194588 CHUKWUMA FUMNANYA DIVINE.

SOBEL OPERATOR.

The Sobel operator sometimes called a sobel Filter is used in image processing and also the physical computer vision, it is said to be within edge detection algorithms where it creates an image from the original emphasizing only the edges in the image and those edges are white lines. The idea behind the sobel operator is based on convolution where as you convolve the original image with a small integer-valued filter (3x3 Matrixes) in both axis (x,y). Below is the formula and computations used;

$$\mathbf{G}_x = \begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

$$\mathbf{G} = \sqrt{{\mathbf{G}_x}^2 + {\mathbf{G}_y}^2}$$

$$\mathbf{\Theta} = \operatorname{atan2}(\mathbf{G}_y, \mathbf{G}_x)$$

I shall paste my code below and some sample image pairs with their originals to show case this operation.

Code for the sobel operation:

```
#include "image.h"
#include <math.h>
#include <stdio.h>


int main()
{
    GrayscaleImage inp;
    inp.Load("images/barbara.png");
    GrayscaleImage out((inp.GetWidth() - 3)+1, (inp.GetHeight() - 3)+1);
    int gradient_x[3][3] = { {-1,0,1},{-2,0,2},{-1,0,1} };
    int gradient_y[3][3] = { {1,2,1},{0,0,0},{-1,-2,-1} };
    for (int y = 0; y < out.GetHeight(); y++)
    {
        for (int x = 0; x < out.GetWidth(); x++)
        {
            int total_x = 0;
            int total_y = 0;
            int a = x;
            int b = y;
            for (int i= 0; i < 3; i++)
            {
                for (int j = 0; j < 3; j++)
                {
                    total_x += inp(a, b) * gradient_x[i][j];
                    a++;
                }
                a = x;
                b++;
            }
            int c = x;
            int d = y;
            for (int i = 0; i < 3; i++)
            {
                for (int j = 0; j < 3; j++)
                {
                    total_y += inp(c, d) * gradient_y[i][j];
                    c++;
                }
                c = x;
                d++;
            }
            int result = round(sqrt(total_x* total_x + total_y* total_y));
            out(x, y) = result;
            if (out(x, y) < 0)
                out(x, y) = 0;
            if (out(x, y) > 255)
                out(x, y) = 255;

        }

    }
    out.Save("images/pot9.png");
}
```

Image Samples.

Image Set 1:

Original Image:

After Applying Sobel operation on the image:

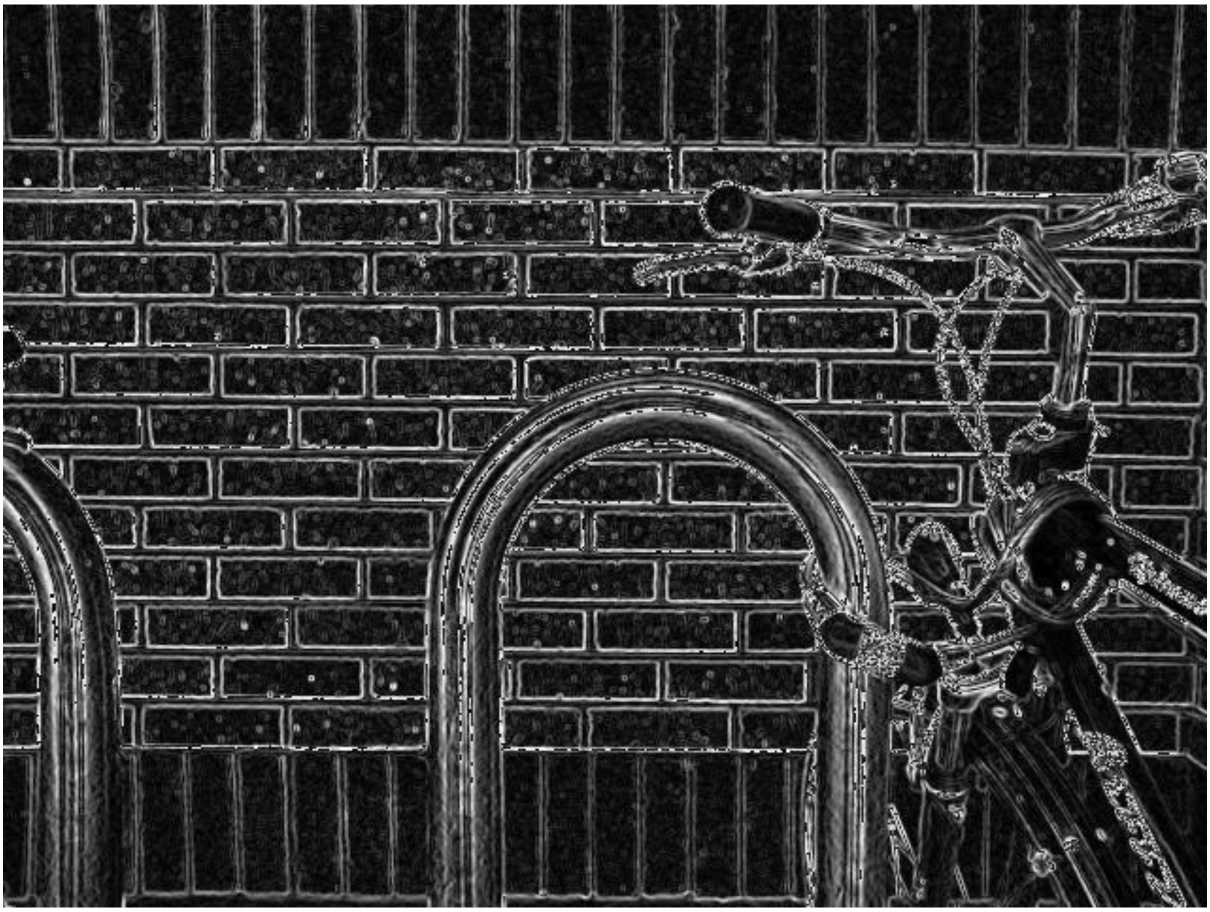
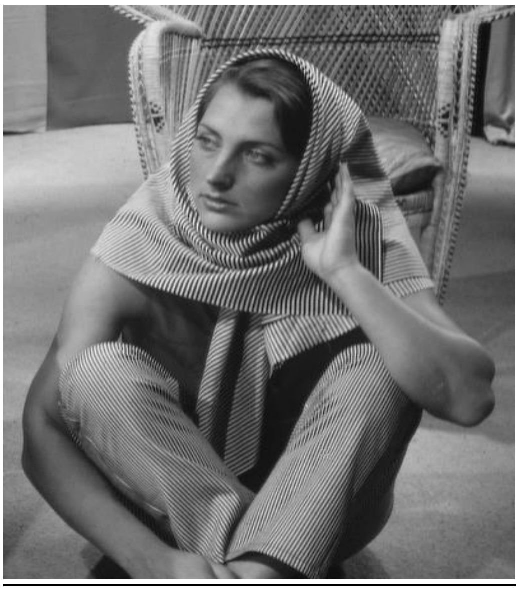


Image Set 2:

Original Image:



After Applying Sobel Operation on the image:

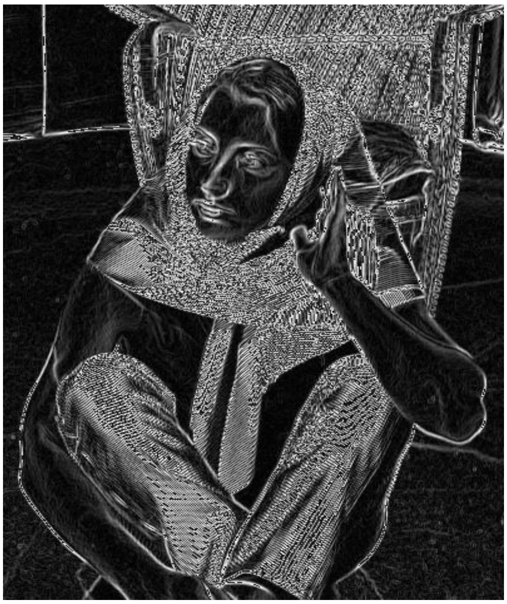
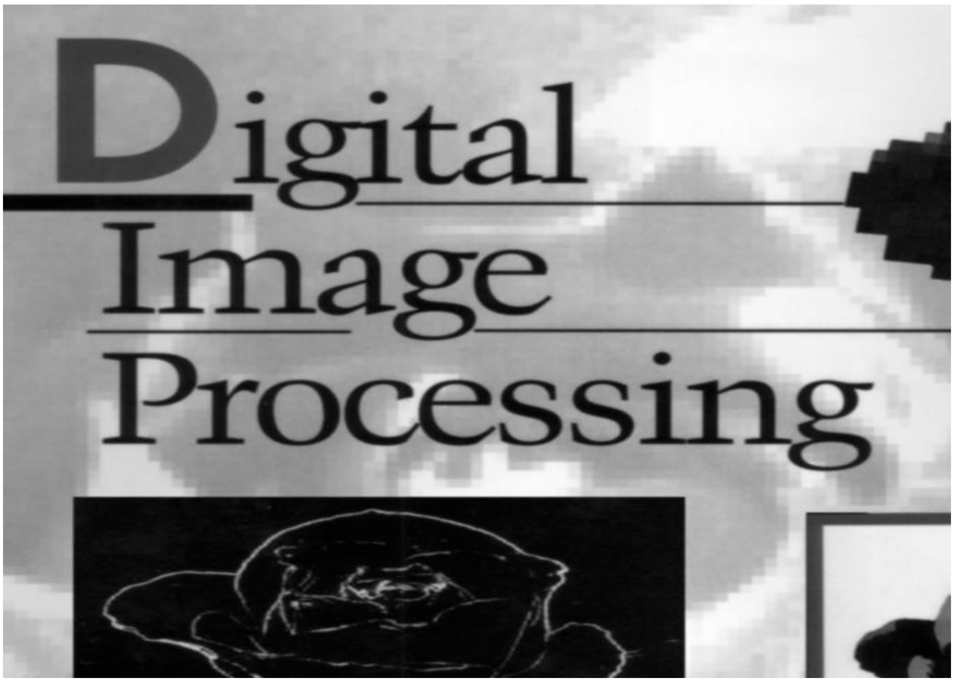


Image Set 3:

Original image:



After applying Sobel Operation on the image:

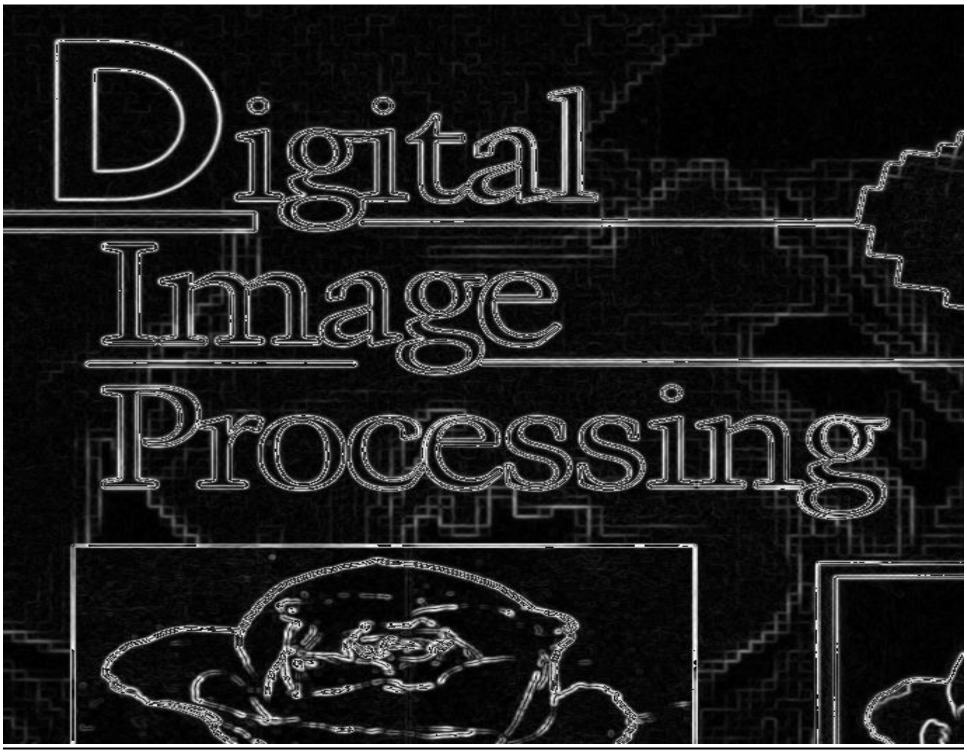
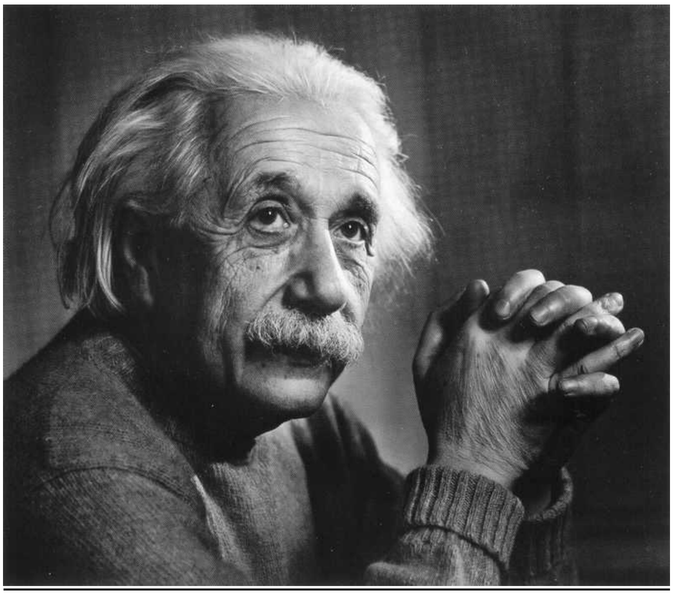
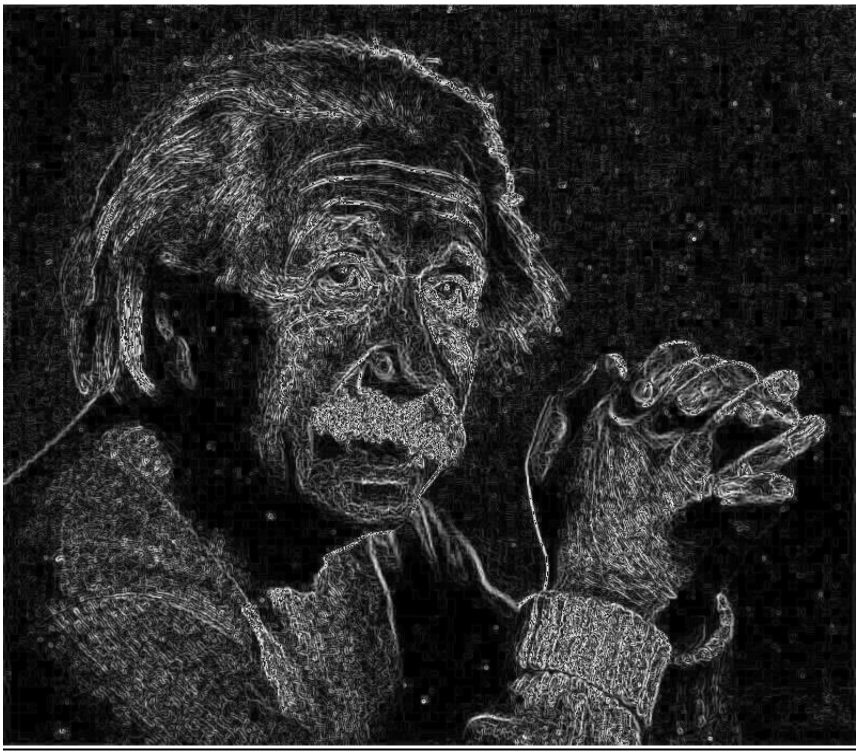


Image Set 4:

Original Image:



After Applying Sobel Operation on the image:



From all our image pairs we can see the vivid effect of our Sobel operation on the images.