UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN
Department of Electrical and Computer Engineering

ECE 498MH Signal and Image Analysis

Homework 3

Fall 2013

Assigned: Friday, September 20, 2013 Due: Friday, September 27, 2013

Reading: Signal Processing First (SPF) Chapter 5

Problem 3.1

Determine whether or not each of the following systems is linear. If nonlinear, give an example of signals $x_1[n] \rightarrow y_1[n]$, $x_2[n] \rightarrow y_2[n]$, and $x_3[n] \rightarrow y_3[n]$ such that $x_3[n] = x_1[n] + x_2[n]$ but $y_3[n] \neq y_1[n] + y_2[n]$.

(a) $y[n] = x[n] - x[0]$

(b) $y[n] = x[n] - 1$

Problem 3.2

Determine whether or not each of the following systems is time-invariant. If time-varying, give an example of signals $x_1[n] \rightarrow y_1[n]$ and $x_2[n] \rightarrow y_2[n]$ such that $x_2[n] = x_1[n-d]$ but $y_2[n] \neq y_1[n-d]$.

(a) $y[n] = x[n] - x[0]$

(b) $y[n] = x[n] - 1$

Problem 3.3

Find the convolution $y[n] = \text{conv}(x[n], h[n])$, for

$$x[n] = \begin{cases} 1 & 0 \leq n \leq 2 \\ 0 & \text{otherwise} \end{cases}$$

and

$$h[n] = \begin{cases} 1 & n = 0 \\ 0.5 & n = \pm 1 \\ 0 & \text{otherwise} \end{cases}$$

Matlab Exercises

Problem 3.4

Download a University of Illinois logo, e.g., from the course web page. Read it into matlab using, for example `A=imread('imark.png');` Use `size(A)` to find out how big it is. Show the image using `image(A);` Extract the red colorplane, and convert it from integer to double-precision arithmetic, using `x=double(A(:,:,1));`, then show it. You should find that the red colorplane is white (high-valued) whenever the color image is orange or white, and black (low-valued) whenever the color image is blue or black.

Try `plot(x(450,:));` to plot one row of the image, and `plot(x(:,380));` to plot one column. Notice that the color is encoded with numbers ranging from 0 (black) to 255 (white).

(a) Create an edge detection filter as `hr=[1 0 -1];`. Apply it along the rows of the image using `xr=conv2(x,hr,'same');`. The "same" command tells matlab that the output of the convolution should be the same size as the input.

Try `plot(x(450,:));` to plot one row of the image. Notice that the filter output is +255 at the beginning of each white region, and -255 at the end of each white region.

Show the edge detection image using `image(xr);` and using `imagesc(xr);`. The `image` command assumes that its input is between 0 and 255; numbers outside this range are discarded. The `imagesc` rescales the input before plotting, so that zeros (no edge) are depicted as gray, -255 (downstep edge) becomes black, and +255 (upstep edge) becomes white. Add a title to your `imagesc` image, print it as a PNG, and include it in your lab report.

(b) Create a horizontal edge detector using `hc=hr'`. Apply it to the image using `xc=conv2(x,hc,'same');`. Plot a column using `plot(x(:,380));`, and notice the upstep and downstep edges. Show the scaled image using `imagesc(xc);`. Add a title, print it, and hand in with your lab report.

(c) Sobel proposed a nonlinear edge detector that is now called a "Sobel mask," implemented as

$$s[n_1, n_2] = (x_r[n_1, n_2])^2 + (x_c[n_1, n_2])^2$$

where $x_r[n_1, n_2]$ and $x_c[n_1, n_2]$ are the row and column edge detector outputs, respectively. Create a Sobel mask. Plot it using `imagesc`. Add a title, print, and hand it in.

(d) Create a noisy image using `xnoisy=x+50*randn(size(x));` Use `subplot(2,2,1);imagesc(xnoisy);` to show it in the upper left quadrant of a four-part image.

Create a row-wise averaging filter as `gr=0.1*ones(1,10)`, apply it using `yr=conv2(xnoisy,gr,'same');`, and put it in the upper right quadrant using `subplot(2,2,2);imagesc(yr);`. Notice that the noise is reduced, but it is also kind of stretched out in the horizontal direction; also, the vertical edges are kind of blurred (you may need to fullscreen your figure to see this).

Create a column-wise averaging filter as `gc=gr'`, apply it using `yc=conv2(xnoisy,gc,'same');`, and put it in the lower left quadrant using `subplot(2,2,2);imagesc(yc);`. Notice that the vertical edges are much sharper in this image!! The horizontal images have been blurred, though.

The best noise reduction results (at the cost of blurred edges in both the vertical and horizontal directions!) is achieved by filtering first using `gr`, then re-filtering the output using `gc`, thus `y=conv2(yr,gc,'same');`. Create this image, and show it in the remaining quadrant

(`subplot(2,2,4);imagesc(y);`).

Print out the four-quadrant display, and include it in your lab report.

It doesn't matter whether gc is applied before or after gr. Try it: compute `y2=conv2(yc,gr,'same');`, then find the difference between these two images using `sum(sum(abs(y-y2)))`. You should find that the difference is a number very close to zero.