Compute output image pixel G[r, c] from neighbors of input image pixel F[r, c].

F[r, c] is an input image of **MaxRow** rows and **MaxCol** columns;
F is unchanged by the algorithm.
G[r, c] is the output image of **MaxRow** rows and **MaxCol** columns.
The border of **G** are all those pixels whose neighborhoods
are not wholly contained in **G**.
w and h are the width and height, in pixels, defining a neighborhood.

```
procedure enhance_image(F,G,w,h);
{
for r := 0 to MaxRow - 1
   for c := 0 to MaxCol - 1
      {
      if [r, c] is a border pixel then G[r, c] := F[r, c];
      else G[r, c] := compute_using_neighbors (F, r, c, w, h);
      } ;
}
procedure compute_using_neighbors (IN, r, c, w, h)
{
using all pixels within w/2 and h/2 of pixel IN[r, c],
compute a value to return to represent IN[r, c]
}
```

Algorithm 5.1 Compute output image pixel **G[r, c]** from neighbors of input image pixel **F[r, c]**.

Exercise 5.7

Implement Algorithm 5.1 in some programming language. Code both the boxcar and median filtering operations and test them on some images such as in Figure 5.9.

could all be computed in parallel. This holds because the input image is unchanged by any of the neighborhood computations. Secondly, the procedure compute_using_neighbors could be implemented to perform either boxcar or median filtering. For boxcar filtering, the procedure need only add up the $w \times h$ pixels of the neighborhood of **F[r, c]** and then divide by the number of pixels $w \times h$. To implement a median filter, the procedure could copy those $w \times h$ pixel values into a local array **A** and partially sort it to obtain their median value.

Control could also be arranged so that only h rows of the image are in main memory at any one time. Outputs **G[r, c]** would be computed only for the middle row r. Then, a new row would be input to replace the oldest row in memory and the next output row of **G[r, c]** would be computed. This process is repeated until all possible output rows are computed.

Years ago, when computers had small main memories, the primary storage for images was on disk and many algorithms had to process images a few rows at a time. Today, such control is still of interest because it is used in image processing boards implementing a pipelined architecture.

5.6 DETECTING EDGES USING DIFFERENCING MASKS

Image points of high contrast can be detected by computing intensity differences in local image regions. Typically, such points form the border between different objects or scene parts. In this section, we show how to do this using neighborhood templates or *masks*. We start by using one-dimensional signals: this helps develop both the intuition and formalism, and is also very important in its own right. The 1D signals could just be rows or columns of a 2D image. The section ends by studying more general 2D situations.

5.6.1 Differencing 1D Signals

Figure 5.10 shows how masks can be used to compute representations of the derivatives of a signal. Given that the signal **S** is a sequence of samples from some function f, then $f'(x_i) \approx (f(x_i) - f(x_{i-1})/(x_i - x_{i-1})$. Assuming that the sample spacing is $\Delta x = 1$, the derivative of $f(x)$ can be approximated by applying the mask $\mathbf{M}' = [-1, 1]$ to the samples in **S** as shown in Figure 5.10 to obtain an output signal **S′**. As the figure shows,

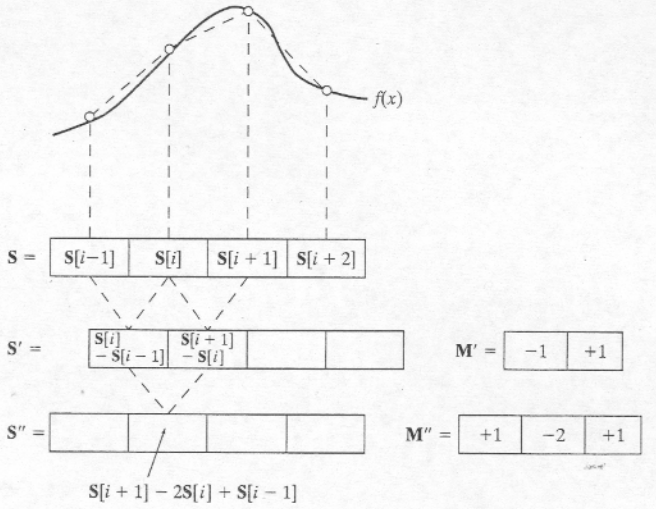


Figure 5.10 (left) The first (**S′**) and second (**S″**) difference signals are scaled approximations to the first and second derivatives of the signal **S**; and (right) Masks **M′** and **M″** represent the derivative operations.

it's convenient to think of the values of $\mathbf{S}'$ as occurring in between the samples of $\mathbf{S}$. A high absolute value of $\mathbf{S}'[i]$ indicates where the signal is undergoing rapid change or *high contrast*. Signal $\mathbf{S}'$ itself can be differentiated a second time using mask $\mathbf{M}'$ to produce output $\mathbf{S}''$ which corresponds to the second derivative of the original function f. The important result illustrated in Figure 5.10 and derived in the equations below is that the approximate second derivative can be computed by applying the mask $\mathbf{M}''$ to the original sequence of samples $\mathbf{S}$.

$$\mathbf{S}'[i] = -S[i-1] + S[i] \tag{5.3}$$

$$\text{mask } \mathbf{M}' = [-1, +1] \tag{5.4}$$

$$\mathbf{S}''[i] = -S'[i] + S'[i+1] \tag{5.5}$$

$$= -(S[i] - S[i-1]) + (S[i+1] - S[i]) \tag{5.6}$$

$$= S[i-1] - 2S[i] + S[i+1] \tag{5.7}$$

$$\text{mask } \mathbf{M}'' = [1, -2, 1] \tag{5.8}$$

If only points of high contrast are to be detected, it is very common to use the absolute value after applying the mask at signal position $\mathbf{S}[i]$. If this is done, then the first derivative mask can be either $\mathbf{M}' = [-1, +1]$ or $[+1, -1]$ and the second derivative mask can be either

mask M = [−1, 0, 1]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	⊗	M	0	0	0	0	12	12	0	0	0	0

(a) S_1 is an upward step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	⊗	M	0	0	0	0	−12	−12	0	0	0	0

(b) S_2 is a downward step edge

S_3			12	12	12	12	15	18	21	24	24	24
S_3	⊗	M	0	0	0	3	6	6	6	3	0	0

(c) S_3 is an upward ramp

S_4			12	12	12	12	24	12	12	12	12	12
S_4	⊗	M	0	0	0	12	0	−12	0	0	0	0

(d) S_4 is a bright impulse or *line*

Figure 5.11 Cross correlation of four special signals with first derivative edge detecting mask $[-1, 0, 1]$; (a) upward step edge, (b) downward step edge, (c) upward ramp, and (d) bright impulse. Note that, since the coordinates of $\mathbf{M}$ sum to zero, output must be zero on a constant region.

mask M = [−1, 2, −1]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	⊗	M	0	0	0	0	−12	12	0	0	0	0

(a) S_1 is an upward step edge

S_2			24	24	24	24	24	12	12	12	12	12
S_2	⊗	M	0	0	0	0	12	−12	0	0	0	0

(b) S_2 is a downward step edge

S_3			12	12	12	12	15	18	21	24	24	24
S_3	⊗	M	0	0	0	−3	0	0	0	3	0	0

(c) S_3 is an upward ramp

S_4			12	12	12	12	24	12	12	12	12	12
S_4	⊗	M	0	0	0	−12	24	−12	0	0	0	0

(d) S_4 is a bright impulse or *line*

Figure 5.12 Cross correlation of four special signals with second derivative edge detecting mask $\mathbf{M} = [-1, 2, -1]$; (a) upward step edge, (b) downward step edge, (c) upward ramp, and (d) bright impulse. Since the coordinates of $\mathbf{M}$ sum to zero, response on constant regions is zero. Note how a *zero-crossing* appears at an output position where different trends in the input signal join.

$\mathbf{M}'' = [+1, -2, +1]$ or $[-1, +2, -1]$. A similar situation exists for 2D images as we shall soon see. Whenever only magnitudes are of concern, we will consider these patterns to be the same, and whenever the sign of the change is important, we will consider them to be different.

Use of another common first derivative mask is shown in Figure 5.11. This mask has 3 coordinates and is centered at signal point $\mathbf{S}[i]$ so that it computes the signal difference across the adjacent values. Because $\Delta x = 2$, it will give a high estimate of the actual derivative unless the result is divided by 2. Moreover, this mask is known to give a response on perfect step edges that is 2 samples wide, as is shown in Figure 5.11(a)–(b). Figure 5.12 shows the response of the second derivative mask on the sample signals. As Figure 5.12 shows, signal contrast is detected by a zero-crossing, which localizes and amplifies the change between two successive signal values. Taken together, the first and second derivative signals reveal much of the local signal structure. Figure 5.13 shows how smoothing of signals can be put into the same framework as differencing: the boxed table below compares the general properties of smoothing versus differencing masks.

Some properties of derivative masks follow:

- Coordinates of derivative masks have opposite signs in order to obtain a high response in signal regions of high contrast.

box smoothing mask M = [1/3, 1/3, 1/3]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	⊗	M	12	12	12	12	16	20	24	24	24	24

(a) S_1 is an upward step edge

S_4			12	12	12	12	24	12	12	12	12	12
S_4	⊗	M	12	12	12	16	16	16	12	12	12	12

(d) S_4 is a bright impulse or *line*

Gaussian smoothing mask M = [1/4, 1/2, 1/4]

S_1			12	12	12	12	12	24	24	24	24	24
S_1	⊗	M	12	12	12	12	15	21	24	24	24	24

(a) S_1 is an upward step edge

S_4			12	12	12	12	24	12	12	12	12	12
S_4	⊗	M	12	12	12	15	18	15	12	12	12	12

(d) S_4 is a bright impulse or *line*

Figure 5.13 (top two rows) Smoothing of step and impulse with box mask [1/3, 1/3, 1/3]; and (bottom two rows) smoothing of step and impulse with Gaussian mask [1/4, 1/2, 1/4].

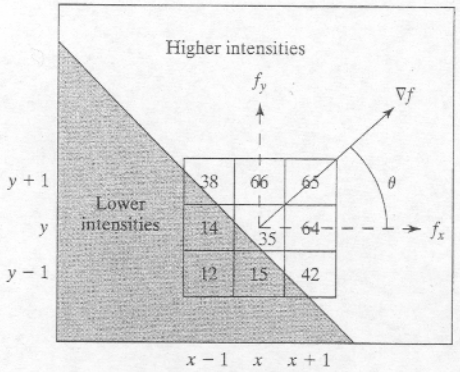
- The sum of coordinates of derivative masks is zero so that a zero response is obtained on constant regions.
- First derivative masks produce high absolute values at points of high contrast.
- Second derivative masks produce zero-crossings at points of high contrast.

For comparison, smoothing masks have these properties:

- Coordinates of smoothing masks are positive and sum to one so that output on constant regions is the same as the input.
- The amount of smoothing and noise reduction is proportional to the mask size.
- Step edges are blurred in proportion to the mask size.

5.6.2 Difference Operators for 2D Images

Contrast in the 2D picture function $f(x, y)$ can occur in any direction. From calculus, we know that the maximum change occurs along the direction of the gradient of the function, which is in the direction $[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}]$ in the picture plane. Figure 5.14 shows that this is quite intuitive when one considers discrete approximations in digital images. We can estimate



$f_y = ((38 - 12)/2 + (66 - 15)/2 + (65 - 42)/2)/3 = (13 + 25 + 11)/3 = 16$

$f_x = ((65 - 38)/2 + (64 - 14)/2 + (42 - 12)/2)/3 = (13 + 25 + 15)/3 = 18$

$\theta = \tan^{-1}(16/18) = 0.727$ rad $= 42$ degrees

$|\nabla f| = (16^2 + 18^2)^{1/2} = 24$

Figure 5.14 Estimating the magnitude and direction of contrast at **I[x, y]** by estimating the gradient magnitude and direction of picture function $f(x, y)$ using the discrete samples in the image array.

the contrast at image location **I[x, y]** along the x-direction by computing $(I[x + 1, y] - I[x - 1, y])/2$, which is the change in intensity across the left and right neighbors of pixel **[x, y]** divided by $\Delta x = 2$ pixel units. For the neighborhood shown in Figure 5.14, the constrast in the x direction would be estimated as $(64 - 14)/2 = 25$. Since our pixel values are noisy and since the edge might actually cut across the pixel array at any angle, it should help to average 3 different estimates of the contrast in the neighborhood of **[x, y]:**

$$\begin{aligned} \partial f/\partial x \equiv f_x \approx \tfrac{1}{3}[&(I[x+1, y] - I[x-1, y])/2 \\ &+ (I[x+1, y-1] - I[x-1, y-1])/2 \\ &+ (I[x+1, y+1] - I[x-1, y+1])/2)] \end{aligned} \tag{5.9}$$

This estimates the contrast in the x-direction by equally weighting the contrast across the row y with the row below and above it. The contrast in the y-direction can be estimated similarly:

$$\begin{aligned} \partial f/\partial y \equiv f_y \approx \tfrac{1}{3}[&(I[x, y+1] - I[x, y-1])/2 \\ &+ (I[x-1, y+1] - I[x-1, y-1])/2 \\ &+ (I[x+1, y+1] - I[x+1, y-1])/2)] \end{aligned} \tag{5.10}$$

Often, the division by 6 is ignored to save computation time, resulting in scaled estimates. Masks for these two contrast operators are denoted by M_x and M_y at the top of Figure 5.15. The gradient of the picture function can be estimated by applying the masks to the 8-neighborhood $N_8[x, y]$ of pixel **[x, y]** as given in Equations 5.11 to 5.14. These

Prewitt: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$

Sobel: $M_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

Roberts: $M_x = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$; $M_y = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$

Figure 5.15 Masks used to estimate the gradient of picture function $f(x, y)$: (top row) Prewitt; (middle row) Sobel; and (bottom row) Roberts.

masks define the *Prewitt operator* credited to Dr. Judith Prewitt who used them in detecting boundaries in biomedical images.

$$\frac{\partial f}{\partial x} \approx (1/6)(M_x \circ N_8[x, y]) \tag{5.11}$$

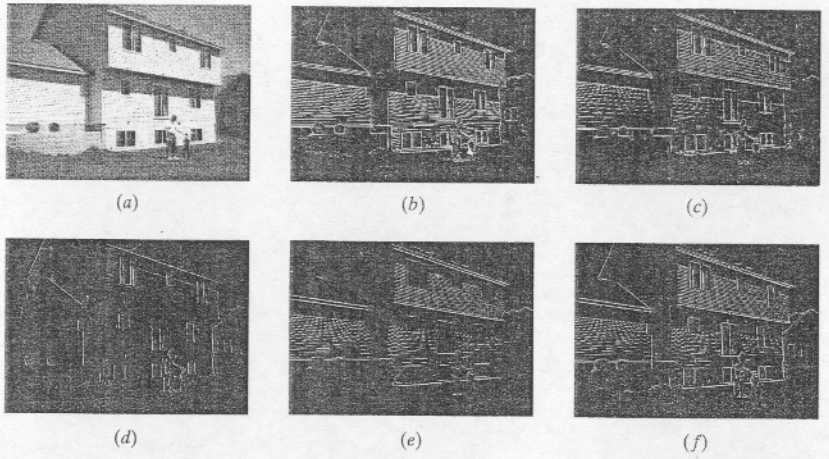
$$\frac{\partial f}{\partial y} \approx (1/6)(M_y \circ N_8[x, y]) \tag{5.12}$$

$$|\nabla f| \approx \sqrt{\frac{\partial f}{\partial x}^2 + \frac{\partial f}{\partial y}^2} \tag{5.13}$$

$$\theta \approx tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right) \tag{5.14}$$

The operation $M \circ N$ is defined formally in the next section: operationally, mask M is overlaid on image neighborhood N so that each intensity N_{ij} can be multiplied by weight M_{ij}; finally all these products are summed. The middle row of Figure 5.15 shows the two analogous *Sobel masks;* their derivation and interpretation is the same as for the Prewitt masks except that the assumption is that the center estimate should be weighted twice as much as the estimate to either side.

The *Roberts masks* are only 2×2; hence they are both more efficient to use and more locally applied. Often referred to as *the Roberts cross operator,* these masks actually compute a gradient estimate at the center of a 4-neighborhood and not at a center pixel. Moreover, the actual coordinate system in which the operator is defined is rotated 45° off the standard row direction. Application of the Roberts cross operator is shown in Figure 5.16: the original input image is shown in the upper left in part (a), while the output from slightly different Roberts operators is shown in (b) and (c). Parts (d) and (e) show the results of using just the intensity differences along the image columns and rows respectively, and (f) shows the row and column detections ORed together. Qualitatively, these results are typical of the several small-neighborhood operators—many pixels on many edges are detected but many are missed. There are responses on the textured grass region and the top of the garage is missed because its intensity matches the sky. The Roberts results should be compared to the results obtained by combining the simple 1D row and column masks



(a) (b) (c) (d) (e) (f)

Figure 5.16 (a) Original image; (b) top 5 percent of the sum of the absolute values of the two Roberts mask response; (c) top 5 percent of the mean-squared value of the two Roberts masks; (d) top 2 percent by absolute value of the responses from y-direction edge mask $[-1, +1]$; (e) top 3 percent of the responses from x-direction edge mask $[-1, +1]^t$; and (f) images of (d) and (e) ORed together. There are minor differences among (b), (c), and (f). (Image courtesy of Ida Stockman.)

(a) (b) (c)

Figure 5.17 (a) Image of noisy squares and rings; (b) mean square response of 3×3 Sobel operator; and (c) coding of gradient direction computed by 3×3 Sobel operator.

shown in Figure 5.16$(d$-$f)$. It is common to avoid computing a square root in order to compute gradient magnitude; alternatives are $max(|\frac{\partial f}{\partial x}|, |\frac{\partial f}{\partial y}|)$, $|\frac{\partial f}{\partial x}| + |\frac{\partial f}{\partial y}|$, or $(\frac{\partial f^2}{\partial x} + \frac{\partial f^2}{\partial y})/2$. Comparing Figure 5.16$(b)$ and (c, f) seems to justify avoiding the square root operation. With these estimates, one must be careful when trying to interpret the actual gradient or gradient direction. Figure 5.17(b) shows the results of using the Sobel 3×3 operator to compute mean square gradient magnitude and Figure 5.17(c) shows an encoding of the gradient direction. The small squares in the original image are 8×8 pixels: the Sobel operator represents many, but not all, of the image edges.

Exercise 5.8

If a true gradient magnitude is needed, why might the Sobel masks provide a faster solution than the Prewitt masks?

Exercise 5.9: Optimality of Prewitt masks.*

The problem is to prove that Prewitt masks give the weights for the best-fitting plane approximating the intensity surface in a 3×3 neighborhood, assuming all 9 samples have equal weight. Suppose that the 9 intensities $I[r+i, c+j]$; $i, j = -1, 0, 1$ of a 3×3 image neighborhood are fit by the least squares planar model $I[r, c] = z = pr + qc + z_0$. (Recall that the 9 samples are equally spaced in terms of r and c.) Show that the Prewitt masks compute the estimates of p and q as the partial derivatives of the least squares planar fit of the intensity function.

Figure 5.18(b, c) show plots of the intensities of two rows of the room scene in (a). The lower row (b) cuts across four dark regions as seen in the image and the plot; (1) the coat on the chair at the left (columns 20 to 80), (2) Dr. Prewitt's chair and dress in the center (columns 170 to 240), (3) the shadow of the rightmost chair (columns 360 to 370) and (4) the electric wires (column 430). Note that the transitions between dark and bright are sharp except for the boundary between the chair and its shadow, which ramps down from brightness 220 to 20 over about 10 pixels. The upper row profile, shown in (c), shows sharp transitions where

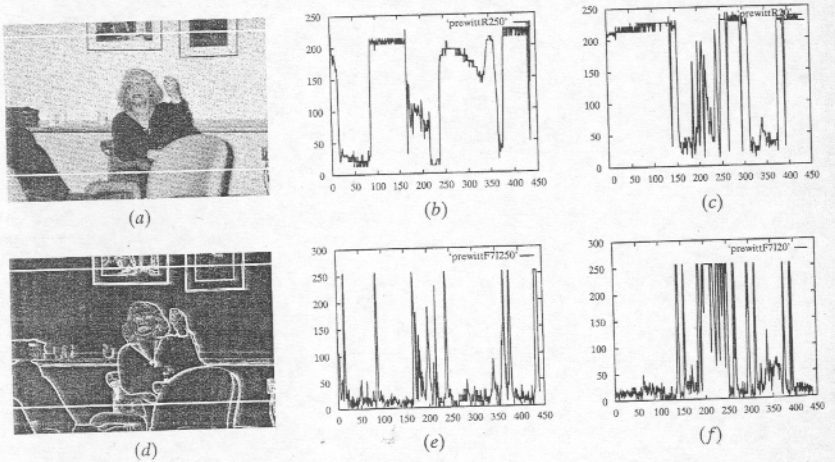


Figure 5.18 (a) Image of Judith Prewitt with two rows selected; (b) plot of intensities along selected lower row; (c) plot of intensities along selected upper row; (d) *gradient image* showing result of $|f_x| + |f_y|$ using the Prewitt 3×3 operator; (e) plot of selected lower row of gradient image; and (f) plot of intensities along gradient of selected upper row.

it cuts across the picture frames, mat, and pictures. The picture at the left shows much more intensity variation than the one at the right. Figure 5.18(d)–(f) shows the results of applying the 3×3 Prewitt gradient operator to the original image. The sum of the absolute values of the column and row gradients f_x and f_y is plotted for the same two image rows shown in parts (a)–(c) of the figure. The highest values of the Prewitt operator correspond well with the major boundaries crossed; however, the several medium level spikes from Dr. Prewitt's chair in (d) between columns 170 and 210 are harder to interpret. The contrasts in the upper row, graphed in (f), are interpreted similarly—major object boundaries correspond well to the object boundaries of the picture frame and mat; however, there is a lot of intensity variation in the leftmost picture on the wall. Generally, gradient operators work well for detecting the boundary of isolated objects, although some problems are common. Boundaries sometimes drop out due to object curvature or soft shadows: on the other hand, good contrast often produces boundaries that are several pixels wide, necessitating a thinning step afterward. Gradient operators will also respond to textured regions, as we shall study in more detail in Chapter 7.

5.7 GAUSSIAN FILTERING AND LOG EDGE DETECTION

The Gaussian function has important applications in many areas of mathematics, including image filtering. In this section, we highlight the characteristics that make it useful for smoothing images or detecting edges after smoothing.

46 Definition. A **Gaussian function** of one variable with spread σ is of the following form, where c is some scale factor.

$$g(x) = ce^{-\frac{x^2}{2\sigma^2}} \tag{5.15}$$

A Gaussian function of two variables is

$$g(x, y) = ce^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{5.16}$$

These forms have the same structure as the normal distribution defined in Chapter 4, where the constant c was set so that the area under the curve would be 1. To create a mask for filtering, we usually make c a large number so that all mask elements are integers. The Gaussian is defined centered on the origin and thus needs no location parameter μ as does the normal distribution: an image processing algorithm will translate it to wherever it will be applied in a signal or image. Figure 5.19 plots the Gaussian of one variable along with its first and second derivatives, which are also important in filtering operations. The derivation of these functions is given in Equations 5.17 to 5.22. The area under the function $g(x)$ is 1, meaning that it is immediately suitable as a smoothing filter that does not affect constant regions, $g(x)$ is a positive even function; $g'(x)$ is just $g(x)$ multiplied by odd function $-x$ and scaled down by σ^2. More structure is revealed in $g''(x)$. Equation 5.21 shows that $g''(x)$ is the difference of two even functions and that the central lobe will be negative with $x \approx 0$. Using Equation 5.22, it is clear that the zero crossings of the second derivative occur at