

## ECE 438 Homework 13, due in class Friday, 12/10/2004.

**Problem 1.** In Matlab, implement the iterative non-linear filter (the Perona-Malik filter) considered in class:

$$\begin{aligned} g^{(i+1)}(m, n) &= g^{(i)}(m, n) + \\ &+ \alpha \left\{ F[g^{(i)}(m+1, n) - g^{(i)}(m, n)] + F[g^{(i)}(m-1, n) - g^{(i)}(m, n)] + \right. \\ &\quad \left. + F[g^{(i)}(m, n+1) - g^{(i)}(m, n)] + F[g^{(i)}(m, n-1) - g^{(i)}(m, n)] \right\}, \quad (1) \end{aligned}$$

for  $1 \leq m \leq M, 1 \leq n \leq N, 0 \leq i \leq I-1$ ,

where  $M$  and  $N$  are the dimensions of the image,  $I$  is the total number of iterations, and  $F(v)$  is the following nonlinear function which depends on the positive real parameter  $K$ :

$$F(v) = \frac{v}{1 + \left(\frac{v}{K}\right)^2}, \text{ for any real number } v.$$

Use the following syntax:

```
function g = PM(f,alpha,I,K)
```

Here,  $f = g^{(0)}$  is the input image to be filtered; alpha is the filter parameter  $\alpha$ ; I is the total number of iterations; K is the filter parameter  $K$ ; and  $g = g^{(I)}$  is the output image, which is the result of  $I$  iterations of the filter.

To calculate  $g^{(i+1)}(m, n)$  around the border of the image, assume that  $g^{(i)}(m, n)$  is symmetrically reflected. In other words, assume that

$$\begin{aligned} g^{(i)}(0, n) &= g^{(i)}(1, n) \\ g^{(i)}(m, 0) &= g^{(i)}(m, 1) \\ g^{(i)}(M+1, n) &= g^{(i)}(M, n) \\ g^{(i)}(m, N+1) &= g^{(i)}(m, N) \end{aligned} \quad (2)$$

**Hints.** Avoid “for” loops. Your program should have only one “for” loop: to iterate Eq. (1) over  $i = 0, 1, \dots, I-1$ . All other computations should be done on entire images. For example, to calculate  $F[g^{(i)}(m+1, n) - g^{(i)}(m, n)]$ , you can use

```
delta_g = [g(2:M,:); g(M,:)] - g;
F_of_delta_g = delta_g./(1+(delta_g/K).^2);
```

(a) Test your program on a noisy binary image, created with the following script:

```
rand('state',sum(100*clock));
test_im = [ones(50,25)*50 ones(50,25)*200]+floor(rand(50,50)*50);
```

Use  $\alpha = 0.25$ ,  $K = 2$ , and 200 iterations:

```
g = PM(test_im, 0.25, 200, 2);
```

Display both the noisy image `test_im` and the filtered image `g`. Recall from Lab 10 that the following sequence of commands should be used to display a grayscale image:

```
image(test_im);  
colormap(gray(256));  
axis('image');
```

- (b) Use your program to filter the noisy race-car image `noise1.tif` from Lab 10. Use the following four different parameter settings.

- (i) Use  $\alpha = 0.25$ ,  $K = 2$ , and 200 iterations:

```
g = PM(double(noise1), 0.25, 200, 2);
```

(This could take 1-2 minutes.)

- (ii)  $\alpha = 0.25$ ,  $K = 2$ , 500 iterations.  
(iii)  $\alpha = 0.25$ ,  $K = 20$ , 20 iterations.  
(iv)  $\alpha = 0.25$ ,  $K = 20$ , 50 iterations.

Display the original noise-free image `race`, the noisy image `noise1` and the four filtered images. Use `subplot` to put the images on the same page. Comment on the quality of the reconstruction in each of the four cases, and explain the differences among the four results.

- (c) Another nonlinear method of noise removal discussed in class is to decompose an image in a basis, set to zero all coefficients that are smaller than some threshold, and then reconstruct an estimate from the remaining coefficients. This procedure, for a wavelet basis, is performed by the Matlab command `wdecmp` which is a part of the Wavelet Toolbox. The syntax is:

```
out = wdecmp('gbl',noise1,wavelet_name,levels,thresh,'h',1);
```

Here, `noise1` is the image to be enhanced; `wavelet_name` is a character string containing the name of the wavelet; `levels` is the depth of the decomposition pyramid; `thresh` is the value of the threshold; and `out` is the output image. Three other inputs stand for other filtering parameters which you need not worry about for this exercise.

Use wavelet thresholding to de-noise the noisy car image. Experiment with different wavelet families, filter orders, pyramid depths, and thresholds. To see the Matlab names for the different wavelets, try `help biorwavf` (for biorthogonal spline wavelets), `help dbwavf` (for Daubechies wavelets), `help symwavf` (for symmlets).

Turn in at least three different output images (for three different parameter settings), and comment on your results (what happens when you vary the filter order, the number of iterations, and the threshold?). For this example, reasonable values for the threshold are in the range 50-100. E.g., to use a Daubechies wavelet of order 2, have 7 filter iterations, and set the threshold at 50, use the following command:

```
out = wdencomp('gbl',noise1,'db2',7,50,'h',1);
```

(To find out more about wavelets, how to choose a wavelet, how to set the threshold, etc, take ECE 648!)

**Problem 2.** The tomographic projection  $g_\theta(t)$  of a 2-D object  $f(x_1, x_2)$  at the angle  $\theta$  and location  $t$  is obtained by

- rotating the  $(x_1, x_2)$  coordinate axes counterclockwise by angle  $\theta$ , and
- computing the line integral of  $f(x_1, x_2)$  along the line which passes through the new horizontal axis at the point  $t$  and is perpendicular to it.

(a) Let

$$f(x_1, x_2) = \begin{cases} 1 & \text{if } x_1^2 + x_2^2 \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Find the tomographic projection  $g_0(t)$  of  $f(x_1, x_2)$  at the angle  $\theta = 0$ . Either a formula or a fully labeled plot of  $g_0(t)$  as a function of  $t$  is acceptable as the answer. (**Hint.** Since  $f(x_1, x_2)$  is a very simple function, the answer can be obtained without explicitly writing down any integrals, by drawing the set of points where  $f(x_1, x_2) \neq 0$  and relating the line integrals that you need to compute to the lengths of certain segments in your drawing.)

(b) Let

$$f'(x_1, x_2) = \begin{cases} 2 & \text{if } 0 \leq x_1 \leq 1 \text{ and } 0 \leq x_2 \leq 1, \\ 0 & \text{otherwise.} \end{cases}$$

Find the tomographic projection  $g'_{\pi/4}(t)$  of  $f'(x_1, x_2)$  at the angle  $\theta = \pi/4$ .