CENG466-Fundamentals of Image Processing Report II

Sena Nur Şengül
Computer Engineering
Middle East Technical University
Ankara, Turkey
e231049@metu.edu.tr

I. Introduction

In this document I have implemented essential frequency domain techniques for improving images and image transforms. I have used python with matplotlib, numpy, cv2, math,scipy,itertools,PIL libraries.

II. IMAGE TRANSFORMS AND ENHANCEMENT

A. STEP I

Firstly, I are required to define read and write functions. I used imwrite, imread like previous take home exam.

B. STEP II-Implementation

Secondly, I applied Fourier transforms,Hadamard and Cosine which is frequency domain transforms, on three color channels for given images, respectively. In the frequency domain, a digital image is converted from spatial domain to frequency domain and image filtering is used for image enhancement for a specific application. Fourier transform image representation in frequency domain, spanned by complex exponential functions. In this assignment, For fourier transform implementation, I have imported fft2 from numpy library.Also I applied logarithm to observe the magnitude component of the Fourier transformed image better.

magnitude = 20*np.log(np.abs(shifted))

Hadamard transform is used for showing images spanned by integer valued basis functions. In hadamard transform instead of available python function , I defined recursion function which creates Hadamard matrix. Because of hadamard matrix is square,I arranged images as a square (size=(1024,1024)) for convolution . After that HMH^{-1} (H^{-1} is equal to HT,H is orthogonal) which is a Hadamard transform formula is applied to image matrices.Final part of first step, I applied dct function from openCV for discrete cosine transform.Discrete cosine transform enable us to represent images spanned by cosine functions.

$$F(\omega)=\int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt$$
$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega$$

Fig. 1. Fourier Transform formula

$$H_1=[1],$$
$$H_2=\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix},$$
$$H_4=\begin{bmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{bmatrix}.$$

Fig. 2. Hadamard matrix

Discrete Fourier/Cosine Transform

$$u_k(i)=\sum_{l=0}^{n-1}\mathbf{T}_n(i,l)\,x_{k-l} \qquad \forall i=0,\dots,n-1$$

where

$$\mathbf{T}_n(i,l)=\begin{cases}\mathbf{F}_n(i,l) = \sqrt{1/n}\,e^{j2\pi il/n} & (DFT)\\ \mathbf{C}_n(i,l) = \sqrt{2/n}\,K_i\cos\left(\frac{i(l+1/2)\pi}{n}\right) & (DCT)\end{cases} \qquad \forall i,l=0,\dots,n-1$$

with $K_i=1/\sqrt{2}$ for $i=0$ and 1 otherwise.

Fig. 3. Cosine Transform formula

C. STEP II- Results

Fig. 4. 1.png

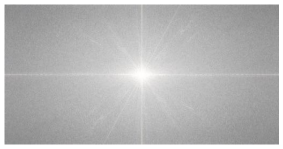


Fig. 5. F1.png

Fig. 6. 2.png

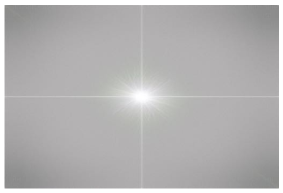


Fig. 7. F2.png

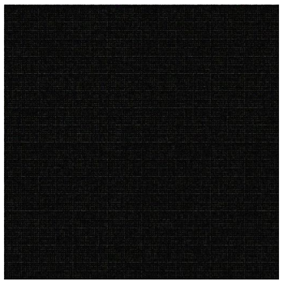


Fig. 8. Hadamard transform of image I

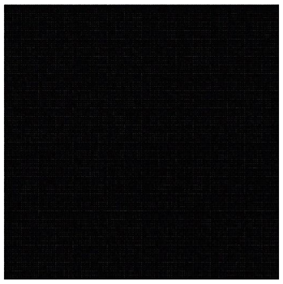


Fig. 9. Hadamard transform of image II

Fig. 10. Cosine transform of image I

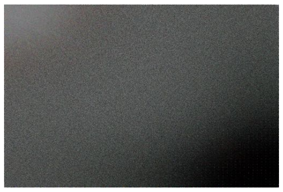


Fig. 11. Cosine Transform of image II

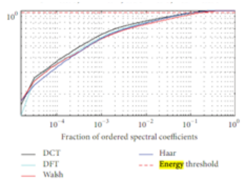


Fig. 12. Energy compaction

Energy compaction means that a big part of the total signal energy contained in a little coefficients

Different transforms have different ability energy compaction for different types of images.Studies show that DCT is in most applications advantageous than other transforms in terms of the energy compaction ability.Information is observable in left corner of image,other fields have zero values. Because of this DCT is used for image compression.compression of Hadamard transform is lower than Discrete Fourier and Discrete Cosine transforms. Fourier transform express periodical extension of signals, which may results in violent discontinuities at signal boundaries so require compact high frequency components to reproduce them. DCT also implies periodical extension, however extension of not the signal itself but of its extended copy . Thanks to such an “even”extension, the extended signal has no interruption at its boundaries, and at boundaries of the initial signal.

D. STEP III- Implementation

In this section, I applied low pass filtering methods which is Ideal low pass, gaussian low pass , second order butterworth low pass to image 3. Actually in low pass filter, I filtered image which filter whose values are higher close the center, and near to zero outside.Low-pass filter can be used only on the Fourier Transform of an image (frequency-domain image). Generally,,After applying low-pass filter,image become a blurred version of the original image.This means that higher frequency components are removed, and lower frequency components remained same as before. Stages are like that:

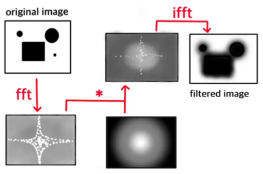


Fig. 13.

I implemented this formula for ideal low pass filtering:

$$H(u, v) = \begin{cases} 1 & \text{if } D(u, v) \leq D_0 \\ 0 & \text{if } D(u, v) > D_0 \end{cases}$$

Fig. 14.

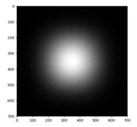


Fig. 15. Example of low pass filter

In addition , Gaussian filter and butterworth filter have a formula as I implemented this:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\left(x^2+y^2\right)/\left(2\sigma^2\right)}$$

Fig. 16. Gaussian filter formula

where σ is the standard deviation of the distribution. I have also assumed that the distribution has a mean of zero (i.e. it is centered on the line x=0). I assigned σ to 6 .

I also np.ravel and np.mgrid functions from numpy library for Gaussian filter.

$$H(u, v) = \frac{1}{1 + [D(u, v) / D_0]^{2n}}$$

Fig. 17. Butterworh low pass filter formula

I used this formula when applying Butterworth filter 0 means maximum distance from center of the circle. The n is order of Butterworth filter which controls thickness of the blacken of border in image.

E. STEP III- Results

In general, You can see that higher frequency component (outer side) are removed and dark in outputs.I used 30,60,90 cuttoff frequencies on same image. I observed when r increases , images are starting to look like the original.

The ideal low pass filter is used to cut off all the high-frequency components of Fourier transformation. Gaussian low pass filters have a smoother output.Butterworth filters have relatively flat maximum for large values of the filter order, n. If I compare smoothness between them, Butterworh is smoothest followed by Gaussian filter.Because butterworh has order and it increase flat.Also is used to remove high-frequency noise with very minimal loss of signal components. Ideal low pass filter has discontinuity which makes irregularities in filter so it has less smoothness.

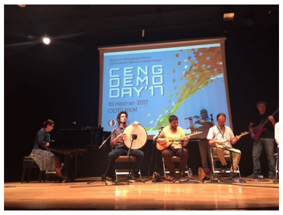


Fig. 18. 3.png

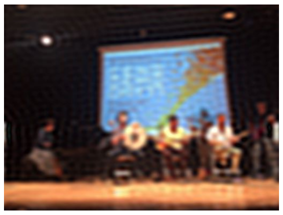


Fig. 19. ILP with r1=30

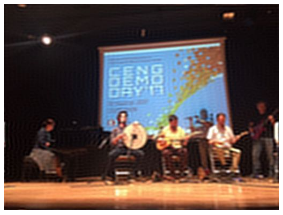


Fig. 20. ILP with r2=60

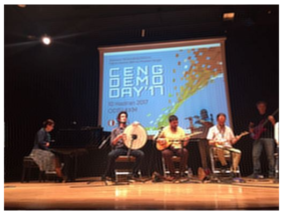


Fig. 21. ILP with r3=90

.

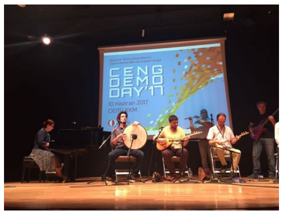


Fig. 22. 3.png

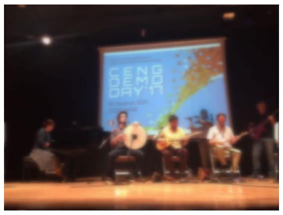


Fig. 23. BLP with r3=30

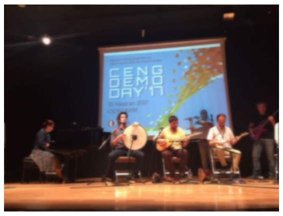


Fig. 24. BLP with r3=60

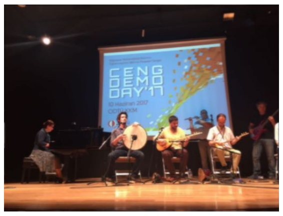


Fig. 25. BLP with r3=90

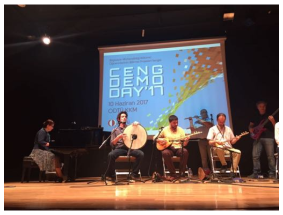


Fig. 26. r3.png

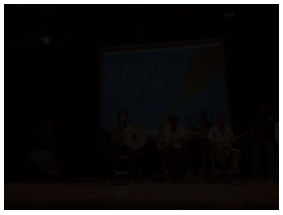


Fig. 27. GLP with r1=30

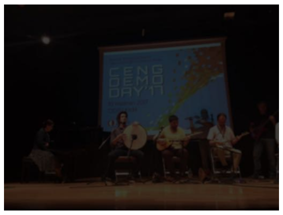


Fig. 28. GLP with r2=60

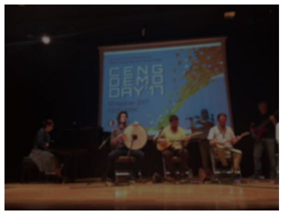


Fig. 29. GLP with r3=90

The concept of filtering and low pass remains the same, but only the transition becomes different and become more smooth.

F. STEP IV- Implementation

A high pass filter is used for sharpening and edge detection.It complements low pass filter and represents high frequency components.Ideal high pass filter which whose values are near the zero close the center, and higher outside.I know that:

$$H_{HP}(u,v) = 1 - H_{LP}(u,v)$$

Fig. 30.

I substract one from filters which I used in low pass filters after that I multiply with fourier transform of image 3 and inverse fourier transform with them.

G. STEP IV- Results

In butterworth and ideal high pass filter , image appears sharper. Also gaussian high pass filter shows emphasize edges but it is not suitable output in my implementation. In other high pass filters, image is dark but in gaussian there are a lot colors. Also I could not give same cut off frequency because compiler gives error when I run with 30,60,90. It is like memory error. In addition,in other implementations I can see that when cut of frequency increases, sharpness decreases.

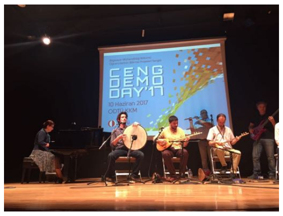


Fig. 31. 3.png

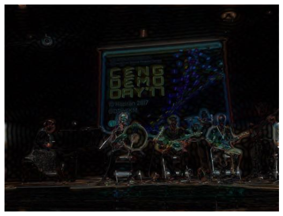


Fig. 32. IHP with r1=30

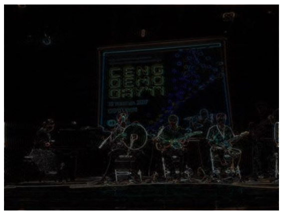


Fig. 33. IHP with r1=60

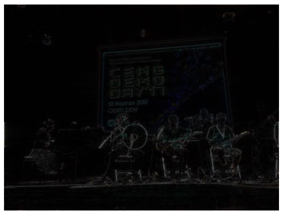


Fig. 34. IHP with r1=90

Fig. 35. 3.png

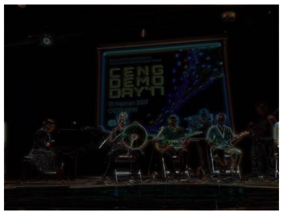


Fig. 36. BHP with r1=30

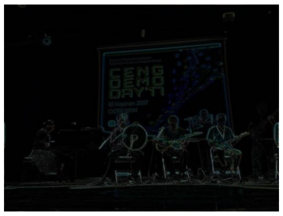


Fig. 37. BHP with r1=60

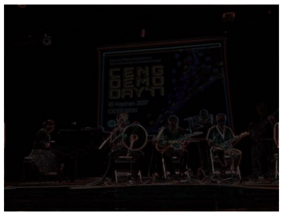


Fig. 38. BHP with r1=90

Fig. 39. 3.png

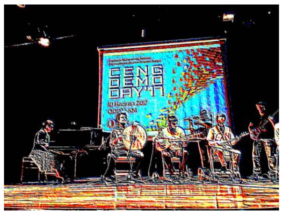


Fig. 40. GHP with r1=6

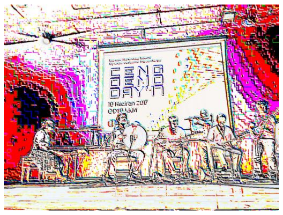


Fig. 41. GHP with r1=8

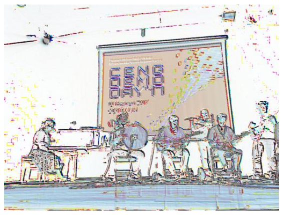
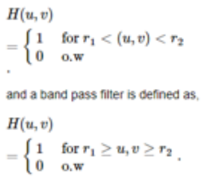


Fig. 42. GHP with r1=10

H. STEP V -implementation

In this step, I am required to apply band reject and band pass filter. They are about put constraints to pass frequency or reject. If threshold frequency band is filtered out, then, the band filter is called band reject. If the band is passed to the output of the filter, then, the band filter is called band pass. Those filters remove noises and give clear images. I applied those formula :



$$H(u,v) = \begin{cases} 1 & \text{for } r_1 < (u,v) < r_2 \\ 0 & \text{o.w} \end{cases}$$

and a band pass filter is defined as,

$$H(u,v) = \begin{cases} 1 & \text{for } r_1 \geq u,v \geq r_2 \\ 0 & \text{o.w} \end{cases}.$$

Fig. 43.

I guessed bandwidth numbers with taking fourier images but I could not appropriate bandwith for removing noises.

I. STEP V - Results

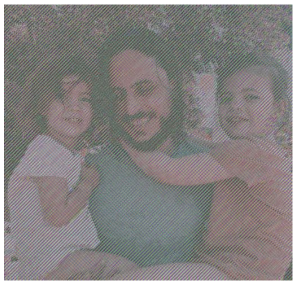


Fig. 44. Band reject filter in image IV

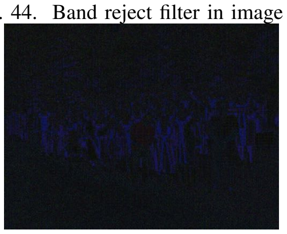


Fig. 45. Band pass filter in image IV

J. STEP VI -implementation

In final part, I applied Space Domain Filtering and Contrast Streching to images. I preferred median filtering, averaging,normalization(contrast streching), gaussian filtering. Median filtering is used for remove noise, gaussian filtering is prefferred for blurring and average filtering does smoothing.I used python cv2 library for all of them.For averaging, cv2.blur, for gaussian cv2.GaussianBlur, for median filtering, cv2.medianBlur and for contrast streching, I preferred using cv2.createCLAHE.

K. STEP V1 - Results

Images after some operations shows us our filters give some contrast and blur.

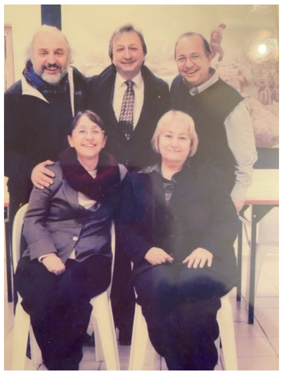


Fig. 46. 6.png

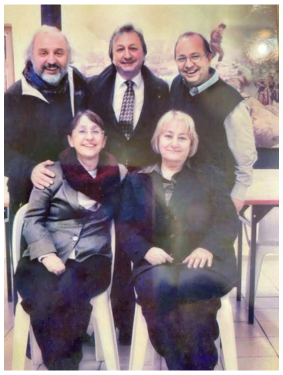


Fig. 47. image VI after filters

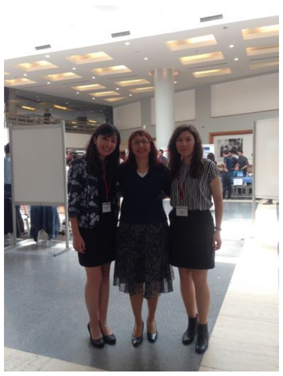


Fig. 48. 7.png

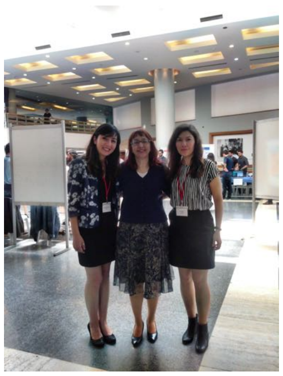


Fig. 49. image VII after filters