

University of Central Florida
CAP5415 - Computer Vision
Problem Set 1
Fall 2010

Assigned: Thursday, August 26, 2010

Due: Thursday, September 9, 2010

Introduction

Each of the sections below constitute one problem. If the problem asks for images, you should turn in a print-out with the requested images. Ideally, your assignment should be composed in a word-processor, such as L^AT_EX or Microsoft Word. You are welcome to write out derivations by hand.

In your writeup, describe the steps you completed for each problem and show the results. Readability will be part of your grade.

1 Warmup

For this problem, turn in the necessary code to complete the following steps. Also turn in the resulting images.

1. Download the images for PS 1 from the course web-page.
2. Choose two images
3. For each of these images
 - (a) Load the image into your environment
 - (b) Blur the image
 - (c) Display the result.
 - (d) Compute the DFT of the image
 - (e) Display the magnitude of the DFT

2 Fast Convolution

One day, you're complaining to your officemate that your current research project requires you to do convolution with very large kernels and that is just too slow. Your officemate smiles at you and cryptically says, "But, computing an FFT on an image is $O(N^2 \log N)$ for an $N \times N$ image."

1. If you have an $N \times N$ image and an $M \times M$ kernel, what is the complexity (in big-Oh notation) of convolving the kernel with the image.
2. How could you use the FFT to perform the same convolution?
3. What is the complexity of convolving the image using the method you just described?
4. Assuming that the big-Oh complexities were the actual complexities, for what sized kernel would the FFT-based method be faster? You may assume that the image is a 512×512 image.

3 Deconvolution

You've taken the most beautiful picture and are editing it. You decide to see what it might look like if it were blurred slightly. Unfortunately, your graphics package crashes just after the blurring and saves over the original. What will you do?

In class, we showed that convolving two signals is equivalent to multiplying their Fourier transforms. In this problem you will explore the limits of undoing a convolution.

For this problem, turn in the necessary code to complete the following steps. Also turn in the resulting images.

1. Make sure the image is stored as floating point values.
2. Blur the image using a convolution kernel. (You should use full or circular convolution to make the later parts work.)
3. Compute the Fourier Transform of the kernel. Make sure you pad the kernel so that it is the same size as the original image.
4. Using the DFT of the kernel and the DFT of the blurred image, how would you invert, or undo, the blurring operation?
5. Try this and see how well it works
6. Now, quantize the blurred image to 8 bits per pixel, (by converting it to unsigned characters, for example), then convert back to floating point values.
7. Now try inverting the convolution. How do the results compare? What do you think caused the difference?

4 Deriving the Convolution Theorem

Show that the DFT of $f[n] * h[n]$ is equal to $F[u]H[u]$, if F and H are the DFTs of f and h . You can assume that circular boundary handling is implied.

5 Median Filter

In the first lecture, we discussed the median filter. It works like a local averaging filter, except instead of taking the mean of a window, the median filter uses the median value in the window. (Look up median in Wikipedia if you do not remember the difference between a mean and a median). MATLAB and Python both provide median filtering operations, `medfilt2` and `medfilt2d` respectively.

Filter the two images for this problem set with median and averaging filters of 3 different sizes. How would you describe the difference in output between the mean and median filter? How does using the median cause these differences?