CSE166 – Image Processing – Homework #3
Instructor: Prof. Serge Belongie
http://www-cse.ucsd.edu/classes/fa10/cse166
Due (in class) 2:00pm Friday Oct 22, 2010.

Reading

• GW Second Edition, 4.3-4.4, 4.6.
  or
GW Second Edition, 4.8-4.9, 4.11.

General Homework Guidelines

• Use the Cover Sheet provided.
• Please attach all code that you use. Attach code at end of submission.
• In general try to keep your answers concise. Use as many words as you need and no more. Also
  work on your presentation skills. This means organize your plots and displays. Always use
  titles and add captions when appropriate. Points will be awarded for clarity and presentation.

Written exercises

  or
GW Third Edition, Problem 4.14

  or
GW Third Edition, Problem 4.34

  or
GW Third Edition, Problem 4.35

  or
GW Third Edition: Show that if a filter transfer function $H(u, v)$ is real and symmetric, then
the corresponding spatial domain filter $h(x, y)$ also is real and symmetric.

  or
GW Third Edition, Problem 4.28

  or
GW Third Edition, Problem 4.30
Matlab exercises

1. Filtered Noise

(a) Consider the filter with Fourier transform

\[ H(u, v) = \frac{1}{u^2 + v^2} \]

on the interval \([-127, 128] \times [-127, 128]\). This is known as a \(1/f^2\) transfer function. Since it blows up at the origin, replace that value with zero. Apply this filter to a \(256 \times 256\) image of normally distributed random noise (use \texttt{randn}). For practical reasons, it is best to perform this operation in the frequency domain. Hint: you will need to use \texttt{meshgrid}, \texttt{fft2}, \texttt{ifft2}, and \texttt{fftshift}. Also, due to numerical error, you will need to use \texttt{real} to look at the real part of the filtered image in the spatial domain.

(b) Display the filtered image along with the original noise image. Quite remarkably, the filtered image should look like a “natural” texture, such as clouds or terrain. What does this suggest about the statistics of natural images vs. that of images of manmade objects?

Things to turn in:

- Code listing for part 1a.
- Printout and written answer for part 1b.

2. Filtering in the Frequency Domain

GW Second Edition, before doing this exercise, review GW Sections 4.6.3 and 4.6.4 to learn about frequency domain filtering, zero-padding, and the relationship between correlation and convolution.

or

GW Third Edition, Read the scanned pages at the end of this homework.

(a) Write an m-file to reproduce Figure 4.41(a-f) (of GW Second Edition) using Frequency domain filtering. Note: the Matlab command for the complex conjugate is \texttt{conj}.

(b) Repeat the previous step using operations in the spatial domain and show that the results are the same. Hint: use \texttt{conv2} and \texttt{rot90}.

Things to turn in:

- Code listing for parts 2a and 2b.
- Printouts of program output for parts 2a and 2b.
with the forward transform. As indicated in Sections 4.2.1 and 4.2.2, it is not unusual to encounter the constants distributed in a different way between the forward and inverse transforms. Therefore, in order to avoid being off by a scale factor, care must be exercised in the placement of the constants when computing the inverse transform if these constants are distributed differently from the way that we do in this book.

4.3 More on Periodicity: the Need for Padding

It was explained in Section 4.2.4 that, based on the convolution theorem, multiplication in the frequency domain is equivalent to convolution in the spatial domain, and vice versa. When working with discrete variables and the Fourier transform, we need to keep in mind the periodicity of the various functions involved (Section 4.6.1). Although it may not be intuitive, this periodicity is a mathematical byproduct of the way in which the discrete Fourier transform pair is defined. Periodicity is part of the process, and it cannot be ignored.

Figure 4.36 illustrates the significance of periodicity. The left column of this figure shows convolution computed using the 1-D version of Eq. (4.2-30): 

$$ f(x) \ast h(x) = \frac{1}{M} \sum_{m=0}^{M-1} f(m)h(x-m). \quad (4.6-20) $$

We also take the opportunity here to explain convolution in a little more detail. To simplify the notation, simple numbers instead of general symbols are used for the height and length of the functions. Figures 4.36(a) and (b) show the two functions we wish to convolve. Each function consists of 400 points. The first step in convolution is to mirror (flip) one of the functions about the origin. In this case, this was done to the second function, which is shown as $h(-m)$ in Fig. 4.36(c). The next step is to "slide" $h(-m)$ past $f(m)$. This is done by adding a constant, $x$, to $h(-m)$; that is, we form $h(x-m)$, as shown in Fig. 4.36(d). Note that this is only one displacement value. This simple step is a frequent source of confusion when first encountered. It helps to remember that this is precisely what convolution is all about. In other words, to perform convolution we flip one of the functions and slide it past the other. At each displacement (each value of $x$) the entire summation in Eq. (4.6-20) is carried out. This summation is nothing more than the sum of products of $f$ and $h$ at a given displacement. The displacement $x$ ranges over all values required to completely slide $h$ past $f$. Figure 4.36(e) shows the result of completely sliding $h$ past $f$ and computing Eq. (4.6-20) at each value of $x$. In this case, $x$ had to range from 0 to 799 for $h(x-m)$ to slide completely past $f$. This figure is the convolution of the two functions. Keep clearly in mind that the variable in convolution is $x$.

We know from the convolution theorem introduced in Section 4.2 [see Eq. (4.2-31)] that we can obtain exactly the same result given in Eq. (4.6-20) by taking the inverse Fourier transform of the product $F(u)H(u)$. However, we also know from the discussion on periodicity earlier in this section that the discrete Fourier transform automatically takes the input functions as periodic. In other words, using the DFT allows us to perform convolution in the frequency domain, but the functions are treated as periodic, with a period equal to the length of the functions.
FIGURE 4.36 Left: convolution of two discrete functions. Right: convolution of the same functions, taking into account the implied periodicity of the DFT. Note in (j) how data from adjacent periods corrupt the result of convolution.

We can examine the implications of this periodicity with the aid of the right column of Fig. 4.36. Figure 4.36(f) is the same as Fig. 4.36(a), but with periods of the same function extending infinitely in both directions (extended sections are shown dashed). The same applies to Figs. 4.36(g) through (i). We now perform convolution by sliding $h(x - m)$ past $f(m)$. The sliding is accomplished by varying $x$, as before. However, now the periodic extensions of $h(x - m)$ introduce values that were not there in our computations on the left side of Fig. 4.36. For example, when $x = 0$ in Fig. 4.36(i), we see that part of the first extended period to the right of $h(x - m)$ lies inside the part of $f(m)$ in Fig. 4.36(a) that starts at the origin (shown solid). As $h(x - m)$ slides to the right, the section that was inside $f(m)$ starts to move out to the right, but it is replaced by an identical section from the left side of $h(x - m)$. This causes the convolution to have a constant value, as shown in the segment $[0, 100]$ in Fig. 4.36(j). The section...
ment from 100 through 400 is correct, but periodicity starts again, thus causing part of the tail of the convolution function to be lost, as can be seen by comparing the solid lines in Figs. 4.36(j) and 4.36(e).

In the frequency domain the procedure would be to compute the Fourier transforms of the functions in Figs. 4.36(a) and (b). According to the convolution theorem, the two transforms would then be multiplied and the inverse Fourier transform taken. The result would be the 400 points comprising the convolution shown in solid in Fig. 4.36(j). This simple illustration shows that failure to handle the periodicity issue properly will give incorrect results if the convolution function is obtained using the Fourier transform. The result will have erroneous data at the beginning and have missing data at the end.

The solution to this problem is straightforward. Assume that \( f \) and \( h \) consist of \( A \) and \( B \) points, respectively. We append zeros to both functions so that their individual periods, denoted by \( P \). This procedure yields extended, or padded, functions given by

\[
f_{e}(x) = \begin{cases} f(x) & 0 \leq x \leq A - 1 \\ 0 & A \leq x \leq P \end{cases}
\]  

(4.6-21)

and

\[
g_{e}(x) = \begin{cases} g(x) & 0 \leq x \leq B - 1 \\ 0 & B \leq x \leq P \end{cases}
\]  

(4.6-22)

It can be shown (Brigham [1988]) that, unless we choose \( P \geq A + B - 1 \), the individual periods of the convolution will overlap. We already saw in Fig. 4.36 the result of this phenomenon, which is commonly referred to as wraparound error. If \( P = A + B - 1 \), the periods will be adjacent. If \( P > A + B - 1 \), the periods will be separated, with the degree of separation being equal to the difference between \( P \) and \( A + B - 1 \).

The results obtained after extending the functions in Figs. 4.36(a) and (b) are shown in Figs. 4.37(a) and (b). In this case we choose \( P = A + B - 1 \) (799), so we know that the periods of the convolution will be adjacent. Following a procedure identical to the previous explanation, we arrive at the convolution function shown in Fig. 4.37(e). One period of this result is identical to Fig. 4.36(e), which we know to be correct. Thus, if we wanted to compute the convolution in the frequency domain, we would (1) obtain the Fourier transform of the two extended sequences (each of which is 800 points long); (2) multiply the transforms; and (3) compute the inverse Fourier transform. The result would be the correct 800-point convolution function shown in the period highlighted in Fig. 4.37(e).

Extension of these concepts to 2-D functions follows the same line of reasoning. Suppose that we have two images \( f(x, y) \) and \( h(x, y) \) of sizes \( A \times B \) and \( C \times D \), respectively. As in the 1-D case, these arrays must be assumed periodic with some period \( P \) in the \( x \)-direction and \( Q \) in the \( y \)-direction. Wraparound error in 2-D convolution is avoided by choosing

\[
P \geq A + C - 1
\]  

(4.6-23)

and

\[
Q \geq B + D - 1.
\]  

(4.6-24)
FIGURE 4.37
Result of performing convolution with extended functions. Compare Figs. 4.37(e) and 4.36(e).
The periodic sequences are formed by extending \( f(x, y) \) and \( h(x, y) \) as follows:

\[
f(x, y) = \begin{cases} f(x, y) & 0 \leq x \leq A - 1 \quad \text{and} \quad 0 \leq y \leq B - 1 \\ 0 & A \leq x \leq P \quad \text{or} \quad B \leq y \leq Q \end{cases} \quad (4.6-25)
\]

and

\[
h(x, y) = \begin{cases} h(x, y) & 0 \leq x \leq C - 1 \quad \text{and} \quad 0 \leq y \leq D - 1 \\ 0 & C \leq x \leq P \quad \text{or} \quad D \leq y \leq Q \end{cases} \quad (4.6-26)
\]

The issue of padding is central to filtering. When we implement any of the frequency domain filters discussed in this chapter, we do it by multiplying the filter transfer function by the transform of the image we wish to process. By the convolution theorem, we know that this is the same as convolving the spatial representation of the filter with the image. Thus, unless proper padding is implemented, the results will be erroneous. This is illustrated in Fig. 4.38. For the sake of simplicity in the figure, we assume that \( f \) and \( h \) are square and that they are

![Diagram](image-url)
both of the same size, where \( h \) is the inverse DFT of the filter \( H(u, v) \). Figure 4.38(a) shows what the result of filtering would be if the images were not padded. This is the generic result we would obtain if we computed the Fourier transform of an input image that was not padded, multiplied it by a filter function of the same size (also not padded) and computed the inverse transform. The result would be of size \( A \times B \), the same as the input image, as shown in the top left quadrant in Fig. 4.38(a). As in the 1-D case, the leading edges of the image would contain erroneous data induced by periodicity (shown shaded), and there would be missing data at the trailing edges. By properly padding the input image and the filter function as shown in Fig. 4.38(b), the result would be a correctly filtered image of size \( P \times Q \), as shown in Fig. 4.38(c). This image is twice the size of the original in both directions, and thus has four times as many pixels. However, as will be shown shortly, the area of interest typically is cropped out of this larger image.

It is important to note that the approach just described calls for the frequency domain filter function to be inverse transformed, padded with 0's, and then forward transformed. All other aspects of filtering are as described in Section 4.2.1. Note also that the inverse Fourier transform of the filter has both real and imaginary parts. Although the imaginary components of the filters with which we deal in this book typically are many orders of magnitude smaller than the real components, it is not good practice in general to ignore imaginary components in intermediate Fourier computations. Thus, both real and imaginary components are padded prior to generating the padded frequency domain filter and computation of the forward transform.

Figure 4.39 shows the padded spatial representation (only the real part is shown) of the ideal lowpass filter used to generate Fig. 4.12(c). The padded area of 0's is shown in black. An ideal lowpass filter was selected for illustration because it has the most visible “structure” in the spatial domain. The padding used
Figure 4.40 Result of filtering with padding. The image is usually cropped to its original size since there is little valuable information past the image boundaries.

was the minimum size required, which, when images and filters are squares of the same size, simply doubles the size in both dimensions.

Figure 4.40 shows the result of filtering with padded functions using the approach just discussed. It is easy to visualize how convolving the filter in Fig. 4.39 with a padded version of Fig. 4.12(a) would generate Fig. 4.40. It also is evident in this case that three-quarters of the result contain no valuable information, so cropping back to the original image gives the desired filtered result. We are assured by using padding that the cropped image is free of wraparound error.

4.6 The Convolution and Correlation Theorems

Convolution was introduced in Section 4.2.4 and its implementation was discussed in additional detail in Section 4.6.3. We repeat it briefly here to facilitate comparison with a similar process called correlation. The discrete convolution of two functions \( f(x, y) \) and \( h(x, y) \) of size \( M \times N \) is denoted by \( f(x, y) \ast h(x, y) \) and is defined by the expression

\[
f(x, y) \ast h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(m, n)h(x - m, y - n). \quad (4.6-27)
\]

From the discussion in Section 4.2.4, we know that the convolution theorem consists of the following relationships between the two functions and their Fourier transforms:

\[
f(x, y) \ast h(x, y) \iff F(u, v)H(u, v) \quad (4.6-28)
\]

and

\[
f(x, y)h(x, y) \iff F(u, v) \ast H(u, v). \quad (4.6-29)
\]
The correlation of two functions \( f(x, y) \) and \( h(x, y) \) is defined as

\[
f(x, y) \ast h(x, y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f^*(m, n) h(x+m, y+n) \quad (4.6-3)
\]

where \( f^* \) denotes the complex conjugate of \( f \). We normally deal with real functions (images), in which case \( f^* = f \). The correlation function has exactly the same form as the convolution function given in Eq. (4.6-27), with the exception of the complex conjugate and the fact that the second term in the summation has positive instead of negative signs. This means that \( h \) is not mirrored about the origin. Everything else in the implementation of correlation is identical to convolution, including the need for padding.

Given the similarity of convolution and correlation, it is not surprising that there is a correlation theorem, analogous to the convolution theorem. Let \( F(u, v) \) and \( H(u, v) \) denote the Fourier transforms of \( f(x, y) \) and \( h(x, y) \), respectively. One-half of the correlation theorem states that spatial correlation \( f(x, y) \ast h(x, y) \), and the frequency domain product, \( F^*(u, v)H(u, v) \), constitute a Fourier transform pair. This result, formally stated as

\[
f(x, y) \ast h(x, y) \Leftrightarrow F^*(u, v)H(u, v), \quad (4.6-3)
\]

indicates that correlation in the spatial domain can be obtained by taking the inverse Fourier transform of the product \( F^*(u, v)H(u, v) \), where \( F^* \) is the complex conjugate of \( F \). An analogous result is that correlation in the frequency domain reduces to multiplication in the spatial domain; that is,

\[
f^*(x, y)h(x, y) \Leftrightarrow F(u, v) \ast H(u, v). \quad (4.6-3)
\]

These two results comprise the correlation theorem. It is assumed that all functions have been properly extended by padding.

As we know by now, convolution is the tie between filtering in the spatial and frequency domains. The principal use of correlation is for matching. In matching, \( f(x, y) \) is an image containing objects or regions. If we want to determine whether \( f \) contains a particular object or region in which we are interested, let \( h(x, y) \) be that object or region (we normally call this image a template). Then, if there is a match, the correlation of the two functions will be maximum at the location where \( h \) finds a correspondence in \( f \). Preprocessing like scaling and alignment, is necessary in most practical applications, but the basic idea of the process is performing the correlation.

Finally, we point out that the term \textit{cross correlation} often is used in place of the term \textit{correlation} to clarify that the images being correlated are different. This is as opposed to \textit{autocorrelation}, in which both images are identical. In the latter case, we have the \textit{autocorrelation theorem}, which follows directly from Eq. (4.6-3):}

\[
f(x, y) \ast f(x, y) \Leftrightarrow |F(u, v)|^2, \quad (4.6-35)
\]

On the right side, we used the fact that the product of a complex quantity and its complex conjugate is the magnitude of the complex quantity squared. In other words, this result states that the Fourier transform of the spatial autocorrelation is the power spectrum defined in Eq. (4.2-20). Similarly,

\[
|f(x, y)|^2 \Leftrightarrow F(u, v) \ast F(u, v). \quad (4.6-36)
\]
Figure 4.41 shows a simple illustration of image padding and correlation. Figure 4.41(a) is the image and Fig. 4.41(b) is the template. The image and template are of size $256 \times 256$ and $38 \times 42$ pixels, respectively. In this case, $A = B = 256$, $C = 38$, and $D = 42$. This gives the minimum values for the expanded functions: $P = A + C - 1 = 293$ and $Q = B + D - 1 = 297$. We
chose equal padding dimensions of 298 × 298. The padded images are shown in Figs. 4.41(c) and (d). The spatial correlation of the two padded images is displayed as an image in Fig. 4.41(e). As indicated in Eq. (4.6-31), the correlation function was obtained by computing the transforms of the padded images, taking the complex conjugate of one of them (we chose the template), multiplying the two transforms, and computing the inverse DFT. It is left as an exercise for the reader (Problem 4.23) to discuss what Fig. 4.41(e) would look like if we had taken the complex conjugate of the other transform instead.

As expected, we see in Fig. 4.41(e) that the highest value of the correlation function occurs at the point where the template is exactly on top of the “T” in the image. As in convolution, it is important to keep in mind that the variables in the correlation function in the spatial domain are displacements. For example, the top left corner of Fig. 4.41(e) corresponds to zero displacement of one function with respect to the other. The value of each pixel in Fig. 4.41(e) is the value of the correlation function at one location of displacement; that is, for one specific value of the pair (x, y) in Eq. (4.6-30). Also, we note that the correlation function has the same dimensions as the padded images. Finally, Fig. 4.41(f) shows a horizontal gray-level profile passing through the highest value in Fig. 4.41(e). This figure simply confirms that the highest peak in the correlation function is located at the point where the best match of the template and the image occurs.

4.6.5 Summary of Properties of the 2-D Fourier Transform

All the properties of the Fourier transform discussed in this chapter are summarized in Table 4.1. A footnote identifies the items requiring that functions be padded in order to avoid incorrect results. As before, the double arrows are used to denote that the expressions form a Fourier transform pair. That is, the expression on the right of the double arrows is obtained by taking the forward Fourier transform of the expression on the left; the expression on the left is obtained by taking the inverse Fourier transform of the expression on the right.

4.6.6 The Fast Fourier Transform

As indicated in Section 4.1, one of the main reasons that the DFT has become an essential tool in signal processing was the development of the fast Fourier transform (FFT). Computing the 1-D Fourier transform of M points using Eq. (4.25) directly requires on the order of M² multiplication/addition operations. The FFT accomplishes the same task on the order of M log₂ M operations. If, for example, M = 1024, the brute-force method will require approximately 10⁶ operations, while the FFT will require approximately 10⁴ operations. This is a computational advantage of 100 to 1. If this advantage does not seem significant, imagine being able to complete a given project in one year as opposed to 100. This is the difference between possible and practically impossible. And the story gets better. The bigger the problem, the greater the computational advantage. If, for instance, M = 8192 (2¹³), the computational advantage grows to 600 to 1. These type of numbers are great motivators for wanting to learn more about how an FFT algorithm works. In this section we take a look at the derivation of a fundamental decomposition of the DFT that leads to the FFT. The focus is on the FFT of an