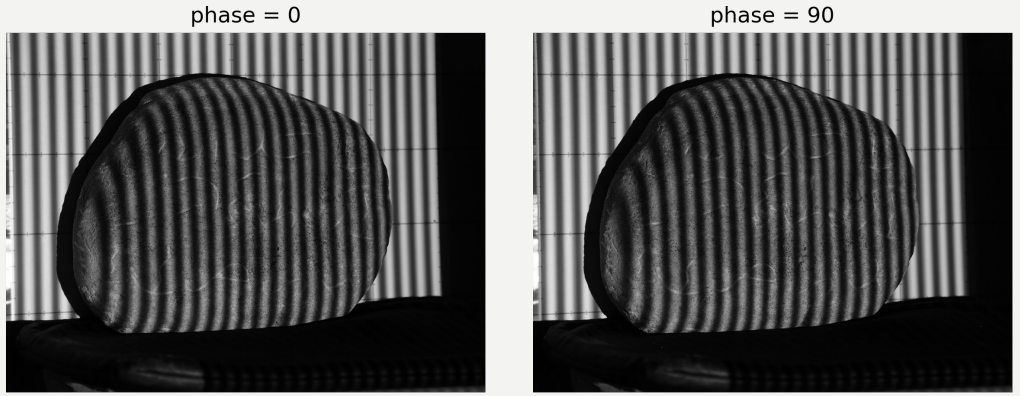
PHASE SHIFT PROFILOMETRY

The goal of this activity is to reconstruct the given object using phase shift profilometry. Now, the summary of the process is as follows:

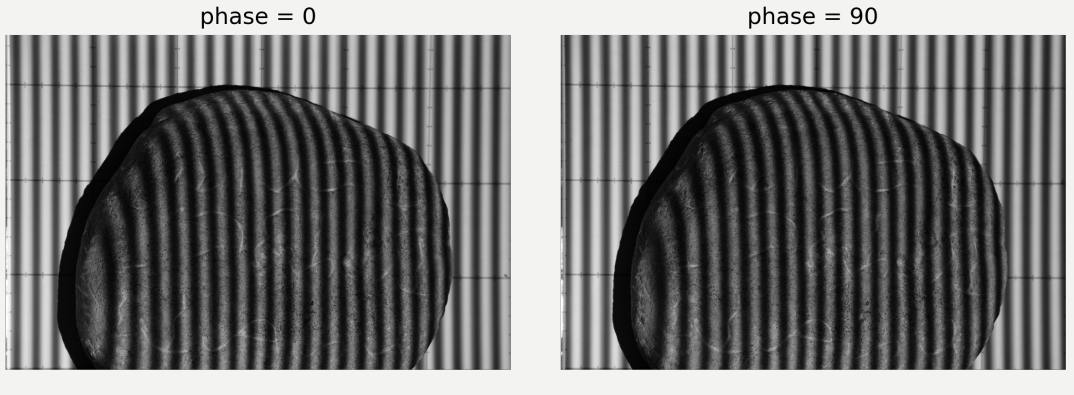


TRANSFORMING GIVEN IMAGES

These are the given objects but as we can see, they are taken at a different perspective. So, we use cv2 in Python to convert the perspective of this such that it would appear that the camera was positioned directly in front of the object.



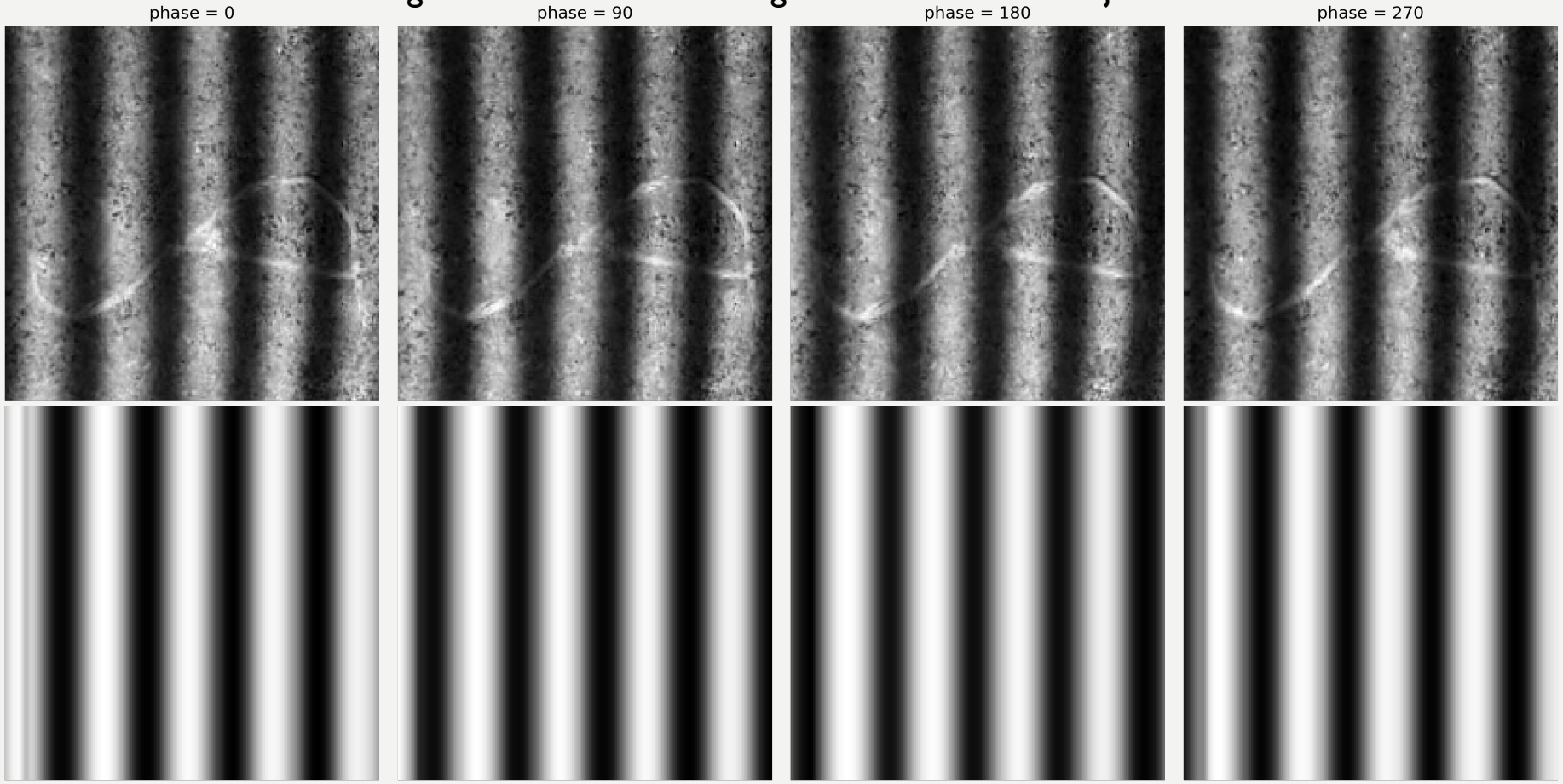
Original Images



Transformed Images

TRANSFORMING GIVEN IMAGES

From the transformed images, we can now actually get the region of interest (object) as well as our background (reference). Note that the reference values were averaged from the actual background of our rock object.



PHASE WRAPPING AND UNWRAPPING

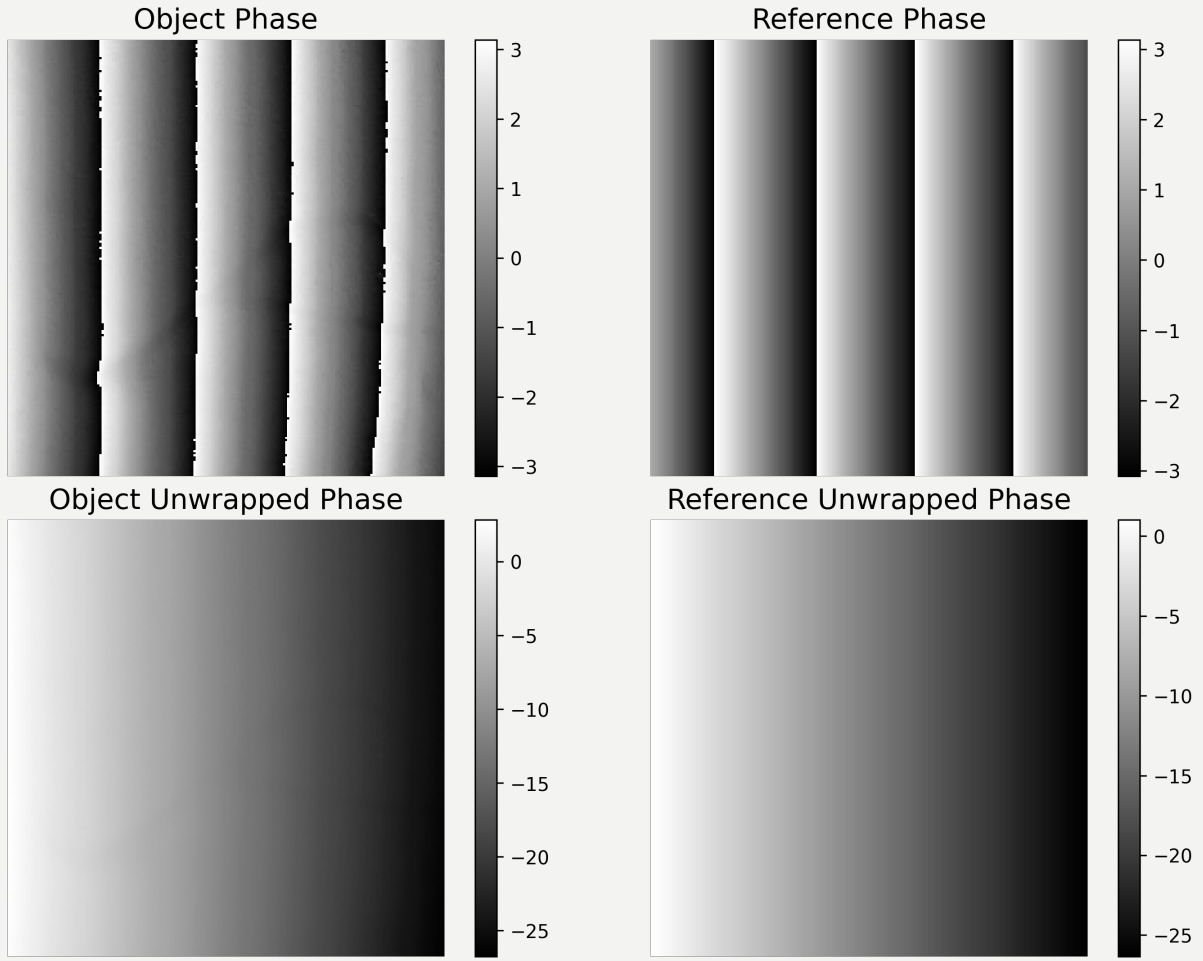
To perform phase wrapping, we calculated the intensity functions of each images and used them to calculate the phase

$$I_1(x,y) = I_{mean}(x,y) + I_{mod}(x,y)\cos(0)$$
$$I_2(x,y) = I_{mean}(x,y) + I_{mod}(x,y)\cos\left(\frac{\pi}{2}\right)$$
$$I_3(x,y) = I_{mean}(x,y) + I_{mod}(x,y)\cos(\pi)$$
$$I_4(x,y) = I_{mean}(x,y) + I_{mod}(x,y)\cos\left(\frac{3\pi}{2}\right)$$

$$\phi(x,y) = \tan^{-1}\left(\frac{I_4(x,y) - I_2(x,y)}{I_1(x,y) - I_3(x,y)}\right)$$

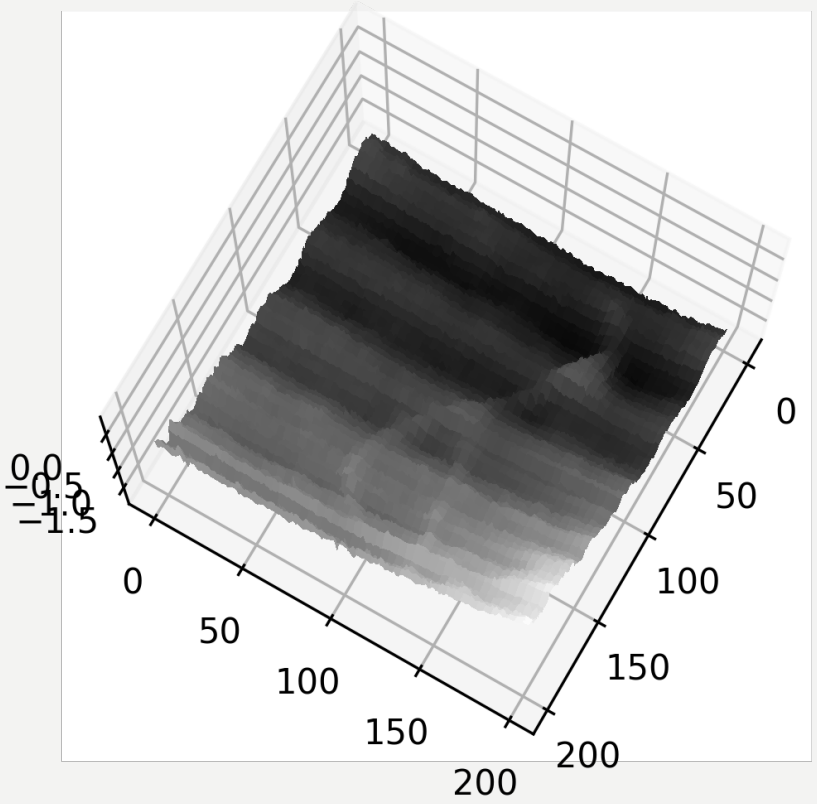
Then we need to unwrap the phase because of the range of possible values of tangent inverse. To do this, we used the built-in unwrap function in Python's numpy package.

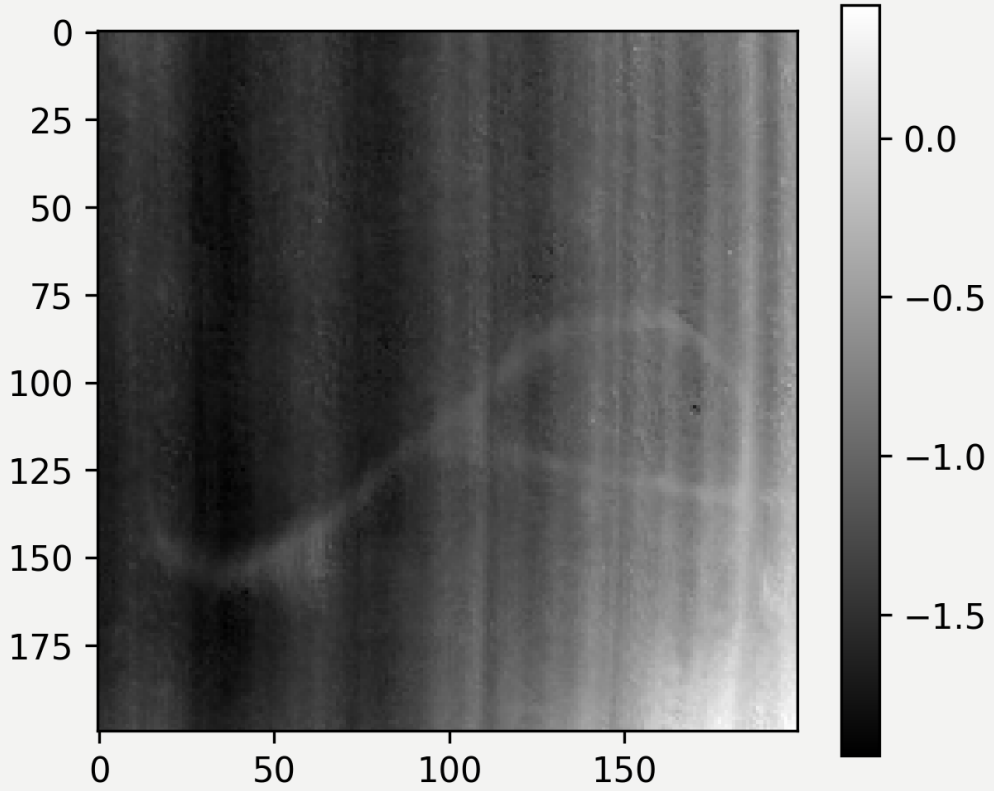
We see from the figure on the right that the wrapped phase (top) has phases only from –pi to pi while the unwrapped phase (bottom) removes that bound.



OBJECT RECONSTRUCTION

To reconstruct the image, we simply subtract the unwrapped phases of the object and reference, and we get this 3D and 2D reconstructed image. Notice that we are now able to see the object (letter-A looking shape). But we can make this appear better by applying Fourier filtering to our reconstructed object.

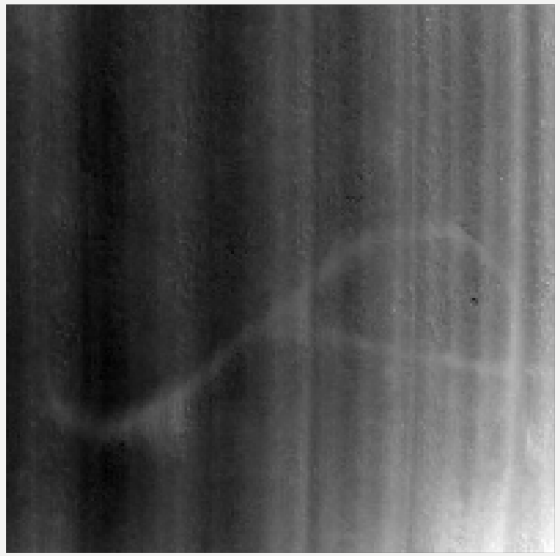




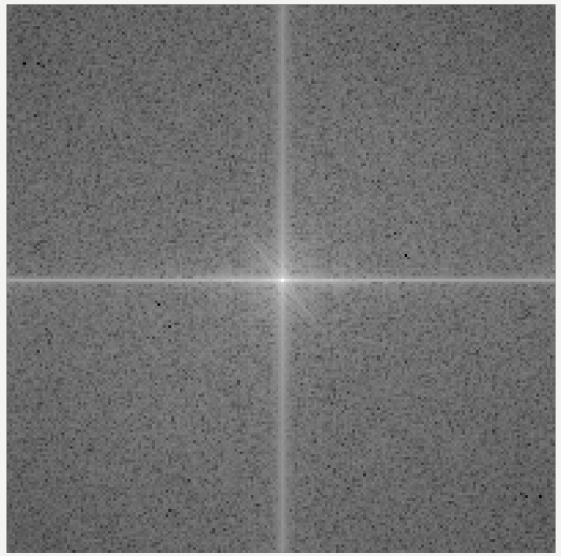
OBJECT RECONSTRUCTION W/ FOURIER FILTERING

The point of Fourier filtering is to enhance any image by removing the repeating patterns seen in the image. This is done by putting a mask on the image and this mask is based on the Fourier transform of the image. We can see from the rightmost picture that the recovered image has lost the 'streaks' found on the original reconstructed object.

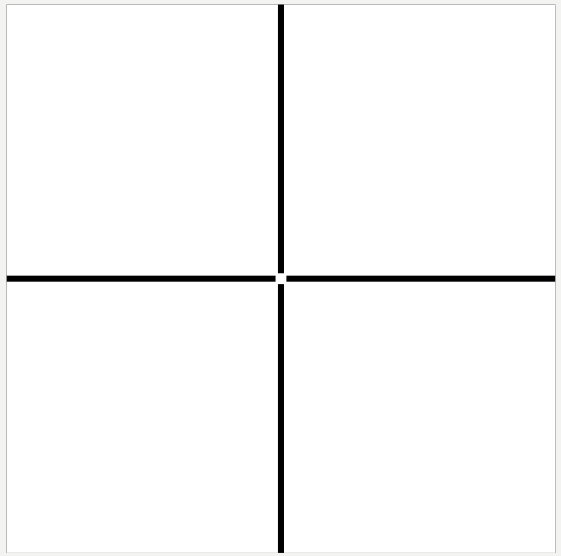
reconstructed object



Fourier transform (log)



mask used

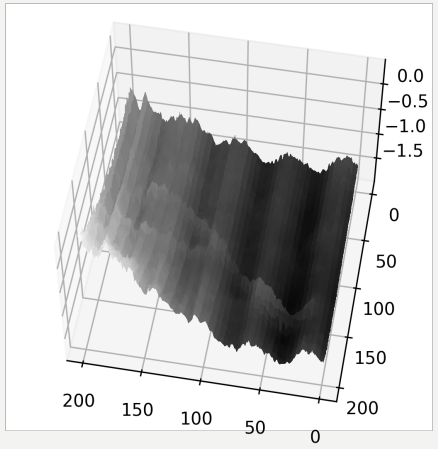


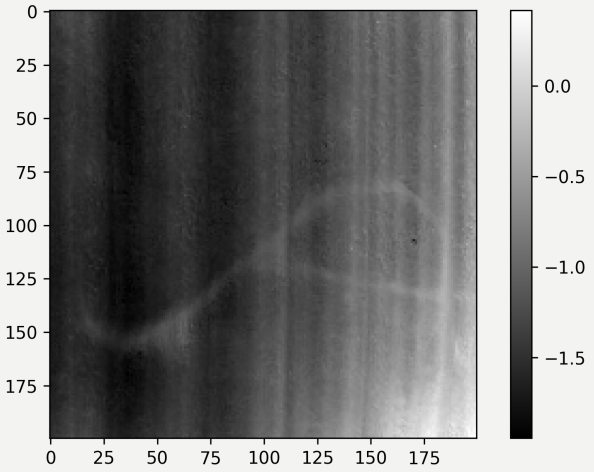
Recovered reconstructed image

Note: We can also adjust the size of the mask used if we want :D

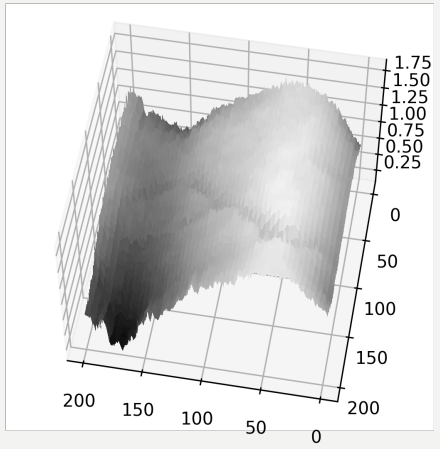
OBJECT RECONSTRUCTION COMPARISON

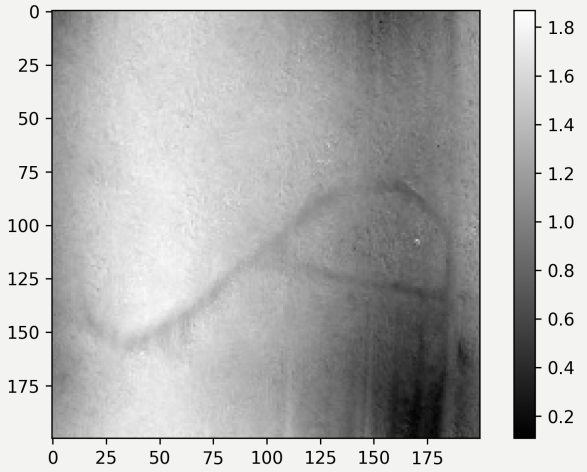
Without Fourier Filtering





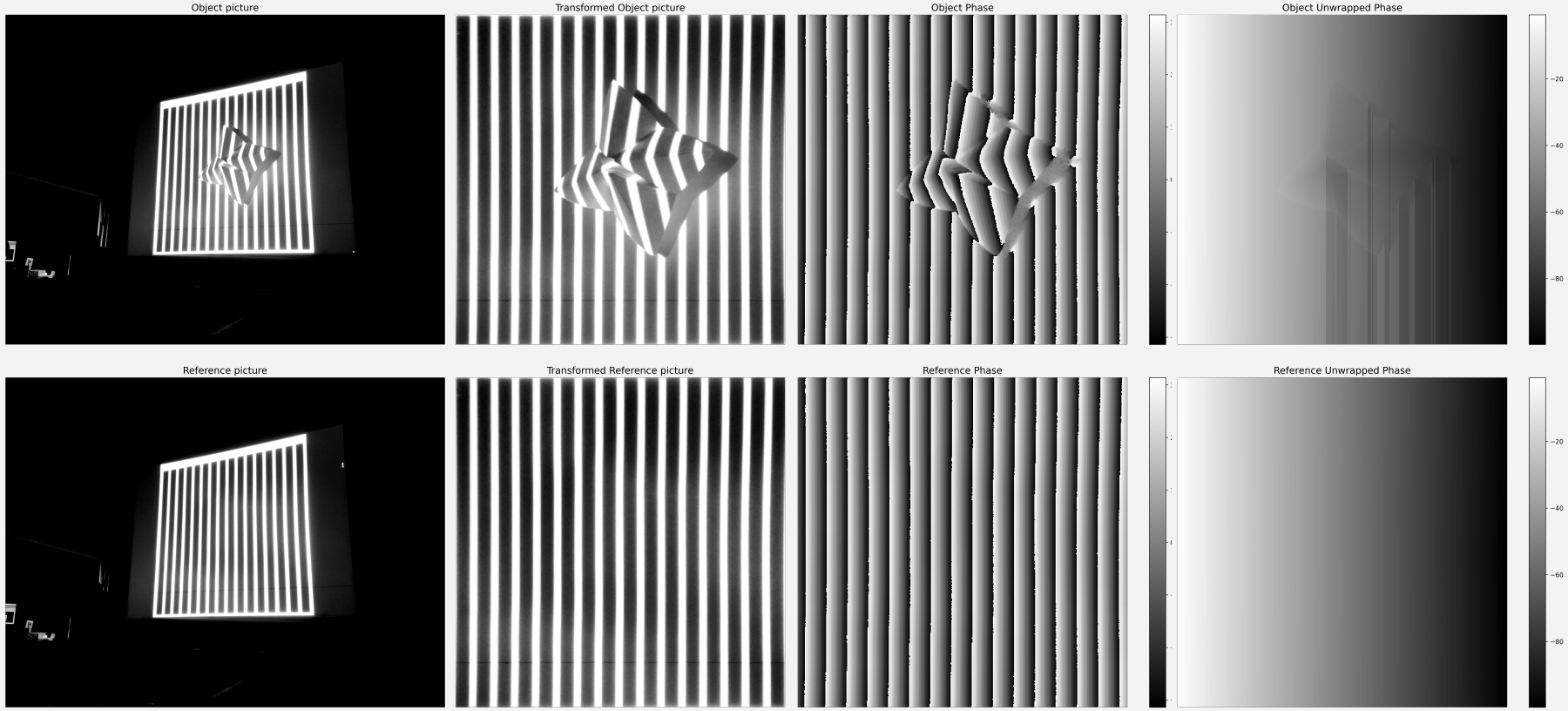
With Fourier Filtering

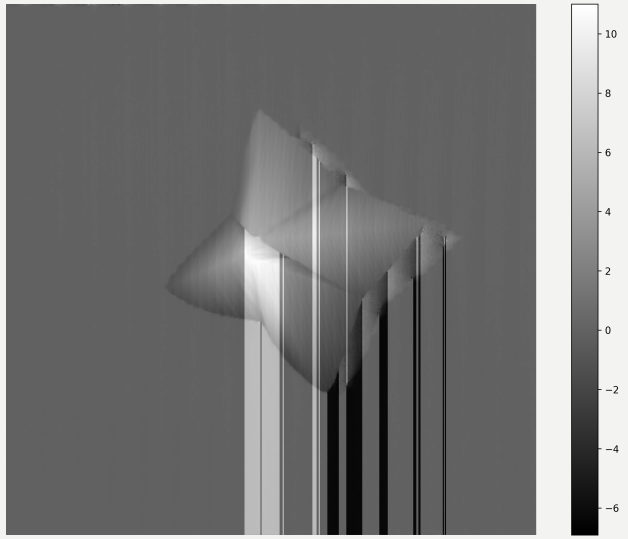




ANOTHER EXAMPLE: FLOWER

Now, let's look at another example where we now have an image of the reference plane. This dataset was taken during my undergrad days.





Note that we cannot apple Fourier transform here because the noise that we are seeing is not a pattern. So to further enhance this, one must use other filtering techniques.

SUMMARY PLUS REFERENCES

Summary

From this activity, I was able to use my undergrad knowledge of profilometry and Fourier filtering. This was actually a pretty fun activity because I learned that even if no reference or background image given, one can actually average out the parts of the photo with no object to get a reference image.

One thing that I was not able to do was to use the quality-guide phase unwrapping which is why I am reducing my points in the Technical correctness part of self-score.

References

[1] https://www.geeksforgeeks.org/perspective-transformation-python-opencv/

Score:

Technical Correctness – 28
Quality of Presentation – 30
Reflection – 30
Ownership – 10

Total – 98/100