INDIAN INSTITUTE OF TECHNOLOGY

KHARAGPUR

DIGITAL IMAGE PROCESSING LABORATORY

A REPORT ON

EXPERIMENT 05

**Frequency Filtering and Discrete Cosine Transform**

|  |  |  |
| --- | --- | --- |
| Name : | Mayukh Roy | Srijita Saha Roy |
| Roll. No.: | 18EC65R06 | 18EC65R07 |

03.09.2018

Group No. 03

**DEPT OF ELECTRONICS AND ELECTRICAL COMMUNICATION**

**ENGINEERING**

**VISUAL INFORMATION AND EMBEDDED SYSTEMS**

**Table of Contents**

**Sl. No. Topic Page No.**

1. Introduction 1
2. Algorithm 3
3. Results 4
4. Analysis 9
5. References 10

**Introduction**

Frequency filters process an image in frequency domain. The image is Fourier transformed, multiplied with the filter function and then retransformed into the spatial domain. Attenuating high frequencies results in a smoother image in the spatial domain, attenuating low frequencies enhances the edges.

**Discrete Fourier Transform (DFT):** DFT converts a finite sequence of equally spaced samples of a function into a same-length sequence of equally-spaced samples of the discrete-time Fourier transform(DTFT), which is a complex valued function of frequency.

**Ideal LPF:** A low-pass filter(LPF) is a filter that passes signals with a frequency lower than a selected cut-off frequency an attenuates signals with frequencies higher than the cutoff frequency. The exact frequency response of the filter depends on the filter design. The transfer function for the ideal LPF is given by:

H(u,v)=

**Ideal HPF**: A high-pass filter (HPF) is an electronic filter that passes signals with a frequency higher than a certain cutoff frequency and attenuates signals with frequencies lower than the cutoff frequency. The amount of attenuation for each frequency depends on the filter design. A high-pass filter is usually modelled as a linear time-invariant system. . The transfer function for the ideal LPF is given by:

H(u,v)=

**Gaussian LPF:** Gaussian LPF is Gaussian blur, attenuating high frequency signals. Its amplitude Bode Plot (the log scale in frequency domain) is a parabola. It is non-causal which means the filter window is symmetric about the origin in the time-domain. The transfer function of a Gaussian LPF is defined as:

*H(u,v)=*

**Gaussian HPF:** Gaussian HPF is given by:

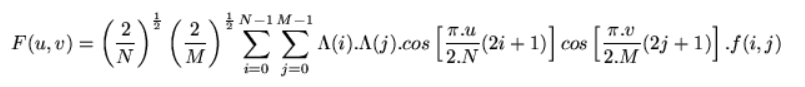
*H(u,v)= 1-*

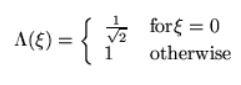
**Butterworth LPF:** Butterworth filter is a type of signal processing filter designed to have a frequency response as flat as possible in the passband. The transfer function of a Butterworth low pass filter of order n with cutoff frequency at distance from the origin is defined as:

*H(u,v)=*

**Butterworth HPF:**The transfer function of a Butterworth high pass filter of order n with cutoff frequency at distance from the origin is defined as:

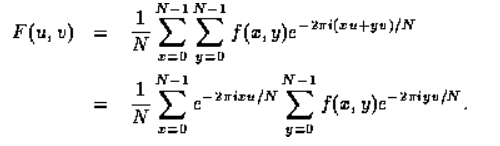
*H(u,v)=*

**Discrete Cosine Transform (DCT):** DCT expresses a finite sequence of data points in terms of a sum of cosine functions oscillating at different frequencies. In particular, a DCT is a Fourier-related transform similar to the discrete Fourier Transform, but using only real numbers. The formula is given as:



**Algorithm**

**FFT:**

**Step 1:** Rewrite DFT equation as:

The right hand sum is basically just a one-dimensional DFT if *x* is held constant. The left hand sum is then another one-dimensional DFT performed with the numbers that come out of the first set of sums.

So, we can compute a two-dimensional DFT by

* performing a one-dimensional DFT for each value of *x*, ***i.e.*** for each column of *f*(*x*,*y*), then
* performing a one-dimensional DFT in the opposite direction (for each row) on the resulting values.

This requires a total of 2-*N* one dimensional transforms, so the overall process takes O(N2logN) time.

**Step 2:** Divide sequence into N/2 blocks. Keep on dividing recursively to reach N=2. Perform FFT on smallest block. Perform up conversion.

**DCT:**

**Step 1:** Divide image in to 8x8 blocks.

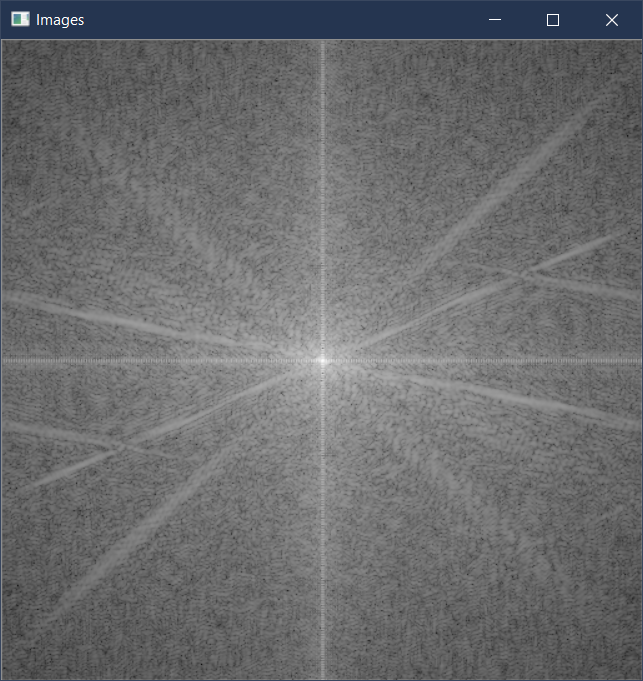
**Step 2:** Perform DCT on those blocks.

**Step 3:** Compress image based on threshold.

**Step 4:** Perform inverse DCT.

**Results**

**FFT Spectrum:**



**Ideal LPF:**

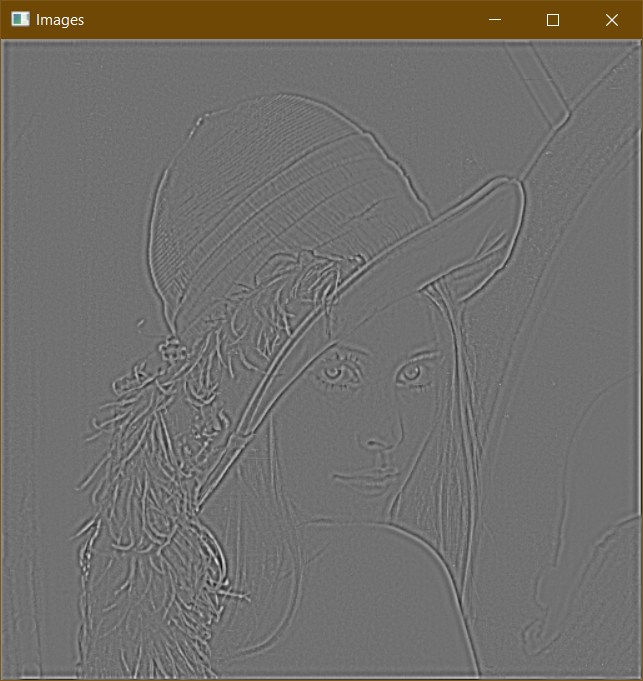


**Ideal HPF:** 



******Gaussian LPF:**

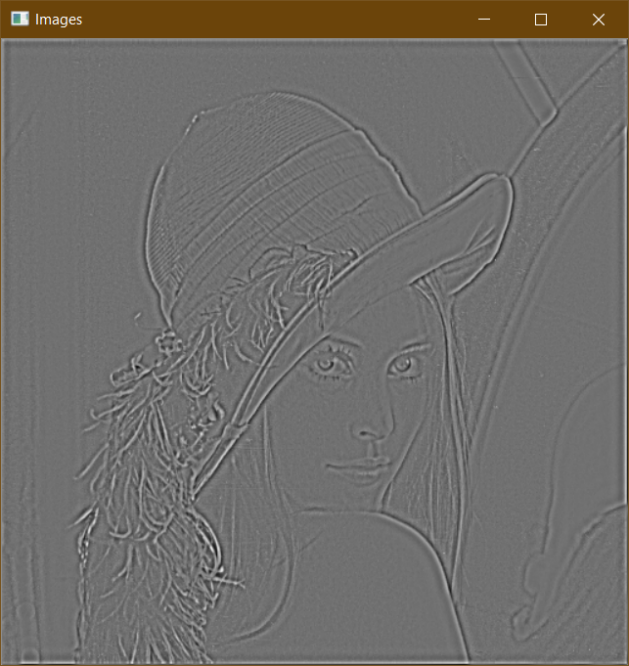
**Gaussian HPF:**

****

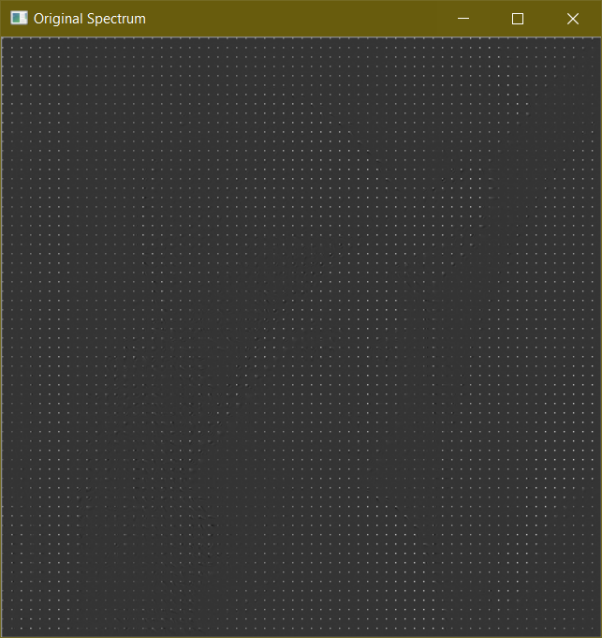
**Butterworth LPF:**

****

**Butterworth HPF:**

****

**DCT:**

**Spectrum:**

**Uncompressed Image after inversion:**

**Highly Compressed Image:**

**Analysis**

* Frequency filtering is a general process as it processes the entire image at once and ignores the presence of any object regions in the image.
* Multiplication operation is easier to perform than convolution. Hence frequency domain filtering is easier to perform.
* We see that the low pass filtered output in Gaussian and Butterworth LPF is better than ideal LPF because the mask used in case of the ideal LPF is very abrupt, whereas in the other the higher frequencies are gradually attenuated.
* In HPF also, we see that the edges are more prominent in the outputs and hence they are quite useful in edge detection.
* LPFs are mainly used to remove noise and remove details in the image.
* If HPF is applied to a noisy image, then the noise gets amplified. So various denoising techniques should be performed first before applying HPF to it.
* We see from DCT spectrum, that most of the energy is confined in the low frequency regions. Hence, by truncating the high frequency components in the image, effective compression can be done easily without reducing quality of the image.
* The DCT operation is easier to implement on hardware since, they only consider cosine terms and no complex math is required to implement DCT.
* We see that the compressed image is almost visually same as the original image. However, the PSNR value decreases as we gradually increase the compression of the image. This shows that the quality of the image is reducing even though it is not perceivable visually.
* By performing DCT in patches, localized compression can be done without reducing the quality of the image to a great extent. However, the amount of compression that can be done reduces also in this case.

**Contributions:**

**Mayukh Roy: Frequency Domain Filters, DCT**

**Srijita Saha Roy: FFT Implementation**

**References**

1. “18.11.1.2 Algorithms (2D FFT).” OriginLab Corporation - Data Analysis and Graphing Software - 2D Graphs, 3D Graphs, Contour Plots, Statistical Charts, Data Exploration, Statistics, Curve Fitting, Signal Processing, and Peak Analysis, www.originlab.com/doc/Origin-Help/FFT2-Algorithm.
2. Author, Retired Author Retired. “2D FFT of an Image in C#.” CodeProject - For Those Who Code, CodeProject, www.codeproject.com/Articles/44166/2D-FFT-of-an-Image-in-C.
3. “Fast Fourier Transform.” Wikipedia, Wikimedia Foundation, en.wikipedia.org/wiki/Fast\_Fourier\_transform.
4. “The Discrete Cosine Transform (DCT).” Remote Procedure Calls (RPC), users.cs.cf.ac.uk/Dave.Marshall/Multimedia/node231.html.
5. “The Fast Fourier Transform (FFT) Algorithm.” AllSignalProcessing.com, 12 June 2015, allsignalprocessing.com/fast-fourier-transform-fft-algorithm/.