ECE661 Fall 2024: Homework 6

TA: Rahul Deshmukh (deshmuk5@purdue.edu)

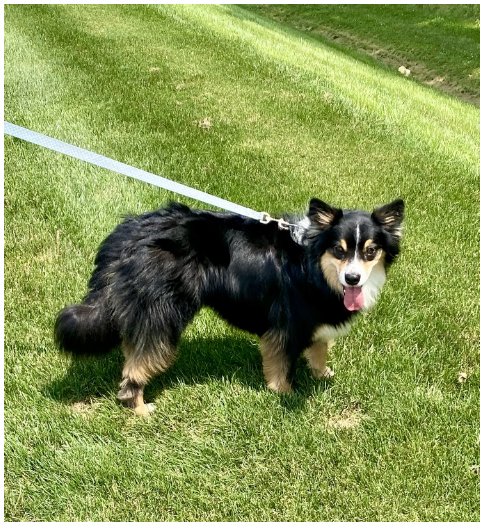
Due Date: Midnight, 16 Oct 2024

Late submissions will be accepted with penalty: -10 points per-late-day, up to 5 days.

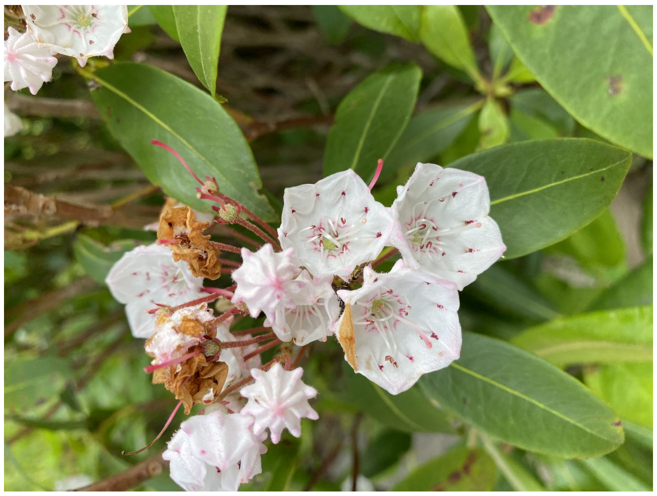
Turn in typed solutions via BrightSpace. Additional instructions can be found at BrightSpace.

1 Introduction

In this homework, you will implement image segmentation and contour extraction algorithms. Specifically, you're provided with the two input images as shown in Fig. 1. The expected results should show the separation of a foreground object (dog and flower) from the background. This homework will require per-image parameter tuning to obtain visually acceptable results, i.e., visual separation of a foreground object from the background. Note that some amount of noise is acceptable.



(a) Image 1



(b) Image 2

Figure 1: Input Images.

2 Theory Question

Lecture 15 presented two very famous algorithms for image segmentation: The Otsu Algorithm and the Watershed Algorithm. These algorithms are as different as night and day. Present in your own words the strengths and the weaknesses of each. (Note that the Watershed algorithm uses the morphological operators that we discussed in Lecture 14.)

3 Programming Tasks

3.1 Task 1

For Task 1, you will implement two variants of the Otsu algorithm as described below.

3.1.1 Image segmentation using RGB values

A common practice of using the Otsu algorithm on RGB images is that you treat each channel separately, i.e., you obtain per-channel segmentation first and then combine these results using logical 'AND' operation. As part of the parameter tuning, you should try to run the Otsu algorithm in an iterative fashion. More concretely, you use the foreground segmentation returned by the previous iteration to refine it further by running the Otsu algorithm only on the foreground portion of the image.

3.1.2 Texture-based segmentation

Here is one more strategy for using the Otsu algorithm for image (foreground and background) segmentation. You first extract some texture-based features and then use the Otsu algorithm on these features. There now exist many sophisticated approaches to obtain texture-based features. For this homework, you will implement the simplest sliding window based approach. First, convert a given RGB image to grayscale. Place a window of $N \times N$ at each pixel, subtract the mean intensity value and compute the intensity variance within each window as a texture measure at the center pixel. For border pixels, try multiple options, e.g., assume the pixel intensity values outside the image border as 0. Repeat this step for different values of N, e.g., $N = 3, 5,$ and 7. You also need to tune these parameters for the given images. Now use these three feature maps as your channels and repeat the steps described in the previous subsection. Also try the iterative Otsu algorithm to obtain better quality results.

3.1.3 Contour Extraction

Once you get the image segmentation using the Otsu algorithm, you can save it as a binary mask, where the foreground pixels are marked as 1 and the background pixels as 0. Now, you will implement your own algorithm for contour extraction using these binary masks.

3.2 Notes

1. You must use your own implementation for the Otsu algorithm and for contour extraction. You need to show the results using RGB channels as well as texture features.
2. You can use a function from OpenCV or scikit-image for converting RGB images to the grayscale images.

3.3 Task 2

Repeat the steps in Task 1 on two of your own images. For each of your own image, be sure to state the intended foreground object. Additionally, briefly comment on what is working and what is not with your segmentation and contour extraction algorithms on your own images.

4 Submission Instructions

Include a typed report explaining how did you solve the given programming tasks.

1. Turn in a zipped file, it should include (a) a typed self-contained pdf report with source code and results and (b) source code files (only .py files are accepted). Rename your .zip file as hw6_<First Name><Last Name>.zip and follow the same file naming convention for your pdf report too.

   There should be two items in your submission - (1) Homework PDF (2) ZIP file containing source code (*.py files) and text files (if any).
2. If you are submitting past deadline, **make only one late submission on BrightSpace**. Otherwise, we cannot guarantee that your latest submission will be pulled for grading.
3. Your pdf must include a description of
   - Your answer to the theoretical question in Section 2.
   - A clear description of your implementation of the Otsu algorithm using (a) RGB channels and (b) Texture features.
   - Your observations on the performance of your segmentation algorithms using RGB channels as well as texture features.
   - A clear description of your contour extraction algorithm.
   - Optimal set of parameters that produced good segmentation and contour results. For example, the number of iterations that produced good results for the iterative version of the Otsu algorithm.
   - The image segmentation and contour extraction results for the given set of images as well as your own.

- Your source code. Make sure that your source code files are adequately commented and cleaned up.

4. To help better provide feedbacks to you, make sure to **number your figures**.
5. The sample solutions from previous years are for reference only. **Your code and final report must be your own work.**

Homework File Size

For your homework submissions, please ensure that your reports (PDFs) are **under 10Mb**.

Some ways to reduce your report sizes are:

- Downsample your input/output image when including in the reports.
- When including plots in your reports, save them as PDFs instead of PNG/JPEG. PDFs are vector formats which are lightweight and do not pixelate when zooming into the plot.

Strike a balance between the file size and image quality displayed in your report. This will help your TA in easy distribution of homeworks for grading without maxing out the disk space. Moreover, this will help you when you upload your homeworks to your individual git repos. You don't want to upload 40 Mb PDF to your git repo!

Finally, do not include the images in your ZIP files for your homework submissions. Ideally, the ZIP file should be under 1Mb because it only contains ASCII (*.py /*.txt) files.