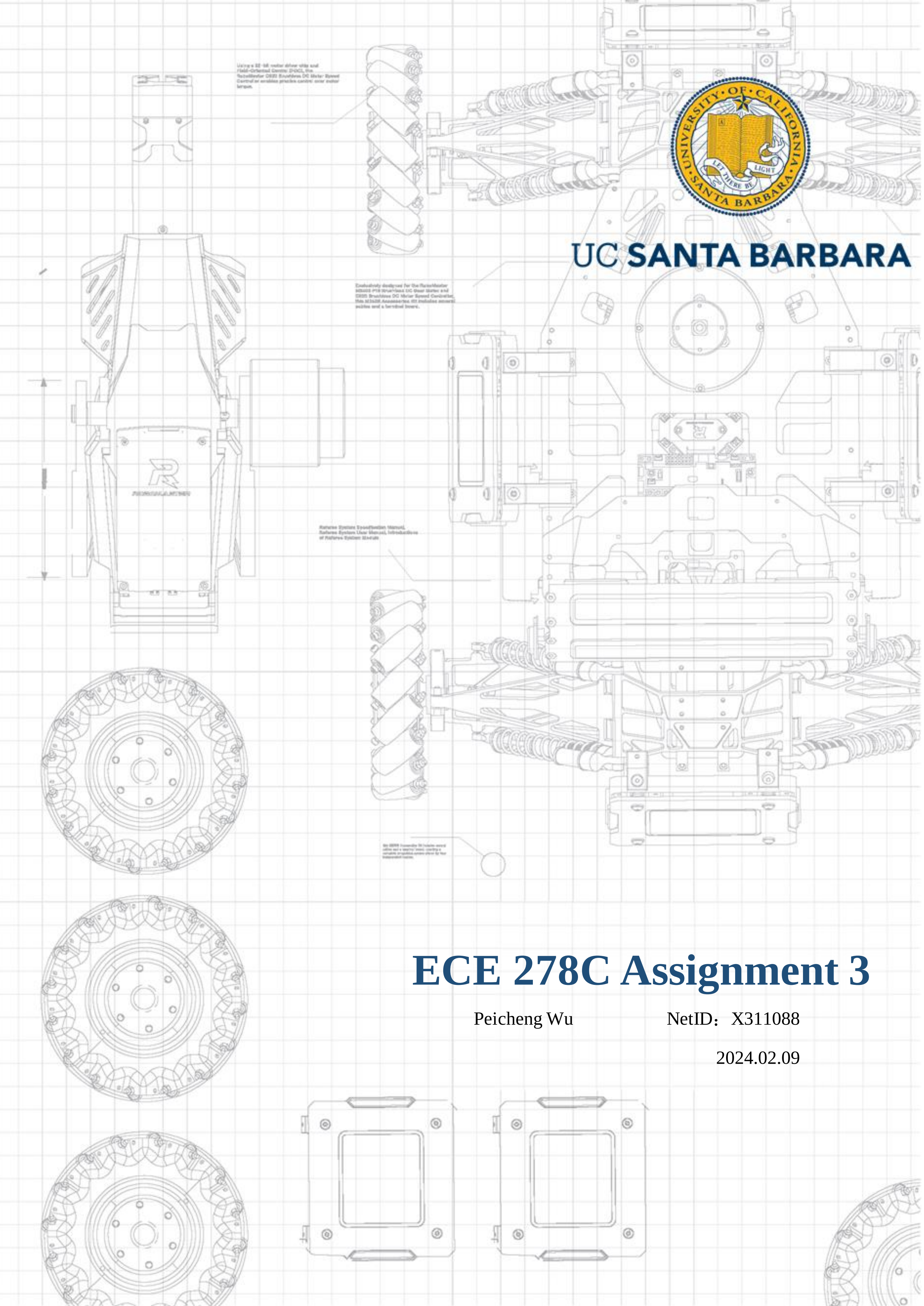

UC SANTA BARBARA

ECE 278C Assignment 3

Peicheng Wu NetID：X311088

2024.02.09

Catalog

1. Platform: VScode

Visual Studio Code, often referred to as VSCode, is a free and open-source code editor developed by Microsoft. It's renowned for its lightweight design, speed, and robust capabilities. Available for Windows, macOS, and Linux, VSCode offers built-in Git integration, an integrated terminal, and a debugger. Its power lies in its extensibility, with a vast marketplace of extensions that add support for various programming languages, debuggers, and tools. The editor's appearance and behavior are highly customizable, and its Intellisense feature provides smart code completions. Thanks to its open-source nature, it has a vibrant community that continually contributes to its development and enhancement.

2. Question

Consider a single centered point source,

scatter	*scatter location*
(x_0, y_0)	$(0,\ 0)$

The horizontal receiver aperture is organized in the form of a centered linear receiver array with a span of 60λ (from x= -30λ to x = +30λ). This horizontal linear receiver array is placed at

$$y = y_0 + 60\,\lambda$$

With quarter-wavelength spacing (λ/4) spacing, there are 241 complex wavefield data samples in total over the 60λ-long linear aperture.

- Repeat the exercise in Assignment 2 with only the phase information of the received waveform data, by setting the magnitude of the data to a constant.
- Repeat the exercise with only the phase information of the received waveforms and phase-only integration kernel (Green’s function) for image reconstruction.
- Plot the magnitude distribution of your reconstructed images.

3. Solutions

In the field we have equation:

$$g(x,y,z) = s(x,y,z) * h(s,y,z)$$

And now we just assume source as the impulse function:

$$s(x,y,z) = \delta(x,y,z)$$

Thus, for each receiver, we can write the received signal

$$g(x,y,z) = \frac{1}{j\lambda r} exp\left(\frac{j2\pi r}{\lambda}\right)$$

According the backward propagation:

$$\dot{s}(x,y,z) = g(x,y,z) * h^{*}(x,y,z)$$

$$= \iiint \left[\frac{1}{j\lambda r} exp\left(\frac{j2\pi r}{\lambda}\right)\right]\left[\frac{-1}{j\lambda r'} exp\left(\frac{-j2\pi r'}{\lambda}\right)\right] dx' \ dy' \ dz'$$

$$= \iiint \frac{1}{\lambda^2 r r'} exp\left(\frac{j2\pi\left(r-r'\right)}{\lambda}\right) dx' \ dy' \ dz'$$

Where r means the distance from source scatter to the receiver and r' means the distance from source region point to the receiver.

So we can just write the equation of the reconstructed image:

$$\dot{s}(x,y,z) = \iiint \frac{1}{\lambda^2 r r'} exp\left(\frac{j2\pi\left(r-r'\right)}{\lambda}\right) dx' \ dy' \ dz'$$

Then we can use this equation calculate all the points in the source region.

And when it is **phase only term**, the equation will be:

$$\dot{s}(x,y,z) = \iiint \frac{-1}{j\lambda r} exp\left(\frac{j2\pi\left(r-r'\right)}{\lambda}\right) dx' \ dy' \ dz'$$

Also, we can write the full phase only term equation:

$$\dot{s}(x,y,z) = \iiint exp\left(\frac{j2\pi r}{\lambda}\right) \times exp\left(\frac{j2\pi r'}{\lambda}\right) dx'dy'dz'$$

4. Results

4.1 Reconstructed images

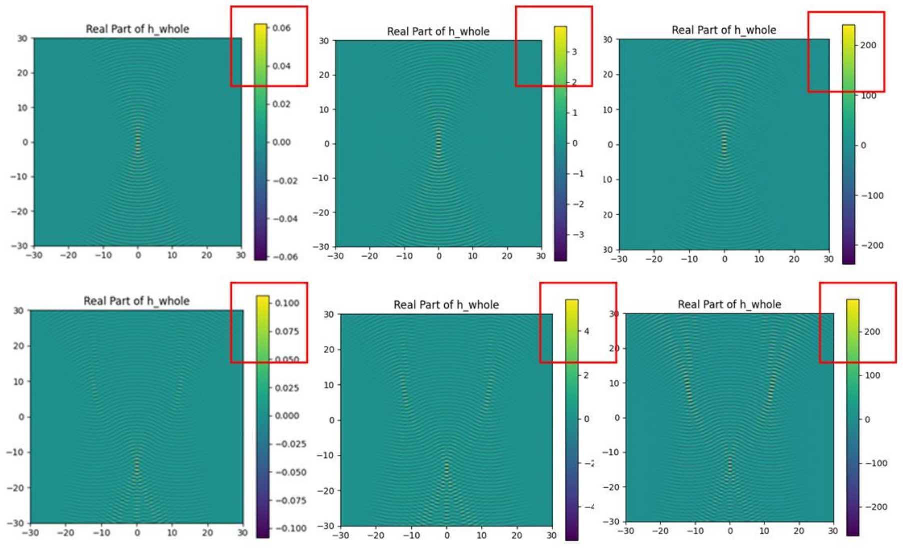

Fig.1 Reconstructed image

The image has been constructed by linear receivers. Although we can see the pattern of image is the same, the intensity is totally different. The normal Green's function intensity is only **0.06**, only phase term intensity is **3** and full only phase term intensity is more than **200**. This also shows in the magnitude distribution.

4.2 Magnitude distribution

For the magnitude, we can find more interesting things:

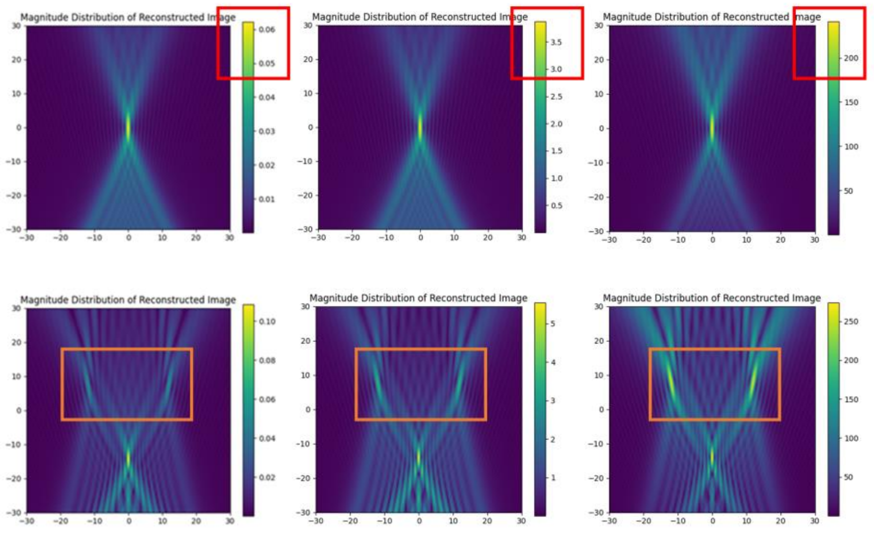

Fig.2 Magnitude distribution of reconstruction image

We can see the fig.5, from left to right is normal term, phase only term and full phase only term. The intensity also increases from left to the right. Notably, full phase only term **source point intensity seems same**, while normal term and phase term source **point intensity is different**.

4.3 FFT: Frequency domain

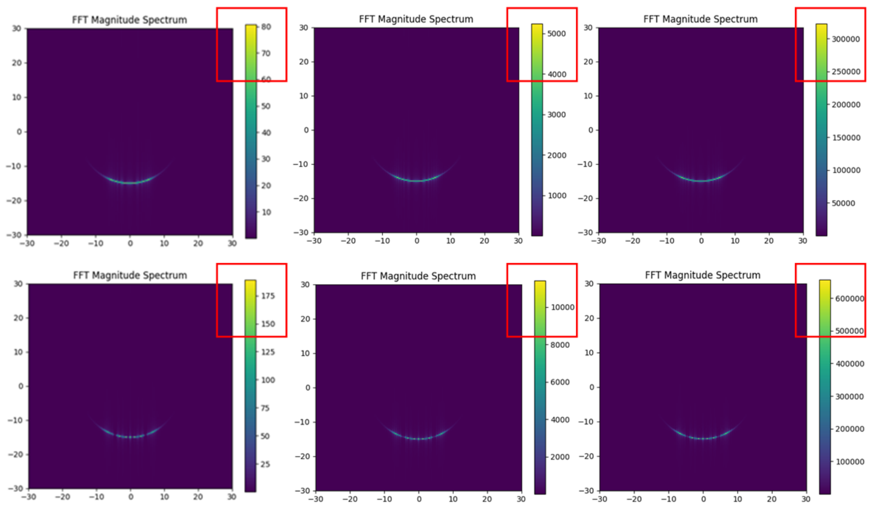

Fig.3 Magnitude distribution and FFT

In the frequency domain, we can see the same thing in the magnitude distribution. The normal term, phase only term and full phase term have different intensity. Because the FFT is like a **convolution process**, so the intensity changes **very rapidly**.

5. Appendix

```python
import numpy as np
import matplotlib.pyplot as plt
from scipy.fft import fft2, fftshift
import argparse

def generate_receivers_locations(lambda_0, y_offset, span):
    delta_x = lambda_0 / 4
    num_points = int(span / delta_x) + 1
    x_positions = np.linspace(-span / 2, span / 2, num_points)
    scatter_locations = [(x, y_offset) for x in x_positions]
    # print(scatter_locations)
    return scatter_locations

def generate_source_region(lambda_0, X_length, Y_length):
    delta_x = lambda_0 / 4
    num_points_x = int(X_length / delta_x) + 1
    num_points_y = int(Y_length / delta_x) + 1
    x_positions = np.linspace(-X_length / 2, X_length / 2, num_points_x)
    y_positions = np.linspace(-Y_length / 2, Y_length / 2, num_points_y)
    source_locations = [(x, y) for y in y_positions for x in x_positions]
    # print(source_locations)
    return source_locations

def calculate(source_scatter):

    lambda_0 = 1
    X_length = 60 * lambda_0
    Y_length = 60 * lambda_0
    y_offset = 60 * lambda_0
    span = 60 * lambda_0
    source_locations = generate_source_region(lambda_0, X_length, Y_length)
    receivers_locations = generate_receivers_locations(lambda_0, y_offset, span)

    num_points_x = int(X_length / (lambda_0 / 4)) + 1
    num_points_y = int(Y_length / (lambda_0 / 4)) + 1
    h_whole = np.zeros((num_points_y, num_points_x), dtype=complex)
    temp = np.zeros((num_points_y, num_points_x), dtype=complex)

    for x_cc, y_cc in source_scatter:
        for idx, (x_s, y_s) in enumerate(source_locations):
            y_idx = idx // num_points_x
            x_idx = idx % num_points_x
            h = 0
            for x_g, y_g in receivers_locations:
                r_g = np.sqrt((x_g - x_cc)**2 + (y_g - y_cc)**2)
                r_s = np.sqrt((x_s - x_g)**2 + (y_s - y_g)**2)
                if args.phase_only:
                    h_n = (-1/(1j*r_s)) * np.exp(1j * 2 * np.pi * (r_g - r_s) / lambda_0)
                elif args.full_phase_only:
                    h_n = np.exp(1j * 2 * np.pi * (r_g) / lambda_0) * np.exp(-1j * 2 *
np.pi * (r_s) / lambda_0)
                h += h_n
            h_whole[y_idx, x_idx] = h
        temp += h_whole
    return temp


if __name__ == "__main__":
    parser = argparse.ArgumentParser(description='Different question in the
assignment')
    parser.add_argument('--Q1', action='store_true', help='Process one target')
    parser.add_argument('--Q2', action='store_true', help='Process three target')
    parser.add_argument('--phase_only', action='store_true', help='by setting the
magnitude of the data to a constant')
    parser.add_argument('--full_phase_only', action='store_true', help='phase-only
integration kernel')
    args = parser.parse_args()
    if args.Q1:
        source_scatter = [(0, 0)]
    elif args.Q2:
        source_scatter = [(0, 15), (-12, -9), (12, -9)]

    h_whole = calculate(source_scatter)

    plt.figure(figsize=(12, 5))
    plt.subplot(1, 2, 1)
    plt.imshow(h_whole.real, extent=(-30, 30, -30, 30))
    plt.title('Real Part of h_whole')
    plt.colorbar()
    plt.subplot(1, 2, 2)
    plt.imshow(h_whole.imag, extent=(-30, 30, -30, 30))
    plt.title('Imaginary Part of h_whole')
    plt.colorbar()
    plt.show()

    fft_image = fftshift(fft2(h_whole))

    plt.figure(figsize=(6, 5))
    plt.imshow(np.abs(fft_image), extent=(-30, 30, -30, 30))
    plt.title('FFT Magnitude Spectrum')
    plt.colorbar()
    plt.show()

    plt.figure(figsize=(6, 5))
    plt.imshow(np.abs(h_whole), extent=(-30, 30, -30, 30))
    plt.title('Magnitude Distribution of Reconstructed Image')
    plt.colorbar()
    plt.show()
```

You can also find the code on the github: UCSB-ECE-278C/assignment3 at main · percyance/UCSB-ECE-278C (github.com)

When you want to run: please type python assignment2.py --Q1 --phase_only

Q1 means one target, Q2 means three targets

phase_only and full_phase_only is choose different term.