SOUTH DAKOTA SCHOOL OF MINES & TECHNOLOGY

MEMORANDUM

To: Dr. Randy C. Hoover

From: Sharif Anani

Subject: Edge Map Creation

Date: October 28, 2015

---

1 Problem Summary

In this problem, we are to use what was produced in the last homework (convolution/correlation function) to create edge maps from images using certain kernels that can produce gradients.

2 Background

Edges can be thought of as large changes over a short period or distance. In grayscale images - which are images made from intensity values only - an edge is a large change in intensity from one pixel to the other.
To find edges in a function $y(t)$, we can look at $\frac{d}{dt}y(t)$ and we can see that the edges of $y(t)$ will be where the values of $\frac{d}{dt}y(t)$ are at their local exterma.
in a two dimensional setting, this can be thought of as the gradient (∇) of an image. For example, say the image is denoted by $I(u, v)$, then the gradient of the image

$$\nabla(I(u,v)) = \frac{d}{du}I(u,v) + \frac{d}{dv}I(u,v) \tag{1}$$

To calculate the gradient of an image, we can use the finite difference kernels that when convoluted with an image, produce the gradient in one direction.
The general form of these kernels is:

$$\nabla_x = \frac{1}{p}\begin{bmatrix} -1 & 0 & 1 \\ 2-p & 0 & p-2 \\ -1 & 0 & 1 \end{bmatrix}, \nabla_y = \frac{1}{p}\begin{bmatrix} -1 & 2-p & -1 \\ 0 & 0 & 0 \\ 1 & p-2 & 1 \end{bmatrix}$$

Each choice of p will generate a slightly different gradient, where the Isotropic kernel $(p = 2+\sqrt{2})$ gives less sensitivity to edge orientation.

3 Approach Taken

As with any probelm, this problem is broken to smaller problems/steps. To start with, the image smoothened using `imgaussfilt` and convolved with each of the kernels. After we have the two results, we can now use the magnitude of the gradient. The magnitude of the gradient is obtained by

$$|\nabla(I(x,y))| = \sqrt{\nabla_x^2 I(x,y) + \nabla_y^2 I(x,y)} \tag{2}$$

In the previous result, values of the intensity map will be at their extremes at the edges. To produce a binary edge map, we need to threshold the produced image using some metric.
The chosen metric in this case is distance from the mean. This is done by normalizing the image by subtracting the mean from every pixel and dividing the image by its standard deviation. This way the image has a standard deviation of 1 and a mean of zero. The image is then flattened into a 1-D array and looped through. Every pixel that has an absolute value of more than 0.9 is set to 1, and anything less is set to 0. In pseudo-code, this is expressed as:

```
for all j:
{
if(abs(I(j))>1
I(j)=1;
else
I(j)=0;
}
```

4 Results

The steps will be tested on three demo images available in MATLAB: `cameraman.tif coins.png football.jpg`
Note that in the result illustration, only the horizontal gradient is shown, but both are calculated for the magnitude of the gradient. Only one gradient was displayed to keep things symmetrical in the figure window. The results were as follows:

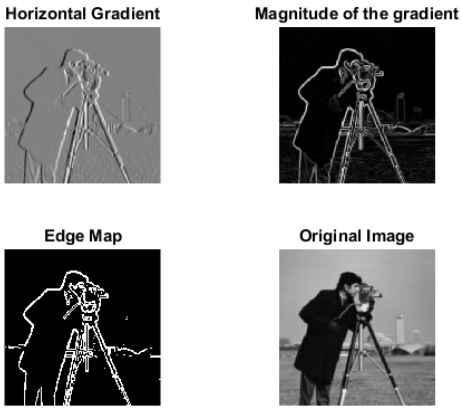

Figure 1: Cameraman Result

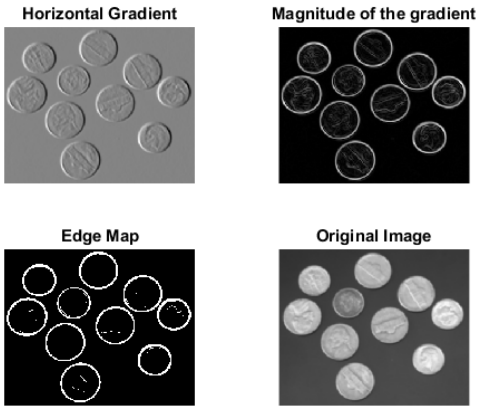

Figure 2: Coins Result

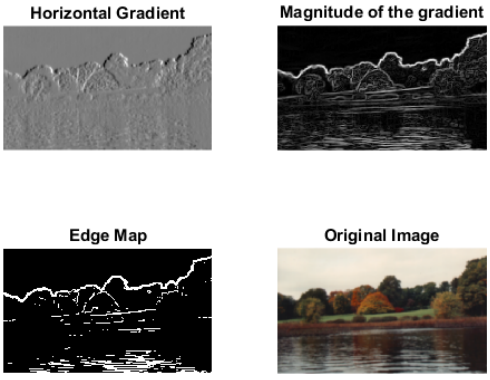

Figure 3: Autumn Result