CENG466 Digital Image Processing Take Home Exam 3 Solutions

Yekta Demirci
Electrical and Electronics Department
Middle East Technical University
Ankara, Turkey
e209360@metu.edu.tr

Zumrud Shukurlu
Computer Engineering Department
Middle East Technical University
Ankara, Turkey
e2174761@ceng.metu.edu.tr

***Abstract*—In this document, the solutions proposed for Take Home Exam 3 are explained and the results are discussed.**

***Index Terms*—mathematical morphology, object counting, segmentation, k-means**

I. Introduction

The steps followed to do Take Home Exam 3 are explained clearly. The results are shown and discussed at the end of each section. All the codes are implemented in MATLAB language. In the first part, morphological operations are used to count the objects in the scene.

II. Part 1: Object Counting

The task in this section is to extract the balloons from the scene and count them with **morphological operations**. To solve the problem, we implemented morphological operations in MATLAB and used different combinations of them for each image to retrieve the balloons. We implemented **`erosion()`** and **`dilation()`** and used those two functions to write the rest of the morphological operations (`opening()`, `closing()`, `smoothing()`, `bottomhat()`).

- `erosion()`: Just like in the convolution operation, we take the subsection of the image that is under the given mask. Then we take the elements that have matching indices with the 1s in the mask and take the minimum of them to calculate the output pixel. We shift the structuring element over the whole image and do the operations as explained to calculate the whole output image. We adopted this approach from [2]. We also pad the input image with its border values accordingly to obtain an output of the same size as the input image. The method works both for grey-scale images and for binary images. You can see the formal definition of our approach below. Here $f(x, y)$ is the input image and $b(x, y)$ is the structuring element.

$$(f \ominus b)(x, y) = min\{f(x + x', y + y')|(x', y') \in D_b\}$$

- `dilation()`: The function was written using the same approach with erosion, but taking the maximum of the chosen elements in the subsection of the input image for each output pixel:

$$(f \oplus b)(x, y) = max\{f(x - x', y - y')|(x', y') \in D_b\}$$

- All morphological operations takes a grey-scale or binary image and a flat structuring element as input and gives the modified image as the output.
- We have also written a simple function `disk()` to create a disk shaped structuring element, which is basically a square matrix with 1s within a given radius from the center and 0s outside the radius. We use disk shaped strels for all images, because the shape of balloons resemble a disk.
- Also, we apply `imbinarize()` function of MATLAB in the end to display the balloons as a collection of white pixels.
- `bottomhat()` operation is applied on all images, because it is mainly used in pictures with a light background and dark objects. In our case, all of the given images match to this criteria.
- We determined the strel used with bottomhat to be of the same size with the balloons in the image approximately.
- To count balloons, `imlabel()` function of MATLAB was used, which finds the connected components in the given image.
- Images were converted to grey-scale by using `rgb2gray()` from MATLAB, except for A3.png, where the red component of the original image was used, because it gave a better output.

A. A1.png and A2.png

There was a high contrast between the balloons and the background in these images, so it was trivial to extract the balloons here. We used a disk shaped structuring element of radius 20 (`disk(20)`) and applied `bottomhat()` operation. All balloons were extracted in this step, but in the first image, the rightmost balloon had few pixels left after the operation. Therefore to improve the visibility, `dilation()` function with `disk(4)` was applied.

In the case of A2.png, the image was opened with `disk(4)` afterwards, because the result had two disjoint connected components for one of the balloons.

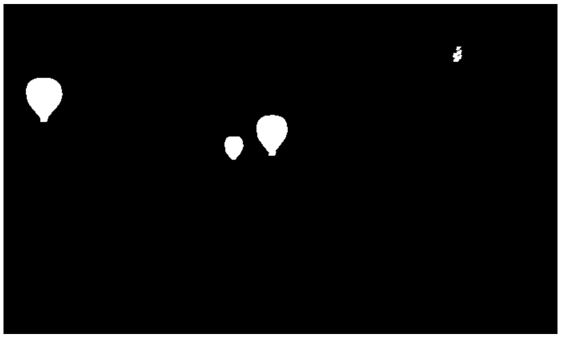

Fig. 1. part1_A1.png

B. A3.png

The textures in this image was highly varying, so the first step we took was smoothing the image with morphological smoothing. Then bottomhat was utilized next and the image was made black and white afterwards. We eliminated the ground from this image manually. Though we smoothed the image, the balloons near the ground was mixed with the ground itself and could not be extracted. So the balloons whose background was the sky were counted. After binarization, the image was eroded with `disk(2)` to remove undesired components, which were not balloons. In the end, before counting, it was dilated again with `disk(10)` to grow the size of remainders of small balloons and unify the disjoint parts of some balloons. At the rightmost edge, there is an unwanted remainder of the ground that could not be removed with morphology.

Fig. 2. part1_A3.png

C. A4.png

At the first step, `bottomhat()` function with `disk(40)` was used here. After binarizing the image, it was opened with `disk(5)`, because there were slight disconnections between pixels of one balloon. the ground was removed from the image manually after binarizing, because trying to remove it with morphology also removed all other objects in the image.

D. A5.png

Before extracting balloons, we applied morphological smoothing in this image, to homogenize the background since the white clouds in the image disrupted the output. Then bottom-hat operation with `disk(20)` was employed. After binarizing the image, opening was used to join the disconnected sets of individual balloons. In the end, the image was dilated with a small strel to point out balloons that were too small to see.

III. Part 2: Segmentation

In this part 3 images were given. River, forest and sky parts were required to be segmented. For this purpose, **k-means clustering** and **morphological operations** are used.

- We used k-means clustering as the `segmentation_function()` and mathematical morphology was applied to each result to refine the clusters.
- We separated all images to two clusters with k-means, converted the two-label image to a binary image to perform morphology.
- Though there were three parts (sky, ground, water) in the first image, partitioning the image to three clusters mapped the sky and water to the same label, so it was ineffective. Two clusters generated the same result without assigning two different labels to the ground unnecessarily.

Fig. 3. Result of k-means clustering on B1.png

- In k-means, after initialization of centroids, Euclidean distances are calculated between each centroid and the nearest data to them. Then the mean of these distances are taken to update the centroid locations. This process goes on until the mean of distances change. By this way data is clustered according to where they accumulate the most. As a result 2 variables are returned. One is showing the cluster information of each data, and the other one is showing converged centroid locations.
- In the morphology part, all images were opened with small disk shaped structuring elements (strel) whose size were proportional to undesired parts in the image. Opening erodes those parts and recovers the remaining parts to their original size bu dilating it again.

Fig. 4. B2.png after kmeans and opening

- To create borders, we applied morphological gradient, which is the difference of the dilated and eroded images.

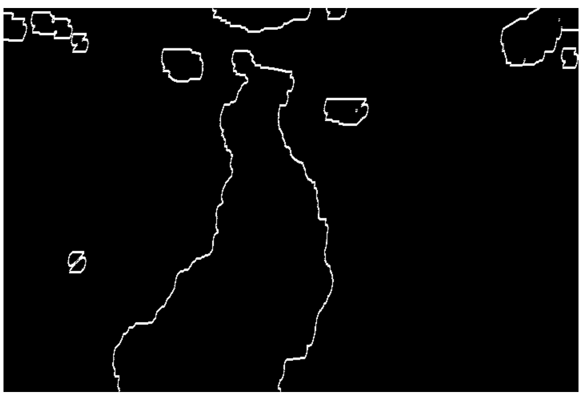

Fig. 5. Gradient image of B3.png after kmeans and morphology

- Opening the image with bigger strels to remove all redundant parts distorted the shape of the borders, so they were tolerated.
- Then we mapped all borders onto original images to to make visuals more clear.

Fig. 6. the3_B3_output.png

REFERENCES

[1] Gonzalez, R. C., & Woods, R. E. (2002). Digital image processing. Upper Saddle River, N.J: Prentice Hall. Chicago.

[2] Gonzalez, R. C., Woods, R. E., & Eddins, S. L. (2009). Digital Image processing using MATLAB. (pp. 521-524). United States: Gatesmark Publishing.