UNIVERSITY OF NEW SOUTH WALES

COMP9517: Computer Vision

2023 Term 2 Assignment

Zeal Liang
z5325156

June 19, 2023

Task 1: Otsu Thresholding

In this task, I used the Otsu thresholding method for image binarization. Otsu thresholding is a method to automatically determine the threshold value and it selects the best threshold value by maximizing the inter-class variance for image segmentation.

The following are the methods and steps I used:

1. Calculate and normalize the histogram: Since different images may have different number of pixels, the size of the histogram may also be different. In order to make the histograms of different sizes comparable with each other, we need to normalize them. Normalizing a histogram is the process of dividing each histogram entry by the total number of pixels so that each entry has a value between 0 and 1. This way, we can compare histograms of different sizes with each other without having to worry about their different sizes.

2. Iteration over all thresholds: After initializing the variables for the best threshold and the maximum variance I iterate over all possible thresholds from 0 to 255.

3. Calculating pixel probabilities and weights and means for class 1 and class 2: For each threshold, I calculate the probability and weight of pixels below the threshold as class 1, and the probability and weight of pixels above the threshold as class 2.

4. Calculating inter-class variance: using the weights and means of class 1 and class 2, I calculated the inter-class variance. The between-class variance measures the degree of difference between the two categories, and a larger variance indicates a better segmentation of the categories. The formula for calculating the inter-class variance is given by:

$$\sigma_B^2 = p_0 p_1 (\mu_0 - \mu_1)^2$$

5. Update the optimal threshold and maximum variance: If the current interclass variance is greater than the previous maximum variance, I update the maximum variance and set the current threshold to the optimal threshold.

6. Apply optimal threshold: Based on the optimal threshold, I binarize the original image and set the pixels below the threshold to 0 (black) and the pixels above the threshold to 255 (white).

Task 2: Isodata Thresholding

In this task, I used Isodata thresholding method for image binarization. Isodata thresholding is an iterative based threshold selection method which minimizes the variance between two categories by continuously adjusting the threshold values.

The following are the methods and steps I used:

1. Calculate and normalized histogram: same as **Task 1**

2. Initialize variables: Initialize the threshold to 0 and set the new threshold to the histogram-weighted average.

3. Convergence iteration: Enter the iteration process and stop iteration when the current threshold value and the new threshold value are the same. Otherwise, continue to the next step.

4. Calculate the pixel probabilities and weights of class 1 and class 2: same as **Task 1**

5. Calculate the mean value of class 1 and class 2: same as **Task 1**.

6. Calculate the new threshold: By averaging the mean values of class 1 and class 2.

7. Judging convergence: If the current threshold and the new threshold are the same, the convergence condition is reached and the iteration is stopped. Otherwise, continue to the next iteration.

8. Apply the best threshold: same as **Task 1**

9. The pseudo code is as follows

Algorithm 1: Isodata Thresholding

Input: Image
Output: Threshold
Function `IsodataThresholding(`*Image*`):`
  Data: Select an arbitrary initial threshold t
  Data: Compute μ_0 and μ_1 with respect to the threshold
  Data: Update the threshold to the mean of the means: $t = (\mu_0 + \mu_1)/2$
  while *the threshold changes* **do**
    Data: Compute μ_0 and μ_1 with respect to the threshold
    Data: Update the threshold to the mean of the means: $t = (\mu_0 + \mu_1)/2$
  end
  Data: Upon convergence, the threshold is midway between the two class means
  return *Threshold*

Task 3: Triangle Thresholding

In this task, I used Triangle thresholding method for image binarization.The Triangle method is based on finding the threshold of the maximum distance which is the maximum distance between a straight line segment and the histogram distribution.

The following are the explanation and steps of the method:

1. Find the highest or lowest gray level: Based on the peaks of the histogram, the highest or lowest gray level is determined. The location of the highest or lowest gray level is determined by comparing the number of pixels on either side of the peak.

2. Calculate the slope and intercept of a straight line: Calculate the slope and intercept of a straight line using the number of pixels and the height between the peak and the highest gray level.

3. Find the threshold for the maximum distance: traverse the pixel values of the histogram from the range between the highest gray level and the peak. For each pixel value, calculate the distance between the straight line and the histogram and find the threshold value that makes the distance maximum.

4. Apply the best threshold: same as **Task 1**

5. The pseudo code is as follows

Algorithm 2: Find the highest or lowest gray level, depending on the peak

Input: Histogram, Peak
Output: GrayLevel
Function `GetGrayLevelPoint(`*hist, peak*`)`:
  if $\sum_{i=0}^{peak-1} hist[i] > \sum_{i=peak}^{255} hist[i]$ **then**
    for $i \leftarrow 0$ **to** 255 **do**
      if $hist[i] \neq 0$ **then**
        return i;
  else
    for $i \leftarrow 255$ **to** 0 **do**
      if $hist[i] \neq 0$ **then**
        return i;

Algorithm 3: Triangle Method for Threshold Computation

Input: Input image I
Output: Automatically computed threshold value T
Function `Triangle(`I`)`:
  Data: Compute the histogram H of image I using the function $histogram(I)$;
  Data: Find the peak (r_p, h_p) in histogram H using the function $argmax(H)$;
  Data: Find the highest gray level point (r_m, h_m) in histogram H using the function $GetGrayLevelPoint(H, peak)$;
  Data: Construct a straight line $l(r)$ from (r_p, h_p) to (r_m, h_m);
  Data: Initialize the maximum distance d_{max} to $-\infty$;
  Data: Initialize the threshold value T to 0;
  for *each gray level r* **do**
    Compute the distance $d(r) = l(r) - h(r)$ using the function $distance(l(r), h(r))$;
    if $d(r) > d_{max}$ **then**
      Update d_{max} to $d(r)$;
      Update T to r;
    end
  end
  return T;
return `Triangle(`I`)`;

Task 4: Compare Thresholding Techniques

Differences in results and interpretation:

The Otsu method and the Isodata method performed basically the same in these five images. These two methods do not perform well in the Algae image. This may be because the distribution of gray levels is more concentrated in this plot and it is difficult to find a clear peak to determine the threshold.

On the other hand the Triangle method performs poorly in the Nuclei and Rubik plots, although it has excellent performance in the Algae plot. This is because the Triangle method is based on the slope of the histogram and tries to find a ”steep” region in the histogram and use it as a threshold for segmentation.

This method is more suitable for images with a large concentration of pixel grayscale and is able to find the boundary between the foreground and the background more accurately.

To summarize:

1. Otsu method: It is suitable for images with significant gray level differences between the foreground and background. The Otsu method can provide accurate threshold segmentation when the distribution of gray levels in the image is relatively uniform or when there is an obvious bimodal distribution.

2. Isodata method: It is suitable for images with a more uniform distribution of gray levels. When there is no significant gray level difference between foreground and background, Isodata method can provide better segmentation results.

3. Triangle method: It is suitable for images with obvious boundaries and uneven distribution of gray levels. When there is a large asymmetry in the distribution of gray levels or an obvious single-peaked distribution, the Triangle method can accurately find the boundary between the foreground and the background.

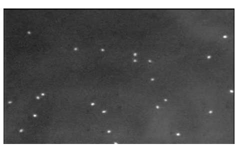
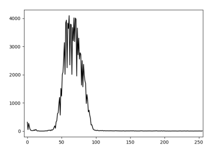
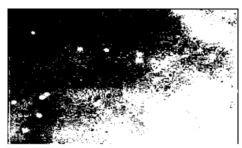
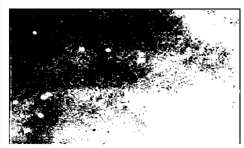
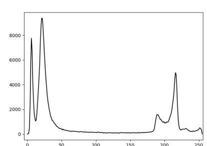
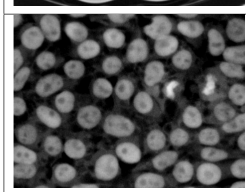
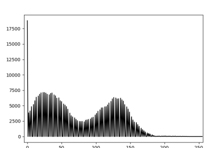
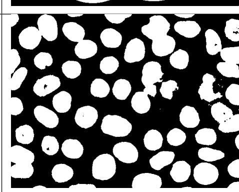
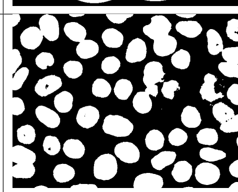
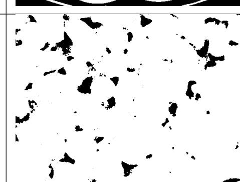
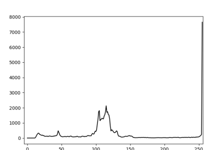
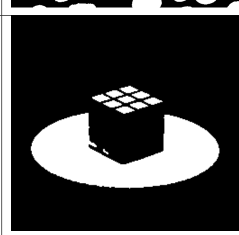
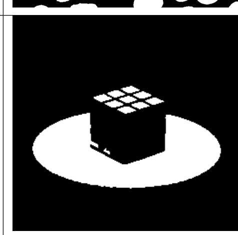
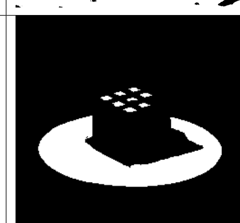
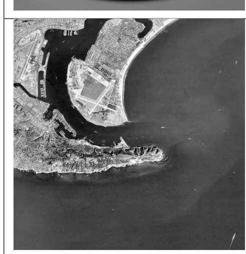
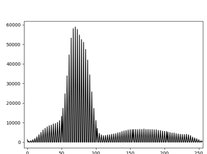
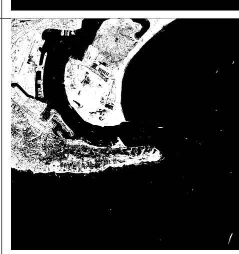
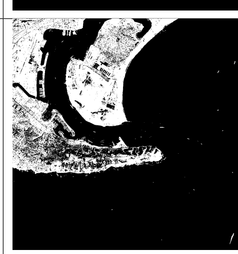
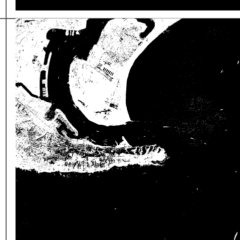
Original Image	Histogram	Otsu	Isodata	Triangle
				
				
				
				
				

Table 1: Comparison of different thresholding methods on various images.