Homework 6
Computer Vision, Spring 2021
Due Date: April 12, 2021
Total Points: 10

This homework contains one programming challenge. All submissions are due at midnight on **April 12, 2021**. The challenge should be submitted according to the instructions in the document "**Guidelines for Programming Assignments.pdf**".

runHw6.py will be your main interface for executing and testing your code. Parameters for the different programs or unit tests can also be set in that file.

Before submission, make sure you can run all your programs with the command `python runHw6.py` with no errors.

The numpy package is optimized for operations involving matrices and vectors. Avoid using loops (e.g., for, while) whenever possible—looping can result in long running code. Instead, you should "vectorize" loops to optimize your code for performance. In many cases, vectorization also results in more compact code (fewer lines to write!).

Challenge 1: Your task is to develop an application that allows a user to refocus a scene, much similar to the two web applications shown here. We will call it a refocusing app.

In a conventional image, the focus of a scene is fixed at the time of capture. How can a refocusing app refocus a scene? The trick is that, instead of using a single image, a refocusing app uses a focal stack—a sequence of images captured at different focus settings—as an internal representation of a scene. When a user chooses a scene point to be "refocused", the app picks the "best focused" image from the focal stack, and displays the image. How does the app determine the "best focused" image? You have learned the answer in the Depth from Focus/Defocus lecture, and now it is your turn to apply the knowledge to refocus a real-world scene.

The task is divided into two parts, each corresponding to a program you need to write and submit.

a. Write a program named `generateIndexMap()` that generates an index map from a focal stack. An index map is an image with each pixel corresponding to a scene point. The integer intensity of each pixel indicates the index of the best focused layer associated with the corresponding scene point.

   Index map generation can be carried out in two steps, and the two steps need to be performed with caution to produce a quality result. The first step is to compute a focus measure for every pixel in every image in a focal stack. We will use the modified Laplacian (described in the lecture) as the focus measure. Read the formulation of the focus measure carefully and design your program to optimize for speed. A naïve implementation with unnecessary loops will result in a long running program.

   The second step is to find the layer with the maximal focus measure for each scene point. Here we will not fit a Gaussian to the computed focus measures, as finding the precise depth is not the goal. Instead, we will simply choose the layer with the maximum focus measure as the best focused layer. Be careful, the computed focus measure could be noisy, thus you may need to smooth the data (e.g., with a moving average filter) before selecting the maximal focus measure.

   Before calling `generateIndexMap()`, write a function named `loadFocalStack()` to load a focal stack into memory and create gray-scale versions of each image. Refer to `challenge1a.py` for the specifications.
   (6 Points)

b. Write a program named `refocusApp()` that executes the following tasks in a loop: (1) display an image in the focal stack; (2) ask a user to choose a scene point (you can use the `ginput()` and `waitforbuttonpress()` functions from the `matplotlib.pyplot` package); (3) refocus to the image such that the scene point is focused. The program terminates when the user chooses a point outside of the displayed image frame. Use the index map computed in (a) to facilitate refocusing.
   (4 Points)