COMP9517: Computer Vision

2024 T2 Lab 4 Specification

Maximum Marks Achievable: 2.5

This lab is **worth 2.5% of the total course marks**.

The lab files should be submitted online. Instructions for submission will be posted closer to the deadline. **Deadline for submission is Week 7, Monday 8 July 2024, 18:00:00.**

Objective: This lab revisits important concepts covered in the Week 5 lectures and aims to make you familiar with implementing specific algorithms.

Software: You are required to use OpenCV 3+ with Python 3+ and submit your code as a Jupyter notebook (see coding and submission requirements below). In the tutor consultation session this week and next, you can ask any questions you may have about this lab.

Materials: The sample images to be used in this lab are available via WebCMS3.

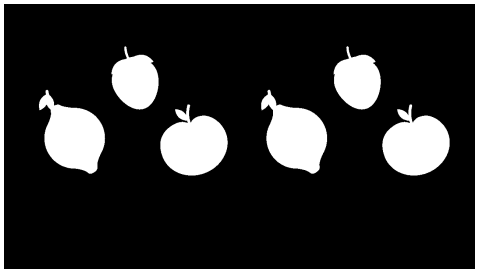
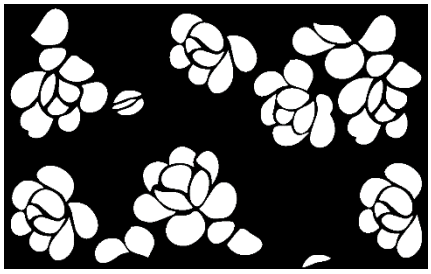
Submission: All code and requested results are assessable after the lab. Submit your source code as a Jupyter notebook (.ipynb file) that includes all outputs (see coding requirements at the end) by the above deadline. The submission link will be announced in due time.

Task (2.5 marks)

In many computer vision applications, image segmentation is a crucial first step towards quantitative image analysis. As discussed in the lectures, there are many possible image segmentation methods, ranging from very simple to very complicated.

Despite its simplicity and limitations, intensity thresholding is still a very popular image segmentation method. This is because imperfections in the segmentation results can often be filtered out by postprocessing using binary morphological operators.

The goal of this lab is to write an algorithm that performs image segmentation and subsequent analysis using just intensity thresholding and binary morphological operators. More specifically, the algorithm should be able to take any of the given lab images, produce a segmentation result containing only the bigger objects that are completely contained in the image, and print the number of objects (see the example results below).

Fruits	Leaves	Pears
		
		
n = 6	**n = 72 (light blue only)**	**n = 16**

Hints: Convert the input image to grayscale, apply intensity thresholding (using either a single threshold or multiple to segment out intensities within a range), use binary morphology operators to fill any holes and filter out noise pixels in the segmentation, use binary reconstruction and subtraction to get rid of segmented objects touching the image boundary, set the remaining objects that are smaller than some size threshold to black, and then count and print the final number of objects.

The above example results were produced by a single algorithm. The only differences in execution for each image were the parameter values of some steps of the algorithm. Clearly define your algorithm's parameters up front (see below).

Coding Requirements

Make sure that in your Jupyter notebook, the input images are readable from the location specified as an argument, and all output images and other requested results are displayed in the notebook environment. All cells in your notebook should have been executed so that the tutor/marker does not have to execute the notebook again to see the results.

If your algorithm has any user parameters, their values must be specified at the beginning of your code, so that users can easily change and experiment with these parameters.

Released: 24 June 2024