CSC3014 Computer Vision

Lab 3: Thresholding and Region Growing

A. Global Thresholding

This can be considered as the simplest approach in image segmentation. All we need to do is to select a threshold (or multiple thresholds) to segment out the region-of-interest. This is considered as a pixel-based approach because it does not take into consideration the contextual information of the image. The most important part of this approach is to select the appropriate thresholds. This can be done by plotting the histogram and then try to select a threshold from there. The general idea is to select a point at the middle of the valley between two peaks.

To perform segmentation using thresholding, you can choose to write your own code or just use the cv2.threshold() function. As shown below is a sample code that demonstrates how to use the function.

```
import cv2                                                    #import OpenCV
import numpy as np                                            #import numpy package
from matplotlib import pyplot as pt                           #import pyplot for plotting figure

img = cv2.imread("coins.bmp", 0)                              #Read coins
retVal, threshImg = cv2.threshold(img,110,1,cv2.THRESH_BINARY) #Apply thresholding

pt.figure()                                                   #Create a figure window
pt.imshow(threshImg,cmap="gray")                              #Display the output in grayscale
pt.show()                                                     #Display the output
```

The function will return two outputs. The first output is the threshold value that is used to compare with all the pixels, while the second output is the binary image after thresholding is applied. This first output is only useful when we are using the Otsu's method to perform segmentation. Because Otsu's method will automatically calculate the threshold that should be used, this allows us to know the exact threshold value that was applied during thresholding. If we are not using Otsu's method, then it will just take the same threshold value that we have specify.

There are four input parameters to this function. First one is the input image. Second one is the threshold value that will be used during thresholding. Third one is the value that will be used to represent pixels with intensity higher than the threshold. The last one is the different styles of thresholding. In this case, we will only be using the standard style that we have covered, cv2.THRESH_BINARY. You may refer to the OpenCV documentation if you would like to know more about the other styles that you can choose from.

> **Exercise**
>
> Given an image "sudoku.bmp", try to segment out the text by using global thresholding. Do not worry if your output is not perfect. We will try to fix that by using adaptive thresholding in the next section.

B. Adaptive Thresholding

In cases where the illumination is not uniform, then very often we will not be able to obtain a "good' output by using thresholding. To overcome this issue, we can use adaptive thresholding that will apply different thresholds for different parts of an image. The threshold to be used for each part is determined based on the local statistics (intensity value of surrounding pixels). To do this, we can directly use the cv2.adaptiveThreshold() function from OpenCV, as shown in the sample code below.

```
threshImg = cv2.adaptiveThreshold(img,1,cv2.ADAPTIVE_THRESH_MEAN_C, cv2.THRESH_BINARY,3,0)
```

There are six input parameters to this function. Similarly, the first one is the input image. Second one is the value that will be used to represent pixels with intensity higher than the threshold. We are not required to specify the threshold value because the function will automatically calculate the threshold value based on the surrounding pixels. Third one is to select how the threshold value is calculated. In this case, we are using cv2.ADAPTIVE_THRESH_MEAN_C, which means the threshold value is the mean of the surrounding area. In addition to this, you may also choose to use cv2.ADAPTIVE_THRESH_GAUSSIAN_C, whereby the threshold value is the weighted mean of the surrounding pixels. Fourth one is the style of thresholding, and we will just use back the standard style, cv2.THRESHOLD_BINARY. The fifth one is the block size that determines the size of the surrounding area that should be considered when calculating the threshold value. The last one is a constant which will be subtracted from the mean (or weighted mean, depends on the third parameter).

Exercise

Given an image "sudoku.bmp", try to segment out the text by using adaptive thresholding. The result should be much better in this case when compared to the previous one.

C. Region Growing

Different from thresholding that is considered as pixel-based approach, region growing is a region-based approach that considers the contextual information. Users are first required to select certain locations and use them as the seeds. They are serving as the starting region for the growing process. Then, the homogeneity of the surrounding pixels of a seed is evaluated. If a surrounding pixel exhibits same characteristic as the seed (e.g. same intensity value or the difference is within a certain range), it will be added to the region. This evaluation and growing process continues until no further growing is possible for the region. When it is done for one seed (one region), it moves on to the next seed and repeat the same evaluation and growing process. The entire process will stop after all the seeds have been processed.

Exercise

Unfortunately, there is no existing region growing function from OpenCV for us to use. Hence, your task in this exercise is to develop your own region growing function. The function name as well as the number of parameters (not limited to) you need to include are described below.

regionMap = regionGrowing(inputImage, seedMap, threshold, connectivity)

Parameter	Dimension	Description
regionMap	2D	Output that illustrates how the input image is segmented. Use ‘1’, ‘2’, ‘3’, …, ‘n’ to categorize each pixel (or each region).
inputImage	2D	The input image that the user would like to process. A standard 8-bit grayscale image.
seedMap	2D	An array that shows the location of all the seeds. Any location with non-zero value is considered as a seed. You may decide how you want the user to determine the locations. The user will have to follow the “rule” that you have set. For example, users put ‘1’ at location that considered as seed and you will determine which seed to grow first (Option 1). Or the users will have to determine themselves the sequence of growing with ‘1’, ‘2’, ‘3’, … ‘n’ (Option 2). You can choose either one.
threshold	Scalar	The threshold used to perform the homogeneity test. It should be an integer number.
connectivity	Scalar	To choose between 4-connectivity and 8-connectivity. Use ‘4’ to select the former and ‘8’ for the latter.

Given below is an example of the output (regionMap) that you should obtained, based on the set of sample inputs (inputImage, seedMap, threshold, and connectivity). You can use this to evaluate the function that you have developed (you should run a few more evaluations).

<table>
<tr><th>Parameter</th><th>Value</th></tr>
<tr><td>inputImage</td><td>
<table>
<tr><td>5</td><td>5</td><td>5</td><td>2</td><td>1</td><td>1</td><td>1</td><td>7</td></tr>
<tr><td>2</td><td>5</td><td>5</td><td>2</td><td>2</td><td>1</td><td>1</td><td>6</td></tr>
<tr><td>2</td><td>5</td><td>4</td><td>4</td><td>4</td><td>6</td><td>6</td><td>6</td></tr>
<tr><td>2</td><td>2</td><td>4</td><td>4</td><td>4</td><td>6</td><td>6</td><td>6</td></tr>
<tr><td>3</td><td>2</td><td>2</td><td>4</td><td>7</td><td>7</td><td>7</td><td>7</td></tr>
<tr><td>3</td><td>3</td><td>2</td><td>4</td><td>1</td><td>1</td><td>1</td><td>1</td></tr>
<tr><td>3</td><td>3</td><td>2</td><td>2</td><td>4</td><td>2</td><td>2</td><td>1</td></tr>
<tr><td>3</td><td>3</td><td>2</td><td>2</td><td>4</td><td>2</td><td>2</td><td>2</td></tr>
</table>
</td></tr>
<tr><td>seedMap</td><td>
Option 1:
<table>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>
OR
(Continue Next Page)
</td></tr>
</table>

<table>
<tr><th>Parameter</th><th>Value</th></tr>
<tr><td>seedMap</td><td>Option 2:
<table>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>2</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>3</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>4</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
<tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td></tr>
</table>
</td></tr>
<tr><td>threshold</td><td>1</td></tr>
<tr><td>connectivity</td><td>8</td></tr>
<tr><td>regionMap</td><td>
<table>
<tr><td>2</td><td>2</td><td>2</td><td>1</td><td>1</td><td>1</td><td>1</td><td>3</td></tr>
<tr><td>4</td><td>2</td><td>2</td><td>1</td><td>1</td><td>1</td><td>1</td><td>3</td></tr>
<tr><td>4</td><td>2</td><td>2</td><td>2</td><td>2</td><td>3</td><td>3</td><td>3</td></tr>
<tr><td>4</td><td>4</td><td>2</td><td>2</td><td>2</td><td>3</td><td>3</td><td>3</td></tr>
<tr><td>4</td><td>4</td><td>4</td><td>2</td><td>3</td><td>3</td><td>3</td><td>3</td></tr>
<tr><td>4</td><td>4</td><td>4</td><td>2</td><td>4</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>4</td><td>4</td><td>4</td><td>4</td><td>2</td><td>4</td><td>4</td><td>4</td></tr>
<tr><td>4</td><td>4</td><td>4</td><td>4</td><td>2</td><td>4</td><td>4</td><td>4</td></tr>
</table>
</td></tr>
</table>