COMP 448/548 – Medical Image Analysis
Homework #1

Due: 23:59, April 24, 2023

In this homework, you will design an algorithm for detecting and segmenting cells in live cell images of the CAMA-1 cell line. You will work on three images that are provided in this assignment. For each image, there are three files:

- *im*.jpg:* RGB image file
- *im*_gold_cells.txt:* Text file containing cell label annotations. In this file, cells are labeled from 1 to N, where N is the number of cells. All pixels of the same cell are labeled with the same id. Background pixels are labeled with 0.
- *im*_gold_mask.txt:* Text file containing a rough mask for foreground (cell) pixels. In this file, foreground pixels are labeled with 1 and background pixels with 0.

Below is an example for such an image, its annotated cells, and its mask. Note that annotated pixels of the same cell are represented with a different color (this coloring is just for illustration, selection of these colors is random). Also note that these annotations are not perfect (remember our discussion about the difficulties about annotating images) and may have some inconsistencies. Please use annotations as they are and do not try to correct them.

Your algorithm should have the following three steps, each of which you will implement and evaluate:

1. **ObtainForegroundMask**: It inputs an RGB image and outputs a map of foreground pixels where foreground and background pixels should be labeled with 1 and 0, respectively.
2. **FindCellLocations:** It inputs the RGB image and the foreground map, which you will have calculated in the first step, and outputs a set of points (x and y coordinates). Each point will correspond to an approximate location of a cell.
3. **FindCellBoundaries:** It inputs the RGB image as well as the foreground map and the set of points, which you will have calculated in the previous steps, and outputs a segmentation map of cells. In this segmentation map, cells should be labeled from 1 to N. Pixels of the same cell should be labeled with the same id. Background pixels should be labeled with 0.

Below are the details of requirements you should comply with for implementing each step. As detailed below, you will design an algorithm for each step. In your design, you may use any techniques we have discussed in class. These techniques may exactly be the same with those that are given on the course slides or may be related to them. However, you cannot use any techniques of deep learning. It is important to note that this homework is for you to make a simple design of your own. Thus, although your design should be technically sound, there is no single "best" answer/design for the homework.

In your implementation, you may use any programming language. You may also use the built-in library functions for these techniques (e.g., for filters, morphology operators, etc.). Obviously, if there is a built-in function for the entire step (e.g., a function for finding the centers when an image and a mask are given), **you are NOT allowed to use it**.

What to submit: Submit the following via Blackboard.

- Your source codes. Put all of your source codes in a zip file called ***ID_HW1.zip***
- A maximum of 8-pages report (follow the instructions separately given for each part). You are expected to write your report neatly and properly. The format, structure, and writing style of your report as well as the quality of the tables and figures will be a part of your grade. Use reasonable font sizes, spacing, margin sizes, etc. You may submit either a one-column or a double-column document. The filename of your report should be ***ID_HW1.pdf***

PART 1: ObtainForegroundMask

This step (algorithm) takes an RGB image as an input. You may use the original RGB image or its grayscale equivalent. You should make your own design, in which you may use thresholding, clustering, filters, texture analysis, etc. I strongly recommend you to consider preprocessing the image and postprocessing the result. You may use filters and/or morphological operators for postprocessing.

After completing its implementation, run this algorithm on the three images that are provided. For each image, compare your estimated foreground mask with the gold standard (stored in a file called *im*_gold_mask.txt*). Calculate the pixel-level precision, recall, and F-score metrics.

What to submit:

- Source codes of your implementation.
- Report (a maximum of 2 pages for this part): Pseudocode of your design, list of its parameters (if any), a brief discussion about how you select the values of these parameters, visual results (estimated mask) for each of the three images, a table of the quantitative metrics (one row for each image).

PART 2: FindCellLocations

This step (algorithm) takes the RGB image and the estimated foreground map, which is the output of Part 1, as inputs, and outputs a set of points (x and y coordinates), each of which corresponds to an approximate location of a cell. Your design should be based on the use of distance transforms.

However, if you compare the masks (the ones stored in files called *im*_gold_mask.txt*) that are given in this assignment with those that are used in the course slides, you may realize that it would not be possible to identify the regional maxima of the outer distance transform as cell locations. Thus, you should find "a way" to define a more effective distance transform. In this part, you are asked to make such a definition. *Hint: Check the original RGB images. There are very obvious white boundaries in between adjacent cells. In one design possibility, you may identify these boundaries (fully or partially) and may calculate the distance from each foreground pixel to its closest white boundary.*

After defining your distance transform, calculate it for each image and identify the regional maxima of the noise-suppressed distance transform as cells. If necessary, postprocess the resulting regional

maxima map to obtain better cell locations. As the requirement of this part, you should decide on what techniques your design will use for distance transform calculation and postprocessing.

After completing its implementation, run this algorithm on the three images that are provided. For each image, calculate the cell-level precision, recall, and F-score metrics using the gold standard (stored in a file called *im*_gold_cells.txt*). In this calculation, find the number of true positives (TP) as follows: First calculate the centroid pixel for each regional maximum and match this centroid with the gold standard cell that it belongs to. (If the centroid is a background pixel in the gold standard, it means that this centroid corresponds to a false positive.) Afterwards, count the gold standard cells with which exactly one centroid matches. This gives you the number of TPs.

What to submit:

- Source codes of your implementation.
- Report (a maximum of 2 pages for this part): Pseudocode of your design, list of its parameters (if any), a brief discussion about how you select the values of these parameters, visual results (regional maxima map) for each of the three images, a table of the quantitative metrics (one row for each image).

PART 3: FindCellBoundaries

This step (algorithm) takes the RGB image, the estimated foreground map, which is the output of Part 1, and the estimated set of cell locations, which is the output of Part 2, as inputs, and estimates a segmentation map of cells. In this segmentation map, cells should be labeled from 1 to N. Pixels of the same cell should be labeled with the same id. Background pixels should be labeled with 0. Your design should be based on the use of a region growing algorithm.

In the design of the region growing algorithm, you should use the estimated set of cell locations as the initial seeds. However, as the requirement of this part, you should decide the marking function and the stopping condition of this region growing algorithm.

After completing its implementation, run this algorithm on the three images that are provided. For each image, calculate the cell-level Dice index and the intersection-over-union metrics for the thresholds of 0.5, 0.75, and 0.9, using the gold standard (stored in a file called *im*_gold_cells.txt*).

Here it is important to note that, in the metric calculations, you should match the estimated cells with the gold standard cells with respect to their overlapping pixels (as explained in Slides 23-26 of the sixth slide set). You should not directly compare the ids since the estimated ids might be different that the ids given in the gold standard. For example, pixels of an estimated cell with id=5 may 95 percent overlap with those of a gold standard cell with id=35. This is a TP cell even though the ids are different. You should make your calculations accordingly.

What to submit:

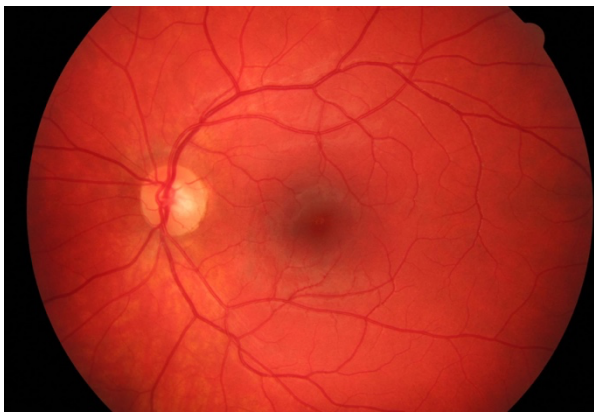
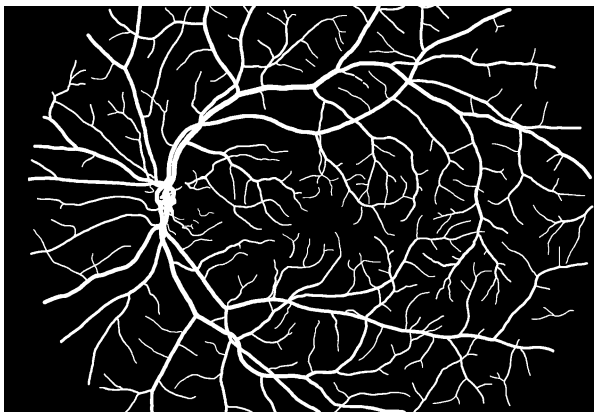
- Source codes of your implementation.
- Report (a maximum of 2 pages for this part): Pseudocode of your design, list of its parameters (if any), a brief discussion about how you select the values of these parameters, visual results (segmentation map) for each of the three images, a table of the quantitative metrics (one row for each image). To visualize different cells in the segmentation map, assign colors to segmentation ids.

PART 4: Segmentation of blood vessels in fundus photography images

This is an additional part in which you will work on another type of medical images. The problem is to segment blood vessels in fundus photography images. One example image together with the blood vessel annotations are given below. For this part, you are free to make your own design. However, you cannot use any techniques of deep learning.

After completing the implementation of your own design, run this algorithm on the three images that are provided. For each image, compare your estimated vessel mask with the gold standard (stored in a file called **_gold.png*). Calculate the pixel-level precision, recall, and F-score metrics.

Note that in the **_gold.png* files, foreground and background pixels are stored as 1 (not 255) and 0. Thus, when you open it in an image editor, you will see almost a black image, not the one illustrated below. To see the blood vessels, you need to visualize gold == 1.

What to submit:

- Source codes of your implementation.
- Report (a maximum of 2 pages for this part): Pseudocode of your design, list of its parameters (if any), a brief discussion about how you select the values of these parameters, visual results (segmentation map) for each of the three images, a table of the quantitative metrics (one row for each image).