Introduction

In this assignment you'll learn about Fourier domain image operations and explain their effects on the images. You'll compute the 2-D Fourier Transform of an image, perform some operations, then take the inverse 2-D Fourier Transform. There are 4 parts to this assignment, each part examines a different operation done in the Fourier domain.

1. Perform 2D DFT and inverse 2D DFT of two images.
2. Examine what happens to an image when you set the phase of each frequency component to zero, but preserve the magnitude.
3. Examine what happens to an image when you eliminate the variation in the magnitude of each frequency component, but preserve the phase.
4. Examine what happens when an image is low pass filtered.

Exercise

In each of the 4 parts of this assignment, we will do the following three steps:

1. Compute the 2D DFT of an image.
2. Perform a frequency domain operation. This step is different for each part (see below).
3. Take the inverse 2D DFT and save it to disk.

Part 0: Calculating the 2D DFT of an Image

In this part, you will use the MATLAB function “fft2” that computes the 2D DFT of an 2D image and a function “ifft2” which computes the inverse 2D DFT. Detailed documentation can be found

To compute the 2D DFT of an image, first read the image into MATLAB using the following code:

```
image_1 = imread('roll.png');

image_2 = imread('univ.png');
```

In step 1, perform the 2D DFT of the two images:

```
image_1_fft = fft2(image_1);
```

image_2_fft = fft2(image_2);

In step 2, perform frequency domain operation. In this part, we skip this step.

In step 3. the inverse 2D DFT of the two images:

image_1_rec = ifft2(image_1_fft);

image_2_rec = ifft2(image_2);

image_1_fft and image_2_fft are complex images in the frequency domain. Please plot their magnitude images in both the normal and log scales:

imshow(abs(image_1_fft),[])

imshow(log(abs(image_1_fft)),[])

Where is the zero-frequency (i.e. omega = 0) located? If it is not located at the centers of image_1_fft and image_2_fft, can you do something to move it to the center? (check out the MATLAB functions fftshift and ifftshift)

Part 1: Vanishing Phase

In Step 2 in the procedure above, perform the opration in the frequency domain, remove the phase of each complex frequency sample, but preserve the magnitude. Note that you can do this in one line (per pixel) by using the MATLAB function “abs”: image_1_fft = abs(image_1_fft). Compare the result you obtained in step 3 to the original image. They should not look alike at all. Why?

After taking the inverse Fourier transform, the imaginary part coefficients are very small with respect to the real part and can be neglected (they are not written to the output file). Why?

Part 2: Constant Magnitude

This time, in Step 2, set the magnitude of each complex frequency in the Fourier domain to one, but preserve the phase. Note that you can do this too in one line (per pixel) by using the MATLAB function “abs”. Hint: each Fourier frequency sample can be written r*exp(j*theta), where r is the magnitude

and theta is the phase angle, change each sample so that it is just exp(j*theta).

Note the following important side-effect. When you set the magnitude in the Fourier domain to such a small value, the inverse FFT yields very small intensity values in the resulting output image. To deal with this effect so you can view the output image, scale the values of the real part of the inverse FFT to fill the range [0, 255] (again the values of the imaginary part are very small with respect to those of the real part and therefore can be neglected). The scaling is done as follows: Suppose the minimum value in the real part of the Inverse FFT is m, and the maximum value is M. Then the pixel values are modified as

modified_value = (initial_value - m)*255/(M-m)

The output image should contain some information, but much of it is lost. Why?

Part 3: Low Pass Filtering

Let “omega_max” denote the maximum frequency. This time, before step 2, apply a low pass filter to the image's spectrum - set all frequency samples outside the passband (i.e. |omega| > 0.5*omega_max) to zero, and leave all the samples inside the passband (i.e. |omega|<=0.5*omega_max) unchanged. You need to use the MATLAB function “fftshift” to move the zero frequency to the center (i.e. origin). After fftshift, the maximum frequency is then obtained at the boundary of the frequency image. Try different values for the cutoff frequency (for example, 0.1, 0.3, 0.8, etc.).

After you the high-frequency components are set to zero, use the MATLAB function “ifftshift” to move the zero-frequency away from the origin to undo the effect from the function “fftshift”. Then perform inverse FFT on the low-pass filtered frequency image. The output image should look similar to the original, but the extent of the difference depends on the cut-off frequency of the low pass filter.

Once you do the low-pass filtering operation in frequency domain, you will take the Inverse FFT. Your output image may contain a few corrupted pixels due to round-off errors during computation. Round-off errors at some pixels will fall outside the range [0, 255]. Perform the same operation to correct this as we used in part 2 (in the side-effect section).

Handin Instructions

- Do **not** include your source code in your report.
- Hand in the code files, the report and the images in one zip file.
- Points will be deducted for code that doesn't contain clear comments, or produces errors at run time.

Report Contents

Be sure to include the following points in the discussion of your write-up along with the introduction and conclusion parts

1. Where is the zero-frequency (i.e. omega = 0) located after applying the 2D DFT? Why is it not at the center of the frequency image?
2. Explain why the imaginary component of the image is negligible after performing the specific operations in the transform domain and taking the inverse 2D DFT.
3. Explain why the image does not look like the original after removing the phase in Part 1.
4. Why doesn't altering the magnitude (Part 2) change the appearance of the image as much as altering the phase (Part 1)?
5. In your subjective opinion, what is the smallest size (lowest cut-off frequency) of the low pass filter that can be applied to the test image without causing a significant degradation of image quality?
6. What does this tell you about the information content in the Fourier domain (low versus high frequencies)?