CS 216 Homework 3

Zachary DeStefano, 15247592

Due Date: May 9, 2014 in class

Problem 1

k-Means Clustering results

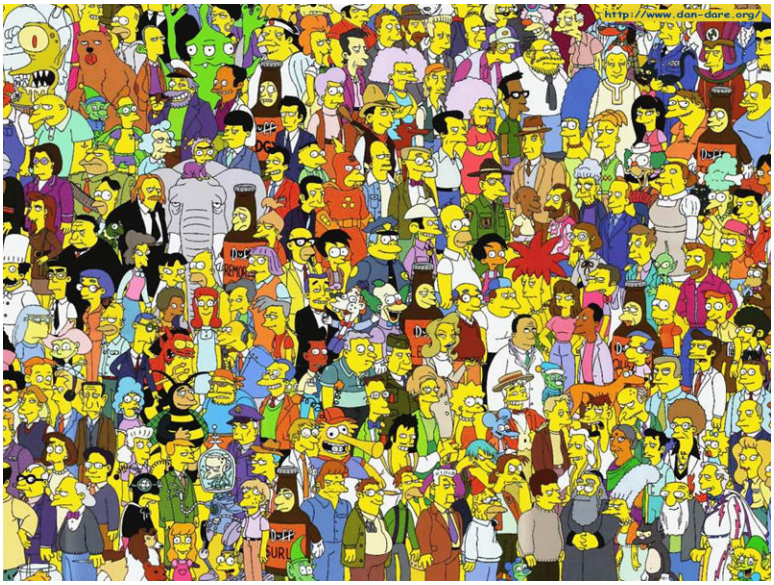

Figure 1: Original Image

Figure 2: The k-Means Image when $k = 2$

Figure 3: The k-Means Image when $k = 5$

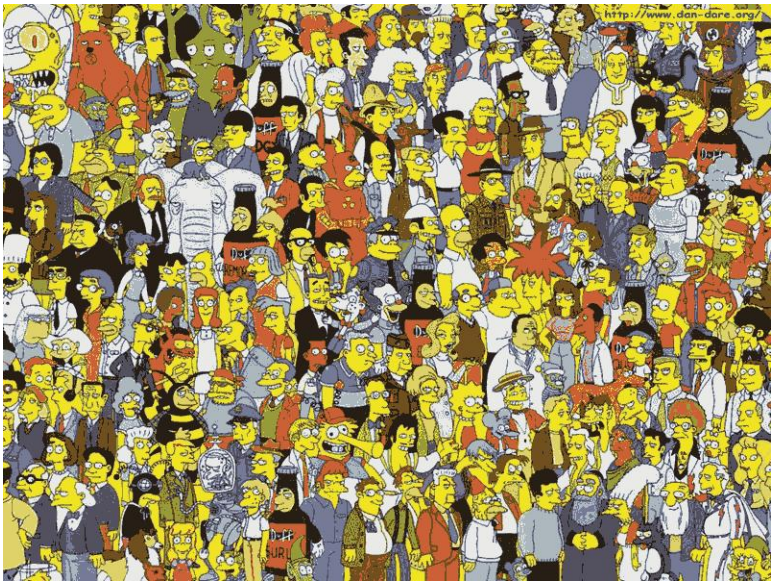

Figure 4: The k-Means Image when $k = 10$

Result when skewing the Red Channel

If we multiply the red channel by 100, then the mean will tend more toward the red channel than the other channels. The result will be a red tint on the final result image. That is exactly what happened with the following images which are the same k-means images as above but the red channel was multiplied by 100 before k-Means was done.

Figure 5: The k-Means Image when $k = 2$ and a skewed red channel

Figure 6: The k-Means Image when $k = 5$ and a skewed red channel

Figure 7: The k-Means Image when $k = 10$ and a skewed red channel

Problem 2

Here is my original image:

Figure 8: The image of the zebra

8 filter images

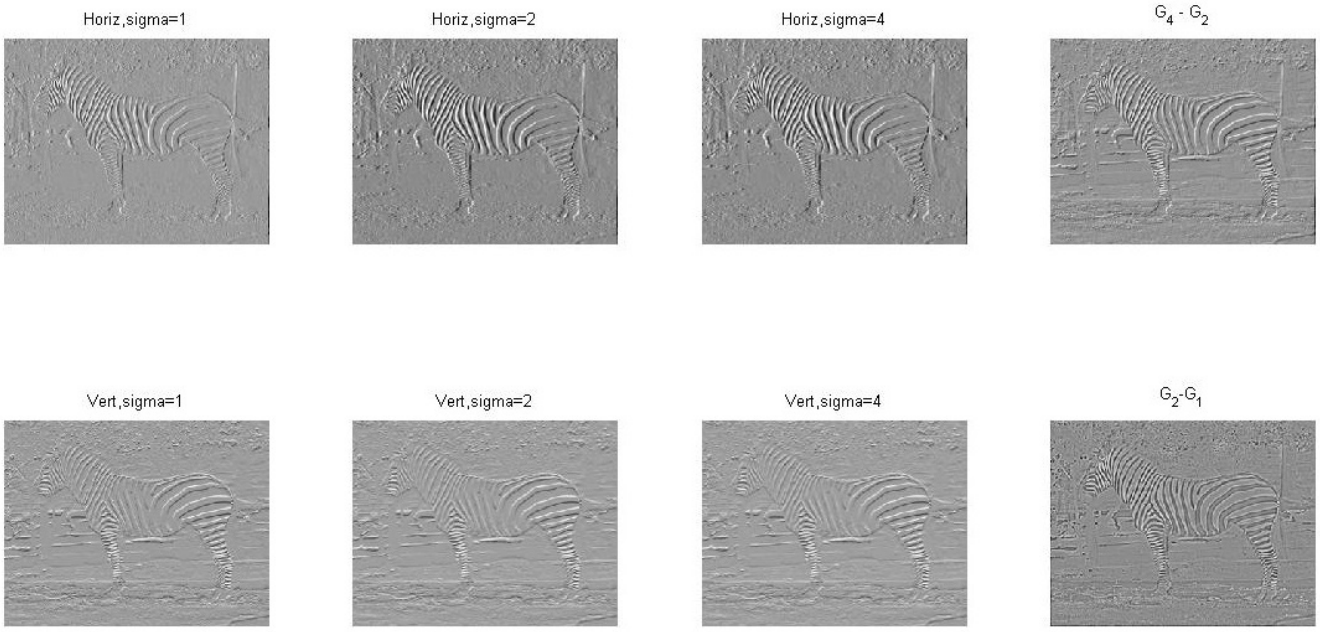

Figure 9: The Filter Images

Matlab code

Here is the parent script that calls the function:

```
imname = 'zebra_small.jpg';
imageData = im2double(rgb2gray(imread(imname)));
%
%h1 means horizontal derivative, sigma 1
%h2, h4, v1, v2, v4 follow the same thing
%g42 is gaussian 4-2, g21 is similar
[h1,v1,h2,v2,h4,v4,g42,g21] = get8FilterImages(imageData);
figure

subplot(2,4,1)
imshow(h1,[])
title('Horiz,sigma=1');
```

```
subplot(2,4,2)
imshow(h2,[])
title('Horiz,sigma=2');

subplot(2,4,3)
imshow(h4,[])
title('Horiz,sigma=4');

subplot(2,4,4)
imshow(g42,[])
title('G_4 - G_2');

subplot(2,4,5)
imshow(v1,[])
title('Vert,sigma=1');

subplot(2,4,6)
imshow(v2,[])
title('Vert,sigma=2');

subplot(2,4,7)
imshow(v4,[])
title('Vert,sigma=4');

subplot(2,4,8)
imshow(g21,[])
title('G_2-G_1');
```

Here is the main function that gets the information:

```
function [ horizDeriv_Sigma1_imageData,vertDeriv_Sigma1_imageData,...
    horizDeriv_Sigma2_imageData,vertDeriv_Sigma2_imageData,...
    horizDeriv_Sigma4_imageData,vertDeriv_Sigma4_imageData,...
    gaussDiff_4_2_imageData,gaussDiff_2_1_imageData] = get8FilterImages( imageData
%GET8FILTERIMAGES Summary of this function goes here
%   Detailed explanation goes here

[horizDeriv_Sigma1_imageData,vertDeriv_Sigma1_imageData] =...
    computeDerivImages(imageData,1);
[horizDeriv_Sigma2_imageData,vertDeriv_Sigma2_imageData] =...
```

```
    computeDerivImages(imageData,2);
[horizDeriv_Sigma4_imageData,vertDeriv_Sigma4_imageData] =...
    computeDerivImages(imageData,4);
gaussDiff_4_2_imageData = computeGaussDiff(imageData,1,2);
gaussDiff_2_1_imageData = computeGaussDiff(imageData,2,4);
end
```

This is the helper function that gets the gaussian derivatives:

```
function [horizDerivImage,vertDerivImage] = computeDerivImages( imageData,sigma )
%COMPUTEDERIVIMAGES Summary of this function goes here
%   Detailed explanation goes here

gaussFilt = fspecial('gaussian',sigma);
rowNum = ceil(sigma/2);
gaussFilter = gaussFilt(rowNum,:);
filteredImageData = conv2(imageData,gaussFilter,'same');
horizDerivFilter = [1 -1];
horizDerivImage = conv2(filteredImageData,horizDerivFilter,'same');
vertDerivFilter = [1;-1];
vertDerivImage = conv2(filteredImageData,vertDerivFilter,'same');

end
```

This is the helper function that computes the gaussian differences

```
function [filterDiffImageData] = computeGaussDiff( imageData,sigma1,sigma2 )
%COMPUTEDERIVIMAGES Summary of this function goes here
%   Detailed explanation goes here

gaussFilt1 = fspecial('gaussian',sigma1);
gaussFilt2 = fspecial('gaussian',sigma2);
filteredImageData1 = conv2(imageData,gaussFilt1,'same');
filteredImageData2 = conv2(imageData,gaussFilt2,'same');
filterDiffImageData = filteredImageData2-filteredImageData1;

end
```

Problem 3

These are the image patches that were filtered and then analyzed. The left one is the neck of the zebra. The middle one is the picture of the tree leaves above the zebra's back. The right one is the grass in front of the zebra. The radius of the patch was scaled and it was scaled in such a way so that the patch would capture that object and only that object.

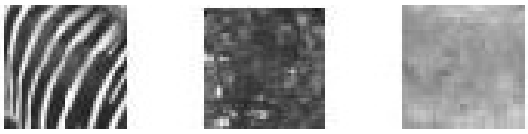

Figure 10: Image Patches that were analyzed

Here is the visualization for each of the mean absolute response vectors.

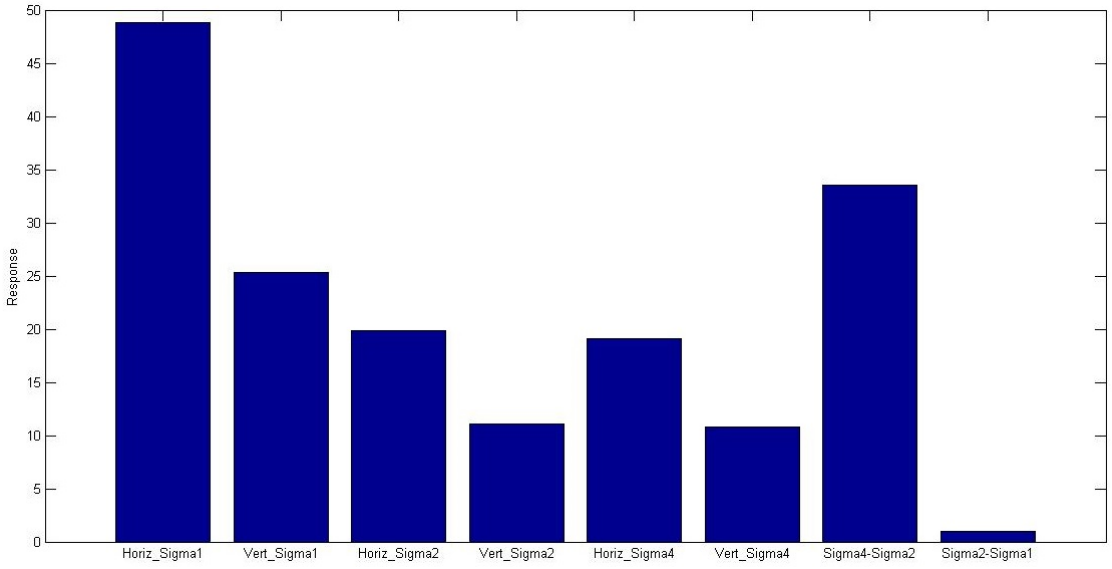

Figure 11: Bar graph visualization for the neck image patch

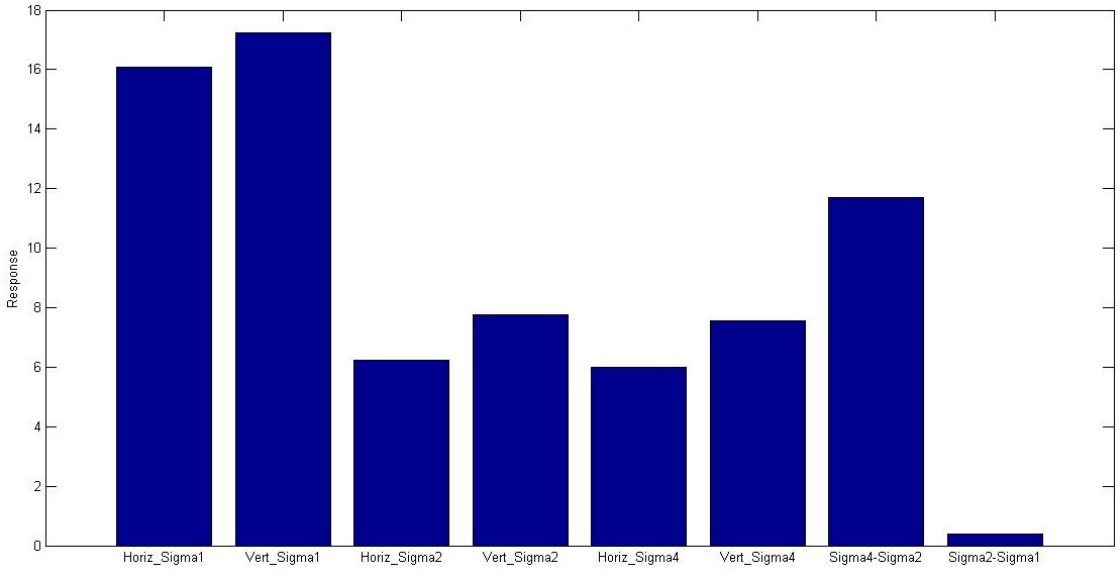

Figure 12: Bar graph visualization for the leaves image patch

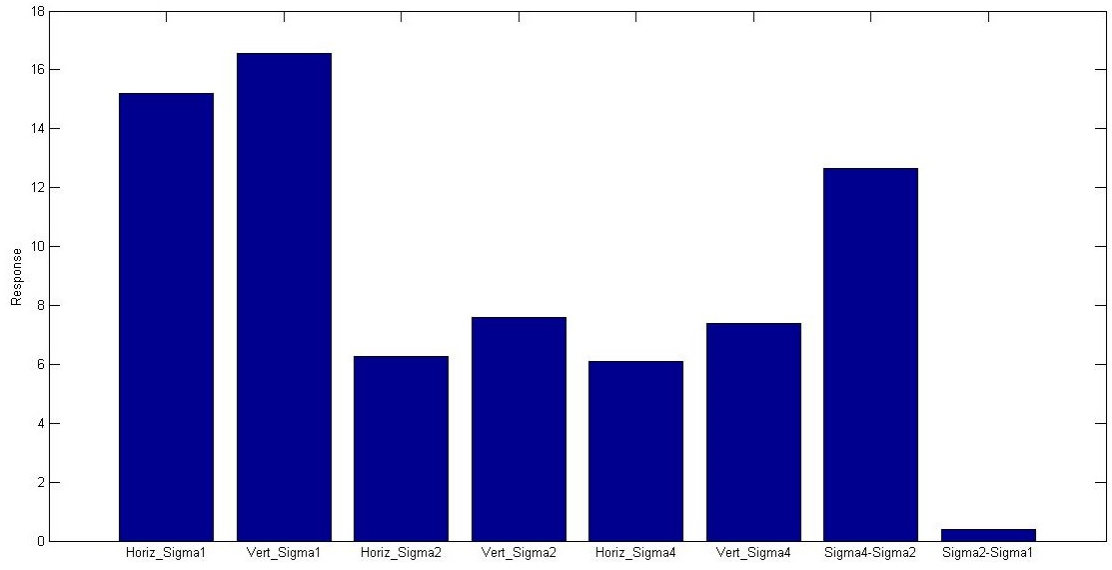

Figure 13: Bar graph visualization for the grass image patch

Here are the filterbank images for each of those patches.

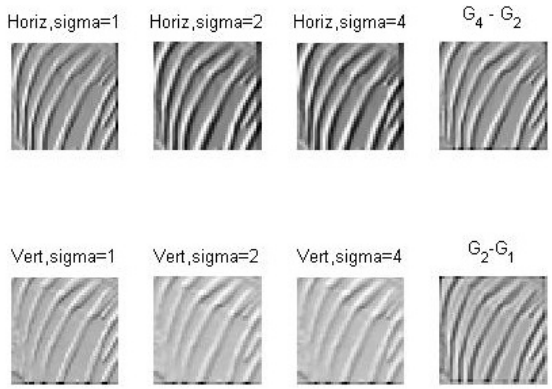

Figure 14: Filterbank for the neck image patch

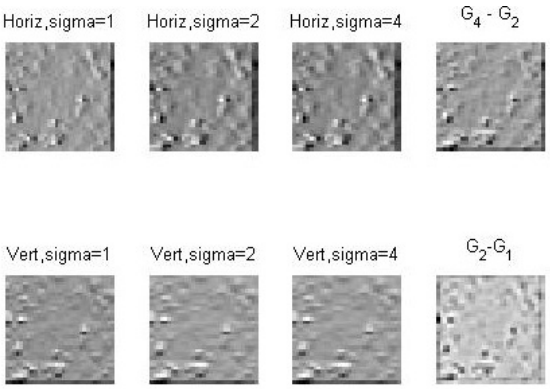

Figure 15: Filterbank for the leaves image patch

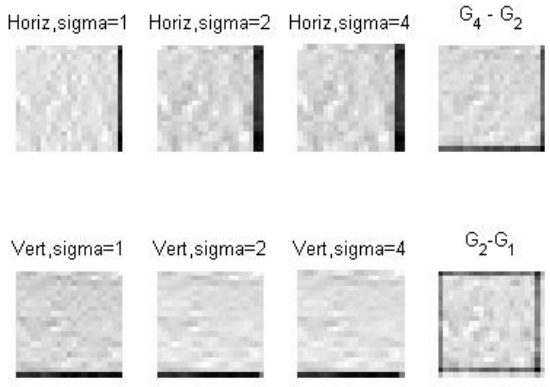

Figure 16: Filterbank for the grass image patch

Problem 4

Here is the color map created by applying the 8 filters to the zebra picture above and then running k-means on the 8-dimensional data created by putting all the images into one large array.

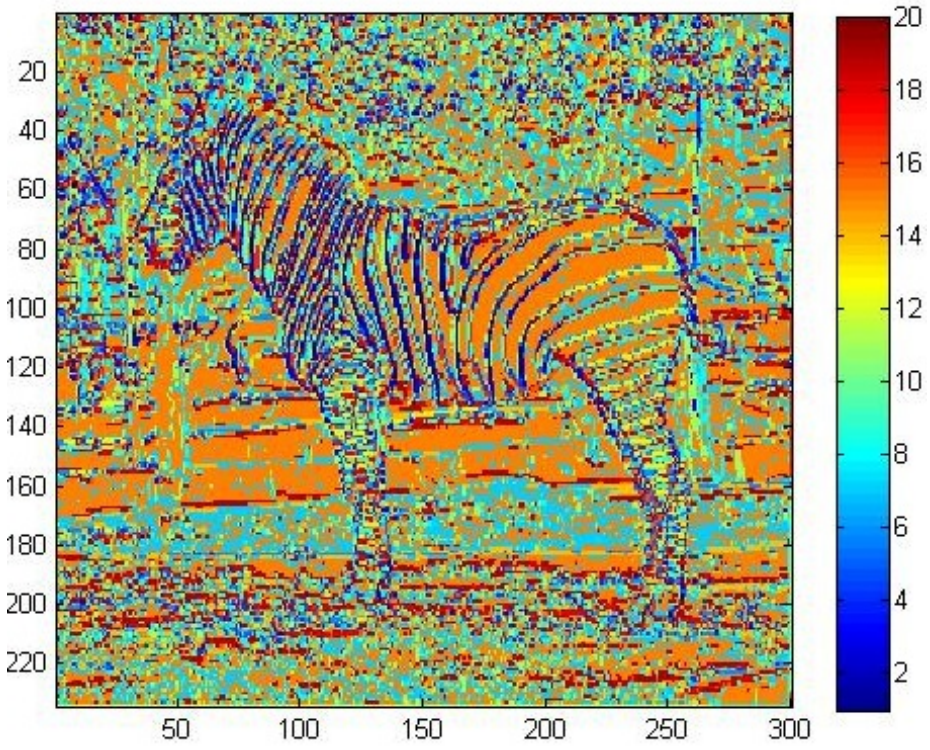

Figure 17: Imagesc visualization of the cluster center labels for the zebra image

Problem 5

Part A

For segtest2.jpg, the results were really nice. The dragon ended up in the foreground and the rest of the image was in the background as expected. There did not need to be any tuning of the λ to accomplish that. Here is the result:

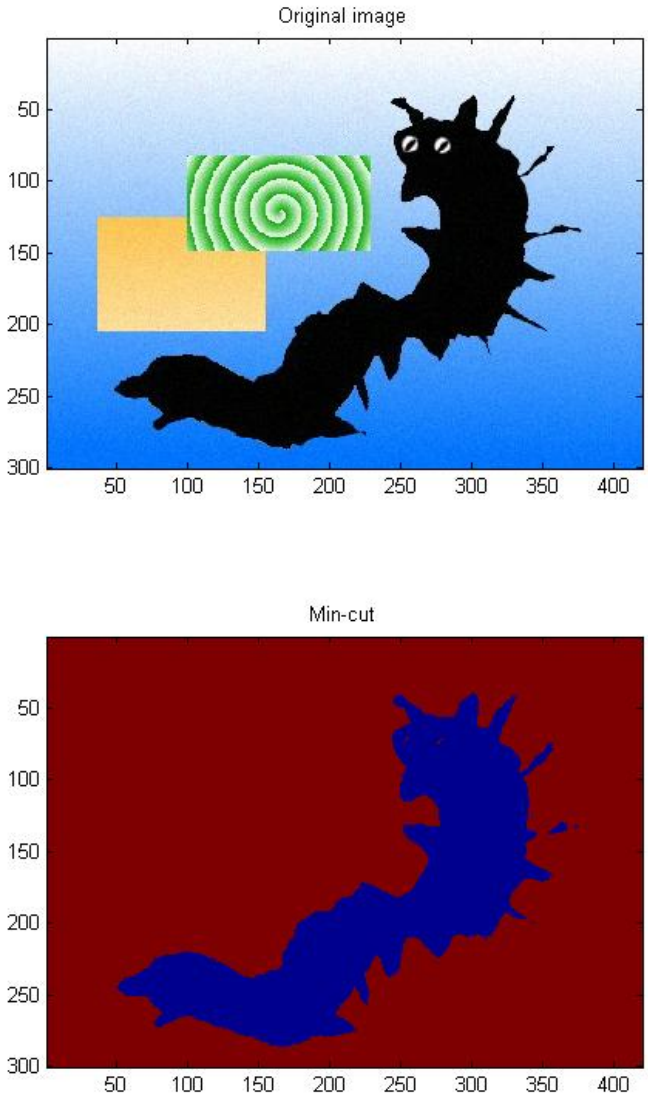

Figure 18: Min cut results for segtest2.jpg

For segtest1.jpg, the results were worse. I tried to make the maroon ball the foreground and the rest the background but it did not quite divide it that way. I made $\lambda = 4$ so that there was it was more likely to select similarity between neighbors to try and offset the effects of similarity in colors between foreground and background objects. It ended up capturing the maroon ball as the foreground but other objects were included as well as part of the background gradient. The background included part of the background gradient as well as other things too. This was all likely due to color similarity between neighboring foreground and background objects. Here are the results.

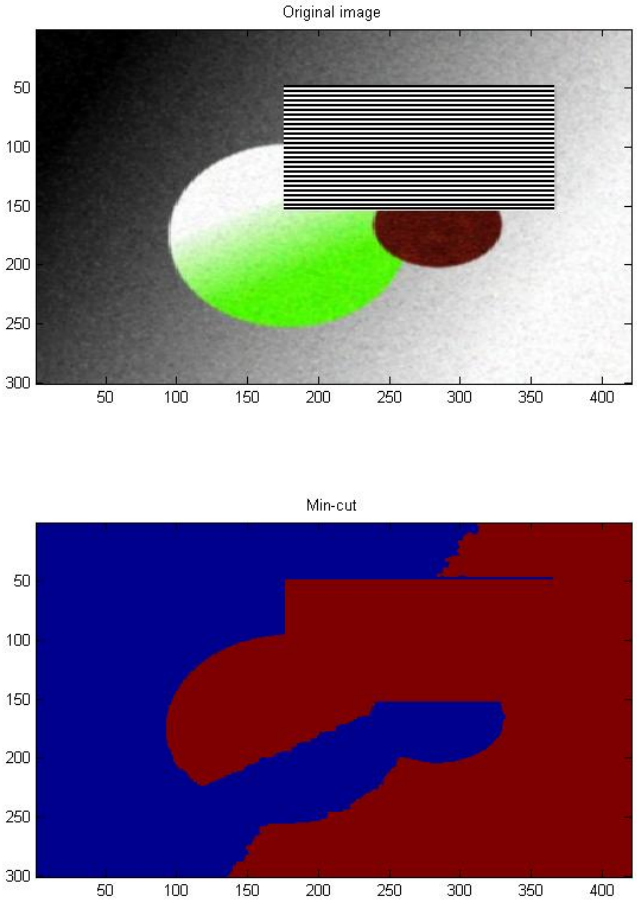

Figure 19: Min cut results for segtest1.jpg

I decided to use a picture of a hockey puck as an interesting example. A hockey puck has a relatively uniform color so I wanted to see if it's shape would come out when using the min cut algorithm. As it turns out, the shape did appear very nicely. It did not need to tune the λ parameter for this case.

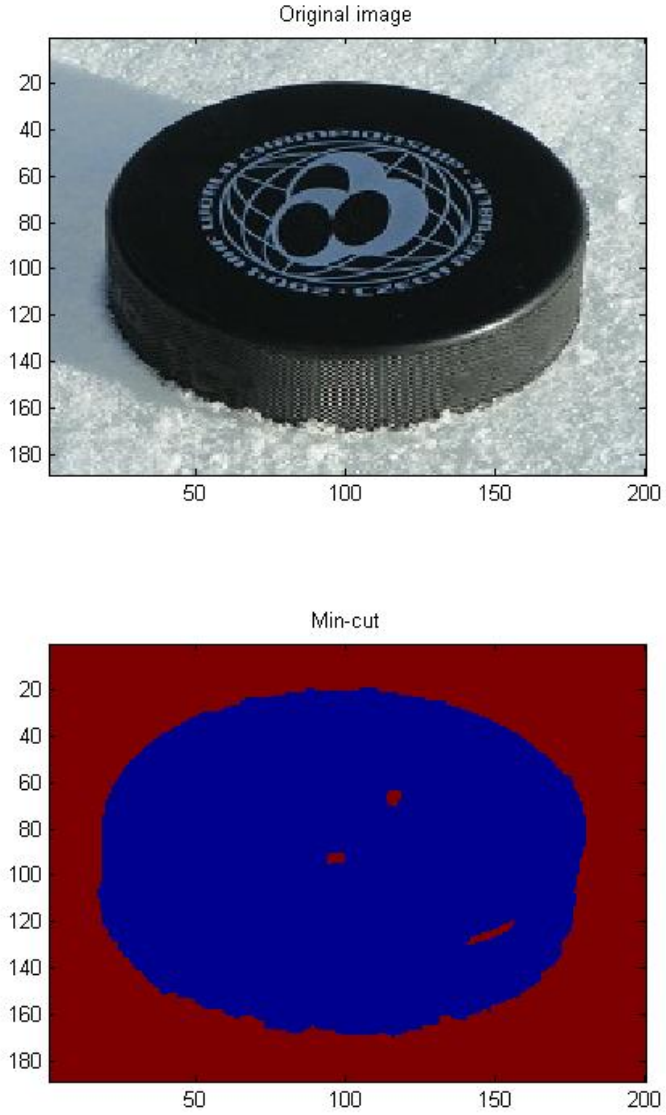

Figure 20: Min cut results for a hockey puck picture

Part B

With this problem, I decided to use the filterbank image data for the unary matrix. The filterbank output showed the edges very well so if the background and foreground are defined by whether there are a lot of edges or color changes, then this approach can be better than using just the colors. For segtest2.jpg, here

was the filter bank output.

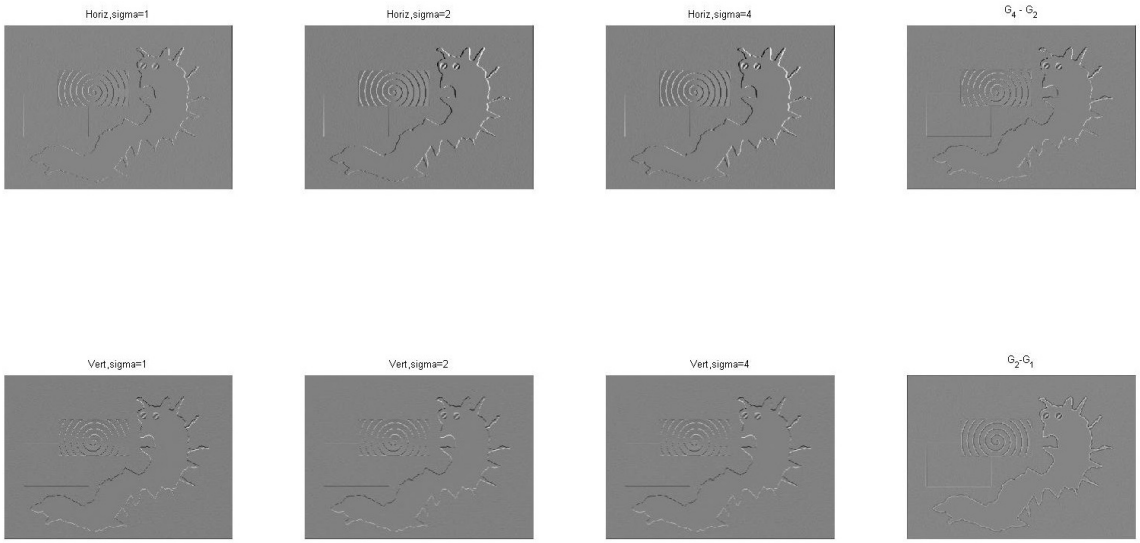

Figure 21: Filterbank for segtest2.jpg

I used $\lambda = 0.001$ to make it more likely to detect similarity in filter bank data and attempted to detect the spiral using the filterbank image data. It worked somewhat well and detected the spiral as well as the outline of the dragon as the foreground and then the rest as the background. It likely detected the outline of the dragon due to the fact that there were edges there too so the filter bank data was similar. Here was the result:

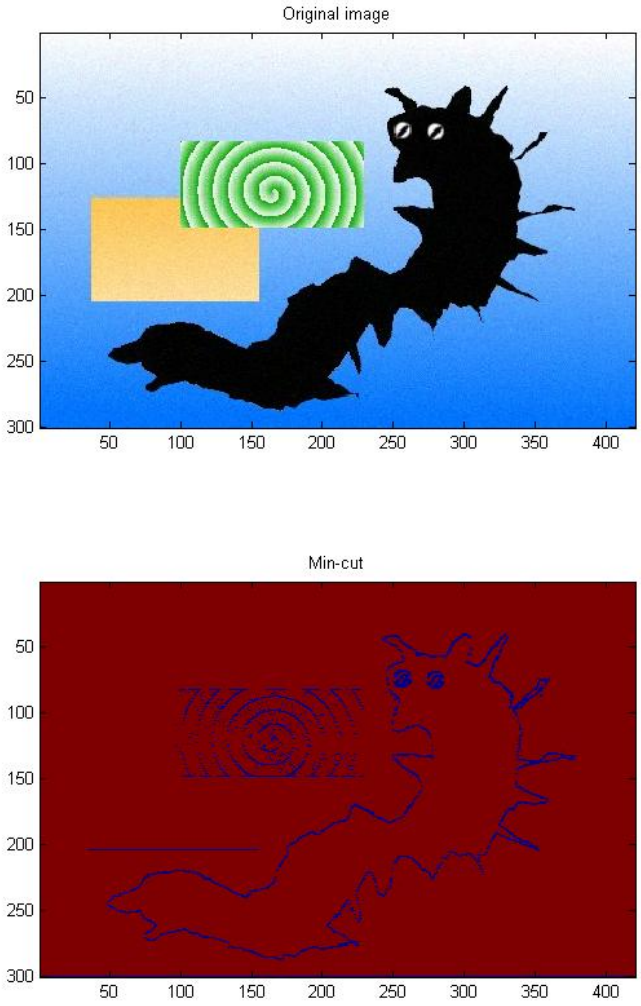

Figure 22: Min cut results for segtest2.jpg

For segtest1.jpg, I took advantage of the fact that the filterbank showed edges well in order to see if I could get it to select the black/white gradient as the foreground and everything else as the background. In the filterbank image set, the black/white gradient shows up quite prominently, as can be seen below:

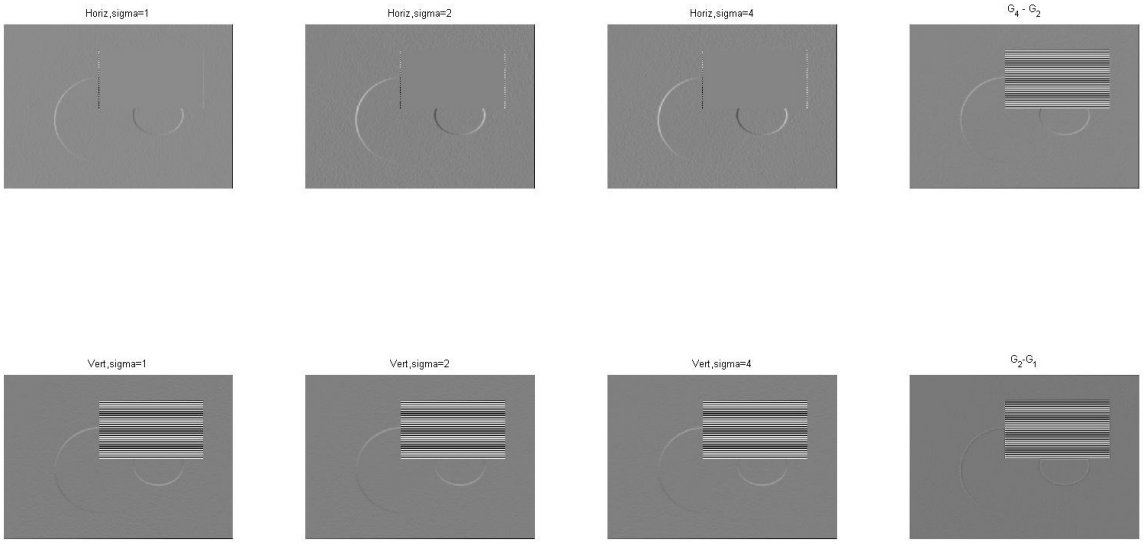

Figure 23: Filterbank for segtest1.jpg

After using $\lambda = 0.01$ to make sure it is more likely to select pixels similar in filterbank coordinate than color, it ended up selecting most of the gradient as the foreground and the rest as the background, as desired. I picked a black part of the black/white gradient as the foreground seed pixel and a white part of the background as the seed background pixel. Here is the result:

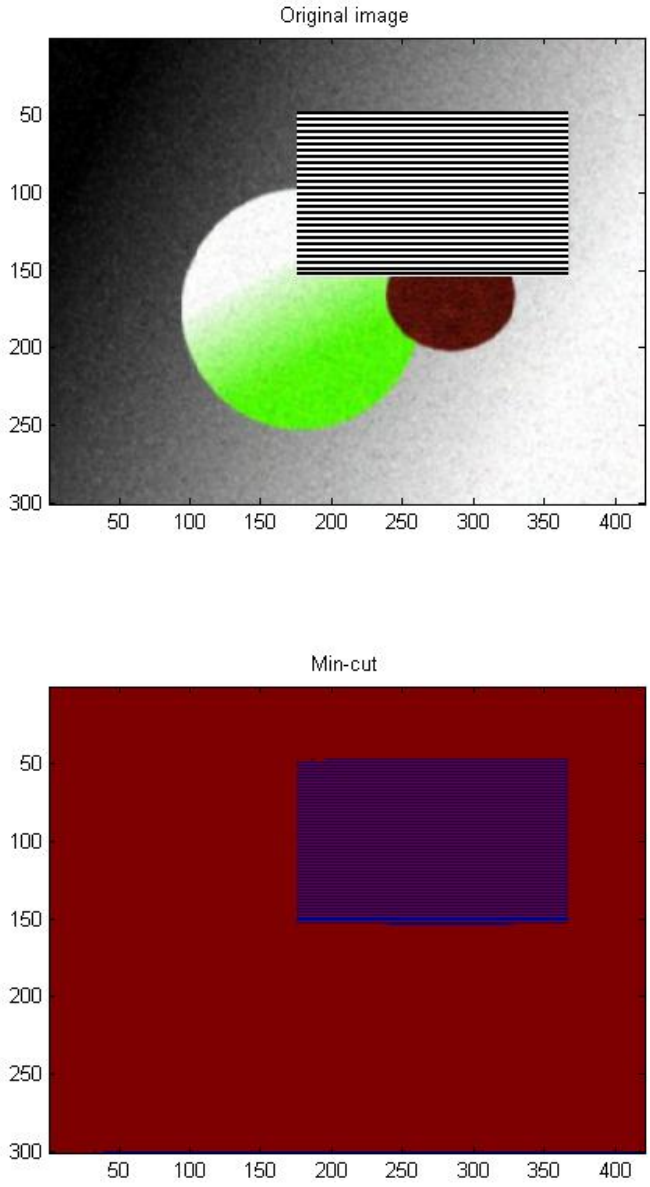

Figure 24: Min cut results for segtest1.jpg

With the hockey puck image, using the filter bank was not advantageous because there were not too many gradients. The filter bank images though were good at detecting where the snow was so I tried to see if it would detect that as the background. Here is the filter bank output:

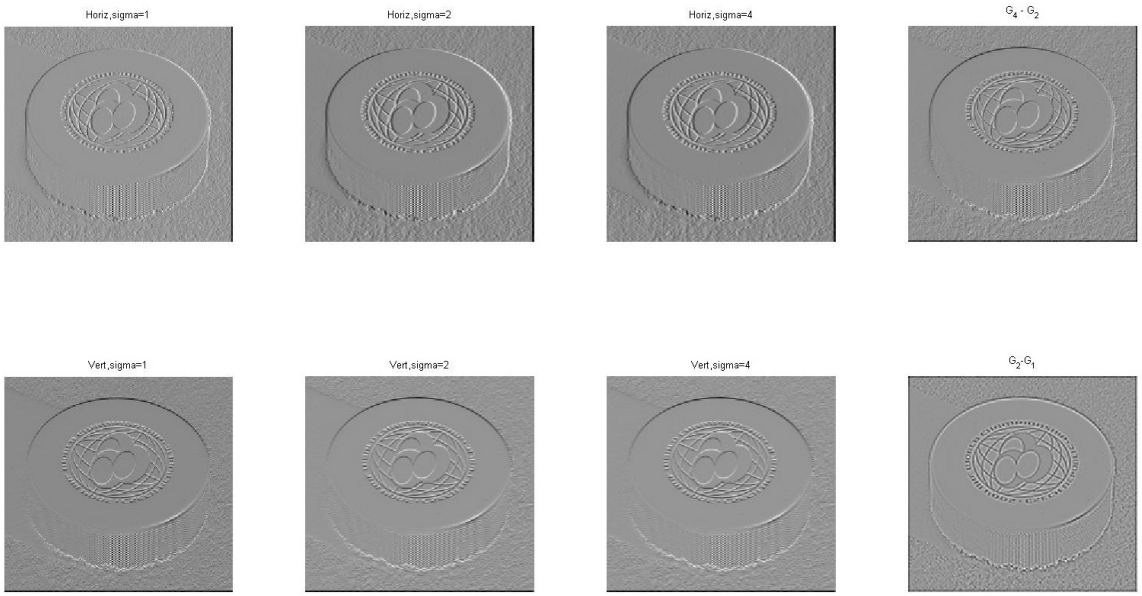

Figure 25: Filterbank for segtest3.jpg

I used $\lambda = 0.001$ and selected the black top of the puck as the foreground and then the white part of the snow as the background. It did alright at trying to detect the two but there was a lot of noise due to similarities in the filterbank images between various parts of the image where there are not edges.

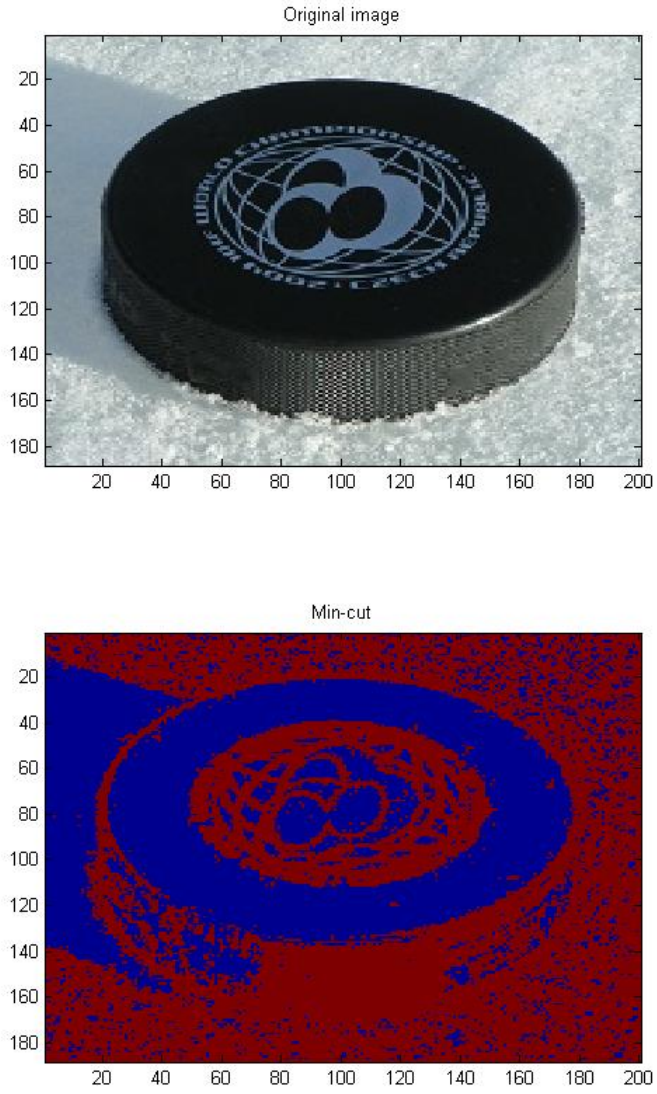

Figure 26: Min cut results for hockey puck image

Problem 6

In the Department of Ophthalmology at UC Irvine, they have pictures of collagen fibers in the eye and they want to use these pictures as a basis for a 3D reconstruction of the fibers. The first step in the 3D reconstruction will be segmenting the images and putting them together. In the images, the fibers look like lines stacked up on top of each other. These lines can merge together as well as split apart into two lines.

Here is a rough outline of the steps of the project:
1. Segment the images to extract out the line data.
2. Use the line information to line up images side by side
3. Get long images that are formed by a series of images put side by side