FUN with **Fourier** transforms

BIT Course Image Processing, Retrieval and Analysis

Prof. Dr. Christian Bauckhage

Students: Ola Al Naameh, Arul Selvam, Tatiana Novikova, Fida Mohammad, Can Yuce, Oguz Uzman, Amal Amouri

*The **Fourier transform** decomposes a function of time (a signal) into the frequencies that make it up, in a way similar to how a musical chord can be expressed as the amplitude (or loudness) of its constituent notes.**

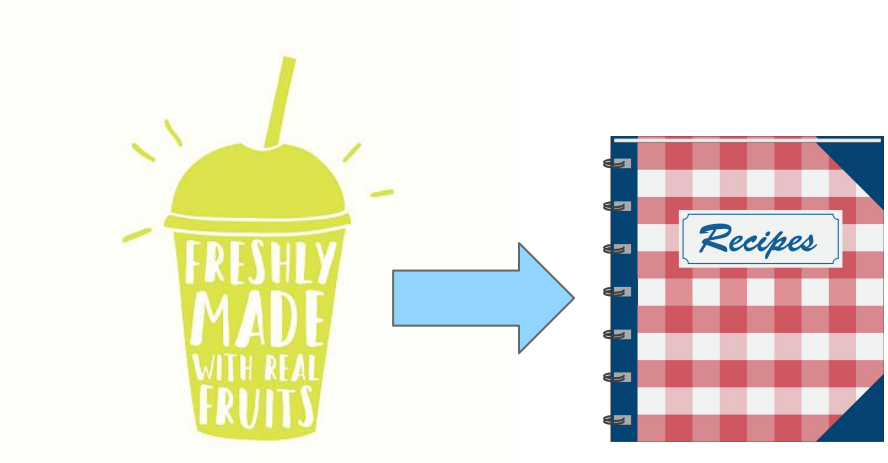
* Source: https://en.wikipedia.org/wiki/Fourier_transform

Generalities

- Implementation of the solution is in **Python.**
  - Using **NumPy** and **SciPy** modules.
  - Anaconda package (not obligatory).

Fourier transform ver.1

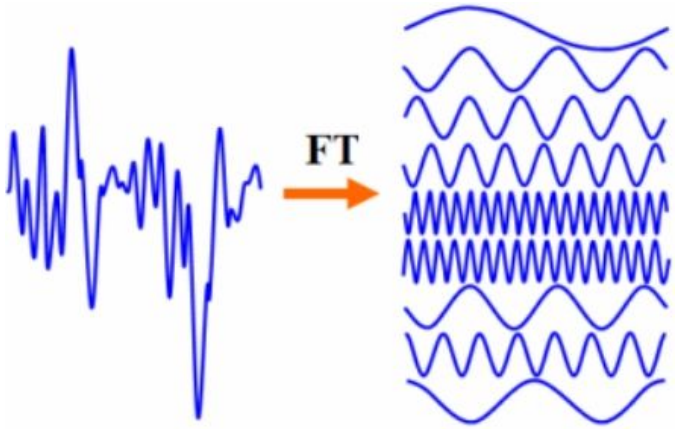
- **What does the Fourier Transform do?** Given a smoothie, it finds the recipe.
- **How?** Run the smoothie through filters to extract each ingredient.
- **Why?** Recipes are easier to analyze, compare, and modify than the smoothie itself.
- **How do we get the smoothie back?** Blend the ingredients!



Fourier transform ver.2

- Here's the **"math"** version of the previous slide:

  The Fourier Transform takes a time-based pattern, measures every possible cycle, and returns the overall "cycle recipe" (the amplitude, offset, rotation speed for every cycle that was found).



Task 1

Task 1.1: warm-up

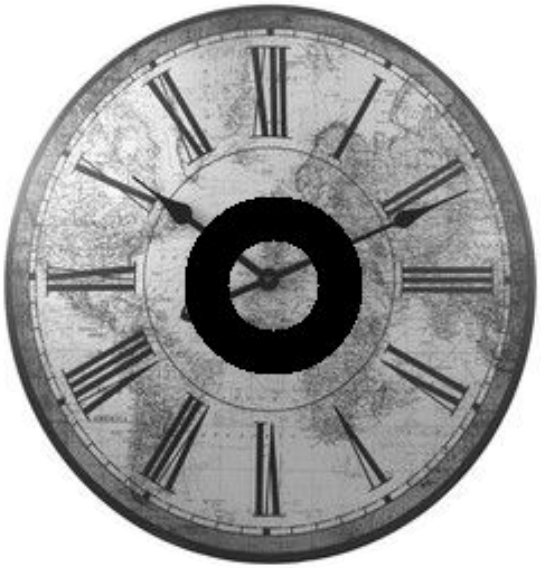
- **Why?**
  - Getting familiar with simple intensity images.
- **How?**
  - Reading an intensity image into a 2-dimensional array.
  - Calculate the width and height of the image.
  - Calculate the euclidean distance of each pixel from the centre.
  - Suppress all the pixels whose distance is greater than an arbitrary R min and less than R max.

A Python Code Sample

```
res = np.array([[0 if r_min <= np.linalg.norm([x-(h/2.0), y-(w/2.0)])
<= r_max else img[x, y] for y in range(w)]\for x in range(h)], dtype=np.uint8)
```

Task 1.1: warm-up

- **Results**



Rmin = 20, Rmax = 40

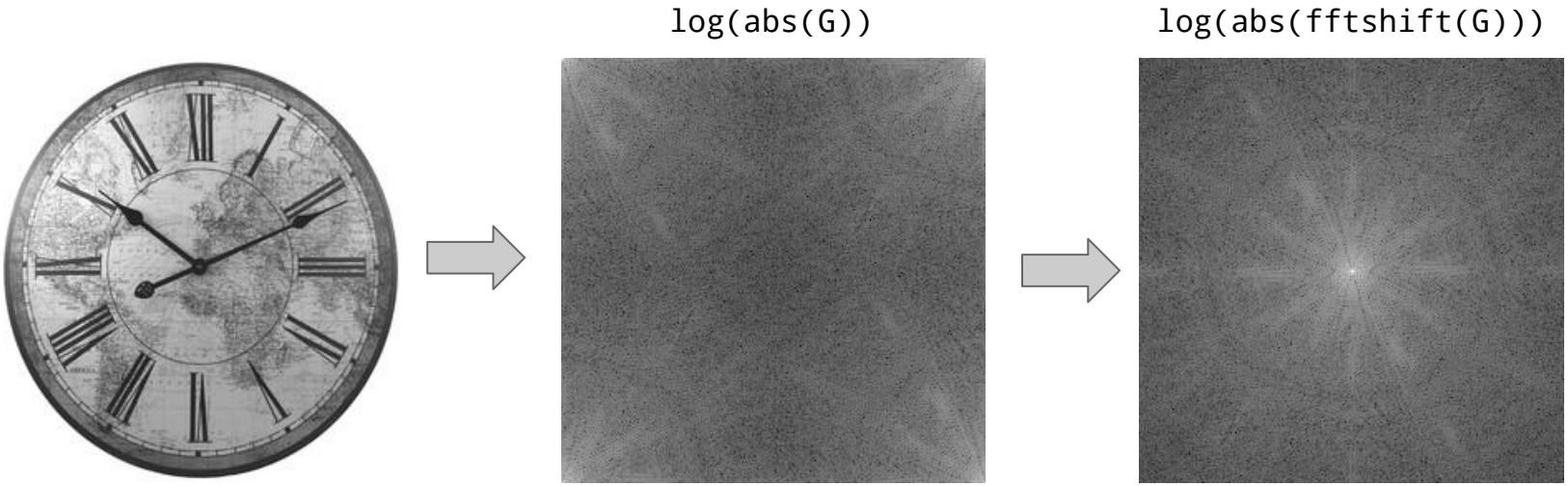
Rmin = 20, Rmax = 80

- We learnt how to work with digital images like reading, writing and selecting individual pixels

Task 3

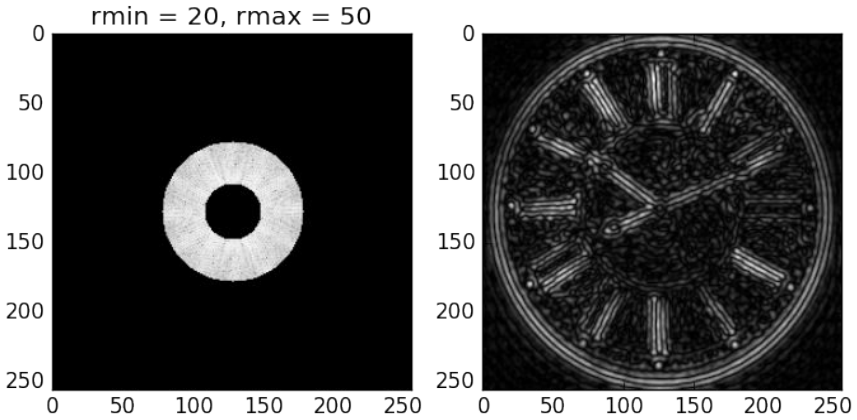
Task 1.3: Implementing a band pass filter

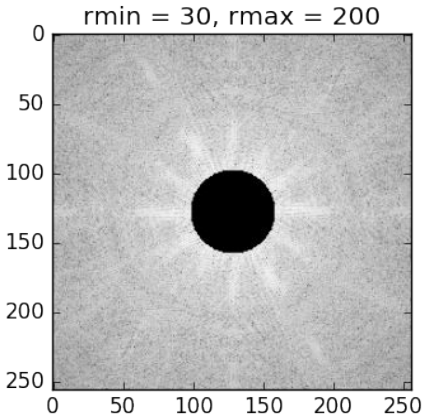
- **Why?** Implementing a band pass filter
- **How?**
  - Perform Fourier transform on the images.
  - Shift the zero-frequency spectrum to the center.
  - The pixels which are closer to the center than rmin and further than rmax are set to 0. Thus, suppressed.

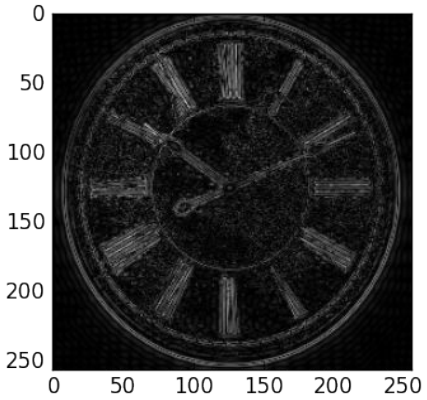


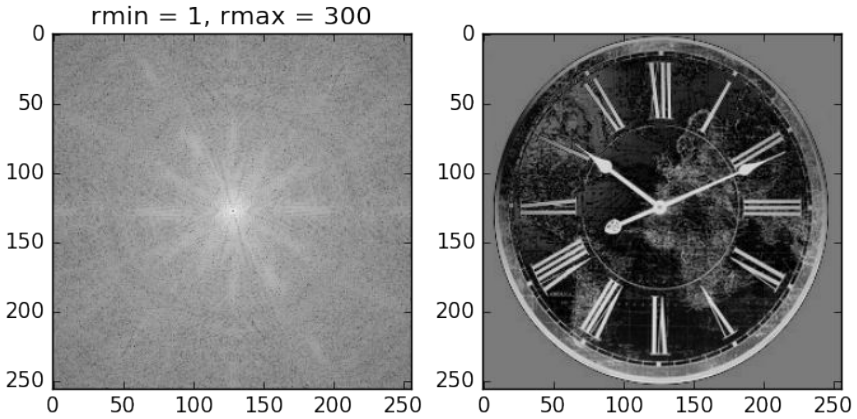
Task 1.3: Implementing a band pass filter

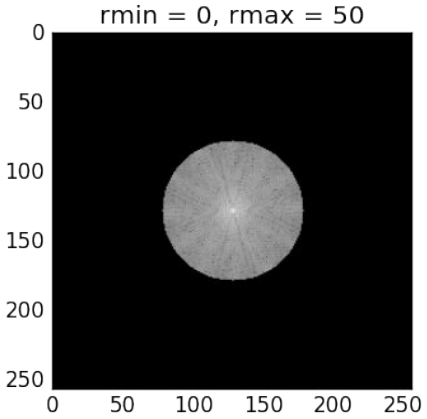
- **Results**

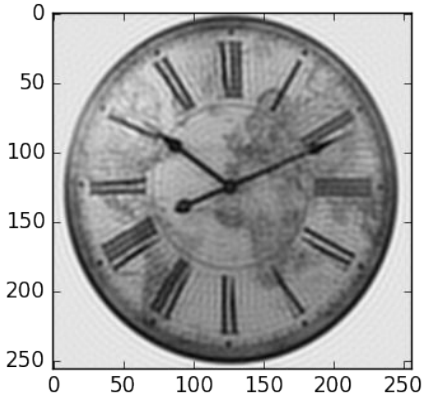












Task 1.3: Implementing a band pass filter

- **Results**
  - The lower frequencies contain 'more important' details about the image.
  - Removing zero frequency changes image drastically.
  - Removing some of the highest frequencies reduces some edge sharpness, but their effects aren't as visible until many of them are removed.

Task 4

Task 1.4: exploring the importance of phase

- **Why?**
  - Exploring the importance of phase.
- **How?**
  - Calculate Fourier transforms G and H of images $g(x,y)$ and $h(x,y)$.
  - Calculate the Magnitude of G by taking the absolute of every pixel in G.
  - Calculate the Phase of H by calculating the angle between imaginary and real parts of H.
  - Recalculate Real and Imaginary components and apply IFFT.

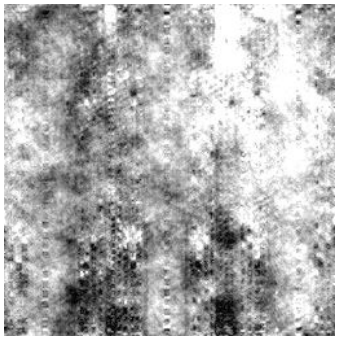
Task 1.4: exploring the importance of phase

building.png

clock.png

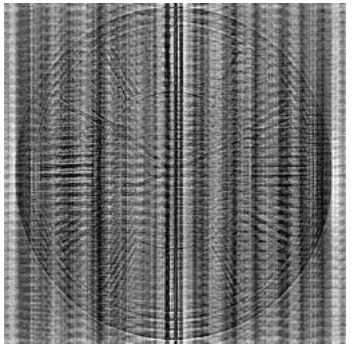
lena.png

Phase of building.png' and magnitude of 'clock.png'.



Phase of 'lena.png' and magnitude of 'clock.png'.

Phase of 'clock.png' and magnitude of 'building.png'



Task 1.4: exploring the importance of phase

- **Results**
  - The phase component influences the reconstructed image more than the magnitude component does.
  - The magnitude component is similar for most of the natural occurring images but phase component varies a lot.
  - Image features are related with the phase component.
  - Thus phase component is more important while reconstructing the images.

Conclusion

- Implementation...
- Using **NumPy** and **SciPy** modules helped to..

Thanks!

Any questions?