ECE203A: HOMEWORK #4

Due: November 9, 2004

Problems from the textbook: 4.5, 4.7, 4.9, 4.14

Computer Problems:

spatial

a) Apply the 3×3 Laplacian filter with the -8 in the center to the triangle image and the cat image. You can ignore the border regions where the convolution sum cannot be computed so that the output image will be smaller than the input image. Add a constant to the image so that the smallest negative value becomes zero and then scale into the range 0-255. Also apply the sharpening filter defined as the input image minus the Laplacian to the cat and triangle images. Leave the four displayable filtered images in your directory.

b) Generate an image of $|F(u, v)|$ with the DC in the center for the cat and triangle images.

freq.

ECE263A: HOMEWORK #4 Solution
Fall 2004

4.5

$$\begin{aligned} h_{hp}(x,y) &= \mathcal{F}^{-1}\left[1 - H_{lp}(u,v)\right] \\ &= \mathcal{F}^{-1}[1] - \mathcal{F}^{-1}\left[Ae^{-(u^2+v^2)/2\sigma^2}\right] \\ &= \delta(0,0) - A2\pi\sigma^2 e^{-2\pi^2\sigma^2(x^2+y^2)} \end{aligned}$$

4.7 The equally spaced, vertical bars in the lower left part of the image.

4.9 The multiplications by $(-1)^{x+y}$ in steps a) and e) are just for centering and cancel each other out. Let

$$\begin{aligned} F(u,v) &= \mathrm{Re}(F(u,v)) + j\mathrm{Im}(F(u,v)) \\ F^*(u,v) &= \mathrm{Re}(F(u,v)) - j\mathrm{Im}(F(u,v)) \\ \mathcal{F}^{-1}\left[F^*(u,v)\right] &= \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\left[\mathrm{Re}(F(u,v)) - j\mathrm{Im}(F(u,v))\right]e^{jX} \end{aligned}$$

where $X = 2\pi(ux/M + vy/N)$

$$\mathcal{F}^{-1}\left[F^*(u,v)\right] = \sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\left[\mathrm{Re}(F(u,v)) - j\mathrm{Im}(F(u,v))\right]\left[\cos(X) + j\sin(X)\right]$$

The real part of $\mathcal{F}^{-1}\left[F^*(u,v)\right]$ is

$$\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}\mathrm{Re}(F(u,v))\cos(X) + \mathrm{Im}(F(u,v))\sin(X)$$

which is equal to the real part of

$$\sum_{u=0}^{M-1}\sum_{v=0}^{N-1}F(u,v)e^{j(-X)} = f(-x,-y)$$

The image $f(-x,-y)$ is the image on the right.

4.14 a) The spatial average is

$$g(x,y) = 0.25\left[f(x,y+1) + f(x+1,y) + f(x-1,y) + f(x,y-1)\right]$$

From Eq. (4.6-2) in the book,

$$\begin{aligned} G(u,v) &= 0.25\left[e^{j2\pi v/N} + e^{j2\pi u/M} + e^{-j2\pi u/M} + e^{-j2\pi v/N}\right] F(u,v) \\ &= H(u,v)F(u,v) \end{aligned}$$

where

$$H(u,v) = 0.5\left[\cos(2\pi u/M) + \cos(2\pi v/N)\right]$$

is the filter transfer function in the frequency domain.

b) The filter has its maximum value at $(u,v) = (0,0)$ and falls off as we move away from $(u,v) = (0,0)$. This is the characteristic of a lowpass filter.

ECE~~234~~203A: HOMEWORK #3 Solution Fall 2004

3.12 A histogram contains no information about the spatial arrangement of pixels in an image meaning that different images can have the same histogram. In general, we cannot use the histograms h_f and h_g to determine the histograms in problems a,b,c,d. For the special case where one of the input images has a constant gray level we can determine the histograms for problems a,b,c,d. Let us suppose that image $f(x,y)$ has the constant gray level C. The solution for $g(x,y)$ having a constant gray level is similar.

a) If $f(x,y)$ has the constant gray level C, then the histogram of $f(x,y)+g(x,y)$ is given by

$$h_{sum}(r_k) = h_g(r_k - C)$$

b) If $f(x,y)$ has the constant gray level C, then the histogram of $f(x,y)-g(x,y)$ is given by

$$h_{diff}(r_k) = h_g(C - r_k)$$

c) If $f(x,y)$ has the constant gray level C, then the histogram of $f(x,y)\times g(x,y)$ is given by

$$h_{mult}(r_k) = h_g(r_k/C)$$

d) If $f(x,y)$ has the constant gray level C, then the histogram of $f(x,y)/g(x,y)$ is given by

$$h_{div}(r_k) = h_g(C/r_k)$$

3.17 a) Consider a 3×3 mask first. Since the coefficients are 1 (we are ignoring the 1/9 scale factor), the net effect of the lowpass filter operation is to add the gray levels of the pixels under the mask. Initially, it takes 8 additions to produce the response of the mask. However, when the mask moves one pixel location to the right, it picks up only one new column. The new response can be computed as

$$R_{new} = R_{old} - C_1 + C_3$$

gl where C_1 is the sum of pixels under the first column of the mask before it was moved and C_3 is the similar sum in the column it picked up after it moved. For a 3×3 mask it takes $\cancel{2}$ additions to get C_3 (C_1 was already computed). To this we add one subtraction and one addition to get R_{new}. Thus, a total of 4 arithmetic operations are needed to update the response after one move. This procedure moves from left to right along a row of the image. When we get to the end of a row, we move down one pixel and continue the scan for the next row.

For a mask of size $n \times n$, $(n-1)$ additions are needed to obtain C_3, plus the single subtraction and addition needed to obtain R_{new}, which gives a total of $(n+1)$ arithmetic operations after each move. Note that we are ignoring the $n^2 - 1$ additions that are performed once at the start of each row. This is reasonable if n is much smaller than the number of columns in the image. A brute-force implementation would require $n^2 - 1$ additions after each move.

b) The computational advantage is

$$A = \frac{n^2 - 1}{n + 1} = n - 1$$

3.24 Yes. The Laplacian and the averaging are linear operators so it does not matter which one is applied first.