M.Sc. (Five Year Integrated) in Computer Science
(Artificial Intelligence & Data Science)

Fourth Semester

Assignment

23-813-0403: DIGITAL SIGNAL PROCESSING

Submitted in partial fulfillment
of the requirements for the award of degree in
Master of Science (Five Year Integrated)
in Computer Science (Artificial Intelligence & Data Science) of
Cochin University of Science and Technology (CUSAT)
Kochi

Submitted by

ALINA GEORGE
(81323004)

DEPARTMENT OF COMPUTER SCIENCE
COCHIN UNIVERSITY OF SCIENCE AND TECHNOLOGY (CUSAT)
KOCHI-682022

MARCH 2025

Contents

ABSTRACT

This project aims to explore the concept of Parseval's energy preservation in the context of the Fourier Transform. By applying Parseval's theorem, the project seeks to establish the equivalence of energy in both time and frequency domains.

This concept is particularly significant in signal processing, where it ensures accurate energy representation and analysis. The theorem is applicable in both continuous and discrete Fourier Transforms (DFT), playing a vital role in validating computational algorithms and maintaining energy conservation. Understanding Parseval's energy preservation property is essential for various applications, including audio processing, image compression, and communication systems, where efficient and accurate signal representation is crucial.

This paper also explores magnitude spectrum analysis using the 2D Fourier Transform, highlighting the distribution of frequency components in both normal and centered frequency domains. By applying logarithmic scaling, the visibility of low and high frequencies is enhanced, aiding in the analysis of an image's frequency characteristics through visual comparison.

PARSEVAL ENERGY CONSERVATION

THEOREM:

Parseval's theorem is a fundamental principle in Fourier analysis that establishes the equivalence of energy in the time and frequency domains. It states that the total energy of a signal, represented as the integral of its squared magnitude in the time domain, is equal to the integral of the squared magnitude of its Fourier Transform in the frequency domain.

$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2$$

To implement and verify Parseval's energy preservation property using the Discrete Fourier Transform (DFT), follow these steps:

Step 1: Signal Generation

Create a sample signal using a simple function such as a sine wave, square wave, or a combination of signals.

Step 2: Compute the Energy in Time Domain

Calculate the energy of the signal in the time domain using the formula:

$$E_t = \sum_{n=0}^{N-1} |x[n]|^2$$

where x[n] represents the signal samples, and N is the total number of samples.

Step 3: Perform the Fourier Transform

Apply the DFT to convert the signal from the time domain to the frequency domain using the formula:

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-\frac{j2\pi kn}{N}}$$

Step 4: Compute the Energy in Frequency Domain

Calculate the energy in the frequency domain using:

$$E_f = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2$$

1. E_t = Energy in the time domain
2. $x[n]$ = Discrete-time signal values
3. N = Total number of samples
4. $|x[n]|^2$ = Magnitude squared of each signal value

Step 5: Verification of Parseval's Theorem

Compare the computed energies from both domains using Parseval's theorem

PROGRAM

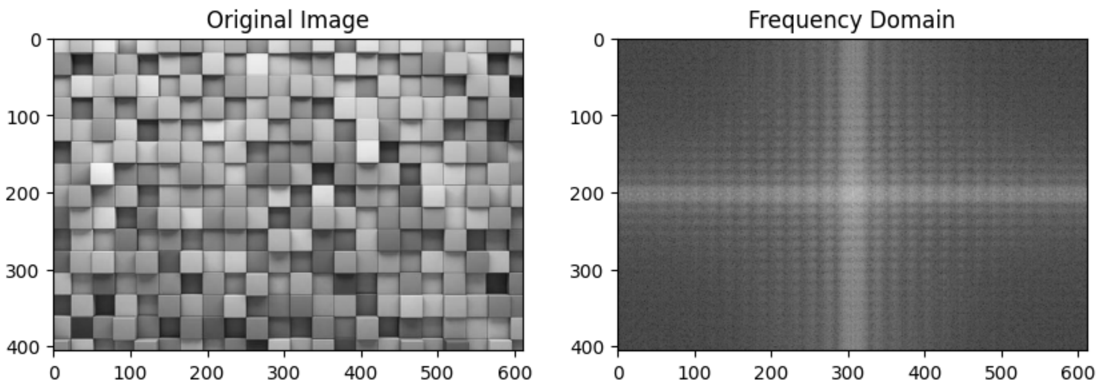
```
from google.colab import files
import numpy as np
import matplotlib.pyplot as plt
from scipy.fft import fft2, fftshift
import matplotlib.image as mpimg

# Upload image
uploaded = files.upload()

# Assuming you upload a file named 'test.png', use the uploaded file n
image_path = list(uploaded.keys())[0]  # Get the first uploaded file n
```

```
# Load the image (if it's a color image, it will be converted to grays
img = mpimg.imread(image_path)

# If the image is colored (e.g., RGB), convert it to grayscale
if img.ndim == 3:
    img = np.mean(img, axis=-1)  # Convert to grayscale by averaging t

# Compute 2D Fourier Transform (Frequency domain)
F_uv = fft2(img)

# Shift the zero-frequency component to the center
F_uv_shifted = fftshift(F_uv)

# Calculate energy in the spatial domain
energy_spatial = np.sum(np.abs(img)**2)

# Calculate energy in the frequency domain
energy_frequency = np.sum(np.abs(F_uv_shifted)**2) / img.size

# Print the results
print(f"Energy in spatial domain: {energy_spatial}")
print(f"Energy in frequency domain: {energy_frequency}")

# Verify Parseval's theorem (energy conservation)
assert np.isclose(energy_spatial, energy_frequency), "Energy conservat

# Optionally, visualize the image and its frequency domain representat
plt.figure(figsize=(10, 5))

plt.subplot(1, 2, 1)
plt.imshow(img, cmap='gray')
```

```
plt.title('Original Image')

plt.subplot(1, 2, 2)
# Visualize the log of the absolute value of the shifted frequency dom
plt.imshow(np.log(np.abs(F_uv_shifted) + 1), cmap='gray')  # Log scale
plt.title('Frequency Domain')

plt.show()
```

Result

```
Choose Files tiles.jpg
• tiles.jpg(image/jpeg) - 57207 bytes, last modified: 2/6/2025 - 100% done
Saving tiles.jpg to tiles (1).jpg
Energy in spatial domain: 5388984599.666666
Energy in frequency domain: 5388984599.666671
```



CONCLUSION

- **Energy in the spatial domain:** Represents the total energy of the image (sum of pixel intensities squared).
- **Energy in the frequency domain:** Represents the total energy after transforming the image to the frequency domain and summing the squared magnitudes of the Fourier coefficients.
- **Verification:** If the assertion doesn't raise an error, the energy conservation law is satisfied.

MAGNITUDE SPECTRUM ANALYSIS

The magnitude spectrum is a critical component of Fourier Transform analysis, representing the intensity of different frequency components of an image. By applying the 2D Fourier Transform to an image, we can observe the distribution of frequency components in both normal and centered frequency domains. The normal frequency spectrum places the zero-frequency component at the top-left corner, while the centered frequency spectrum shifts the zero-frequency component to the center of the image. In this analysis, a logarithmic scale is applied to the magnitude spectrum to enhance the visibility of both low and high-frequency components.

$$\text{Magnitude Spectrum} = |F(u, v)| \tag{1}$$

where:

- $F(u, v)$ is the Fourier Transform of the image.
- $|F(u, v)|$ represents the magnitude of the Fourier Transform coefficients, capturing the intensity of different frequency components.

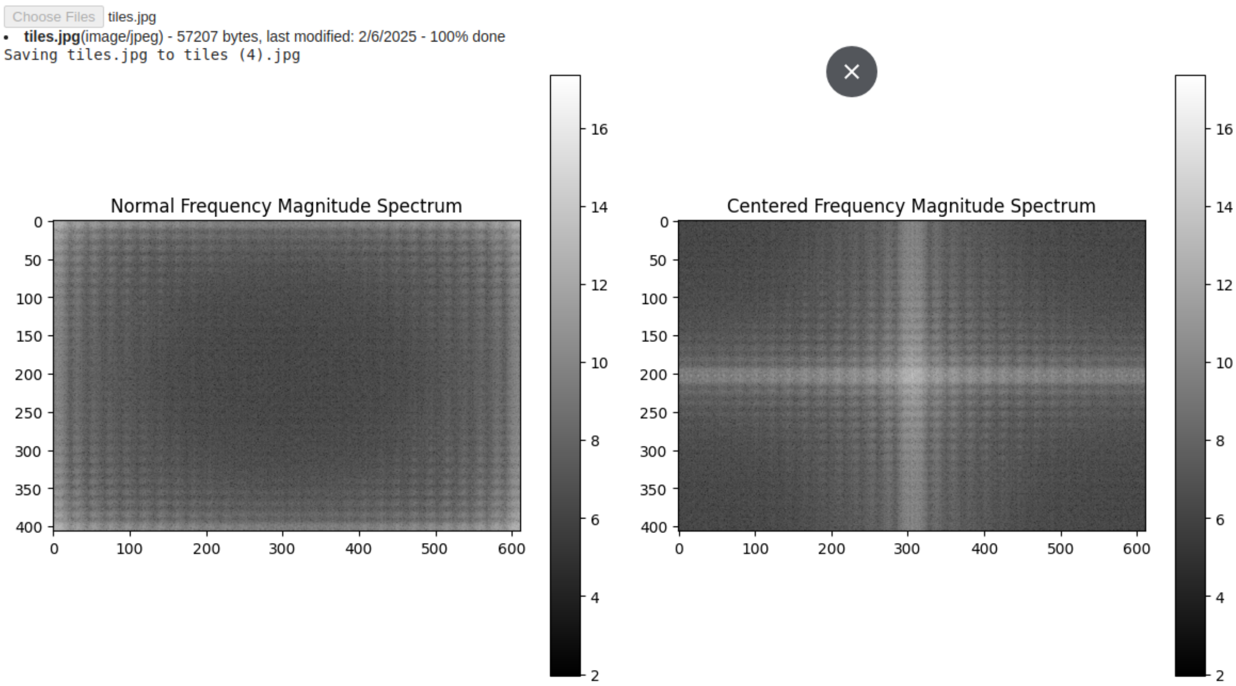
This analysis allows us to visually explore the frequency characteristics of the image by examining both the normal and centered magnitude spectrums.

PROGRAM

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.fft import fft2, fftshift
import matplotlib.image as mpimg
from google.colab import files

# Upload image
uploaded = files.upload()

# Assuming you upload a file named 'test.png', use the uploaded file name
image_path = list(uploaded.keys())[0]  # Get the first uploaded file name

# Load the image
img = mpimg.imread(image_path)

# If the image is colored (e.g., RGB), convert it to grayscale
```

```
if img.ndim == 3:
    img = np.mean(img, axis=-1)  # Convert to grayscale by averaging the channels

# Compute 2D Fourier Transform (Frequency domain)
F_uv = fft2(img)

# Compute magnitude spectrum (normal frequency rectangle, no shift)
magnitude_spectrum_normal = np.abs(F_uv)

# Compute magnitude spectrum (centered frequency rectangle,
shift zero-frequency component)
F_uv_shifted = fftshift(F_uv)
magnitude_spectrum_centered = np.abs(F_uv_shifted)

# Plot both the magnitude spectrums
plt.figure(figsize=(12, 6))

# Plot the normal (uncentered) frequency spectrum
plt.subplot(1, 2, 1)
# Log scale for better visualization
plt.imshow(np.log(magnitude_spectrum_normal + 1), cmap='gray')
plt.title('Normal Frequency Magnitude Spectrum')
plt.colorbar()

# Plot the centered frequency spectrum
plt.subplot(1, 2, 2)
# Log scale for better visualization
plt.imshow(np.log(magnitude_spectrum_centered + 1), cmap='gray')
plt.title('Centered Frequency Magnitude Spectrum')
plt.colorbar()

plt.tight_layout()
plt.show()
```

RESULT



CONCLUSION

- **Magnitude Spectrum Normal:** Represents the frequency components of the image with the zero-frequency component at the top-left corner of the image.
- **Magnitude Spectrum Centered:** Represents the frequency components with the zero-frequency component shifted to the center of the image.
- **Logarithmic Scaling:** Used to enhance the visibility of both low and high-frequency components in the spectrum.
- **Visualization:** By comparing both frequency spectrums, we can analyze the frequency distribution and the image's overall frequency characteristics.