Sikender Shahid
0981476
02/2019

Introduction

Binary image operations are much more computationally less demanding than lets say dealing with color dimension. In Binary images, pixel intensities are converted to either of the two designated values. More often it is either ‘1’ and ‘0’, or ‘255’ and ‘0’. However this can be decided for the application on had. The pixel intensity value is decided based on a calculated threshold. The aim is to promote a divide with respect to contrast.

In this assignment, the goal is to identify all the cells in a image. Be able to figure out a count number, area, and center of each pixel. To achieve that, be able to plot a histogram with respect to the image under analysis. When the final image is developed, be able to compress the image and save it. To view the image at a later time, be able to decompress the image.

Binary Image

Methods

The first step in developing binary images is to calculate the frequency of each pixel intensity within a bit value range from 0 to 255. This is represented by a histogram. Initially create a histogram array based on the bit value range with all elements set to zeros. A scan is performed from first pixel to the end, then using the pixel value as an index to the histogram array, increment the count designated in the location. This histogram can be normalized to make other computation less complex. However this was not performed.

To be able to binarize the image, a threshold is needed as discussed in the introduction. The goal is to calculate an optimal threshold for mapping each pixel intensities to two possible values. The threshold can not be simply used by averaging method. Analysis on the histogram is performed. Looking at the histogram there is no centered peek. The spread is good, however the peak resides further to the bright intensity range. The brightest pixel value is 255 based on the bit range 8. The darkest pixel value is 0. the method used incorporates calculating an expectant of a pixel range with respect to the histogram array. This is performed by setting the initial threshold value to the average of the brightest pixel and the darkest pixel. The histogram array is divided based on the threshold then the expectant of the two arrays is calculated. Taking the the average of the two expectant values is the new threshold. Looking at the first order change of each expectant is the deciding factor if this new threshold value is to be decided. When the change of the two expectant no longer occurs then the last threshold hold value is the optimal threshold value. Helper function, expectation is written since this is a repeating operation. Here we normalize the histogram and calculate the probably of each pixel value. Then using the probability of the pixel values, the most occurring average value from probability theory is realized.

Once the optimal threshold is realized. To map each pixel value, a scan is performed checking if the value should be either of the two values. In this assignment to be able to view the image clearly as the output, the following two values were used. Max value of 255 for dark pixels, and 0 for the bright pixels. This is also a inverting operation. In the original image the cells are of the dark range while the background is light. If the cell is found which is below the threshold limit then that pixel is mapped to be bright with the value of 255. if a background is found in the original which is above the pixel threshold then that pixel is mapped to be dark pixel with value of 0.

Findings & Observation

Threshold is critical part in segmenting or binarizing an image. This operation in a perfect contrast scenario can be computed through averaging. However in most cases there is an addition of noise along with pixels varying over a range. When tested with the averaging method, the image that appeared happened to be mostly dark. This means that the image must have more surface area coverage by the background since the image was also inverted. Cells intensity is not fixed to one value, which consists of a range. Considering these factors, dictated that the averaging method is wrong.

Using the expectant to figure out the average running pixel intensity is more optimal. This is because the pixel ranges were divided and rechecked often to compute the optimal threshold value. This is different from calculate just the average over all the pixel values.

Road blocks

Deciding whether binary values 1s and 0s to be used versus 255s and 0s. To be able to visualize the change its better to use 255s and 0s as the pixel intensity value. To perform bit-wise operations its is better to use 1s and 0s as the pixel intensity values. In this assignment the code used allowed for both use cases, where bit-wise operations were performed and being able to visualize the image without overhead of performing double the raster scans discussed in Cell Counting section.

Cell Counting

methods

In counting the cells, the following were the steps that are needed to be performed. Each cell is considered blob, where the couting of number of blobs is needed. Each blob is marked by a region number. For each blob the statistics such as area, and center pixel is to be determined. Lastly a image needs to be created which marks each blob with blob number and area.

For assigning each blob a number and determining the count of pixel the blob contains, the following method was performed. First a 5-pixel cross window is created. This is considered the window mask. Using this window a padding performed on the image. Then the window is scanned through the image looking for three things. A new region, and two specific positioning of a neighboring pixel from the center. As the scan progresses, a binary operations are performed on the image with the masks. First to extract the region that falls under a blob classification, then checking if it is a neighboring pixel or a new region. If it is a new region then a region counter is incremented, memory is allocated, and the current center pixel's location is stored in a temporary array. If a new region is detected or a specific neighboring pixel, then a check to see if this position top pixel or diagonal pixel has already been accounted for. If the top or top right diagonal pixel has not been accounted for then the temporary storage of pixels is placed within a dictionary. This dictionary keeps track of the region number and all the positions that are within the region. When checking for the top or right diagonal pixel's previous recording, a loop is performed on a reverse sorted dictionary elements.

The statistics were trivial computation. The area is the number of pixel in the region, and the center pixel is the median of the pixels recorded for a region. The area is in unit pixel, and the center position that is the centroid is approximated to the median.

Marking the pixel region is performed as a scatter plot. Each centroid for each region number is used as the position to mark on the blobs. The plot is returned as a numpy array element.

Findings & Observation

When using the 5-pixel cross window operations to check for new region or neighboring region, the elements of the lower and left from the center pixel are neglect-able. This is with respect to the window raster scan, because we want to be able to capture pixel locations of lower bound for a blob. An Ignore mask is created for the 5-pixel cross window mask.

Road blocks

Capturing all true pixel regions for a blob. To correctly be able to capture the true pixel regions, fine tuning the algorithm took most of the time.

Plotting road blocks with respect to returning the correct formatting so the top level source code can save the image.

Compression

Findings & Observation

This compression algorithm dealing with binary images was lossless. No bit chopping or approximation is performed. There are either 255s or 0s in a location. A scan is performed and counting of the region pixels.

Road blocks

There were no road blocks to this algorithm.

Difficulties with understanding the problem

Different forms of centroid calculation on the non homogeneous structures.