CS485/685 Computer Vision

Spring 2010 – Dr. George Bebis

Homework 2 – Solutions

1. Consider the subimage shown below. Find the gradient magnitude and gradient direction at the center entry using **(i)** the Prewitt operator, **(ii)** the Sobel operator.

4	1	6	1	3
3	2	7	7	2
2	5	7	3	7
1	4	7	1	3
0	1	6	4	4

(i) Prewitt

$$M_x = \begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix}$$

$$M_y = \begin{matrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{matrix}$$

(ii) Sobel

$$M_x = \begin{matrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{matrix}$$

$$M_y = \begin{matrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{matrix}$$

Using Prewitt: ①

$$f_x = 2(-1) + 7(0) + 7(1) + 5(-1) + 7(0) + 3(1) + 4(-1) + 7(0) + 1(1) =$$
$$= 0$$

$$f_y = 2(-1) + 7(-1) + 7(-1) + 5(0) + 7(0) + 3(0) + 4(1) + 7(1) + 1(1) =$$
$$= -4$$

$$M = |f_x| + |f_y| = 4$$

$$D = atan^{-1}(f_y/f_x) = -90^o$$

Using Sobel:

$$f_x = 2(-1) + 7(0) + 7(1) + 5(-2) + 7(0) + 3(2) + 4(-1) + 7(0) + 1(1) =$$
$$= -2$$

$$f_y = 2(-1) + 7(-2) + 7(-1) + 5(0) + 7(0) + 3(0) + 4(1) + 7(2) + 1(1) =$$
$$= -4$$

$$M = |f_x| + |f_y| = 6$$

$$D = atan^{-1}(f_y/f_x) = -116.6^o$$

2. Problem 2 (OpenCV book, page 190)

a,b) The result is the same using the 2D mask or the 1-D masks.

2

Let us suppose an $N \times M$ image.

At each pixel location, we need to apply a mask of size $n \times m$. This would require $n \times m$ multiplications. The total would be $n \times m \times N \times M = n^2 N \times M$ $(n = m)$

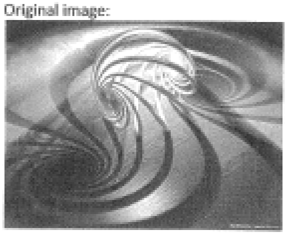
If we now assume 1D masks of size n, it would require n multiplications at each pixel. Since we have to convolve twice (once for the rows, one), we would need $2n$ multiplications The total would be $2n \times N \times M$.

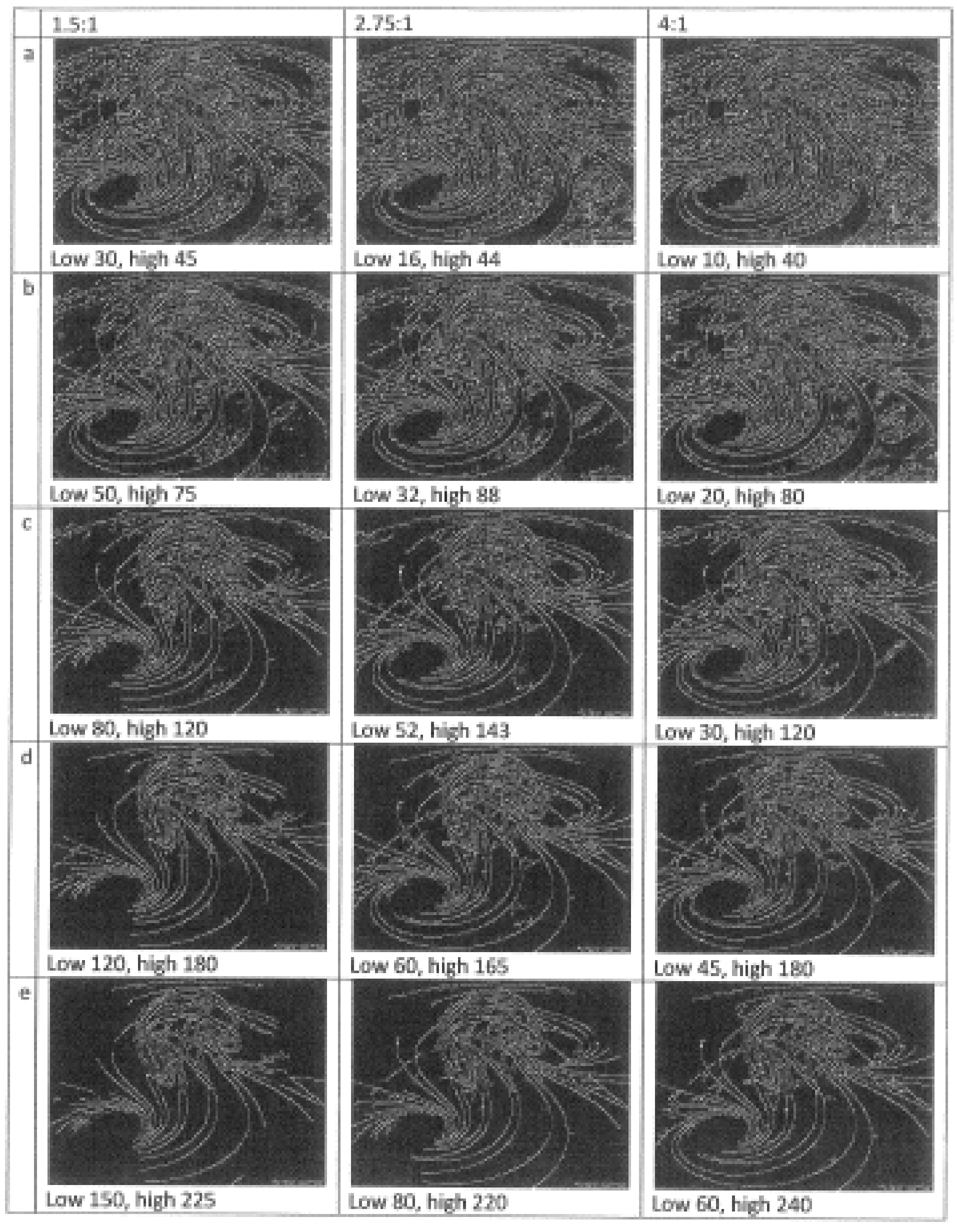
Using big-O notation
2D case: $O(n^2 NM)$
1D case: $O(nNM)$ — cheaper.

3. Problem 7 (OpenCV book, page 191)

The larger the difference between the low and high thresholds, the more pixels falls into the uncertainty interval and they might become edge points. The smaller the difference between the low and high thresholds, the more edges will be filled in. This is obvious by observing the results below. Decreasing the low threshold will increase the number of edge points (also, it will allow more noise to pass) while increasing the high threshold will decrease the number or edge points (also, some strong edges will be eliminated).

Original image:





4. Generate the mask for $255 \times \nabla^2 G(x, y)$, for $\sigma=1$. Truncate all the mask values to the nearest integer.

```
//-------------------------------------------------------------
// This program calculate the Laplacian-Gaussian mask
//
//-------------------------------------------------------------

#include <iostream.h>
#include <iomanip.h>
#include <math.h>

int main ()
{
    double sigma, elementValue;
    int maskSize,i,j;

    cout << "Enter the variance of the Gaussian mask:";
    cin >> sigma;

    //Determine the mask size

    maskSize= int (5* sigma +0.5);
    if (maskSize % 2 == 0) maskSize ++;

    cout << endl << "The optimal mask size is: " << maskSize << endl << endl;

    cout << "The elements in the Laplacian-Gaussian mask are:" << endl << endl;

    //Calculate the melements in the mask

    for (i=-maskSize/2 ; i<=maskSize/2; i++)
    {
        for (j=maskSize/2; j>=-maskSize/2; j--)
        {
            elementValue = int (255*(i*i+j*j-2*pow(sigma,2))*exp(-(i*i+j*j)/2.0/sigma/sigma)/p
ow(sigma,4));
            cout << setw(7) << elementValue;
        }
        cout << endl;
    }

    return 0;
}

/* Test case:

Enter the variance of the Gaussian mask:1

The optimal mask size is: 5

The elements in the Laplacian-Gaussian mask are:

     28     62     69     62     28
     62      0   -154      0     62
     69   -154   -510   -154     69
     62      0   -154      0     62
     28     62     69     62     28
Press any key to continue

*/
```

Graduate Students Only:

Prove the following properties of the Gaussian function $G(x) = e^{\frac{-x^2}{2\sigma^2}}$

(a) Symmetry: $G(x) = G(-x)$
(b) Scaling: $G^{\sigma}(x) * G^{\sigma}(x) = G^{\sqrt{2}\sigma}(x)$

Using property (b), propose a more efficient way to compute $(f(x) * G^{\sigma}(x)) * G^{\sigma}(x)$. Justify your answer by comparing the number of calculations (i.e., multiplications/additions) required.

) $G(x) = e^{-x^2/2\sigma^2}$ 4

(a) $G(-x) = e^{-(-x)^2/2\sigma^2} = e^{-x^2/2\sigma^2} = G(x)$

(b) $G^{\sigma}(x) * G^{\sigma}(x) = G^{\sqrt{2}\sigma}(x)$

$G^{\sigma}(x) * G^{\sigma}(x) = \int e^{-t^2/2\sigma^2} e^{-(x-t)^2/2\sigma^2} dt$

$= \int e^{-(x^2 - 2xt + 2t^2)/2\sigma^2} dt =$

$= \int e^{-(x^2/2 - 2xt + 2t^2 + x^2/2)/2\sigma^2} dt$

$= e^{-x^2/4\sigma^2} \int e^{-(\sqrt{2}t - x)^2/2\sigma^2} dt$

substitute $\sqrt{2}t - x = s \Rightarrow ds = \sqrt{2}dt$

$= \sqrt{2} e^{-x^2/4\sigma^2} \int e^{-s^2/2\sigma^2} ds =$

$2\sigma\sqrt{\pi} e^{-x^2/2(\sqrt{2}\sigma)^2}$

We want to compute $\underbrace{(f(x) * G^{\sigma}(x))}_{Z(x)} * G^{\sigma}(x)$ (5)

or: $Z(x) = f(x) * G^{\sigma}(x) \longrightarrow 5\sigma$ multiplications/pixel

$W(x) = Z(x) * G^{\sigma}(x) \longrightarrow 5\sigma$ —— " ——

Total: $5\sigma + 5\sigma = 2 \times 5\sigma$ multiplications/pixel

Instead of using 2 steps, we can use one step: $W(x) = f(x) * G^{\sqrt{2}\sigma}(x)$

Total: $5\sqrt{2}\sigma$ multiplications/pixel

(Cheaper!!)

mask size = 5σ