

Original Research Paper

Digital High-Pass Filters with Milder High-Pass Effect on Digital Images

Issa A. Al-Shakhrh

Department of Physics, University of Jordan, Amman, Jordan

Article history

Received: 29-10-2014

Revised: 04-06-2015

Accepted: 04-06-2015

Abstract: Image filtering consists of modifying the original image by logically “reimaging” it with a mathematical imaging device in which spatial response can be controlled by the user. Image filtering is performed by a mathematical operation called convolution, which is simply the successive replacement of each point in the original image by a new value produced by a weighted combination of the original point and its surrounding neighbour points. Filtering generally requires definition of a filtering kernel or small matrix; often a few filtering kernels are predefined in imaging computer systems. The filtering kernel is generally square with a matrix size of 3×3 pixels, 5×5 pixels or 7×7 pixels. We consider the use of two-dimensional, second-order derivatives for image enhancement. The approach basically consists of defining a discrete formulation of the second-order derivative and then constructing a filter mask based on that formulation. Ten spatial high-pass filters (masks) are developed, then implemented and tested in our laboratory by using programs that were written in Borland c^{++} and visual Fortran. The results of the application of the developed Laplacian and Laplacian high-pass digital filters (masks) on digital images (either edge detection, sharpening of high frequency regions (fine details) accentuation), comparing between the effect of different dimensions filters 3×3 and 5×5 and milder high pass effect are presented and demonstrated. As the size of the filter (mask) gets larger and/or the weight of the center pixel of the kernel gets higher, the sharpening effect becomes more and more. Second-order derivatives have a strong response to fine detail, such as thin lines and isolated points.

Keywords: High-Pass Digital Filters, Laplacian Masks, Image Enhancement, Milder High-Pass Effect

Introduction

Image processing, with the intent of improving display information, was one of the first applications of the computer in nuclear medicine (Brown *et al.*, 1971). Image filtering consists of modifying the original image by logically “reimaging” it with a mathematical imaging device in which spatial response can be controlled by the user. The difference between a logical and a real imaging device (e.g., scintillation camera) is that the response function of the logical device can have negative values, whereas that of the real device can only be positive. The practical result of this difference is that the logical imaging device used for image filtering can be specified to improve spatial resolution and make edge and count

density transitions more obvious. Alternatively, the response can be specified so that filtering smooths the image and reduces the noise so that small differences in count densities can be more readily perceived by the viewer (Erickson, 1985).

More information on general image processing may be obtained from different references (Gonzalez and Witz, 1977; Jain, 1988; Gonzalez and Woods, 2002) and more detailed descriptions of various nuclear medicine analysis techniques may be found in Gottschalk (1988; Erickson and Rollo, 1993).

The term filtering is used to indicate operations that either smooth or sharpen images; that is, increase or decrease image blurring. In planar nuclear medicine applications, however, only smoothing is frequently

employed. While filtering can be described mathematically in complex terms, the actual operation is usually simple (Faber, 1994).

Filtering generally requires definition of a filtering kernel or small matrix; often a few filtering kernels are predefined in imaging computer systems. The filtering kernel is generally square with a matrix size of 3×3 pixels, 5×5 pixels or 7×7 pixels. The numbers contained in the matrix are called weights. In a sense, the kernel is placed so that its middle pixel is on top of one pixel in the image. The kernel will then cover a 3×3 (or 5×5, etc.) pixel region of the image (Faber, 1994).

The high-pass filter accentuates high-frequency spatial components while leaving low-frequency components untouched. A common high-pass mask is composed of a 9 in the center location with -1s in the surrounding locations as shown below (L1 Mask):

L1 Mask

-1	-1	-1
-1	9	-1
-1	-1	-1

We can see that the coefficients add to 1 and that smaller coefficients surrounded the large positive center coefficient.

The principal objective of sharpening is to highlight fine detail in an image or to enhance detail that has been blurred, either in error or as a natural effect of a particular method of image acquisition. Uses of image sharpening vary and include applications ranging from electronic printing and medical imaging to industrial inspection and autonomous guidance in military systems (Gregory, 1994).

The aim of the present study was to construct digital spatial high-pass filters (masks), applying these filters on digital images and comparing between the efficiency (enhancement the appearance of fine details in the image, tracing the borders (edges) of the small and large regions in the image), comparing between the effect of different dimensions filters 3×3 and 5×5 and milder high pass effect of these filters.

Theory

We consider in some detail sharpening filters that are based on first- and second-order derivatives, respectively.

The derivatives of a digital function are defined in terms of differences. A basic definition of the first-order derivative of a one-dimensional function $f(x)$ is the difference Equation 1:

$$\frac{\partial f}{\partial x} = f(x+1) - f(x) \quad (1)$$

We used a partial derivative here in order to keep the notation the same as when we consider an image function of two variables, $f(x, y)$, at which time we will be dealing with partial derivatives along the two spatial axes.

Similarly, we define a second-order derivative as the difference Equation 2:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1) + f(x-1) - 2f(x) \quad (2)$$

In most applications, the second derivative is better suited than the first derivative for image enhancement because of the ability of the former to enhance fine detail. For this and for reasons of simpler implementation and extensions, we will focus attention initially on uses of the second derivative for enhancement.

Use of Second Derivatives for Enhancement-the Laplacian

We consider the use of two-dimensional, second-order derivatives for image enhancement. The approach basically consists of defining a discrete formulation of the second-order derivative and then constructing a filter mask based on that formulation.

It can be shown (Rosenfeld and Kak, 1982) that the simplest isotropic derivative operator is the Laplacian, which, for a function (image) $f(x, y)$ of two variables, is defined as:

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} \quad (3)$$

In order to be useful for digital image processing, this equation needs to be expressed in discrete form. There are several ways to define a digital Laplacian using neighborhoods. Taking into account that we now have two variables, we use the following notation for the partial second-order derivative in the x-direction Equation 4:

$$\frac{\partial^2 f}{\partial x^2} = f(x+1, y) + f(x-1, y) - 2f(x, y) \quad (4)$$

And, similarly in the y-direction, as:

$$\frac{\partial^2 f}{\partial y^2} = f(x, y+1) + f(x, y-1) - 2f(x, y) \quad (5)$$

The digital implementation of the two-dimensional Laplacian in Equation 3 is obtained by summing these two components Equation 6:

$$\nabla^2 f = \begin{bmatrix} f(x+1, y) + f(x-1, y) \\ + f(x, y+1) + f(x, y-1) \end{bmatrix} - 4f(x, y) \quad (6)$$

This equation can be implemented using the mask shown below (Laplacian digital mask L2 Mask):

L2 Mask			
0	1		0
1	-4		1
0	1		0

The diagonal directions can be incorporated in the definition of the digital Laplacian by adding two more terms to Equation 5, one for each of the two diagonal directions. Since each diagonal term also contains a $-2f(x, y)$ term, the total subtracted from the difference terms now would be $-8f(x, y)$. The mask used to implement this new definition is shown in the following mask (L3 Mask):

L3 Mask			
1	1		1
1	-8		1
1	1		1

Because the Laplacian is a derivative operator, its use highlights gray-level discontinuities in an image and deemphasizes regions with slowly varying gray levels. This will tend to produce images that have grayish edge lines and other discontinuities, all superimposed on a dark, featureless background. Background features can be "recovered" while still preserving the sharpening effect of the Laplacian operation simply by adding the original and Laplacian images. It is important to keep in mind which definition of the Laplacian is used. If the definition used has a negative center coefficient, then we *subtract*, rather than add, the Laplacian image to obtain a sharpened result. Thus, the basic way in which we use the Laplacian for image enhancement is as follows Equation 6b:

$$g(x, y) = f(x, y) - \nabla^2 f(x, y) \quad (6a)$$

if the center coefficient of the Laplacian mask is negative

21	115	21	115	21	115	21
115	21	115	21	115	21	115
21	115	21	115	21	115	21
115	21	115	21	115	21	115
21	115	21	115	21	115	21
115	21	115	21	115	21	115
21	115	21	115	21	115	21

(A)

$$g(x, y) = f(x, y) + \nabla^2 f(x, y) \quad (6b)$$

if the center coefficient of the Laplacian mask is positive

All the constructed high-pass filters depend basically on Laplacian masks (L2 and L3 masks) and Laplacian image that is represented mathematically by Equation (6a).

Simplifications

In practice, Equation 6a is usually implemented with one pass of a single mask. The coefficients of the single mask are easily obtained by substituting Equation 5 for $\nabla^2 f(x, y)$ in Equation 6a:

$$g(x, y) = f(x, y) - \left[\begin{matrix} f(x+1, y) + f(x-1, y) \\ + f(x, y+1) + f(x, y-1) \end{matrix} \right] + 4f(x, y)$$

$$= 5f(x, y) - \left[\begin{matrix} f(x+1, y) + f(x-1, y) \\ + f(x, y+1) + f(x, y-1) \end{matrix} \right]$$

Methods

Image filtering is performed by a mathematical operation called convolution, which is simply the successive replacement of each point in the original image by a new value produced by a weighted combination of the original point and its surrounding neighbor points. Using larger filter functions, such as 5×5 or 7×7 , will cause larger portions of the original image to have an effect on the value of the new image points (3).

The counts (density) in the image pixels underneath each element of the 3×3 kernel are multiplied by the kernel weights and the results are all summed together. The image pixel beneath the center of the kernel is replaced by this summation. This operation is repeated for every image pixel. Figure 1 shows the process graphically for 3×3 filter function. In this process, each image pixel is replaced by a weighted average of itself and some of its neighbours in its immediate neighbourhood.

(1/40)

1	-2	1
-2	44	-2
1	-2	1

(B)

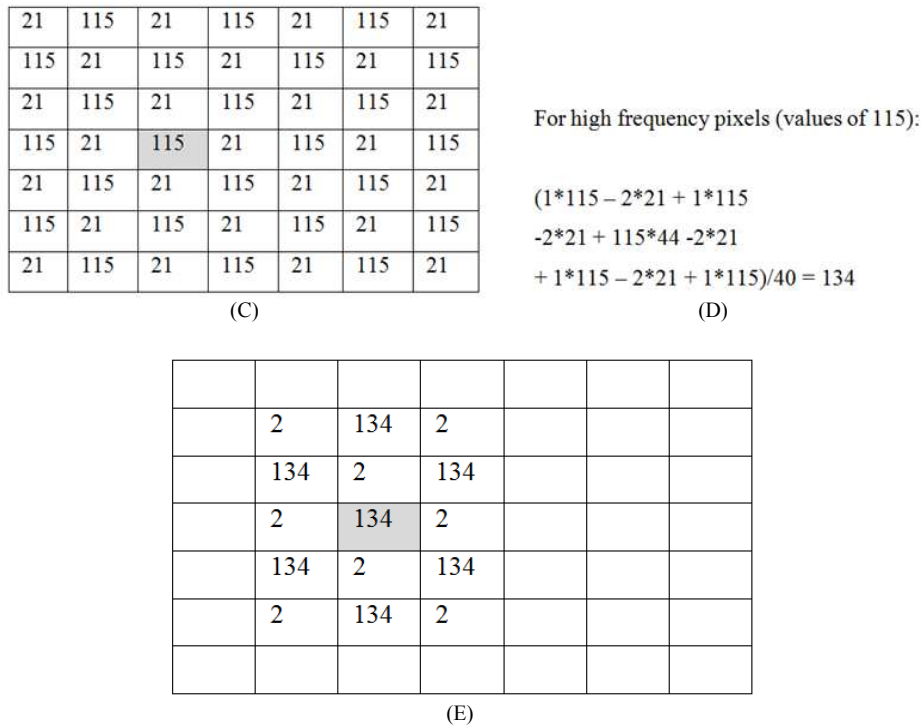


Fig. 1. Sharpening high-passed spatial filtering operation. (A) The original image matrix. (B) The 3 x 3 sharpening kernel. (C) Centering the kernel over one pixel of the image. (D) Weighted average of the center pixel and its neighbours: each pixel under the kernel is multiplied by the kernel weight over it, and the results are summed together. (E) Steps in C-D repeated for each image pixel; this is the final high-passed spatial sharpened image

The following spatial high-pass filters (masks) are developed or constructed, then implemented and tested in our laboratory by using programs that were written in Borland c++ and visual Fortran as shown in Appendices 1 and 2:

<p>Mask 1 (1/12)</p> <table style="margin-left: 20px;"> <tr><td>-1</td><td>-1</td><td>-1</td></tr> <tr><td>-1</td><td>20</td><td>-1</td></tr> <tr><td>-1</td><td>-1</td><td>-1</td></tr> </table> <p>Mask 2 (1/24)</p> <table style="margin-left: 20px;"> <tr><td>-1</td><td>-1</td><td>-1</td></tr> <tr><td>-1</td><td>32</td><td>-1</td></tr> <tr><td>-1</td><td>-1</td><td>-1</td></tr> </table> <p>Mask 3 (1/36)</p> <table style="margin-left: 20px;"> <tr><td>-1</td><td>-1</td><td>-1</td></tr> <tr><td>-1</td><td>44</td><td>-1</td></tr> <tr><td>-1</td><td>-1</td><td>-1</td></tr> </table> <p>Mask 4</p> <table style="margin-left: 20px;"> <tr><td>0</td><td>-1</td><td>0</td></tr> <tr><td>-1</td><td>4</td><td>-1</td></tr> <tr><td>0</td><td>-1</td><td>0</td></tr> </table>	-1	-1	-1	-1	20	-1	-1	-1	-1	-1	-1	-1	-1	32	-1	-1	-1	-1	-1	-1	-1	-1	44	-1	-1	-1	-1	0	-1	0	-1	4	-1	0	-1	0	<p>Mask 5 (1/28)</p> <table style="margin-left: 20px;"> <tr><td>0</td><td>-1</td><td>0</td></tr> <tr><td>-1</td><td>32</td><td>-1</td></tr> <tr><td>0</td><td>-1</td><td>0</td></tr> </table> <p>Mask 6 (1/40)</p> <table style="margin-left: 20px;"> <tr><td>1</td><td>-2</td><td>1</td></tr> <tr><td>-2</td><td>44</td><td>-2</td></tr> <tr><td>1</td><td>-2</td><td>1</td></tr> </table> <p>Mask 7 (1/28)</p> <table style="margin-left: 20px;"> <tr><td>1</td><td>-2</td><td>1</td></tr> <tr><td>-2</td><td>32</td><td>-2</td></tr> <tr><td>1</td><td>-2</td><td>1</td></tr> </table> <p>Mask 8 (1/42)</p> <table style="margin-left: 20px;"> <tr><td>0</td><td>-2</td><td>0</td></tr> <tr><td>-2</td><td>50</td><td>-2</td></tr> <tr><td>0</td><td>-2</td><td>0</td></tr> </table> <p>Mask 9 (1/32)</p> <table style="margin-left: 20px;"> <tr><td>-1</td><td>-1</td><td>-1</td></tr> <tr><td>-1</td><td>40</td><td>-1</td></tr> <tr><td>-1</td><td>-1</td><td>-1</td></tr> </table>	0	-1	0	-1	32	-1	0	-1	0	1	-2	1	-2	44	-2	1	-2	1	1	-2	1	-2	32	-2	1	-2	1	0	-2	0	-2	50	-2	0	-2	0	-1	-1	-1	-1	40	-1	-1	-1	-1
-1	-1	-1																																																																																
-1	20	-1																																																																																
-1	-1	-1																																																																																
-1	-1	-1																																																																																
-1	32	-1																																																																																
-1	-1	-1																																																																																
-1	-1	-1																																																																																
-1	44	-1																																																																																
-1	-1	-1																																																																																
0	-1	0																																																																																
-1	4	-1																																																																																
0	-1	0																																																																																
0	-1	0																																																																																
-1	32	-1																																																																																
0	-1	0																																																																																
1	-2	1																																																																																
-2	44	-2																																																																																
1	-2	1																																																																																
1	-2	1																																																																																
-2	32	-2																																																																																
1	-2	1																																																																																
0	-2	0																																																																																
-2	50	-2																																																																																
0	-2	0																																																																																
-1	-1	-1																																																																																
-1	40	-1																																																																																
-1	-1	-1																																																																																

Mask 10	0	1	1	1	0
	1	-4	-2	-4	1
(1/68)	1	-2	80	-2	1
	1	-4	-2	-4	1
	0	1	1	1	0

Appendix 1

// This program includes the implementation and testing
 // all the constructed high-pass Laplacian filters

```
#include <graphics.h>
#include <dos.h>
#include <alloc.h>
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
#include <math.h>
#include <fstream.h>
```

```
FILE *source,*dest;
char far *buf;
char file[20];
unsigned char ch;
unsigned int c1,c2;
int count = 0;
long position = 0;
unsigned size;
int jj;
const int im_size = 200;
```

```
char matrix[im_size][im_size];
int i=im_size,j = 0;
enum flag {T = 1,F = 0} flag = T;
```

```
void Mask123( int i, int j);
void Mask4(int i, int j);
void Mask5(int i, int j);
void Mask67( int i, int j);
void Mask8 (int i,int j);
void Mask9(int i,int j);
void Mask10(int i,int j);
```

```
void add_matrices(char a[][im_size],char b[][im_size],char c[][im_size]);
```

```
//=====
void main()
{
// printf("Enter The file to be shwon: ");
// gets(file);
ofstream matrix3;
matrix3.open("matrix3.dat");
int gdriver = DETECT, gmode, errorcode;
int x = 0, y = 480, color, maxx, maxy,
```

```
maxcolor, seed;
```

```
/* initialize graphics and local variables */
initgraph(&gdriver, &gmode, "c:\\borlandc\\bgi");
```

```
/* read result of initialization */
```

```
errorcode = graphresult();
/* an error occurred */
if (errorcode != grOk)
{
printf("Graphics error: %s\n",
grapherrormsg(errorcode));
printf("Press any key to halt:");
getch();
/* terminate with an error code */
exit(1);
}
if ((source = fopen("c:\\borlandc\\bmp\\ribs.bmp","rt"))
== NULL)
{
closegraph();
fprintf(stderr, "Cannot open input file.\n");
}
```

```
position = ftell(source)+118;
fseek(source, position, SEEK_SET);
while (! feof(source) )
{
```

```
fread(&ch,sizeof(char) , 1, source);
count++;c1 = c2 = 0;
c1 = int(ch) >> 4;
ch = ch<<4;
c2 = int(ch) >> 4;
if(flag == T)
if(i>= 0 && j<= im_size)
{
matrix[i][j] = char(c1);
matrix[i][++j] = char(c2);

if(++j>= im_size)
{j = 0,i = i-1;flag=F;}
}
```

```
// putpixel(x, y, c1);
// putpixel(++x, y, c2);
++x;
if(++x>getmaxx())
{x=0;y--;flag = T;}
} //while
```

```
// copy the original matrix in a file called matrix3.dat
```

```
// -----
for(j = 0;j<= im_size;++j)
for(i = 0;i<= im_size;++i)
```

```

        matrix3<<matrix[i][j];
        matrix3.close();
//-----

//getch();
ifstream matrix2;
matrix2.open("matrix3.dat");//re-open file matrix3.dat in
read mode

cleardevice();
for(j = 0;j<= im_size;++j)
    for(i = 0;i<= im_size;++i)
        putpixel(j+100, i+80, int(matrix[i][j]));

    for(i = 0;i<= im_size;++i)
        for(j = 0;j<= im_size;++j)
            {
                if(i<= 0) matrix[i][j] = 0;
                if(j<= 0 || j>= im_size-1) matrix[i][j] = 0;
                if(i>0 && j>0 && j<im_size-1)
//=====
Mask123(i,j);
Mask4(i,j);
Mask5(i,j);
Mask67(i,j);
Mask8(i,j);
Mask9(i,j);
Mask10(i,j);
//=====
                if(matrix[i][j]<0 ) matrix[i][j]=0;
            }
// x = 100;
// y = 0;
for(j = 0;j<= im_size;++j)
    for(i = 0;i<= im_size;++i)
        putpixel(j+320, i+30, int(matrix[i][j]));
//        putpixel(j+50, i+80, int(matrix[i][j]));
// add the original matrix with the processed one and
save
// it in the matrix (matrix)
for(j = 0;j<= im_size;++j)
    for(i = 0;i<= im_size;++i)

        { matrix2>>ch;
          matrix[i][j] = char(int(ch)-int(matrix[i][j]));
        };
getch();
//cleardevice();
//=====
//=====
// draw the image of the produced matrix
//-----
    for(j = 0;j<= im_size;++j)
    for(i = 0;i<= im_size;++i)
        putpixel(j+320, i+250, int(matrix[i][j]));

```

```

//=====
//=====
getch();
matrix2.close();
fcloseall();
closegraph();
exit(1);
};
//=====
//=====
// Functions Declaration
//=====
//=====
void Mask1 (int i, int j)
{
    matrix[i][j]=
    (-1*matrix[i-1][j-1]-1*matrix[i-1][j]-1*matrix[i-
    1][j+1]
    -1*matrix[i][j-1]+20.0*matrix[i][j]-
    1*matrix[i][j+1]
    -1*matrix[i+1][j-1]-1*matrix[i+1][j]-
    1*matrix[i+1][j+1])/12;
//***** Mask2
// (-1*matrix[i-1][j-1]-1*matrix[i-1][j]-1*matrix[i-
    1][j+1]
// -1*matrix[i][j-1]+32.0*matrix[i][j]-
    1*matrix[i][j+1]
// -1*matrix[i+1][j-1]-1*matrix[i+1][j]-
    1*matrix[i+1][j+1])/24;

//***** Mask3
// (-1*matrix[i-1][j-1]-1*matrix[i-1][j]-1*matrix[i-
    1][j+1]
// -1*matrix[i][j-1]+44.0*matrix[i][j]-
    1*matrix[i][j+1]
// -1*matrix[i+1][j-1]-1*matrix[i+1][j]-
    1*matrix[i+1][j+1])/36;

void Mask4 (int i, int j)
{
    matrix[i][j]=
    4*matrix[i][j]-(1*matrix[i+1][j]-1*matrix[i-1][j]
    -1*matrix[i][j+1]-1*matrix[i][j-1]);
};

void Mask5 (int i, int j)
{
    matrix[i][j]=
    (+0*matrix[i-1][j-1]-1*matrix[i-1][j]+0*matrix[i-
    1][j+1]
    -1*matrix[i][j-1]+32.0*matrix[i][j]-
    1*matrix[i][j+1]
    +0*matrix[i+1][j-1]-
    1*matrix[i+1][j]+0*matrix[i+1][j+1])/28;
};

```

```
//
void Mask6 (int i, int j)
{
    matrix[i][j]=
    (1*matrix[i-1][j-1]-2*matrix[i-1][j]+1*matrix[i-
    1][j+1]
    -2*matrix[i][j-1]+44.0*matrix[i][j]-
    2*matrix[i][j+1]
    +1*matrix[i+1][j-1]-
    2*matrix[i+1][j]+1*matrix[i+1][j+1])/40;

// ***** Mask7
// (1*matrix[i-1][j-1]-2*matrix[i-1][j]+1*matrix[i-
// 1][j+1]
// -2*matrix[i][j-1]+32.0*matrix[i][j]-
// 2*matrix[i][j+1]
// +1*matrix[i+1][j-1]-
// 2*matrix[i+1][j]+1*matrix[i+1][j+1])/28;

};
//=====
void Mask8 (int i, int j)
{
    matrix[i][j]=
    (0*matrix[i-1][j-1]-2*matrix[i-1][j]+0*matrix[i-
    1][j+1]
    -2*matrix[i][j-1]+50.0*matrix[i][j]-
    2*matrix[i][j+1]
    +0*matrix[i+1][j-1]-
    2*matrix[i+1][j]+0*matrix[i+1][j+1])/42;

};
//=====
void Mask9 (int i, int j)
{
    matrix[i][j]=
    (-1*matrix[i-1][j-1]-1*matrix[i-1][j]-1*matrix[i-
    1][j+1]
    -1*matrix[i][j-1]+40.0*matrix[i][j]-
    1*matrix[i][j+1]
    -1*matrix[i+1][j-1]-1*matrix[i+1][j]-
    1*matrix[i+1][j+1])/32;

};
//===== dimension 5x5
void Mask10 (int i, int j)
{
    matrix[i][j]=
    (0*matrix[i-2][j-2]+1*matrix[i-2][j]-
    1+1*matrix[i-2][j]
    +1*matrix[i-2][j+1]+0*matrix[i-
    2][j+2]+1*matrix[i-1][j-2]
    -4*matrix[i-1][j-1]-2*matrix[i-1][j]-4*matrix[i-
    1][j+1]
    +1*matrix[i-1][j+2]+1*matrix[i][j-2]-
    2*matrix[i][j-1]
```

```
+80*matrix[i][j]-
2*matrix[i][j+1]+1*matrix[i][j+2]
+1*matrix[i+1][j-2]-4*matrix[i+1][j-1]-
2*matrix[i+1][j]
-
4*matrix[i+1][j+1]+1*matrix[i+1][j+2]+0*matrix[i+2][j-
2]
+1*matrix[i+2][j]-
1+1*matrix[i+2][j]+1*matrix[i+2][j+1]
+0*matrix[i+2][j+2])/68 ;
};
```

Appendix 2

```
C PROSS77.For ===== PP14 input data ----
PP11 output data
C
C THIS PROGRAM CONVOLVES ANY
OPERATOR OR FILTER
C WITH A MATRIX (I,J) FOR ANY DIMENSION
OF I AND J.
C
C IT WAS TESTED FOR (7,7) MATRIX.
C
C DIMENSION RR(7,7),DD(7,7)
C OPEN(8,FILE='LPT1:')
OPEN(3,FILE='c:\Issa\pp14.d',STATUS='OLD'
)
OPEN(4,FILE='c:\Issa\PP11.txt',STATUS='Old
')
READ(3,*)RR
WRITE(4,5)RR
WRITE(4,7)
c FORMAT(/)
WRITE(*,5)RR
5 FORMAT(7(F7.1))
WRITE(*,7)
7 FORMAT(/)
C
C
N=7
K=7
DO 27 I=1,N
DO 28 J = 1,K
DD(I,J) = (RR(I-1,J-1)-2*RR(I-1,J)+RR(I-1,J+1)
$ -2*RR(I,J-1)+44*RR(I,J)-2*RR(I,J+1)
$ +RR(I+1,J-1)-2*RR(I+1,J)+RR(I+1,J+1))/40
28 CONTINUE
27 CONTINUE
c OPEN(1,FILE='c:\Issa\PP2.O',STATUS='OLD')
WRITE(4,5)DD
WRITE(*,5)DD
C
```

STOP
END

Results

The results (effects) of the application of the developed Laplacian and Laplacian high-pass digital

filters (masks from mask1 to mask10) on digital images, either edge detection, sharpening of high frequency regions (fine details) accentuation, comparing between the effect of different dimensions filters (3×3 and 5×5) and milder high pass effect are indicated and demonstrated on Fig. 2 to 9.

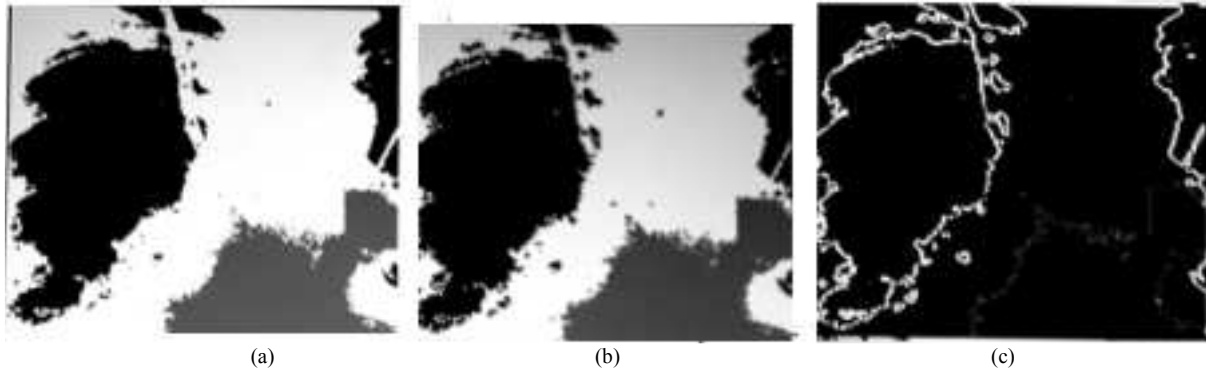


Fig. 2. (a) Original radiographic image; (b) Laplacian high-passed filtered image using mask1; (c) The resulting image (subtracting b from a)

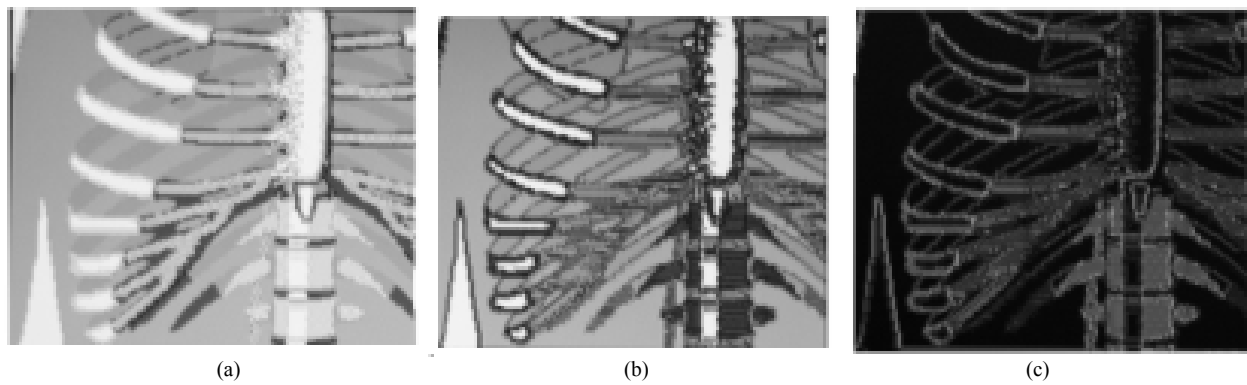


Fig. 3. (a) Original ribs photographic image; (b) Laplacian high-passed filtered image; using mask1 (c) The resulting image (subtracting b from a)

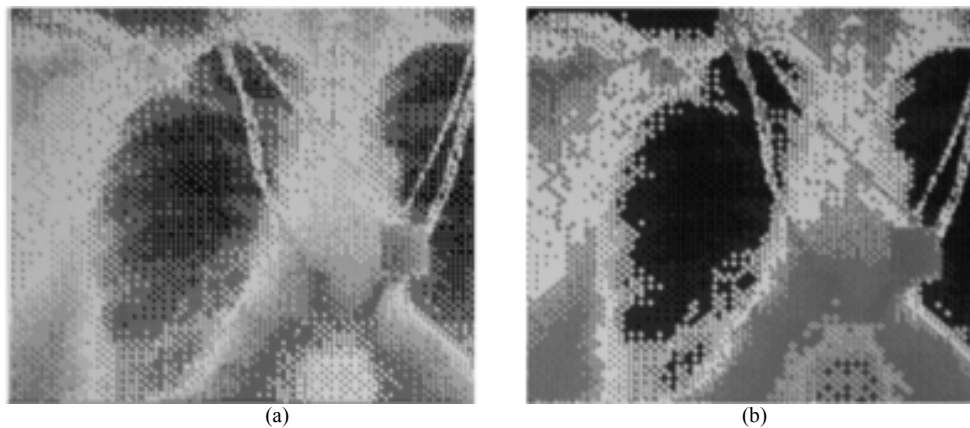


Fig. 4. (a) Original noisy chest x-ray with artefact; (b) The resulting Laplacian high-passed image using mask 5

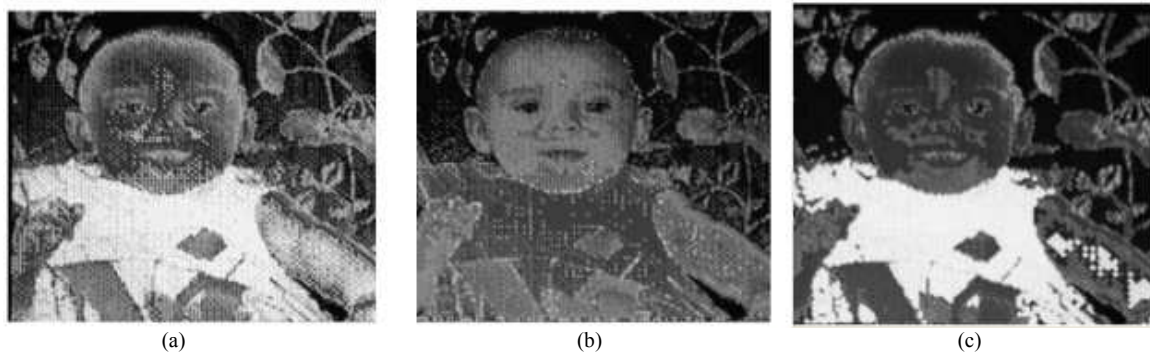


Fig. 5. (a) Original noisy baby photographic image; (b) Laplacian high-passed filtered image using mask 5 (5×5 dimensions); (c) Laplacian high-passed filtered image using mask 10 (3×3 dimensions)

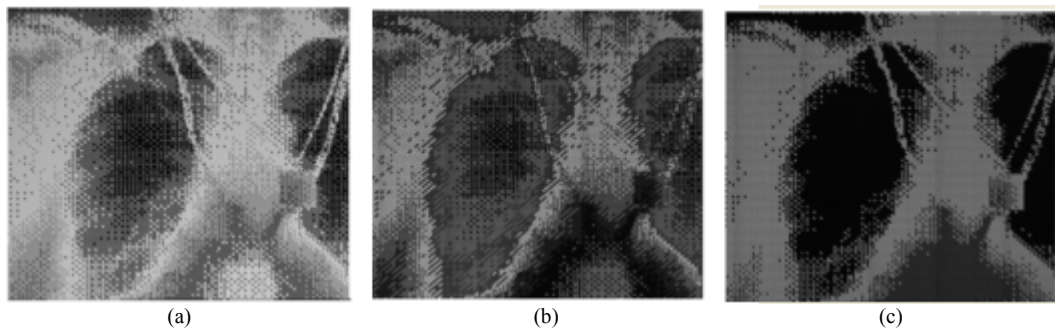


Fig. 6. (a) Original noisy radiographic chest x-ray image; (b) Laplacian high-passed filtered image using mask 5 (5×5 dimensions); (c) Laplacian high-passed filtered image using mask 10 (3×3 dimensions)

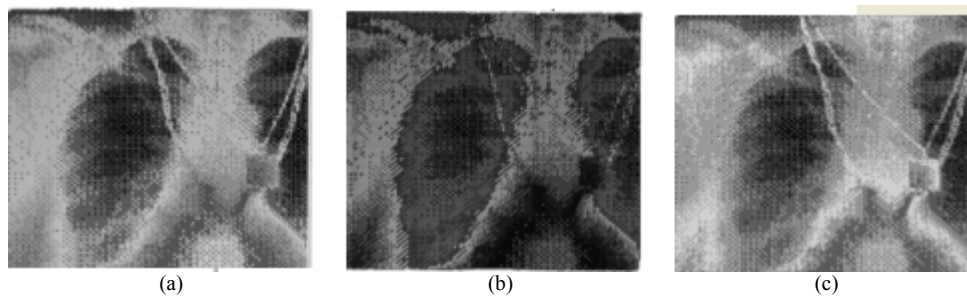


Fig. 7. (a) Original noisy chest radiographic image; (b) Laplacian filtered image using mask 4; (c) The resulting image (subtracting b from a)

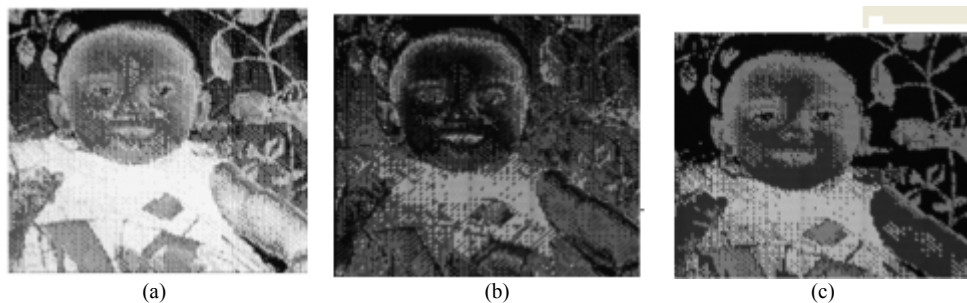


Fig. 8. (a) Original noisy baby photographic image; (b) Laplacian filtered image using mask 4; (c) The resulting image (subtracting b from a)

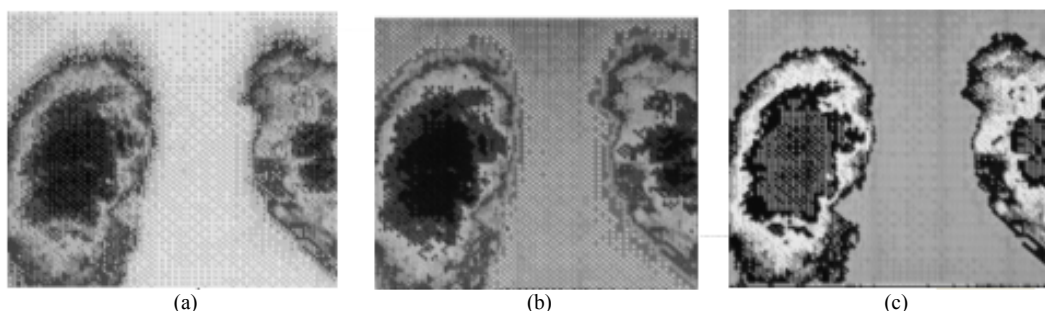


Fig. 9. (a) Original noisy lungs radiographic image; (b) Laplacian high-passed filter image using mask 3; (c) The resulting image (adding a to b), The resulting image shows a milder high-pass effect

Discussion

Computer processing is a prerequisite for display and quantification of all modern medical images. The outcome of clinical reading and diagnosis directly depends on the technique and type of processing performed. The appearance of images can be drastically changed, which may either enhance or degrade the presentation of important information. Complicated mathematical techniques can be applied to image data to provide either accurate or inaccurate quantitative measures of physiological function. To ensure that image quality is enhanced and quantified values are correct, the user must have a competent understanding of the processing techniques (Faber and Folks, 1994).

General understanding of a processing technique includes the knowledge of what an operation really does, how it works and why it is important. The grasp of this information allows the user to adjust strategies to particular situations. Fully understanding the basis of processing procedures helps in solving problems and overcoming processing failures.

In spatial high pass filters the mask coefficients add to 1. A large coefficient generally appears in the center of the mask, surrounded by smaller positive and negative coefficients.

As the size of the filter (mask) gets larger and/or the weight of the center pixel of the kernel gets higher, the sharpening effect becomes more and more (Fig. 5 and 6 for 3×3 and 5×5 dimensions masks).

Figure 4, high-passed Laplacian image, shows the result obtained using mask 5. The detail in this image is unmistakably clearer and sharper than in the original image. Adding the image to the Laplacian restored the overall gray level variations in the image, with the Laplacian increasing the contrast at the locations of gray-level discontinuities. The net result is an image in which small details were enhanced and the back-ground tonality was perfectly preserved. Results like these have made Laplacian-based enhancement a fundamental tool used frequently for sharpening digital images.

High-pass Laplacian can be applied for edge detection, this can be demonstrated by subtracting high-passed Laplacian image from the original image (Fig. 2 and 3). The results obtained from subtracting (using mask 1) from the original image, for edge detection are similar to those that obtained by applying Robert Gradient <http://homepages.inf.ed.ac.uk/rbf/HIPR2/sobel.htm>. In certain cases the results are better especially for images that have large regions of homogeneous intensities.

The fact that the high-pass mask contains a large positive coefficient in the center surrounded by smaller coefficients gives us a clue about its operation. It tells us that the center pixel in the group of input pixels being processed carries a high influence, whereas the surrounding pixels act to oppose it. If the center pixel possesses a brightness that is vastly different from that of its immediate neighbours, the surrounding pixel effect becomes negligible and the output value becomes an accentuated version of the original center pixel. The large difference indicates a sharp transition in gray level, which in turn indicates the presence of high frequency components (Gonzalez and Woods, 2002). We would therefore expect the transition to be accentuated in the output image. On the other hand, if the surrounding pixel brightness are large enough to counteract the center pixel's weight, the ultimate result is based more on an average all pixels involved.

It is interesting to note that if all pixel brightness in a 3×3 group are equal, the result is simply the same value. This is equivalent to the low-pass filter's response over constant regions. This means that the high-pass filter does not attenuate low-frequency spatial components. Rather, it emphasizes high-frequency components while leaving low-frequency components untouched.

Spatial high-pass filtering sharpens an image by accentuating high-spatial-frequency details. The convolution mask weighting coefficients are selected to vary the cutoff point where higher frequencies become accentuated. Additionally, high-pass filtering can render more visible any details that are obscured

by haziness and poor focus in the original image. Further, the resulting high-pass filtered image can be summed with the original image to create milder high-pass filter effects (Fig. 9).

High-pass filters are used to highlight high-frequency details within an image, such as lines, points and edges. Images that appear spatially dull or poorly defined can be sharpened with high-pass filtering techniques.

Accentuating high-frequency image information with high-pass filtering can enhance the characteristics of high-frequency objects of interest. For example, an image of a spatially detailed object--say, a noisy radiographic image and a noisy baby photograph can be enhanced by this way. If the Laplacian image is subtracted from the original image, by using mask 4, the high-pass effect will sharpen the lung and baby features (Fig. 7 and 8).

Conclusion

As the size of the filter (mask) gets larger and/or the weight of the center pixel of the kernel gets higher, the sharpening effect becomes more and more.

Second-order derivatives have a strong response to fine detail, such as thin lines and isolated points.

Sharpening could be accomplished by spatial differentiation, fundamentally, the strength of the response of a derivative operator is proportional to the degree of discontinuity of the image at the point at which the operator is applied. Thus, image differentiation enhances edges and other discontinuities (such as noise) and deemphasizes areas with slowly varying gray-level values (Gonzalez and Woods, 2002).

General understanding of a processing technique includes the knowledge of what an operation really does, how it works and why it is important. The grasp of this information allows the user to choose the suitable filters for particular situations. Fully understanding the basis of processing procedures helps in solving problems and overcoming processing failures.

Acknowledgment

This work was carried out during a sabbatical year (2009/2010) from the University of Jordan.

Funding Information

The author has no support or funding to report.

References

- Brown, D.W., D.L. Kirch, T.W. Reyerson, A.J. Throckmorton and A.L. Lillbourn *et al.*, 1971. Computer processing of scansusing Fourier and other transformations. *J. Nuc. Med.*, 12: 287-91. PMID: 5580849
- Erickson, J.J., 1985. Nuclear medicine computersoftware. *J. Nuc. Med. Technol.*, 13: 140-149.
- Erickson, J.J. and F.D. Rollo, 1983. *Digital Nuclear Medicine*. 1st Edn., J.B. Lippincott, Philadelphia, pp: 240.
- Faber, T.L., 1994. Computer processing methods for nuclear medicine images. *J. Nuc. Med. Technol.*, 22: 145-162.
- Gonzalez, R.C. and P.A. Witz, 1977. *Digital Image Processing*, 1st Edn. Addison-Welsey, London, pp: 503.
- Gonzalez, R.C. and R.E. Woods, 2002. *Digital Image Processing*. 2nd Edn., Prentice Hall, Upper Saddle River, NJ., ISBN-10: 0201180758, pp: 793.
- Gottschalk, A., 1988. *Diagnostic Nuclear Medicine*. 2nd Edn., Williams and Wilkins, Baltimor, MD., pp: 1154.
- Gregory, A.B., 1994. *Digital Image Processing*. 1st Edn., John Wiley and Sons, pp: 452.
- Jain, A.K., 1988. *Fundamentals of Digital Image Processing*. 1st Edn., Prentice-Hall, Englewood Cliffs NJ., ISBN-10: 0133361659, pp: 569.
- Rosenfeld, A. and A.C. Kak, 1982. *Digital Picture Processing*. 2nd Edn., Academic Press, New York, ISBN-10: 0125973012, pp: 435.