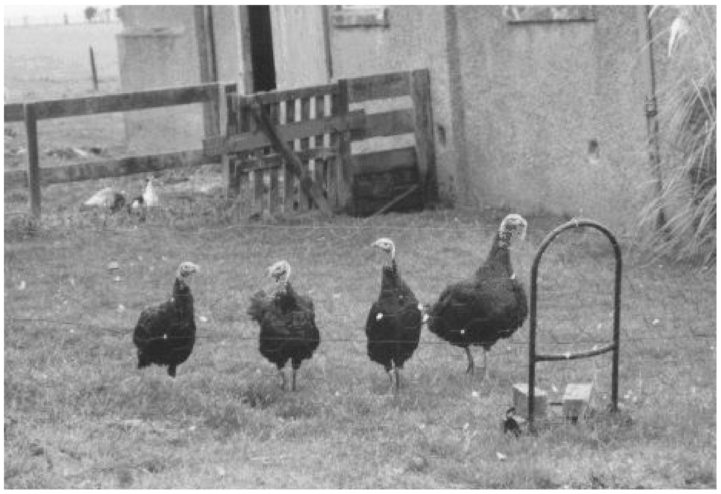
Ravi Shankar Kannan
U1064922
ravi.kannan@utah.edu

Fall 2017
Image processing - project 1 report

1. Histogram:

This histogram function is implemented in 'histogram.m'. The histogram function gives the relative frequency of the pixel intensities specified between the minimum threshold. The function takes a 2D array (image) I as input (integers or floats) and returns a 1D array of floats (length n) that give the relative frequency of the occurrence of grey scale values in each of the n bins that equally (to within integer round off) divide the range of (integer) values between min and max. For each image shown, ie, the turkey image's intensity values between 100 to 130 occur at almost the same frequency. In the image, Lenna, the intensity of 0 and 255 is very minimum, and it can be observed in the histogram. For the brain image, a lot of grey(near 120s) appears which explains the frequency on the center to be more.



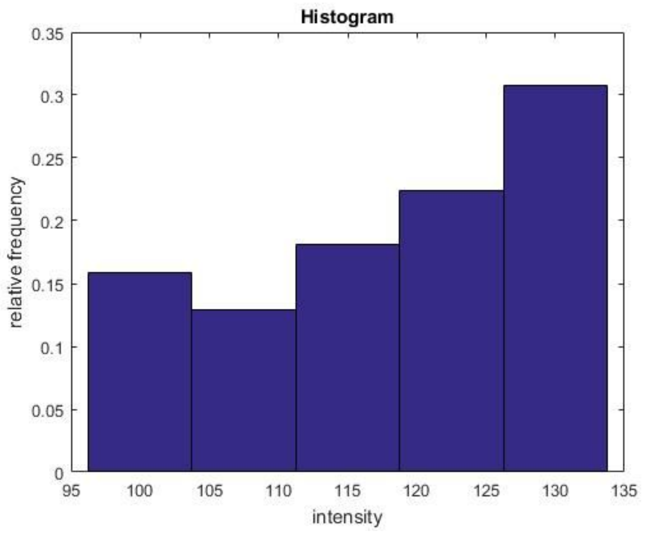
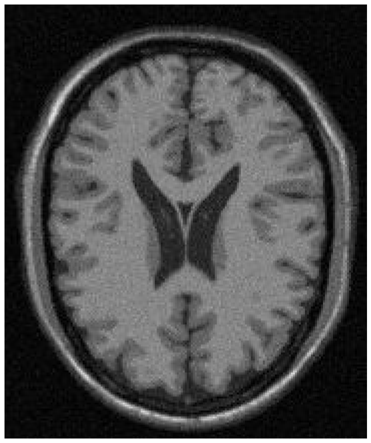


Figure 1. Left: image:'turkeys.tif';
Figure 2: Right: Histogram values for image 'turkeys.tif'. Threshold: min:100 max:130 length: 5



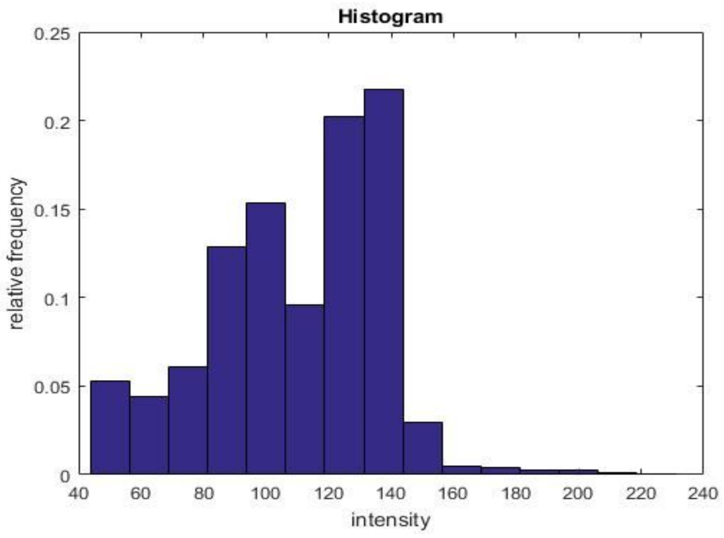
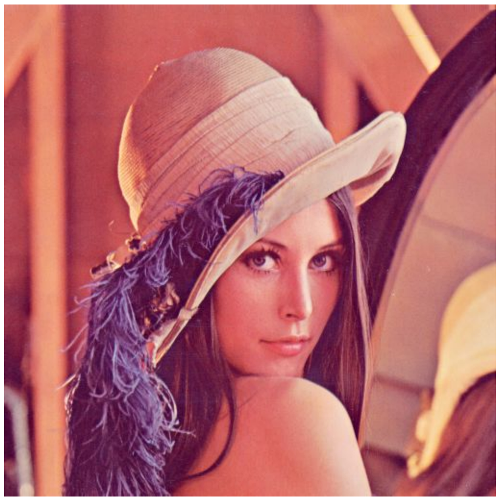


Figure3: Left: image:'brain.tif'

Figure 4: Right: Histogram values for image 'brain.tif'. Threshold: min:50 max:225 length: 15



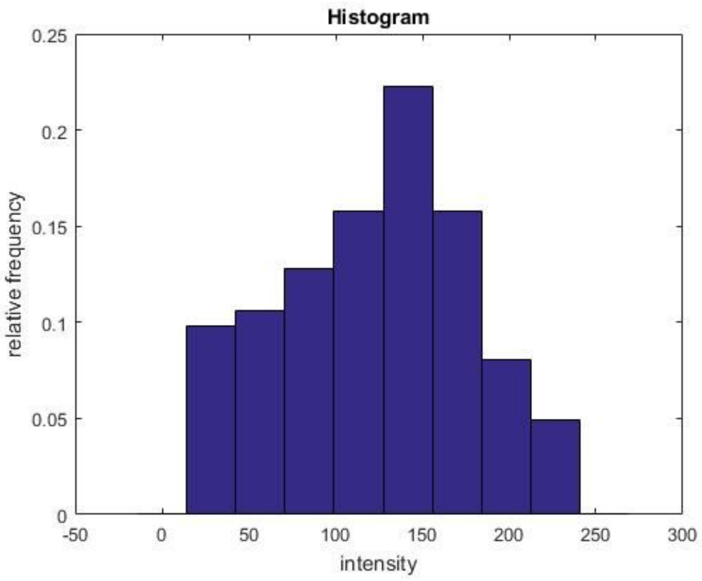
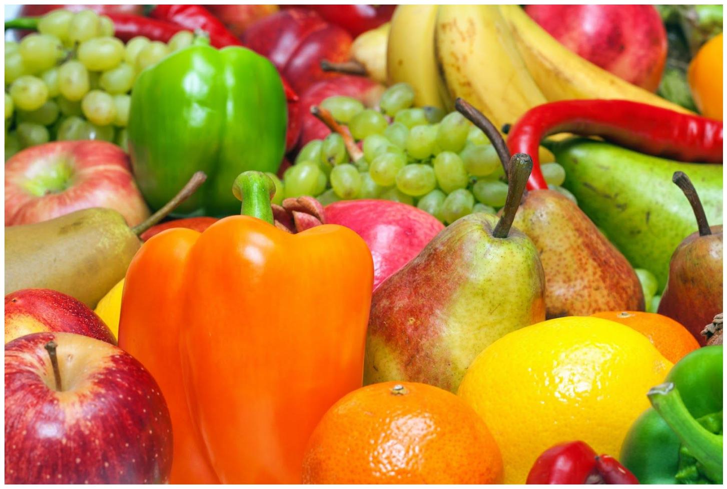


Figure5: Left: image:Lenna.png''

Figure5 Right: Histogram values for image 'Lenna.png''. Threshold: min:0 max:255 length: 10



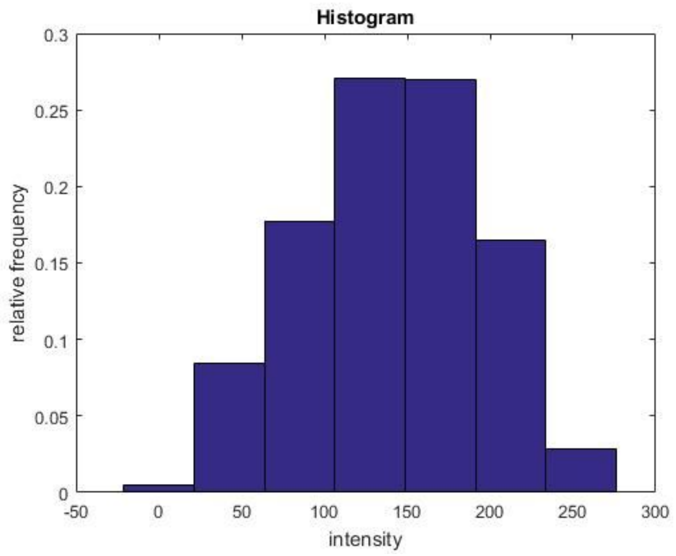
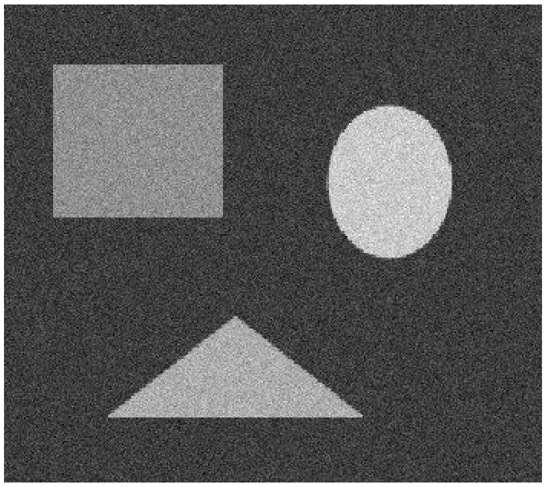


Figure6: Left: image:vegetables.jpg

Figure6: Right: Histogram values for image vegetables.jpg’. Threshold: min:0 max:255 length: 7



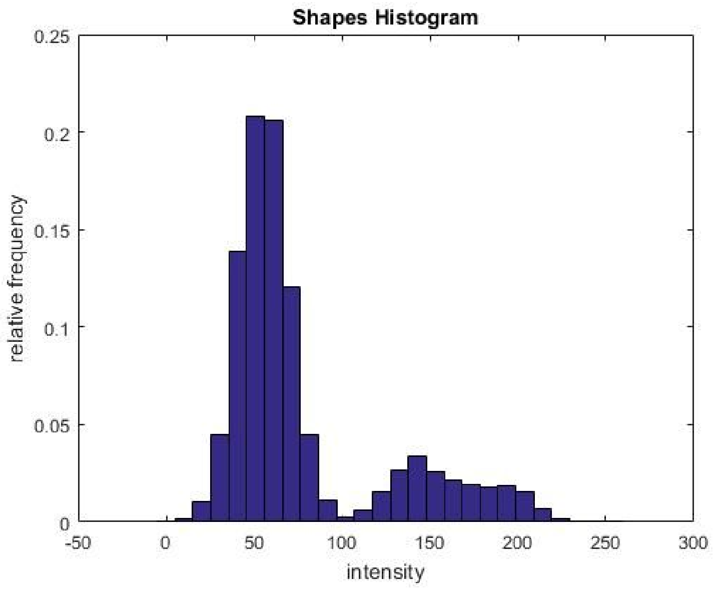


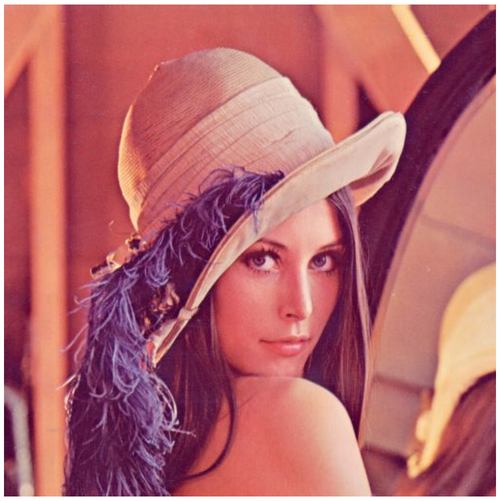
Figure7: Left: image:shapes_noise.tif

Figure7: Right: Histogram values for image shapes_noise.tif’. Threshold: min:0 max:255 length: 10

2. Connected Components Analysis

a. Dual threshold:

The dual threshold function is implemented in binary_image.m’ file. The function takes two thresholds and returns a binary image of pixels that fall between them. The intensities between the two thresholds appear white and the other intensities appear black. This was done by checking the pixel intensities between the minimum and maximum thresholds.



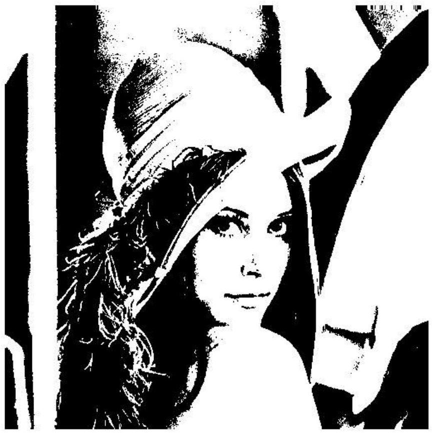
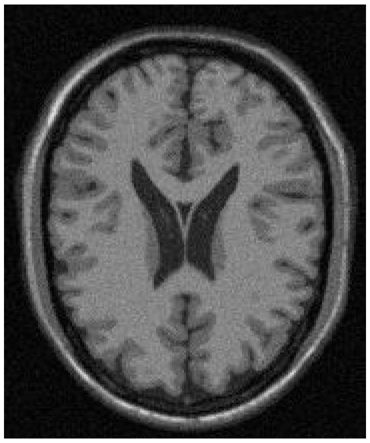


Figure7: Left: image:Lenna.png’’

Figure8: Right: Dual threshold of Lenna.png. Min threshold: 120, max threshold: 255



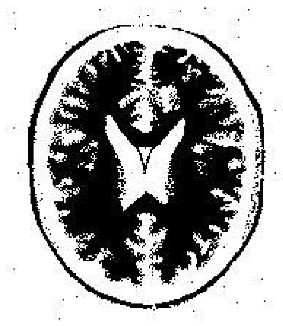


Figure9: Left: image:brain.jpg’’

Figure10: Right: Dual threshold of brain.tif. Min threshold: 0, max threshold: 100

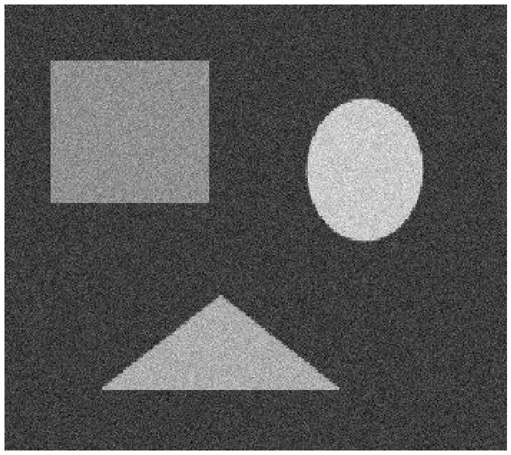
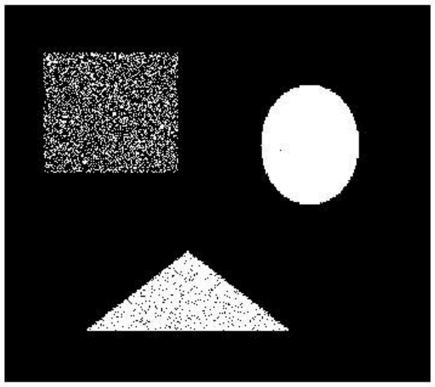



Figure11: Left: image:shapes.jpg’’
Figure12: Right: Dual threshold of shapes.tif. Min threshold: 0, max threshold: 100

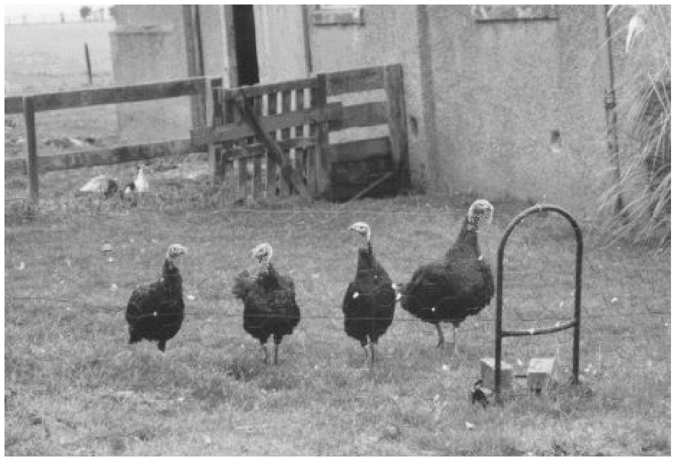
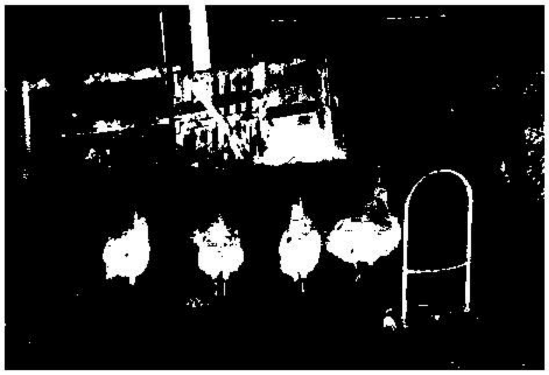



Figure13: Left: image:shapes.jpg’’
Figure14: Right: Dual threshold of shapes.tif. Min threshold: 10, max threshold: 90

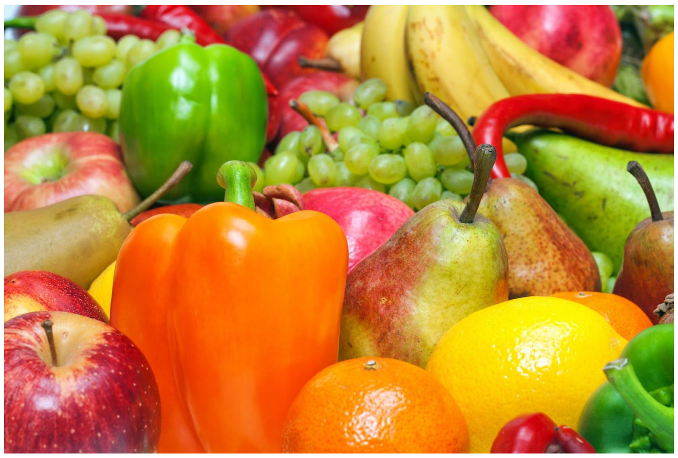
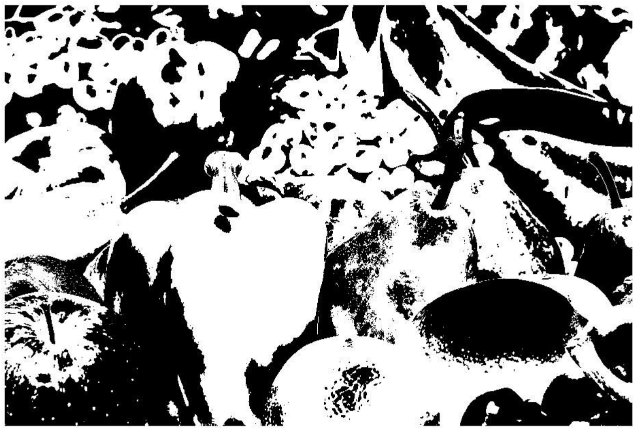



Figure13: Left: image:veg.jpg’’
Figure14: Right: Dual threshold of veg.tif. Min threshold: 120, max threshold: 200

b. Flood Fill:

The flood fill function is implemented in flood_fill.m’ file. The flood fill finction takes inputs of binary image, a seed point, and label value (for the output) and a label image (for the output) and returns a flood fill on the label image with the label value. The method ois implemented by finding the neighbouring pixels having the same binary value. The adjacency followed is 4. The filling in that direction stops when a neighbour with opposite binary value is enouuntered and is completed when all the connected pixels with the same pixel intensity in binary is filled.

The seed point is passed from the main file and the filling starts from the seed point and finds the neighbouring pixels having the same pixel intensity as the seeded pixel.

The following points were seeded:
For turkeys image, the 2nd turkey from the left was given as seed point.
For Brain image, the centre of the brain was given as seed point.
For Lenna’s image, the point below her face was given as seed point.
For noisy shapes image, the rectangle on the top left was given as seed point.

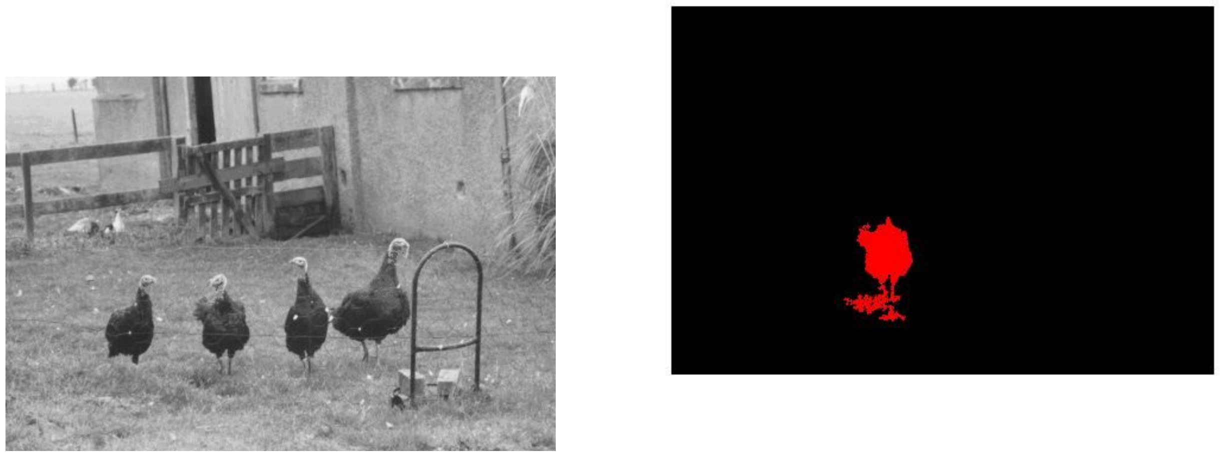


Figure 15:Top Left & Right: Flood Filled Turkey,

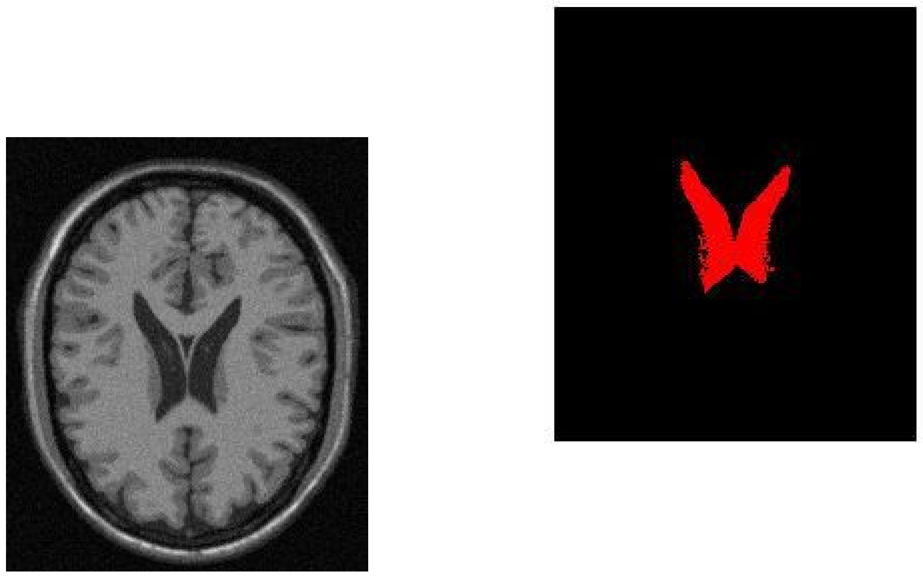


Figure 16: Bottom Left & Right: Flood Filled brain

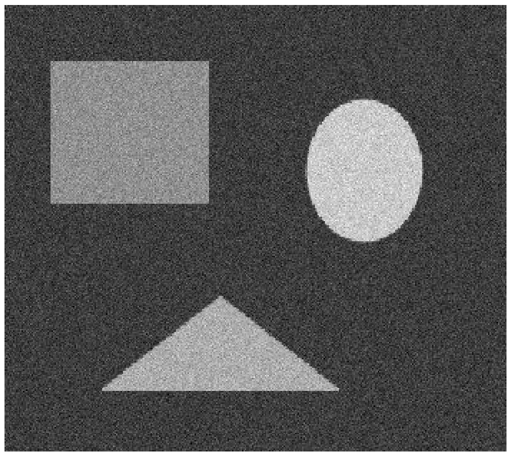
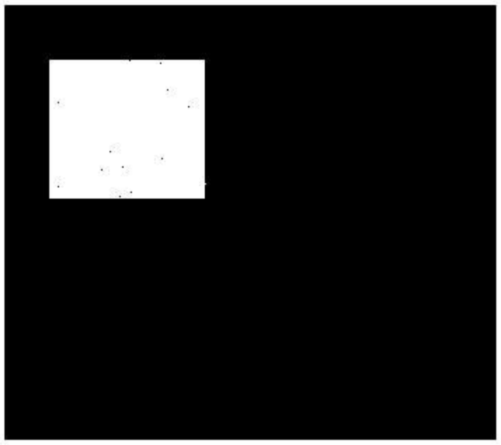



Figure 17 Flood filled noisy shapes;

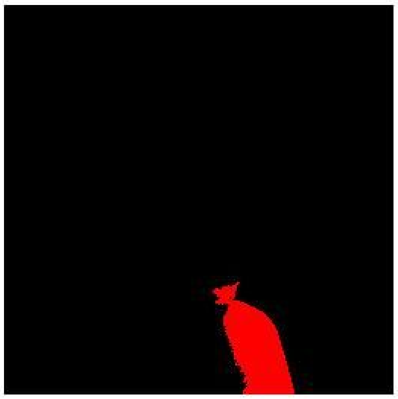


Figure 18: Flood Filled Lena

c. Connected Component: (Images along with topological denoising)

The Connected component function is implemented in connected_component.m’ file. The connected component routine takes as input a binary image, starting label value, label image, and returns the revised label image, with new labels in the CCs for this binary map, and an ending label value. The routine is implemented by labelling all the neighbouring pixels having the same intensity as connected. When all the connected pixels are completed, the label is incremented and the method is repeated until all the pixels are marked with labels. Once this is done, each label is given a different color at random.

Once one component is labelled, the algorithm remembers the labelled regions, the does not label them again. The algorithm is repeated until all the pixels are filled with a label. Depeding on the label value, the pixels are then filled with a random color.

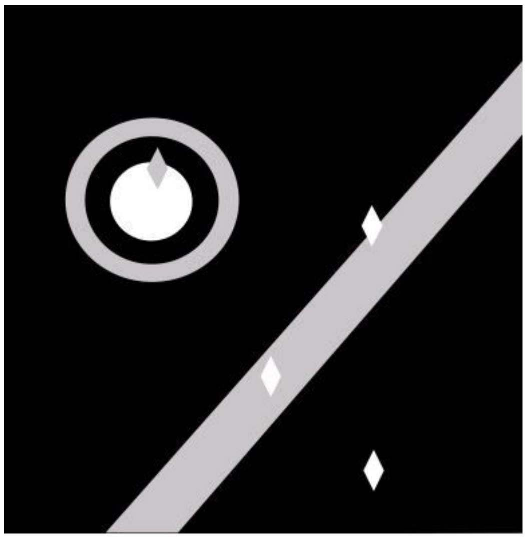
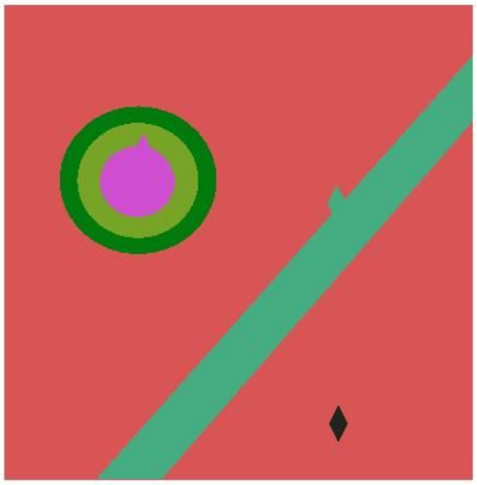



Figure: Connected components - Topo.jpg

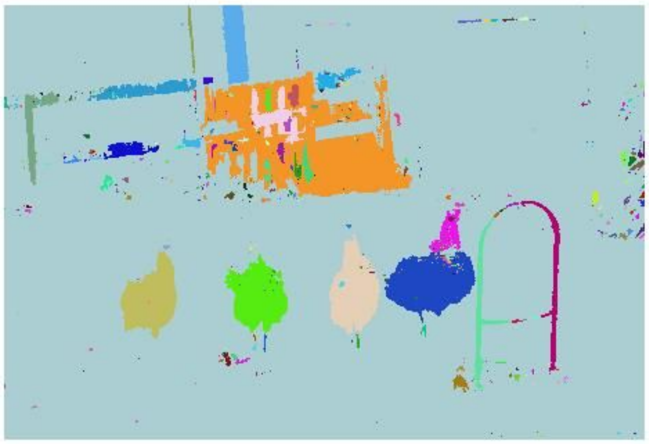


Figure: Turkeys; Connected Components

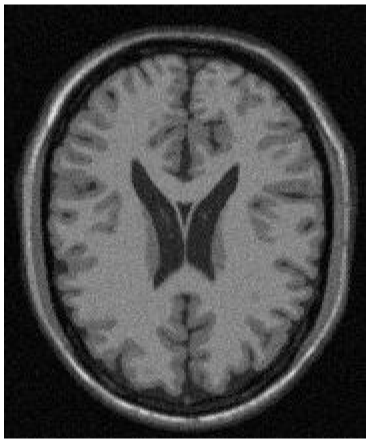
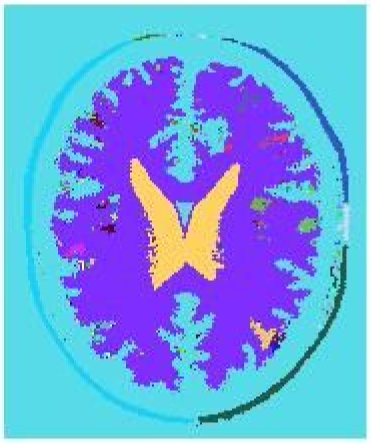



Figure: Brain; Connected components of brain

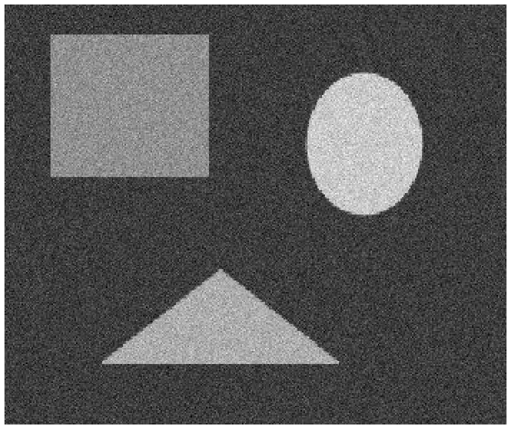
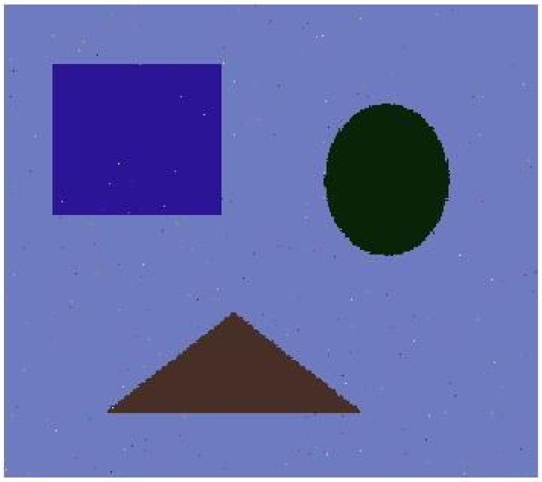



Figure: Noisy Shapes & Its connected components

3. Topological Denoising

The Topological Denoising function is implemented in ‘topological_denoise.m’ file. The inputs are the connected component image, the noise threshold and the ending label value. The noise threshold determines, how many pixels are considered to be noise.

I have converted the labels for these components to the label of an adjacent component which has the largest boundary with the little one. When there are multiple labels surrounding a noisy region, the denoising region is decided by the label that surrounds the maximum. To denoise, I first applied the dual threshold to make it binary, and then applied connected component function to find the connected components in the image. Then, the denoising function is applied which removes the noise on the image. For the below image, the threshold needs to be high so that the diamond shaped particles are denoised. But for the other images, like the brain, the noise is very small in size. And hence, the threshold is also given a smaller value.

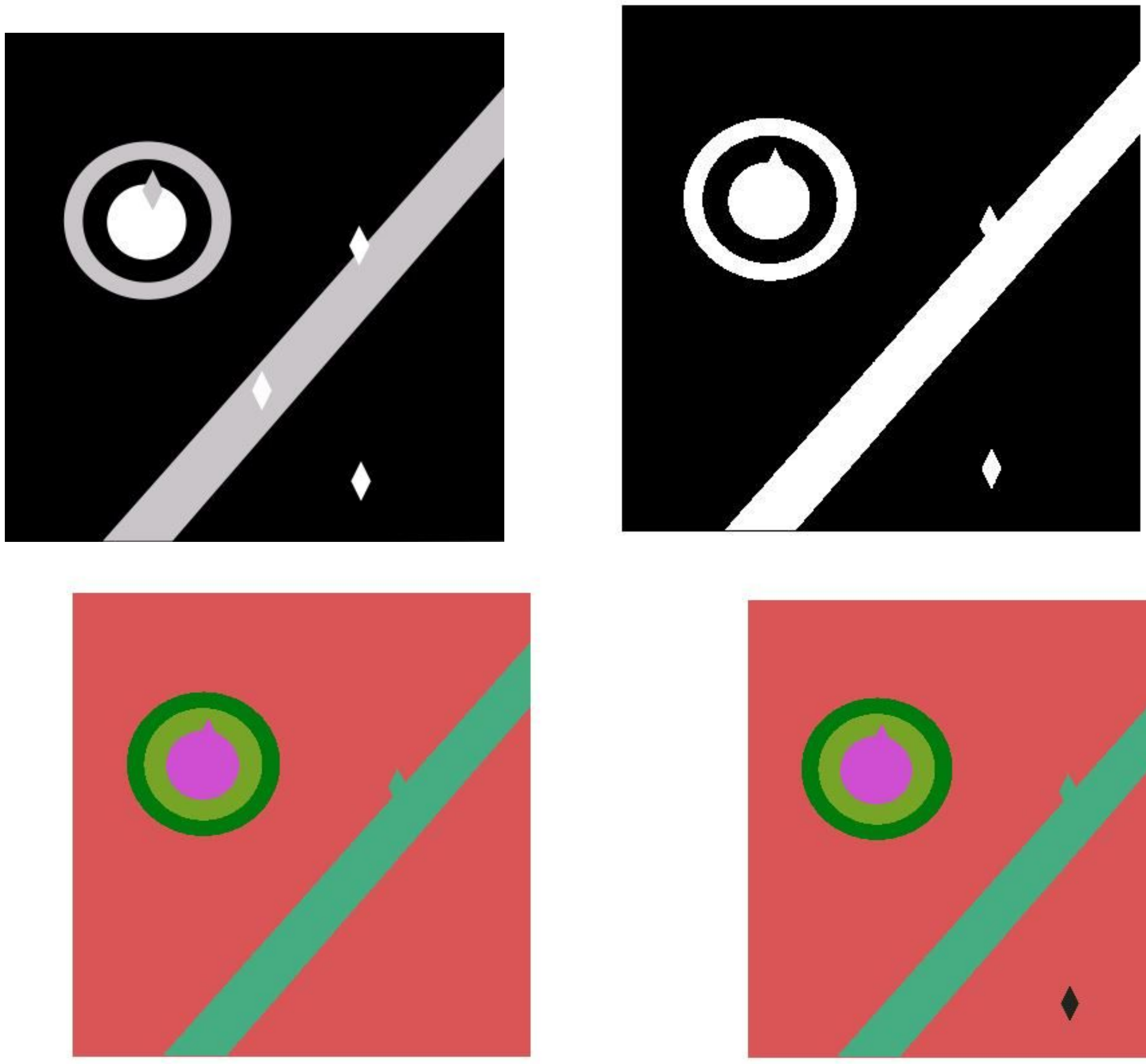
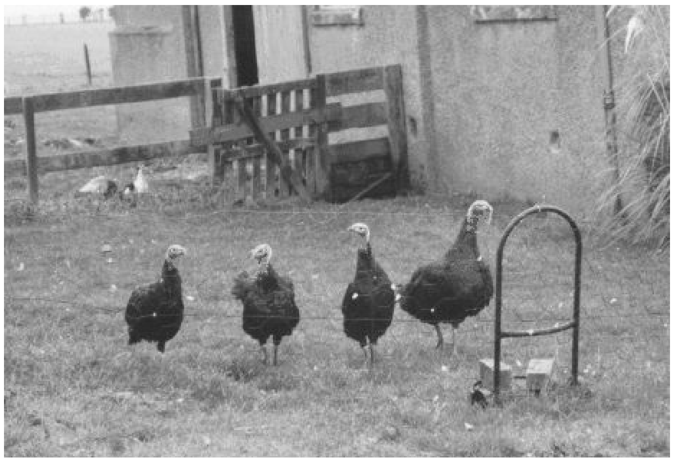
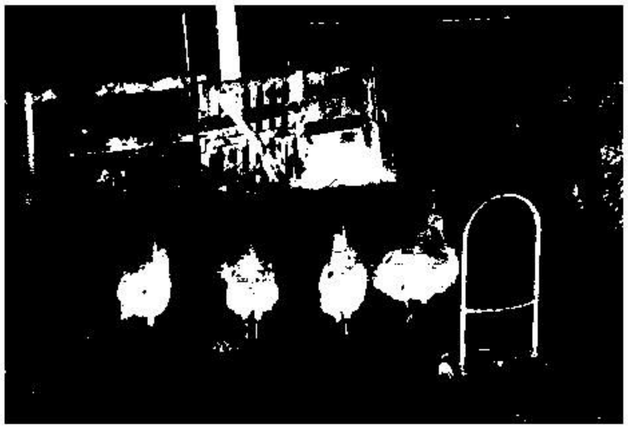
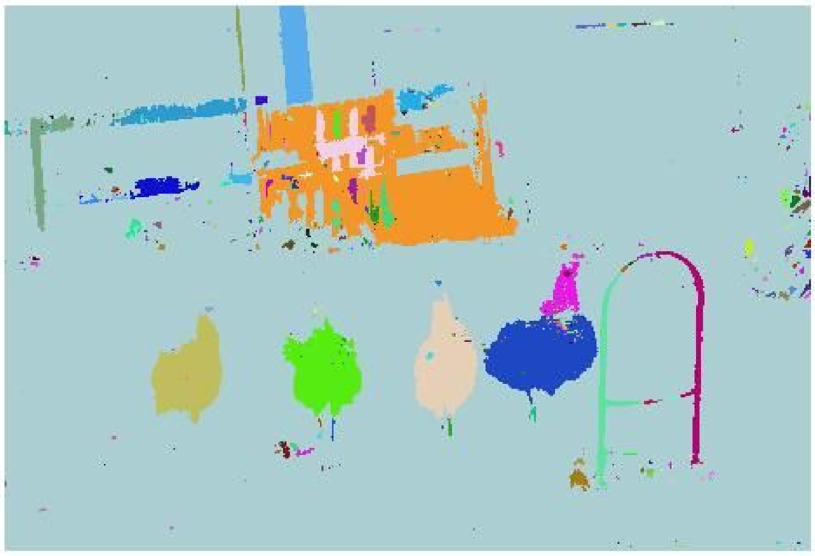


Figure19: Top Left: image:topo.jpg' - Original greyscale image'
Figure12: Top Right: Dual threshold of veg.tif. Min threshold: 150, max threshold: 255
Figure 21: Bottom Left: Connected components on the topo.jpg image
Figure 22: Bottom Right: Denoised topo.jpg image. Threshold: 200



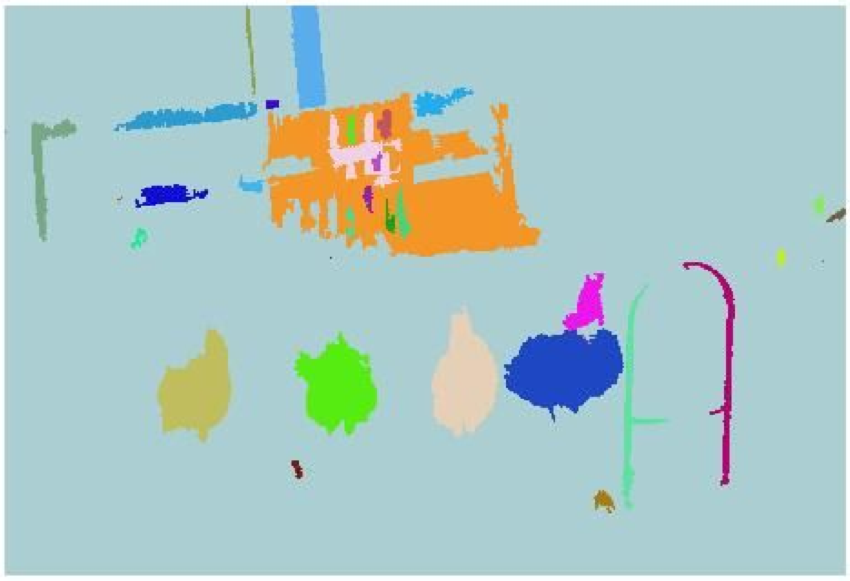


Figure23:Top Left: image:turkeys.tif’’
Figure24:Top Right: Dual threshold of turkeys.tif. Min threshold: 10, max threshold: 90
Figure 25: Middle: Connected components on the turkeys image
Figure 26: Bottom: Denoised turkeys image. Threshold: 20

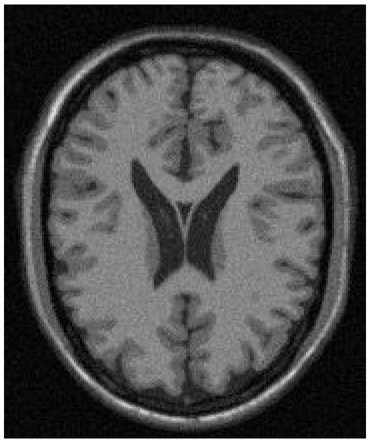
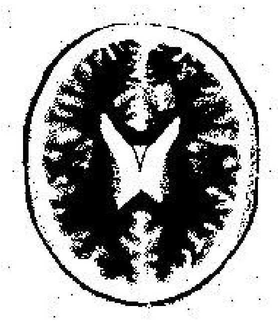
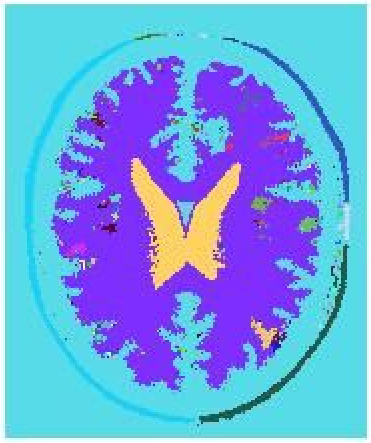
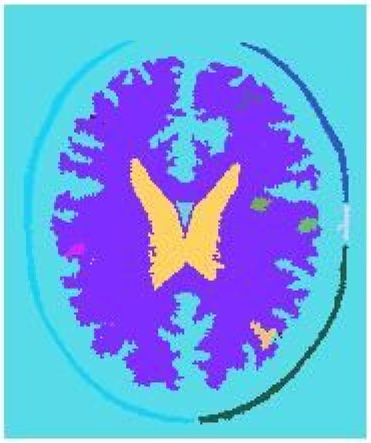





Figure27: Top Left: image:brain.jpg''
Figure28: Top Right: Dual threshold of brain.tif. Min threshold: 0, max threshold: 100
Figure 29: Bottom Left: Connected components on the brain.jpg image
Figure 30: Bottom Right: Denoised brain.jpg image. Threshold: 30

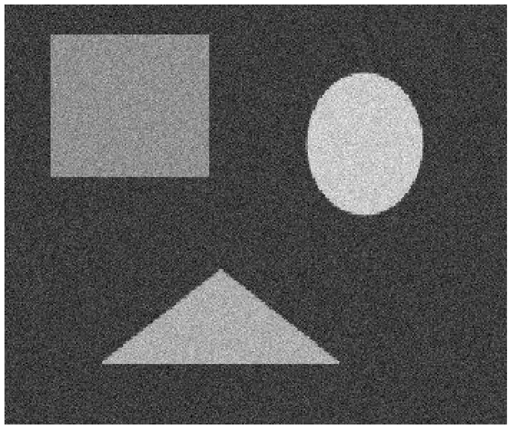
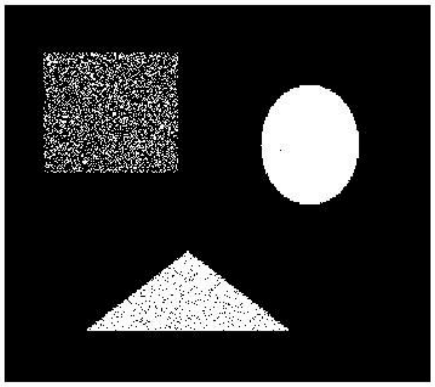
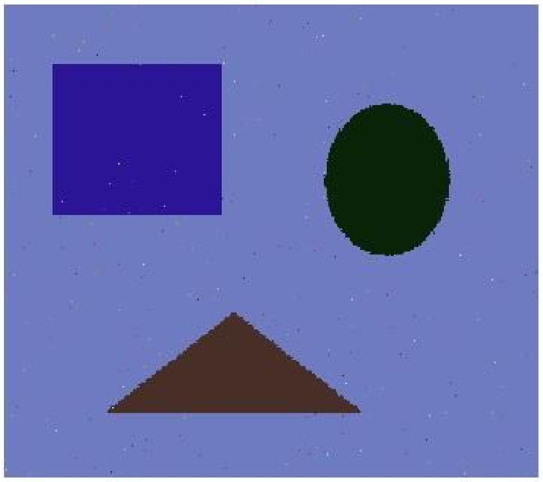
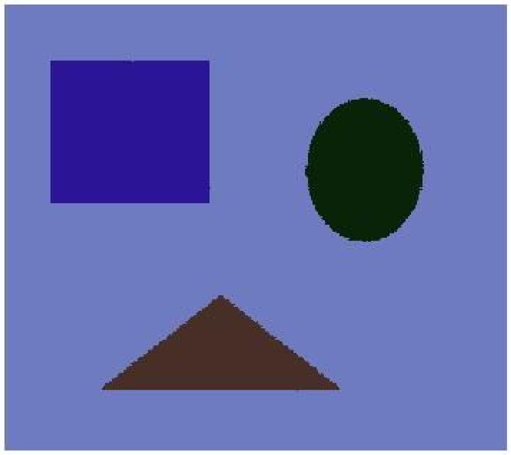
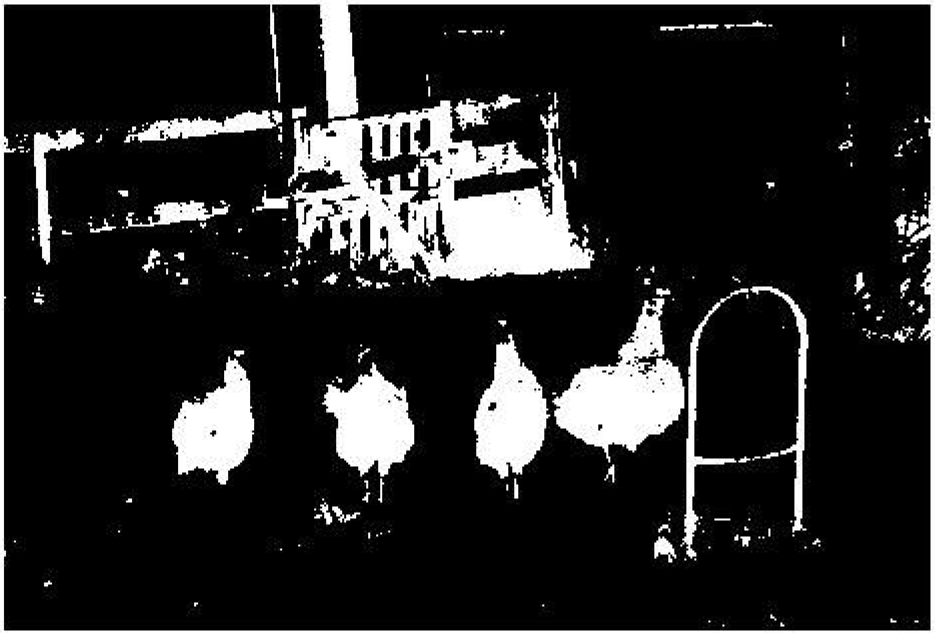
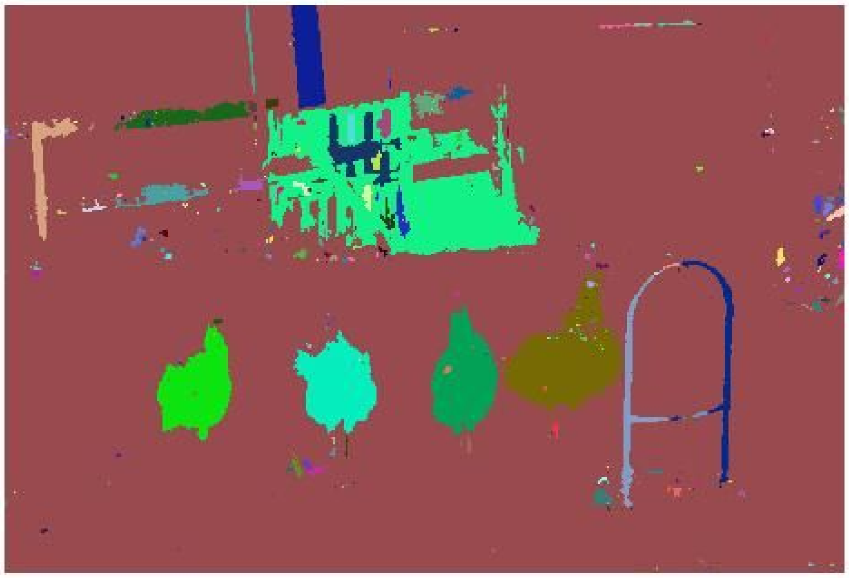





Figure31: Top Left: image:shapes_noise.tif’’
Figure32: Top Right: Dual threshold of shapes_noise.tif. Min threshold: 0, max threshold: 100
Figure 33: Bottom Left: Connected components on the shapes_noise image
Figure 34: Bottom Right: Denoised shapes image. Threshold: 30





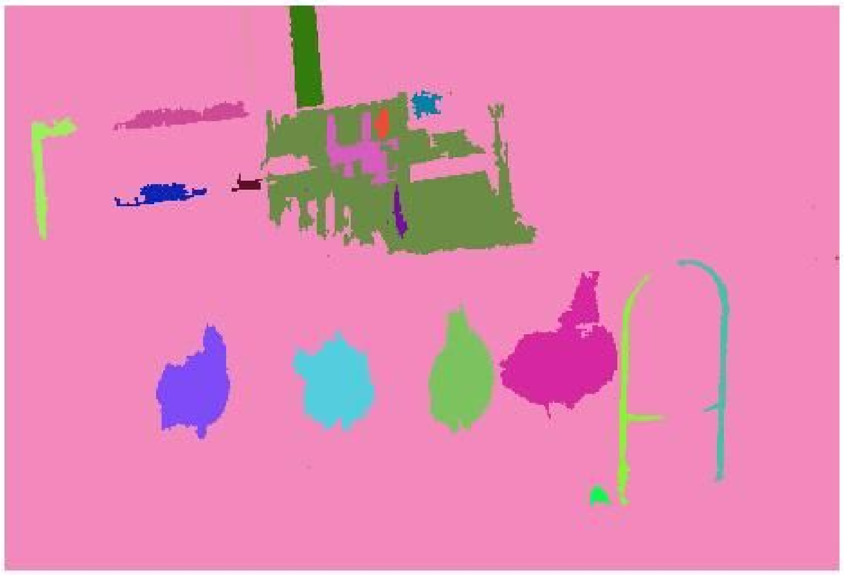


Figure35: Top: Dual threshold of turkeys.tif. Min threshold: 0, max threshold: 100
Figure36: Middle: Connected components on the turkeys image
Figure 37: Bottom: Denoised turkeys image with Threshold: 30

4. Motion detection:

The motion detection between two images has been implemented using the connecting components and topological denoising methods: ‘topological_denoise.m’ and ‘connected_component.m’ file along with a few lines of code in the main.m file.

The denoising technique helps in removing the unwanted pixels which helps in identifying the dog in this picture.

Challenges faced were, if the moving object has the same intensity as the background which it obscures, it is difficult to accurately determine the moving object since the subtraction intensity at that pixel is 0. This can be seen at some portions where the dog’s tail is disconnected. Also, there are additional regions with some change in sunlight which is mistaken as a motion detection (because of the change in sunlight made the intensity to change).

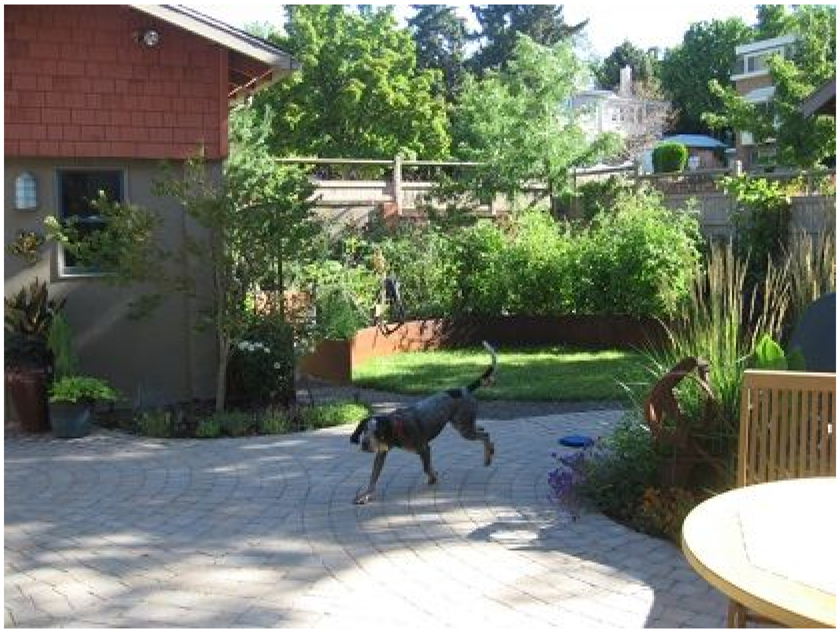
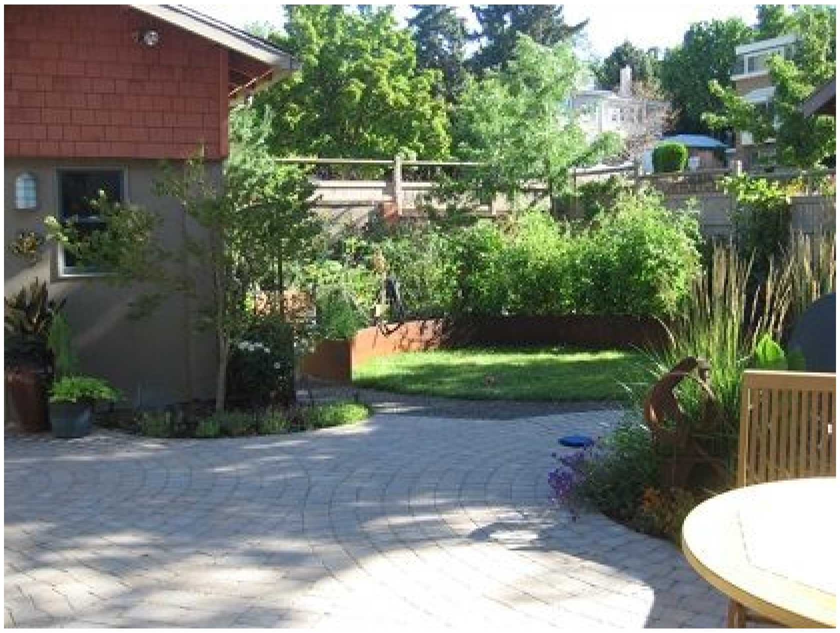



Figure 38: Top: houndog1.tif, bottom: houndog2.tif

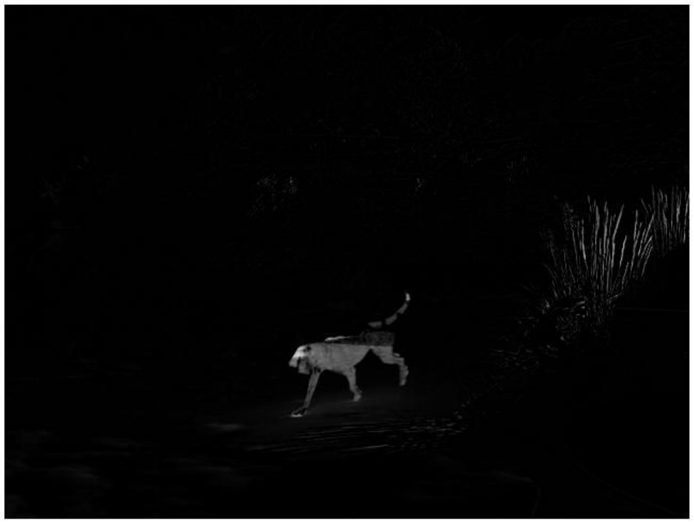


Figure 39: houndog.jpg (difference of houndog1.tif and houndog2.tif) grayscale image.

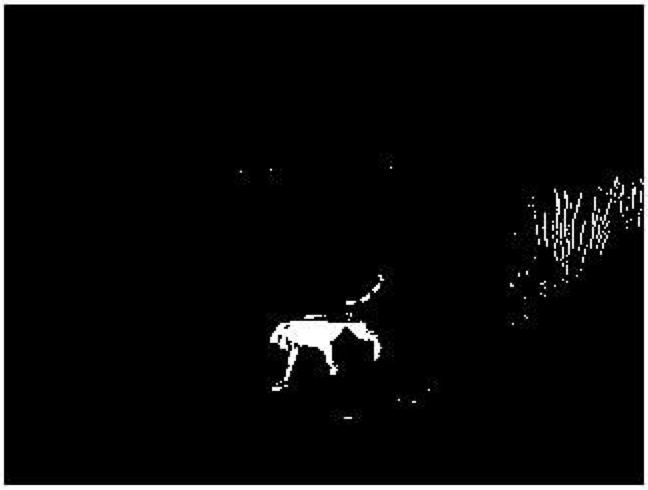


Figure 40: houndog difference image as binary image Min & max Threshold: 40, 255

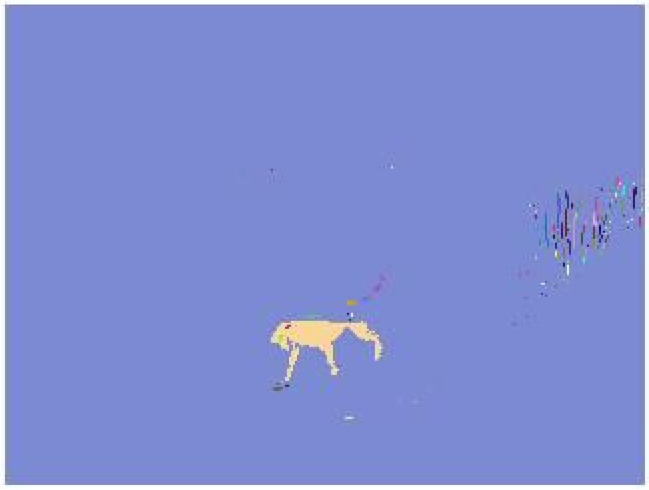


Figure 41: houndog difference image’s connected components

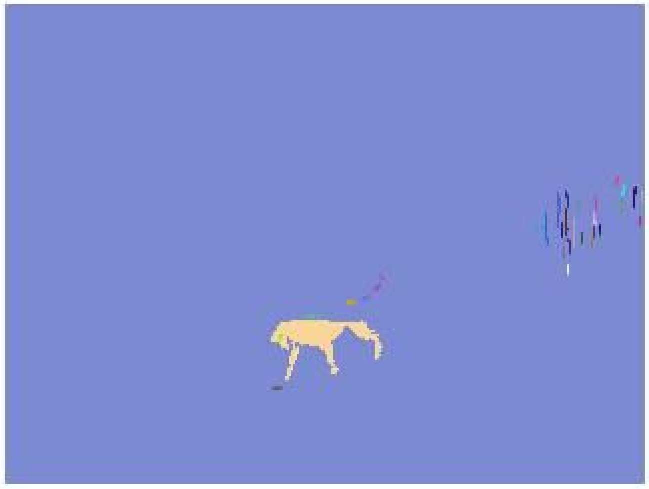


Figure 42 houndog motion captured after denoising

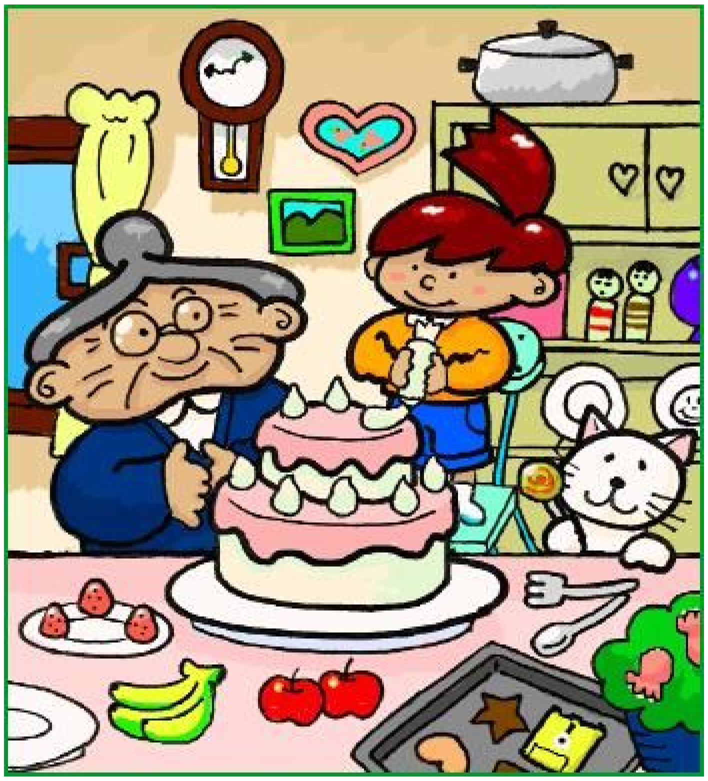
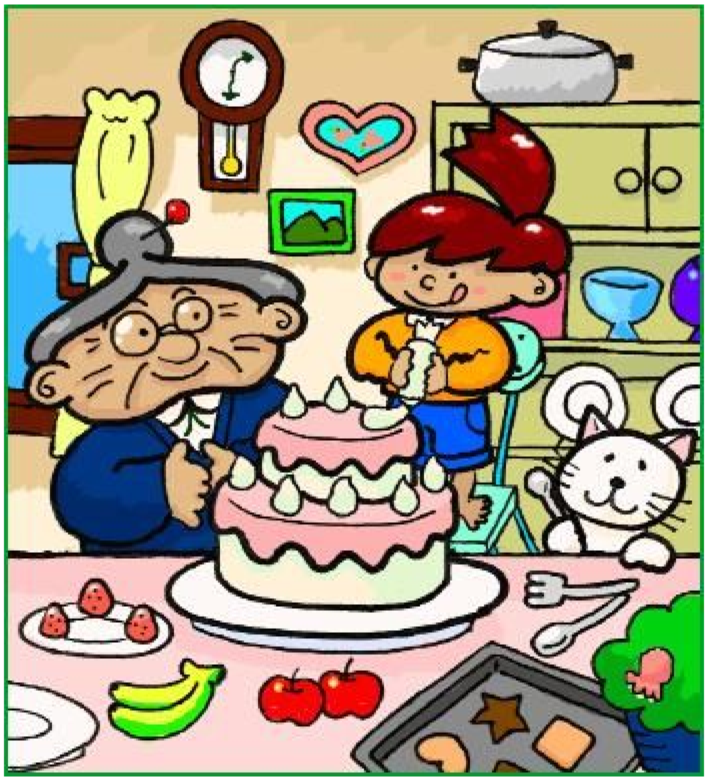



Figure43 Left: Cake1.jpeg, Figure 44: Right: Cake2.jpeg

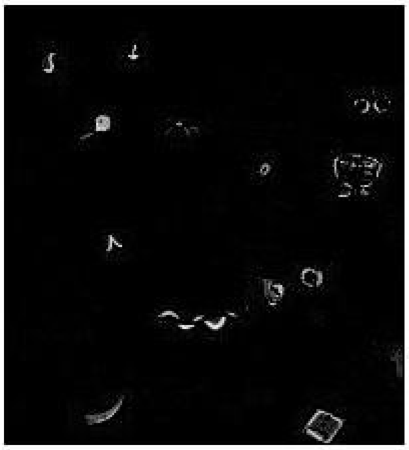


Figure 45: cake.jpg (difference of cake1.jpeg and cake2.jpeg) grayscale image.

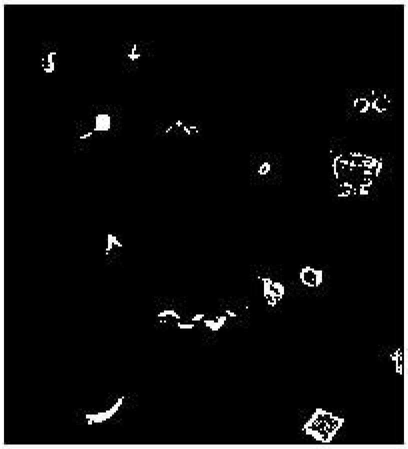


Figure 46: cake difference image as binary image Min & max Threshold: 40, 255

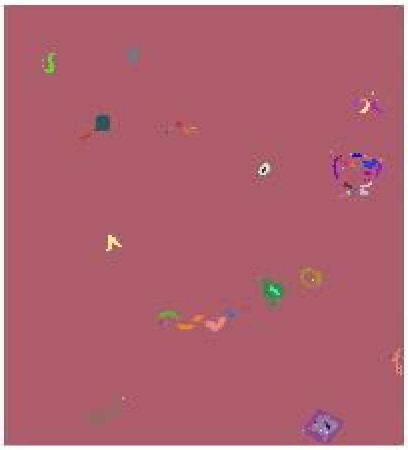


Figure 47: cake difference image's connected components

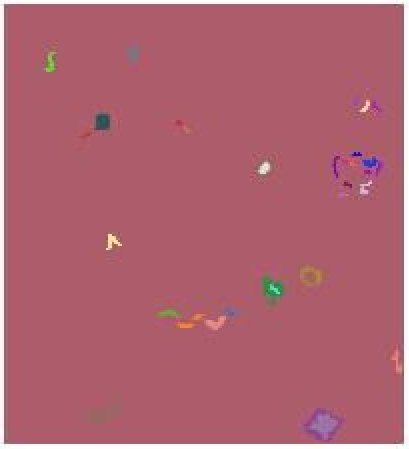


Figure 48 cake motion captured after denoising(i.e difference in both images captured)