AE2IIP Coursework 2018-2019

This is the assessed coursework for IIP. It is worth **40%** of the final mark.

Deadline: 3 May 2019 16:00

Task Description and Guidelines

Task 1: Implement a face detector using HSV color space or YUV color space. (**10 marks**)

Requirements & Guidelines

- This task is to implement a face detector using HSV color space or YUV color space.
- You should learn the clusters of face pixels based on some training images. **Hints:** You may find the face images from internet resources, crop the face region, and find the clusters of face pixels in either H-S space or U-V space. (**3 marks**)
- Then you should test the face detector on the given test images: **face1.jpg** and **face2.jpg**, to obtain the binarized image, where the face is foreground and other pixels are background. **Hints:** Basically, the images pixels fail into the face cluster that your find in the previous step will be considered as the face pixels and the rest will be considered as the background. (**3 marks**)
- The binarized image may have false face pixels or miss some face pixels. Apply the morphological operation on the binarized image, to remove noise. **Hints**: The following operations may be applied. (**4 marks**)
  - ❖ Morphological closing or morphological opening.
  - ❖ Connected component analysis.
  - ❖ Other methods that could remove noise in binary images.
- You should implement the three major steps in functions, i.e. one in each function. For example:
  - ❖ *faceModel = getFaceModel()*;
  - ❖ *binarizedFaceImage = getBinarizedFaceImage(testImage, faceModel)*;

testImage is a color test image and *binarizedFaceImage* is a binarized image where white pixels denote the face and black pixels denote background.

- ❖ *FinalFaceImage = getMorphFace(binarizedFaceImage);*

FinalFaceImage is a binarized image where white pixels denote the face and black pixels denote background.

- You should implement one main function that calls other functions to generate all the results, e.g. task1main().
- Report the following in your report:
  - ❖ The face-pixel cluster(s) you find in H-S space or U-V space, e.g. [H_start, H_end, S_start, S_end]. You may have more than one cluster.
  - ❖ The binarized images before morphological operations.
  - ❖ Your final binarized image for face detection, where face is the foreground (white) and the rest is background.

Marking Criteria

You should include the explanation of each step of your method, together with the intermediate results of each step in your report. Marks have been allocated to each step. Marks will be given by considering:

- Whether you clearly describe your method;
- Whether you well justify your algorithm;
- Whether you successfully obtain the desired output.
- The quality of face detection, e.g. minimize false detection, and try to detect all face pixels.
- Whether your codes are well documented, including comments, and explanation of variables at the beginning of each function.

Task 2: Implement and compare spatial filters. (**14 marks**)

Requirements & Guidelines

- You should implement the following spatial filters: (**5 marks**)
  - ❖ 3×3 mean filter.
  - ❖ 5×5 Gaussian filter with σ=1.
  - ❖ 3×3 median filter.
  - ❖ 3×3 anisotropic filter with the similarity function of *(D-d)/D*.
  - ❖ 5×5 bilateral filter with with σ=1 for both Gaussian functions.
- Create the following distorted images: (**3 marks**)
  - ❖ Add Gaussian noise with σ=20 on RGB channels of lena.jpg.
  - ❖ Add 10% of salt & pepper noise on RGB channels of lena.jpg.
  - ❖ Convolute each of RGB channels of lena.jpg with a 7×7 Gaussian filter with σ=2.
- Apply five implemented filters on the distorted images you created above. Compare the resulting images. Discuss which filter(s) are more effective to each of the distorted images. (**6 marks**)
- You should implement one main function that calls other functions to generate all the results, e.g. task2main().

Marking Criteria

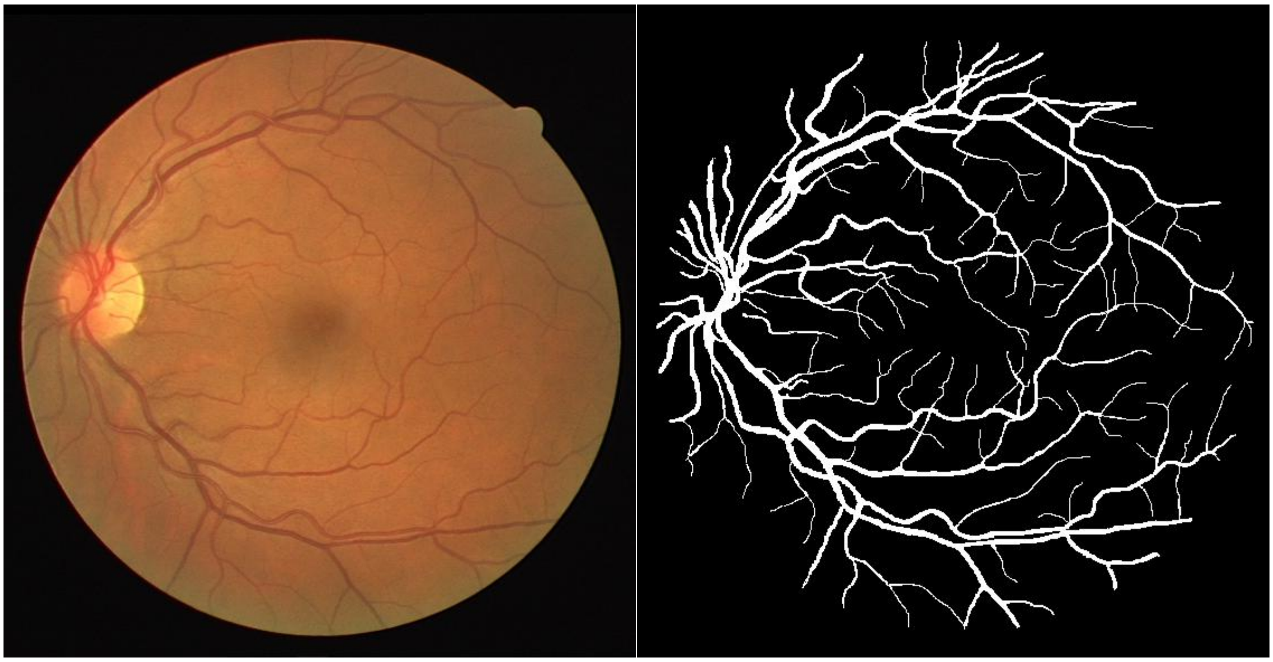
You should include the explanation of each filter you implement, the distorted images and the filtered images in your report. Marks have been allocated to each step. Marks will be given by considering:

- Whether you correctly implement the filter.
- Whether you correctly implement the distorted image.
- Whether you successfully obtain the filtered images.
- Whether your arguments for the choice of filters are reasonable.
- Whether your codes are well documented, including comments, and explanation of variables at the beginning of each function.

Task 3: Segment the retina blood vessel. (**16 marks**)

Retina blood vessel segmentation is challenging. Given the image on the left, it is very difficult to accurately segment the blood vessel as shown on the right, especially the thin ones. You are free to design your own algorithm to segment the blood vessel, down to per pixel level.

NOTE: You are not required to achieve the state-of-the-art results, but you should achieve satisfactory results.



Requirements & Guidelines

- 40 retina images are provided, together with the label images indicating where the blood vessel is, and the mask image masking out the background of retina image.
- You should **test your algorithm on retina_images/1.tif only**, and show the segmentation results for this image. You should never use label_images/1.tif when you design your algorithm. All other images being provided can be used as training images when you design your algorithm.
- When test your algorithm, you should compare the segmentation results of your algorithm with the ground-truth image (label_images/1.tif). You should report the following:

- ❖ The percentage of blood vessel pixels that is being correctly classified as blood vessel. Denoted as P.
- ❖ The percentage of background pixels (only consider the region in the mask.) that is being correctly classified as background. Denoted as N.
- ❖ The percentages of pixels are being correctly classified. (only consider the region in the mask). Denoted as T.
- ❖ For example, assume in the mask we have 100 pixels, 10 pixels are retina and 90 pixels are background. The segmentation result is that 12 pixels are retina (only 8 are true retina and 4 are false retina) and 88 pixels are background. (only 86 are true background and 2 are false background.) Then P = 8/10 = 80%; N = 86/90 = 95.56%; T = (8+86)/100=94%.

- ● You may consider the following filters: (It is up to you to choose which one to use and how many you may use. You may use other filters if appropriate. **You may use FILTERS from internet resources, with proper acknowledgement**.
  - ❖ Laplacian of Gaussian filter.
  - ❖ Difference of Gaussian filter.
  - ❖ Canny filter.
  - ❖ Match filter.
  - ❖ Gabor filters.
- ● You may directly process the filtered images to obtain the per-pixel classification results of the retina image. Or you may try to combine the results from several filters, e.g. to treat the filtered output of other images as the training samples, and train a per-pixel classifier to classify the test image retina_images/1.tif.
- ● You should implement one main function that calls other functions to generate all the results, e.g. task3main().

Marking Criteria

You should include the explanation of your algorithm, the intermediate results and the final results in your report. Marks will be given by considering:

- The quality of your per-pixel level segmentation results for the first image. You should try to maximize both P and N. (**6 marks**)
- The explanation and justification of your algorithm. (**6 marks**)
- Discuss advantages and disadvantages of your algorithm. (**2 marks**)
- Whether your codes are well documented, including comments, and explanation of variables at the beginning of each function. (**2 marks**)

Reference and Plagiarism

You can only use the build-in functions of matlab, or the codes you implement by yourself, unless otherwise stated. I will run the plagiarism detector tools to check for similarities between submissions and web-based material.

You are reminded of the School's Policy on Plagiarism.

Report

The page limit is 10 for the report. Font size 12. All images on the report should be large enough for visual inspection of the image quality.

How to submit

Online submission via Moodle. You should zip all the matlab files, the result images and the PDF file for the report in **ONE** zipped file. (For task 3, **only upload the segmentation result of the test image**.) You should name the zip file using your name and student ID, e.g. “DongChen_1234567.zip”. Please note that every next submission overwrites the files in the previous one, so if you submit several times, make sure that your last submission includes all the necessary files. Please also submit a **HARDCOPY** of your PDF file to the faculty office.