CS 422/522: Digital Image Processing
Homework 4 (Fall '13)

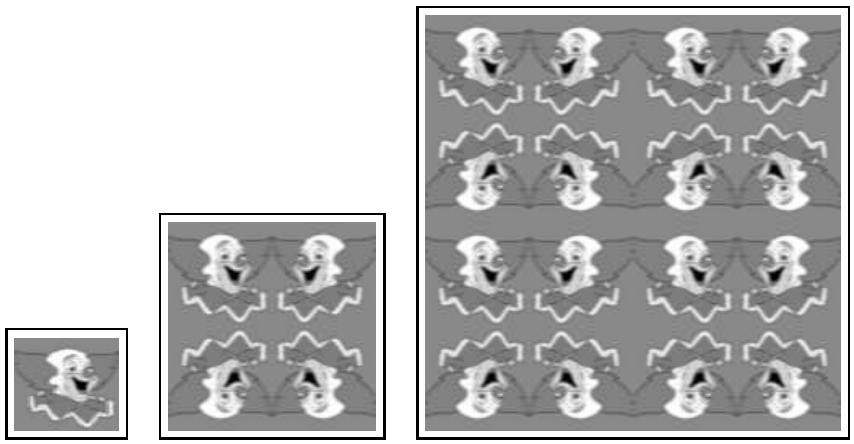
1 Image Watermarking

1. Using the non-watermarked image *rove* as reference, recover the watermark from the image *watermarked-rove*. Both images are stored in ASCII *.pgm* format with 12 bit integer grey values and can be downloaded from the class webpage. Now save both images in ASCII *.pgm* format with 8 bit integer grey values (the default) and repeat the operations you previously used to recover the watermark.

2. Write a function *make-watermark* which takes a square image, *image*, an integer, *n*, and a float *epsilon*, as arguments. You may assume that the image is of size 2^m (for integer *m*) and has grey values in the range $[0, 255]$. The function returns a floating point image representing a watermark of size 2^{m+n}, with grey values in the range $[1-\varepsilon, 1+\varepsilon]$, constructed recursively according to the scheme shown in Figure 1. Construct a watermark using an image of your choice using a value of $n \geq 2$.

3. Write a function, *insert-watermark*, which takes two square images of equal size (2^m for integer *m*) representing an input image *image* and a watermark *mark* as arguments, and returns *image* watermarked with *mark*. Use your function to watermark an image of your choice with a watermark of your choice. Save your image in ASCII *.pgm* format with 8 bit integer grey values.

4. Write a function, *recover-watermark*, which takes two square images of equal size (2^m for integer *m*), representing a watermarked image, *marked − image*, and a non-watermarked image, *image*, and an integer *n*. Your function should average over the spatial symmetries (recursively to depth *n*) in the watermark to improve the signal-to-noise ratio. Recover the watermark from the watermarked image you created and saved in the last problem.



(a) (b) (c)

Figure 1: (a) Example watermark for $n = 0$. (b) For $n = 1$.(c) For $n = 2$.

2 Frequency Domain Filtering

1. Write a function *ideal-lowpass* which takes an integer n and a float $width$ as arguments and returns the frequency domain representation of an ideal lowpass filter of size $n \times n$. Your filter should be one for spatial frequencies less than $width$ and zero otherwise. Apply your filter to a square image of your choice (of size $n = 2^k$ for integer $k \geq 7$) and $width = 16$. See Figure 2 (a).

2. Write a function *gaussian-lowpass* which takes an integer n and a float *variance* as arguments and returns the frequency domain representation of a Gaussian lowpass filter of size $n \times n$. Your filter should be a Gaussian of variance *variance* centered on the zero spatial frequency. Apply your filter to a square image of your choice (of size $n = 2^k$ for integer $k \geq 7$) and $variance = 32$. See Figure 2 (b).

3. Write a function *ideal-bandpass* which takes an integer n and floats *center* and $width$ as arguments and returns the frequency domain representation of an ideal bandpass filter of size $n \times n$. Your filter should be one inside a band of width $width$ centered on spatial frequency *center* and zero otherwise. Apply your filter to a square image of your choice (of size $n = 2^k$ for integer $k \geq 7$), $center = 32$ and $width = 16$. See Figure 2 (c).

4. Write a function *gaussian-bandpass* which takes an integer n and floats

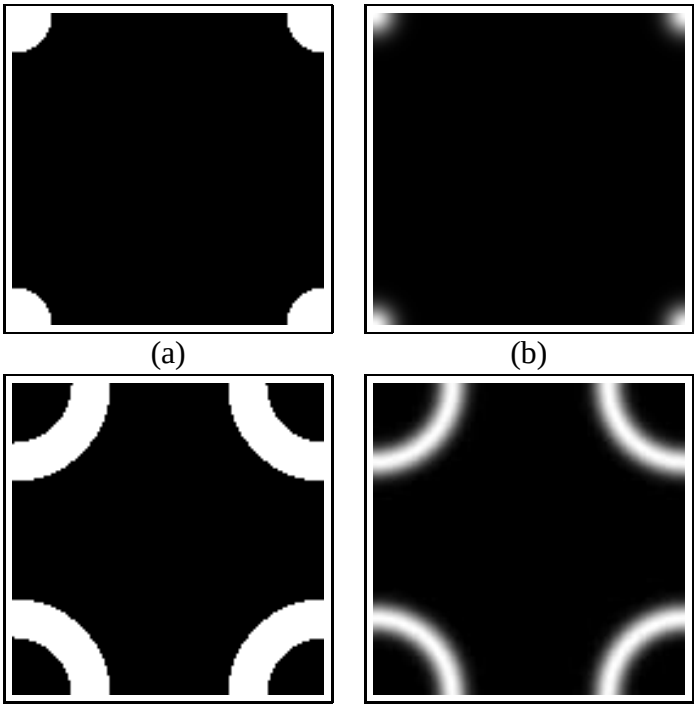


Figure 2: (a) Ideal lowpass filter. (b) Gaussian lowpass filter. (c) Ideal bandpass filter. (d) Gaussian bandpass filter.

center and *variance* as arguments and returns the frequency domain representation of a Gaussian bandpass filter of size $n \times n$. Your filter should be an annulus with Gaussian cross-section with variance *variance* and mean *center*. Apply your filter to a square image of your choice (of size $n = 2^k$ for integer $k \geq 7$) and *variance* $= 32$. See Figure 2 (d).

3 Optimal Linear Filtering

There are three directories on the class webpage named *signal*, *noise*, and *test*. These directories contain images of the surface of a silicon wafer imaged with a scanning tunneling microscope by Prof. Sang Han of the UNM Chemical and Nuclear Engineering Department. In the images in the *noise* and *test* directories, the atoms which comprise the crystal lattice of the silicon wafer are clearly visi-

ble. Unfortunately, this prominent background lattice complicates the process of automatically classifying the number and size of the *island* features which are of specific interest to Prof. Han. The images in the *signal* directory were imaged in a way which supresses the background lattice but leaves the *island* features unchanged.

1. Compute the average power of all of the images in the *signal* directory. Call this image $S(u,v)$. Display $\log S(u,v)$.

2. Compute the average power of all of the images in the *noise* directory. Call this image $N(u,v)$. Display $\log N(u,v)$.

3. Compute and display the frequency domain representation of the Wiener filter
$$W(u,v) = \frac{S(u,v)}{S(u,v)+N(u,v)}.$$

4. Apply the Wiener filter to each of the images in the *test* directory. Display the images before and after Wiener filtering.