Assignment 2 - Contrasting different methods of edge detection

Taylor Foxhall
tfoxhal1@binghamton.edu

October 13, 2015

1 Questions

Problem 1. Lightest to heaviest smoothness: M4, M3, M2, M1
You can easily see why if you compare the ratios of the center weight and the sum of the neighbors' weights. M4 will give you the same image back, so no smoothing will occur. M3 has a 4:4 (or 1:1) ratio. Then M2 has a 4:12 (or 1:3) ratio, followed by M1 which has a 1:8 ratio, there will be the most amount of smoothing.

Problem 2. Both the first and second order derivatives create a "ramp," more clearly showing us where the edges are. However, the second order derivative creates a thicker padding and emphasizes a low point and a high point with a zero spacing them out. The first derivative does not emphasize both a low and high point, only one of the two. Therefore, the second order derivatives create a better enhancement filter, and they tend to be easier to implement too.

Problem 3. A single pass matrix S for applying edge detection is using M is:

$$S = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

Problem 4. Blurring the image first is better because if you apply the blurring afterwards you're going to lose information about the edge. Not only that, but after the original image is blurred, generally parts of the image that are similar will remain nearly unaffected, but regions of an image that have differences (where an edge is) will be smoothed out, so edge detection will be easier to calculate.

Problem 5.

$$M1 = \begin{bmatrix} 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 48 & 64 & 64 & 4 & 4 & 4 \\ 4 & 4 & 64 & 64 & 64 & 64 & 4 & 4 \\ 4 & 4 & 64 & 64 & 64 & 64 & 4 & 4 \\ 4 & 4 & 56 & 64 & 64 & 23 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \end{bmatrix}$$

$$M2 = \begin{bmatrix} 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 64 & 64 & 64 & 64 & 4 & 4 \\ 4 & 4 & 64 & 64 & 64 & 64 & 4 & 4 \\ 4 & 4 & 64 & 64 & 64 & 64 & 4 & 4 \\ 4 & 4 & 64 & 64 & 64 & 64 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \\ 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 \end{bmatrix}$$

Clearly, $M2$ is a better filter for this image because there is practically no noise left in the filtered image, where $M1$ still has some noise along the edges. The contrast between the center of the image in $M2$ is more clear than the contrast between the center of the image in $M1$.

2 Programming

2.1 Unsharp Masking

A Laplacian mask M was used to generate the unsharp image, which was then subtracted from the original image. I set M to be:

$$M = \begin{bmatrix} -1 & -1 & -1 \\ -1 & 9 & -1 \\ -1 & -1 & -1 \end{bmatrix}$$

which was applied to every pixel that had bordering pixels. This method was applied to f1 and f2:

Which generated e1 and e1:

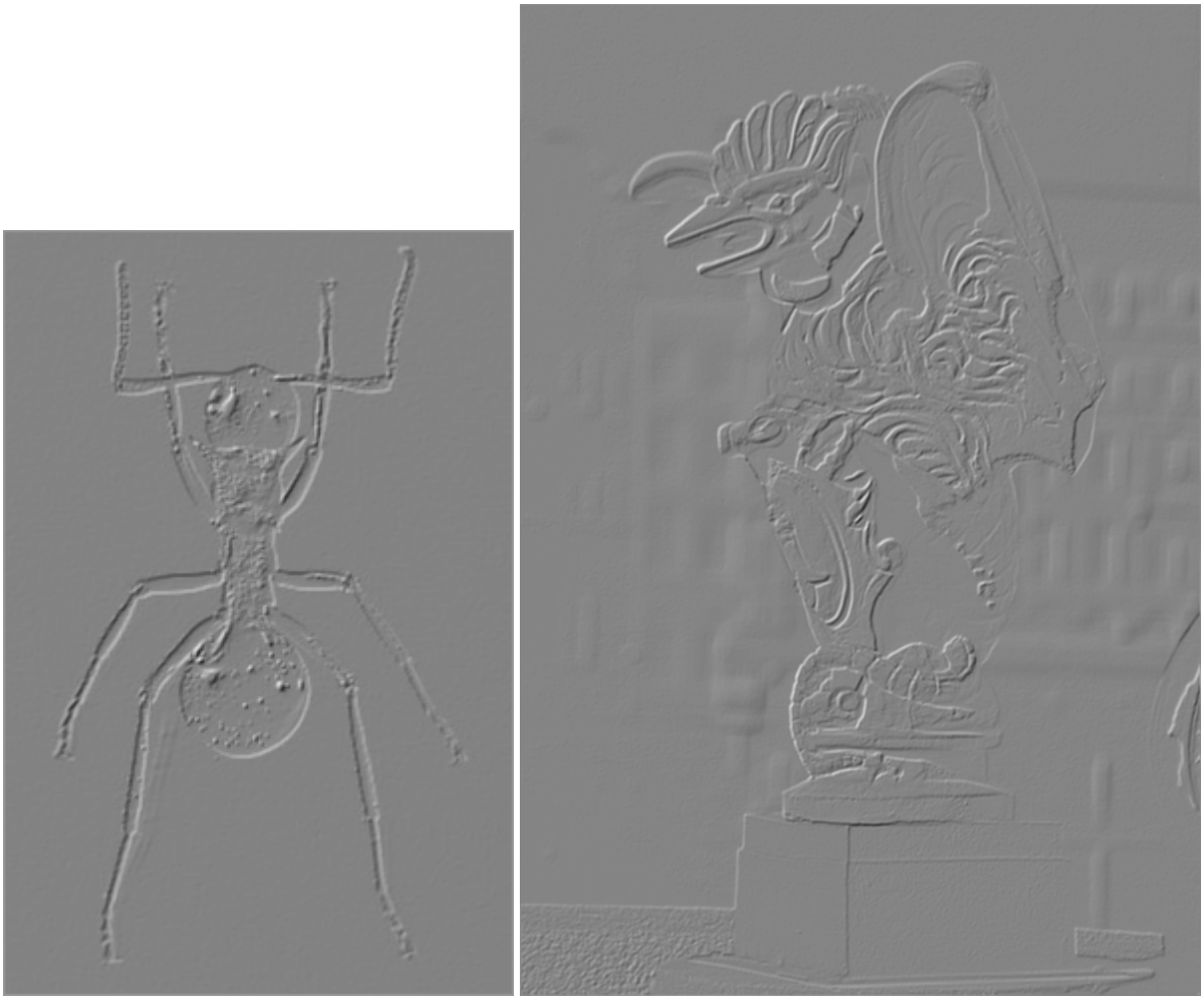
2.2 Sobel Operation

The Sobel operators are defined as:

$$S_x = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & -0 \\ 1 & 2 & 1 \end{bmatrix}$$

$$S_y = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

When applied to a base image like in unsharp masking, each generate a gradient image in the x and y direction respectively. When averaged together, a better gradient image is generated. Es1 and es2 are generated from the Sobel applied to f1 and f2:



2.3 Laplacian of Gaussian

We can also use the Laplacian of Gaussian for edge detection. I generated two masks, one of size 7x7 and with $\sigma = 1.4$ and another of size 11x11 with $\sigma = 5$. The generated masks are approximately:

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 1 & 2 & 2 & 1 & 2 & 2 & 1 \\ 1 & 2 & 0 & -1 & 0 & 2 & 1 \\ 1 & 1 & -1 & -5 & -1 & 1 & 1 \\ 1 & 2 & 0 & -1 & 0 & 2 & 1 \\ 1 & 2 & 2 & 1 & 2 & 2 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$M_2 = \begin{bmatrix} 4 & 3 & 2 & 1 & 0 & 0 & 0 & 1 & 2 & 3 & 4 \\ 3 & 2 & 1 & 0 & 0 & -1 & 0 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & -1 & -1 & -2 & -1 & -1 & 0 & 1 & 2 \\ 1 & 0 & 0 & -1 & -2 & -3 & -2 & -1 & 0 & 0 & 1 \\ 0 & 0 & -1 & -2 & -3 & -3 & -3 & -2 & -1 & 0 & 0 \\ 0 & -1 & -2 & -3 & -3 & -5 & -3 & -3 & -2 & -1 & 0 \\ 0 & 0 & -1 & -2 & -3 & -3 & -3 & -2 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 & -2 & -3 & -2 & -1 & 0 & 0 & 1 \\ 2 & 1 & 0 & -1 & -1 & -2 & -1 & -1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 0 & -1 & 0 & 0 & 1 & 2 & 3 \\ 4 & 3 & 2 & 1 & 0 & 0 & 0 & 1 & 2 & 3 & 4 \end{bmatrix}$$

With some adjustments made to originally generated masks to make the sum of all elements closer to zero. The images e1-1, e1-2, e2-1, e2-2 generated by these masks are:

Of the two sets, the set generated by M_1 seem to be more detailed. This may be because the sigma used to generate them is smaller, therefore the variance of the Gaussian is smaller, so the detail is more fine, rather than having the thick blurred edges of M_2.

3 References

OpenCV 3.0 Documentation: http://docs.opencv.org/3.0.0/