Enhancement of Digital Image by C Programming Language

Darun Kesrarat

Faculty of Science and Technology, Assumption University
Bangkok, Thailand

Abstract

Nowadays, there are various Information Technology (IT) devices but one of the most popular IT devices is the digital camera. Basically, the digital camera is used to capture and store images. The digital images are being stored and reproduced in a way that is quite different from the processing steps of the traditional film because the information is being kept in digital format. The digital data in the computer system can be accessed and performed by using a ready-to-use program or a developed program. Thus, C programming language can be written to enhance the quality or to retouch the digital image when the ready-to-use software is not required. Some of the image enhancement operations are discussed in this article. The true color image requires more color representations and processing time comparing to the gray scale one.

Keywords: Digital image, image enhancement, low pass filter, high pass filter, image scaling.

Introduction

The term digital image is defined as a framework that “is stored as numerical values on optical or magnetic media. This term is also used for any print created from such a set of stored numerical values.” (Anon. 2006a)

Within the area of a digital image, pixels are square boxes used to represent the color of the original image. (Anon. 2002; Anon. 2006b). The increase of the number of pixels improves the resolution of the image as shown in Figs. 2 and 3.

From the said figures, it is quite noticeable that the resultant image in framework B of Fig. 3 looks more similar to the target image of Fig. 1 than the one in framework A of Fig. 2. That is why the digital cameras with more pixels usually display better and more beautiful images than the ones with lesser number of pixels.
Moreover, the color representation affects dramatically the quality of the digital images. On the basis of the previous statement that one pixel can display the value of one color only; the binary system of the said value can be represented by either 0 (black color) or 1 (white color) as shown in Fig. 4.

A digital image can be represented in black and white, gray scale, or true color format. The more number of colors to represent, the closer colors to the original image will be. In an eight-bit gray scale image, a pixel can have the color from black color (represented by 0 or ‘\0’ character) to white color (represented by 255 or a space) as shown in Fig. 5.

Sometimes, the quality of the taken image may not be acceptable because of the noise, poor contrast, poor color, and poor resolution. The poor quality image can be improved by the computer program.
characters format. The C programming codes, as shown in Fig. 6, will sequentially read the values start from the top-left pixel to the bottom-right pixel. Then, all the values are stored in the computer memory for further processing.

**Image Enhancement**

The primary objective of the image enhancement is to adjust the digital image so that the resultant image is more suitable than the original image for a specific application.

There are many image enhancement techniques. They can be categorized into 2 general categories. The first category is based on the direct manipulation of pixels in an image, for instance: image negative, low pass filter (smoothing), and high pass filter (sharpening). Second category is based on the position manipulation of pixels of an image, for instance image scaling.

In the first category, the image processing function in a spatial domain can be expressed as:

\[ g(x, y) = T[f(x, y)] \]  

where \( T \) is the transformation function, \( f(x, y) \) is the pixel value of input image, and \( g(x, y) \) is the pixel value of the processed image.

**Image Negative**

Image negative is one of the image enhancement techniques that reverses the pixel value from black to white and vice versa. The intensity of the output image decrease when the intensity of the input increases (Anon. 2004). The pixel values of the output image can be calculated as follows:

\[ g(x, y) = |(f(x, y) - 255)| \]

Based on C programming language, the codes used to read and transform the pixel values from the input image to the output image are shown in Fig. 7. The resultant image of the said programming codes is shown in Fig. 8.

```c
fp = fopen("Result.raw","wb");
for (i = Height-1; i >= 0; i--)
    for (j = 0; j < Width; j++)
    {
        pic1[i][j] = abs(pic1[i][j]-255);
        fprintf(fp,"%c",pic1[i][j]);
    }
fclose(fp);
```

**Low Pass Filter (Smoothing)**

Low pass filter, also known as "smoothing", is used to remove the high spatial frequency noise from a digital image. Noise is often included during the analog-to-digital conversion process. It is a side-effect of light energy in the physical-to-electronic-pattern conversion. (Clifford Watson, 1994) Moreover, low pass filter can blur a digital image as to bridge its small gaps in line or curve sections.

A low pass filter usually uses a moving-window-frame to operate one pixel of the image at a time by changing the pixel value regarding to some function of a "local" pixel region covered by the said window frame. This operator will gradually move over the image from left-to-right and top-to-bottom in order to change all the pixel values of the image.
Based on the coefficient of moving-window-frame in Fig. 9, it will replace each pixel value in the input image \( f[x,y] \) by an average weight of the neighbor pixels in that region. For instance, let \( k \) denotes the area of moving-window-frame, \( f[x+p,y+q] \) denotes the weighted sum where \( p = -k \) to \( k \), and \( q = -k \) to \( k \) for some positive \( k \), the weight can be calculated as follows (assuming that \( k=1 \)):

\[
g(x,y) = \frac{f(x,y)+f(x-1,y)+f(x,y-1)+f(x-1,y-1)+f(x+1,y)+f(x,y+1)+f(x+1,y+1)+f(x+1,y+1)+f(x-1,y+1)}{9} \tag{3}
\]

The weight must be a non-negative value when \( p \) and \( q \) are equal to zero. If all the calculated weights are equal then this is a mean filter. As shown in Fig. 10, low pass filter can remove unwanted noise of the original image.

**High Pass Filter (Sharpening)**

The primary objective in image sharpening is to highlight the fine details or to enhance the blurred details of the image due to the noise and motion effects. Enhancing the high-frequency components by the image sharpening technique will have the spatial filter shape with a high positive component at the center (Robyn Owens, 1997). The simplest form of high pass filter should have positive coefficients near its center and negative coefficients in the outer region as shown in Fig. 11.

\[
g(x,y) = \frac{((8*f(x,y))-f(x-1,y)-f(x,y-1)-f(x-1,y-1)-f(x+1,y)-f(x,y+1)-f(x+1,y+1)-f(x+1,y+1)-f(x-1,y+1))}{9} \tag{4}
\]

Similar to the low pass filter, high pass filter uses the moving-window-frame with difference coefficients to change the pixel values. The weight can be calculated as follows (assuming that \( k=1 \)):

\[
g(x,y) = \frac{((8*f(x,y))-f(x-1,y)-f(x,y-1)-f(x-1,y-1)-f(x+1,y)-f(x,y+1)-f(x+1,y+1)-f(x+1,y+1)-f(x-1,y+1))}{9}
\]

**Fig. 10. Resultant of the low pass filter image**

**Fig. 11. Coefficient of a moving-window-frame for high pass filter.**

**Fig. 12. Resultant of the high pass filter image**
Image Scaling

Scaling of an image usually means the alteration in the amount of pixels in a digital image. A digital image can freely be scaled up or down in vertical and horizontal directions. Some of the pixels must be discarded when the image is scaled down, and some pixels must be added when it is scaled up. In scaling up, the values of the new pixels must be calculated by referring to their neighbors’ pixel values. Image scaling can be done by the sequence of coordinate mapping and re-sampling operations. Coordinate mapping operation is used to define the new boundary of the image whereas re-sampling operation is used to define the new pixel values.

As shown in Fig. 13, when the image is enlarged 10%, the new amount of pixels of that image must be recalculated. The coordinate mapping allows us to relate the position of the pixels in the resulting image to their positions in the original image. As shown in Fig. 13, the output image has the same size of the original image but with more number of pixels.

When the new amount of pixels is defined in the output image, the new values for the additional pixels \( y \) – interpolated pixels – must be recalculated by referring to the values of their neighbor pixels as follows:

\[
y = j \times \frac{\text{old size}}{\text{new size}} \tag{5}
\]

Where \( j \) is the position of the pixel in new image size. After the said calculation, if the decimal value of \( y \) is less than 0.5 then round it down, otherwise round it up to the next value and store that value in the corresponding pixel. As shown in Fig. 14, pixel number 5 and 6 in the output image (with 11 pixels) are sharing the same color value of the original image (with 10 pixels). C programming code used for the said image scaling is shown in Fig. 15.

```
position = j * ((double)old_width/new_width);
diff = fmod(position,1);
real = (int)position;
if((j == new_width-1) || (j==0))
{
    if(j==new_width-1)
    {
        pic2[i][j] = pic1[i][old_width-1];
    }
    else if(j==0)
    {
        pic2[i][j] = pic1[i][0];
    }
}
else if (diff < 0.5 )
{
    pic2[i][j] = pic1[i][real];
}
else if (diff >= 0.5 )
{
    real++;
    pic2[i][j] = pic1[i][real];
}
```

Fig. 15. C programming code for image scaling

True Color Image

The image enhancement operations on the gray scale image are discussed in the previous section. A true color image, 24 bits
image say, has 16,777,216 colors in the image. The said colors are separated into 3 primary groups – red, green, and blue – each of them requires 8 bits to store the color values. The combination of the primary colors produces other colors as shown in Fig. 16 and Table 1.

![Fig. 16. RGB Color Model.](image)

To perform the image enhancement operations as mentioned, C programming codes must be applied to each group separately. Therefore, the processing time of the true color image will be approximately 3 times longer than the gray scale one.

Table 1. Some 24-bit color representations.

<table>
<thead>
<tr>
<th>Red</th>
<th>Green</th>
<th>Blue</th>
<th>Resultant Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Black</td>
</tr>
<tr>
<td>255</td>
<td>255</td>
<td>255</td>
<td>White</td>
</tr>
<tr>
<td>255</td>
<td>0</td>
<td>0</td>
<td>Red</td>
</tr>
<tr>
<td>0</td>
<td>255</td>
<td>0</td>
<td>Green</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>255</td>
<td>Blue</td>
</tr>
<tr>
<td>255</td>
<td>255</td>
<td>0</td>
<td>Yellow</td>
</tr>
<tr>
<td>0</td>
<td>255</td>
<td>255</td>
<td>Cyan</td>
</tr>
<tr>
<td>255</td>
<td>0</td>
<td>255</td>
<td>Magenta</td>
</tr>
</tbody>
</table>

Conclusion

The digital camera facilitates people to take the picture and transfer it to the computer as the digital image for later processing. Sometimes, the quality of the taken images must be improved for the reproduction. People may need to perform the image enhancement operations by using a ready-to-use program, but some of the said software may not be flexible enough to support the detailed operation. Some digital image file format uses the lossy compression to remove the redundancy of the color values and to maintain the small file size. In RAW image format, it uses lossless compression to store the color value without any data lost of the original image. C programming language can help people who really want to perform the detailed-operations on the digital image that is saved with RAW format. The image enhancement operations mentioned in this article are the primary ones that can improve the quality of the digital image.

References


