Connor McCurley

EEE 6512 **Homework 5 Due November 3, 2018** Fall 2018

Part I MATLAB Programming

For this question, we were provided the following image (Fig. 5) and asked to separate the foreground (lake) from the background (rest of the image).

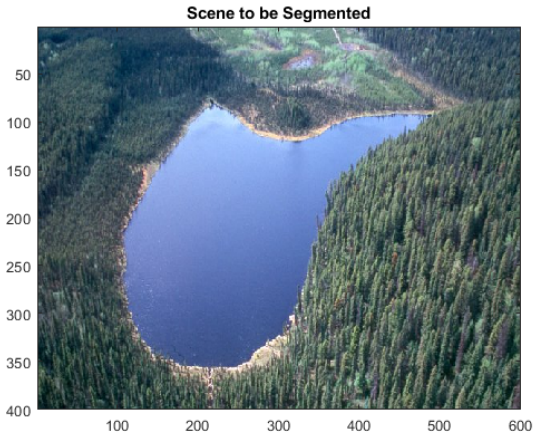

Figure 1: RGB scene to be segmented

To accomplish the segmentation, I first applied Gaussian smoothing to each color channel with $\sigma = 2$. This makes sense to do since we don't want to consider all of the high-frequency "noise" scattered throughout the image, such as in the trees. Since we only cared about the border regions between the foreground and background, I then applied a Sobel edge detector in each color channel to find the first derivative edges. The histograms fro Otsu's method only considered these edge pixels. The results of the edge detection are shown in Fig.2

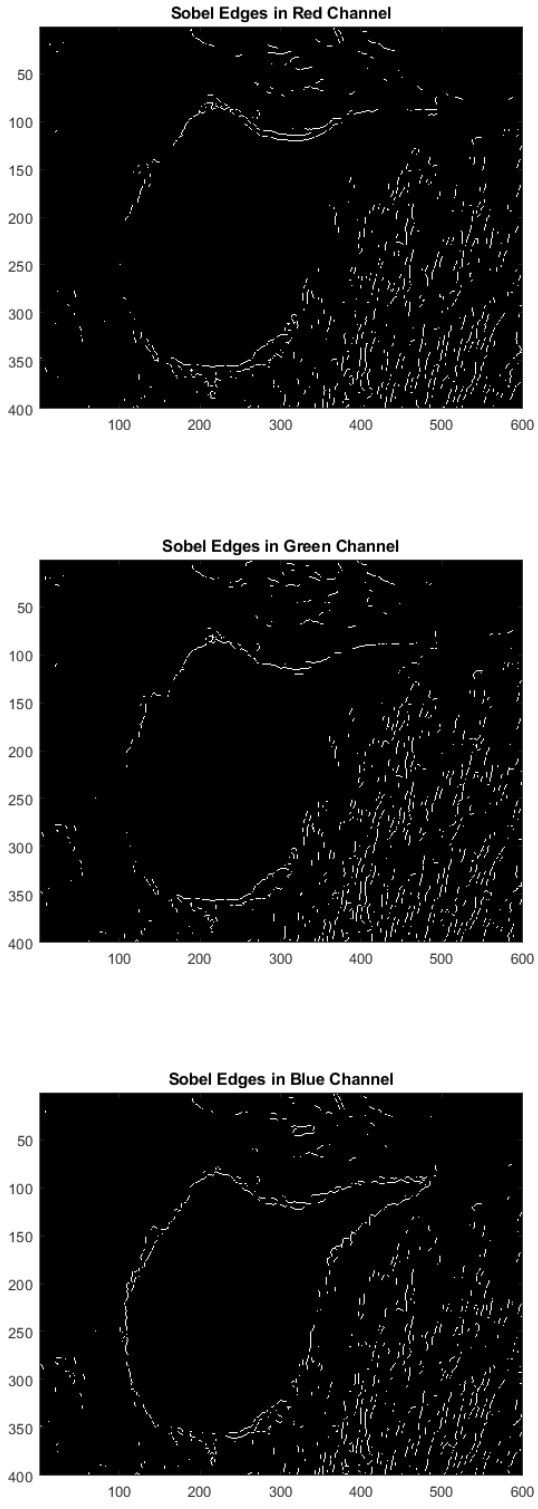

Figure 2: From top to bottom: Sobel edges detected in the red channel, green channel, and blue channel

The normalized histogram was found in each color channel using code written for HW02, and only considered the pixels found from the Sobel Edge detection. The normalized histograms of edge pixels are shown in Fig.

3.

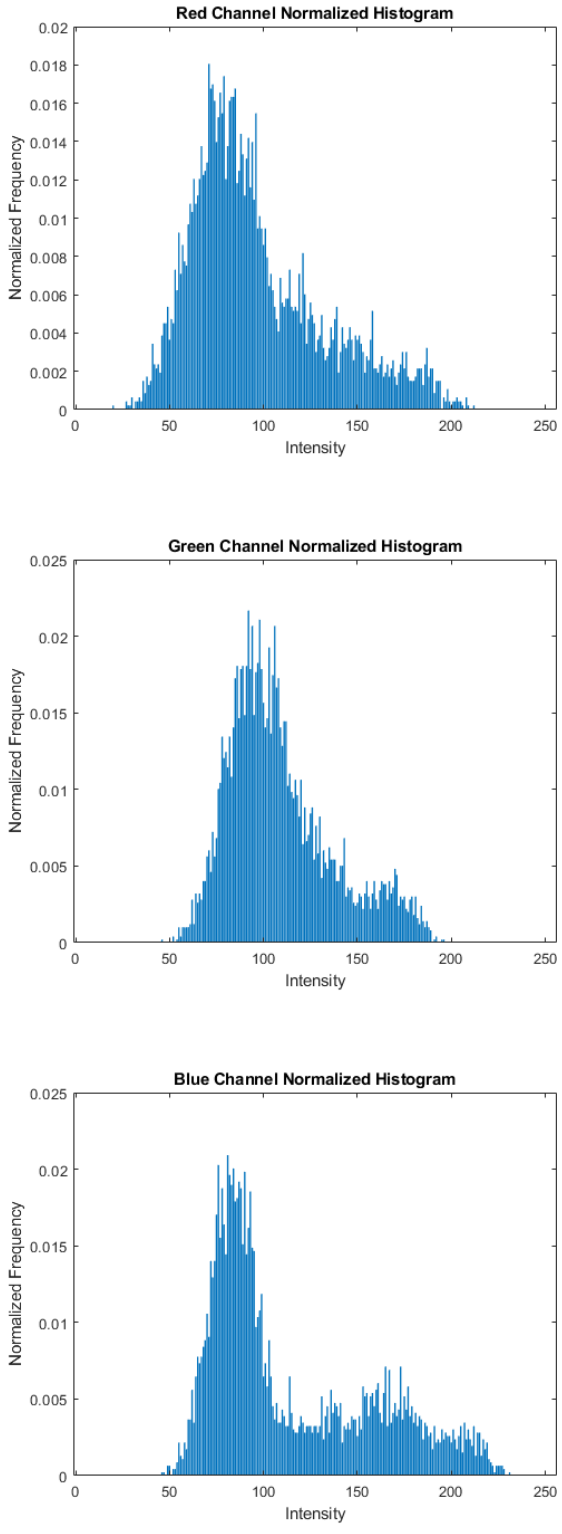

Figure 3: From top to bottom: Normalized histogram of Sobel detected edge pixels detected in the red channel, green channel, and blue channel

It is clear from the normalized histograms that two modes exist in both the green and blue channels. This was promising for segmentation using Otsu's methods since it effectively finds the optimal boundary between two modes in a distribution.

Otsu's method was then performed in each color channel following equations 10-49 through 10-62 in the textbook. The function I wrote, *otsu.m* took in the rgb scene to be segmented, performed Gaussian filtering, Sobel edge detection, and Otsu's method in each channel. It then returned a three channel image containing the binary Otsu segmentation for each color channel. The segmentation results using Otsu's Method for each color channel are shown in Fig. 4.

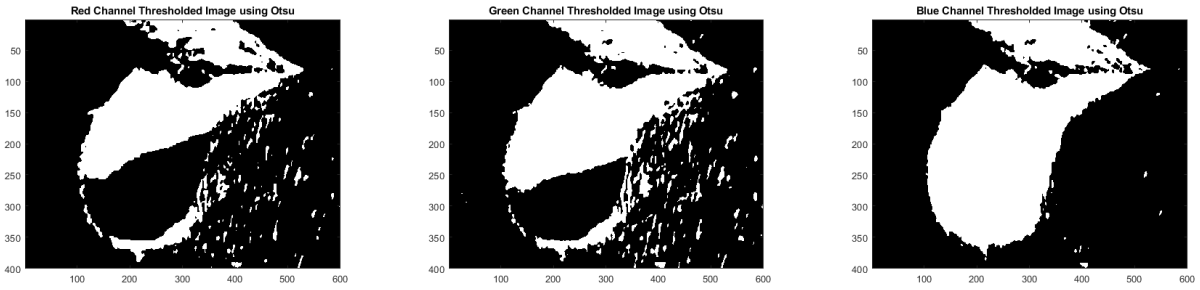

Figure 4: From left to right: Normalized histogram of Sobel detected edge pixels detected in the red channel, green channel, and blue channel

Although it was clear that the blue channel could have been used solely to extract the foreground, we wanted this to be an automated process. Therefore, the channels needed to be aggregated to utilize the foreground information contained in each. This was accomplished using a logical OR operation between each channel. The resulting image from the combination is shown in figure

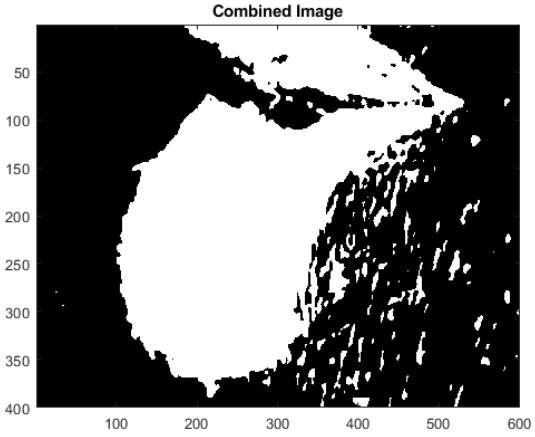

Figure 5: Binary image formed from the logical OR operation between rgb color channels segmented with Otsu's method

The resulting image contained a significant amount of "noise" outside of the foreground. To remove these spots, morphological opening (erosion then dilation) was utilized to rid the image of spots smaller than a 10x10 disk structuring element (Fig. 6).

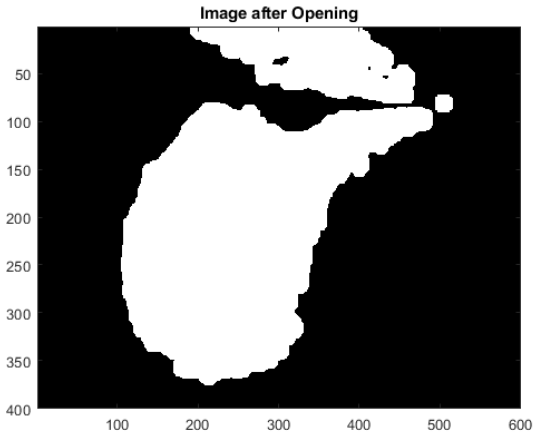

Figure 6: Binary image after opening with a 10x10 disk structuring element

This effectively removed the noisy areas outside of the foreground. Next, morphological closing (dilation then erosion) with a 15x15 disk structuring element was performed to close the small holes within the foreground (Fig. 7). This was necessary before boundary extraction so that no false object boundaries would appear within the foreground.

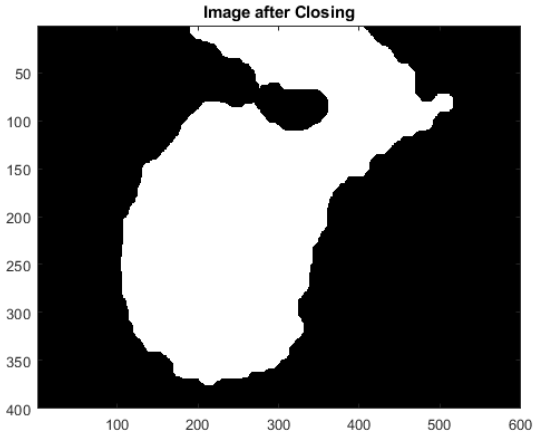

Figure 7: Binary image after closing with a 15x15 disk structuring element

The las step was to extract the boundary of the foreground. This was done by subtracting the binary image eroded with a 3x3 square of ones structuring element from the binary image. The eight neighborhood of each pixel on the boundary was set to 1 for visualization. Results of the boundary extraction are shown in figures 8 and 9.

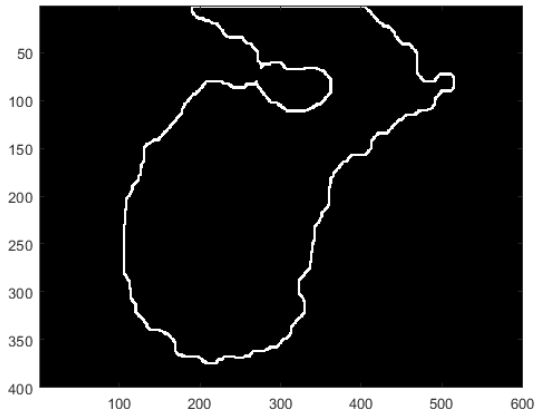

Figure 8: Boundary mask found by subtracting the closed image eroded by a 3x3 square of ones from the closed image. The boundary was widened by setting the eight neighborhood of the boundary pixels to 1's.

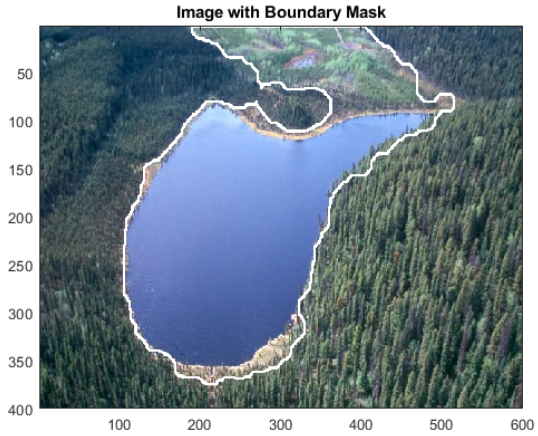

Figure 9: Original scene overlain with the computed boundary mask.

Part II Extra Credit

Accompanying code is provided in $mccurleyHW04.m$, $removeInterference.m$, $pseudoColor.m$, and $colorSegmentation.m$