

MIT OpenCourseWare
<http://ocw.mit.edu>

9.35 Sensation And Perception
Spring 2009

For information about citing these materials or our Terms of Use, visit: <http://ocw.mit.edu/terms>.

9.35 Spring 2009
Problem Set 3 – 100 points.

1. Frequency decomposition of images (50 pts)

So far we have been simulating the ‘response of populations’ of cells with simple linear filters which could approximate different parts of the early visual stream. These filters preferentially select a piece of the spatial frequency structure in a local patch of an image, while throwing away the rest. We will now deal directly with this spatial frequency structure.

A fast Fourier transform (fft2 in Matlab) decomposes a 2D matrix into its component spatial frequencies. The output of fft2 is a matrix of complex numbers, wherein each number represents the amplitude and phase of a particular spatial frequency. In Matlab, this is a matrix in which the lowest frequencies are at the edges, and the highest frequencies are in the middle. In this representation, the first position corresponds to the DC offset (0 cycles/pixel – the mean of intensity) of the image, and locations towards the center of the matrix represent increasing frequencies up to the Nyquist limit of 1 cycle for every two pixels. Thus, pixels equidistant from the origin represent equal frequencies with different angular components in x and y (later we will rearrange the matrix to put low frequencies in the middle, but this circular symmetry will remain). The magnitude (absolute value, obtained with `abs()`) of the complex number is the amplitude of that component, while the angle (obtained with `angle()`) is the phase.

1.a) (10 pts) Use the “peppers” image (from pset 2) and a picture of yourself. Resize and/or crop both original images so they have *exactly* the same dimensions. Use `fft2` to decompose both images (peppers, and yourself) into their component frequencies. Use `fftshift` to reshape the Fourier transform output so that the lowest frequencies are in the middle (this is a more intuitive way to display the Fourier transform). Use `abs()` to display just the amplitude spectrum, and `angle()` to display just the phase. Now, plot a cross section through the center of the amplitude spectrum. Since low frequencies are in the middle and frequencies increase with distance from the center, what does the drop-off of amplitude with distance from the middle peak tell us about the image? How did we, the TA’s, know that an image of you would fall off like this?

1.b) (15 pts) For a complex number, C , the following identity holds:

$$C = \text{abs}(C) .* \exp(i .* \text{angle}(C))$$

You can use this fact to reconstruct a complex number representation, C , of amplitude and phase from the individual components.

For each of your two images, create a white noise image of the same size by using `randn` (white noise is defined as equal amplitude at all frequencies with random phase – this is equal to random Gaussian noise in pixel intensities). Decompose each white noise image into its amplitude and phase of its Fourier components.

Scramble each of your two images by creating a new FFT matrix (complex number representation) using amplitudes from the original image and the phases from the white noise image using the identity above. Reconstruct each image using `ifft2` (use `ifftshift` when appropriate!). If `ifft2` returns a complex matrix (and you've done everything right), just take the real component of the output (this is likely due to rounding errors, you can also use `ifft2(..., 'symmetric')` to avoid these rounding errors).

Now, create two 'whitened' images using the original phases, but white noise amplitudes, and display the reconstructions. Why do the whitened images appear to have higher frequencies?

1.c) (15 pts) Now, create two complimentary images by using the phase from one image and the amplitude from the other. Reconstruct and display the resulting images. Which attribute of the Fourier components, amplitude or phase, has more to do with our perception of the image? Why?

1.d) (10 pts) Take the Gaussian filter `G1` (from `pset2`) and pad the matrix with zeros (using `padarray`) so that it is exactly the same size as one of your images. Now take the FFT of the filter itself and display the amplitude spectrum (remember to use `fftshift`). In problem 2, we convolved the image with this filter, but we can obtain identical results by element-wise multiplying the FFTs of each and using the result to reconstruct the image (try it and see). Explain why this makes sense, and why the amplitude of the Gaussian filter FFT looks the way it does (intuitively, not mathematically). Now, display FFT amplitudes of `G2`, the DoG and two of the Gabor filters (from `pset2`), and why they look like the way they do.

2. “Hybrid” image (50 pts):

You can filter (or attenuate) specific frequencies by multiplying the FFT amplitude by a radial mask in which some frequencies have weight 0. This is what we did in 1.d. This time, instead of creating a radial mask from the FFT of one of the previously used filters, create a mask manually by using `meshgrid`, and `cart2pol`.

2.a) (5 pts) Load the `adelson.tif` and `wolfe.tif` images, and compute their FFTs, use `fftshift` to move the low spatial frequencies into the middle (since the images have odd dimensions, the center pixel will now be the mean intensity), and separate the amplitude and phase of the components. Use `imagesc` to display the amplitudes of the two images side by side.

2.b) (10 pts) Define a 249×249 radial mask that filters out all high frequencies more than 16 units away from the matrix center (low-pass filter, since it leaves only allows components with frequencies greater than 16 cycles per image), and another mask that filters out all low frequencies less than 16 units away from the matrix center (high-pass filter, since it leaves only high-frequencies.). Display both masks.

2.c) (15 pts) ‘Filter’ the amplitude spectra of Adelson and Wolfe by multiplying the first by the high-pass mask, and the second by the low-pass mask. Reconstruct the images using `ifft2`, and show both pictures independently. Normalize the images (so they both span the same range of intensities), and combine the two images by adding their pixel intensities together. Display this image.

2.d) (20 pts) What does this look like? As you move away from the image, at what distance do you switch from seeing Jeremy Wolfe to seeing Prof. Adelson? Why does this switch with distance?

