Assignment #1

- Adwaith Venkataraman (Andrew ID: adwaithv)

Question 1

Analysis:

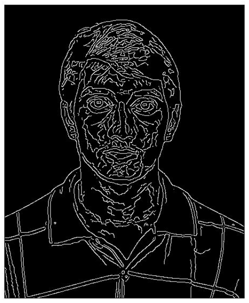
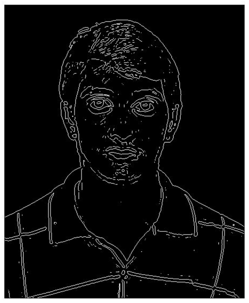
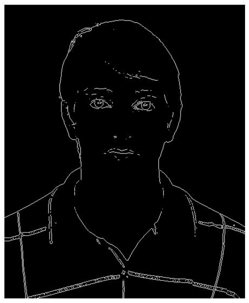
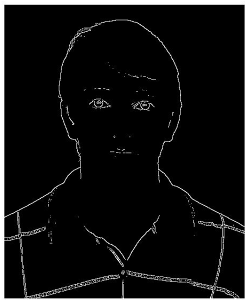
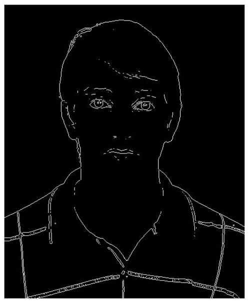
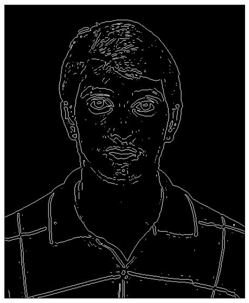
1) The original image used was of resolution 450*550. When converted to gray scale, the red, blue and green values of the image are assigned specific values such that the final image obtained is of gray scale, while retaining the luminance. When the same is repeated for a lower resolution image or the original i.e., 225*275, the result is similar.

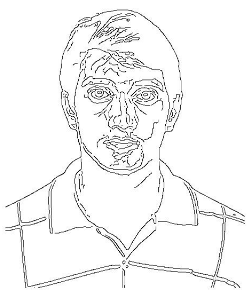
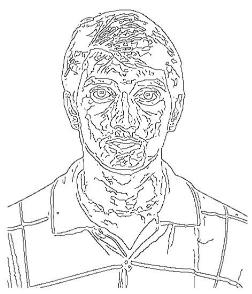
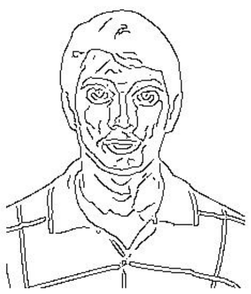
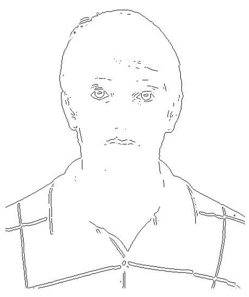
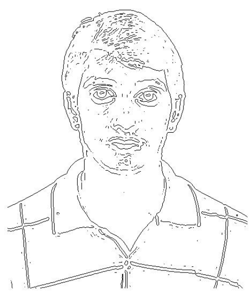
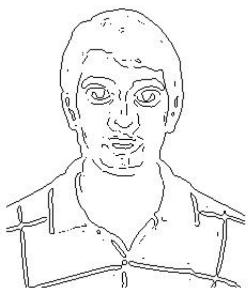
2) To perform the edge detection algorithms, the original color image must be converted to gray scale, and then the edge detection algorithm is applied to the gray scale image. In general, it is observed that at a lower threshold the number of edges detected at the boundaries of pixels also increases. However, the detection patterns vary with various algorithms. For instance, with the 'canny' algorithm and with a lower threshold, the number of edges detected increases. This is because the minimum required difference between adjacent pixels is reduced, hence more edges are detected. With a lower resolution, the image is packed with a lesser number of pixels. Therefore, the edges detected seem to appear thicker, although the number of edges detected is lesser than that in the higher resolution image.

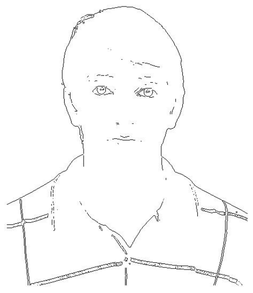
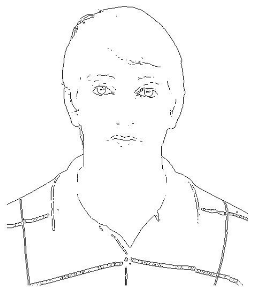
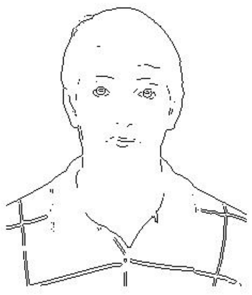
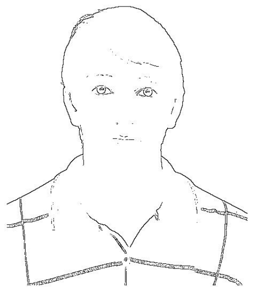
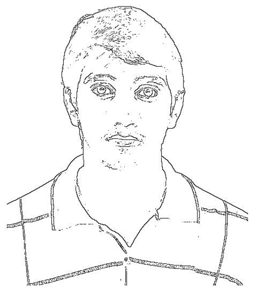
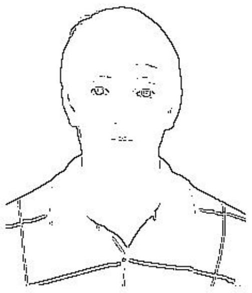
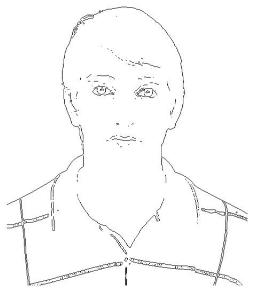
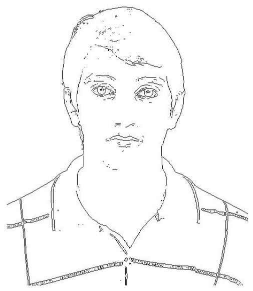
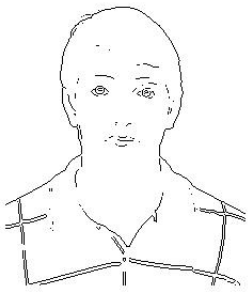
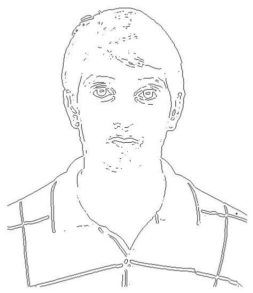
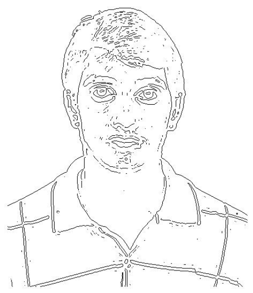
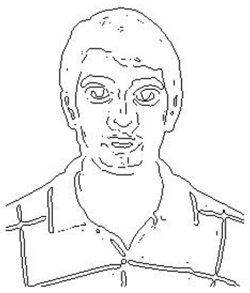
3) With different edge detection algorithms, the detected edges varied. Among the algorithms, Sobel and Prewitt have the advantage of being simple, however they are inaccurate. The Laplacian of Gaussian method is known to conduct a test a wider area around every pixel, thereby finding the right position of the edges. However, it does reflect the correct orientation of the edges due to the presence of the Laplacian filter. Among all, the Canny algorithm returns the more accurate results. This is due to the use of probability to find the error rate.

Results:

Original Image		**Image in Gray Scale**	
Higher Resolution	Lower Resolution	Higher Resolution	Lower Resolution
Prior to Image Reversal			

			
Canny	Laplacian of Gaussian (Log)	Prewitt	Roberts
			
	Sobel	Zerocross	

Edge Detection Algorithm	**Higher Resolution**		**Lower Resolution**
	Higher Threshold	Lower Threshold	
Canny			
Laplacian of Gaussian (Log)			

Prewitt			
Roberts			
Sobel			
Zerocross			

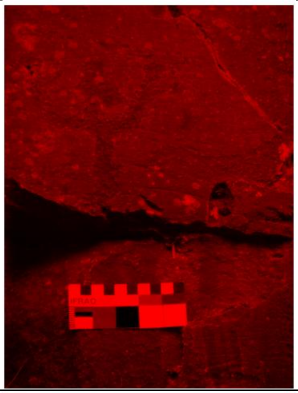
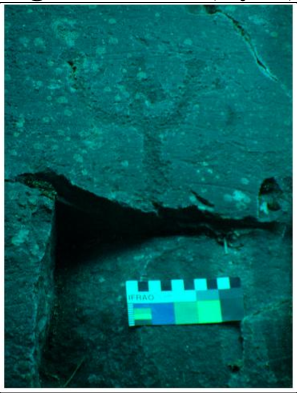
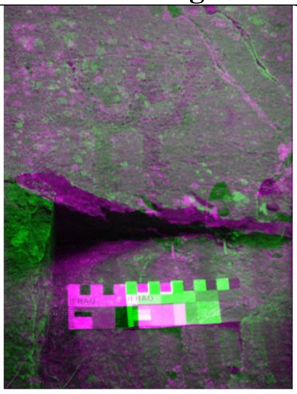
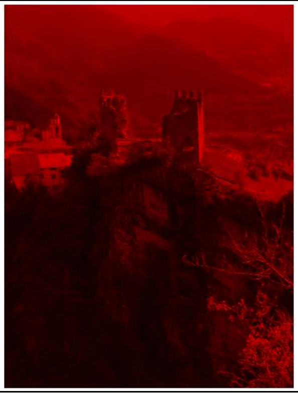
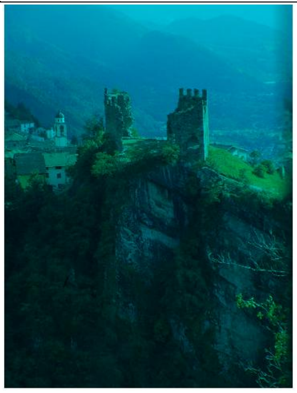
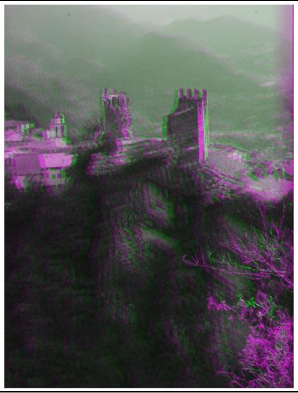
<u>Question 2a</u>

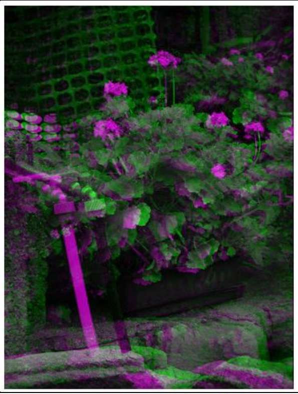
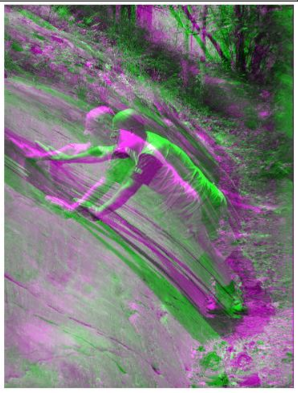
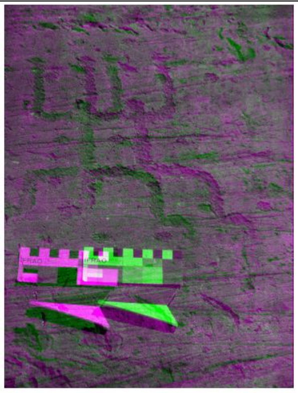
Analysis:

The following steps were followed to produce the anaglyph:

- The given stereo image was read and stored into a variable.
- It was then cropped into two exact halves, vertically, with respect to the pixel ratio. This was performed by using the 'Imcrop' function in matlab.
- The left half (channel) and right half (channel) were stored in separate variables.
- The left channel's blue and green components were made zero i.e., the red channel. And the right channel's red component was made zero i.e., cyan.
- The left and right channels were then overlapped with the inherent displacement, to generate the 3-D image. This action was performed using the Imfuse function of matlab.

Results:

Left Channel (Red)	Right Channel (Cyan)	3-D Image
		
		

Appendix:

1) Source Code for Q1.

```
a=imread('0003.jpg'); %read the original image
b=rgb2gray(a); %convert to gray scale
c=edge(b,'canny'); %'Canny' edge detection algorithm
d=edge(b,'log'); %'Laplacian of Gaussian' edge detection algorithm
e=edge(b,'prewitt'); %'Prewitt' edge detection algorithm
f=edge(b,'roberts'); %'Roberts' edge detection algorithm
g=edge(b,'sobel'); %'Sobel' edge detection algorithm
h=edge(b,'zerocross'); %'Zerocross' edge detection algorithm

ci=imcomplement(c);
di=imcomplement(d);
ei=imcomplement(e);
fi=imcomplement(f);
gi=imcomplement(g);
hi=imcomplement(h);

[c,thresh]=edge(b,'canny',0.09); %original threshold = 0.0188 & 0.0469
imshow(imcomplement(c));

[d,thresh]=edge(b,'log',0.01); %original threshold = 0.0031
imshow(imcomplement(d));

[e,thresh]=edge(b,'prewitt',0.11); %original threshold = 0.0960
imshow(imcomplement(e));

[f,thresh]=edge(b,'roberts',0.05); %original threshold = 0.1117
imshow(imcomplement(f));

[g,thresh]=edge(b,'sobel',0.07); %original threshold = 0.0978
imshow(imcomplement(g));

[h,thresh]=edge(b,'zerocross',0.004); %original threshold = 0.0031
imshow(imcomplement(h));

x=imresize(a,0.5); %Changing the resolution of the original image
```

2.a) Source Code for Q2.a

```
img=imread('1.jpg'); %Read a stereo image
right=imcrop(img,[0 0 360 479]); %Crop the right half
left=imcrop(img,[360 0 360 479]); %Crop the left half
right(:,:,1)=0; %Set the right channel's red component to zero i.e., cyan
```

```
left(:,:,2)=0; %Set the left channel's green component to zero
left(:,:,3)=0; %Set the left channel's blue component to zero
imshow(right);
imshow(left);
img_3D=imfuse(left,right); %Overlap the left and right channels to get the 3D image
imshow(img_3D);
```