School of Engineering and Physical Sciences
Electrical Electronic and Computer Engineering

B31XM – Matlab Lab Part 1
Fourier Theory. Image Analysis in Frequency Domain

.

1. Objectives:

We will investigate the notion of spectrum and simple filtering in the frequency domain. As we will see, in all cases, we can interpret the effect of these filters in the frequency domain easily while their effect is not obvious in the time domain.

During this session, you will learn & practice:

1- More programming in matlab
2- More examples of functions and scripts
3- Fourier transform and its applications
4- Simple Filtering techniques
5- Phase information and its relation to image structure

2. Resources required:

In order to carry out this session, you will need to download images and matlab files from VISION posted under assessment/Lab1. Please download the zip files containing the following images:

- Image of Lena
- Landsat image
- sonar image
- sar image

We will also use the circuit and flowers images in Matlab (circuit.tif, peppers.png).
You are of course encouraged to try these programs out on images of your choice.

Please download the following functions and script from the zip file in VISION:

- `fft2d.m`
- `ifft2d.m`
- `phase_only.m`
- `random_phase.m`
- `random_magnitude.m`
- `quant_fft.m`
- `move_image.m`
- `SimpleFiltering.m`

Experiment 1:. **Phase and magnitude manipulation:**

Download the `fft2d` program, the `ifft2d` program and the `phase_only` programs.

As you have already seen in class, the phase of the spectrum is tightly related to the structure of the image. This will be illustrated in the following.

- Load an image (for example `circuit.tif`).
- Display the image in a figure;
- Now use the `phase_only` function (`help phase_only`).
- Comments?

The function `random_phase` calculates the FFT of an image, and leaves the magnitude unchanged but replaces the phase with random values. The inverse Fourier transform is then performed to re-create the image. Use this function to investigate the effect of replacing the phase with random values.
For example, try:-

```
ima = imread('lena.tif');
figure(1);
colormap(gray);
imagesc(ima);
figure(2);
colormap(gray);
ima_out=random_phase(ima);
```

What happens if you repeat the above, but use the images `sonar.tif` or `sar.tif`? What can you conclude about these images?

The function `random_magnitude` calculates the FFT of an image and leaves the phase unchanged, but replaces the magnitude with random values. The inverse Fourier transform is then performed to re-create the image. Use this function to investigate the effect of replacing the magnitude with random values

For example, try:-

```
ima = imread('lena.tif');
figure(1);
colormap(gray);
imagesc(ima);
figure(2);
colormap(gray);
ima_out=random_magnitude(ima);
```

What can you conclude from the above experiments?

What happens if you repeat the above, but use the images `sonar.tif` or `sar.tif`? What can you conclude about these images?

Experiment 2.1: Simple Fourier Filtering:

The high frequencies of the spectrum correspond to the edges while the low frequencies corresponds to the 'filling'. An easy way to demonstrate that is to take the Fourier transform of an image, cut the high (resp low) frequencies, and Fourier transform back the result. Try it using the `SimpleFiltering` program.
For those of you who have done filtering before, what are the issues involved with this type of filtering? Is the fact the we are using only N samples to calculate the Fourier transform an issue?

Experiment 2.2: Developing Fast-Fourier Transform

The purpose of this project is to develop a 2-D FFT program "package". Your implementation must have the capabilities to:

(a) Multiply the input image by $(-1)^{x+y}$ to center the transform for filtering.

(b) Compute the 2-D DFT and the spectrum

(b) Multiply the resulting (complex) array by a real filter function (in the sense that the the real coefficients multiply both the real and imaginary parts of the transforms). And compute the spectrum.

(c) Compute the inverse 2-D Fourier transform.

(d) Multiply the result by $(-1)^{x+y}$ and take the real part.

(e) Display the image

Experiment 2.3: Fourier Spectrum and Average Value

(a) Use image “imageA” to compute its (centered) Fourier spectrum.

(b) Display the spectrum.

(c) Use your result in (a) to compute the average value of the image.

Experiment 3: Compression and DCT

As we now know, the FFT of an image (generally real) is a complex number. We can extract the phase and the magnitude of the spectrum. The numbers representing them normally have a finite precision on computers. We will now investigate whether this affects the results and how.
Compression algorithms tend to quantize the spectrum or some features related to the spectrum (for example Discrete Cosine Transform for JPEG).

Therefore, we will investigate how quantization affects the image.
Is the phase more sensitive to quantization than the magnitude?

You can explore this using the `quant_fft` program or other quantization algorithms in matlab. Load an image (for instance `circuit.tif`) and apply the algorithm to this image by varying the parameters of the function. For example, play with the number of levels on which you quantify the magnitude and the phase of the spectrum. Comments?

Submit the report at Vision Week 5.