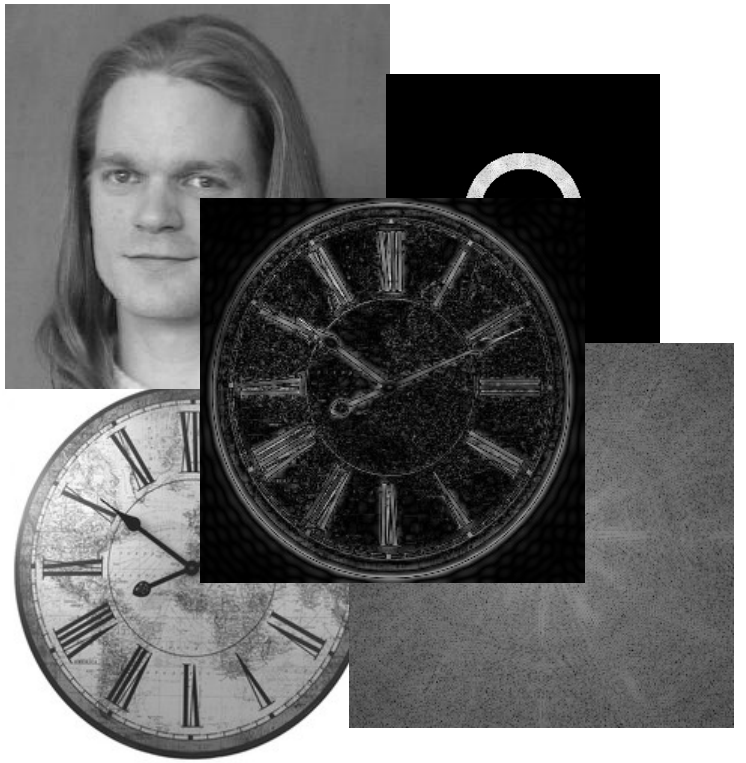
FUN WITH FOURIER TRANSFORMS

IMAGE PROCESSING, RETRIEVAL AND ANALYSIS (I) – WINTER 2016

Sahib Mukhija, 374154
Shubham Kapoor, 373997
Vatsala Sharma, 374843
Shipra Pant, 374023
Mayesha Tasnim, 374098

OUR TASKS

- Task I – Rendering a Circular Band on a Given Image
- Task II – Implementing Band Pass Filter
- Task III – Exploring The Importance of Phase



TASK 1 – RENDERING A CIRCULAR BAND ON A GIVEN IMAGE

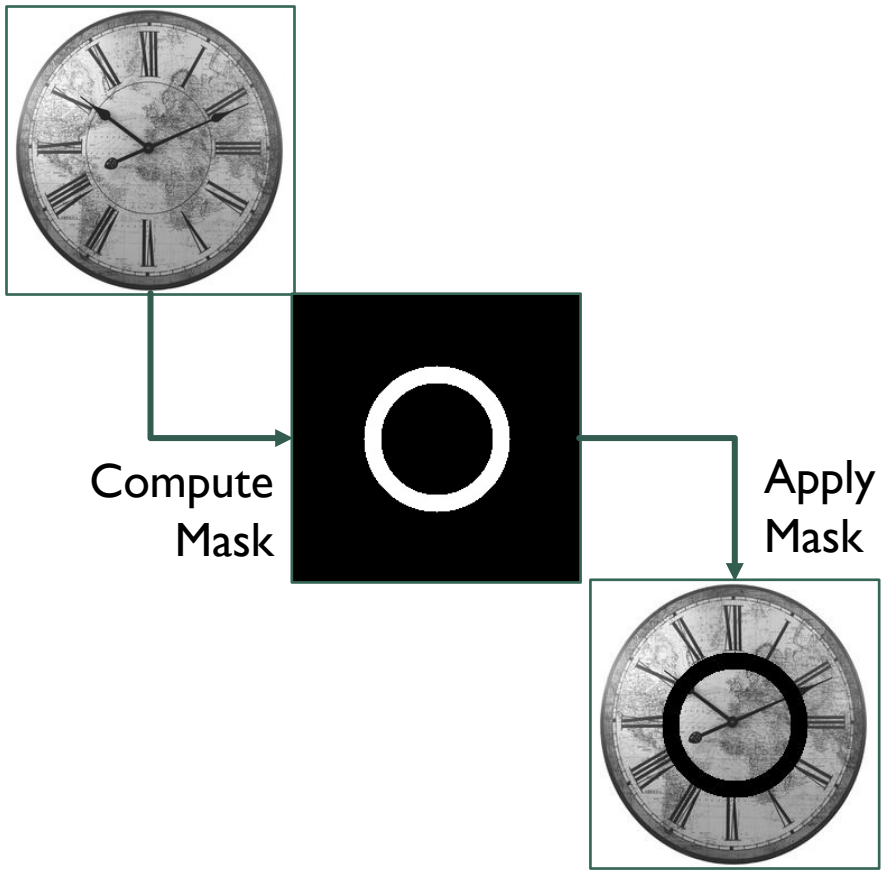
- Hands-on Introduction to NumPy and SciPy
- Can be done in different approaches
- Naïve approach:
  - Iterate through every Image Pixel
  - Longer execution time
  - Nested loops increase code complexity in terms of readability

$$\tilde{g}(x,y) = \begin{cases} 0, & \text{if } r_{\text{min}} \leq \|(x,y) - (\frac{w}{2}, \frac{h}{2})\| \leq r_{\text{max}} \\ g(x,y), & \text{otherwise} \end{cases}$$

```
for i in range(width):
 for j in range(height):
  #Value of g_tilda between R_min & R_max
   if(g_tilda(i, j, r_min, r_max) == 0):
     img[i,j]=0
```

TASK 1 – OUR WORKING APPROACH

- Applying a Boolean Mask
  - MeshGrid (NumPy) and Boolean Index Arrays
  - Masking allows us to work on the pixels at once
- Working Process
  - STEP 1: Create a Matrix containing the distances of each pixel from the center of the image
  - STEP 2: Create a Boolean Index Array i.e. mask by comparing each distance with Rmin and Rmax
  - STEP 3: Modify the original image according to the values in the mask



TASK 1 – CODE MODULARITY

```python
def main():
  (w,h) = inputImg.shape
  rMin,rMax = (50,65)

  #computing mask matrix
  mask = maskMat(rMin,rMax,w,h)

  #modifying image with mask
  inputImg[mask] = 0
```

```python
def maskMat(rMin, rMax, w, h):

  #computing distance matrix
  dist = distanceMat(w, h)

  #creating Boolean array
  mask = ((dist>=rMin)&(dist<=rMax))

  return mask
```

```python
def distanceMat(w, h):

  #get dimension range of image
  xs = range(w)

  #create co-ordinate matrices
  X,Y = np.meshgrid(xs,xs)

  #compute Euclidian distance
  dist = np.sqrt(np.square(X - w/2)
       + np.square(Y - h/2))

  return dist
```

TASK 1 – WHAT WE HAVE LEARNED

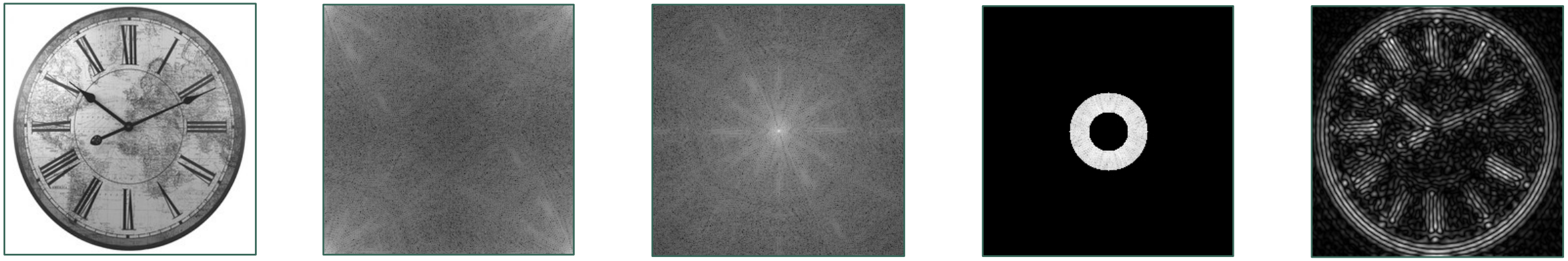
- **Basics**: Doing this task helped us learn how to do image processing with the help of Python libraries (NumPy, SciPy and MatPlotLib)
- **Vectorize**: Using vectorization approach, it becomes easy to write and understand the code. Also, it uses the full potential of pre-compiled NumPy routines
- **Masking**: Using 2D Boolean Index Arrays to apply masks on images

TASK II – IMPLEMENTING BAND PASS FILTER

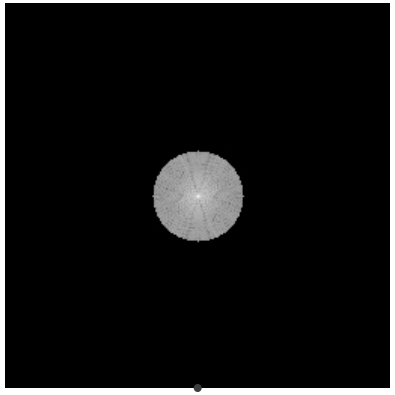
TASK II – WORKING PROCESS

- STEP 1: Apply Fast Fourier Transform (numPy.fft.fft2) to original image
- STEP 2: Apply numPy.fft.fftshift on the obtained frequency spectrum to center it at origin
- STEP 3: Filter out frequencies by applying the inverse of the Mask used in Task I
- STEP 4: Apply Inverse Shift and Inverse Fourier Transform (numPy.fft.ifft2) on the filtered frequency spectrum and obtain image

TASK II - RESULTS

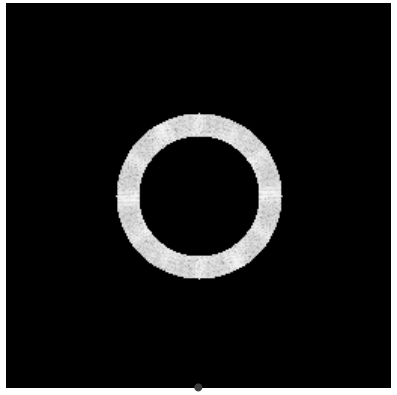


TASK II – ANALYZING RESULTS

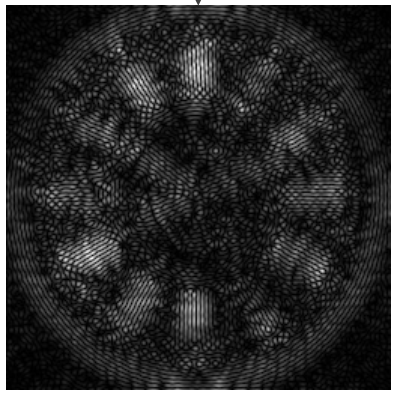


Low Rmin
Low Rmax

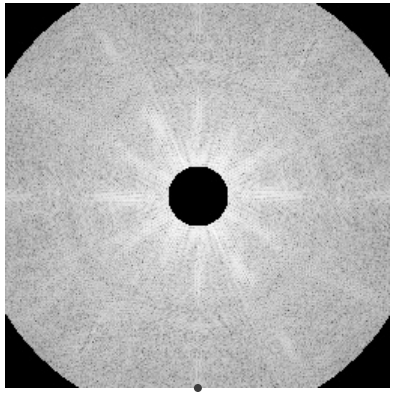
Smoothed out Image



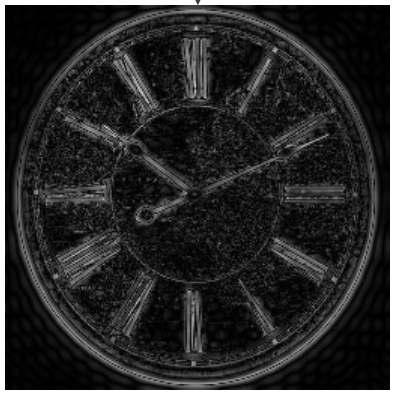
Medium Rmin
Medium Rmax



Obscure Image



Low Rmin
High Rmax

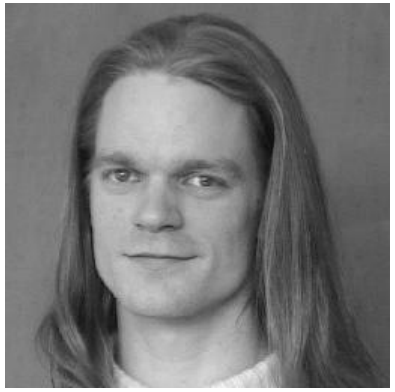


Sharpened Image (Pronounced Edges)

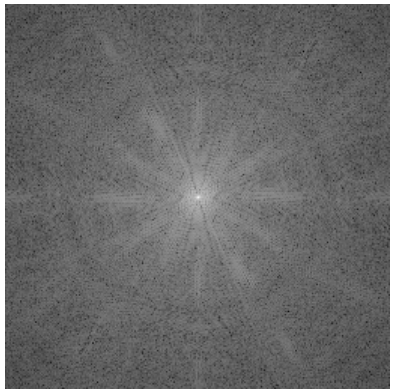
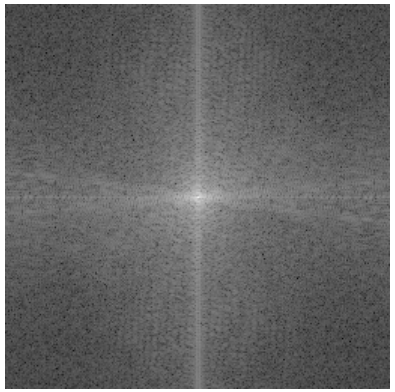
TASK 11 – WHAT WE HAVE LEARNED

- **Fourier Transform**: How to perform Fourier Transform on images (or in a broader sense to 2-dimensional vectors)
- **Visualizing FT**: We can now see at first hand that applying FT on a function gives us its frequency distribution. It is of course to be noted that we can only visualize the real components of the transformation
- **Effects of filters**: Filtering out higher frequencies reduces noise in the image, while filtering out the lower frequencies gives us the edges
- **Use of fftshift**: After applying FFT, it is necessary to perform a shift so that zero frequency values lie at the center of the spectrum. This is due to the technicality of FFT algorithm implementation.

TASK III - EXPLORING THE IMPORTANCE OF PHASE

I.  

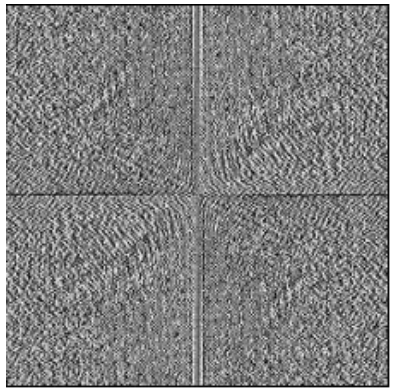
Take Two Images

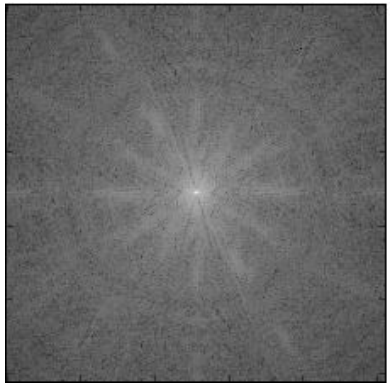
II.  

Compute Fourier Transform, Do fftshift

TASK III - EXPLORING THE IMPORTANCE OF PHASE

III.





Take the magnitude A of one image and the phase B of another

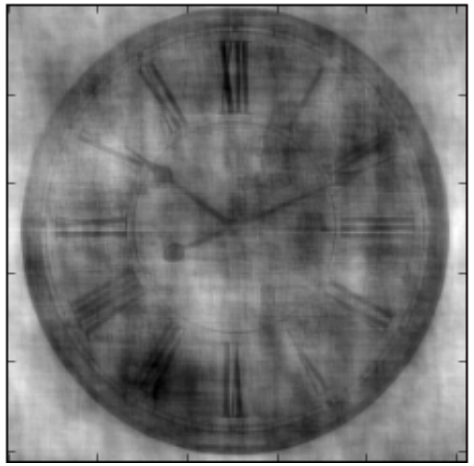
IV.

Compute an image by combining the magnitude and phase

$$AcosB + iAsinB = Ae^{iB}$$

TASK III - RESULTS

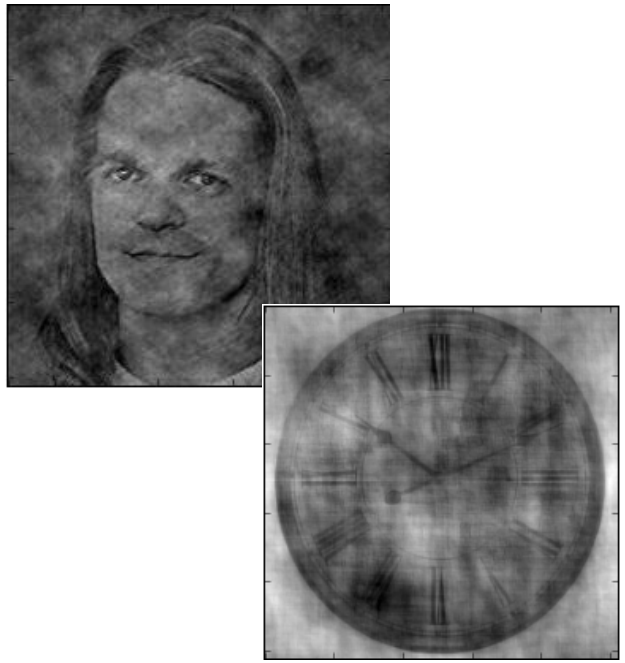
When phase of ‘bauckhage.jpg’ is used



When phase of ‘clock.jpg’ is used

TASK III – WHAT WE HAVE LEARNED

- Most of the image information is stored in the Phase of its Fourier Transform
- This is because the Phase stores what is *unique* about the signal of the image pixel



Thank You