Ehsan Ayoubi
CAP 5415 – Computer Vision
Programming Assignment 5

Image Segmentation

This report will cover my programming assignment 5 for CAP 5415 (Computer Vision).

Implementation

The program implements the following methods:

1. **Grayscale loading** – load an image and represent it as a NumPy matrix.
2. **Histogram computation** – calculate the intensity distribution of pixel values.
3. **Class probability calculation** – compute cumulative probabilities for each potential threshold.
4. **Mean intensity calculation** – determine the mean intensity for foreground and background classes.
5. **Between-class variance** – compute σ^2_b to measure separation between classes.
6. **Threshold optimization** – find the threshold that maximizes between-class variance.
7. **Binarization** – apply the threshold to segment the image into binary output.

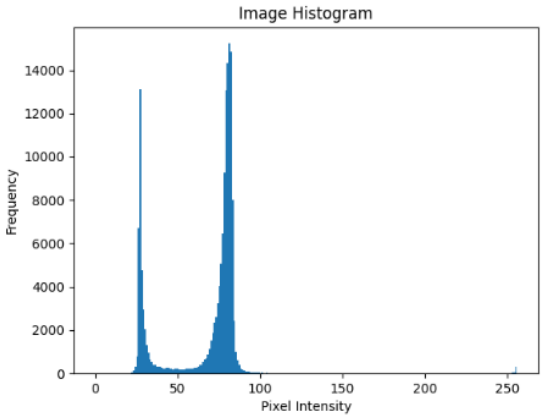
Note: a user can manually select their own threshold for testing. More details and step to replicate can be found on this repository.

The **Otsu Thresholding** algorithm followed the following pseudo-code:

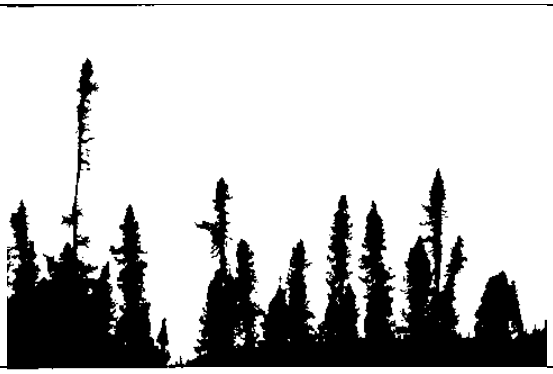
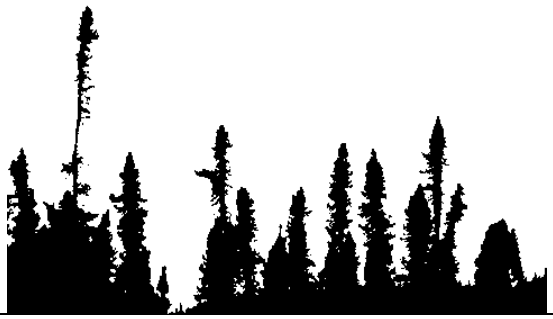
Input: Histogram H_I, max intensity $G_{\max}$, step size T_0
Initialize: $u = T_0$, $T = u$, $S_{\max} = 0$
while $u < G_{\max}$ **do**
    Step 1: Compute Class Probabilities (P_1, P_2)
        $P_1 = \sum_{i=0}^{u} p(i), \quad P_2 = \sum_{i=u+1}^{G_{\max}} p(i)$
    Step 2: Compute Class Means (μ_1, μ_2)
        $\mu_1 = \frac{1}{P_1} \sum_{i=0}^{u} i \cdot p(i), \quad \mu_2 = \frac{1}{P_2} \sum_{i=u+1}^{G_{\max}} i \cdot p(i)$
    Step 3: Compute Between-Class Variance
        $\sigma_b^2(u) = P_1 P_2 (\mu_1 - \mu_2)^2$
    Step 4: Update Optimal Threshold
    if $\sigma_b^2(u) > S_{\max}$ **then**
        $S_{\max} = \sigma_b^2(u)$
        $T = u$
    end if
    $u = u + T_0$
end while
Return T

Results

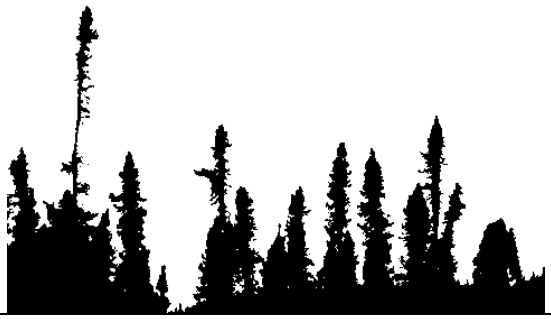
Image 238011.jpg



This image was one of the easier to segment because there was a very clear distinction between the foreground and background. We can see two clear peaks in the histogram, indicating the image consists of majority two colors.

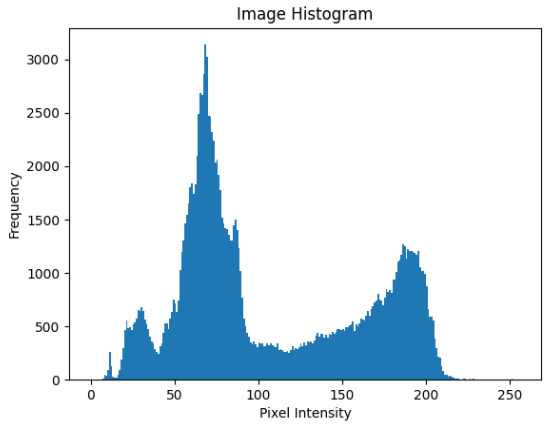
Manual Threshold 50	
Manual Threshold 55	

Results

Manual Threshold 60	
Otsu Threshold 54	

Results

Image 41004.jpg



This image was significantly more difficult than the previous, since it contained many more different shades of different colors. For example, the grass in the front was a darker shade than the rest of the grass, and certain furs on the animal was a brighter color than the rest, making it very difficult to perfectly segment a specific feature.

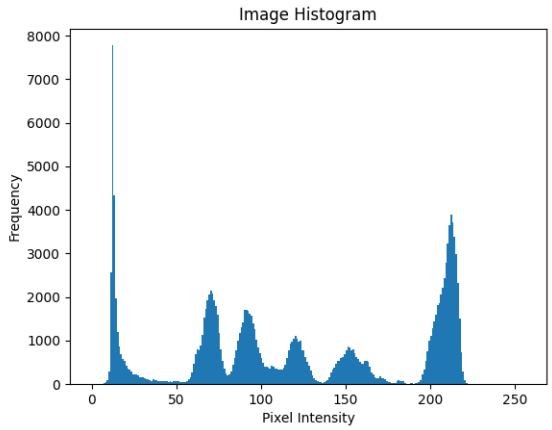
Manual Threshold 115	
Manual Threshold 120	

Results

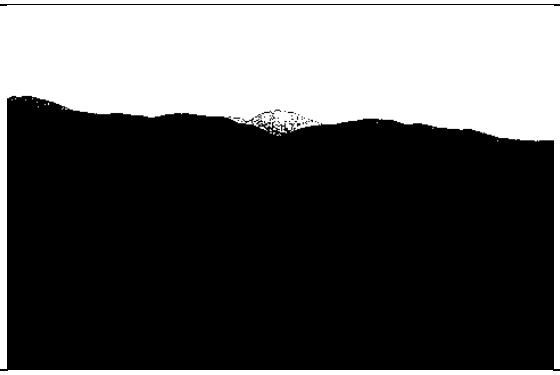
Manual Threshold 130	
Otsu Threshold 119	

Results

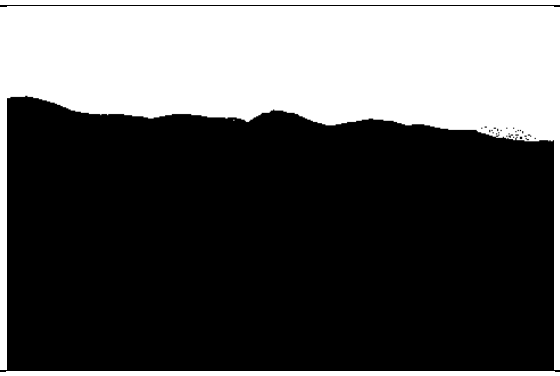
Image 55067.jpg



I believe this image is a perfect example of “Otsu thresholding doesn’t always work.” This image arguably gives over 6 different segments, and for best results, the user would have to manually indicate where they want to segment. For example, to separate the sky from the mountains, the ideal threshold would be at about 175, whereas Otsu predicted 131.

Manual Threshold 170	
Manual Threshold 175	

Results

Manual Threshold 180	
Otsu Threshold 131