HW3

Preparation

Please make sure you have read Lectures 6 - 9, in particular, Lectures 7 – 8.

You may find **Slide Deck 8, pp. 54 – 68**, very useful. You may read them while you are implementing the solutions. (These slides were not covered in the class due to time constraints.)

Please read Textbook chapters 15 – 18, at least those covered by lectures.

The notations in this slide deck may be a bit different from the lecture slides! The goal is to make implementation easier.

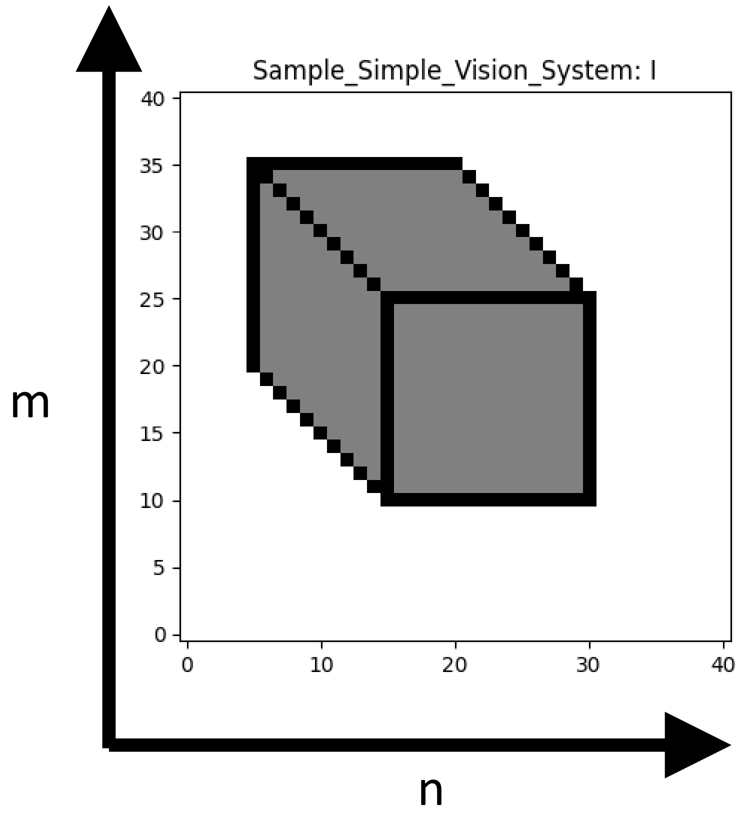
Caution

- Please do not import packages (like scikit learn) that are not listed in the provided code.

- In this homework, you are NOT allowed to use NumPy's or other Python libraries' built-in **convolution, DFT, IDFT, and filter functions.** If you use them, you will get 0 points for the entire assignment.

Convention

- In this homework, given a map (or a matrix), say I
  - I[n, m] means the i-th horizontal index (left-right) and j-th vertical index (bottom-up)
  - n >= 0, m >= 0



Color images

- Please note that a color image I means that the image I is a 3D tensor. The 3rd dimension corresponds to R, G, and B.

- In this homework, you will process each channel separately. You can extract each by I[:, :, c], where c is between 0 and 2.

Question 1: Convolution

- There are many variants, but in this homework, please follow the formula below.
  - Given an image I, we will first do zero padding (think about why)

I

I_pad

Question 1: Convolution

- Then, we perform
- $I_{out}[n,m,c] = \sum_{k=0}^{K-1} \sum_{l=0}^{L-1} I_{pad}[n+k,m+l,c]\, h[K-1-k,L-1-l]$
- K and L are the 2D kernel h's shape
- The range of *n* and *m* are [0, N-1] and [0, M-1], respectively, where N and M are the input image I's shape (not I_pad)
- That is, I_{out} will have the same shape as the input image I

Question 1: Convolution

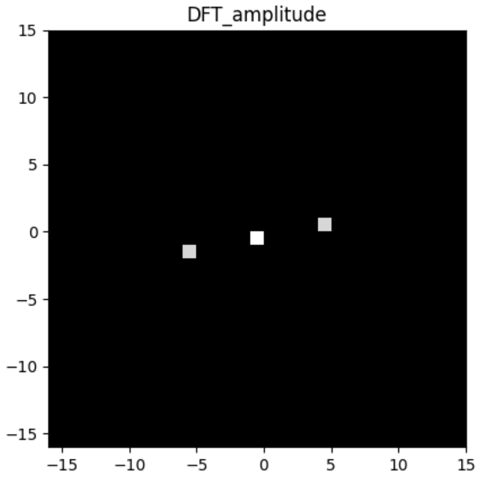
- In your implementation, if you can implement $\sum_{k=0}^{K-1} \sum_{l=0}^{L-1} [something]$ without using a for loop, it will save a lot of computation time in Python

Question 2: DFT

- Please follow the following formula
- $I_{\text{out\_real}}[u, v, c] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} I[n, m, c] \cos\left(-2\pi\left(\frac{un}{N} + \frac{vm}{M}\right)\right)$
- $I_{\text{out\_imaginary}}[u, v, c] = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} I[n, m, c] \sin\left(-2\pi\left(\frac{un}{N} + \frac{vm}{M}\right)\right)$
- The range of u, v are [0, N-1] and [0, M-1], respectively!

Question 2: DFT

- When visualization, we have implemented a function to change the range to be surrounding (0, 0).



- However, whenever you access $I_{\text{out\_real}}$, $I_{\text{out\_imaginary}}$, $I_{\text{out\_amplitude}}$, $I_{\text{out\_phase}}$, which are of shape N x M x 3, $[u, v, c]$ means horizontal frequency u and vertical frequency v

Question 2: DFT

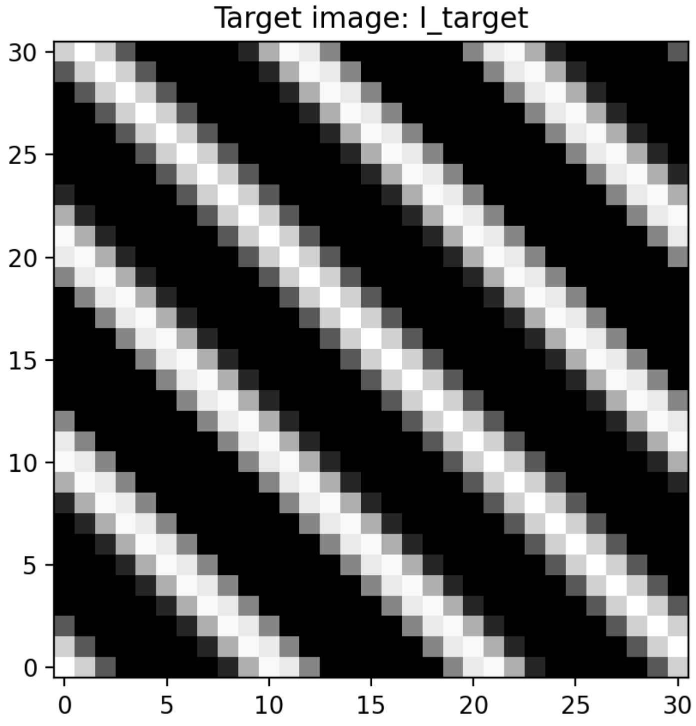
- In your implementation, if you can implement $\sum_{k=0}^{K-1} \sum_{l=0}^{L-1} [something]$ without using a for loop, it will save a lot of computation time in Python

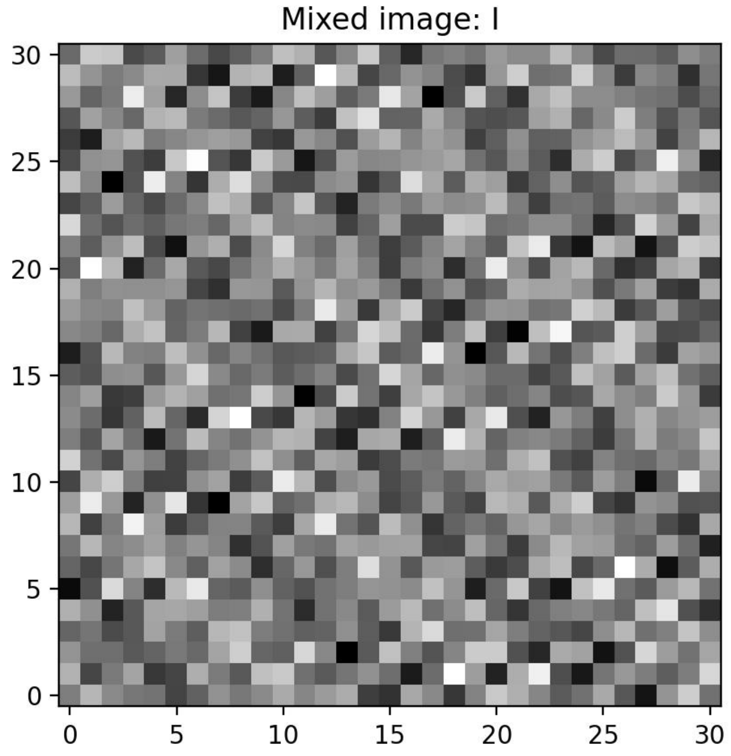
Questions 3 – 7

- Please follow the homework instructions in GitHub

Question 8: Recovering the target image

- Target image and the mixed image with other frequency components





Question 8: Recovering the target image

- Your goal is to recover the target image, given the mixed image and target image’s frequencies