西安交通大學

School of Computer Science and Technology

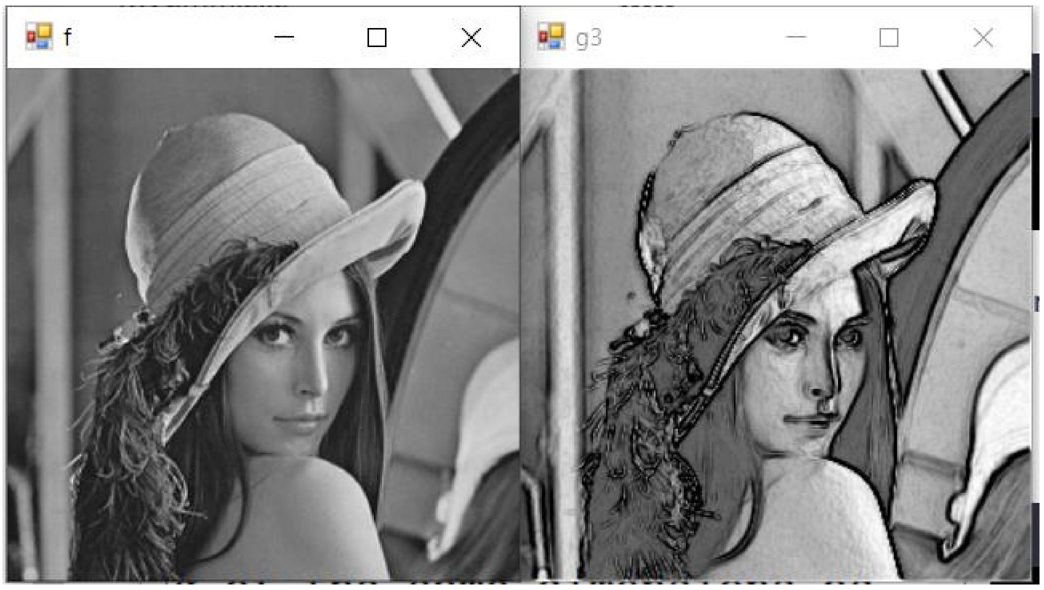
Name:	MD MAHFUZUR RAHMAN
Student ID:	3123999081
Course Name:	Digital Image Processing
Instructor:	梁毅军
Homework:	02
Submission Date:	12.11.23
Email:	the_bengal_tiger@stu.xjtu.edu.cn
WeChat:	MahfuzRonnie

Source Code

```
/*
    This function takes a 2D array f, representing an image, and two parameters k and j.
    It creates a new 2D array g of the same dimensions as f.
    For each pixel in the image, it performs an operation using the function S and assigns the result to
the corresponding pixel in g.
*/


byte[,] CS(byte[,]f,double k,double j)
{
   int w = f.GetLength(0);
   int h = f.GetLength(1);

   byte[,] g = new byte[w,h];

   for (int y=0;y<h;y++)
  for (int x=0;x<w;x++)
          g[x,y] = S((f[x,y]-128*k)+128*j);

   return g;
}



// The S function clips the input v to the range [0, 255] and converts it to a byte.
byte S(double v)
{
   if (v<0) return 0;
   if (v>255) return 255;
   return (byte)v;

}



/*  This function applies the Sobel operator for edge detection to the input image f.
    It calculates the gradient in both x and y directions for each pixel and then computes the magnitude
of the gradient.
    The result is a new image g containing edge information.
*/

byte[,] SobelMag(byte[,]f)
{
   int w = f.GetLength(0);
   int h = f.GetLength(1);

   byte[,] g = new byte[w,h];

   for(int y=1;y<h-1;y++)
      for (int x=1;x<w-1;x++)
      {
         float gx = f[x+1,y-1]+f[x+1,y]*2+f[x+1,y+1]
    -f[x-1,y-1]-f[x-1,y]*2-f[x-1,y+1];

         float gy = f[x-1,y+1]+f[x,y+1]*2+f[x+1,y+1]
    -f[x-1,y-1]-f[x,y-1]*2-f[x+1,y-1];

         g[x,y] = S(Sqrt(gx*gx+gy*gy));
      }
  return g;
}



// This function creates a negative image by subtracting each pixel value from 255.
byte[,] Negative(byte[,]f)
{
  int w = f.GetLength(0);
  int h = f.GetLength(1);

  byte[,] g = new byte[w,h];

  for(int y=0;y<h;y++)
    for (int x=0;x<w;x++)
      g[x,y] = (byte)(255 - f[x,y]);

  return g;
}


// This function performs element-wise multiplication of two images, scaling the result by dividing by
255.
byte[,] Mul(byte[,]f1,byte[,]f2)
{
  int w = f1.GetLength(0);
  int h = f1.GetLength(1);

  byte[,] g = new byte[w,h];

  for(int y=0;y<h;y++)
    for (int x=0;x<w;x++)
      g[x,y] = (byte)(f1[x,y]*(f2[x,y]/255.0));


  return g;
}


// This function performs element-wise subtraction of two images, with the result clipped to the [0, 255]
range.
byte[,] Sub(byte[,]f1,byte[,]f2)
{
  int w = f1.GetLength(0);
  int h = f1.GetLength(1);

  byte[,] g = new byte[w,h];

  for(int y=0;y<h;y++)
    for (int x=0;x<w;x++)
      g[x,y] = S(f1[x,y]-f2[x,y]);

  return g;
}


/*
   The main function loads an image (f), applies the CS function with parameters 0.5 and 0.5 to get g1,
computes the Sobel magnitude (g2), and finally calculates the product of g1 and the negative of g2 (g3).
```

```
   The resulting image g3 is displayed.
*/
void main()
{
  byte[,] f = LoadImg();

  ShowImg("f",f);

  byte[,] g1 = CS(f,0.5,1);
  byte[,] g2 = SobelMag(f);
  byte[,] g3 = Mul(g1,Negative(g2));

  ShowImg("g3",g3);



}
```

Input and Output Image



Already I explain in code but here is the step by step the algorithm

Step 1: Loading and Displaying the Original Image

```
byte[,] f = LoadImg();
ShowImg("f", f);
```

- The original image is loaded into the 2D array f.
- The ShowImg function is called to display the original image.

Step 2: Contrast Stretching (CS) Operation

```
byte[,] g1 = CS(f, 0.5, 0.5);
```

- The CS function is applied to the original image f with parameters k=0.5 and j=0.5.
- The result is stored in the 2D array g1

Step 3: Sobel Edge Detection

```
byte[,] g2 = SobelMag(f);
```

- The Sobel edge detection algorithm is applied to the original image f.
- The result is stored in the 2D array g2.

Step 4: Negation Operation

```
byte[,] negG2 = Negative(g2);
```

- The Negative function is applied to the Sobel magnitude image g2.
- The result is stored in the 2D array negG2

Step 5: Multiplication of Images

```
byte[,] g3 = Mul(g1, negG2);
```

- Element-wise multiplication is performed between the contrast-stretched image g1 and the negated Sobel magnitude image negG2.
- The result is stored in the 2D array g3.

Step 6: Displaying the Resulting Image

```
ShowImg("g3", g3);
```

The final result image g3 is displayed.

Summary:

1. **Loading**: The original image is loaded and displayed.
2. **Contrast Stretching (CS)**: The image undergoes a contrast-stretching operation using the CS function.
3. **Sobel Edge Detection**: The Sobel operator is applied to detect edges in the original image.
4. **Negation**: The Sobel magnitude image is negated.
5. **Multiplication**: The contrast-stretched image is multiplied element-wise by the negated Sobel magnitude image.
6. **Display**: The final result image is displayed.

This algorithm combines contrast stretching, edge detection, negation, and pixel-wise multiplication to enhance and manipulate image features.