EECS 504 Computer Vision: HW4

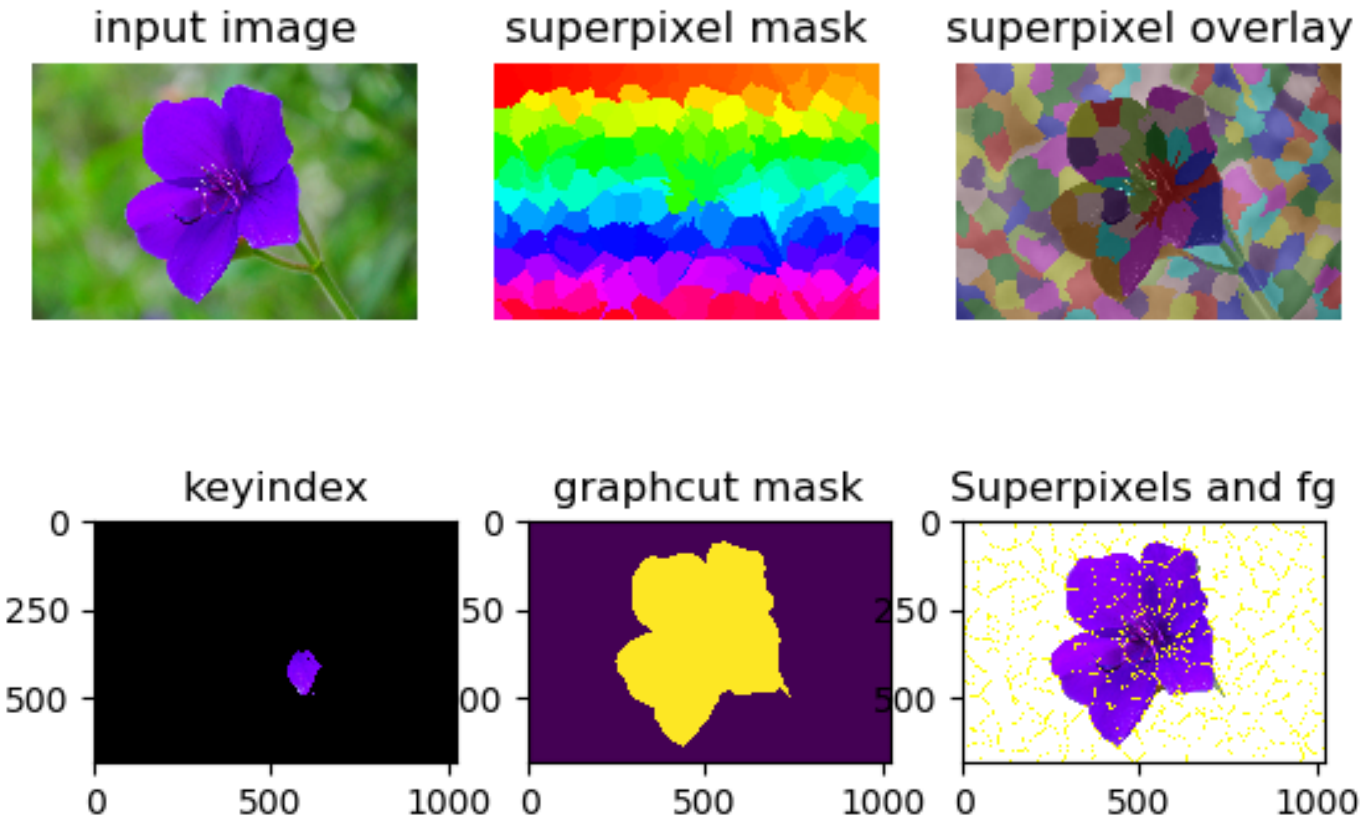
Term: Fall 2018
Instructor: Jason J. Corso, EECS, University of Michigan
Due Date: 11/16 23:59 Eastern Time

Constraints: This assignment may be discussed with other students in the course but must be written independently. Programming assignments should be written in Python using the open-source library ETA. Over-the-shoulder Python coding is strictly prohibited. **Web/Google-searching for background material is permitted. However, everything you need to solve these problems is presented in the course notes and background materials, which have been provided already.**

Data: You need to download `hw4.zip` to complete this assignment. All paths in the assignment assume the data is off the local directory. You need to run this homework inside the ETA environment you have created in Homework 0.

Problem 1 (55): Foreground-background graph-cut

We discussed the figure-ground graph-cut case study in class. You will implement this method for figure-ground extraction on color images using a superpixel graph and color histograms as the feature. A basic MRF is implemented in the graph. The figure below shows the expected results and facets of the assignment.



The provided code includes the core framework, an implementation of the SLIC superpixel method, and an implementation of the Ford-Fulkerson algorithm for computing the max-flow. You will implement the histogram feature representation and all aspects of taking the superpixel result and implementing the graph on top of it as well as reading out the results of the graph cut as a two-class segmentation.

The function `example()` provides a full run through the whole method for you. You have to submit the results from running this function. You will also need to run few additional methods.

Note that the output numpy files for each step are provided. These are for your help and benefit, and it is used to compute accuracy for submission. You will not report results from the numpy files as your own

1.(8) Color Histogram Features

Implement the missing body of function `histvec()`, which creates a color histogram feature vector. Follow the comments in the code for the details. We call this in functions `img_reduce()` and `q1()`.

Run function `q1()`, which will generate two output images `q1_Superpixels.png` and `q1_result.png`. Include these outputs in the writeup.

Also, inspect the code in function `q1()` and answer the following question in one sentence:
Describe the object in the image that is covered by the superpixel used to compute the histograms in `q1()`.

2.(15) Superpixel adjacency

(a) (7) Implement the missing body of function `seg_neighbor()`, which computes the adjacency matrix for the superpixel graph. Follow the comments in the code for the details. We call this in the graph cuts function.

Run function `q2()`, which will generate two output images `q2_Superpixels.png` and `q2_result.png`. Include these outputs in your writeup.

(b) (5) Implement a small function to compute the average node degree. Report the average degree and include the code as plain text in the writeup. (No need to submit the original file.)

(c) (3) Why is the adjacency graph not a perfect banded diagonal matrix?

3.(17) Graph-cuts

(a) (8) Implement the missing two bodies of function `graphcut()`, which :- (1) creates the graph by defining the capacity matrix and (2) extract the results after running the max-flow/min-cut method. See the comments and refer to class notes for details.

Run `q3()`, which will generate two output images `q3_Superpixels.png` and `q3_result.png`. Include these outputs in your write-up.

(b) (5) In a few sentences, relate the adjacency matrix to the capacity image. Be sure to cover **all** nodes of the graph in your description.

(c) (4) Please explain why the capacity between adjacent nodes in the graph that have resulted from superpixels is downweighted with respect to the capacity between nodes in the graph and the special source and sink nodes.

4.(15) Graph-cuts Study

Use the function `example()` here and change accordingly (for each input image you will have two output images `example_Superpixels.png` and `example_result.png`). You can change other parts of the code too if needed, but be clear to note it.

(a) (2) Run function `example()` as provided to run through a full example and show the resulting segmentation on the flower. Include the result in your write-up.

(b) (5) Run `example()` using the `flag1.jpg` image. Manually select a stripe on the flag. Are you able to get a full segmentation of the stripe and no other regions? If so, explain what you did to make this possible. If not, explain why this is hard. Include an rendering of the figure to substantiate your explanation either way.

(c) (5) Run function `example()` using the `porch1.png` image. Are you able to segment the boots perfectly? If so, explain what you did to make this possible. If not, explain why this is hard. Include an rendering of the figure to substantiate your explanation either way.

(d) (3) On the `porch1.png` image again, are you able to segment either of the baskets perfectly? My guess is no. Can you describe (but do not implement) a way to change this system to make this more possible?

Submission Process: Submit a single typewritten pdf with your answers to these problems, including all plots and discussion. Submit the pdf to Gradescope.

For coding assignments, include your code verbatim in your writeup for each question. Pack the original program files into one zip file and upload it to Canvas. **Code should be well-commented for grading**

Grading and Evaluation: The credit for each problem in this set is given in parentheses at the stated question (sub-question fraction of points is also given at the sub-questions). Partial credit will be given for both paper and python questions. For python questions, if the code does not run, then limited or no credit will be given.